# Video Enhancement for ADAS Systems based on FPGA and CNN Platform

**Alireza Fasih** *, **Christopher Schwarzlmueller** *,
**Kyandoghere Kyamakya** *, **Fadi Al Machot** *

* *Institute of Smart System Technologies, Transportation Informatics Group, Alpen-Adria University Klagenfurt, Universitaetsstrasse 65-67, Klagenfurt, Austria*
*tel/fax: +43 (0) 463 2700 3549*
*e-mail: {alireza.fasih, christopher.schwarzlmueller, kyandoghere.kyamakya, fadi.almachot}@uni-klu.ac.at*

***Submitted: 31/03/2010***
***Accepted: 12/05/2010***
***Appeared: 25/05/2010***
©*HyperSciences.Publisher*

**Abstract:** In road-transportation terminology, ADAS (Advanced Driver Assistance System) is a combination of monitoring-, alarm-, and control systems for increasing safety level in vehicles. Most of ADAS systems are infrastructure- independent and autonomous. This means these systems must work without any external awareness systems on the roads during the drive. Nevertheless, there are some systems which can cooperate with other vehicles (i.e. CACC - Cooperative Adaptive Cruise Control) or awareness systems on the road. Camera is the main part of monitoring and observation unit in ADAS systems. In some cases, fusion of camera data (video frames) and LIDAR (Light Detection And Ranging) leads to more safety and confidence in data. Most of ADAS sub systems such as Adaptive Cruise Control (ACC) , Lane Departure Warning (LDW) , Automatic Parking, Collision Warning Avoidance (CWA) systems and Lane Change Assistance are based on video information. Enhancing video frames is possible by common image processing techniques. These classical algorithms with the sequential processing architecture unit take lot of resources and energy. For producers of ADAS, the main challenges for enhancing image quality are speed and manufacturing cost. It is possible to overcome these problems by developing a proper framework and architecture. In this research, we propose a new method of CNN design based on ESL (Electronic System Design). To implement this model we used fixed-point arithmetic technique and discrete model for CNN.

*Keywords:* Cellular Neural Network, Advanced Driver Assistant System, Computer Vision, Image Processing, Reconfigurable Hardware.

## NOMENCLATURE

| | |
|---|---|
| ACC | Adaptive Cruise Control |
| ADAS | Advanced Driver Assistant Systems |
| BRAM | Block Random-Access Memory |
| CACC | Cooperative Adaptive Cruise Control |
| CAS | Collision Avoidance Systems |
| CCD | Charge-Coupled Device |
| CMOS | Complementary Metal Oxide Semiconductor |
| CNN | Cellular Neural Networks |
| CWA | Collision Warning Avoidance |
| DDA | Digital Differential Analyzer |
| DDR2 | Double Data Rate |
| DVI | Digital Visual Interface |
| ESL | Electronic System Design |
| FIFO | First In First Out |
| FMC | FPGA Mezzanine Card Standard |
| FPGA | Field Programmable Gate Array |
| FSL | Fast and Simplest Link |
| HDL | Hardware Description Language |
| HDMI | High Definition Multimedia Interface |
| LDW | Lane Departure Warning |
| LDWS | Lane Detection Warning System |
| LVDS | Low-Voltage Differential Voltage |
| MPMC | Multi Port Memory Controller |
| PLB | Processor Local Bus |
| VHDL | Very High Speed Integrated Circuit Hardware Description Language |

## 1. INTRODUCTION

Over the last few decades ADAS made their way to an important part of the automobile Jones (2001). The most important reason for this trend is that every minute, on average, at least one person dies in vehicle crash Jones (2001). A further reason is that computer parts became cheaper and more powerful. Due to these reasons, car developers are trying to produce robust and powerful ADAS. Common ADAS are responsible to increase the security in cars (e.g. the Anti-Lock Braking Systems and Electronic Stability Program) and efficiency of traffic flow

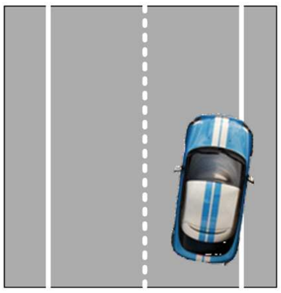
Fig. 1. Lane departure warning.

on urban. In the case that an accident is not avoidable, ADAS is responsible to minimize the effect of accident on the passenger Küçükay and Bergholz (2004). Newer systems are able to support the driver under real-time conditions. Image processing-based applications can control modern vehicles (e.g. autopilot for parking). Advanced Driver Assistant Systems (ADAS) can be subdivided into three different categories: safety-DAS, comfort-DAS and economy-DAS Küçükay and Bergholz (2004). In this paper we set the focus to design a video framework enhancement for ADAS which are responsible for the safety. In detail we work on video processing of ADAS which can be used as high level warning system and control unit. Examples for these ADAS are CACC which controls the distance to other cars, LDWS Polychronopoulos and Beuter (2006) which warn the driver if he unintentionally cross the lane of the road or makes a active steering intervention (see Fig. 1), or CAS which help the driver to avoid accidents. In many cases of LDWS, a camera is installed behind the windshield in a car. These images taken from the camera will be analyzed by a lane detection algorithm which searches the lane markings in each image. The system is able to return information about the width of the road, the curvature of the lane and about the position of the car on the road. These parameters will be used as input for the warning algorithm. Road sign detection is a further important vision based system.

