文章编号: 1000−8152(2012)12−1537−06

多光点位置实时检测系统及其应用

余祖俊[1,2], 王 尧[2], 朱力强[1,2], 郭保青[2]

(1. 北京交通大学 轨道交通控制与安全国家重点实验室, 北京 100044;

2. 北京交通大学 机械与电子控制工程学院, 北京 100044)

摘要: 提出了一种多个光点中心位置的实时同步检测系统, 该系统可以实时地同时获取多个光点在图像中的位置. 以解决现有光点位置检测系统无法同时测量多个光点或无法进行实时检测的问题. 该系统采用CCD相机采集带有多个光点的图像, 并用现场可编程门阵列(FPGA)实现图像采集的控制, 图像分割算法和光点中心位置的计算, 从而大大提高了图像处理的速度. 实验表明该系统具有较高的精度, 速度可以达到实时测量的要求, 并且使用硬件资源较少, 能够在中低密度的FPGA上实现. 最后本文介绍了该系统在基于激光的列车车体姿态检测中的应用.

关键词: 光斑分析; 现场可编程门阵列; 图像处理; 实时检测

中图分类号: TP216 **文献标识码**: A

Real-time measurement system for central positions of multiple light spots and its application

YU Zu-jun[1,2], WANG Yao[2], ZHU Li-qiang[1,2], GUO Bao-qing[2]

(1. State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing 100044, China;

2. School of Mechanical Electronic and Control Engineering, Beijing Jiaotong University, Beijing 100044, China)

Abstract: Existing spot centroid measurement systems are either unable to measure multiple light spots at the same time, or unable to measure light spot in real-time. We propose a synchronous real-time measurement system which can measure multiple centroids of light spots simultaneously. This system uses a CCD camera to acquire images including spots, and uses a field programmable gate array (FPGA) to implement the acquisition control, the image segmentation and the calculation of the centroid of the spots. Consequently, the processing speed of the system is highly increased. According to the experimental result, the processing result of this system is accurate and the processing speed is high enough for real-time measurement. Moreover, the FPGA implementation uses limited hardware resource; it can be implemented in small-scale FPGA. Its application on laser-based attitude measurement system is introduced.

Key words: blob analysis; field programeable gate array; image processing; real-time measurement

1 Introduction

In many applications[1–2], e.g., laser based measurements, star detection in astronomical telescope images, etc., the central positions of divers light spots in an image need to be measured simultaneously. In general, the position of a light spot is detected by the PSD (position-sensitive detectors), QD (four-quadrant detector), or CCD (charge-coupled devices)[3–5]. PSD and QD usually can only measure one blob position at one time due to the restriction of their structures. Although paper [4] proposed a method for measurement for multiple spot positions simultaneously using one PSD, this method needs modulation to the light sources, and will be too complex when numerous spots need to be measured.

Using a CCD camera can easily measure multiple light spots at the same time. Usually, the light spots will be represented as blobs in the image acquired by a CCD camera. And an image processing approach, blob analysis, needs to be applied to the image to calculate the positions to these blobs, then the position of the spots can be determined by some simple calibration techniques. Blob analysis is a basic problem in image processing, and is applied to various practical applications. Blob analysis focuses on extracting features, like the number of pixels, the center of gravity or the orientation, of connected regions in an image. However, blob analysis implemented by software is slow, since the main steps, including image acquisition, image segmentation, and feature extraction, must be executed sequentially.

Some work on field programeable gate array

Received 5 March 2012; revised 11 July 2012.

This work was supported by the National ‘863’ Program of China under Grant (No. 2011AA11A102), the New Century Excellent Talent Support Project of the Ministry of Education (No. NCET–11–0572), and the State Key Laboratory of Rail Traffic Control and Safety (No. RCS2009ZT012).

(FPGA) implementation of spot centroid detection algorithm have been reported recently[6–10]. Chen[6] built a spot detection system based on FPGA applied in space laser communication, but it can only detect one spot for its lack of connective region analysis. Huang and Liu[7] proposed a FPGA implementation of infrared spots detection developed form the algorithm in [8], which measures the positions of many spots simultaneously at remarkable high speed. However, since their implementation has a lot of intermediate results to restore, it costs a numerous memory resources, so it is not suitable for use on small scale FPGA.

