Defence Science Journal, Vol 49, No 1, January 1999, pp. 41-48

Target Handing over System using Pyramid Processing

M. Sankar Kishore
Research Centre Imarat, Hyderabad - 500 069.

ABSTRACT

The image seen by the low resolution (LR) sensor and the image (of the same scene) seen by the operator through the high resolution (HR) sensor are different in spatial resolution. To establish the correlation between these two images, the HR image needs to be resampled and made similar to the LR sensor image by applying a pre-processing technique. The pre-processing will be carried out by handing over system that resamples the HR sensor image making it compatible with LR sensor image and hands over the resampled image to the LR sensor through a serial link. The pre-processing technique reported by Boland[1] has been studied, analysed[2] and implemented[3] using 86 family of processors. Automatic target handing over system discusses the implementation of pre-processing technique, computational complexity and criticality of the execution time. The execution time reported was 1.5 s, whereas the requirement is of the order of few milliseconds for the typical set of conditions. This paper discusses the implementation of a suitable handing over algorithm[1] Emphasis has been to develop hardware and software to reduce the execution time, which has been brought down from 1.5 s to 40 ms for a typical set of conditions. Emphasis has also been given to reduce the transmission time by applying suitable pyramid processing techniques. The hardware is designed around *i*486 processor, and the software is developed in PL/M86.

1. INTRODUCTION

Acquisition and recognition of the target is not possible through the low resolution (LR) sensor because of the limited size and field of view (FOV). Hence, a high resolution (HR) sensor is required for acquisition and recognition of the target of interest[3]. High resolution (HR) sight and LR sensor are co-mounted on the launcher. The target acquired by the HR sight is supposed to appear at the centre of FOV of the LR sensor. After recognising the target through the HR sensor, the scene around the centre of the FOV can be resampled to make it compatible to LR sensor image in spatial resolution. The resampled image can be used as a reference image for locating its position in LR sensor FOV by image correlation techniques. Once the reference area is located in the LR sensor image, the LR sensor is automatically trained to bring the located area in the centre of its FOV. Image correlation technique continues updating reference information at a sufficiently fast rate, typically tens of milliseconds and thus the LR sensor keeps tracking the target area. Thus, the target is located in a given area[2,3]. To make the two images to have same resolution, a suitable pre-processing technique[1,2] is applied on the HR sensor image.

A suitable algorithm has been studied and is found to be satisfactory[1,2]. The system has been realised both in hardware and software[3]. The execution time was 1.5 s for a typical test condition. In this paper, emphasis has been given to reduce the execution time by reducing the

Received 01 January 1997, revised 10 June 1998

computational complexity of the algorithm and to reduce the transmission time by applying a pyramid processing technique. The resampled image was saught to be transmitted to the LR sensor through a serial link.

1.1 Handing over Algorithm & its Analysis

Consider two images of the same scene, taken by two different sensors: one with HR and the other with LR. The two images of the same scene may differ due to difference in (i) FOV (ii) sensor resolution (iii) sensor geometry (iv) frame rate and (v) sampling rate.

Let W_h be the horizontal scale factor and W_v be the vertical scale factor which are the functions of the above parameters between HR and LR sensor images. The simplified algorithm[1,3] is as follows:

$$\begin{aligned} Lr(i,j) = \frac{1}{W_v W_h} \Bigg\{ & \sum_{m=u+1}^{x} \sum_{n=v+1}^{y} Hr(m,n) \\ & + \sum_{n=v+1}^{y} \Big[(u-(i-1)W_v)\, Hr(u,n) + (i\, W_v - x)\, Hr(x+1,n) \Big] \\ & + \sum_{m=u+1}^{x} \Big[(v-(j-1)W_h) Hr(m,v) + (jW_h - y) Hr(m, y+1) \Big] \\ & + (u-(i-1)W_v)\, (v-(j-1)W_h) Hr(u,v) \\ & + (jW_h - y)\, Hr(u, y+1) \\ & + (i\, W_v - x)(v-(j-1)W_h) Hr(x+1,v) \\ & + (jW_h - y) Hr(x+1,\ y+1) \Bigg\} \end{aligned} \qquad (1)$$

where

u = Trunc $[(i\text{-}1)\ W_v + 1]$

v = Trunc $[(j\text{-}1)\ W_h + 1]$

x = Trunc $[i\ W_v]$

y = Trunc $[j\ W_h]$

The above algorithm, converts the $Hr\ (i,j)$ image into $Lr\ (i,j)$ image with W_h and W_v scale factors. It can be observed that it is computationally complex.

