Super-resolution Enhancement of Video

Christopher M. Bishop, Andrew Blake
Microsoft Research
Cambridge, CB1 2HN, U.K.
{*cmbishop,ablake*}*@microsoft.com*
research.microsoft.com/{∼cmbishop,∼ablake}

Bhaskara Marthi*
University of California, Berkeley.
bhaskara@cs.berkeley.edu
www.cs.berkeley.edu/∼bhaskara/

Abstract

We consider the problem of enhancing the resolution of video through the addition of perceptually plausible high frequency information. Our approach is based on a learned data set of image patches capturing the relationship between the middle and high spatial frequency bands of natural images. By introducing an appropriate prior distribution over such patches we can ensure consistency of static image regions across successive frames of the video, and also take account of object motion. A key concept is the use of the previously enhanced frame to provide part of the training set for super-resolution enhancement of the current frame. Our results show that a marked improvement in video quality can be achieved at reasonable computational cost.

1 Introduction

The term super-resolution has been applied to a wide variety of problems ranging from blur removal by deconvolution in single images [5] through to the creation of a single high resolution image from multiple low resolution images having sub-pixel relative displacements [3, 7]. In all cases the goal is to increase the resolution (number of pixels) in the image while at the same time adding appropriate high frequency information.

Here we consider the task of spatial resolution enhancement in video sequences. Since the typical resolution of display devices is often much higher than that of video, particularly in the case of video streamed over the internet, there is considerable interest in being able to exploit the high resolution of modern display devices when rendering video.

Consider first the problem of resolution enhancement of a single still image. When an image of some given resolution is to be displayed on a device of higher resolution, it is a straightforward problem to render the pixels of the high resolution display through some form of interpolation, for example using cubic splines. Unfortunately, this adds no additional information in the high frequency ranges which the device is capable of displaying.

Some progress can be achieved by convolving the image with a filter designed to enhance the amplitude of the frequency spectrum in the higher frequency ranges, with the aim of sharpening up existing image details. Unfortunately, this also amplifies any noise in the image, and is capable of producing only minor improvements in image quality.

The addition of new high frequency information is fundamentally an ill-posed problem since there are many high resolution images which, under any given smoothing and sub-sampling process, will give rise to the same low resolution image. In the absence of additional information, the best we can hope to do is to find a high resolution image which is *perceptually plausible* so as to give the viewer the impression of viewing an image which is of higher quality than the original. We wish to apply a similar goal to the problem of video resolution enhancement.

Our approach builds on that used by [2] for the case of static images. It involves the assembly of a large database of patch pairs, in which each pair comprises

* Work done during summer internship at MSRC.

a rectangular patch from the high frequency component of a high resolution image and a corresponding patch from a smoothed and down-sampled version of that image. This dictionary can then be used, via a search procedure, to augment a new low resolution image with appropriate high frequency information. The training images used to construct the dictionary can be taken from an ensemble of images chosen, so far as possible, to be representative of the kind of images to which the algorithm might be applied.

In Section 2 we discuss the generalization of this approach to the problem of video super-resolution. We first demonstrate that a simple frame-by-frame application of the static image approach leads to unacceptable temporal artifacts. In particular, regions of the video corresponding to static parts of the scene acquire distracting scintillations. These are addressed in Section 3 through the introduction of a prior encouraging temporal continuity in the inferred high frequency information.

For parts of the scene involving moving objects there are additional artifacts, again taking the form of annoying scintillations. These can be mitigated by applying a motion based prior, as discussed in Section 4. A key concept here is that, if a given frame has been successfully super-resolved, then it can provide a transient source of training data for the solution of the inference problem in the subsequent frame. Our results demonstrate a significant improvement in the quality of the video while introducing relatively few artifacts.

2 Independent Reconstruction of Successive Frames

First consider the super-resolution enhancement of each frame of a video sequence independently, following the approach of Freeman et al. [2] for static images. It consists of a training phase, in which a set of independent training images is used to construct a dictionary of exemplars expressing the local relationship between the medium and high frequency bands of typical images, and a subsequent inference phase in which this dictionary is used to add plausible high frequency information to a new image containing only low and medium frequencies.