Therefore, this paper proposes a real-time measurement system based on CCD and FPGA, which can measure multiple light spot simultaneously at sufficient high speed. This system uses a CCD camera to acquire images, and a FPGA to implement the blob analysis algorithm, which combine thresholding and connected component labelling and centroid calculations in one pass. The implementation costs very little hardware resources. Since the FPGA executes the main step of blob analysis in parallel, so it dramatically accelerates the processing speed, and can measures multiple blobs simultaneously in real-time.

2 Architecture of the measurement system

Fig.1 shows the structure of the system proposed in this paper, which mainly includes 4 parts: A CCD camera, a video decoder, a static random access memory (SRAM) and a FPGA.

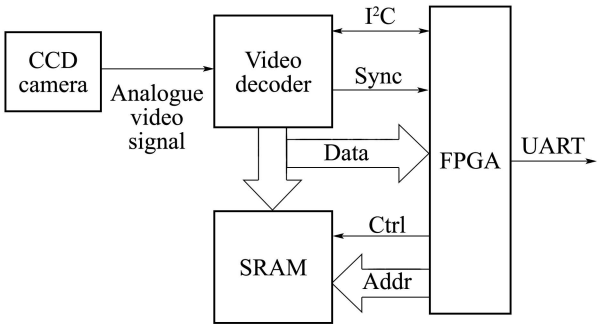

Fig. 1 Frame of the image processing system

A CCD camera and a video decoder compose the image acquisition subsystem. The CCD camera is used to acquire images of light spots. Since the exposure time is very short, the light spots can be considered as being short simultaneously. The output of the CCD camera is a standard analogue video signal, so a video decoder is used to sample, decode and finally convert the signal into digital image data. Since a frame contains 576 lines, and the sample rate of the decoder is 720 samples per line and accuracy of the analog-to-digital converter (ADC) in the decoder is 8-bits, the image data output by the decoder will be a 256 degree gray-level image in size of 720×576. The compound synchronization signals are also abstracted from the video signal by the video decoder, to provide the timing information for the FPGA.

The FPGA mainly has 4 functions: i) the FPGA controls the image acquisition by configuring the video decoder via I2C bus. ii) The FPGA produces the address and control signal for the SRAM according to the synchronization signals provided by the video decoder, to store the image data to the SRAM correctly. iii) The FPGA reads image data form SRAM and processes the image to calculate the position of light spots in the image. iv) The FPGA outputs the result through an universal asynchronous receiver/transmitter (UART) port. The SRAM is used to buffer the image data and intermediate results.

3 FPGA implementation of the measurement algorithm

3.1 Algorithm of measurement of spot centroid

The measurement method of the spot centroid used in this paper consists of three steps: the image segmentation, the connected region abstraction and the centroid calculation.

In the image segmentation step, the pixels of the spots are distinguished from those of the background. For most cases the intensity of pixels in the blobs is significantly different from the intensity of pixels in the background, e.g., in the attitude measurement system (see section 5), the camera is installed in a closed box and the only light source is the laser beams, so the blobs is much brighter than the background (as shown in Fig.4). In addition, considering the realization of the algorithm in FPGA, we adopt the fix-level thresholding algorithm for image segmentation, which considers a pixel as background when its gray-level is lower than a threshold. Although this algorithm is simple, it is effective for this application and easy for implementation in FPGA.

The image segmentation finds out all the pixels of blobs, and it should be decided which blob they belong to. Since the pixels that belong to the same blob must be connected, we can use 4-connectivity of the pixels to find out all the pixels belong to each blob. In this step, we search every pixel and assign it a label. A label is an integer. Having the same label indicates the pixels are connected.

Finally, central position of each connected region is computed as

$$\bar{x}=\frac{\sum\limits_{i\in\Omega}x_i m_i}{\sum\limits_{i\in\Omega}m_i},\ \bar{y}=\frac{\sum\limits_{i\in\Omega}y_i m_i}{\sum\limits_{i\in\Omega}m_i}. \tag{1}$$

Where Ω is a set of the pixels that belong to a same blob, m_i is the gray-level of the ith pixel, x_i, y_i are the coordinates of the ith pixel. $\bar{x}$, $\bar{y}$ are the coordinates of the central position[11].

