A Hardware/Software Co-design Approach for Real-Time Binocular Stereo Vision Based on ZYNQ (Short Paper)

Yukun Pan[1], Minghua Zhu[1(✉)], Jufeng Luo[2], and Yunzhou Qiu[2]

[1] Hardware/Software Co-Design Technology and Application Engineering Research Center, East China Normal University, Shanghai 200062, China
mhzhu@sei.ecnu.edu.cn
[2] Shanghai Internet of Things CO., LTD, Shanghai 201899, China

Abstract. Based on the ZYNQ platform, this paper proposes a hardware/software co-design approach, and implements a binocular stereo vision system with high real-time performance and good human-computer interaction, which can be used to assist advanced driver assistance systems to improve driving safety. Combining the application characteristics of binocular stereo vision, the approach firstly modularizes the system's functions to perform hardware/software partitioning, accelerates the data processing on FPGA, and performs the data control on ARM cores; then uses the ARM instruction set to configure the registers within FPGA to design relevant interfaces to complete the data interaction between hardware and software; finally, combines the implementation of specific algorithms and logical control to complete the binocular stereo vision system. The test results show that the frame rate with an image resolution of 640 * 480 can reach 121.43 frames per second when the FPGA frequency is 100M, and the frame rate is also high for large resolution images. At the same time, the system can achieve real-time display and human-computer interaction with the control of the graphical user interface.

Keywords: Binocular stereo vision system · Hardware/software co-design · Data interaction · Processing system · Programmable logic

1 Introduction

In recent years, autonomous driving and advanced driver assistance systems (ADAS) have become more and more important to reduce traffic accidents. The research on ADAS has reached a new craze under the promotion of computer vision and artificial intelligence. Binocular stereo vision is an important branch of computer vision [1]. The 3D scene perception method based on binocular stereo vision has many advantages in security, detection characteristics, cost and scope of application [2]. Although the accuracy of using radar for 3D scene perception is high in ADAS, the cost and difficulty of this approach are also high. Therefore, there are many researches applying binocular stereo vision to ADAS [3, 4]. The biggest advantage of binocular stereo vision is that it can achieve a certain accuracy of target recognition and ranging to realize some ADAS functions such as forward collision warning under the premise of

H. Gao et al. (Eds.): CollaborateCom 2018, LNICST 268, pp. 719–733, 2019.
https://doi.org/10.1007/978-3-030-12981-1_50

low development cost [5]. Therefore, the binocular stereo vision system has some research significance and application value.

In order to enable binocular stereo vision to assist ADAS, the binocular stereo vision system needs real-time performance and human-computer interaction. However, the relevant algorithms of binocular stereo vision are intensive to compute and complex to implement. The traditional CPU is good at logical control and human-computer interaction, but the CPU instruction set is slow to process complex algorithms and cannot achieve real-time performance. Most researchers often use FPGA to implement complex algorithms [6, 7], which can achieve high real-time performance. However, FPGA is good at parallel computing, but has poor control and management capabilities, which makes the existing single FPGA implementation system lack human-computer interaction. At present, with the continuous development and maturity of hardware/software co-design approach, the system using hardware/software co-design approach has both the advantages of hardware's high-speed computing and the advantages of software's control and interaction. Therefore, hardware/software co-design is an effective approach for the binocular stereo vision system with real-time performance and human-computer interaction. In [8], the author uses the hardware/software co-design and embedded system to realize real-time stereo vision and focus on reducing the complexity of the processing algorithm while maintaining the accuracy of disparity images. Compared with the current state-of-the-art pure FPGA implementation, it still has advantages in processing speed. But the system in [8] lacks human-computer interaction and does not fully exploit the advantages of hardware/software co-design approach. So how to effectively combine the advantages of CPU processor and FPGA to carry out hardware/software co-design to obtain the optimization of logical control and data processing is the problem to study for the binocular stereo vision system.

