Low-Cost and Real-Time Super-Resolution over a Video Encoder IP.

Conference Paper · January 2003

DOI: 10.1109/ISQED.2003.1194713 · Source: DBLP

4 authors, including:

Gustavo Marrero Callico
Universidad de Las Palmas de Gran Canaria
148 PUBLICATIONS **1,001** CITATIONS
SEE PROFILE

Antonio Nunez
Universidad de Las Palmas de Gran Canaria
141 PUBLICATIONS **462** CITATIONS
SEE PROFILE

Low-Cost and Real-Time Super-Resolution over a Video Encoder IP

Gustavo M. Callicó, Antonio Núñez
Applied Microelectronics Research Institute
Department of Elect. and Automatic Engineering
ULPGC, Canary Islands, Spain
gustavo,nunez@iuma.ulpgc.es

Rafael Peset Llopis[1], Ramanathan Sethuraman[2]
[1]*Philips Consumers Electronics*
[2]*Philips Research Laboratories Eindhoven,*
Royal Philips Eindhoven, The Netherlands
rafael.peset.llopis, ramanathan.sethuraman@philips.com

Abstract

This paper addresses a low-cost and real-time solution for the implementation of super-resolution (SR) algorithms over SOC (System-On-Chip) platforms in order to achieve high-quality image improvements. Low-cost constraints are accomplished in the sense that SR is performed without developing a specific hardware, but re-using a video encoder IP block. This encoder can be used either in compression mode or in SR mode. This video encoder together with the new SR features constitutes an IP block inside Philips Research, upon which several SOC platforms are being developed. Furthermore, this work can be easily adapted to other video encoder platforms.

1. Introduction

The straightforward way to increase the resolution of an image is to use higher resolution sensors, at the expense of higher cost. This however results in a decrease of the size of the active pixel area where the integration of light is performed. As lower amounts of light reach the sensor it will be less sensitive to shot noise. It has been estimated that the minimum photo-sensor size is around $50\mu m^2$ [1], a limit that has already been reached by the CCD technology. One solution to this problem is to increase the resolution using algorithms such as the super-resolution (SR) algorithm, wherein high-resolution images are obtained with low-resolution sensors at low cost. SR is also a smart way to perform image zooming without using mechanical parts to move the lenses.

1.1. SR algorithms and previous work

From the first frequency domain based SR algorithm, initially proposed by Huang and Tsay in 1984 [2], several SR algorithms have been proposed with diverse results. In order to obtain significant improvements in the resulting SR image, some amount of aliasing in the input low-resolution images must be provided. Although unlike [3] this approach is not performed in the frequency domain, we maintain the general scheme of registration and restoration. This work is more related to [4] in the sense that it is performed in the spatial domain and starts from a set of low-resolution displaced images, but without a blur correction and without the need of a precise knowledge of the motion among images. As in [5] we take into account sub-pixel displacements and like [6] we use the approach of multiple image restoration. Nevertheless, all the previous methods assume global and uniform translation among the different pictures, whereas this work makes no assumption about the motion among scenes.

The SR algorithm presented in this paper is the non-iterative version of the algorithm presented in [7] in order to enable real-time implementation. The new algorithm is able to combine in a single step a new incoming frame, keeping all the information from the previous frames. As we allow new frames to be combined (provided that some temporal correlation among them exists) the quality of the resulting frame will be considerably increased, whenever the incoming frames are shifted with respect to the reference one, in the sense that they incorporate some new information. Nevertheless, we have an obvious trade-off between the quality and the time spent obtaining the super-resolved frame: the higher the number of frames combined, the longer the time to deliver a new SR frame.

1.2. The hybrid video encoder platform

We begin with the hardware/software platform that exists in a hybrid video encoder. We seek to modify the existing platform as less as possible (low cost constraints) in order to map the SR algorithm. Although we have used an existing hybrid video encoder developed by Philips [8,9] the SR algorithm can probably easily fit in any other video encoder.

