Hardware realization of video enhancement and compression algorithms

Harish Reddy Kolanu
Department of ECE,
NIT Rourkela
Email: harish9446@gmail.com

Priyadarshini Nagrale
ARDE, DRDO
Pune, India.
Email: pnagrale@arde.drdo.in

Manish Okade
Department of ECE,
NIT Rourkela
Email: okadem@nitrkl.ac.in

Kamalakanta Mahapatra
Department of ECE,
NIT Rourkela
Email: kkm@nitrkl.ac.in

***Abstract*—This paper presents implementation aspects of video enhancement and compression algorithms on hardware platforms. The implementation is motivated by the fact that sufficient studies have not been carried out earlier on hardware realization of video algorithms since major focus in the research community has been on analyzing algorithms from a software perspective and many a times ignoring the hardware board level aspects which are very critical from the defence point of view. This paper is a novel attempt to bridge this gap and present the practical considerations of video processing algorithms when run on hardware boards. We utilize both a general purpose board namely Raspberry Pi 3 as well as a dedicated video processing board namely TI's DaVinci DM6437 to show the performance of existing video algorithms so that when prototype development is carried out the designer could tradeoff performance and cost aspects depending on the application scenario. Primary focus in this work is on the utility of video processing algorithms for surveillance applications like UAV's and missiles.**

***Index Terms*—video enhancement, video compression, hardware boards, Raspberry Pi, TI DaVinci.**

I. Introduction

The domain of onboard video processing utilizing dedicated hardware evaluation boards is less established due to two major reasons, firstly, such processing needs qualified personnel for operating and programming the video processing boards and secondly the cost consideration of such video processing modules are on the higher side. However, with increasing benefits being offered by such on-board processing more and more effort is slowly and steadily being directed towards embedded hardware design and development for video processing applications. This paper is motivated on the premise that video acquisition, enhancement and compression would be targeted under the challenging constraints of achieving fast computational time and final deployment to be miniaturized to fit small diameter spaces like missiles/rockets and other Unmanned Aerial Vehicles (UAV's). One such application being targeted in the present work is to utilize low resolution cameras (QQVGA) and estimate the lowest possible compression size for each acquired video frame so that it can be transferred asynchronously via UART to the host controller. Before compressing few enhancement algorithms also would be applied on the acquired video frames so that it will be useful for extracting information from the acquired video frames. In the present work, the impact of achieving twin design constraints namely fast computations along with being deployable in small spaces for video enhancement and compression applications is studied. This is carried out by firstly investigating prototype development using video processing hardware boards. Prior to this a brief review of state-of-the-art methods in this domain is presented below.

Video acquisition along with compression of acquired frames has been investigated earlier by many researchers [1]–[4]. Majority of existing scenarios look at this aspect from the post processing point of view where considerable resources like computing power/memory in terms of high end servers are easily available. However, many application scenarios require this process to happen in real time that too in resource constrained environments where size and computing power are scarce. Only a handful of work have been carried out in this area with researchers mainly focusing on utilizing video processing boards for such purposes [5]–[10]. Texas Instruments Da-Vinci family [9], [10] has been very popular in this domain and this video development platform has been previously utilized for applications ranging from compression, target detection and tracking, face recognition to name a few. In [9], a video acquisition and compression codec system was designed using TMS320DM6446 with good results. However, it did not consider the enhancement aspects of the acquired frames and its performance was not purely real time as they did not strictly carry out the detailed timing analysis. Moreover, further FPGA implementation and porting aspects also was not in their objectives and design goals. Da-Vinci board was also previously utilized in speech data analysis by way of implementation of G.729 codec for audio applications [7] namely Voice over IP (VoIP). Moving target detection and tracking using Da-Vinci technology was utilized for implementation of intelligent security based video surveillance system in [6]. Similar work was also carried out simultaneously by Zhao and Jiang [8]. Implementation of H.264 decoder accounting for DMA resource and cache optimization for efficient memory access was explored in [7]. With this review of the state-of-the-art it can be concluded that the Da-Vinci video processing family is best fit for the proposed application being dealt in the current work. The choice of Da-Vinci family for prototype development is motivated by the fact that it can balance effectively the performance, power and cost considerations. However, to validate the claims made in this paper we also use a general purpose hardware board namely Raspberry Pi 3

and show the performance tradeoffs of the video processing algorithms.

