International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 08, August-2015

Image Fusion for Enhanced Vision System using Laplacian Pyramid

Abhilash G, T.V. Rama Murthy
Department of ECE
REVA Institute of Technology and Management
Bangalore-64, India

V. P. S Naidu
MSDF Lab, FMCD,
CSIR-National Aerospace Laboratories (NAL),
PB No.1779, Bangalore, India

***Abstract*— Aviation safety is the primary concern in this era of air travel. As a major step towards improving safety of air transport, a system called Integrated Enhanced and Synthetic Vision System (IESVS) is being developed at NAL, Bangalore. One of the sub-blocks that drive this system is the Enhanced Vision System (EVS). EVS uses visual feedback from two cameras mounted on aircraft nose that deliver real-time video streaming. These video streams are combined or 'Fused' to render an enhanced multi-spectral video stream that will enable the pilot to get a better view of the surroundings during poor visibility conditions. This paper discusses a detailed real-time implementation of the Laplacian Pyramid based fusion algorithm implemented on a C++ platform. The input images are acquired from Charge Coupled Device (CCD) (colour) and Infrared (IR) camera and passed into the algorithm. The captured images are registered using a suitable affine transform. The registered images are used for construction of their respective Laplacian pyramids and the two pyramids are fused. Using the output pyramid structure, the fused image is reconstructed. The base of the fused pyramid is obtained as the output of the Fusion algorithm. This fused image is then rendered using Open graphic library (OpenGL).**

Keywords*— *Laplacian Pyramid, Enhanced vision system, Image fusion, Image processing

I. INTRODUCTION

Images are the most prominent means of communication in our daily lives. This is primarily because the information conveyed by images is highest for the human visual system that is most developed among all our perceived senses. But our vision is still limited only to selected band of frequencies that may be perceived by the naked eye. To overcome this limitation, the process of fusion is used that allows the means to visualize a scene via multiple spectra that is physically impossible with the naked eye [1].

As a means of performing this process of fusing multiple images, the more common approaches used may be either pixel based or frequency based. While the pixel based approach performs fusion using the pixel data directly, the frequency based method requires the generation of the spectral data using which fusion may be carried out. While the latter is a more preferred method as it offers higher precision in the spectral content, the former allows faster implementations and with no additional hardware requirement.

The primary causes of mishaps during air travel are poor visibility during take-off and landing. As solutions many systems have been developed. One such initiative by NASA [2] called the Synthetic Vision System (SVS) project uses innovative synthetic image generation based on image processing and graphics to eliminate aircraft accidents caused by inadequate vision. To augment the SVS project, an Enhanced Vision System (EVS) is used to provide real-time feedback from a suite of sensors to the pilot. The sensors are used to provide more information about the surrounding, especially during poor visibility conditions such as rain, snow, fog or haze. NAL is currently developing an indigenous system called Integrated Enhanced and Synthetic Vision System (IESVS) that has EVS as one of its primary functional units. This paper covers a preliminary fusion algorithm implementation for the EVS developed in the laboratory level, prior to its testing on an aircraft. One of the primary functions of the EVS is to capture real-time video stream data from multi spectral sensors, which are then fused to obtain a multi-spectral fused video stream. However, its effectiveness is only validated, when the fused image is rendered at a minimum frame rate of 15 fps (frames per second).

The EVS application developed uses data input from two on-board sensors, which in this case are the FLIR (Forward Looking Infra-Red), and CCD (Charge Coupled Device) sensors. The data stream obtained from the two sensors are captured from a frame grabber and passed into the algorithm. The C++ based fusion algorithm developed uses this data and performs fusion operation.

Section II covers the overview of the EVS, the parts of the EVS that is being developed in this application and its various blocks. Section III covers the Laplacian and Gaussian pyramids, also the fusion algorithms implementation is discussed in detail. Section IV covers the overview of the implementation and the results obtained.

II. EVS DEVELOPMENT

The EVS developed and referred to in [2] uses a combination of two infrared sensors and a visible sensor for data acquisition. In the present implementation only a single FLIR sensor and a CCD (color) camera have been used. The specifications of the hardware are detailed in section IV. Following this image acquisition hardware called frame grabber is used. Then an image registration process is performed on the FLIR camera to adjust its FOV (Field-of-View) with that of the CCD camera. Then the two images are captured and fused.