The architecture used in Philips is shown in Figure 1. The software tasks are executed on an ARM processor and the hardware tasks are executed on the VLIW processors (namely, pixel processor, motion estimator processor, texture processor and stream processor). The pixel processor (PP) communicates with the pixel-domain (image sensor or display) and performs line to stripe (16 lines) conversion. The motion estimator processor (MEP) evaluates a set of candidate vectors received from software and selects the best vector for full, half and quarter pixel refinements. The output of the MEP consists of motion vectors, sum-of-absolute-difference (SAD) values, and intra metrics. This information is used in

0-7695-1881-8/03 $17.00

software (running on ARM) to determine the encoding approach for the current macro-block (MB).

The texture processor (TP) performs the encoding of MBs and stores the decoded MBs in the loop memory. The output of the TP consists of VLE (variable length encode) codes for the DCT coefficients of the current MB. The stream processor (SP) packs the VLE codes for coefficients and VLE codes for headers generated by software on the ARM.

2. Algorithm description

Although the iterative version described in [7] offers very good image quality mapped onto a hybrid video encoder, the challenge is to create a new type of algorithm that, using the same resources, could operate in a single step, i.e. a non iterative algorithm. The main idea is based on the following considerations:

- Every new image adds new information that must be combined into a new high-resolution grid.
- It is impossible to know 'a priori' (for the super-resolution algorithm scope) the position of the new data and whether or not they contribute new information

Based on the previous considerations, the following algorithm has been developed:

1. Initially, the first low-resolution image is taken and its pixels are placed in a high-resolution grid, leaving the uncovered pixels to a zero value. From now on, this process will be denominated 'up-sample holes'. As the size-increase is a factor of two in both directions, the location and relationship among the pixels of high and low resolution is as seen in Figure 2.
2. Next, the contributions of the pixels are generated. The contributions are the weights assigned to each pixel to denote the amount of information provided to that pixel position. As we are initially combining several low-resolution images in a grid 2-by-2 times bigger, an initial contribution of 4, for ½ pixel precision in low-resolution will be enough. If the resolution of the motion estimator was increased or the motion-estimation was performed in high-resolution, higher values would be necessary. These contributions are expressed over the high-resolution grid. Thereby, the contributions of the image in Figure 2 are shown in Figure 5 (b)-left, pointing out that the original pixels have maximum contribution (four) and the rest have zero value.
3. Now, the relative displacements between the next input image and the first image, that will be the reference, are estimated. These displacements are stored in memory, as they will be used later on.
4. Steps 1 and 2 are applied to the new input image, i.e. it is adapted to the high-resolution grid, leaving the missing pixels to zero and generating the initial contributions.
5. In this step, both the new image over the high-resolution grid and its associated contributions are motion compensated toward the reference image. The real contributions of the new pixels to the high-resolution reference image will be reflected in the compensated contributions.
6. Now, the linear summation of the initial image and contributions with the compensated image and contributions is performed. This summation assures further noise reduction in the resulting image.
7. Steps 3 to 6 are applied to the next incoming images.
8. Once step 7 is finished, we will have a high-resolution image with the summation of all the compensated pixels and a memory with the summation of all the compensated contributions. Then the high-resolution image is adjusted depending on the contributions, as it is indicated in equation (1), where N is the number of frames to be combined.

$$SR(i,j) = N \cdot \frac{\sum_{fr=1}^{N} Motion\_Compesate\left(Upsample\_Holes\left(LR\_I[fr]\right)\right)}{\sum_{fr=1}^{N} Motion\_Compesate\left(contributions[fr]\right)} \quad (1)$$

9. After the adjustment, it is possible that some pixel positions remain empty, i.e., from the input image set

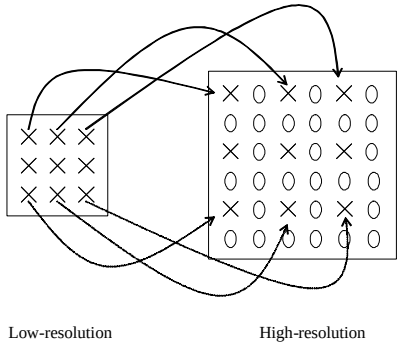


Figure 2. Mapping of the low-resolution pixels in the high-resolution grid, leaving holes for the missing pixels.