2.1 Training

The starting point is a set of training images of high resolution. Using bandpass filtering (discussed below) we can decompose each image into the sum of three images containing respectively high, medium and low spatial frequencies. A fundamental assumption is that the high frequency content of an image is independent of the low frequency content, given the medium frequency component. For each training image, therefore, a large number of rectangular patches are extracted from the medium frequency component together with the corresponding patches from the high frequency component. To assist with generalization the patch pairs are normalized using a local measure of the energy in the image. Details of filtering and patch construction are given in the appendix Each medium frequency patch is a vector of length $3 \times 7 \times 7 = 147$ (allowing for the three colour channels). In order to reduce this dimensionality we next perform principal component analysis on the set of medium frequency patches to reduce their dimensionality to 20. This considerably speeds up the nearest neighbour computations needed to match test patches into the dictionary.

The dictionary of medium/high resolution patch pairs is then compiled into a tree, to facilitate fast search, as Freeman et al do. We have employed a variant of their technique using a kd-tree in a coordinate system defined by PCA. We first select the component i of the 20-dimensional space along which the set of medium frequency patches has greatest variance, and split the data set in two sets using a cut at the median value m of that component. This cut is represented by a node in the graph at which we store the values of i and m. The partitioning is repeated on each of the subsets recursively until a tree of some given depth is created. Each leaf node of the tree represents the corresponding sub-set of medium frequency patches (along with their high frequency counterparts). For the experiments reported in this paper the training data comprises $N = 200,000$ patches taken from several images. We use a tree of depth 10, which has 1,024 leaf nodes each therefore representing roughly 200 patch pairs.

During the construction of the tree we also build up an index into the training images, and the location within those images, from which each patch pair arose, as

well as the corresponding inverse index. We shall denote the medium frequency patches from the training set by the vectors $\mathbf{x}_k$ where $k = 1, \ldots, M$ and M is the total number of patch pairs in the dictionary. The corresponding high frequency patches will be denoted $\mathbf{y}_k$. Also $\mathbf{z}_k = (\mathbf{x}_k, \mathbf{y}_k)$ will denote the k^{th} patch pair, and the complete training dictionary will be denoted $\mathcal{D}_0 \equiv \{\mathbf{z}_k\}$.

2.2 Inference

Inference proceeds first by extracting the medium frequency component image from the test image. Then, for each patch from this intermediate frequency image, the closest matching medium frequency patch from the training dictionary is found; finally the corresponding high frequency patch from the dictionary is used to add high frequency details to the original test patch. The criterion for closest match is based on L_2 norm, and modified by a prior encouraging consistency of neighbouring patches.

Consider a patch $\mathbf{x}$ from the medium frequency component of the test image. The cost function which determines the optimally matching patch pair from the training dictionary is given by

$$\mathcal{L}_k(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_k\|^2. \qquad (1)$$

Although this cost function finds plausible high frequency patches at each location in the test image, it does not ensure spatial consistency between adjacent patches. To achieve this we consider high frequency (5×5) patches on a square, 4-pixel grid , giving overlap regions of width 1 pixel. Then an added penalty term measures the mismatch of $\mathbf{y}$ to its neighbours in the synthesized high frequency image, giving a total cost function:

$$\mathcal{E}_k^{(\alpha)}(\mathbf{x}) = \mathcal{L}_k + \alpha \mathcal{V}(\mathbf{y}_k, \{\mathbf{y}_{k'} : k' \in \mathcal{N}_{\mathbf{x}}\}) \qquad (2)$$

where $\mathcal{N}_{\mathbf{x}}$ denotes the set of patch labels corresponding to patches which are neighbours of patch $\mathbf{x}$ in the synthesized high frequency image, and $\mathcal{V}(\cdot, \cdot)$ measures the sum-squared L_2-norm difference of (R,G,B) values in the overlap regions. The parameter α determines the relative importance of the spatial consistency term compared to the medium frequency matching term and is set using the prescription of [2] which gives $\alpha = 0.6$. The total cost function is the sum of terms of the form (2) for every patch in the test image. The dictionary patch index k is selected as the one minimizing $\mathcal{E}_k$.

In principle the optimization of (2) is complex due to the spatial consistency term. This is resolved in practice by an approximate procedure, in which the high frequency super-resolution image is built up by raster scanning the test image, and choosing, at each step, a dictionary patch based only on those high frequency patches that have already been filled in. (Where high frequency patches overlap, the corresponding pixels of the super-resolution high frequency image are determined by averaging.)

