Towards the Use of Super-Resolution in Biomedical Systems-on-Chip

Gustavo M. CALLICO, Sebastian LOPEZ, Felix TOBAJAS, Valentin DE ARMAS, Jose F. LOPEZ, Antonio NUÑEZ and Roberto SARMIENTO

Institute for Applied Microelectronics (IUMA) & Department of Electronic Engineering and Automatics (DIEA)
University of Las Palmas de Gran Canaria (ULPGC), Las Palmas de Gran Canaria, E-35017, SPAIN

ABSTRACT

Super-resolution is a smart process capable of generating images with a higher resolution than the resolution of the sensor used to acquire the images. Due to this reason, it has acquired a significant relevance within the medical community during the last years, especially for those specialties closely related with the medical imaging field. However, the super-resolution algorithms used in this field are normally extremely complex and thus, they tend to be slow and difficult to be implemented in hardware. This paper proposes a new super-resolution algorithm for video sequences that, while maintaining excellent levels in the objective and subjective visual quality of the processed images, presents a reduced computational cost due to its non-iterative nature and the use of fast motion estimation techniques. Additionally, the algorithm has been successfully implemented in a low-cost hardware platform, which guarantees the viability of the proposed solution for real-time biomedical systems-on-chip.

Keywords: Super-resolution, medical imaging, motion estimation, hardware mapping.

1. INTRODUCTION

The imaging field is gaining an increasing importance within the worldwide medical community, as it greatly helps the health personnel in order to establish a correct diagnosis. Although the technology involved in these systems has experienced a considerable advance during the last decades, in many of them there is still need of increasing the resolution of the images beyond the resolution provided by the respective sensor. A smart solution to this problem is to increase the resolution using algorithms such as the super-resolution (SR) ones, wherein high-resolution images are obtained using low-resolution sensors at lower costs. In this sense, super-resolution can be defined as a technique to increase the image resolution of pictures by exploiting the spatio-temporal correlation of data in several displaced images. In fact, different super-resolution algorithms have proven to be very effective in the fields of magnetic resonance imaging (MRI) [1], positron emission tomography (PET) [2] or three-dimensional microscopy [3], just to name some. However, these algorithms present a complex and iterative nature that prevent their use in other brand new scenarios with real-time constraints and/or high levels of integration associated, like smart capsule endoscopy systems.

This paper addresses low-cost solutions for the implementation of SR algorithms over SoC (System-on-Chip) platforms in order to achieve high-quality image improvements. These solutions are based on the following three tips: a) to radically modify the SR algorithms, breaking their iterative behavior in order to speed up their execution, b) to accelerate as much as possible the motion estimation stage, as it represents the most computationally intensive process within the SR procedure, and c) to avoid the need of developing specific SR hardware by reusing the logic present in an existing SoC imaging platform. Several results will be presented in terms of amount of memory needed, computational load and image quality in order to demonstrate the viability of the proposed solution for the next-generation of intelligent biomedical imaging systems.

The remainder of this paper is organized as follows. Section 2 presents the basis of the dynamic SR algorithm used along this work. Section 3 details the study performed in this work in order to accelerate as much as possible the motion estimation stage of the SR algorithm without sacrificing the visual quality of the processed images. Section 4 presents the details of the implementation of the algorithm onto a generic hardware platform and finally, Section 5 outlines the conclusions obtained in this work as well as future research lines.

2. THE DYNAMIC SR ALGORITHM FOR VIDEO

The approach to SR followed in this work consists on gathering information from a shifted image set in order to integrate all the available information in a new super-resolved image.

After an exhaustive study of previous work in SR, the first super-resolution algorithm (SRA) developed during this research was described in [4], where a static iterative SRA was successfully mapped onto a generic hybrid video coding HW/SW platform. Nevertheless, the algorithm exhibited an iterative behavior which prevents its real-time execution. In pursuit of a real-time implementation, the algorithm was firstly modified to avoid the iterative behavioral obtaining a non-iterative version for the static SRA described in [5], where the mapping details onto the aforementioned platform are detailed in [6]. Although this version works properly for real time applications, it exhibits a major drawback in the memory requirements, which results to be very high for a low-cost single-chip implementation. In order to overcome these drawbacks, this paper proposes a new dynamic SRA for video. In particular, the proposed algorithm is composed by three different and somehow independent processes: registration, fusion and restoration.