A. EVS

The basic scheme of EVS is as shown in Figure 1.The EVS scheme receives the input from the two vision sensors (cameras). Each input, an image frame is captured and passed by the frame grabber. The registration block changes the FOV of the FLIR image, i.e. it register's the FLIR image. This is a process where in each pixel in the FLIR is mapped to the pixels in the color camera. This mapping is performed by Affine transform whose computation discussed in section III. The registered images are then passed as input into the fusion algorithm. Prior to fusion the Laplacian and Gaussian pyramids of each image are generated and these are used to generate the fused image. Following fusion, the fused image which is a gray scale image is displayed by a basic rendering program developed using OpenGL.

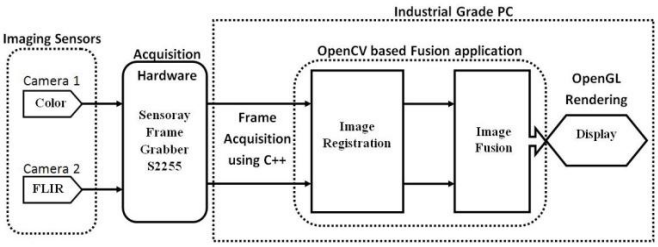

Figure 1. Basic blocks of EVS scheme

B. Hardware & Software

The overview of the various hardware and software used for the implementation is as follows:

1) The PC being used for the development of this application uses an Intel core i7 processor. It has a 3GB DDR3 RAM and a 512MB Nvidia GeForce 8400GS graphics accelerator. The system uses Windows XP Professional edition operating system.

2) Frame grabber is a Model 2255 USB video capture device.

3) All the programming, debugging and testing is being carried out using Microsoft Visual Studio 2008.

4) OpenCV version 2.4 is being used in the project for the following purposes:

- Defining the basic image data type for image capture.
- Developing the functions for pyramid generation.
- Developing the function for image registration and fusion.

OpenCV functions being pre-tested fast are used to enable faster implementation. Hence most image processing operations have been implemented using OpenCV.

5) OpenGL has been specified by the aircraft industry as a standard for rendering. Hence a simple texture rendering scheme is used to display the captured image using OpenGL.

6) Acquisition sensors used are the CCD camera and FLIR camera. The vision sensor specifications are provided in Table-1. Both sensors are powered by 12V DC source. The power supply unit is housed along with the Frame grabber module that provides the 12V DC supply. CCD detects 0.4-0.78µm band and it can capture runway markings, skyline, and city lights in good visibility conditions. FLIR senses radiation in 7.5-14µm band and it can capture background scenery, terrain features, and obstacles at night and in other low visibility conditions.

7) MATLAB has been used to calculate the affine transform used for image registration. A basic code has been developed using MATLAB inbuilt registration tool *cpselect*. Also a preliminary testing of the pyramid based fusion function was carried out in MATLAB.

Table 1. VISION SENSOR SPECIFICATIONS

Specification	CCD	FLIR
Sensor type	HDR 100dB CMOS1/4'' 6μsqm pixel	FPA, uncooled micro bolometer
Lens	6.8mm M12x0.5 S-Mount	19mm
Interface	Analog NTSC	
FOV	36ºH x 27ºV	
Format	640 480	

C. Image Acquisition

As described earlier image acquisition is carried out using Sensoray frame grabber Model S2255. Using two analog channels the captured image frames are transferred in BGR ordered standard windows bmp format. The captured frames are transferred a high-speed USB 2.0 interface to the PC. It is powered through the USB port. The image capture function is developed by the Sensory proprietary software development kit (SDK). The function for this is coded in C++. The images are captured at a resolution of 480x640 by the frame grabber. The function is developed such that is transfers the captured image into OpenCV 'Mat' – type data, so that all future image processing may be carried out using OpenCV.

The manufacture specification [4] of the hardware used states that frame capture may be carried out at a rate of 30 fps while capturing from 2 analog channels simultaneously.

D. Image Registration

Image registration is defined as the process of transforming different sets of data into one coordinate system. Data may be multiple photographs, data from different sensors, from different times, or from different viewpoints. Registration is necessary in order to be able to compare or integrate the data obtained from these different measurements [5]. The simplest form of image registration is the one were in control points are selected from the two images and based on the point positions they are transformed. The transformation operation here is basically a combination of horizontal and vertical scaling, shearing, transverse and longitudinal rotations that are applied to each obtained image.