3.2 FPGA implementation of the algorithm

For the convenient for the realization of the algorithm described above on a FPGA, we modify some aspect of the algorithm discussed above. For central position computation as formula (1), the intermediate results that need to be stored will be very large, and will use too much memory space. Furthermore, the delay of a multiply is long in FPGA, which is a restriction of processing speed. Therefore, we binarize the image, so that the value of a pixel on blob is 1. The formula (1) can be simplified as

$$\bar{x} = \frac{\sum\limits_{i\in\Omega} x_i}{N},\ \bar{y} = \frac{\sum\limits_{i\in\Omega} y_i}{N}. \tag{2}$$

Where N is the number of the pixels on a blob. The formula (2) leaves out the multiply, therefor the implementation using formula (2) reduces the intermediate results, saves the memory space, increases the processing speed. Furthermore, experiments using MATLAB show that the reduced accuracy by formula (2) is acceptable.

In addition, the algorithm described above needs to search the image several times, so it isn't appropriate to be implemented in FPGA. Consequently, we combine the image segmentation, the connected region abstraction and the addition operation of the central position calculation into one search, and execute the division operation finally. The detail of the implementation is described as follow.

The structural diagram of the image processing part in the FPGA is shown in Fig.2. It has four parts, includes an extra memory interface (EMI), an image segmentation module, a central position calculator and a controller. In Fig.2 the data signal is shown as solid arrow, and the control signal is represented by dashed arrow, while the clock signal and the reset signal are not shown.

The EMI provides the interface to the SRAM for reading the image data (gray-level). Its address is computed according to the line and column coordinates of the pixel in the image, which are provided by the LCC (lines and columns counter). When enable signal is asserted, the EMI reads the gray-level of the pixel indexed by the address.