2. SYSTEM SPECIFICATIONS

The ground computer (GC) is a subsystem in the launcher, which interacts with all the subsystems in the vehicle and passes on the information to the LR sensor. A scanner converts the non-standard signals generated by the HR sensor to a standard composite format. The scanner also generates graphics to aid the operator during alignment and firing. The scanner is a processor-based subsystem with a serial link for transmitting data to and from handing over electronic (HOE). First, the GC gives a start command to HOE and the scanner. Then the scanner gives the pixel location of the target to HOE, which in turn resamples the image around those coordinates and generates LR image (which is having same spatial resolution as that of the LR sensor image). The HOE transmits the LR image data to LR sensor through GC as shown in Fig. 1.

The data is transmitted to GC through RS 232 serial link. It is recommended that for reliable data transmission, 9600 baud rate is optimum. For illustration, the transmission of a 16 x 16 image with 2 bytes of checksum at 9600 baud takes nearly 300 ms, (i.e. to transmit data from HOE to GC it takes 300 ms and similarly from GC to seeker, another 300 ms). The HOE resampling time is of the order of few tens of milliseconds but the transmission time is much longer. Hence, an attempt has been made to reduce the transmission time also by applying pyramid processing technique.

2.1 Reduction in Computational Complexity

In a typical configuration, the horizontal W_h and vertical W_v scaling factors are 14.20 and 14.634 respectively, i.e., 14.2 pixels horizontally and 14.634 pixels vertically are to be considered to generate one pixel in the LR image. In other words, to generate 16 x 16 LR image from a HR image with these scaling factors, the following tasks have to be performed :

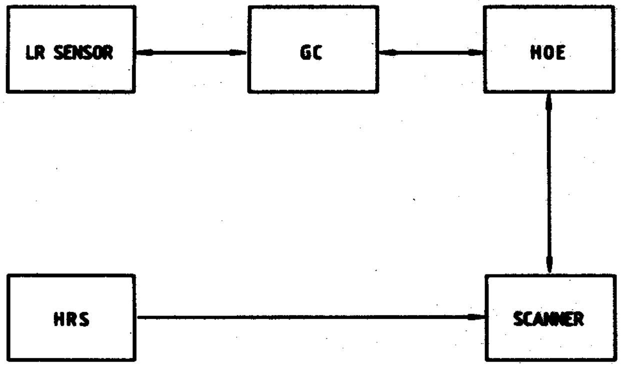

Figure 1. Block diagram of data flow

Floating point multiplications - 4096

Floating point additions - 12544

Integer additions - 1397576

Thus, the algorithm is computation-intensive. Once the image data is available in the memory, the computation time mainly depends on the speed of the processor and coprocessor, and the software code. In the 8086/87-based system[3], the processor and coprocessor take 1.5 s for generating LR image. In another typical configuration, the W_h and W_v become 16.89 and 15.14, respectively. The execution time with the 8086/87-based design hardware goes up to 9.36 s.

2.1.1 Effect of Fractional Pixels

A study has been carried out to find the effect of fractional pixels on the boundaries for large scaling factors (W_h and W_v). For generating LR image, only complete pixels in the HR image are considered. For resampling, partial/boundary pixels are neglected. For a particular W_h and Wv, let $Lr(l, m)$ is pixel value computed using all pixels (including boundary pixels) and Lr_e (l, m) is the pixel value computed by neglecting partial/ boundary pixels. Then, the RMS error is given by

$$RMS\ error = \left\{ \frac{1}{M^2} \sum_{l=1}^{M} \sum_{m=1}^{M} \left[Lr(l,m) - Lr_e(l,m) \right]^2 \right\}^{1/2} \qquad (2)$$

A fractional value x is added to both the scaling factors. The RMS error between the $Lr(l, m)$ and $Lr_e(l, m)$ is computed for modified scaling factors. The process is repeated by varying x from 0.1 to 0.9. The same procedure is repeated for different sets of W_h and W_v. This study has been carried out on a number of images and some of the results are given in Tables 1 and 2.