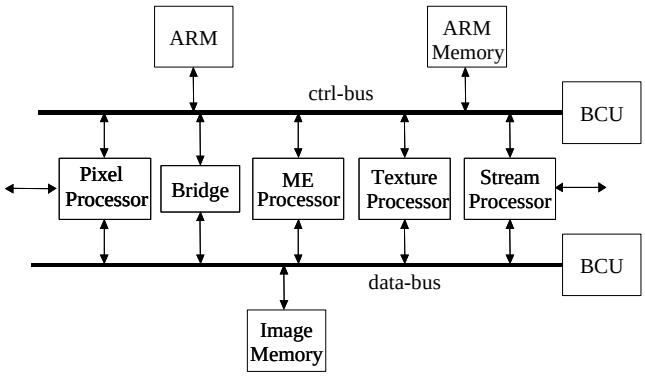


Figure 1. Architecture for the multi-standard video/image codec developed in Philips Research.

certain positions do not add new information. This case will be denoted with a zero, both in the high-resolution image position and in the contribution and the only solution is to interpolate the zeroes with the surrounding information. However, we cannot conclude that a zero in the image implies that the value must be interpolated, because a zero is a possible and valid value in an image. A pixel will be interpolated only if its final contribution is zero.

Assigning to the accumulative memory HR_A a length of 12 bits, 16 frames can be safely stored in it. In the worst case, we would accumulate a value of 255, which multiplied by 16 gives 4080, that fits in 12 bits. Figure 3 shows this algorithm in pseudo-code, using the memories and the resources of the Philips video hybrid encoder.

With respect to the iterative version of [7], not only the iterative behaviour has been removed, but also one of the motion estimations has been eliminated. As the motion estimation is performed in low-resolution and the motion compensation in high-resolution, it is necessary to multiply by two all the motion vectors to scale them properly.

3. Implementation on the video encoder

In order to fit the SRA in the video encoder originally developed by Philips Research, it was necessary to perform several changes in the architecture, namely in the motion compensator and in the chrominance treatment, thereby creating a more flexible SOC platform.

```
nr_frames = scale*scale, M'= scale*M, N'= scale*N
HR_B, HR_S, HR_T, HR_T2, HR_Cont and HR_S2 are all of 8
bits and size [M'][N'].
HR_A is 12 bits and size [M'][N'].
LR_I[M][N] for the motion estimation
Store the first frame in LR_I_0[M][N] and the remainders in
LR_I[M][N]
HR_A.lum    = Upsample_Holes(LR_I_0.lum)
HR_A.chrom = Upsample_Neighbours(LR_I_0.chrom)
HR_Cont     = Create_image_contributions
FOR fr = 1 .. nr_frames-1
    MV = Calc_Motion_Estimation ( LR_I, LR_I_0)
    MV = 2 .* MV
    HR_S.lum    = Upsample_Holes(LR_I.lum)
    HR_S.chrom = Upsample_ Neighbours(LR_I.chrom)
    HR_S2 = Create_image_contributions
    HR_T   = Motion_Compensation(HR_S, MV)
    HR_T2 = Motion_Compensation(HR_S2, MV)
    HR_A     = HR_A + HR_T
    HR_Cont = HR_Cont + HR_T2
END FOR
HR_A = 4*HR_A /HR_Cont
If  (HR_Cont(i,j)==0) THEN HR_B = Interpolate(HR_A(i,j))
ELSE   HR_B = HR_A
Clip(HR_B, 0, 255) // result image in HR_B
```

Figure 3. Kernel pseudo-code of the non-iterative versions of the super-resolution algorithm.