The hardware/software co-design means meeting system-level objectives by exploiting the synergism of hardware and software through their concurrent design [9]. In the early years, for the problems and challenges faced by embedded system design, some researchers explored a new design methodology—hardware/software collaborative design approach [10]. Its fundamental thought is to coordinate the hardware subsystem and software subsystem in the design process, so that each step is the optimal result of hardware and software considerations. Compared with the drawbacks of designing hardware and software architecture independently in traditional methods [11], the hardware/software co-design approach comprehensively analyzes the system functions and existing resources according to the system target requirements, and maximizes the concurrency between hardware and software to collaboratively design the system architecture so that the system can achieve the best performance [12]. Embedded systems using hardware/software co-design approach can have the advantages of high computational speed, high flexibility, low cost, low power consumption, and short development cycle [8]. Therefore, the hardware/software co-design is the best choice for some systems that require complex computations and good human-computer interaction. With the continuous development of technology, the main research object of hardware/software co-design has changed from a configurable embedded computer system to a single chip integrated with embedded systems [10]. ZYNQ (Xilinx All Programmable Zynq-7000 SoC) is an embedded processor chip that combines high-

performance ARM processor with high-capacity FPGA. It has both hardware and software programmable features. However, there are always some challenges in the hardware/software co-design on the ZYNQ platform [13]. Firstly, how to reasonably perform hardware/software partitioning can coordinate the software scheduling control and hardware data processing. Secondly, how to effectively manage data between hardware and software makes data to be processed quickly and accurately. Finally, how to design and implement the system's algorithms and logical control is the main problem in system construction on the ZYNQ platform.

This paper proposes a hardware/software co-design approach for real-time binocular stereo vision system. This approach solves these design difficulties on the ZYNQ platform well, and the binocular stereo vision system based on ZYNQ implemented by the co-design approach has high real-time performance and good human-computer interaction. By analyzing the application characteristics of the binocular stereo vision, the approach firstly modularizes the system's functions to perform hardware/software partitioning, then uses FPGA to implement the complex algorithms to accelerate data processing in the Programmable Logic (PL) of ZYNQ, and uses ARM processor to complete the image acquisition, program logical control, and image display in the Processing System (PS) of ZYNQ. Moreover, the approach uses the PS instruction set to configure the registers of the PL to design relevant interfaces to complete the data interaction between PL and PS, implements the system's algorithms and logical control, and fully utilizes the advantages of the ZYNQ heterogeneous multi-core processor [13] to complete the system construction, so that the stereo vision system achieves the best performance. The key contributions of this work include firstly proposing a hardware/software co-design approach for binocular stereo vision, and then correctly and efficiently realizing the data interaction between PL and PS, finally implementing the system based on the co-design approach and ensuring that the system has a certain speedup ratio and human-computer interaction in the actual test environment.

The rest of the paper is organized as follows. In Sect. 2, our hardware/software co-design approach is introduced. We describe the co-design of real-time binocular stereo vision system in Sect. 3. We implement the system and analyze experimental results and system performance in Sect. 4. And then conclude this paper in Sect. 5.

2 Hardware/Software Co-design Approach

The hardware/software co-design approach proposed in this paper solves the ZYNQ platform ARM+FPGA co-design difficulties from three aspects: hardware/software partitioning, data interaction between PL and PS and implementation of module algorithms and logical control, and builds a complete binocular stereo vision system.

The System Hardware/Software Partitioning. For binocular stereo vision, the system functions are modularized and each module is divided by considering the performance requirements, implementation cost, modifiability and nature of computation [11]. The data logical control is divided into the PS and controlled by ARM core. The complex data processing algorithms are divided into the PL and accelerated by FPGA. Such

hardware/software partitioning can make the system have high computational speed and good interactivity.

Efficient Data Interaction Between PL and PS. The system designs the hardware and software relevant interfaces to complete their data interaction. This is the core thought of the hardware/software co-design approach proposed in this paper.

Figure 1 shows the PL and the PS data interaction structure. In the ZYNQ platform, system can exchange information between PL and PS through the advanced extensible interface (AXI) bus. So we design the interface of data control and transmission for the PL and the PS to complete data interaction by the AXI bus. The control interface is designed by custom registers in the PL and the PS, which has a unique physical mapped address via the AXI4_Lite bus. The PL and the PS can access each other's registers through the system's general port AXI (AXI_GP) and physical mapped address, and then generate interactive signals by reading and writing registers' value to achieve the communication of the control interface. The data transmission interface is implemented in the PL by the system's high performance AXI (AXI_HP), and is implemented in the PS by the system's video for Linux 2 (V4L2) driver interface [13].