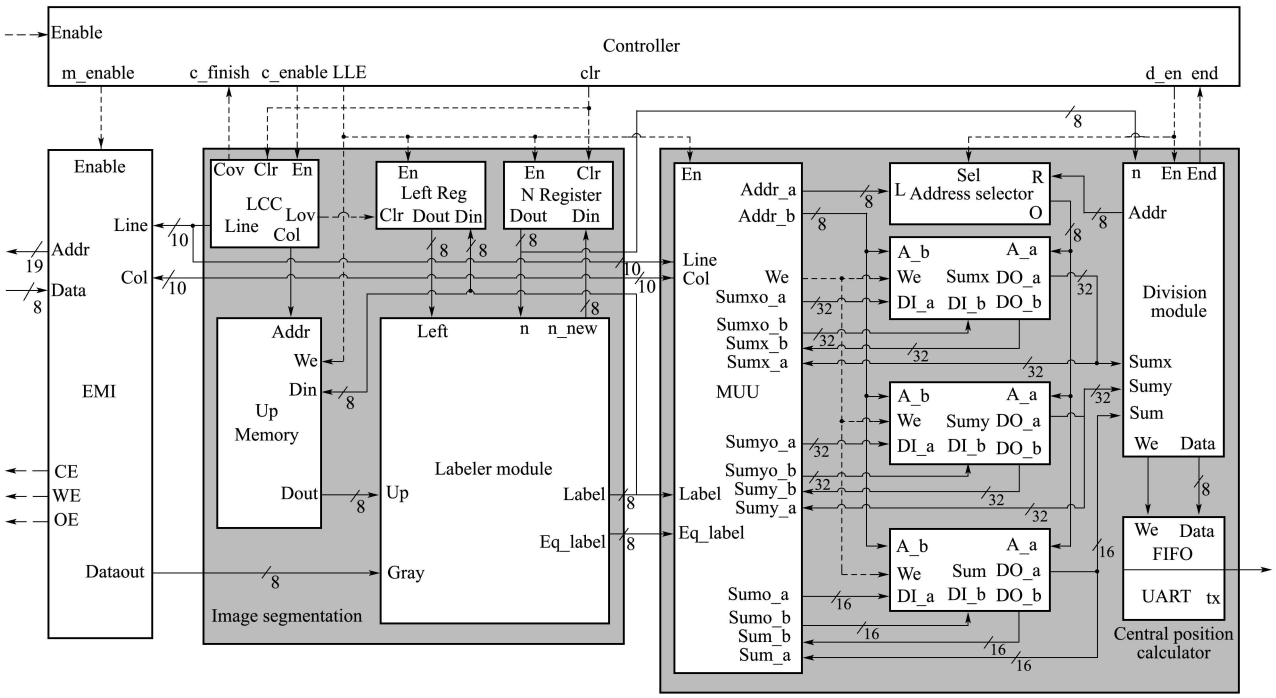

Fig. 2 Structural diagram of the image processing module in the FPGA

Image segmentation module assigns a label to every pixel. A label is a nonnegative integer. Pixels with the same label belong to a same blob. Pixels are background pixels, if their labels are zero. So after the labeling processing the image is segmented into different blobs and background. The detail of the labeling is described as follows.

Two registers Left and N are set up to store the label of left pixel and the maximum of the label assigned already. Meanwhile, a random access memory (RAM) block called Up, whose address is the column coordinate, is established to store the labels of the pixels in the last line. The LCC provides the line coordinate (Line) and the column coordinate (Col) of the pixel is currently processing. The pixels are read form left to right and top to bottom. When the enable

signal is high, the LCC updates the value of the Line and the Col to point to next pixel. The line overflow (LOV) signal will assert when the last pixel in a line is processed. The column overflow (COV) signal will assert when the last pixel in an image is processed. The labeler module is a combinational logic, which yields the label and the equal-label of a pixel according to the value of the Left register and the Up RAM. If the equal-label of a pixel is nonzero, the pixels with equal-label and the pixels with label are belonging to the same blob although they have different labels. The operation method of this module is shown as follows.

First, a threshold is decided. If the gray-level of a pixel is lower than the threshold, both the label and equal-label are 0. If the gray-level is higher than the threshold, the output is described in the following 4 situations.

1) If $\mathrm{Left} > 0$ and $\mathrm{Up} = 0$, then $\mathrm{Label} = \mathrm{Left}$, $\mathrm{Eq\_Label} = 0$;

2) If $\mathrm{Left} = 0$ and $\mathrm{Up} > 0$, then $\mathrm{Label} = \mathrm{Up}$, $\mathrm{Eq\_Label} = 0$;

3) If $\mathrm{Left} > 0$ and $\mathrm{Up} > 0$, then the left pixel and the up pixel are actually belonging to the same blob, which means the label of left pixel and up pixel are equal. There are two different situations needed to be handled separately:

i) If $\mathrm{Left} = \mathrm{up}$, then $\mathrm{Label} = \mathrm{Up}$, $\mathrm{Eq\_Label} = 0$;

ii) If $\mathrm{Left} \neq \mathrm{up}$, then $\mathrm{Label} = \mathrm{Up}$, $\mathrm{Eq\_Label} = \mathrm{Left}$;

4) If $\mathrm{Left} = 0$ and $\mathrm{Up} = 0$, then $\mathrm{Label} = N + 1$, $\mathrm{Eq\_Label} = 0$.

When the Label signal and the Eq_Label signal are stable, the controller asserts the signal called labels lock enable (LLE) to store the new value into the Left register and the Up RAM.

Now we use an image that contents 5 × 5 pixels as an example to illustrate the processing principle of the labeler. As shown in Fig.3, (a) is image data, the threshold used is 160. There are two blobs in the image; one including 6 pixels is in the center of the image, while another is in the right bottom of the image. (b) is the label that is outputted by the algorithm, whose output of the eq_label is shown in (c). Since when a pixel is processing only the labels of its left and up pixels are concerned, a new label, which is different from the label assigned to the pixel belongs to the same blob, may be assigned to this pixel. This situation is shown when pixel (3, 2) is processing, but when the next pixel is processing, the equal label will output 2 to indicate that the label 2 and label 1 are equal.