3.1 Adjustments in the motion compensator

The motion compensator implemented in the existing video encoder was designed to avoid visual distortions in the resulting images when decompressing them, and in that sense, when an image is shifted out of the physical boundaries, it fills the empty zone by replicating the borders. As the motion vectors are usually small compared with the image size and due to the lower attention of the human eye to the borders compared with the centre of the images, this effect is negligible. But, when we want to obtain super-resolution improvements, the artificial introduction of non-existing data results in quality degradation in its borders. The solution was to modify the motion compensator to fill the empty values with zeroes, so that the SRA would have an opportunity to fill the holes with valid values coming from other images. The algorithm does not expect the spontaneous generation of "new" data, which have not been taken into account, neither in the previous stages of the algorithm or in the contributions.

3.2 Adjustments in the chrominances

Due to the different sampling scheme used for the luminance and the chrominances, it is necessary to perform some modifications in the proposed algorithm to obtain super-resolution improvements also in the chrominance images.

First of all, for the chrominance pixels, the way to take the samples to the high-resolution grid or up-sample, cannot be the same as with the luminance. This fact is reflected in Figure 4, where it can be seen how every chrominance pixel affects four luminance pixels, and therefore, when the chrominance high-resolution grid is generated, every pixel must be replicated four times, in order to keep the chromatic coherence, as shown in Figure 5 (a).

For the same reason, the initial contributions cannot be same as luminance. As there is no zero-padding, all the chrominance pixels must initially have the same contribution weights. Figure 5 (b) shows the initial contributions for the luminance and the chrominances.

It must be noticed that the output image will be four times larger (2× in horizontal and vertical direction) than the input image, so memory requirements increase.

The motion estimation and motion compensation tasks are performed using the motion estimator and the motion compensator blocks, but have been modified to work in quarter pixel precision. The arithmetic operations such as additions, subtractions and arithmetic shifts are implemented on the texture processor. The overall control is done in software, on the ARM.

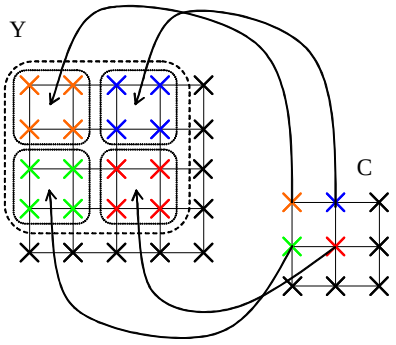
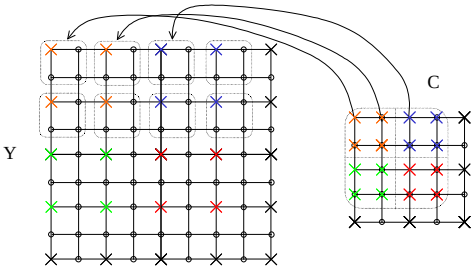


Figure 4. Relationship between the chrominance and the luminance for sampling format YCbCr 4:2:0.



(a)

Luminance contributions:

4	0
0	0

Chrominance contributions:

4	4
4	4

(b)

Figure 5. Mapping of the chrominance C to the high-resolution grid by means of replicating the pixels and its relationship with the luminance Y (a). Initial contributions for the luminance and chrominance images (b).

4. Results

In order to demonstrate the quality increase in the SR image when combining several low resolution images, we have designed the following experiment: A set of 12 displacement vectors have been generated, wherein the first is the zero vector and the remaining eleven are random vectors. The first displacement vector is (0,0) to assure that the resulting image will remain with zero displacement with respect to the reference, enabling reliable quality measurements. From this vector set, the first three vectors are applied to the first frame of the KRANT sequence and these frames are used to compose a super-resolved image (frame 0 in the Figure 8-a.1) based on three low-resolution frames. After that, a new vector is added to the previous set and these four vectors are applied again to the frame 0 of KRANT to generate a new super-resolved image based on four input images, and so

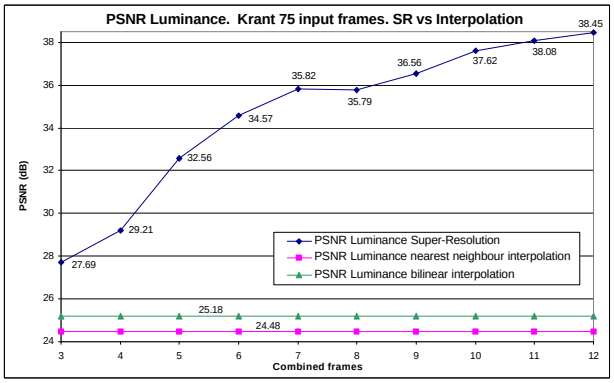


Figure 6. PSNR of the Krant sequence with 10 incremental output frames.