Table 1. RMS error in neglecting boundary pixels

Scaling factors	Fractional values (x)							
W_h = 12 + W_v = 13 +	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80
Image 1	0.53	0.56	0.49	0.53	0.36	0.54	0.47	0.48
Image 2	0.40	0.45	0.36	0.38	0.28	0.42	0.37	0.39
Image 3	1.47	1.66	1.44	1.69	0.92	1.43	1.30	1.61
Image 4	1.30	1.05	1.19	1.18	0.79	1.19	1.22	1.24

From these tables, it is found that the RMS error due to neglecting partial pixels is of the order of 1 on 256 grey level. Thus, the processing time can be reduced by neglecting the boundary pixels for large scaling factors.

2.1.2 Effect on Coarse Resampling

A study has been carried out to find the effect of coarse resampling. Let $Lr(l, m)$ be the pixel value computed from HR image with number of lines N and sampling rate S. Let $Lr_x(l, m)$ be the pixel value computed from HR image with number of lines N/x and sampling rate S/x. The RMS error due to coarse resampling is given by

Table 2. RMS error in neglecting boundary pixels

Scaling factors	Fractional values (x)							
W_h = 7+ W_v = 8+	0.01	0.20	0.30	0.40	0.50	0.60	0.70	0.80
Image 1	0.61	0.49	0.62	0.51	0.59	0.63	0.63	0.66
Image 2	0.57	0.60	0.68	0.94	0.56	0.52	0.66	0.60
Image 3	1.61	1.79	1.50	1.74	1.89	1.80	1.82	1.63
Image 4	1.70	1.59	1.38	1.43	1.88	1.98	1.72	1.71

$$RMS\ error(x) = \left\{ \frac{1}{M^2} \sum_{l=1}^{M} \sum_{m=1}^{M} \left[Lr(l,m) - Lr_x(l,m) \right]^2 \right\}^{1/2} \quad (3)$$

The scaling factors are made

$$\left[\frac{W_h}{2}, \frac{W_v}{2}\right], \left[\frac{W_h}{4}, \frac{W_v}{4}\right] \text{ and } \left[\frac{W_h}{8}, \frac{W_v}{8}\right]$$

. In each case, the RMS error between the original image and the image computed using reduced scaling factors is found. This exercise is carried out on a number of images. The results of some of the images are given in Table 3.

Table 3. RMS error in coarse resampling

Scaling factors	RMS error (x) for		
$W_v = 13$ $W_h = 12$	$X = 2$	$X = 4$	$X = 8$
Image 1	1.29	3.05	4.46
Image 2	1.15	2.24	3.73
Image 3	1.77	4.35	9.89
Image 4	1.40	4.08	7.89

From Table 3, when x = 2, the error due to coarse resampling is of the order of 1 or 2 in 256 grey levels. Thus, by coarse resampling, the information lost is negligible. Even when sampling rate is 4 MHz and alternate HR lines only are considered, the RMS error is of the order of 1 or 2 grey level in 256 grey levels. This little variation does not affect the correlation performance. Thus the processing time can further be reduced by neglecting boundary pixels and by coarse resampling.

2.2 Pyramid Processing

The Brut pyramid[4,5] is computationally an efficient algorithm for generating two-dimensional bandpass representation of an image. The pyramid decomposition process is fully invertible so that the original image can be exactly recovered from its bandpass components. Invertibility makes this algorithm ideal for image enhancing operations. The Brut pyramid is an algorithm used to separate an image into a set of contiguous spatial frequency bands. The original size image which may be 512 x 512 pixels in size or (2^9 x 2^9) is filtered by appling a low-pass filter which passes only half the frequency bands. Since, the maximum frequency in the new image is half that of the original, the image may be resampled eliminating half the samples in both directions and producing a four-fold reduction in data. The new image, which can be represented as 256 x 256 pixels, can itself be filtered. This process is known as a reduction operation and may be repeated in a pipeline fashion.

