International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:4, No:12, 2010

FPGA Implementation of a Vision-Based Blind Spot Warning System

Yu Ren Lin and Yu Hong Li

***Abstract*—**Vision-based intelligent vehicle applications often require large amounts of memory to handle video streaming and image processing, which in turn increases complexity of hardware and software. This paper presents an FPGA implement of a vision-based blind spot warning system. Using video frames, the information of the blind spot area turns into one-dimensional information. Analysis of the estimated entropy of image allows the detection of an object in time. This idea has been implemented in the XtremeDSP video starter kit. The blind spot warning system uses only 13% of its logic resources and 95k bits block memory, and its frame rate is over 30 frames per sec (fps).

***Keywords*—**blind-spot area; image; FPGA

I. Introduction

THE blind spots of a vehicle are the areas of the road that are not visible while looking through either the rear-view or the side-view mirrors. Many accidents have occurred with cars in blind spots in situations of passing or changing lanes. For example, a driver who is going to change lanes looks in the side mirror to confirm that the lane is free, but a car suddenly comes from behind, just when the driver is about to change lanes. If the driver ignores the blind spots, this critical situation often results in an accident. The "Blind Spot Detection System" helps prevent this possibility by gathering blind spot information. Past systems have often widened the range of side mirrors or installed two cameras on both sides of a vehicle to gather blind spot information. With these systems, drivers must still concentrate on the road and decide when lane changing is safe. The "Blind Spot Warning System" is an intelligent vehicle application designed to prevent distraction-related accidents. When the driver wants to change lanes, the system will warn the driver if changing lanes is not safe, now.

The blind spot warning system uses special technologies to estimate the distance of an approaching vehicle. One of the available tools is the ultrasonic radar, which detects distance using ultrasonic distance detection technology. Uvais Qidwai used a low-cost ultrasonic sensor and Fuzzy Inference System as the main decision logic forces of the blind spot warning system [1]. Radars, however, usually have blind spots and a smaller view. The range of blind spots correlates with the number of the installed radars. It is difficult for radars to detect an object moving in a large area because of its limited detection distance. In contrast, the vision-based blind spot warning system can overcome the problem of blind spots. Vision-based blind spot warning systems use CCD (Charge Coupled Device) cameras or CMOS (Complementary Metal Oxide Semiconductor) cameras that are installed on both sides of a vehicle to capture images of the blind spot areas. There are several technologies and methods for implementing vision-based blind spot warning systems, including motion information [2][3][4], using color and edge [5][6], optical flow [7], AdTM [8] and converting images into one dimensional information [9].

Y. K. Wang [2] and Y. Zhu[3] used the robust method to achieve reliable detection of overtaking vehicles. Using the robust method, one can obtain reliable motion estimations against a variety of image noise. Axel Techmer used motion estimation and contour extraction to detect and track objects [4]. Luo-Wei Tsai used the edges and color of static images to detect vehicles. In contrast with traditional methods, they found the color of vehicle using the color transform method [5]. Kai She proposed a real-time on-road vehicle tracking method. The method builds a static model for the target with color and shape [6]. Parag H. Batavia proposed an optical flow based on an obstacle detection system [7]. M. Krips used AdTM to detect the overtaking object [8]. C.T Chen proposed a method for converting the information of a blind-spot area into 1-D information [9].

Vision-based solutions often require large amounts of memory to handle video data streams and image processing, which increases the hardware and software complexity of these solutions. However, DSP (Digital Signal Processor) cannot implement some of these functions. Besides, using DSP increases greatly the system cost. Therefore, these systems use FPGA(Field Programmable Gate Array) instead of DSP. In general, a FPGA offers more functionality and costs less than a DSP design [10].

The approach of this paper is extension of research by C.T. Chen [9], and is organized as follows. Section II introduces the basic proceeding step. Section III discusses the FPGA architecture. Section IV covers the experimental and implementation results.