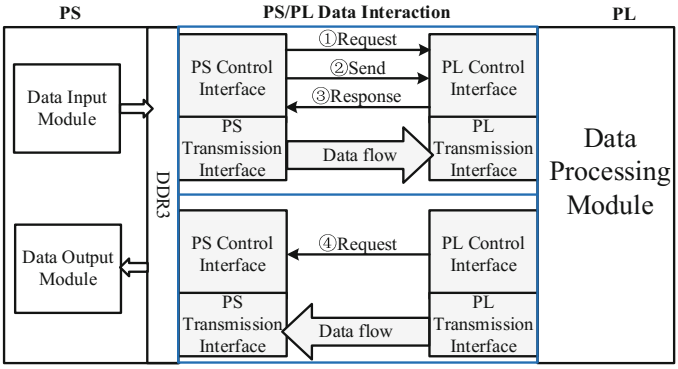

Fig. 1. Diagram of the PL and the PS data interaction structure

When the PS sends the acquired data from the double data rate (DDR3) memory to the data processing module of the PL, the PS control interface needs to request the module whether can receive data through the PL control interface, corresponding to ① in Fig. 1. After obtaining the module receivable data signal, the PS shall send the transmission signal to the PL through the PS control interface, and the PL shall give a response signal after receiving the signal through the PL control interface, corresponding to ② and ③ in Fig. 1. After a short delay waiting for the signal interaction to complete, the PL begins to receive the data of the PS through the data transmission interface. The significance of ② and ③ is to enable the system to correctly control and transmit one frame data. Otherwise, it will cause the system to lose data or receive the previous frame data during the data interaction. When the PL sends processed data to the PS, the PS control interface needs to apply the memory buffer and enable the stored signal, and then the PL control interface requests the PS control interface whether can

store the processed data, corresponding to the ④ in Fig. 1. When the PL gets the storable signal, it starts to transmit data through the transmission interface, and the PS stores the received data into the memory buffer. Due to the data processing module processes and transmits the data while receiving the data, the receiving and transmission of each frame data are parallel and continuous. Therefore, there is no need for ② and ③ in the data interaction from PL to PS. In this way, the data interaction of the system between PL and PS is completed efficiently.

Implementation of the Module Algorithms and Logical Control. The hardware designs the algorithms module's intellectual property (IP) core and data flow control between the IP cores. The software completes the data logical control, including data acquisition, storage and transmission. The entire system architecture is built through the implementation of the specific module algorithms and logical control. The binocular stereo visions system's modularization design is described in detail in Sect. 3.

Applying the hardware/software co-design approach to the binocular stereo vision system can efficiently complete hardware/software partitioning, and perform data interaction between PL and PS quickly and accurately. Combining the implementation of the module algorithm and logical control can ensure the system's real-time performance and human-computer interaction. The approach can also be applied to other embedded systems for machine vision and image processing, and has good universal property and application value.

3 The Proposed Co-design System

The process of designing and implementing the real-time binocular stereo vision system mainly includes camera acquisition, camera calibration, stereo rectification, stereo matching and 3D scene application. Figure 2 shows the flow chart of the system. Camera acquisition is the system's data source. Camera calibration requires the camera's internal parameters, external parameters and distortion parameters to lay the data foundation for subsequent modules [14]. The camera parameters of the same camera are fixed. Therefore, it is convenient to carry out camera calibration offline once and then write the value of these parameters to the system. In order to achieve the best performance of the system, this paper uses the mature algorithm principle of Zhang's calibration plate calibration method [15] and the MATLAB calibration toolbox for camera calibration [16, 17]. Stereo rectification is the strict alignment of pixels on the same line of binocular images [18]. This paper uses Bouguet's stereo rectification algorithm principle [19] and encodes it based on the relevant functions of the open source computer vision library (OpenCV). Stereo matching is a technique for recovering depth information from a planar image. It calculates the position difference between corresponding pixels according to the rectified binocular images, and forms a disparity image that can obtain depth information of the 3D scene. This paper uses the stereo matching algorithm which is designed by our research group based on AD census algorithm [20] and has a good representation in binocular stereo vision system. The disparity images obtained by stereo matching can be used for 3D scene application such as autonomous

vehicles and robot navigation. In the rest of this Section, we apply our approach described in Sect. 2 to introduce the co-design of binocular stereo vision system from system architecture, logic design of the PL and programming design of the PS.