Image registration is the task of finding the motion between two or more views of the same scene, although it does not necessarily describe the real motion of either the camera or the scene. For this purpose, a motion estimation stage is used, as it estimates the motion vectors between the current frame and each frame within a working window. In order to estimate

motion vectors in a feasible way, the frame is split into macroblocks (blocks of 16×16 pixels). It is important to note that, in order to obtain SR improvements, the precision of the motion estimator has to be at least the inverse of the scale factor (zoom scale). To improve the image quality by means of SR techniques, it is necessary to obtain information from the nearby frames and combine it with the current frame. This is the goal of the fusion stage, as it uses the positions pointed by the estimated motion vectors within the search window in order to add information. However, if the nearby frames do not have enough fresh information to be added to the current frame under processing, there will be some empty positions in the super-resolved frame. To display the processed frame properly, these empty positions, named holes within the field of this paper, must be interpolated by the restoration stage. Due to their inherent nature and functionality, the processes of registration, fusion and restoration will be referred along this paper as *motion estimation*, *shift and add*, and *fill holes*, respectively, as it is stated in Figure 1 where the SR approach followed in this work is depicted. As it is seen in this figure, the motion is firstly estimated by computing a set of motion vectors per macroblock (MB) with sub-pixel accuracy (¼ pixel) that will be used to compensate the motion of every new incoming frame towards the frame used as the reference one in every frame-time. A temporal window of WIN frames before and after the current processing frame has been used together with a search area of SA pixels around the current MB of size MBS. Next, all the information is gathered in a higher resolution grid by the *shift & add* process that will create the first super-resolved draft image. Finally, if any pixel is not filled throughout these processes, it will be interpolated by the *fill holes* process using a bilinear surface interpolator.

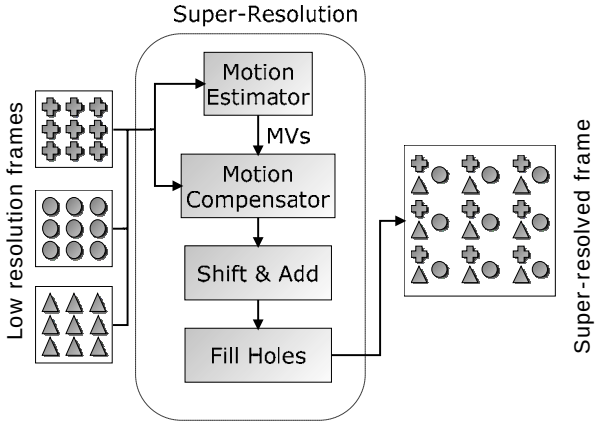


Figure 1. General approach for SR

An average comparison among the previously reported super-resolution algorithms is shown in Figure 2 in terms of quality of the super-resolved sequence (measured as the peak signal-to-noise ratio, PSNR) and memory requirements. As it is depicted, the single-step approach followed by the dynamic SRA reduces the memory requirements when compared with the static iterative SRA in [4], and the static non-iterative SRA in [5] and [6]. In particular, moving from static iterative to static non-iterative, produces a 40.32% increase in the memory requirements while the quality increases in 31.39%. However, moving from the static to the dynamic SRA (both non-iterative) decreases the quality in 1.92 dB (only a 6.3 %) while the memory requirements decrease in 94.84%.

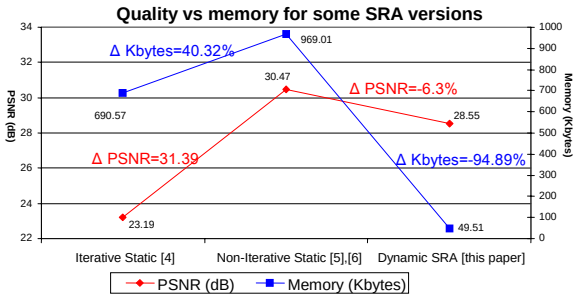


Figure 2. Comparison between the different SR algorithms

Although the proposed dynamic SRA represents an excellent tradeoff between the quality of the super-resolved images obtained and the amount of memory needed, it still has associated a huge computational cost that jeopardizes its real-time execution. In this sense, a profile of the SR execution reveals that, in average, about 51% of the computational cost relies on the ME process. Therefore, any effort focused on decreasing the computation load of this process will considerably speed up the overall SR process, allowing a reliable hardware implementation.