The contributions made in this paper are listed below;

1) Firstly, a comprehensive realization of video enhancement and compression algorithms on hardware boards namely Raspberry Pi 3 and TI DaVinci DM6437 along with subjective and objective evaluations benchmarking the hardware study based on performance aspects.
2) Secondly, inferences are drawn from the study with regard to the choice of the hardware board based on the suitability of the application at hand i.e. video enhancement and compression for UAV's, missiles etc. This gives an overall assessment from the defence point of view since its applicability would be for real time video processing.

II. Hardware Board Details

In this section we provide a brief description of the Hardware boards used in the study.

A. Raspberry Pi 3

The internal block diagram of Raspberry Pi 3 is shown in Fig. 1. The Raspberry Pi 3 is a mini size Linux computer widely utilized in automation and IoT applications due to its ease of programming as well as its architecture supporting many complex tasks. In this study we utilize the Raspberry Pi 3 which has the following specifications and priced roughly at Rs. 3000 ($ 40); ARM Cortex-A53 CPU 4 x 1.2GHz, Broadcom VideoCore IV GPU, LPDDR2 RAM of size 1GB at frequency 900 MHz, Broadcom BCM2837, SOC Ethernet which supports 10/100, wireless wifi support 2.4GHz, IEEE 802.11n standard, BLE 4.1 Classic, micro SD card support for storage, GPIO header of size 40 pins, HDMI video output port, USB 2.0 support, Camera Interface (CSI) and Display Interface (DSI).

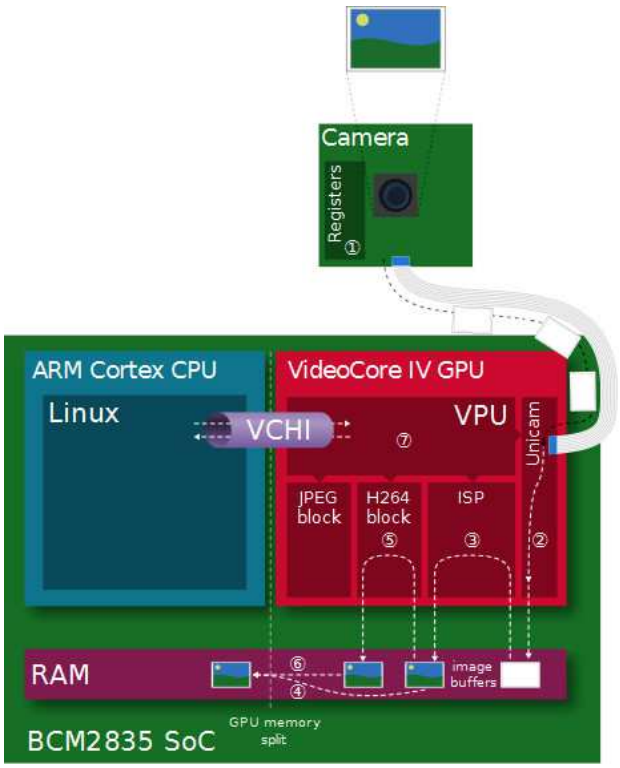

Fig. 1. Block diagram of Raspberry Pi 3.

B. Texas Instrument (TI) DaVinci DM6437

TI DaVinci DM6437 is a dedicated audio/video processing hardware board utilized as an development platform for audio and video processing algorithms and architectures. In this study we have utilized TI DaVinci DM6437 as shown in Fig. 2 which has the following specifications and priced roughly at Rs. 84,000 ($ 1146); The CPU is od TMS320DM6437 DSP which has a clock speed of 600 MHz, TVP5146M2 onboard Video Decoder with video DAC outputs, DDR2 DRAM of size 128MB, CAN and UART for I/O interfaces, Flash memory of 16M, NAND Flash of size 64MB, SDRAM of size 2MB, stereo audio codec of standard AIC33, IC for EEPROM and expanders Ethernet Interface which supports 10/100 JTAG for debugging, single DC supply voltage +5V. DM6437 video processing subsystem consists of front end and back end subsystems. The front end accepts RAW input from the camera source. The back end system consists of display drivers, video encoder, and audio driver. Video compression is done at the back end system and can be displayed to the external display unit via NTSC or PAL format.