2	56	67	55	5
54	155	180	162	45
140	180	255	180	98
55	152	200	145	49
1	68	69	73	255

(a) Gray-level input

0	0	0	0	0
0	0	1	1	0
0	2	1	1	0
0	0	1	0	0
0	0	0	0	3

(b) Label output

0	0	0	0	0
0	0	0	0	0
0	0	2	0	0
0	0	0	0	0
0	0	0	0	0

(c) Eq-label output

Fig. 3 An example of input & output of labeler module

The central position calculator (CPC) divides its work into two stages. When each pixel of the image is processing, the CPC accumulates the coordinates and the numbers of pixels of each blobs. After all the pixels in the image are processed, the CPC computes the central position of each blob.

In the CPC three dual ports RAMs: sum$x$, sum$y$ and sum, are established to store the x coordinate, the y coordinate and the number of pixels for each blob. These RAMs use the label as their addresses, and their data are updated by the memory update unit (MUU). When the LLE is asserted, the value of the label and the eq_label is locked into the MUU, and outputted through the Addr_a and Addr_b ports. The MUU uses the method introduced to update the RAMs.

1) $\mathrm{sum}x[\mathrm{label}] = \mathrm{sum}x[\mathrm{label}] + \mathrm{sum}x[\mathrm{eq\_label}] + \mathrm{col}$; $\mathrm{sum}\,x[\mathrm{eq\_label}] = 0$;

2) $\mathrm{sum}y[\mathrm{label}] = \mathrm{sum}y[\mathrm{label}] + \mathrm{sum}y[\mathrm{eq\_label}] + \mathrm{line}$; $\mathrm{sum}y[\mathrm{eq\_label}] = 0$;

3) $\mathrm{sum}[\mathrm{label}] = \mathrm{sum}[\mathrm{label}] + \mathrm{sum}[\mathrm{eq\_label}] + 1$; $\mathrm{sum}[\mathrm{eq\_label}] = 0$.

The update is accomplished in two clock periods.

At the first clock period, the MUU reads the data from the RAM space indexed by the label and the eq_label. At the second clock period, the MUU writes the new data into the RAMs.

After a pixel is processed, the controller enables the LCC to generate the coordinates of the next pixel. When the LOV is asserted, the Left register will be cleared, and when the COV is asserted, it indicates that all the pixels in the image have been processed. Finally, the controller enables the division module (DM) in the CPC. The DM reads the data of sum$x$, sum$y$, sum RAMs from address 1 to N (the maximum label). When the number of pixels of a blob is big enough, the DM will compute the central position of this blob. The result will be converted into serial form, and be transmitted through UART.

4 Experiment result

We evaluate our hardware implementation on the Altera Stratix II series FPGA (EP2S60F1020C3). The hardwire is designed by Verilog HDL and the dual ports RAM are generated by Altera Megafunction. The design is synthesized by Altera Quartus II 8.0. Table 1 shows the hardware resource used by our hardware implementation with comparison to the method proposed in [7].

Table 1 Hardware resource usage

Resource	Our implement	Huang[7]
Logic units	2572	8578
Memory bits	30720	199026
Pins	61	159
PLL	1	4
Embedded multiplier(9 bits)	0	0

In the experiment, the FPGA obtains the images form a CCD camera, and detects the positions of spots, then sends both the image data and result to a PC. Fig.4 shows the image received by PC and a '+' is marked on the centroid of spots detected by FPGA, while its coordinates is shown aside the spot. The test images are 720×576 size, 256 degree gray-level, the threshold used in the experiment is 100, and the FPGA is running at 100MHz clock rate.

To confirm the validity of result by hardware implementation, we also use software to implement the same function based on three different computer vision libraries: MATLAB IPT, OpenCV and Halcon. Table 2 shows the spot coordinates measured by FPGA, MATLAB, OpenCV and Halcon using the image shown in Fig.4. Noticing that the coordinates start from 1 in the MATLAB, while they start from 0 in other three platform, so the results are completely the same, that means the hardware implementation of the algorithm is correct. However the running speed of hardware implementation of FPGA is faster than the software implementations. Table 3 shows the comparison of running time costed by our hardware implementation, three software platform described above and two other hardware implementations proposed by other authers. Although our implementation is slower than Huang's[7], our implementation uses much fewer resources as shown in Table 1.