The first step is to low-pass filter the original image g_0 to obtain image g_1. It is assumed that g_1 is a reduced version of g_0 in both (horizontal & vertical) resolutions. In a similar way, g_2 as a reduced version of g_1 , and so on. Filtering is performed by a procedure equivalent to convolution with one of a family of symmetric weighing functions. The sequence of images $g_0, g_1, g_2, \ldots, g_n$ is called Gaussian pyramid. The Gaussian images are generated according to the REDUCE function, defined as

$$g_k\ (i,j) = REDUCE\ g_{k-1}$$
$$0 < K < Q \text{ for } Q \text{ level} \quad (4)$$

which is a shorthand notation for

$$g_k\ (i,j) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} W(m,n)\, g_{k-1}\ (2i+m, 2j+n) \quad (5)$$

for all samples, i,j, where $0 < i < Ck$ and $0 < j < Rk$. Here, Ck, Rk are the columns and row dimensions of image k. By the two-dimensional filtering or neighbourhood window function, $W(m, n)$ is constrained by:

(a) $W(m,n) = W(m)*(n)$ Filter separability

(b) $W(i) = W(-i)$ Filter symmetry

(c) $\sum_i W(i) = 1$ Normalised

(d) $W(0) + 2\sum_{i\,even} W(i) = 2\sum_{i\,odd} W(i)$ Equal contribution

By choosing $W(0) = 0.4$ and substituting in the above set of equations, one gets $W(1) = 0.25 = W(-1)$ and $W(2) = 0.05 = W(-2)$. As explained, in case[4,5] the shape of the equivalent weighing functions for Gaussian pyramid converges rapidly to a characteristic form with successively higher levels of the pyramid so that only its scale changes. However, this shape does depend on the choice of $W(0)$ in the generating kernel. By choosing the Gaussian kernel as explained above, the equation can be simplified as follows:

$$g_k(i,j) = \sum_{m=-2}^{2} W(m) \sum_{n=-2}^{2} W(n)\, g_{k-1}(2i+m, 2j+n) \tag{6}$$

$$= \sum_{m=-2}^{2} W(m)\Big[0.05\, g_{k-1}(2i+m, 2j-2) + 0.25\, g_{k-1}(2i+m, 2j-1) + 0.4\, g_{k-1}(2i+m, 2j) \; 0.25\, g_{k-1}(2i+m, 2j+1) + 0.05\, g_{k-1}(2i+m, 2j+2)\Big] \tag{7}$$

It can be written as a matrix product given as below:

$$g_k(i,j) = W^T\, G\, W \tag{8}$$

where W is 1 x 5 and G is 5 x 5 matrices, and

$$W = \big[\, W(-2)\, W(-1)\, W(0)\, W(+1)\, W(+2) \,\big]^T \tag{9}$$

The EXPAND process is repeated for the entire set of Gaussian images according to

$$g_{k,n}(i,j) = 4 \sum_{m=-2}^{2} \sum_{n=-2}^{2} W(m,n)\, g_{k,n-1}\left[\frac{(i-m)}{2}, \frac{j-n}{2}\right] \tag{10}$$

for nodes i, j where $0 < i < Ck - n$ and $0 < j < Rk - n$, and only for integer values of

$$\frac{(i-m)}{2} \quad \text{and} \quad \frac{(j-n)}{2}$$

$$g_{k,n} = 4 \sum_{m=-2}^{2} W(m) \left\{ 0.05\left[\frac{i-m}{2}, \frac{j+2}{2}\right] + 0.25\left[\frac{i-m}{2}, \frac{j+1}{2}\right] + 0.4\left[\frac{i-m}{2}, \frac{j}{2}\right] + 0.25\left[\frac{i-m}{2}, \frac{j-1}{2}\right] + 0.05\left[\frac{i-m}{2}, \frac{j-2}{2}\right] \right\} \tag{11}$$

The resampled image generated by Eqn (1) can be reduced in size (four-fold) by applying Eqn (8) and transmitting to LR sensor. At the LR sensor, it can be expanded by applying Eqn (11).

3. SYSTEM DESIGN

The block diagram of HOE hardware is shown in Fig 2. The hardware has been designed and developed using *i*486 DX 25 MHz Processor. To increase the processing speed, internal cache memory is designed for optimum operation. The software has been developed in PL/M 86 language. Substantially, fewer PL/M statements are necessary for a given application than if it were programmed in assembly language[6], thereby reducing the software development time.