In detail the inference stage comprises the following steps

- Take the test image and decompose into the sum of low, medium and high frequency images (as discussed above).
- Scan over the medium frequency image in a raster (using a grid spacing of 4×4 pixels) and at each step find the 100 patch pairs from the training set dictionary which (to a good approximation) have the smallers values (1). This is achieved using a best bin first approach discussed below.
- From this sub-dictionary of 100 candidate patch pairs, select the best matching pair by minimization of (2).
- Build up a super-resolution high frequency image by including the high frequency patch from the best matching patch pair. Where high frequency patches overlap, use the average pixel values.
- Once the high frequency image is completed add it to the low and medium components of the test image to obtain the desired super-resolution image.

2.3 Efficient search

For this algorithm to be practical it is clearly important that the search over the dictionary be performed efficiently. Freeman et al. [2] use a space slicing algorithm due to [4]. We adopt a recent approach, known to be highly efficient in other domains of application, based on the kd-tree, whose construction was

Figure 1: Examples of two frames from a test video sequence showing the original frames (left), the low plus medium frequency components which form the input to the super-resolution algorithm (middle), the output of the super-resolution algorithm (right).

described in Section 2.1, together with the technique of "best bin first" [1]. This is used to find the 100 candidate patches efficiently. For each new test patch, the tree is first traversed to find the lead node in which the test patch belongs. During the traversal a priority queue is maintained which specifies, at each decision branch, the distance of the test patch from the cut boundary associated with the alternative branch. The test point is then compared exhaustively to all of the training set patches associated with that leaf node to find the current 100 best matches. Then the next closest leaf node region is determined using the priority queue and the corresponding leaf node set examined exhaustively, revising the list of 100 best candidates in the process. This repeats until the search terminates (when the worst of the 100 best candidates is closer than the nearest remaining leaf node region) or when some maximum number of leaf nodes have been examined (we choose this maximum number to be 100).

2.4 Flicker artifacts

As a starting point we take a video sequence and apply the above algorithm to each frame in the sequence independently. As can be seen from Figure 1 the super-resolution algorithm is able to add plausible high frequency details to generate images of improved perceptual quality. However, viewing the entire video sequence, it is apparent that there are serious perceptual artifacts, arising from the lack of temporal consistency in the algorithm. Even slight differences between successive frames can cause different high frequency patches to be chosen, and although the corresponding medium frequency patches may be relatively consistent, the high frequency patches might not be. The result is shimmering, or scintillation, of the sequence[1].

2.5 Measuring temporal flicker

Since the goal of our framework is the enhancement of the perceptual quality of video, the ultimate measure of performance is necessarily a subjective one. Its objective quantification would therefore involve extensive, and hence time consuming, user studies. For research purposes it is convenient to have a simple quantitative measure of performance that we can use to compare and optimize our algorithms. We use high resolution test images, available for evaluation only but hidden to the reconstruction algorithm, as ground truth against which to compare the performance of different super-resolution algorithms. Our video test-sequences are generated from these orig-

[1] It is difficult entirely to convey such results in a written paper. Therefore sample video sequences illustrating the results presented here accompany this paper, and can also viewed at http://research.microsoft.com/∼cmbishop/super-res.htm.

inal high resolution, uncompressed images, through smoothing and down-sampling.[2] However, we are not interested simply in the difference of our super-resolution video from ground truth directly, but in how well we manage to suppress the scintillation artifacts. To measure this we apply a high pass filter (accepting only frequencies greater than one quarter of frame rate) to the difference of the super-resolution video sequence and the ground truth, and is achieved using a windowed Fourier transform applied to the sum of the RGB channels. This gives a measure of the flicker in the super-resolution video, that can then be averaged either over pixels, to give a temporal trace of the average flicker, or over frames, to give an image showing the spatial distribution of average flicker. Note that is is possible for the super-resolution video to be perceptually convincing and yet differ significantly from the original high resolution source. Nevertheless in practice we have found this measure of performance to be useful in comparing algorithms and optimizing parameters.

In order to emphasize the distinction between static and moving parts of the video sequence we consider the evaluation of the flicker measure over two regions denoted by the red and green polygons in Figure 2. In Figure 3 we show the flicker traces corresponding to these regions. The red rectangle corresponds to a region of the image around the edge of the whiteboard which is almost static during the video sequence, while the green rectangle corresponds to a region near the edge of the face which changes substantially during the sequence due to motion of the head.