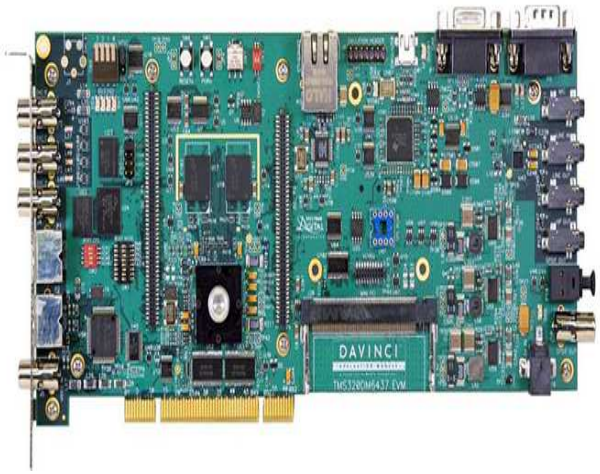

Fig. 2. Texas Instrument (TI) DaVinci DM6437.

III. Proposed Implementation

This section outlines the implementation aspects of two video processing tasks namely enhancement and compression on hardware boards. Two different hardware boards are chosen, the first one a general purpose hardware platform i.e. Raspberry Pi 3 which is re-purposed to execute video processing tasks and the second one a dedicated video processing hardware board i.e. TI DaVinci DM6437. The motivation for the choice of the boards is to present the different performance aspects of the two hardware boards so that it will provide

implementation considerations when video tasks are executed on hardware so that for real time surveillance tasks one could tradeoff the performance aspects.

A. Video Enhancement

In this sub-section the video enhancement algorithm utilized along with how it is executed on two hardware boards i.e. the Raspberry Pi 3 and the TI DaVinci DM6437 are outlined. Firstly, QQVGA resolution (160×120) video is acquired utilizing a camera and this video is in .raw (uncompressed) format. The acquisition is directly interfaced with the two hardware boards via the USB port. The first step is to carry out the pre-processing of the acquired frames which is done using histogram matching technique. This method which is also known as histogram specification is an image enhancement technique wherein a target histogram is specified and the input image under analysis is processed so that its histogram matches with the target histogram. This technique is better than histogram equalization since in equalization the histogram is converted to uniform histogram irrespective of the image content. However, since the acquired video frames in the defence scenarios are in outdoor environment (UAV's and missiles), every time having uniform probability density function may not necessarily enhance the video content. Moreover, the histogram matching technique is particularly suitable in the present scenario since different video frames will require varied amount of enhancement depending upon acquisition conditions since the application is for UAVs and missiles which are fit with an onboard camera. Additionally, some amount of blurring will also be present since the UAV would be in motion whilst capturing the video frames and histogram matching technique can effectively counter this to a large extent which saves explicit utilization of video restoration methods. These are the primary reasons for the choice of histogram matching for the purpose at hand.

Mathematically, histogram matching [11], [12] is given as follows. Let 'r' and 'z' denote the intensity levels of input and enhanced images respectively. Then $p(r)$ and $p(z)$ will denote the respective probability density functions. Now let 's' be a random variable which is given as

$$s = T(r) = (L-1)\int_0^r p(r)dw \tag{1}$$

where, 'w' is a dummy variable of integration. Now, we define a random variable 'z' with the property

$$G(z) = (L-1)\int_0^z p_z(t)dt = s \tag{2}$$

where, 't' is a dummy variable of integration. It then follows from these two equations that $G(z) = T(r)$ and therefore 'z' must satisfy the condition

$$z = G^{-1}[T(r)] = G^{-1}(s) \tag{3}$$

Eq. (1), (2) and (3) demonstrate that a video frame whose intensity levels have a specified probability density function can be obtained from a given frame and accordingly the enhancement can be carried out effectively.

Raspberry Pi 3: The histogram matching algorithm described above is implemented on Raspberry Pi 3 hardware board using video sequences which are in raw YUV format. The algorithm is run frame by frame and its performance aspects are noted whose details are presented in the experimental results section where the frame resolution utilized, timing analysis, enhancement aspects both subjective and objective are discussed and important inferences are drawn.

TI DaVinci DM6437: The same histogram matching algorithm is also run on TI DaVinci DM6437 board utilizing the same video sequences to maintain uniformity in comparison. As expected the enhancement quality is similar to that achieved by Raspberry Pi 3 since the algorithm is same. However, there is significant difference in the execution times for the histogram matching algorithm on the DaVinci board due to it being a dedicated video processing board. For detailed analysis the readers are referred to the experimental results section.