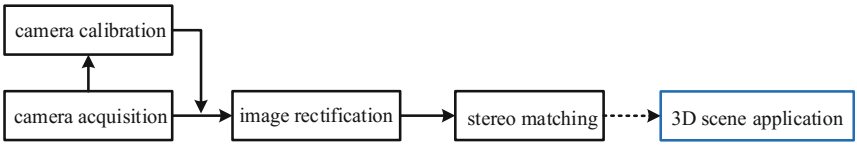

Fig. 2. A flow chart of binocular stereo vision system

3.1 System Architecture

Our goal is to design a real-time binocular stereo vision system. In order to ensure that the system has a certain speedup ratio and human-computer interaction, we combine the algorithms complexity and the characteristics of the ZYNQ platform, and use the hardware/software partitioning described in Sect. 2 to divide the system into the PL and the PS. Based on this work we designed the overall architecture of the system which is shown in Fig. 3. The system experiment platform mainly includes ZYNQ, DDR3 memory, USB driver-free binocular camera, high definition multimedia interface (HDMI) monitor, and an ordinary PC for auxiliary processing.

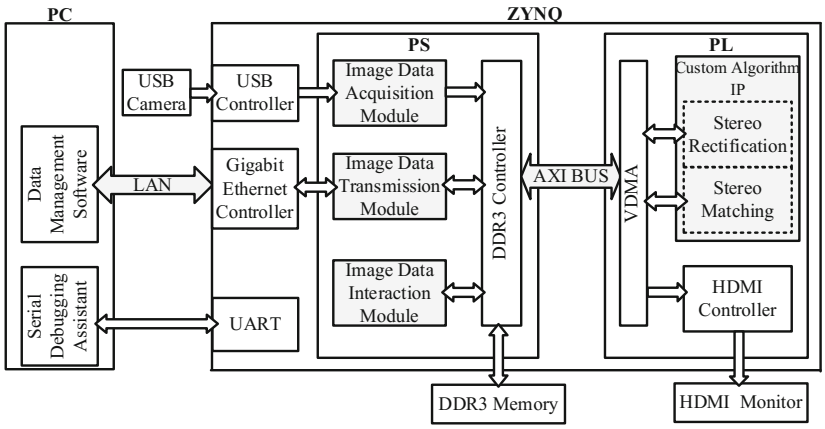

Fig. 3. Diagram of system architecture design

The PL utilizes all-purpose modular design idea in ZYNQ; it contains the following custom algorithm IP cores and Xilinx official IP cores at least:

Custom Algorithm IP. Because stereo rectification algorithm and stereo matching algorithm are complex in binocular stereo vision system, they are designed as the hardware IP cores to achieve parallel acceleration on FPGA. Because of Zynq-7000

obtains full support of Xilinx Vivado High-Level Synthesis (HLS) tool [21]. When some algorithms are designed with C language successfully, it is simple for HLS to synthesize the C language into Verilog and package into IP cores which can run on FPGA. Therefore, we can easily complete the design and synthesis of the hardware IP core through C language programming, shortening the hardware development cycle.

Data Transmission Controller. This module designed with AXI Video Direct Memory Access (VDMA) need to realize two functions. First, the image data captured by the PS is transmitted to the PL through the AXI_HP interface and corresponding control signals are generated to complete data control and interaction. Second, the processed image data by the PL is transmitted to the PS through the AXI_HP interface and corresponding control signals are generated to complete data control and interaction.

The PS needs to complete multi-task management and logical control in ZYNQ. Therefore, the following tasks need to be done in the PS:

Image Data Acquisition Module. Because the image data acquisition doesn't require complicated computation, the V4L2 video capture interface is used in the PS encoding to drive the USB controller to complete image data acquisition and storage.

Image Data Transmission Module. By using the User Datagram Protocol (UDP) communication in the PS encoding, the Ethernet port controller is driven to quickly transmit useful image data to the PC.

Image Data Interaction Module. The V4L2 drive interface is used in the PS encoding to control the VDMA of the PL to achieve the data interaction between PL and PS. This is an important module of the PS.

In addition, the system architecture also includes DDR3 memory for data storage via DDR3 controller, HDMI monitor for real-time image display and relevant control operation, and universal asynchronous receiver/transmitter (UART) interface for debugging the entire system through the serial debugging assistant.