Yu Ren Lin is with the R&D Division, Automotive Research and Testing Center, Lugang, Changhua County, Taiwan(R.O.C) (phone:886-4-7811222; fax:886-4-781-2336; e-mail:ericlin@artc.org.tw)

Yu Hong Li is with the R&D Division, Automotive Research and Testing Center, Lugang, Changhua County, Taiwan(R.O.C) (e-mail:yhli@artc.org.tw)

II. ALGORITHM OF APPROACHING VEHICLE DETECTION

The approaching vehicle detection algorithm this study proposes is based on the image entropy estimation method. Figure 1 shows all the steps in the algorithm.

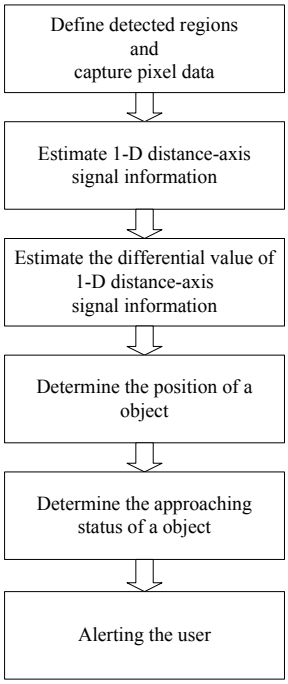

Fig. 1 Algorithmic process of approaching vehicle detection

The first step is defining the region of interest. Next, the 2-D image is converted to 1-D distance-axis signal information. Equation (1) yields an estimation of the 1-D distance-axis signal information:

$$I(x) = -\sum_{i=1}^{n} \log(Pi) \times Pi \quad (1)$$

.

$$Pi = \frac{Gi}{T_{pixels}}$$

wherein G_i is the number of Gray level i appearing in the interested region of image, T_{pixels} is the number of effective pixels in the interested region of image, P_i is the probability that Gray level i appears in the interested region, n is the Gray level in the interested region, and x is the beginning position of the 1-D distance-axis in the interested region.

Next, Eq. (2) estimated the differential value of the 1-D distance-axis signal information at a single time point:

$$I'(x) = \frac{I(x) - I(x-a)}{a} \quad (2)$$

The value of increment *a* will be set at about 10 for a camera with a resolution of 320x240. For cameras with a resolution of 640x480, the value of *a* will be set at about 15-20. The higher a camera's resolution is, the larger the increment *a* may be. The position of the approaching object is determined according to whether the differential is greater than or equal to a threshold. The approaching status is determined according to whether the difference is greater than or equal to a threshold.

To detect the movement of an object, $I_t(X)$ and $I_{t+1}(X_{t+1})$ are subtracted to attain the difference ($\Delta I_{t+1}(X)$) according to Eq. (3):

$$\Delta I_{t+1}(x) = I_{t+1}(x) - I_t(x) \quad (3)$$

When an object moves to Position X_{t+1}, $\Delta I_{t+1}(X)$ is the maximum value. The differential value of $I_{t+1}'(X_{t+1})$ and $\Delta I_{t+1}(X_{t+1})$ have obvious peaks at Position X_{t+1} where the object appears. Therefore, one can determine if an object is approaching by comparing the value of $I_{t+1}'(X_{t+1})$ and $\Delta I_{t+1}(X)$ with a threshold value or not. In the final step, when the values of $I_{t+1}'(X_{t+1})$ and $\Delta I_{t+1}(X)$ are greater than or equal to the alert limit set by the user, the user is alerted by a LED, a buzz, or a speaker.

III. FPGA IMPLEMENT OF APPROACHING VEHICLE DETECTION ALGORITHM

Figure 2 shows a block diagram of the proposed approaching vehicle detection algorithm, which consists of five main parts: a memory controller, a Gray level histogram calculation unit, a log calculation unit, an entropy calculation unit, and an approaching warning unit. First, the pixel data of interested region is written in the block memory. These pixel data are read from memory when the camera puts out pixel data that is not in the interested region. The Gray level histogram calculation unit then calculates the Gray level histogram of the interested region. Next, the Gray level histogram value is converted to a log value with a log calculation unit. After calculating the log value, the entropy calculation unit is used to calculate entropy ($I_t(X)$). Then approaching warning unit calculates the differential value of the 1-D signal ($I_{t+1}'(X_{t+1})$) and the difference of 1-D signals of at least two adjacent time points ($\Delta I_{t+1}(X)$). When the $I_{t+1}'(X_{t+1})$ and the $\Delta I_{t+1}(X)$ are greater than or equal to the alert limit, the approaching warning unit alerts the user.