The system is able to extract rules (e.g. the speed limit) from road signs and automatically adjust the speed of the vehicle by using a single camera and a system which uses pattern matching to identify signs Shneier (2005). ACC systems use laser beams, cameras or radar to measure the distance from the vehicle they are in to the car ahead and its speed relative to theirs Jones (2001). If the car ahead is too close, the system reacts and reduces the speed of the vehicle they are in, until a desired distance is reached. If the car ahead increases its speed, the ACC system increases the vehicle speed too, until both cars drive with the same speed. CWA systems use a video stream to track an object in front of the vehicle. It is able to warn the driver if a collision is possible. If a collision seems not avoidable the system makes an intervention Jones (2001). In all systems which are mentioned above we need a good quality video to detect objects and marks/signs. Environment is highly dynamic and it can affect the video processing result. Main problems are foggy weather, rainy weather and effect of sun on the video during the sunrise and sunset. All of these problems can influence the quality of video. Most of ADAS systems dont have the ability to enhance these degraded images. Costly modules for image enhancement must be added which are in common not well suited for image real-time processing or cannot handle dynamic changes in environment. By using the CNN paradigm on a FPGA we are able to promise a real-time compliant platform for enhancing video streams.
This paper is organized as follows: firstly we give a short description about video enhancement in general. Secondly we propose a system architecture and hardware platform specification. Thirdly we show a discrete and fixed-point architecture for CNN in Section 4. In Section 5 we give a description about ESL design. In Section 6, we show a method of hardware and software integration and after that in Section 7, we show a simple application on this platform for contrast enhancement. Finally we give a short conlcusion and discuss our future research in Section 8.

## 2. ENHANCEMENT AND RECONSTRUCTION OF DEGRADED IMAGE SEQUENCES FOR ADAS

There are many interruptions which cause to a bad quality of an image. We distinguish between image enhancement and image reconstruction. The first technique does not increase the information content of the image data. Image enhancement only increases the dynamic range of chosen features. Image reconstruction is the application of reconstructing a degraded image by a model. In context of ADAS we are only talking about influences which exist in outdoor environments. One kind of noise disturbance is bad pixels and also white noise. This is due to the CCD problem in a digital camera which acquires the visual information. In case of a not adequately configured camera it is possible that images get blurred. A further influence which degrades images is the influence of bad contrast. Bad contrast is caused through bad lighting conditions in the outdoor environment (e.g. weather based or day-time dependent). The problem of reconstructing weather degraded images is that the influence of weather is not uniformly distributed over the whole image. In that case it is necessary to create a model for the degeneration effect (e.g. fog) and use this model as support to reconstruct images adequately Acharya and Ray (2005). Other weather effects which corrupt the quality (if an image is well suited for ADAS) of an image are snow or rain. All these effects complicate e.g. the extraction of road signs or road lanes. For enhancement of images which are used for ADAS there exists a broad spectrum of algorithms. Classical approaches for reducing (or smoothing) noise are spatial lowpass, high-pass or band-pass filtering. Further methods are unsharp masking and crisping, directional smoothing, and median filtering Acharya and Ray (2005). Contrast enhancement of images is the transforming of pixel intensities so that they not occupy only a small portion of the available range of intensities. This is done through a histogram modification. Linear contrast stretching and histogram equalization are two very common approaches for contrast enhancement. Further methods for contrast enhancement are based on the frequency domain, like homomorphic filtering Acharya and Ray (2005). Image restoration is necessary due to the imperfection of the imaging system or in the transmission channel. Moreover image restoration is required if degradation due to atmospheric conditions (like fog) appears or due to relative motion between the object and the camera. To improve the quality of images acquired by optical sensors like a camera the following techniques are used. Common methods are the inverse filtering, Wiener filter, tomographic reconstruction, or image restoration by using Bisprectrum Acharya and Ray (2005). Special approaches for rainy or foggy weather degraded image reconstruction can be found in Narasimhan and Nayar (2003); Garg and Nayar (2004) which use models for the degeneration effect. Most of these problems occur in ADAS systems and affects their reliability. The goal of this paper is designing a reconfigurable and robust universal framework which also provides real-time images processing.