3. FAST MOTION ESTIMATION FOR DYNAMIC SR

Motion estimation represents, as it has been highlighted, a key task in the SR process, where the final quality of the super-resolved sequence critically depends on the accuracy of the motion vectors. From a hardware perspective, the block matching motion estimation algorithms suppose a higher guarantee for implementation, being the Full Search (FS) algorithm the only one that exhaustively evaluates all the MVs in a predefined search area, thus guarantying the lowest cost function used and the minimal distortion. The cost function commonly used is the Summation of the Absolute Differences (SAD) evaluated pixel-by-pixel between the given reference macroblock and every candidate position. The price to pay for this minimal distortion is a very high computational cost, given as a direct proportion with the square of the size of the search area. In this sense, a tradeoff problem between quality and computational load is encountered, being highly recommendable a situation where the quality loss, compared with the FS algorithm, is negligible with respect to a significant reduction in the number of operations to be performed.

For this purpose, a two-stage strategy has been followed in this work. In the first stage, a preliminary selection has been performed using nine fast block matching algorithms (FBMAs). Based in these results, the four FBMAs that exhibited the best performance in terms of quality and computation load have been selected to be evaluated in a second detailed stage. In this second stage, additional simulations have been performed in order to help in the decision of which FBMA could better substitute the FS algorithm for SR.

For the first preliminary selection and among all the ME algorithms available in the current literature, nine FBMA algorithms have been chosen based on their relevance in video compression environments. These algorithms are:

1. The Three-Step Search (TSS), described in [7].
2. The New Three Step Search (NTS), described in [8].
3. The Four step search (FSS), described in [9].
4. The Two Dimensions Logarithmic (TDL) search, described in [10].
5. The Cross Search Algorithm (CSA), described in [11].
6. The Diamond Search Algorithm (DSA), described in [12].
7. The Block Based Gradient Descent Search (BBGDS), described in [13].
8. The One at a Time Search (OTS), described in [14].
9. Parallel Hierarchical One Dimensional Search (PHODS), described in [15].

These nine algorithms, together with the FS algorithm as the reference, have been tested inside the SR environment returning data related to quality and computational load for each FBMA. In Figure 3, the higher-level block diagram of the SR-core is shown inside the setup environment. Every test starts reading the configuration file where several parameters as the images size, the number of frames to be processed, the block matching algorithm to be used, and the search area, among others, are fixed. Firstly, the input High-Resolution (HR) sequence is decimated in a given integer scale factor (factor 2 in this case) obtaining the Low Resolution (LR) sequence. At the same time, the LR sequence is interpolated using the same scale factor, as it will be used as the lower image quality to surpass. Once here, the SR process starts using the LR sequence as input.

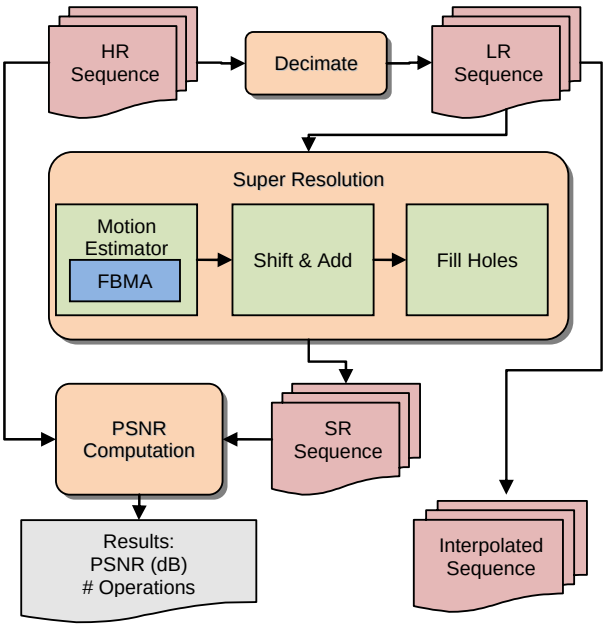


Figure 3. Test setup to select the best FBMA

In order to allow a reliable comparison among all the FBMAs, six real-life CIF sequences (352×288 pixels) commonly used in image processing testing (DEADLINE, FLOWERS, NEWS, MOBILE, CHILDREN AND FOREMAN), have been processed. The results obtained are depicted if Figure 4, where the quality obtained and the computation of the search algorithms are shown for the set of sequences used. It is important to mention that, in these charts, only the results obtained with the algorithms selected for the second analysis stage are shown. The quality is given as the average PSNR computed during 100 frames while the computational cost is given as the average number of operations performed by the ME algorithms.