The Sync (HS,VS) signals are separated from the scanner composite video signal using the sync separator. The video is sampled at 20 MHz and the 8-bit digital data can be written in dual port (DP) memory. The processor receives the target coordinates from scanner and enables the DP

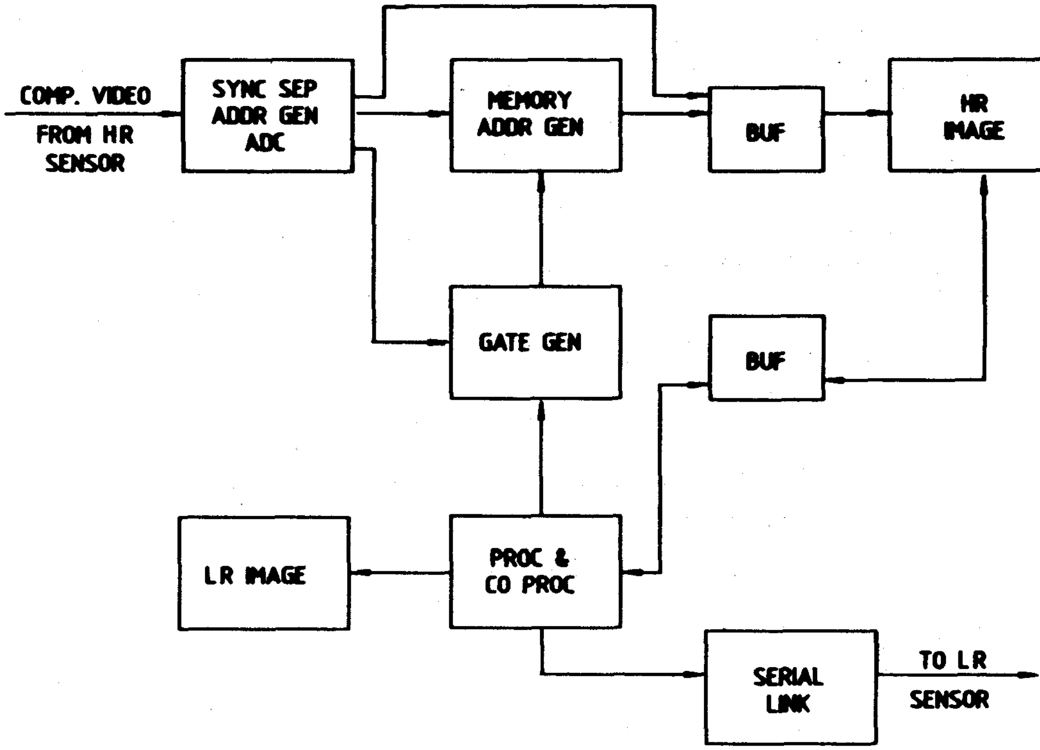

Figure 2. Hardware block diagram

memory to store the video data around the target. The execution time mainly depends on processor speed, and the time taken for reading data from the DP memory.

For fixed scaling factors, the address of the pixels are pre-computed and stored as lookup tables to reduce the execution time. Once the scaling factors are fixed, the algorithm is partitioned in such a way that the comparisons and jump statements are minimised, thereby the data fetching/processing time gets reduced drastically.

On power ON, all the peripherals are initialised and the processor waits for the start signal and for the target coordinates . The HOE generates edges of the target and applies the boundary conditions. Thus, a pre-defined size of the image around the target is captured in the DP memory. The HOE computes the LR image by applying the resampling algorithm [Eqn (1)]. On the resampled data, pyramid processing technique [Eqn (8)] is applied and a reduced version of LR data is generated. The reduced data is then transmitted to the LR sensor through a serial link.

The system is also provided with built-in test facility (BITF). The self-test is divided into two parts. In one part, the HR and LR images are transmitted and displayed on a PC simultaneously to see the input/output picture quality. In the other part, the status of each part of H/W is checked and corresponding messages are displayed on a terminal. The system also checks the sync signals and GATE generation. The memory is tested by writing and reading fixed values (0ffh), (00h), incremental and random pattern. The corresponding status is displayed on a terminal. The same procedure is repeated for testing DP memory.