## 3. SYSTEM ARCHITECTURE AND HARDWARE PLATFORM SPECIFICATION

There are many different architectures for image and video processing. We can classify these architectures in two groups of fixed point and floating point techniques. Fix point methods are faster than floating point methods in similar systems. Floating point techniques need more resources for implementation. The only disadvantage of the fix point technique is the precision of registers. In FPGA and reconfigurable systems we can choose a proper range and optimize the range for each unit. By this way we can overcome to the precession problem. Standard architectures for a video processing system must contain essential modules such as capturing unit, processing unit, memory and video output controller.

DVI, S-video input, composite input and dedicated digital camera are four different ways of capturing video signal for our system. DVI is a video interface standard designed to provide very high visual quality on digital video processing and displaying systems. DVI signals are partially compatible with HDMI signals. To design our system more abstract and also modular we used a FMC video grabber (daughter board). Input signals, capturing configuration and synchronization with FPGA are controlled by a central pico-processor on the board. In the FPGA we can temporary store a part of the video stream in the BRAM. The video buffer contains the pixel values and also vertical and horizontal disabled/enabled signals. Because of resource limitation in FPGA we choose the stream processing (marshal processing) method. In this architecture processing can be done locally, therefore we do not need to analyze the whole image. Dependent on the filter and video processing algorithm, 3 or 5 rows are sufficient. The video processing module has access to the video memory for storing and retrieving video signals. The main functions for video processing are convolution based filters. Dependent on the convolution size, we must separate the video stream in some rows (e.g. 3, 5 or 7) and keep it in the internal FPGA memory. This platform comprises Xilinx® XtreamDSP kit 3400 and Video FMC daughter board which has a PicoBlaze for video signal capturing. It has a DVI Input, Composite/Svideo input and output and camera input. This board has two camera interfaces to allow the capture of data from two cameras at the same time and simultaneously. Camera is a custom CMOS camera based on Micron MT9V022 image sensor chip. This camera sends a High speed LVDS data stream format to the board. It works by 26.6MHz clock rate that is generated by a reference clock which is provided via a fixed-frequency 27MHz oscillator on the board. Fig. 2,

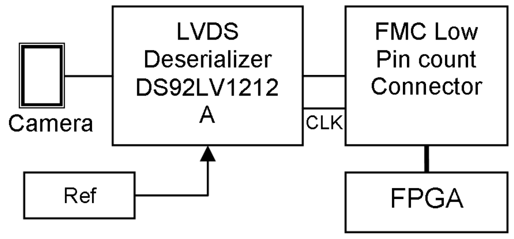

Fig. 2. Camera Connection in FMC Board.

shows the camera connection diagram in FMC board. In the initial step we must define the voltage source level for the FMC board as well.

Xilinx XtremeDSP kit 3400 is ideal for image processing and machine vision applications. This platform provides an embedded design framework that can be customized with user defined video accelerator implemented on the FPGA fabric. It has one Spartan FPGA with 3.4 million system gate and 126 DSP48 internal blockset. This board has 256MB DDR2 and 256MB Compact Flash. Clock rate for FPGA is up to 200MHz.

## 4. DISCRETE AND FIXED-POINT ARCHITECTURE FOR CELLULAR NEURAL NETWORKS

Cellular Neural Network (CNN) is a paradigm of parallel processing that is similar to the neural network with the difference that the connectivity between the cells are locally. The CNN paradigm was introduced in 1988 by Chua and Yang Chua and Yang (1988). CNN is a massive parallel computing paradigm which is well suited for image processing tasks. This characteristic and that it is possible to realize CNN in hardware makes it possible to implement it on FPGA Schwarzlmueller and Kyamakya (2009); Fasih et al. (2008). The advantage of CNN is that the most of the structure is fixed - only the templates must be changed for the different processing steps. Main parts of a CNN module are the convolution module and integrator Martínez-Alvarez et al. (2007). For designing an integrator module we need a memory to keeping data and cell values. According to the (1) we need a feedback template and control template for each cells.

$$\dot{x}_{ij} = -x_{ij} + T_B * u_{ij} + T_A * y_{ij} + I_{ij} \tag{1}$$

Where $T_A$ denotes the feedback template $(3 \times 3)$, and $T_B$ denotes the control template $(3 \times 3)$, $u$, $x$, $y$, and $I$ are the input, the state, the output and the bias term, respectively. The output signal is related to equation (2):

$$y_{ij}(t) = f(x_{ij}(t)) \tag{2}$$

In this term, function $f(x)$ is a nonlinear activation function defined as (3).