The comparison between algorithms has been done by selecting the set of parameters (MBS, SA and WIN) that maximize the quality of the FS, which is different for every sequence. The set of parameters used for every sequence is shown in the header of every chart. As expected, the average quality always drops below the quality of the FS.

The average results for every FBMA in terms of performed operations, percentage of operations with respect to the FS, PSNR, PSNR loss with respect to the FS and the Standard Deviation (SD) of the PSNR loss, are summarized in Table 1.

TABLE 1
AVERAGE RESULTS FOR THE SIX SEQUENCES CONSIDERED

FBMA	Oper.	Oper. as % of FS	PSNR (dB)	PSNR Loss (dB)	SD PSNR Loss (dB)
NTS	$3.25 \cdot 10^8$	5.66%	26.78	0.09	0.11
TDL	$3.55 \cdot 10^8$	6.02%	26.30	0.57	0.39
DS	$2.81 \cdot 10^8$	4.91%	26.68	0.19	0.24
OTS	$1.98 \cdot 10^8$	3.46%	26.57	0.29	0.35

From these data, it is clear that the NTS is the most robust algorithm, which exhibit the lowest average PSNR quality loss (0.09 dB) and PSNR loss SD (0.11 dB). The NTS supposes a good tradeoff between quality (26.78 dB) and computational cost (5.66% of FS). The computation can be decreased by using the DS (4.91% of FS) with only a slight reduction of the quality (0.1 dB) but it is not so robust as the NTS, experimenting severe quality drops in sequences like “News” and “Children”. On the other hand, the OTS offers the lowest computational load (3.46% of FS) but at the cost of a low robustness (0.35 dB). The TDL is not a good candidate as it has the highest computation load (6.02% of FS) together with the lowest average quality (26.3 dB) and robustness (0.39 dB). The robustness of the NTS can also be observed in the six charts of Figure 4, where the PSNR variations are small, no matter what the sequence characteristics are. In order to visually inspect the quality of the images obtained with each FBMA, Figure 5 shows one frame of the “Mobile” sequence obtained with SR using three different FBMAs: NTS, DS, and OTS, and the interpolated image from the low-resolution version. In the bottom part of Fig. 12 an enlarged detail of the same frames reveals the video enhancements for the numbers of the calendar with respect to the bilinear interpolation.

These considerations make the NTS algorithm the best candidate to potentially substitute the FS algorithm in real-time SR application, and due to this reason, it has been selected as the motion estimation algorithm in the implementation of the SR algorithm detailed in the next section of this paper.