on until a super-resolved image based on 12 low-resolution frames have been generated. In total, a number of 3+4+5+6+7+8+9+10+11+12=75 low-resolution frames have been generated to be used as inputs to the SRA. These 75 input frames will generate 10 output frames, whose qualities are shown in Figure 6.

As it was expected, as the number of input frames to be combined increases, the PSNR increases until it reaches 38.45 dB when combining 12 low-resolution input frames. Empirically it can be verified that after combining 6 input frames, i.e. when the 34.57 dB is reached, the human eye hardly perceives enhancements in the quality of the image. In addition, it can be seen that the greater increment in the quality (greater PSNR slope) takes place in the first four output frames. All the established facts lead us to the conclusion that a system can be limited to combine 5 or 6 input frames, depending on the available resources and the desired output quality. Comparing these results with the iterative algorithm in [7] we appreciate that for the best case we reached 34.37 dB after 42 iterations, nearly 4 dB below the combination of 12 low-resolution frames, performing only 11 motions estimations instead of 42 motions estimations. Combining only 6 frames we reach a similar quality (34.57 dB) than using 42 iterations.

In Figure 7 (showing results for combining 12 low-resolution frames) it can be noticed that the image error in the spatial domain (a.2) is highly uniform, which indicates a low error with respect to the reference image. The bi-dimensional Fourier transform in magnitude shows an almost complete removal of the aliasing, exhibiting a minimal image error in the high-power spectral bands (b.2). The minimal spectral error in phase (c.2) also follows the same trend.

In Figure 8 (showing results for combining 3 low-resolution frames) higher amount of aliasing and consequently higher error can be seen.