$$f(x) = \frac{1}{2}\left(|x+1| - |x-1|\right) \tag{3}$$

We can model this equation by a simple approximation technique like DDA. DDA, sometimes called "digital integrating computer", is a digital implementation of the differential analyzer. The integrators in DDA are implemented as accumulators, whereby the numeric results are converted back to a pulse rate by the overflow of the accumulator. The main advantage of the digital integrator, when compared to an analog integrator, is the scalable precision. Also, in a digital integrator based on DDA, we do not have drift errors and noise due to the imperfection of electronic components. By accumulation over time of values in a register we can calculate the integral of signals. The basic digital integrator is expressed by (4).

$$X_{n+1} = X_n + K \cdot S \tag{4}$$

In (4) $X_{n+1}$ denotes the next state of the accumulator used for calculating the integral. The coefficient of $K$ is a constant factor that is less then 1; it is used for time scaling. In this equation $S$ denotes the input signal for integration. We can map this technique on FPGA very easily by writing a behavioral code. After each rising clock pulse, the equation updates the integral value. In this integrator, Rounding or truncation errors are only due to the limitation of registers. Therefore, by increasing the register sizes we have a way to control/reduce this error. This error is cumulative. Thus for low precision registers a lack of accuracy will be observed after long time The only way to overcoming to this problem is setting proper register sizes.

By this way we can compute directly the solution of differential equations. This simulation of analog computing is fully parallel method to solving differential systems such as nonlinear equation and also CNN. For integrator module we must have access to the memory for storing the values and cells output. The only critical term in CNN equation is Integrator, that we can use DDA model for implementing this part. After CNN approximating, we must cascade the cells together. All these steps are implemented in CoDeveloper by Fixed Point method. The result for each three rows will store in the memory. CoDeveloper can handle access to the external memory through the MPMC. This controller is a full feature memory controller that is compatible with standard DDR2 memory devices. This controller must be configured for at least one read and one write port. And for the many high end video processing applications, there is no implicit limit on the number of read or writes ports in MPMC.

Fig. 3, shows the CNN architecture model, which has been implemented by using the fixed point technique. According to the original model of CNN, it consists of two parts, one is constant and the other part is variable during the time. Convolution between control template and input also bias value is always constant during the CNN calculation $T_B * u_{ij} + I$. Therefore, system does not need to recalculate it during the solving CNN. The only part that must be recalculated in DDA iteration is term of $T_A * y_{ij}$. The value of Template Convolution and bias is constant for each pixel and we do not need it to recalculate this value for each CNN's iteration Chua and Yang (1988); Kayaer and Tavsanoglu (2008).

For storing matrix template values and convolution we assigned a 14 bit signed register for each cell, 1bit sign, 3bit integer and 10bit precision. Similar to the standard analog CNN chip, we had limited the range of values between -5 and +5. We need an 18bit register to keep a convolution result. By 1bit sign, 7bit integer and 10bit precision it

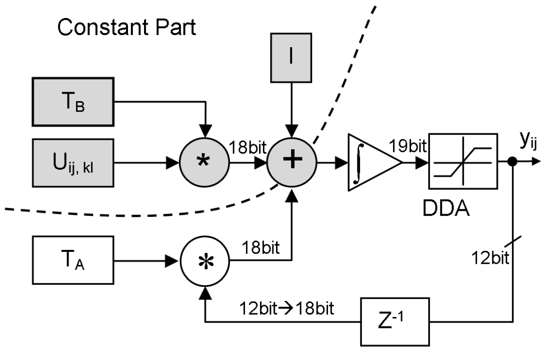

Fig. 3. DDA Based Model for CNN.

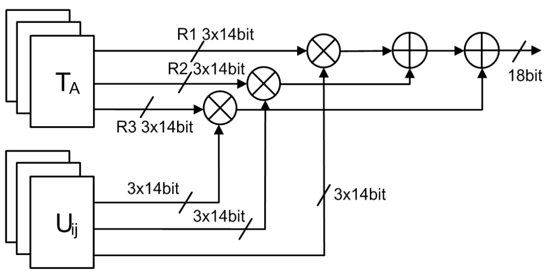

Fig. 4. Convolution diagram for control template.

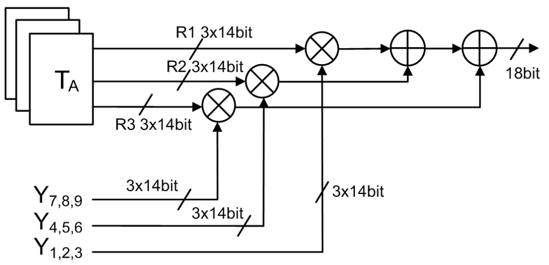

Fig. 5. Convolution template for feedback template.

can store the values from $-127$ up to $+127$ with good accuracy. Maximum value for the convolution is in the range of $(-125, +125]$, if we define a Bias value in the range of $(-3, +3]$, still we can save the value at same range register (18bit).