It is worth noting that the communication of the control interface in the data interaction between PL and PS described in Sect. 2 are implemented by the instruction set in the image data interaction module configuring the registers in the Custom algorithm IP core. After the accurate communication of control signals, the AXI_HP interface executes high-performance data transmission through the VDMA of the PL.

3.2 Logic Design of the PL

Logic design of the PL mainly implements the data processing module algorithms described in Sect. 2, and completes the control of data flow in hardware part. Figure 4 shows the PL architecture of the system.

The PL architecture includes an algorithm processing acceleration channel and a real-time display channel. The DDR3 controller is used to control the reading and writing of image data in the DDR3 memory by the PL. In the algorithm processing acceleration channel, through the definition of the relevant registers in the stereo rectification IP core, the PS encoding can read and write the registers' value through the instruction configuration to achieve the communication of control signals during data interaction. This process corresponds to the PL control interface implementation of the

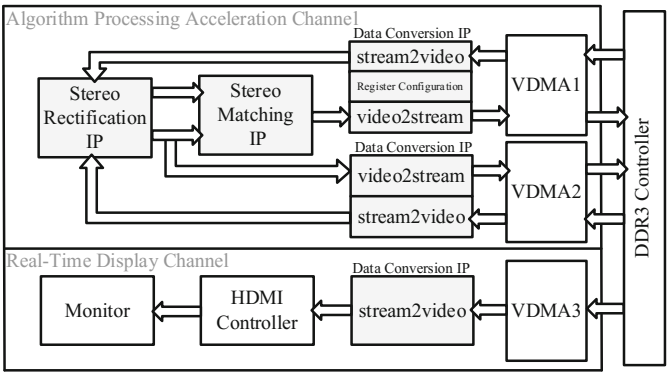

Fig. 4. Diagram of the PL architecture

data interaction between PL and PS described in Sect. 2. After correct communication, the image data in the DDR3 memory is converted into the stream data by the AXI_HP interface and the M_AXI_MM2S port of VDMA1 and VDMA2 respectively, then the data is passed to the custom data conversion IP core that converts the stream data into video data through the M_AXIS_MM2S port of VDMA, finally the data is passed to the stereo rectification IP core and stereo matching IP core. Due to the parameters such as offline calibration and image resolution are required during stereo rectification, there is corresponding register configuration sub-module in the data conversion IP core connected to VDMA1. After the video data is processed by stereo matching IP core, it is converted into the stream data through the data conversion IP core connected to VDMA1, and then sent to the VDMA1 through the S_AXIS_S2MM port, finally the stream data is converted into the memory mapped data by the VDMA1 through the M_AXI_S2MM port and is sent to the DDR3 memory by the AXI_HP interface. The stereo rectification IP core can send the processed left image of binocular images to the DDR3 memory through the data conversion IP core connected to VDMA2 and VDMA2 in the same process. In the real-time display channel, the image data in the DDR3 memory can be sent to the monitor through the VDMA3, the data conversion IP core and the HDMI controller with the same data processing method in the algorithm processing acceleration channel, and the monitor can real-time display image. It is worth noting that during the data transmission process of the PL, due to the bandwidth limitation of the DDR3, the image data in the DDR3 memory is intermittent. In order to perform data transmission correctly and efficiently between different IP cores, it is necessary to use FIFO which is a first-in, first-out data buffer and primarily used for data transfer between different clock domains for data buffering in data transmission. The entire line of image data is controlled by the FIFO buffer, so that the discontinuous data can be efficiently transmitted. Moreover, FIFO buffer reduces data interruption to quickly transmit and process data, and can effectively avoid data processing errors and data loss problems.

3.3 Programming Design of the PS

For completing the data logical control described in Sect. 2, The PS is designed to allow the user to control the execution of the system with simple button operation. In the architecture, the PS is mainly refined into the following modules as shown in Fig. 5: system initialization, image acquisition thread, data interaction thread, display control thread, and UDP processing thread. In order to achieve fast communication among threads, shared memory that implemented by memory mapping is used in the system to complete data resource sharing between threads.