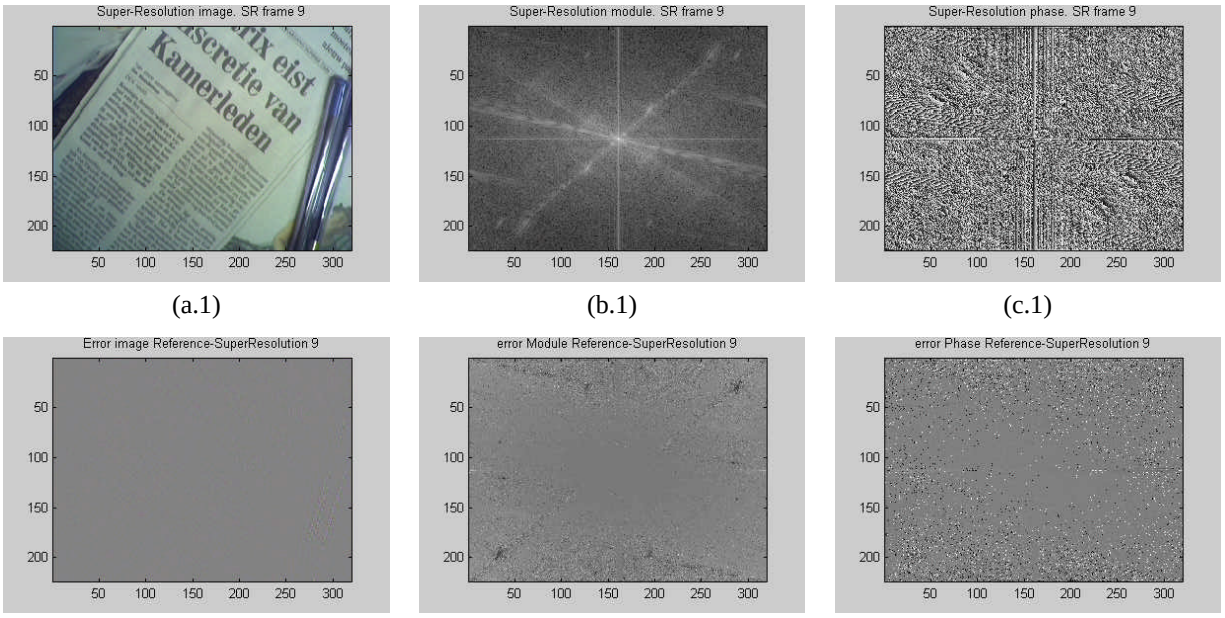


Figure 7. Super-resolved frame after combining 12 low-res frames in the spatial domain (a), in the frequency domain in magnitude (b) and in phase (c), together with their associated errors (2).

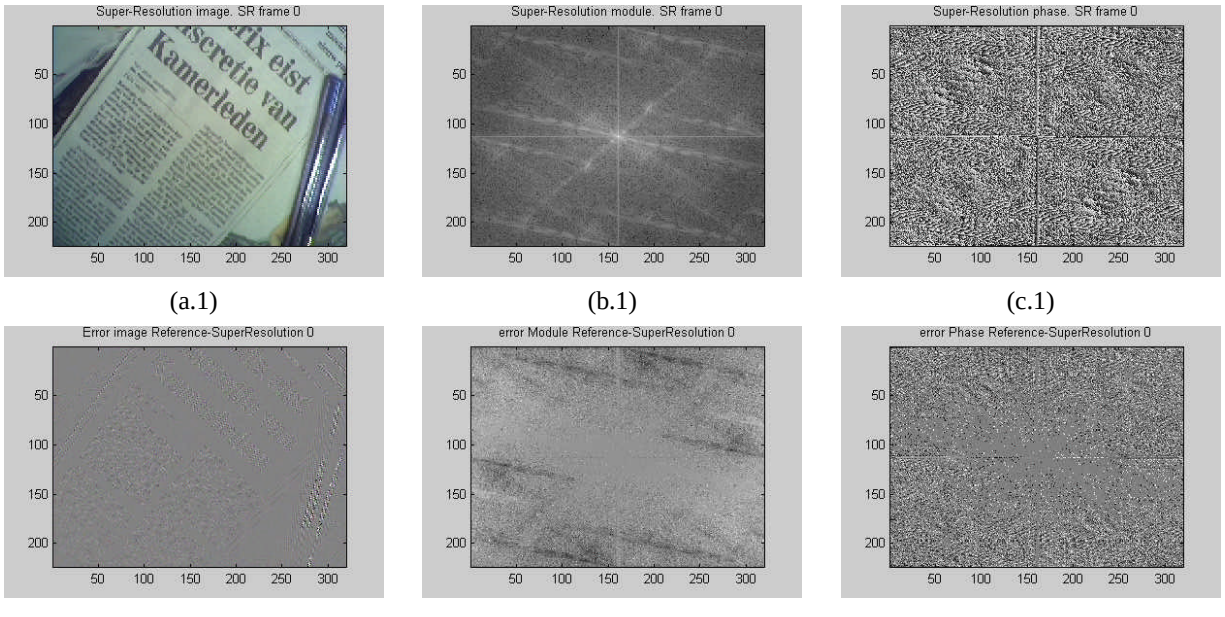


Figure 8. Super-resolved frame after combining 3 low-res frames in the spatial domain (a), in the frequency domain in magnitude (b) and in phase (c), together with their associated errors (2).

The chrominance PSNR (Figure 9) are always above the interpolation levels, although they do not follow as good as the luminance (Figure 6).

Figure 10 shows in detail the upper-right corner of the image KRANT, before (a) and after (b) the super-resolution process, together with the bilinear interpolated image (c). The quality improvement of the image is apparent, wherein the letters above the newspaper headlines are easier to read unlike the enlarged original input image and the interpolated image.