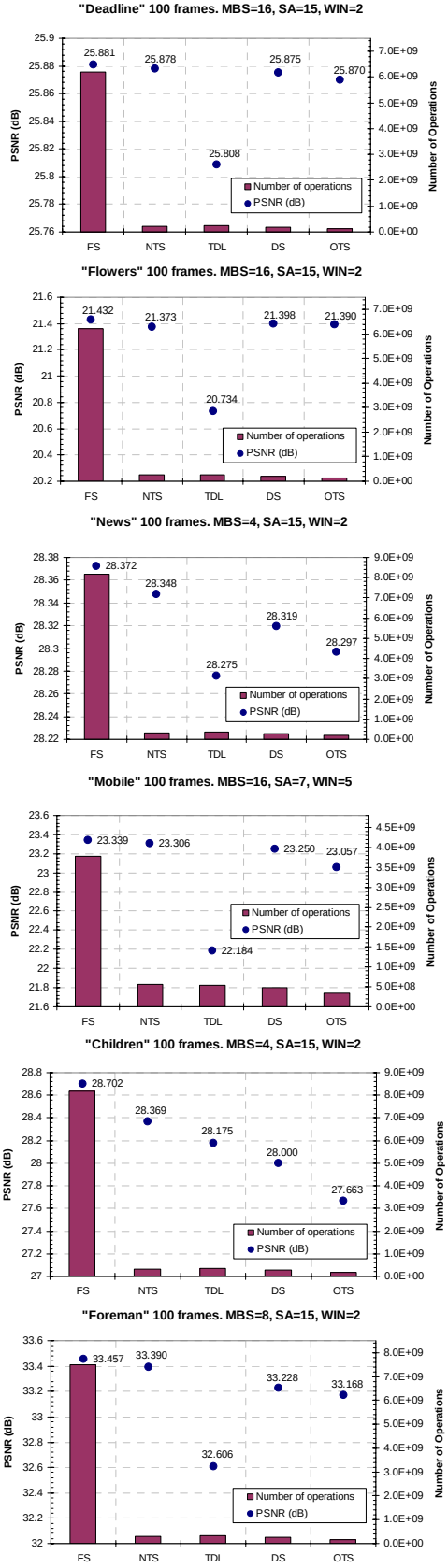


Figure 4. Quality and computational load for the set of CIF sequences used for SR and different FBMAs

Figure 5. Full frame (top) and detail (bottom) of the "Mobile" super resolved sequence using four different FBMAs.

4. IMPLEMENTATION DETAILS

The SR algorithm proposed in this work has been implemented onto the Texas Instruments (TI) TMS320DM642 Evaluation Video Module (EVM) hardware platform. The main device of this platform consists on a fixed point digital signal processor (DSP), named as DM642 in Figure 6, that works at a clock frequency of 720 MHz. The aforementioned DSP has 64 general purpose registers and 8 functional units, performing up to eight parallel instructions. In addition, it counts with a two-level cache memory. The first level has 128 Kbits for the program and 128 Kbits for general data, while the second level has 2 Mbits for program and data. Finally, the hardware platform also counts with some facilities for video applications to be mapped onto it: a 32Mbytes external SDRAM memory, capture video ports that support NTSC, PAL and SECAM formats, and the display video port that supports RGB, HD,

PAL, NTSC and S-Video.

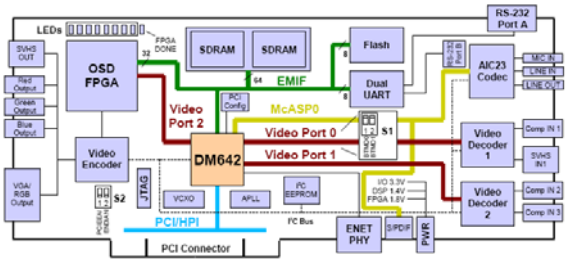


Figure 6. General diagram of the TMS320DM642 EVM platform

It is important to mention that the first implementation mapped onto this hardware platform was not fast enough in order to achieve real-time levels of performance. Due to this reason, the following procedures were considered:

- **Compiler options**: the compiler used has certain options that may improve the processing speed, such as loop unrolling, utilization of SIMD (Single Instruction Multiple Data) instructions, etc.
- **Loop simplification**: in order to have an efficient pipeline within the architecture, the loops in the code have to be as simple as possible, i.e. no function calls, control flow changes, data hazards or too much code inside the body of the loop.
- **Multiple memory accesses**: the DSP used in this work can access up to 64 bits in a single instruction, therefore it is possible to access eight bytes at a time.
- **Memory management**: the most used variables have to be placed into the internal memory, speeding up the access to these variables.

According to these procedures, the following 'operating points' were obtained, being the latencies obtained for each one of them for a 64×80 video sequence depicted in Figure 7:

- O1: First version (without any optimization).
- O2: Some inefficient parts of the code are rewritten based on the DSP profile.
- O3: The compiler options are used in order to improve the performance of the code.
- O4: Floating point operations removal. As the TMS320DM642 is a fixed point DSP, floating point operations are emulated, wasting several instructions and cycles. It is important to note that, since the floating point operations were replaced with fixed point ones, a quality loss was obtained. In this sense, the performance of the fixed-point SRA was tested again for several video sequences, obtaining for all the cases losses not greater than 0.01 dB.
- O5: Optimized library functions are used.
- O6: Multiple memory accesses in a cycle.
- O7: The most frequently used variables are placed into internal memory.
- O8: Further memory access improvements.