Fig. 4, shows the control template and convolution diagram, which is same as feedback template convolution with output signals. Signals after saturation function is in the range of $[-1, +1]$, we must extend it from 12bit to 14bit range before getting convolution on feedback template.

Fig. 5, shows the feedback template which is combination of convolution and adder. According to the (4), for implementing an integrator we need a register to accumulate the input value with the current state/value. In (4) we assign a 2-9 as a $K$ factor.

$$x(n) = x(n-1) + (T_A * y + T_B * u + I)^{2^{-9}} \tag{5}$$

To apply $K$ factor we can do a 9 time shift to right. Depending on the factor $K$ we can increase the accuracy. There is a trade-off between accuracy and speed. For the small $K$ factor, we need more iteration to get the result. On the other hand by this way we can change the time scale. In order to enhance the quality of images for ADAS

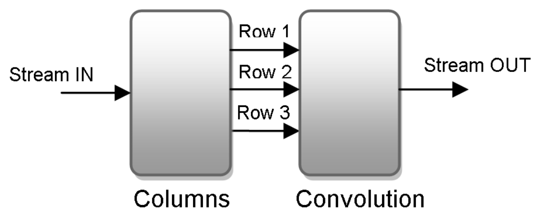

Fig. 6. Convolution and Stream Processing Diagram.

we use CNN templates for noise reduction Aizenberg et al. (2001) and contrast enhancement Gacsádi et al. (2005).

## 5. ELECTRONIC SYSTEM LEVEL DESIGNING

There are many possibilities to design a proper model for Video Processing signals. Main part of an Image processing engine, we need a convolution operator. This basic and simple mathematic model is very useful for implementation of many complex filters such as first derivative and second derivative find edge operations, median filters, noise cancelation and so on. Due to complexity of filters the best way to design a convolution module in FPGA is using HDL programming. There is lot of technique to implement and generate HDL code. One the best technique is using a High end technology of ESL like Impulse CoDeveloper. Impulse CoDeveloper is a proper tool for developing custom IP modules such as convolution and Image processing modules. This tool allows designers to quickly develop custom filter modules in standard c program. By this tool we can convert untimed c program to synchronized HDL code. One of the best facilities of this tool is debugging modules by higher level c standard program such as Microsoft visual studio and so on. CoDeveloper results are fully compatible with standard C and VHDL. Almost all the data interaction between modules and process in CoDeveloper is based on stream passing and shared memory. For creating this convolution modules we designed two parallel c-language process named columns and convolution. The column process has access to the incoming video stream and it can store the pixels values of 3 rows of image in the internal memory buffer for the convolution module. These two main modules for splitting rows and convolution are illustrated in Fig. 6. Designers have access to the MPMC, by this way the program can use the pipelining technique. Depending on the hardware platform we can use pipelining technique in our design. By Pipelining #Pragma switch, which is pre compiler command in C, we can synthesis for-loops command and the internal contents of that in a concurrent mode. If system has access to the dual channel external memory, also it can synthesis some parts of code in a pipelining mode which has interaction with external memory and registers Chu (2008).

These two process will run concurrently, first process read the data and split it in 3 separated rows, and second process apply a convolution on these data. In each clock cycle, columns module reads a pixel value from the live video stream source and it writes these pixels values on the output stream channels. These pixels values will be use by the convolution process.

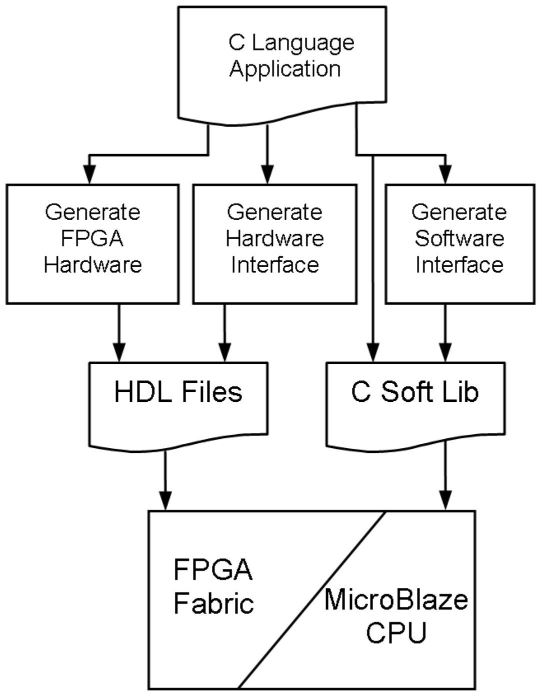

Fig. 7. Impulse CoDeveloper Design Process Diagram.