5. Conclusions

We presented, in this paper, an incremental SR algorithm that can be implemented on a video encoder

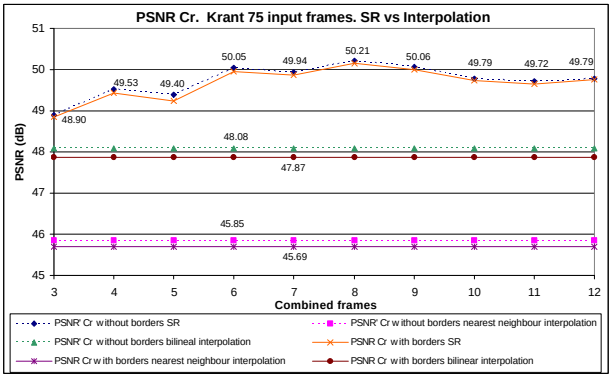


Figure 9. Red chrominance of the PSNR of the super-resolved and the interpolated images for the Krant sequence with 10 incremental output frames with and without borders.

with minimal changes. The experiments carried out exhibit a clear quality increase of the super-resolved image as the number of low-resolution frames to be combined also increases. The introduction of the contribution concept allows the algorithm to be relatively independent from the problems of the borders and, at the same time, proposes adaptive weights for every pixel, depending on the motion and therefore on the new information.

The SR algorithm works both in the luminance and in the chrominances, although the major improvements with respect to the interpolation levels are in the luminance component. This is mainly due to the lower entropy of the chrominances in the YCbCr 4:2:0 sample format, which does not allow a so accurate edge reconstruction as in the luminance. In other words, the chrominances have insufficient high frequency information to reconstruct a high-resolution color image with a quality much above the interpolated levels.

This algorithm presents an interesting approach to implement SR when a video-encoder architecture is available, as the cost in this case would be negligible. If the number of frames to be combined is limited to three or four, it is easy to achieve real-time constraints. Nevertheless, the memory requirements are still high and the application to a video sequence must be further refined.

6. References

[1] T. Komatsu, et al, “Very high resolution imaging scheme with multiple different-aperture cameras,” *Signal Processing: Image Communication* vol. 5, pp. 511-526, Dec. 1993.

[2] T. S. Huang and R. Y. Tsay, "Multiple Frame Image Restoration and Registration," *Advances In Computer Vision and Image Processing* (Ed. -T. S. Huang), vol. 1, JAI Press Inc., Greenwich, CT, 1984, pp. 317-339.

[3] H. Ur and D. Gross, “Improved Resolution from Sub-pixel Shifted Pictures,” *CVGIP: Graphical Models and image Processing*, vol. 54, pp. 181-186, March 1992.

[4] Lucas J. Van Vliet and Cris L. Luengo Hendriks, “Improving spatial resolution in exchange of temporal resolution in aliased image sequences,” in proc. of *11th Scandinavian Conf. On Image Analysis*, Kaugerlussauaq, Greenland, pp. 493-499, 1999.

[5] C. Srinivas, M. D. Srinath, “A Stochastic Model-Based Approach for Simultaneous Restoration I-Multiple miss-registered Images,” *SPIE*, vol. 1360, pp. 1416-1427, 1990.

[6] Marc J. Op De Beeck and Richard P. Kleihorst, “Super-Resolution of Regions of Interest in a Hybrid Video Encoder,” *Philips conference on DSP*, 1999.

[7] Gustavo M. Callicó, et al. “A Low-Cost Implementation of Super-Resolution based on a Video Encoder,”IEEE IECON ,Sevilla, Spain, Nov. 2002.

[8] R. Peset Llopis, et al, "A Low-Cost Low-Power H.263 Video Encoder for Mobile Applications,"*Second International Symposium on Mobile Multimedia Systems & Applications*, Delf, The Netherlands, Nov. 2000.

[9] R. Peset Llopis, et al, “HW-SW Codesign and Verification of a Multi-Standard Video and Image Codec,” *IEEE ISQED*, San Jose, California, March 2001, pp. 393-398.

(a) (b) (c)

Figure 10. Enlarged detail of the original image (a) and the super-resolution image (b) and the bilinear interpolation of the input image (c) for the KRANT sequence of 10 output frames.