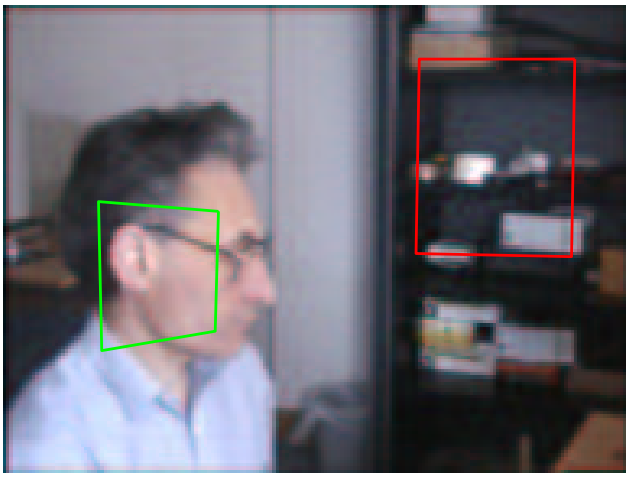

Figure 2: We evaluate the flicker measure over two regions of the video, corresponding to a region which shows static background throughout the sequence (shown in red) and one which contains substantial movement due to the motion of the head (shown in green).

[2]We took care to capture video from a high quality camera and framestore that eschews compression at every stage.

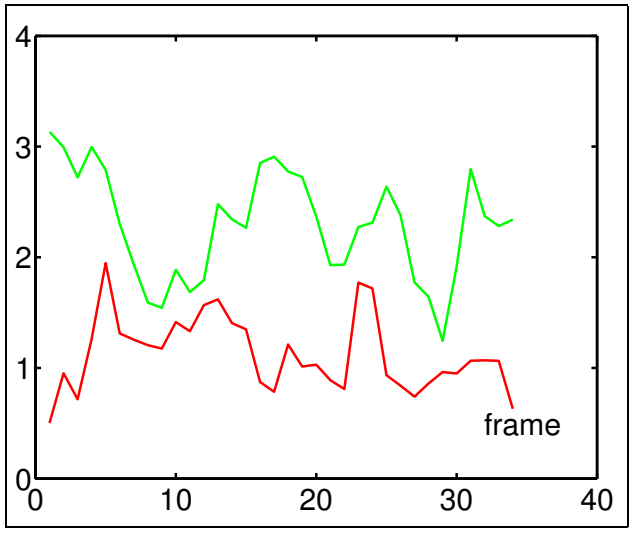

Figure 3: Traces of the flicker measure averaged over pixels in the red region of Figure 2 (red curve) and also averaged over those in the green region (shown by the green curve).

Instead of averaging the error measure over pixels we can average over frames, to give an image of the average flicker. This is shown in Figure 4. Notice that there are significant errors both in regions of the videos sequence which are relatively static, such as the area outlined by the red rectangle in the vicinity of

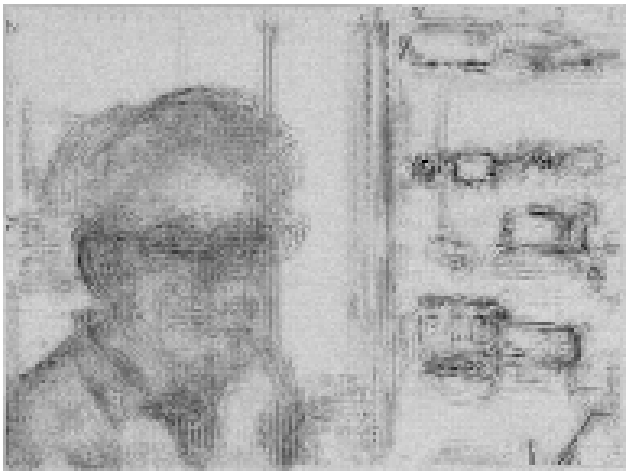

Figure 4: Image of the average absolute error between the super-resolved video and the ground truth video, averaged over all frames.

the whiteboard, and in regions which have moving objects such as the area indicated by the green rectangle around the edge of the face. We must again emphasize, however, that the goal is to produce a perceptually plausible super-resolved video and not, *per se*, to reconstruct the original ground truth.