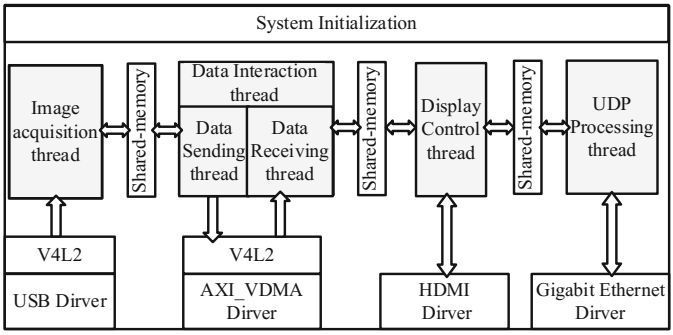

Fig. 5. Diagram of the PS architecture

System Initialization. This module mainly completes some system configuration before starting to execute each module. Firstly, the calibrated parameters are configured by using the memory mapping to write the parameters' value to register configuration sub-module of the data conversion IP core connected to VDMA1 in the PL. Secondly, because some interfaces of the PL such as USB capture interface, data input and output interface is made into a video device in the system. The video device can be operated to control the data transmission between PL and PS through the V4L2 video capture interface and the V4L2 video overlay interface. Therefore, the system initialization should have the relevant configuration of the video device. Finally, the system initialization requires some configuration such as socket definition and binding for UDP processing thread.

Image Acquisition Thread. This thread completes the binocular images data acquisition. The thread grabs the image data through the V4L2 video capture interface command VIDIOC_QBUF and puts it into the buffer queue that has been mapped to the user space, and then other threads fetch image data from the queue through the command VIDIOC_DQBUF. The loop operation of two matching interface command completes the real-time acquisition of image data.

Data Interaction Thread. The thread is divided into a data sending sub-thread and a data receiving sub-thread, corresponding to the PS control interface implementation of the data interaction between PL and PS in Sect. 2. In the PS encoding of the two sub-threads, the memory mapping is used to directly configure the custom registers of the

PL as control signals to complete the control of data interaction between PL and PS. In the data sending sub-thread, the PS encoding fetches the data from the DDR3 memory buffer to the VDMA through the V4L2 video overlay interface command VIDIOC_DQBUF, and then clears the buffer to store the next frame of image data through the command VIDIOC_QBUF. By the same principle, in the data receiving sub-thread, the PS encoding uses the command VIDIOC_DQBUF to write the received data into the DDR3 memory buffer through the AXI_HP interface and VDMA, and then clears the buffer to write the next frame of image data through the command VIDIOC_QBUF.

Display Control Thread. The thread completes the real-time display of image data and implements human-computer interaction through some buttons. The image display area and user control area of the Graphical User Interface (GUI) are designed by the PS encoding. The monitor is directly fetched the image data from DDR3 memory for display. Button operations mainly include the configuration of relevant registers during system initialization, and the control of the system's start and end.

UDP Processing Thread. The thread quickly transmits useful image data to the PC through the Ethernet driver interface, and makes data foundation for the later 3D scene application technology. In order to ensure the system's real-time performance, the thread uses a simple UDP communication for video data transmission. It is worth noting that since the maximum transmission unit is up to 1500 bytes under Ethernet, the thread requires packet sending and packet receiving for one frame of image data.

Before the end of the system, in order to ensure system security, the PS encoding needs to release some resources in system initialization, such as closing the socket, closing the video device and so on. In this way, the PS has a full structured system.

4 Implementation and Experimental Results

In order to successfully run the real-time binocular stereo vision system, this paper uses the ZC706 development board developed by Xilinx with ZYNQ7000 series as an embedded hardware board to implement the system. Figure 6 shows the system development board and peripheral interfaces. The USB interface needs to be connected to a USB free-drive binocular camera for image acquisition and peripheral keyboard and mouse for system human-computer interaction; the Ethernet interface is used to transmit data to the PC; the UART interface is used to connect the serial debugging assistant; the Joint Test Action Group (JTAG) interface is used for system co-simulation and testing; and the HDMI interface is connected to the monitor for real-time processing result display.

The system implements logic design of the PL through the Xilinx Vivado design tool, which can be used for IP core design and packaging, advanced synthesis and implementation of hardware architecture. Table 1 shows the consumption of the main system hardware resources in the PL. The consumption of the main development board resources is within the available range, which can meet the system execution requirements. The PL implements complex algorithms through FPGA parallel acceleration, which increases the system processing speed.