B. Video Compression

On similar lines in this subsection the video compression carried out on the two hardware boards is described. H.264 [13] which is an industry standard for achieving video compression is utilized to achieve the compression. A brief description of the H.264 pipeline is provided for completeness. The H.264 encoder block diagram is shown in Fig. 3. The first frame will go through Transform and Quantize cycle (Intra) while the second frame onwards motion estimation will be carried out to generate the motion vectors which will be entropy coded. Inter frame prediction will be carried out to generate the difference frame which will also be entropy coded. The compressed video will be in .264 format. The acquired video via the embedded camera is in .raw format (uncompressed) and has already undergone enhancement utilizing histogram matching as described in the previous section. The task in hand is to carry out compression which is performed using the H.264 encoder, so that the compressed videos can be sent to the ground base station. The hardware realization on the two boards is described below;

Raspberry Pi 3: Raw video sequence is acquired using pi-camera module and passed through video core present on the Raspberry Pi 3. The front end is developed using nodejs and expressjs while the Raspberry Pi 3 is programmed using python with multi-process approach utilizing queues to store input frames in the buffer. The resolution of the video being acquired along with its frame rate and video capture duration can be set by the user and the recording can be started. Finally the compressed .264 video can be played by any front end media player. The encoded video is then sent to the host PC via UART where the compressed video is saved to the local disk depicting a real scenario wherein the UAV which has this

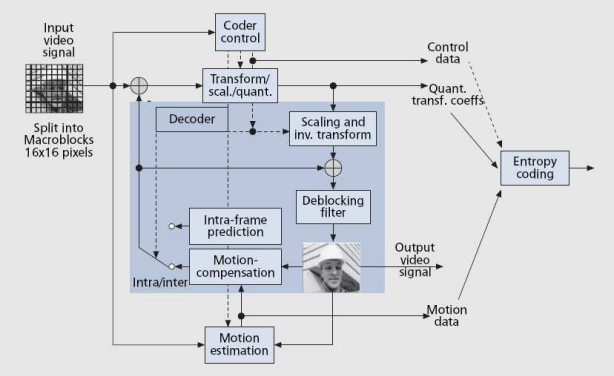

Fig. 3. H.264 encoder block diagram.

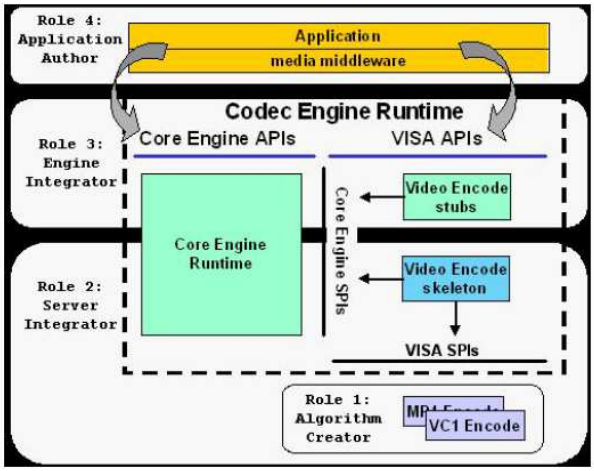

Fig. 4. TI DaVinci Codec block diagram.