## 6. HARDWARE AND SOFTWARE

Hardware modules are significantly faster than software function. But in some case we need more flexibility in our system to access to the data and resource. The only way to get a high performance is hardware and software integration. Using high bandwidth communication path between hardware modules and processor can be very helpful. For example for changing the template in convolution and also in CNN we need a mechanism for loading data and storing in the internal registers. For convolution based video preprocessing we can just load the matrix coefficient that can changes the functionality of the system, and in the case of CNN, we must change, two different templates and bias value. We know that the Impulse C from CoDeveloper is based on the Communication Sequential Processor programming model. And it can generate the entire necessary internal signal for synchronization and communication between hardware modules. But for communication and exchanging values between hardware modules and processor, there is some method such as PLB and FSL. If we need only communication between two certain modules FSL is the best, but for communication between a master and many slave modules PLB can be helpful. For Video and Image processing we do not need any high processing over the processor, and the only reason behind that is initializing, re-configuring and debugging the system Rosinger (2004); Pellerin and Saini (2004); Pellerin et al. (2005a). Fig. 7, shows the process of HDL code and software code generator by CoDeveloper. It generates two different level code, hardware (HDL IP) and software code for integration of software and hardware.

Imuplse C can generate hardware and software interface depending on the architecture, it can be PLB based or FSL based. We have access to driver and low level code for com-

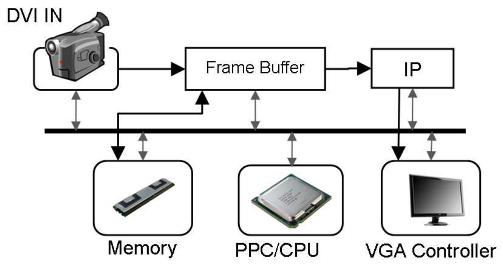

Fig. 8. General architecture for stream processing.

munication between processor (MicroBlaze/PowerPC), convolution module and CNN module. We do not need to store the pixels values of the whole image, and just after convolution calculation for every 3 rows, we can send it to the video output. In CNN we have to keep those pixels values in a buffer for the next n iterations. The number of $n$ is depending on the accuracy and DDA approximation.

For this project we used a 32bit MicroBlaze CPU for combination of hardware and software, hardware convolution modules must connect to the memory and data path according to the Fig. 8 architecture. All the parameters and data can transrecieve between modules and memory which is based on FIFO stream buffers and PLB technique Aung et al. (2007); Pellerin et al. (2005b).

## 7. RESULT

Our first prototype on this system is a video enhancement based on contrast stretching. This technique (i.e. Adaptive Contrast Stretching) is working on gray scale image to enhance the contrast Csapodi and Roska (1996). This method linearly transforms the pixel intensities of the image to the interval of [-1, 1] (see (6)). The variable $pv$ represents an actual pixel intensity which will be transformed. The minimum pixel value in the image is denoted by $pv_{min}$. The maximum pixel intensity is denoted by $pv_{max}$. It increases the difference of intensities between two adjacent pixels – if they do not have the same pixel intensity. The workflow of this method can be found in Fig. 9. Firstly we must determine the minimum (CNN 2) and the maximum (CNN 1) pixel intensities (T1) in an image which must be memorized for this method. Secondly pixel intensities must be stretched over the full pixel intensity interval (CNN 3) by using the minimum and maximum pixel values which was determined in the first step (T2).

$$f(pv) = -1 + 2 \cdot \frac{pv - pv_{min}}{pv_{max} - pv_{min}} \qquad (6)$$

Finally (T3) the input image is stretched adaptively and has now a maximum pixel intensity of +1 and a minimum pixel intensity of −1.

In a final application we need to preprocess the image because this method is sensitive to noise. Therefore we have to use a filter to remove the effect of noise. Fig. 10 shows input video and output result for adaptive contrast stretching.

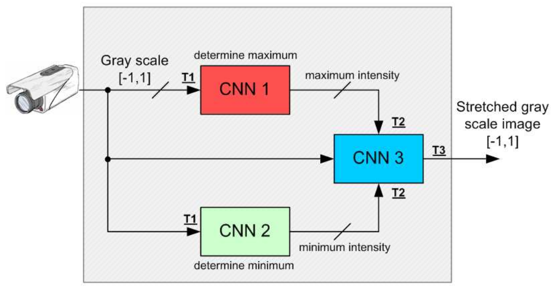

Fig. 9. CNN based adaptive contrast stretching.