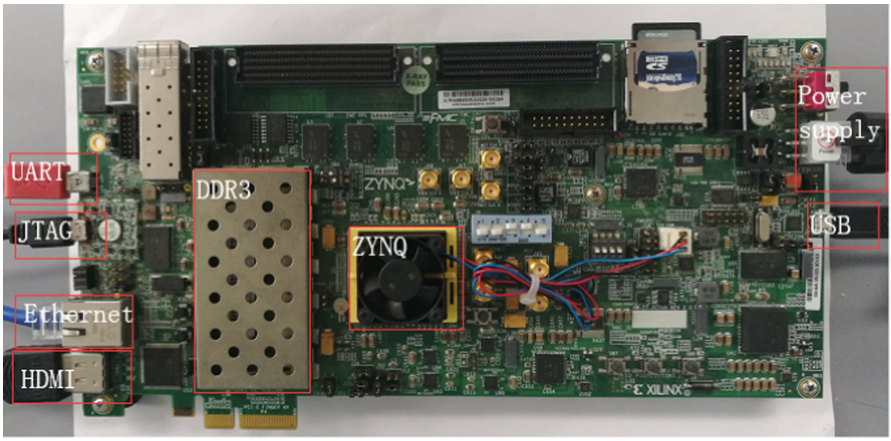

Fig. 6. System development board and peripheral interface

Table 1. Consumption of the system hardware resources

Resource	Utilization	Available	Utilization%
LUT	182797	218600	83.62
FF	171525	437200	39.23
BRAM	486	545	89.17
DSP	268	900	29.78

We use the QT designer tool based on the Linux operating system to complete programming design of the PS and GUI layout of this system, and transplant it to the embedded Linux system of the development board through cross-compilation. Then through the embedded Linux system's relevant configuration of the kernel driver, device tree and file system, the system's startup work is finally completed. Figure 7 shows the GUI of the system at runtime. The left part of the GUI is the image display area, including the original binocular images, the rectified image and the disparity image. The right part of the GUI is the user control area. The user can configure different cameras, image resolution and gain effect of the disparity image, and they also can control the execution of the system through the button. The GUI fully embodies the system's human-computer interaction performance.

In order to get the system's speed comparison between the architecture (FPGA 100 MHz, ARM 667 MHz) and ordinary CPU (2.50 GHz), We use binocular images with a resolution of 640 * 480 to test the processing time of stereo rectification algorithm and stereo matching algorithm several times. In our system architecture, the clock and counter are used to count the hardware processing time from the first line of one frame of image data into the PL to the last line of the frame of image data out of the PL. The software processing time is counted by using the program instruction to record the system's time difference in the PS. Table 2 shows the average processing time for the test image set. It takes nearly half a minute for the ordinary CPU to finish the two