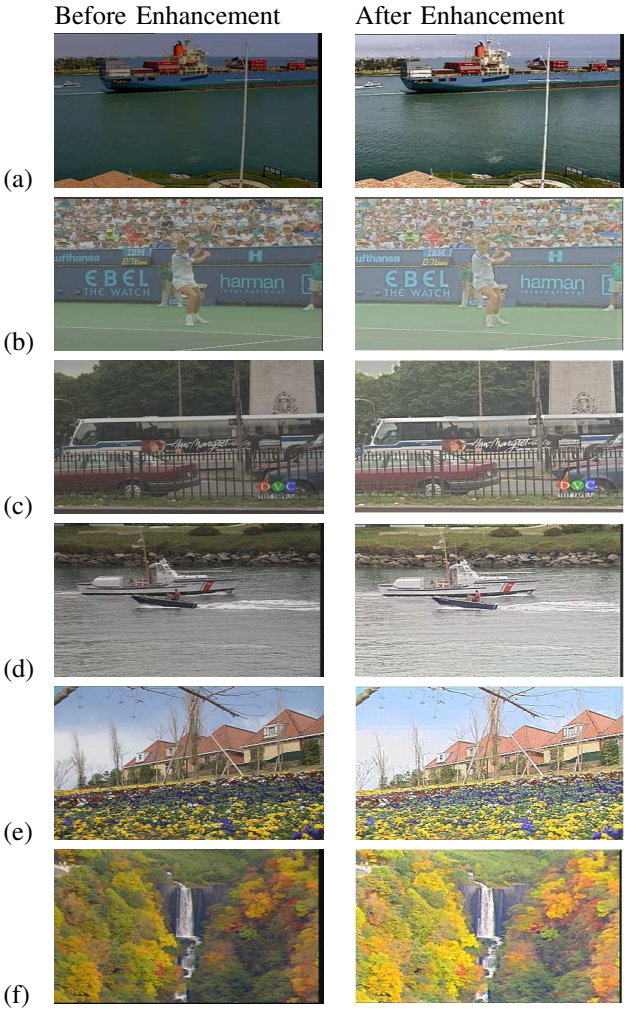

Fig. 5. Qualitative i.e. subjective comparative analysis using histogram matching run on the two hardware boards for the following video sequences: (a) Container (b) Stefan (c) Bus (d) Coastguard (e) Garden (f) Waterfall.

onboard hardware board can send the compressed video to the base station.

TI DaVinci DM6437: DaVinci supports H.264 codec and its other optimizations like compression ratio tuning, deblocking filter tuning etc. DaVinci DM6437 board supports videos in 4:2:2 format. However, standard video sequences available online are in 4:2:0 format and hence they need to be converted to 4:2:2 format which is carried out using the FFmpeg library which supports formats from legacy video formats to the latest cutting edge formats. Once converted to 4:2:2 format, the host PC transfers raw YUV video to DM6437 via Ethernet. The NDL layer in the DSP decodes the data and sends it to the codec engine for video compression. The compressed video is sent back to the host PC through the same network interface. The host PC then saves back the encoded video with .h264 extension. The architecture diagram is shown in Fig. 4. At the host controller, the Joint Model (JM) decoder [14] is utilized to carry out the decoding of the frames (using ldecode.exe of JM) and the videos are played using vlc player to check its visual quality. Detailed performance analysis is reported in the experimental results section.

IV. Experimental Results

The hardware experimentation is carried out using Raspberry Pi 3 and TI DaVinci DM6437 boards and its associated IDE namely Code Composer Studio (CCS) with the source code written in Python. Video sequences namely Container, Stefan, Bus, Coastguard, Garden and Waterfall are used in the study since these video sequences are considered standard in video analysis research. The proposed implementation is objectively compared using PSNR analysis, compression performance analysis and running time analysis. Additionally, in case of video enhancement the subjective results (i.e. qualitative analysis) are shown by taking few sample frames from the videos. The results section is split into two subsections i.e. video enhancement sub-section and video compression sub-section. In both sub-sections the results obtained on two different hardware boards namely Raspberry Pi 3 and TI DaVinci DM6437 are demonstrated and analyzed. Hardware v/s software timing analysis is also carried out in order to benchmark the performance so as to demonstrate

the usefulness of utilizing hardware boards for surveillance applications especially for the defence sector.

A. Video Enhancement Implementation

In this subsection the video enhancement results are shown using both subjective evaluations as well as objective evaluations. Fig. 5 shows few frames taken from the video sequences which are converted to low light and low contrast synthetically using GIMP software. Their enhanced outputs after they have passed through the histogram matching algorithm which has been run on two hardware boards i.e. Raspberry Pi 3 and TI DaVinci DM6437 are shown in Fig. 5. As observed the video frames appear brighter in comparison to their original form which shows the efficacy of the histogram matching algorithm when run on the hardware boards in carrying out the enhancement task. Since UAV's and rockets are mostly deployed in day/night settings such an enhancement is going to prove beneficial in surveillance operations. The next analysis is to validate the objective standing which is carried out using PSNR analysis and running time analysis. Table I shows the PSNR analysis and as observed there is significant improvement for various videos demonstrating the improvement in quality which is also seen subjectively in Fig. 5. It is also to be noted that since the same histogram matching algorithm is run frame by frame on the two boards the subjective performance remains same for both boards as shown in Fig. 5.