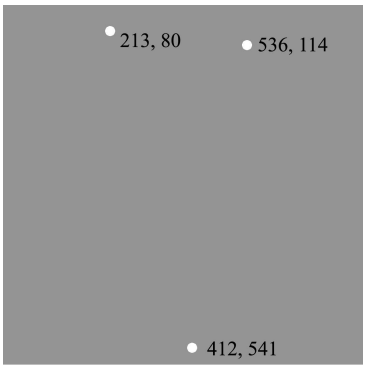

Fig. 4 Test image and processing result by FPGA

Table 2 Measurement result comparison (coordinate of spot in pixel)

Spot	FPGA	MATLAB	OpenCV	Halcon
1	$x=232$ $y=80$	$x=233$ $y=81$	$x=232$ $y=80$	$x=232$ $y=80$
2	$x=536$ $y=114$	$x=537$ $y=115$	$x=536$ $y=114$	$x=536$ $y=114$
3	$x=412$ $y=541$	$x=413$ $y=542$	$x=412$ $y=541$	$x=412$ $y=541$

Table 3 Running time comparision

Platform	Running time/ms
Our hardware implement	20.73
Huang[7]	1.966
Ito[10]	72.4
MATLAB	823.24
OpenCV	46.1
Halcon	44.4

5 Application to laser based attitude measurement

The system proposed in this paper was utilized on a laser based attitude measurement system of the comprehensive test train for high-speed railway. The comprehensive test train (CIT) is used to measure the parameters of railway infrastructures, related to the railway transportation safety. The measuring results of the CIT are impacted by its attitude. Therefore, the attitude of CIT must be measured in real-time during the test. At present, inertial reference systems, based on gyroscopes and accelerometers, are commonly used to measure the attitudes of the CIT, but

gyroscopes and accelerometers have random errors and zero drifts, which makes the system's error increasing with operation time. Consequently, the author proposed a measurement system that can detect the attitude of the train and rectify the inertial reference system[3, 12].

The system proposed includes a train mounted instrument and a ground device. As shown in Fig.5, three train mounted laser diodes (LD) emit three rays of lights, which go through three holes on a screen and reflect from the mirrors of the ground device. The reflected rays will make three blobs on the screen. When the attitude of the train is changed, the positions of these blobs on the screen also change respectively. Therefore, we can establish a coordinate on the ground device, and calculate the 6-DOF relationships between the train and the ground according to the blob positions and the routes of the rays. Furthermore, the positions of the blobs can be measured by a camera and image processing systems.

In this system, one of the key problems is how to measure the positions of the three light spots simultaneously in real-time. The measurement system proposed in this paper exactly matches the requirement. With high accuracy and processing speed of the light spot position measurement system, the attitude measurement achieves absolute accuracy of 1mm in position displacement and 0.2 degree in angular displacement, at 25 Hz sample frequency.

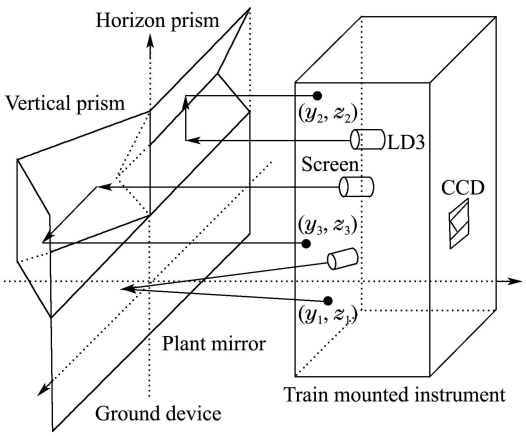

Fig. 5 Schematic diagram of the attitude measurement system

6 Conclusions

This paper proposes a synchronous measurement system of the central positions of multiple spots using a CCD camera and a FPGA. All the processing including the image acquisition, the image processing is accomplished by hardware, which is implemented in FPGA. The experiment results show that spot position measurement is accurate. The hardware implementation have sufficient high speed, while using very few hardware resource.