Since the sensors (cameras) are at fixed distance and are housed in a common mount the transform that performs registrations needs to be computed only once. Following this all subsequent images captured are subject to this pre-defined transformation and registered.

MATLAB provides a registration tool called cpselect that provides a GUI where the images to be fused are accepted and displayed. By means of a point selection tool the reference points may be selected and saved into the workspace. Following this the cp2transform function may be used to generate the affine transform for the registration of the two captured frames. The generated transform may be used throughout the registration process thereafter.

MATLAB usage of cpselect is carried out as follow:

1) Using cpselect tool the points that are identifiable in the two images are as marked as shown in Figure 2.
2) From this the affine transform is generated.
3) Finally the transformed image is as shown in Figure 3.
4) Therefore to verify the registration process the two images are superimposed as in Figure 4.

Save the affine transform matrix obtained in the MATLAB *tform* structure

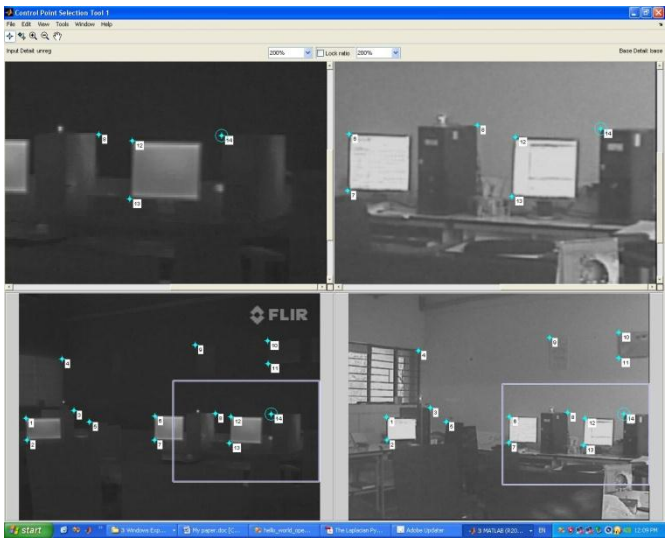

Figure 2. Usage of *cpselect* tool to mark reference points in the two images

Figure 3. Image after registration

It undergoes the required translation and scaling (in this case rotation is not required since both vision sensors are fixed on the same platform). From the registration process, the obtained affine matrix is stored in a MATLAB *tform* structure.

The value of the affine matrix is:

$$\boldsymbol{T} = \begin{bmatrix} 0.9721 & 0.0079 & 0 \\ 0.0112 & 1.1125 & 0 \\ 59.9345 & -32.7049 & 1 \end{bmatrix} \quad (1)$$

Using this affine matrix one of the input images (unregistered) have to be registered to the other (base image). In this case and the rest of the paper, the IR image is registered using the CCD image as the base image. During the fused video stream generation also this same scheme is used.

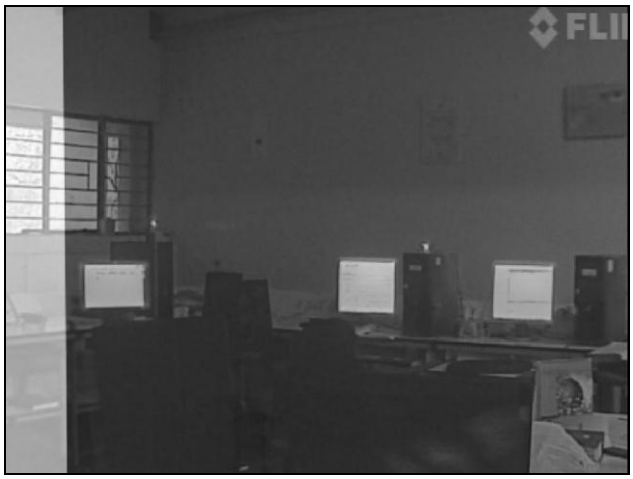

Figure 4. Overlapping of Registered and base image

III. IMAGE FUSION

The requirement of both high spatial and high spectral information in a single image is achieved using an image fusion operation. In general, the problem that image fusion tries to solve is to combine information from several images (sensors) taken from the same scene in order to achieve a new fused image, which contains the best information coming from the original images. Hence, the fused image has better quality in terms of data content than any of the original images.