TABLE I
PSNR EVALUATION

Video Name	Frame Resolution	PSNR (dB)	
		Before Histogram matching	Post Histogram matching
Container	160 × 120	13.73	21.21
Stefan	160 × 120	14.45	17.47
Bus	160 × 120	11.89	22.95
Coastguard	160 × 120	10.65	17.67
Garden	160 × 120	10.65	17.67
Waterfall	160 × 120	10.65	17.67

TABLE II
RUNNING TIME ANALYSIS FOR HISTOGRAM MATCHING ON HARDWARE BOARDS.

Video Name	Frame Resolution	Time per frame (in ms)	
		Raspberry Pi 3	DaVinci DM6437
Container	160 × 120	85.11	08.00
Stefan	160 × 120	88.24	07.80
Bus	160 × 120	90.12	07.70
Coastguard	160 × 120	86.25	08.10
Garden	160 × 120	86.25	08.10
Waterfall	160 × 120	86.25	08.10

Running time analysis is carried out next. This is an important objective metric since by analyzing it the necessity for hardware realization of video processing algorithms can be ascertained. Table II shows the running time per frame when histogram matching is run on the Raspberry Pi 3 and TI DaVinci DM6437 boards for different video sequences. An important observation to be made is that the histogram matching algorithm runs faster on the TI DaVinci DM6437 in comparison to Raspberry Pi 3 to the extent of 10 times. This is due to the fact that DaVinci is a dedicated video processing board while Raspberry Pi is a general purpose processor which has been re-purposed in the current experiment to execute video enhancement algorithm. It is to be noted that the frame resolution is also one of the criteria which determines the running time and larger frame resolutions take more time to execute due to large number of pixels to process. To benchmark the hardware running time against running time measured via simulations, the same histogram matching enhancement algorithm is coded using C and run on an Intel i5 CPU @ 2.3 GHz, 8 GB RAM for the same videos listed earlier and the running time measured is shown in Table III. Analyzing the case of hardware v/s software we can clearly see significant decrease in running times for TI DaVinci hardware board roughly of the order of 40 times clearly warranting more attention in the area of hardware video processing which to the best of our knowledge has not been done earlier and is very essential for time critical defence applications since it gives considerable computing advantages. An important inference from the table is that hardware boards support real time video processing since the execution speed equals the video frame rates for most video sequences of moderate frame resolutions.

TABLE III
RUNNING TIME ANALYSIS FOR HISTOGRAM MATCHING (SOFTWARE V/S HARDWARE).

Video Name	Frame Resolution	Time per frame (in ms)	
		C on Desktop	DaVinci Board
Container	160 × 120	293	08.00
Stefan	160 × 120	295	07.80
Bus	160 × 120	288	07.70
Coastguard	160 × 120	303	08.10
Garden	160 × 120	294	08.10
Waterfall	160 × 120	300	08.10

TABLE IV
COMPRESSION PERFORMANCE ANALYSIS FOR RASPBERRY PI 3.

Frame Resolution	Duration (in sec)	Data Size (KB)	
		Before Compression	After Compression
160 × 120	5	7200	98.00
176 × 144	5	10044	91.00
320 × 240	5	28800	323.50
640 × 480	5	115200	432.50

B. Video Compression Implementation

In this subsection we analyze the compression algorithm performance when run on the two hardware boards. As described earlier the H.264 compression algorithm is utilized. The baud rate is set to max value i.e. 921.6 kbps for both hardware boards. DaVinci DM6437 utilizes XDC tools XDAIS

TABLE V
COMPRESSION PERFORMANCE ANALYSIS FOR TI DAVINCI DM6437.

Frame Resolution	Duration (in sec)	Data Size (MB)	
		Before Compression	After Compression
160 × 120	5	38.20	02.11
176 × 144	5	45.80	0.356
320 × 240	5	296.00	05.86
640 × 480	5	225.00	05.28

to compile packages required for video encoding and decoding. The codec engine is used to configure to compression of incoming video frames. Desired bitrate and GOP can be set by changing the values in header files. The DSP BIO version 5.x, CCS version 5.x, XDCtools version 3.16, codec engine version 3.2 and C64+ compiler to compile embedded C specific to DSP are utilized.