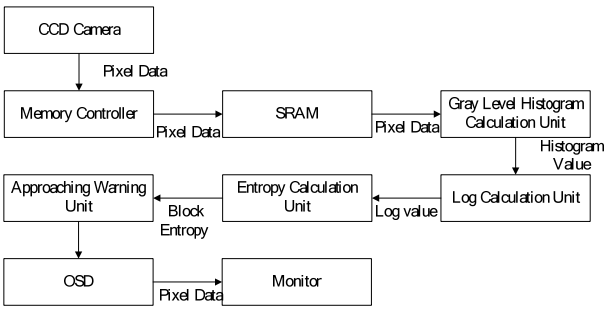

Fig. 2 Block diagram of proposed approaching vehicle detection algorithm

A. Memory Controller

The pixel data of camera is output by row, but column data are needed for the image process. As Fig. 3 shows, if pixel data is written in memory by row and read from memory by column, the pixel data is transposed from row to column.

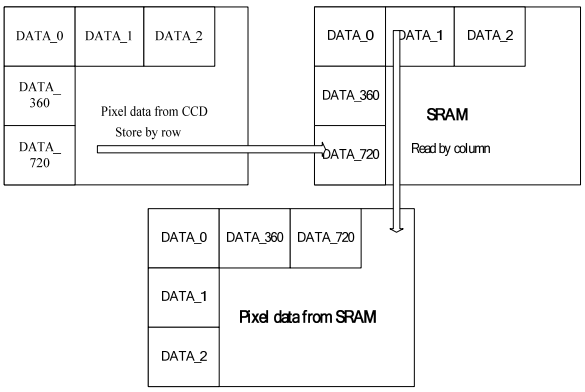

Fig. 3 Transposition of pixel data from rows to columns

However, storing whole frames greatly increases the amount of memory required. To reduce the amount of memory required, the storage unit only retains the data from the interested region. When the camera outputs the pixel data from the interested region, the memory controller turns the memory status from "reading" to "writing." Otherwise, the memory controller maintains the "reading" status.

B. Gray level histogram calculation unit

The Gray level histogram calculation unit calculates the Gray level appearing in the interested region of an image. First, it divides a frame into several blocks. Each block includes five sub-blocks. A sub-block includes one column pixel data from the interested region. As Eq. (4) shows:

$$H_x = S_x + S_{x+1} + S_{x+2} + S_{x+3} + S_{x+4} \quad (4)$$

wherein H is the block histogram value of Gray level appearing in the image's interested region, S is the sub-block histogram value of Gray level appearing in the interested region of the image, x is the position of first block. Equation (5) reveals that:

$$H_x = S_x + S_{x+1} + S_{x+2} + S_{x+3} + S_{x+4} \quad (5)$$
$$H_{x+1} = S_{x+1} + S_{x+2} + S_{x+3} + S_{x+4} + S_{x+5}$$
$$H_{x+2} = S_{x+2} + S_{x+3} + S_{x+4} + S_{x+5} + S_{x+6}$$

To calculate H_{x+1}, S_{x+5} must replace S_x; that is, only five registers per level are necessary to restore the histogram data. Figure 4 shows the architecture of the gray level histogram calculation unit.