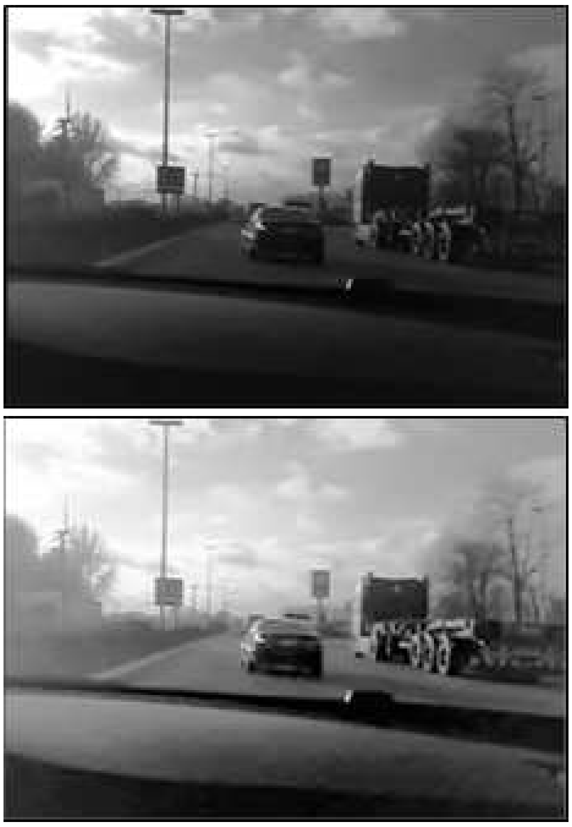

Fig. 10. (above) original image, (below) result after CNN-Based Adaptive Contrast Stretching.

## 8. CONCLUSION

In this research, a new method for CNN emulation on FPGA for real time machine vision application proposed. System is implemented on Xilinx® XtremeDSP kit 3400, which is very flexible for video and image processing application. Next step, we are going to implement a Ethernet-Based connection between Matlab® and Xilinx® XtremeDSP kit 3400 to transfer images and data from Matlab®. This connection is very useful because after that we can CNN modules as an accelerator.

## REFERENCES

Acharya, T. and Ray, A.K. (2005). *Image Processing - Principles and Applications*. Wiley-Interscience.

Aizenberg, I., Aizenberg, N., Hiltner, J., Moraga, C., and zu Bexten, E.M. (2001). Cellular neural networks and computational intelligence in medical

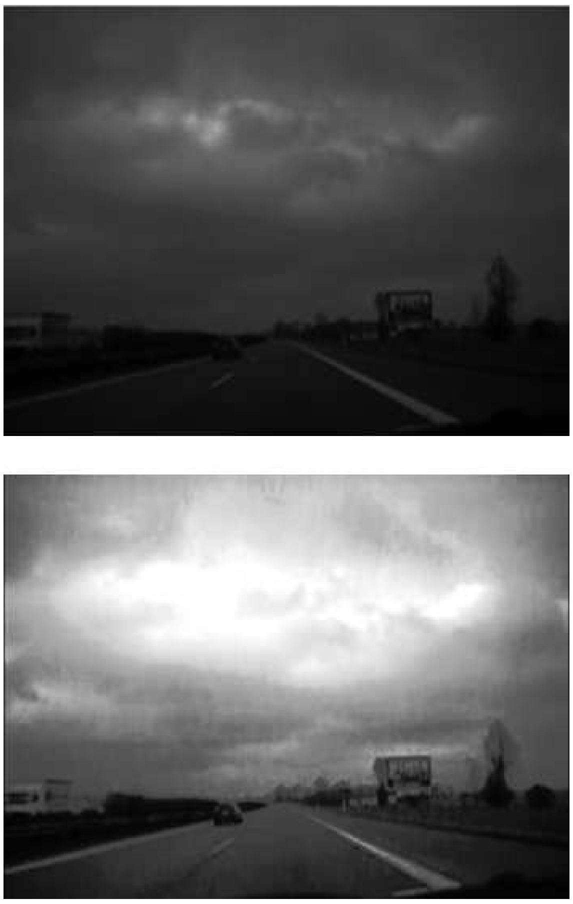
Fig. 11. (above) original image, (below) result after CNN-Based Adaptive Contrast Stretching.

image processing. *Image and Vision Computing*, 19(4), 177 – 183. doi:DOI:10.1016/S0262-8856(00)00066-4. URL `http://www.sciencedirect.com/science/article/B6V09-428127X-1/2/b622e3d4bbf1086e66d0be7bcd1da219`.

Aung, Y.L., Maskell, D.L., Oliver, T.F., Schmidt, B., and Bong, W. (2007). C-based design methodology for fpga implementation of clustalw msa. In *PRIB*, 11–18.

Chu, P.P. (2008). *FPGA Prototyping by VHDL Examples: Xilinx Spartan-3 Version.* Wiley-Interscience, New York, NY, USA.

Chua, L.O. and Yang, L. (1988). Cellular neural networks: Theory. *IEEE Transactions on Circuits and Systems*, 35, 1257–1272.

Csapodi, M. and Roska, T. (1996). Adaptive histogram equalization with cellular neural networks. 81–86.