The compression performance when run on Raspberry Pi 3 board for four different video frame resolutions is shown in Table IV and the compression performance when the DaVinci DM6437 board is utilized is shown in Table V. It is to be noted that the video is captured for 5 seconds for all videos so that the performance comparison is fair for the two boards. As observed, compression achieved is better for DaVinci DM6437 board in comparison to the Raspberry Pi 3 due to multiple optimizations being available on the DaVinci DSP as compared to the Raspberry Pi board. It can also be inferred from both the tables that as the frame resolution increases the size of the video before compression increases corroborating the fact that when the time duration is fixed (5 seconds) increase in resolution would mean larger frames at the specified frame rate leading to increase in data size. When these uncompressed videos are passed through the hardware boards the compression engines on both the boards are enabled and the videos get compressed and the compression performance is specific to the hardware boards optimizations. In this study the two hardware boards performance for similar frame resolutions are demonstrated in Table IV and Table V. As observed the compression performance is better with DaVinci DM6437 board since it offers better compression tuning abilities as compared to the Raspberry Pi 3 board.

V. CONCLUSIONS

This paper presented the hardware realization of video processing algorithms utilizing two boards namely Raspberry Pi 3 and TI DaVinci DM6437. The algorithms chosen were belonging to video enhancement and video compression category. Histogram matching was the enhancement algorithm which was run on the hardware boards and its detailed analysis demonstrating the subjective and objective performance for various video sequences was demonstrated. It can be concluded that DaVinci DM6437 achieves faster run time in comparison to Raspberry Pi 3 for the same frame enhancement performance.

H.264 was the compression algorithm which was realized on the two hardware boards and its detailed performance was also studied. Here too observations supported choice of DaVinci DM6437 if cost considerations are not a priority. Raspberry Pi 3 has its own advantages in the sense that it allows algorithm verification at low costs which many times will be useful when only prototype verification needs to be done. Many other important inferences were drawn from the study carried out in the paper especially from the surveillance perspective suitable for defence applications. To the best of our knowledge such a detailed study has not been carried out earlier in the domain of video processing from the hardware point of view and this work is a novel attempt in this direction. Our future work is focussed on FPGA realization of the said algorithms so that it can be fit in small spaces of restricted diameter like missiles/rockets and other UAV's

VI. ACKNOWLEDGEMENT

This work is supported by ARMREB, DRDO under grant number: ARMREB/ASE/2019/220.

REFERENCES

[1] L. Liu, Z. Li, and E. J. Delp, "Efficient and low-complexity surveillance video compression using backward-channel aware wyner-ziv video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 4, pp. 453–465, 2009.

[2] H. Kim, C. E. Rhee, and H. Lee, "A low-power video recording system with multiple operation modes for h.264 and light-weight compression," *IEEE Transactions on Multimedia*, vol. 18, no. 4, pp. 603–613, 2016.

[3] Y. Hwang, M. Lyu, and C. Lin, "A low-complexity embedded compression codec design with rate control for high-definition video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 4, pp. 674–687, 2015.

[4] M. Afonso, F. Zhang, and D. R. Bull, "Video compression based on spatio-temporal resolution adaptation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 1, pp. 275–280, 2019.

[5] L. Huang, S. Chen, H. Lin, X. Kong, and J. Lin, "Implementation of g.729 codec based on davinci technology," in *International Conference on MultiMedia and Information Technology,*, dec 2008, pp. 11–14.

[6] C. Baoxia, C. Junjie, and D. Yong, "Intelligent security video surveillance system based on davinci technology," in *Fifth International Conference on Measuring Technology and Mechatronics Automation*, 2013, pp. 655–658.

[7] X. Kong, H. Lin, L. Huang, and J. Lin, "Optimization of x264 decoder based on davinci technology," in *International Conference on Biomedical Engineering and Computer Science*, 2010, pp. 1–4.

[8] P. Zhao and Z. l. Jiang, "The design of intelligent monitoring system based on davinci platform," in *9th International Conference on Fuzzy Systems and Knowledge Discovery*, 2012, pp. 1981–1984.

[9] Yakun Liu and Xiaodong Cheng, "The study of video acquisition and compression codec system based on davinci," in *International Conference on Information, Networking and Automation (ICINA)*, vol. 2, 2010, pp. V2–317–V2–321.

[10] D. Talla and J. Golston, "Using davinci technology for digital video devices," *Computer*, vol. 40, no. 10, pp. 53–61, oct 2007.

[11] R. C. Gonzalez and R. E. Woods, *Digital image processing*. Prentice Hall, 2008.

[12] A. K. Jain, *Fundamentals of Digital Image Processing*. USA: Prentice-Hall, Inc., 1989.

[13] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[14] "The h.264/avc jm reference software,http://iphome.hhi.de/suehring/tml/." [Online]. Available: http://iphome.hhi.de/suehring/tml/