In the fusion process stated in [5], the fusion is based on a wavelet-based algorithm. Though this was effectively implemented it was stated as being slow as the wavelets needed to be generated and the execution speed of the process was causing a serious overhead in the frame rate of the final video stream. As a solution to this problem, the present application makes an effort to reduce the overhead by performing a time-domain based fusion algorithm using a simple Laplacian pyramid [6] technique for image fusion.

A. Gaussian Pyramid

The generation of the Laplacian pyramid is carried out using Burt and Adleson's [6] technique. This is a well established image pyramid generation technique which is commonly used in Laplacian and Gaussian pyramid

generation. The Laplacian pyramid is obtained from the Gaussian pyramid. Therefore, to explain the former the latter is essential. The Gaussian pyramid may be generated in two directions. If the pyramid is generated from base to peak, the operation is called REDUCE. If it is generated from peak to base it is called EXPAND. Burt et al. [6] discuss both theses operations in detail. Also both these methods have been implemented as function in both MATLAB and OpenCV as inbuilt functions. Using these operations of expansion and reduction the pyramid generation is carried out. The reduction [6,7] operation for obtaining the kth level image gk of dimension (Ck,Rk) from the (k-1)th level image of dimension (Ck-1,Rk-1) is defined for a Gaussian pyramid is as follows:

$$g_k = R(g_{k-1}) \tag{2}$$

$$g_k(i,j) = \sum_{m=-2}^{2}\sum_{n=-2}^{2} w(m,n)\, g_{k-1}(2i+m, 2j+n) \tag{3}$$

where, $w(m,n) = \hat{w}(m)\hat{w}(n)$ is the one-dimensional kernel that has been (in this case) chosen as of length 5.

$$\hat{w}(0) = a, \tag{4}$$

$$\hat{w}(-1) = \hat{w}(1) = 0.25, \tag{5}$$

$$\hat{w}(-2) = \hat{w}(2) = 0.25 - 0.5a, \tag{6}$$

$$a = 0.375 \tag{7}$$

Similarly the expansion operation allows the generation of the (k-1)th level image gk-1 of size (Ck-1,Rk-1)from the kth level image gk as,

$$g_{k-1} = E(g_k) \tag{8}$$

$$g_{k-1}(i,j) = 4\sum_{m=-2}^{2}\sum_{n=-2}^{2} w(m,n)\, g_k\left(\frac{i-m}{2}, \frac{j-n}{2}\right) \tag{9}$$

Only for terms in which (i-m)/2 and (j-n)/2 are integers are included in this sum. This is defined for $1 \le i \le Ck$ and $1 \le j \le Rk$ where (Ck,Rk) represents the dimension of the image at the kth stage. These equations are the basic equations for expansion and reduction operations. Using these expansion operations repeatedly, the pyramid may be generated for any given image in two different directions. While the expansion operation when repeated, generates a new image whose dimension is double the dimension of the input image, the reduction operation generates an image that is half the size of the input image. An example of a Gaussian pyramid generated using MATLAB impyramid instruction is as shown in Figure 5.

B. Laplacian Pyramid

Generating the Laplacian pyramid is a process that again uses the Gaussian pyramid operations of expansion and reduction. By definition, the Laplacian pyramid is a sequence of error images $L_0, L_1, \ldots, L_N$. Each is the difference between two consecutive levels of the Gaussian pyramid [6,7]. The Laplacian pyramid generation is defined for $0 \le k \le N$,

$$L_{f,k} = g_{f,k} - E(g_{f,k+1}) \tag{10}$$

This step allows the construction of the Laplacian image from base upwards, i.e. from the largest image to the smallest image.

C. Laplacian Pyramid Based Fusion Algorithm

The Image fusion algorithm implemented is based on the generated pyramid algorithm. Hence in the image fusion operation carried out multiple pyramids need to be generated. This pyramid generation was carried out using OpenCV for the developed application. The OpenCV instructions *PyrUp* and *PyrDown* are used to perform the REDUCE and EXPAND operations respectively. The flowchart of the pyramid generation for the inputs and the outputs are shown in Figure 6.