Fasih, A., Chedjou, J., and Kyamakya, K. (2008). Impementation of one dimensional cnn array on fpga a desgin based on verilog hdl. In *First International Workshop on Nonlinear Dynamics and Synchronization (INDS'08)*, volume 1, 31–34.

Gacsádi, A., Grava, C., and Grava, A. (2005). Medical image enhancement by using cellular neural networks. *Computers in Cardiology*, 32, 821–824.

Garg, K. and Nayar, S.K. (2004). Detection and removal of rain from videos. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1, 528–535. doi:http://doi.ieeecomputersociety.org/10.1109/CVPR.2004.79.

Jones, W. (2001). Keeping cars from crashing. *Spectrum, IEEE*, 38(9), 40–45. doi:10.1109/6.946636. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=946636\&tag=1`.

Kayaer, K. and Tavsanoglu, V. (2008). A new approach to emulate cnn on fpgas for real time video processing. In *Cellular Neural Networks and Their Applications, 2008. CNNA 2008. 11th International Workshop on*, 23 –28. doi:10.1109/CNNA.2008.4588644.

Küçükay, F. and Bergholz, J. (2004). Driver assistant systems. In *2nd International Conference of Automotive Technologies (ICAT 2004). Istanbul, Turkey.*

Martínez-Alvarez, J.J., Garrigós-Guerrero, F.J., Toledo-Moreo, F.J., and Ferrández-Vicente, J.M. (2007). High performance implementation of an fpga-based sequential dt-cnn. In *IWINAC '07: Proceedings of the 2nd international work-conference on Nature Inspired Problem-Solving Methods in Knowledge Engineering*, 1–9. Springer-Verlag, Berlin, Heidelberg. doi:http://dx.doi.org/10.1007/978-3-540-73055-2_1.

Narasimhan, S.G. and Nayar, S.K. (2003). Contrast restoration of weather degraded images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(1), 713 – 724.

Pellerin, D., Edvenson, G., Shenoy, K., and Isaacs, D. (2005a). Accelerating powerpc software applications. *Xilinx Xcell Embedded Magazine.*

Pellerin, D. and Saini, M. (2004). Microblaze and powerpc cores as hardware test generators. *Xilinx Xcell Journal.*

Pellerin, D., Thibault, S., and Thibault, E.A. (2005b). *Practical FPGA Programming in C.* Prentice Hall Press Upper Saddle River, NY, USA.

Polychronopoulos, A. and Beuter, A. (2006). The safelane adaptive lane keeping support system. In *International Conference of Automotive Technologies (ICAT 2006). Istanbul, Turkey.*

Rosinger, H. (2004). *Connecting customized IP to the MicroBlaze soft processor using the Fast Simplex Link (FSL) channel, Xilinx Application Note.* Xilinx.

Schwarzlmueller, C. and Kyamakya, K. (2009). Implementing a cnn universal machine on fpga: state-of-the-art and key challenges. In *VXV International Symposium on Theoretical Engineering ISTET 2009*, 347–351.

Shneier, M. (2005). Road sign detection and recognition. In *International Conference on Computer Vision and Pattern Recognition.*

## AUTHORS PROFILE

**Alireza Fasih** is currently a PhD Student and Research Assistant in the Institute for Smart System Technology (IST), Alpen-Adria University of Klagenfurt in Austria. He received his BSc in Hardware Computer Engineering (2004), and MSc in Mechatronic Engineering (2007) from Azad University of Qazvin. His research interests include Computer Vision, Embedded systems, FPGA-Based system design and Cellular Neural Networks.

**Christopher Schwarzlmueller** received in 2007 his MSc in Applied Informatics at the Alpen-Adria University of Klagenfurt, Austria. Currently he is doing his PhD under Prof. Kyandoghere Kyamakya and is working as Project Assistant at the Institute for Smart Systems Technologies of the Alpen-Adria University of Klagenfurt in Austria. His research interests include Computer Vision, and Cellular Neural Networks.

**Kyadoghere Kyamakya** obtained the M.S. in Electrical Engineering in 1990 at the Univerity of Kinshasa. In 1999 he received his Doctorate in Electrical Engineering at the University of Hagen in Germany. He then worked three years as post-doctorate researcher at the Leibniz University of Hannover in the field of Mobility Management in Wireless Networks. From 2002 to 2005 he was junior professor for Positioning Location Based Services at Leibniz University of Hannover. Since 2005 he is full Professor for Transportation Informatics and Director of the Institute for Smart Systems Technologies at the University of Klagenfurt in Austria.

**Fadi Al Machot** is currently a PHD student and Research Assistant in the Institute for Smart System Technology (IST), Alpen-Adria University of Klagenfurt in Austria. He received his Diplom in Computer Science from Potsdam University. His research interests include Computer Vision, Data mining, Context Models and Complex Event Detection & Classification in a Sensor Network-based Video Surveillance System.