We conclude that simple application of the static image super-resolution algorithm to each frame of a video independently gives poor results, since this does not make any attempt to enforce temporal consistency between successive frames.

3 Stasis Prior

The obvious problem with the approach described so far is that it makes no attempt to ensure consistency in the high frequency information added to adjacent frames of the video sequence. A simple way to address this problem is to modify the cost function used to select patches so as to favour the re-use, at each spatial location, of the high frequency patch used at that location in the previous frame. The extent to which this patch re-use is favoured is governed by a new parameter β, so that the cost function becomes

$$\begin{aligned} \mathcal{E}_k^{(\alpha,\beta)}(t) &= \|\mathbf{x}^{(t)} - \mathbf{x}_k\|^2 \\ &\quad +\alpha\mathcal{V}(\mathbf{y}_k, \{\mathbf{y}_{k'} : k' \in \mathcal{N}_{\mathbf{x}}^{(t)}\}) \\ &\quad -\beta I(\mathbf{y} = \mathbf{y}_k(t-1)) \end{aligned} \tag{3}$$

where $I(\cdot)$ is the binary indicator function, and we have explicitly denoted the frame label with the discrete time variable t.

In order to set a suitable value for β it is useful to find a natural scale for this parameter. We do this by evaluating β_0 given by

$$\beta_0 = \left\langle \mathcal{L}_k(\mathbf{x}^{(t-1)}) - \min_{\mathbf{x}\in\mathcal{D}_0} \mathcal{L}_k(\mathbf{x}^{(t)}) \right\rangle_{k,t} \tag{4}$$

where $\langle\cdot\rangle_{k,t}$ denotes an average over all of the patch locations and over all frames of the sequence. By empirical optimization we have found that setting $\beta = \beta_0$ gives good results.

Visually we find a substantial reduction in the level of temporal artifacts in relatively static regions of the video, but only limited improvement, if any, in regions where there is motion. This is reflected in the flicker traces obtained using this modified cost function shown in Figure 5. Compared to Figure 3 we see that there is a substantial improvement in the red curve (static region) and also noticeable improvement in the green curve (motion region). This can also be seen in the temporally averaged error images shown in Figure 6 which should be compared with those in Figure 4.

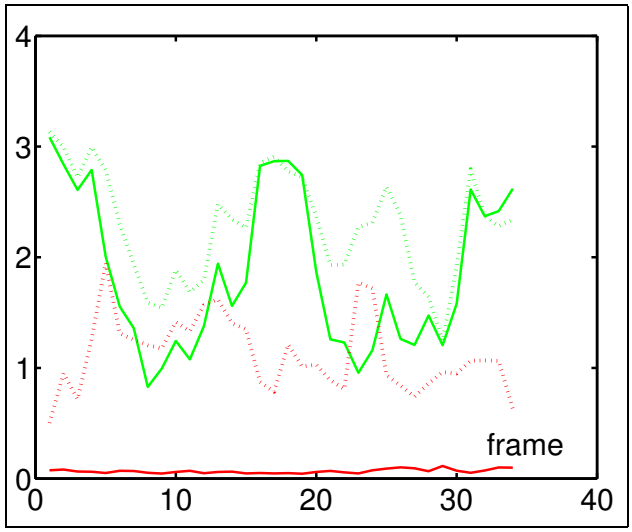

Figure 5: Flicker traces for super-resolution video obtained using the cost function (3), again with the solid red curve taken from a pixel in the red rectangle of Figure 1 and the solid green curve taken from a pixel in the green rectangle in Figure 1. For comparison, corresponding curves from figure 3 are shown dotted here.

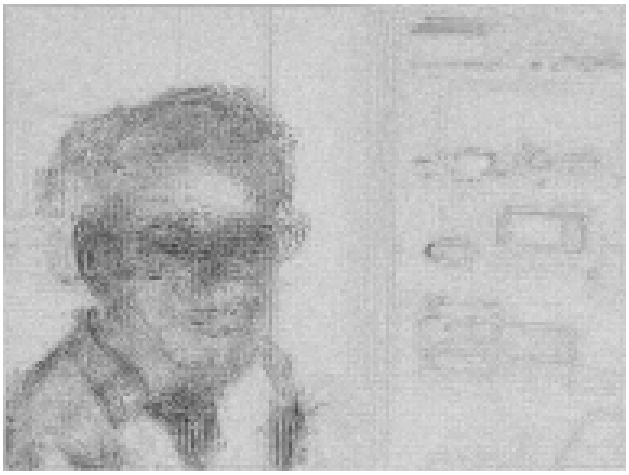

Figure 6: Average flicker image for super-resolved video obtained using the cost function (3) showing the significantly reduced level of artifacts in static regions of the video.