References:

[1] GAO Xiangdong, HUANG Shisheng, YU Yinglin. Application of computer vision in the weld seam tracking control [J]. *Control Theory & Applications*, 2001, 18(1): 26 – 30.
(高向东, 黄石生, 余英林. 计算机视觉在焊缝跟踪控制中的应用 [J]. 控制理论与应用, 2001, 18(1): 26 – 30.)

[2] LI Aiguo, MA Zi, WANG Wenbiao. Calibration for robot-based measuring system [J]. *Control Theory & Applications*, 2010, 27(5): 664 – 667.
(李爱国, 马孜, 王文标. 结构光视觉测量机器人的标定 [J]. 控制理论与应用, 2010, 27(5): 664 – 667.)

[3] WANG Y, YU Z J, ZHU L Q. Position and Attitude measurement for railway vehicles applied in railway infrastructure inspection [C] //*Proceedings of the 3rd International Symposium on Test Automation and Instrumentation*. Beijing: World Publishing Corporation, 2010, 1–4: 1316 – 1321.

[4] QIAN D H, WANG W J, ILENE J. A method of measurement of multiple light spot positions on one position-sensitive detector (PSD) [J]. *IEEE Transactions on Instrumentation and Measurement*, 1993, 42(1): 14 – 18.

[5] ZHAO Xin, WANG Shifeng, TONG Shoufeng, et al. Technology on real-time spot detection with automatic gain function [J]. *Opto-Electronic Engineering*, 2009, 36(11): 20 – 24.
(赵馨, 王世峰, 佟首峰, 等. 具有自动增益调整功能的光斑实时检测技术 [J]. 光电工程, 2009, 36(11): 20 – 24.)

[6] CHEN A S. *The research on spot detection technology of coarse trackong unit about space laser communication based on FPGA* [D]. Jilin: Changchun University of Science and Technology, 2011.

[7] HUANG Maoxiang, LIU Yuncai. Fast infrared spots detection based on FPGA [J]. *Microcomputer Applications*, 2010, 26(10): 54 – 57.
(黄茂祥, 刘允才. 基于FPGA的快速红外光斑检测 [J]. 微型电脑应用, 2010, 26(10): 54 – 57.)

[8] AMIR A, ZIMET L, SANGIOVANNI-VINCENTELLI A, et al. An embedded system for an eye-detection sensor [J]. *Computer Vision and Image Understanding*, 2005, 98(1): 104 – 123.

[9] HAN Y. *Laser spot detection system based on cmos digital image sensor* [D]. Tianjin: Tianjin University, 2010.

[10] ITO Y, NAKANO K. Optimized component labeling algorithm for using in medium sized FPGAs [C] //*Proceedings of the 9th International Conference on Parallel and Distributed Computing, Applications and Technologies*. Los Alamitos: IEEE Computer Society, 2008: 171 – 176.

[11] GONZALEZ R C, WOODS R E, EDDINS S L, et al. *Digital Image Processing Using MATLAB* [M]. New York: Prentice Hall, 2004.

[12] YU Zujun, ZHAO Jieting, ZHU Liqing. Study on measurement method of instantaneous posture of railway vehicles [J]. *Journal of Electronic Measurement and Instrument*, 2007, 21(5): 45 – 48.
(余祖俊, 赵洁婷, 朱力强. 轨道车辆瞬时姿态的测量方案研究 [J]. 电子测量与仪器学报, 2007, 21(5): 45 – 48.)

作者简介:

余祖俊 (1968–), 男, 博士, 教授, 博士生导师, 目前研究方向为轨道交通控制与安全检测, E-mail: zjyu@bjtu.edu.cn;

王 尧 (1986–), 男, 博士研究生, 目前研究方向为机器视觉、图像处理、高速铁路基础设施检测, E-mail: wangyao1@bjtu.edu.cn;

朱力强 (1972–), 男, 博士, 副教授, 目前研究方向为轨道交通安全检测、智能信息处理、神经元网络与机器学习, E-mail: lqzhu@bjtu.edu.cn;

郭保青 (1978–), 男, 博士, 讲师, 目前研究方向为铁路基础设施检测与诊断, E-mail: bqguo@bjtu.edu.cn.