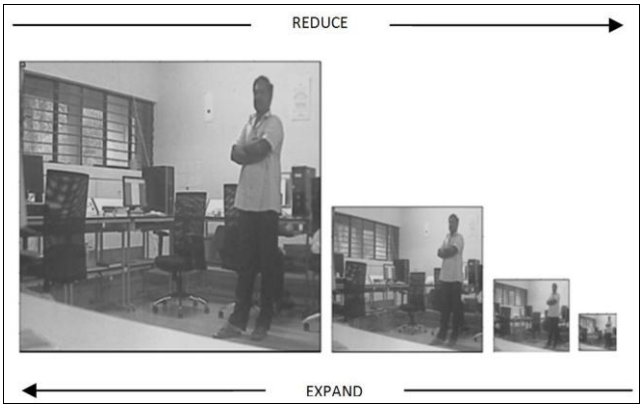

Figure 5. Pyramid construction in two directions

Using this as reference the pyramid generation algorithm used in this paper may be described as follows:

Step 1. The two input images from the IR and the CCD cameras are converted into gray-scale images and these are passed into the fusion algorithm as the base images gc,0 and gir,0of the input Gaussian pyramids in Figure 6.

Step 2. Using these inputs consecutive REDUCE operations {R(gm,n)} are used to obtain the corresponding Gaussian pyramids generated for k+1 levels with the zeroth level being the input image.

Step 3. Following this the Gaussian pyramids are used to generate the corresponding Laplacian pyramids based on the same steps as shown in Figure 6. Here the Laplacian image is Lm,n. As the Gaussian Pyramid has (k+1) levels the Laplacian pyramid will have k levels.

Step 4. The variation of this algorithm begins here. In the output side the Laplacian pyramid is generated first. The output Laplacian pixel values within each Laplacian image is the pixel value with its magnitude being the maximum among the two corresponding pixels in the two inputs at the same stage, i.e. for every kth stage of the Laplacian pyramid each pixel is computed as

$$L_{f,k}(i,j) = sign_k(i,j) \times \max\left(\left|L_{c,k}(i,j)\right|, \left|L_{ir,k}(i,j)\right|\right) \tag{11}$$

$$sign_k(i,j) = \begin{cases} \mathrm{sgn}(L_{c,k}(i,j)), \left|L_{c,k}(i,j)\right| > \left|L_{ir,k}(i,j)\right| \\ \mathrm{sgn}(L_{ir,k}(i,j)), \left|L_{c,k}(i,j)\right| < \left|L_{ir,k}(i,j)\right| \end{cases} \tag{12}$$

$$\mathrm{sgn}(L_{m,l}(i,j)) = \begin{cases} -1, L_{m,l}(i,j) > 0 \\ +1, L_{m,l}(i,j) < 0 \end{cases} \tag{13}$$

This operation is shown in Figure 6

Step 5. To generate the Gaussian pyramid at the output side, the $(k+1)^{th}$ level Gaussian of the output is obtained. This image is the average of the Gaussian images of the $(k+1)^{th}$ level of the two input Gaussian pyramids. Thus the $(k+1)^{th}$ output Gaussian image is give by,

$$g_{f,k}(i,j) = \frac{g_{c,k}(i,j) + g_{ir,k}(i,j)}{2} \quad (14)$$

Step 6. This $(k+1)^{th}$ Gaussian image is used along with the k-levels of Laplacian pyramid to generate the Gaussian pyramid of the fused image. The k^{th} stage of the Gaussian pyramid is obtained as $g_{f,k} = L_{f,k} + E(g_{f,k+1})$ (15)

Step 7. The fused output image is obtained as the zero-level image $I_{0,f}$ of the output Gaussian pyramid.