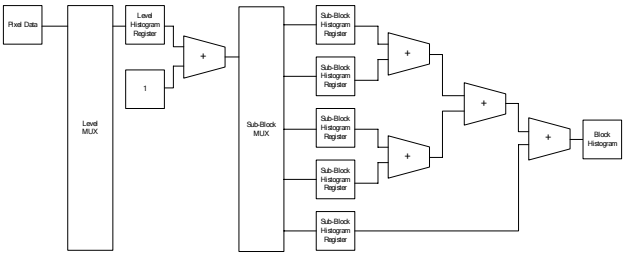

Fig. 4 The architecture of the Gray level histogram calculation unit

C. Log calculation unit

The log calculation unit converts the Gray level into a log value for entropy calculation. The log calculation unit is the computational bottleneck of the approaching vehicle detection algorithm. To implement log calculation in FPGA, one simplifies the complex calculation using the Taylor series. As Eq. (6) shows:

$$\ln(Gi) = \ln(x+1) = \sum_{1}^{\infty} \frac{(-1)^{n+1}}{n} x^n \quad (6)$$
$$x = \frac{Gi}{2^m} - 1$$

Gi is number of Gray level i, m is how many bits the value shifts, x is the decimal of the normalized value, ln(x+1) is the natural log of x+1, n is the order of the Taylor series. The log calculation unit in FPGA can be implemented from Eq.(6). Figure 5 shows the architecture of the log calculation unit.

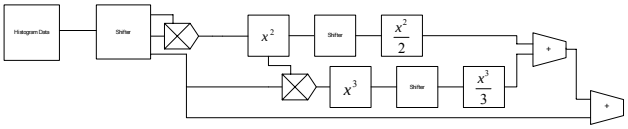

Fig. 5 The architecture of log calculation unit

First, the level histogram value is normalized using shifters. Next, the binary number without the highest bit is taken out to calculate the second and third orders of ln(x+1) with a multiplier. The value of ln(x+1) can be estimated by adders and subtractors. Then log(Gi) can be estimated according Eq. (7):

$$\ln(Gi) = \ln\left(\frac{Gi}{2^m}\right) + \ln(2^m) \quad (7)$$
$$\log(Gi) = \frac{\ln(Gi)}{\ln 10}$$

From Eq. (1), -log(pi) can be defined by Equation(8):

$$-\log(Pi) = \log(T_{pixels}) - \log(Gi) \quad (8)$$

D. Entropy calculation unit

Figure 6 shows the architecture of the entropy calculation unit.

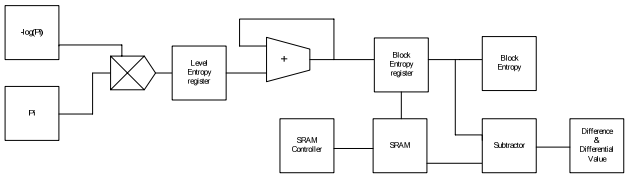

Fig. 6 The architecture of the entropy calculation unit

An entropy calculation unit includes two parts: a level entropy calculation unit and a block entropy calculation unit.The level entropy calculation unit takes the –log(pi), which is estimated with the log calculation unit to calculate –log(pi) × pi by the multiplier. Then, the block entropy calculation unit adds all level entropies and estimates the differential value of block entropies and the difference. Only adders,shifters, and subtractors can be used to implement Eq. (1), which is simplified by the Taylor series.

E. Approaching warning unit

After estimating the entropy, differential value, and difference, these three values can be used to determine if an object is approaching. If an object is approaching, the entropy and differential value will increase (Fig. 7). If the entropy and differential value exceed the threshold set by the user, the approaching warning unit alerts the user with a display, LED, a buzzer, or a speaker.

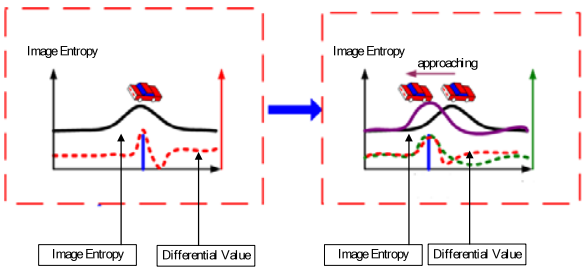

Fig. 7 Behavior description chart of approaching vehicle detection

IV. Implement Results And Experimental Results

A. Implement results

The goal of this study is to implement a low cost blind spot warning system in FPGA. The system has been successfully implemented and tested on the Xilinx XtremeDSP video starter kit. Table 1 shows the utilization of the device, and Table 2 shows the utilization of the block memory.