In order to address the problem of artifacts in regions of the video scene where there are moving objects we need a more sophisticated prior, as discussed in the next section.

4 Motion Prior

Our goal is to achieve temporal coherence between adjacent frames of the video sequence even when there are moving objects. This is difficult to achieve if objects are moving rapidly, but in such cases the perceptual impact of lack of coherence is relatively small.

The major problem arises with slowly moving objects, and we seek to address this using a prior which encourages the added high frequency components to move coherently with the object. It is not sufficient, however, simply to use patches preferentially which were used at nearby locations in the previous frame, since the patches are located on a rather course grid for reasons of computational efficiency (4 pixel spacing in the experiments reported in this paper).

Instead when we enhance a particular frame, we shall consider the previous frame, along with its super-resolved counterpart, to form a transient training set from which additional patch pairs can be extracted to augment the fixed training dictionary $\mathcal{D}_0$. Specifically, for each location in the new frame, we extract patch pairs of medium and high frequency components from the previous frame at locations within some rectangular window centred on that location. This gives rise to a temporary dictionary $\mathcal{D}_k^{(t-1)}$. For the results reported here we consider a window of radius $r = 2$. Note that the patch from the same location k is not included in this transient dictionary since the stasis prior is already handled using the technique of the previous section. Our approach has some similarities to that of [6].

We now select the optimal patch from the combined dictionary $\mathcal{D}_0 \cup \mathcal{D}_k^{(t-1)}$ by minimization of the error function

$$\begin{aligned} \mathcal{E}_k^{(\alpha,\beta,\gamma)}(t) \quad = \quad & \|\mathbf{x}^{(t)} - \mathbf{x}_k\|^2 \\ & +\alpha \mathcal{V}(\mathbf{y}_k, \{\mathbf{y}_{k'} : k' \in \mathcal{N}_{\mathbf{x}}^{(t)}\}) \\ & -\beta I(\mathbf{y} = \mathbf{y}_k^{(t-1)}) \\ & -\gamma I(\mathbf{y} \in \mathcal{D}_k^{(t-1)}). \end{aligned} \tag{5}$$

It should be noted that patches in the transient dictionary $\mathcal{D}_k^{(t-1)}$ will have mid frequency patches that will tend to be much more highly correlated with the mid frequency test patch in the new frame compared with those in the dictionary $\mathcal{D}_0$ (since the former are taken from the previous frame while the latter are taken from an independent set of training images). This needs to be reflected in the value of γ. It should be noted that setting $\gamma = -\infty$ causes the new synthetic patches never to be selected, and so in this case the algorithm reverts to the one presented in the previous section. Setting $\gamma = 0$, however, still leaves a model which can handle motion effects since the additional synthetic patches are included in the complete set of candidate patches and so will sometimes be selected. Some informal empirical optimization leads to a practical value of $\gamma = -\frac{1}{2}\beta_0$.

The resulting super-resolved video shows a considerable reduction in motion-induced high frequency artifacts compared to that obtained using the stasis prior alone. This is borne out by the error curves shown in Figure 7. Compared to Figure 5 we see that a modest reduction in flicker over the moving region (green curve) is obtained, and this is in fact greatest at times of slow motion (eg frames 25–30 in the figure). This improvement is at the expense of a modest increase of flicker in static areas (red curve).