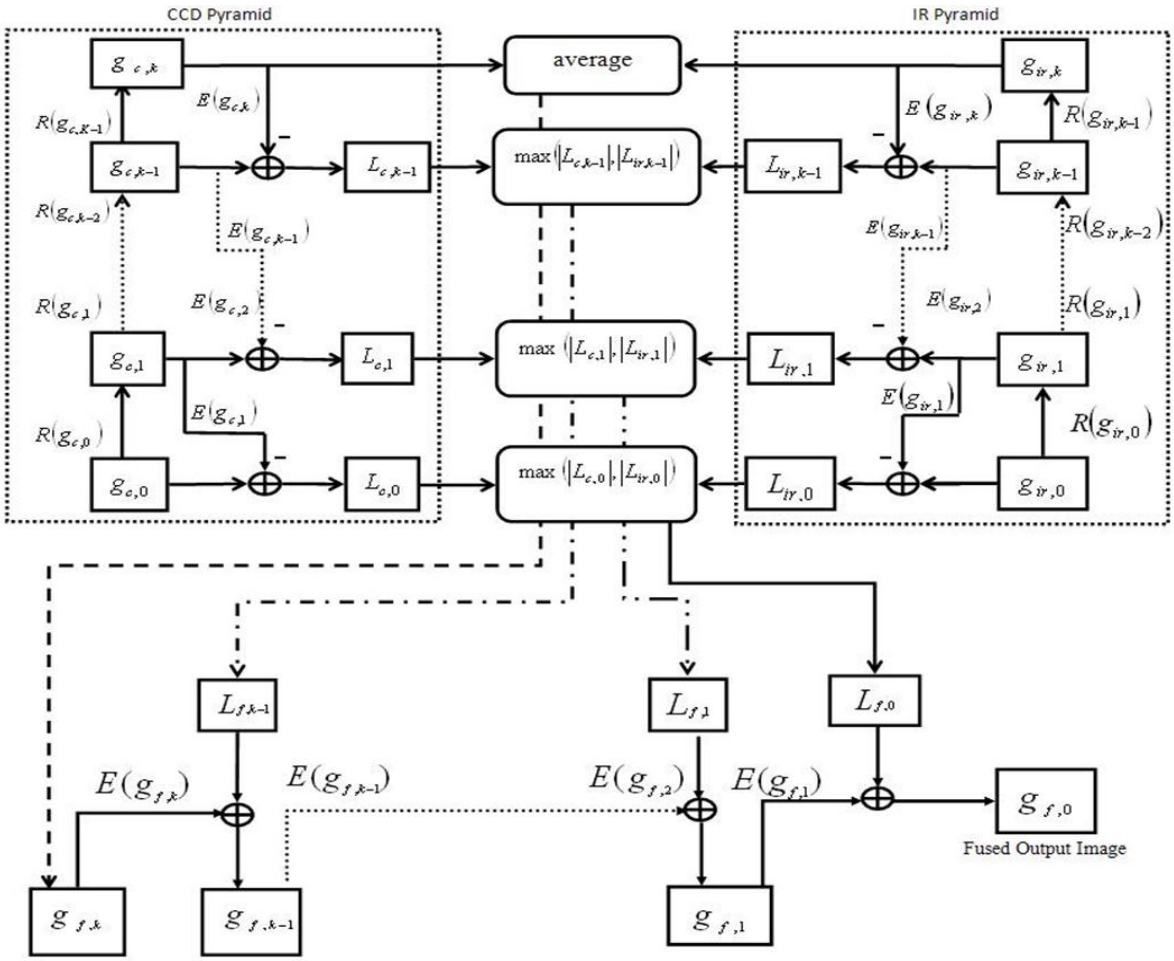

Figure 6. The Flowchart of the Laplacian Pyramid Based Fusion Algorithm

D. Implementation of EVS application based on C++

The application designed for the testing of EVS is based on the system proposed in Figure 1. This application is developed on C++ platform. The image acquisition is carried out using a C++ object for the S2255 frame grabber. The acquired image is then passed into an OpenCV based function that performs the registration operation on the IR image treating the CCD image as the base image. These frames are then passed into the OpenCV based object that performs the fusion operation. This object creates the required 3 pyramids by assigning every acquired image frame to the base of the IR and colour (CCD) pyramids. Following this the pyramids are generated and fusion is carried out as defined by the algorithm described in the previous section. As described in Step 7, the fused image is obtained as the base of the fused image pyramid. The current application uses a rendering based on OpenGL. Figure 7 shows the snapshot of the application

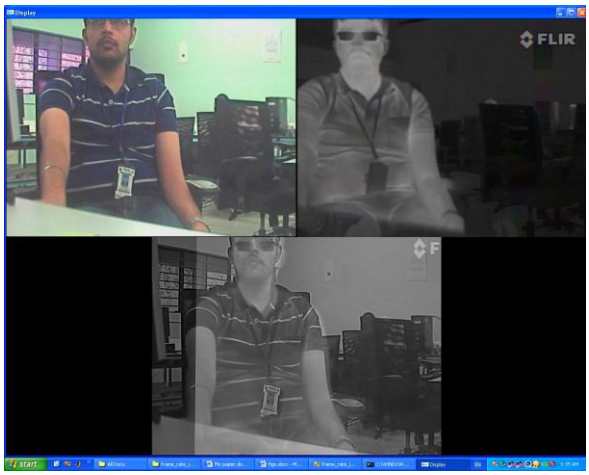

Figure 7. Screenshot of the Fusion Application rendered using OpenGL

E. Fusion Quality Evaluation Metrics

Since this is a primary implementation, the algorithm has not been compared with other algorithms for performance evaluation. But as a purpose of record, the fusion quality evaluation metrics have been evaluated. The used metrics from [8] are the following:

1. Standard Deviation: Standard deviation measures the contrast in the fused image. An image with high contrast would have a high standard deviation. The expression for standard deviation is:

$$\sigma = \sqrt{\sum_{i=0}^{L}(i-\bar{i})\cdot h_{I_f}(i)}, \bar{i} = \sum_{i=0}^{L} i h_{I_f}(i) \quad (16)$$

Where $h_{I_f}(i)$ is the normalized histogram of the fused image $I_f(x,y)$ and L number of frequency bins in the histogram.

2. Entropy: This is used to measure information content of an image. Entropy is sensitive to noise and other unwanted fluctuations. An image with high information content would have high entropy. Using the entropy, the information content of a fused image is:

$$H_e = -\sum_{i=0}^{L} h_{I_f}(i)\log_2 h_{I_f}(i) \quad (17)$$

3. Spatial Frequency: The frequency in spatial domain indicates the overall activity level in the fused image. This is computed as:

$$SF = \sqrt{RF^2 + CF^2} \quad (18)$$

Where,
Row frequency of the image,

$$RF = \sqrt{\frac{1}{MN}\sum_{x=1}^{M}\sum_{y=2}^{N}\left[I_f(x,y) - I_f(x,y-1)\right]^2} \quad (19)$$

Column frequency of the image,

$$CF = \sqrt{\frac{1}{MN}\sum_{x=1}^{M}\sum_{y=2}^{N}\left[I_f(x,y) - I_f(x-1,y)\right]^2} \quad (20)$$

4. The obtained metrics are shown in Table 2. From this it is inferred that the fused image has good contrast definition information. The high entropy indicates the contained information content of the image

Table 2. Fusion Quality Evaluation Metrics

Metric	Value
Standard Deviation (σ)	40.320
Entropy (H_e)	7.0228
Spatial Frequency (SF)	7.6717

IV. CONCLUSION

Laplacian pyramid based image fusion algorithm has been implemented and evaluated on MATLAB for EVS application. Real time video acquisition from IR and CCD vision cameras and fusion has been done on a C++ platform. OpenGL has textures have been used to render the fused image for display purpose. The algorithm is able to render the images around 33 fps, which is more than the expected objective for EVS application. Therefore, the objective of developing a real-time implementation to test the functionality of the Laplacian Pyramid Fusion algorithm has been achieved.

ACKNOWLEDGMENT

We thank Dr. Girija G., Head FMCD, Sri N. Shanthakumar, Head MSDF Group from CSIR-National Aerospace Laboratories, Bangalore for helping us to get through many a hurdle at various stages of the project. We also thank Dr. S.S. Manvi, HOD, Dept. of ECE from Reva ITM, Bangalore for support.

REFERENCES

[1] Gonzalo Pajares and Jesus Manuel de la Cruz, "*A Wavelet-based Image Fusion Tutorial,"* Pattern Recognition, Vol.37, pp.1855-1872, 2004
[2] http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20050160197_2005160937.pdf, Dated 25th Feb, 2013
[3] http://www.sensoray.com/products/2255.html, Dated 25th Feb, 2013.
[4] Barbara Zitova and Jan Flusser, "*Image registration methods: a survey*," Image and vision computing, Vol.21, pp.977-1000, 2003.
[5] VPS Naidu, NarayanaRao P., Sudesh K Kashyap, Shanthakumar N. and Girija G., *"Experimental Study with Enhanced Vision SystemPrototypeUnit,"* MSDF LAB, FMCD, CSIR – National Aerospace Laboratories(Bangalore) - 2011 International Conference on Image Information Processing (ICIIP 2011).
[6] Burt, P., Adelson, E., "*The Laplacian Pyramid as a Compact Image Code*," *Communications, IEEE Transactions on* , vol.31, no.4, pp. 532-540, Apr 1983.
[7] VPS Naidu, "*A Novel Image Fusion Technique using DCT based Laplacian Pyramid*", MSDF Report No.: 1107/IESVS01, 27th May 2011
[8] VPS Naidu and J.R. Raol, *"Pixel-Level Image Fusion using Wavelets and Principal Component Analysis – A Comparative Analysis"*Defence Science Journal, Vol.58, No.3, pp.338-352, May 2008.