TABLE I
DEVICE RESOURCE UTILIZATION (XC3SD3400A)

Logic Utilization	Used	Available	Utilization
Number of slice flip flop	3,190	47,744	6%
Number of 4 input LUTs	5,937	47,744	12%
Number of occupied slices	3,687	23,872	15%
Number of slices containing only related logic	3,687	3,687	100%
Total number of 4 input LUTs	6,235	47,744	13%

TABLE II
BLOCK MEMORY UTILIZATION OF SYSTEM

Unit	Used
Memory Controller	85k bits
Entropy Calculation Unit	5k bits
On-Screen Display	5k bits
Total	95k bits

The blind spot warning system only uses 13% of its logic resources and 95k bits of its block memory.

B. Experimental results

Testing and verification of the blind spot warning system utilized a video database of blind-spot areas. This study's experimental environment was based on the Xilinx XtremeDSP video starter kit. The verifiable video was stored in a portable video recorder and played in the form of NTSC and input to the FPGA. When the system does not alert, the monitor displays a green block; when the system alerts, the monitor displays a red block (Fig. 8).

Fig. 8 Analytic result of the blind spot warning system

The video included three situations (sunny day, bright night, rainy day). The sunny day condition is the best and the rainy day condition is the worst. The frame rate is over 30 frames per second (fps) in day and night.

V. Conclusion

FPGA offers faster processing speed and costs less than traditional DSPs and microcontrollers (MCUs). This research proposes a FPGA architecture and implementation for blind-spot warning systems. The system uses only 13% logic resources and 95k bits of the FPGA's block memory, and its frame rate is over 30 frames per sec. Future blind-spot warning systems may incorporate application-specific integrated circuits to reduce costs.

References

[1] Uvais Qidwai, "Fuzzy Blind-Spot Scanner for Automobiles" in IEEE Symposium on Industrial Electronics and Applications, October 4-6, 2009, pp. 758-763.
[2] Y. K. Wang and S. H. Chen, "A Robust Vehicle Detection Approach," in Proc. Int. IEEE Conf. Advanced Video and Signal-based Surveillance, Sep. 15–16, 2005, pp. 117-122.
[3] Y. Zhu, D. Comaniciu, M. Pellkofer and T. Koehler, "Reliable Detection of Overtaking Vehicles Using Robust Information Fusion", IEEE Trans. Intelligent Transportation Systems, vol. 7, no. 4, pp. 401-414, 2007.
[4] A.Techmer, "Real-time Motion Analysis for Monitoring the Rear and Lateral Road", in Proc. IEEE Intelligent Vehicles Symposium, Jun. 14–17, 2004, pp. 704 – 709.
[5] L. W. Tsai, J. W. Hsieh and K. C. Fan, "Vehicle detection using normalized color and edge map," in Proc. Int. IEEE Conf. Image Processing, Sep. 11–14, 2005, vol.2, pp.588-601.
[6] K. She, G. Bebis, H. Gu and R. Miller, "Vehicle Tracking Using On-Line Fusion of Color and Shape Features," in Proc. 7th Int. IEEE Conf. on Intelligent Transportation Systems, Oct. 3–6, 2004, pp. 731-736.
[7] P. H. Batavia, D. E. Pomerleau and C. E. Thorpe, "Overtaking Vehicle Detection Using Implicit Optical Flow," in Proc. IEEE Conf. on Intelligent Transportation Systems, Nov. 9-12, 1997, pp. 729-734.
[8] M. Krips et al, "AdTM tracking for blind spot collision avoidance", in Proc. IEEE Intelligent Vehicles Symposium., Jun. 14–17, 2004, pp. 544 – 548.
[9] C.T. Chen and Y.S. Chen, "Real-time approaching vehicle detection in blind-spot area", in Proc. 12th Int. IEEE Conf. on Intelligent Transportation Systems, Oct. 4–7, 2009, pp. 1–6.
[10] Sudhir Sharma, Wang Chen, "Using Model-Based Design to Accelerate FPGA Development for Automotive Applications", in 2009 SAE World Congress, January 1-15,2009.