The application of these optimizations has finally led to a DSP-based implementation able to process YUV 4:2:0 CIF video sequences (352×288 pixels) sampled at 20 fps.

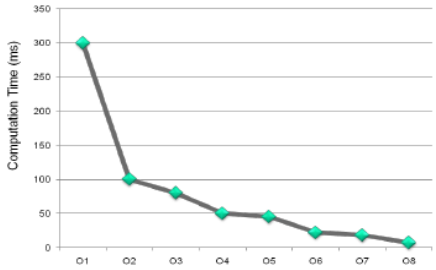


Figure 7. Latency obtained for each of the optimizations considered

5. CONCLUSIONS AND FURTHER RESEARCH

Super-resolution represents a promising solution for the medical imaging field. In this sense, some recent works have proven the usefulness of these techniques when applied to different types of medical images. However, the super-resolution algorithms utilized in these cases suffer from a double drawback that prevents their use in future biomedical systems-on-chip. The first one is their high computational cost, mainly because of the huge number of operations involved in the motion estimation stage. The second is their unsuitability for being implemented into a piece of hardware, due to their iterative and complex nature.

This paper has presented a set of strategies in order to overcome these challenging issues. Firstly, a new non-iterative super-resolution algorithm for video sequences has been introduced, characterized by its good compromise between the quality of the super-resolved video sequence and the amount of memory needed for the super-resolution process. In order to accelerate the execution of such algorithm, a study about the use of fast motion estimation algorithms has been also presented in this paper. The results obtained have revealed that the The New Three Step Search (NTS) algorithm is able to reduce the number of operations needed with respect to a generic exhaustive search algorithm in a 94% with negligible losses in the final video sequence quality. Finally, this super-resolution algorithm has been implemented onto a DSP-based platform. Starting from an initial implementation, the introduction of a set of simple tricks, mainly related with compiler options and the use of cache memories, has allowed to process CIF video sequences sampled at 20 frames per second or, alternatively, 128×160 video sequences sampled at 140 frames per second, assuring the viability of the proposals introduced in this paper.

In order to explore the goodness of the proposed super-resolution algorithms, it has been applied over a set of medical images, following the setup shown in Figure 8. The result obtained by using the bilinear interpolation for an upscale factor of 5 is shown in Figure 9(a), while the super-resolved image for this scale factor is shown in Figure 9(b). As it can be observed from these figures, the results obtained by using the super-resolution algorithm introduced in this paper are significantly better than the ones provided by interpolation, especially for large scale factors, as it is the case of the results shown in Figure 9. These encouraging visual proofs, together with its

aforementioned characteristics in terms of speed and ease of integration, definitely point out the proposed super-resolution technique as a potential candidate for real-time image and video enhancement in future biomedical systems-on-chip.

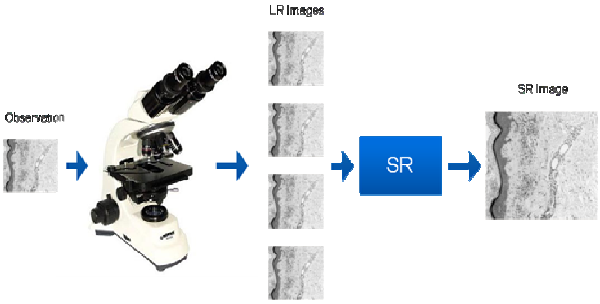
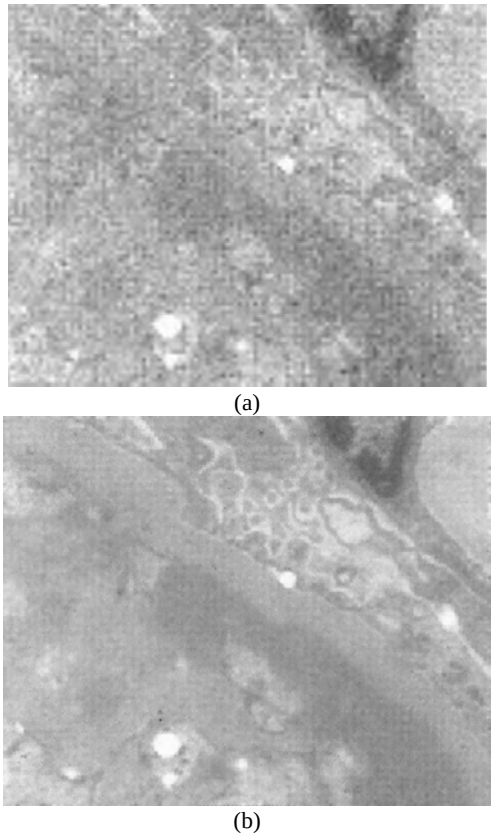