4. VALIDATION OF THE SYSTEM

To evaluate the system in the laboratory both hardware and software, a high resolution video camera and the LR sensor were co-mounted on a movable platform. The image was taken from HR sensor and resampled in HOE hardware. The resampled image (reference) and LR sensor (search) images were transmitted to a PC and

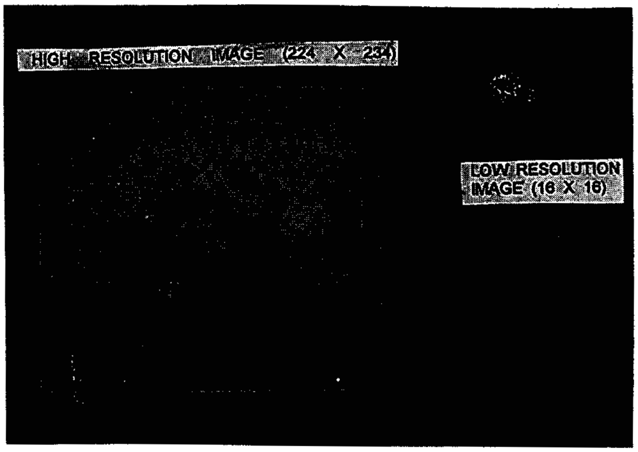

Figure 3. HR (224 ×234) conversion to LR (16 ×16)

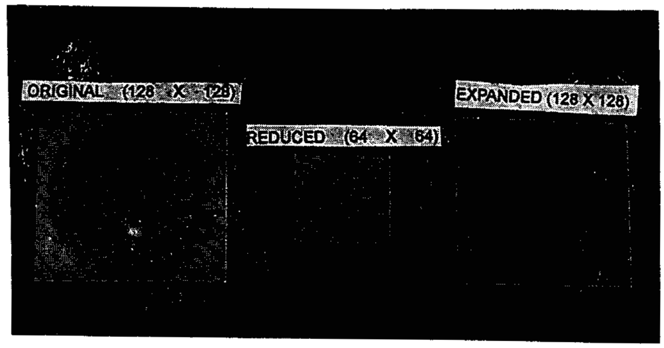

Figure 5. Reduction to level '1' from level '0' and expanding to level '0' (example 1).

correlation was performed using the software. It was repeated several times to validate the system.

For illustration, two photographs with two different scaling factors were included, i.e, one sample converting 224 × 234 image to 16 × 16 (Fig. 3) and the other sample converting 122 × 94 image to 16 × 16 (Fig. 4). The white line rectangle on the HR image showed the size of the image considered for generating the LR image 16 ×16. The resampled image was passed on to LR sensor for registration and it was found that the reference image had registered on the LR sensor image exactly at the specified point. The execution time in each case was computed on the *i*486 processor-based system and the 8086/87-based system[3], respectively. The timings are shown in Table 4. The timings are satisfactorily meeting the project specifications. The resampled image from HOE was transmitted to the LR sensor through a serial link. By applying pyramid processing technique (Eqn 8), the data was compressed to level '1' and transmitted to the LR sensor. At the LR sensor, the data was expanded using Eqn (11) and the correlation performed. It was found that matching took place at the exact point. Level '2' and level '3' reduced and expanded images were also tried and it was found that level '1' is optimum for correlation type of applications. The transmission time at 9600 baud had come down to approx 75 ms by transferring half the size of image.

For illustration, two photographs (Figs 5 and 6) are presented. The level '0' (original) is of 128 × 128 size and level '1' (reduced) is of

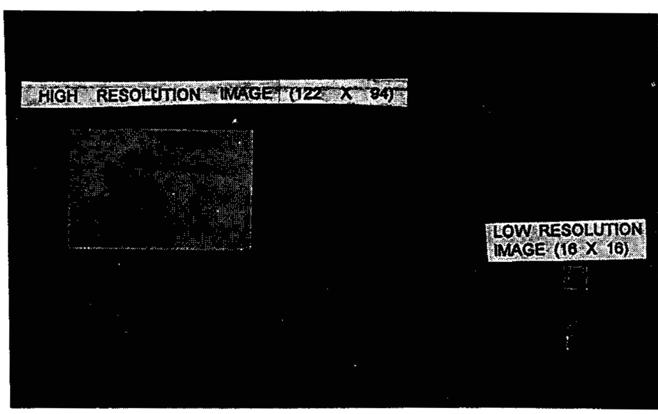

Figure 4. Conversion of HR (122 × 94) to LR (16 × 16)