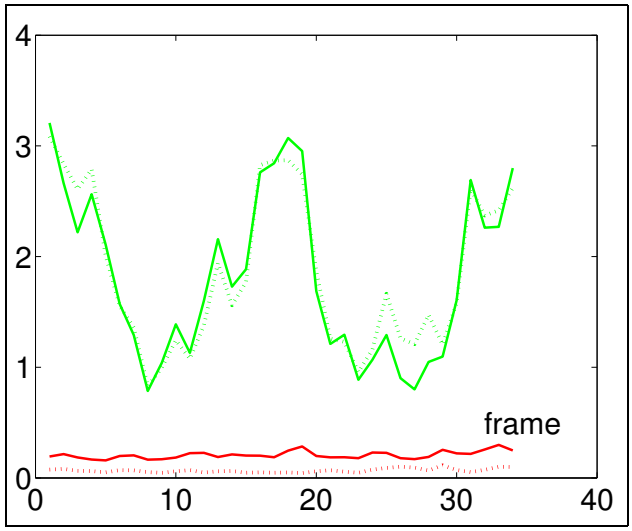

Figure 7: Flicker traces for super-resolution video obtained using the cost function (5), again with the red curve taken from a pixel in the red rectangle of Figure 1 and the green curve taken from a pixel in the green rectangle. For comparison, corresponding curves from figure 5 are shown dotted here.

5 Discussion

We have shown that techniques developed for super-resolution of static images can be successfully extended to the video domain when additional priors are introduced to take account of temporal consistency

between successive frames. The prior for stasis is particularly powerful here, with a modest further improvement coming from a prior that favours slow motion.

A different approach to temporal consistency, which we have also briefly explored, involves priors which favour patches which are proximate, not in the previous frame, but in the *training* image. This can be implemented efficiently using the index compiled during the construction of the kd-tree. Our results indicate that this approach is less powerful than the one described in detail in this paper.

A related task, to which our approach is also applicable, is that of video compression. In this case the whole video would be transmitted at some lower resolution than the original, and the dictionary of patch pairs would be taken from one or more representative high resolution frames of the video. Since the training data is taken from the video itself, the quality of the super-resolution enhancement in this case can be particularly good.

Statistics of patch re-use from this approach show substantial re-use of patches due to the stasis prior, and this bodes well for the compression application. We also note that setting $\alpha = 0$ has little discernable effect on the quality of the results in this case, so some computational speed-up can be achieved by omitting this term from the error function and this would be important for any application to real-time decompression.

References

[1] Jeffrey S. Beis and David G. Lowe. Indexing without invariants in 3d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1000–1015, 1999.

[2] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. *IEEE Computer Graphics and Applications*, 22(2):56–65, 2002.

[3] R. C. Hardie, K. J. Barnard, and E. A. Armstrong. Joint MAP registration and high-resolution image estimation using a sequence of undersampled images. *IEEE Transactions on Image Processing*, 6(12):1621–1633, 1997.

[4] S. A. Nene and S. K. Nayar. A simple algorithm for nearest neighbor search in high dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):989–1003, 1997.

[5] R. R. Schultz and R. L. Stevenson. A Bayesian approach to image expansion for improved definition. *IEEE Transactions on Image Processing*, 3(3):233–242, 1994.

[6] H. Sidenbladh, M. J. Black, and L. Sigal. Implicit probabilistic models of human motion for synthesis and tracking. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, editors, *Proceedings European Conference on Computer Vision ECCV*, volume 1 of *Lecture Notes in Computer Science 2353*, pages 784–800. Springer-Verlag, 2002.

[7] Michael E. Tipping and Christopher M. Bishop. Bayesian image super-resolution. In *Advances in Neural Information Processing Systems*, 2002. Accepted for publication.

Appendix: Constructing training patches

In summary, the training algorithm requires the following steps for patch construction. For each image in the training set (treating the R, G and B channels independently):

1. Perform a local smoothing of the image to remove the high frequency information, leaving an intermediate image which we shall regard as a sum of low and medium frequencies. This is achieved by convolution with the kernel

$$\begin{pmatrix} 0.0 & 0.25 & 0.0 \\ 0.25 & -1.0 & 0.25 \\ 0.0 & 0.25 & 0.0 \end{pmatrix}$$

2. Subtract the smoothed image from the original image to leave a high frequency image.

3. Take the intermediate image and smooth by convolution with a Gaussian kernel to give a low frequency image.

4. Subtract the low frequency image from the intermediate image to leave a medium frequency image. Note that the low, medium and high frequency images all have the same spatial resolution which is the same as that of the starting image.

5. Take the square of the pixel intensities in the medium frequency image, smooth and then take the square root (adding $\epsilon = 0.01$ to avoid subsequent division by zero). Divide both the medium and high frequency images by this energy image to achieve local intensity normalization.

6. Extract all possible patches of size 7×7 pixels from the medium frequency image along with the corresponding (concentric) patches of size 5×5 from the high frequency image.