Figure 8. Setup for super-resolving medical images.



(a)

(b)

Figure 9. Comparison of the results obtained with bilinear interpolation and the proposed super-resolution algorithm

ACKNOWLEDGMENT

The authors would like to thank Prof. Kamran Eshraghian for his encouraging support during this work.

REFERENCES

[1] H. Greenspan, G. Oz, N. Kiryati and S. Peled, "Super-resolution in MRI", **Proceedings of the IEEE International Symposium on Biomedical Imaging**, 2002, pp. 943-946.

[2] J.A. Kennedy, O. Israel, A. Frenkel, R. Bar-Shalom, and H. Azhari, "Super-resolution in PET imaging", **IEEE Transactions on Medical Imaging**, vol. 25, no. 2, 2006, pp. 137-147.

[3] B. Huang, W. Wang, M. Bates and X. Zhuang, "Three-Dimensional Super-Resolution Imaging by Stochastic Optical Reconstruction Microscopy", **Science**, vol. 319. no. 5864, 2008, pp. 810 – 813.

[4] Gustavo M. Callicó, Rafael P. Llopis, Antonio Núñez, Ramanathan Sethuraman, Marc Op de Beeck. "A Low-Cost Implementation of Super-Resolution based on a Video Encoder," **IEEE IECON Conference**, vol. 2, 2002, pp. 1439-1444.

[5] Gustavo M. Callicó, Rafael P. Llopis, Antonio Núñez, Ramanathan Sethuraman. "Low-Cost and Real-Time Super-Resolution over a Video Encoder IP," **IEEE ISQED Conference**, 2003, pp. 79-84.

[6] Gustavo M. Callicó, Rafael P. Llopis, Antonio Núñez, Ramanathan Sethuraman, "Mapping of Real-Time and Low-Cost Super-Resolution Algorithms on a Hybrid Video Encoder", **SPIE Microtechnologies for the New Millenium**, vol. 5117, 2003, pp. 42-52.

[7] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," **Proceedings of the National Telecommunications Conference (NTC)**, 1981.

[8] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," **IEEE Transactions on Circuits and Systems for Video Technology**, vol. 4, no.4, 1994, pp. 438-442.

[9] L. Po and W. Ma, "A novel four-step search algorithm for fast block motion estimation," **IEEE Transactions on Circuits and Systems for Video Technology**, vol. 6, no. 3, 1996, pp. 313–317.

[10] J.R. Jain and A.K. Jain, "Displacement measurement and its application in interframe image coding," **IEEE Transactions on Communications**, vol. 29, no. 12, 1981, pp. 1799-1808.

[11] M. Ghanbari, "The cross-search algorithm for motion estimation," **IEEE Transactions on Communications**, vol. 38, no. 7, 1990, pp.950–953.

[12] S. Zhu and K.K. Ma, "A new diamond search algorithm for fast block matching motion estimation," **IEEE Transactions on Image Processing**, vol. 9, no. 2, 2000, pp. 287–290.

[13] L.-K. Liu and E. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding," **IEEE Transactions on Circuits and Systems for Video Technology**, vol. 6, no. 4, 1996, pp. 419–422.

[14] R. Srinivasan and K. Rao, "Predictive coding based on efficient motion estimation," **IEEE Transactions on Communications**, vol. 33, no. 8, 1985, pp. 888–896.

[15] E. Chan, A. Rodriguez, R. Gandhi and S. Panchanathan, "Experiments on block matching techniques for video coding," **Multimedia Systems**, vol. 24, 1994, pp. 228-241.