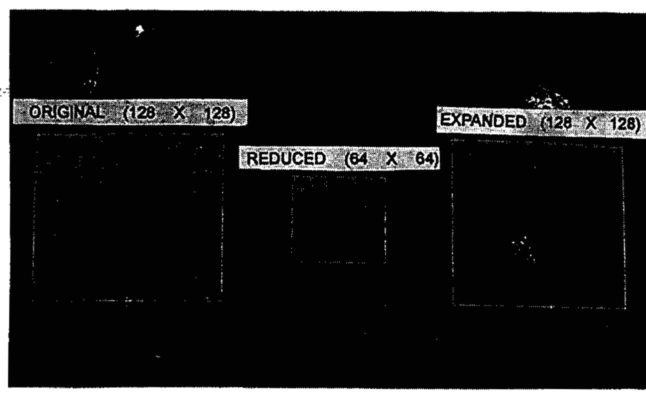

Figure 6. Reduction to level '1' from level '0' and expanding to level '0' (example 2).

Table 4. Test results

Scaling factors	Hardware system		
	Samples	8086/87	*i*486 System
$W_v = 16.89$ $W_h = 15.14$	Image 1	9.36 s	262 ms
$W_v = 5.86$ $W_h = 6.14$	Image 2	0.95 s	36 ms

64 × 64 size and expanded version is of 128 × 128 size.

5. CONCLUSIONS

The HOE unit has been designed with *i*486 processor-based hardware with high speed logic devices. The techniques/methods discussed above have been implemented in the new design both in the hardware and the software. The control logic has been fused in electrically programable logic devices (EPLDs) and firmware in electrically erasable PROMs (E^2PROMs) to keep the security of the design. The software is developed in modular format with lot of BITF. By these techniques, the execution time has been brought down to few tens of milliseconds from few second. The Brut pyramid processing algorithm in its reduced format has been implemented in software to reduce the transmission time considerably for real-time operation.

ACKNOWLEDGEMENTS

The authors wishes to express his sincere gratitude to Lt Gen (Retd) Dr V. J. Sundaram, the then Director, Research Centre Imarat (RCI), Hyderabad, for permitting him to publishing this paper. He also feels deep sense of gratitude to Dr K. Veerabhadra Rao, RCI, Shri N. R. Iyer, Lt Col R. K. Banerjee, DRDL for their kind advice and guidance. He is also thankful to Dr S. B. Gadgil, Sarvashri B. V. Rao, K. Rambabu, K. N. Sekhar and P. V. Ramana, Scientists, RCI, for their useful suggestions at various stages, and to all colleagues who helped directly and indirectly to complete this work.

REFERENCES

1. Boland, J. S.; Pinson, L. J.; Kane, G. R.; Honnell, M.A. & Peters, E.G. Automatic target handoff using correlation techniques. Auburn University, Alabama, 1977. Technical Report. 57-63.
2. Sankar Kishore, M. Image pre-processing system. Paper presented at the International Technical Symposium on Optics & Optical Electronics, Applied Science and Engineering, 6-11 August 1989, San Diego. *SPIE*, **1153**, 140-45.
3. Sankar Kishore, M. Automatic target handing over system. *Def. Sci. J.*, 1994, **44** (1), 69-80.
4. James, H. A. & Roger, F. B. A two-dimensional real-time video pyramid processor. *RCA Review E*, 1986, **47**, 3-31.
5. Peter, J. B. & Edwart, H. A. The Laplacian pyramid as a compact image code. *IEEE Trans. Comput.*, 1983, **31** (4), 512-40.
6. Intel PL/M 86 User's Guide. Intel Literature Department, USA, 1985.

Contributor

Mr M Sankar Kishore obtained his MSc (Tech) Electronics with specialisation in linear digital applications from REC, Warangal and ME (Guidance & Control) from Poona University in 1983 and 1986, respectively. He joined DRDO at the Research Centre Imarat (RCI), Hyderabad, in 1985. His areas of research include image processing and pattern recognition. He is a fellow of IETE; a member of Computer Society of India; Institution of Engineers (IE), India; Aeronautical Society of India and Astronautical Society of India. He has published 14 research papers in national/international journals.