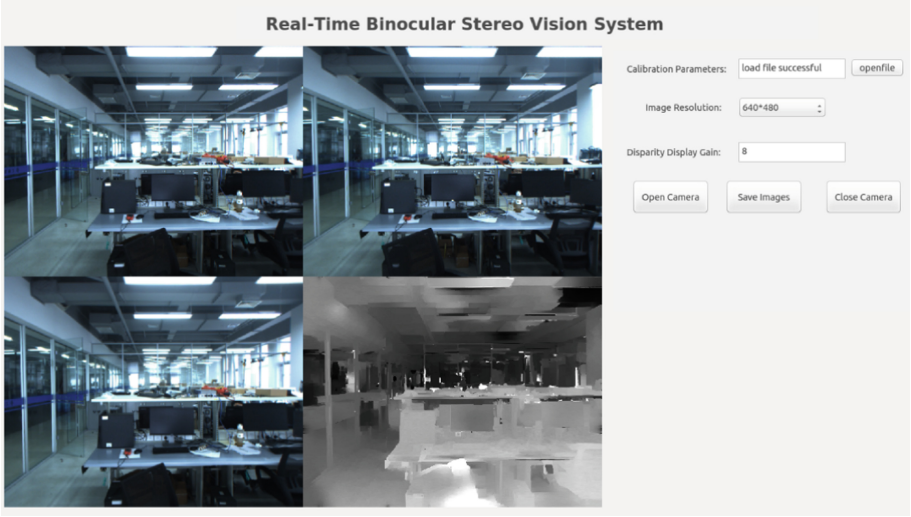

Fig. 7. The GUI of the system

algorithms. For the co-design approach of the system architecture in ZYNQ platform, the two algorithms are parallel acceleration processing. The logical control of the PS consumes 0.00473 s, and the data processing algorithms of the PL consumes 0.00350 s. The architecture only takes milliseconds to process one frame of image data, and which is thousands of times faster than the ordinary CPU. Such a high speedup ratio enables binocular stereo vision system to achieve real-time performance in actual scene application. In addition, in order to highlight the superiority of our co-design approach, we compare the system architecture in this paper and [8] in terms of processing time as shown in Table 3. It can be seen that the system architecture in [8] consumes less hardware resources, but the frame rate of the image is lower than that this paper, and our system also embodies human-computer interaction, so the hardware/software co-design approach proposed in this paper is superior to that in [8], which has high application value in the actual scene.

Table 2. Processing time of complex algorithms

Platform	Stereo rectification	Stereo matching	Total
CPU	0.231 s	29.750 s	29.981 s
ZYNQ	0.00473 + 0.00350		0.00823 s

In order to be able to further test the processing performance and execution efficiency of the system architecture, we tested frame rate through the different resolutions' binocular images. Due to the narrow bandwidth limitation of the development board's USB2.0, the camera's high-resolution transmission is not fast and the overall system's best performance cannot be tested. Therefore, we change the acquisition thread to store

Table 3. Comparison result of system processing time

Design	Device	Image size	Hardware resources	Frame rate
[8]	ZC706	640 * 480	182933 LUTs, 143223 FFs, 100 DSPs, 138 BRAMs	101
This paper	ZC706	640 * 480	182797 LUTs, 171525 FFs, 268 DSPs, 486 BRAMs	121.43

the different resolutions' binocular images in the startup card, and read these images through our co-design system for actual testing, and use the button on the GUI to control the system execution and change image resolution and other interactive work. The processing time is counted by recording the system's time difference from the completion of one frame of image data acquisition to the receiving of the processed frame of image data, and the average processing frame rate of the system is continuously measured by testing 100 frames, 1000 frames, and 10000 frames of image data. The test results are shown in Table 4. It can be seen that the frame rate with an image resolution of 640 * 480 can reach 121.43 frames per second, and the processing speed can exceed 30 frames per second for the full high definition of 1920 * 1080. If the system is applied to ADAS and combines with the signal processing control in the PS, it can help the driver to quickly make correct prediction of some driving operation commands. Therefore, the binocular stereo vision system has important significance in practical application.

Table 4. Image processing frame rate at different resolutions

Resolution	640 * 480	1280 * 720	1280 * 960	1920 * 1080
Frame rate	121.427	66.726	54.960	36.559

5 Conclusion

This paper has presented a hardware/software co-design approach for binocular stereo vision system. We firstly modularize the system's functions and combine the advantages of hardware and software to perform the hardware/software partitioning between modules, and then use the efficient data interaction to make the system have the best performance. Finally, we implement the system based on ZYNQ platform. The experimental results also show that the system has good human-computer interaction and can fully meet the real-time requirements. In the later research work, we plan to further optimize the system architecture and revelant algorithms, so that the processing speed of the system can be accelerated at high resolution, and make the system more perfect. In addition, we will apply the system's disparity images to real-time target tracking and distance measurement in real-world vehicle applications, enabling the system to truly contribute to the advanced driver assistance systems.

Acknowledgments. This work is supported by Shanghai Science and Technology Commission Project (17511106902), Shanghai Youth Science and Technology Phosphorus Project (No. 18QB1403900) and Shanghai Science and Technology Major Funding Project (No. 15DZ1100400).

References

1. Szeliski, R.: Computer Vision: Algorithms and Applications. Springer, New York (2010)
2. Woodfill, J.I., Gordon, G., Buck, R.: Tyzx DeepSea high speed stereo vision system. In: Conference on Computer Vision and Pattern Recognition Workshop, CVPRW 2004, p. 41. IEEE (2004)
3. Janai, J., Güney, F., Behl, A., Geiger, A.: Computer vision for autonomous vehicles: problems, datasets and state-of-the-art (2017). https://arxiv.org/abs/1704.05519
4. Bimbraw, K.: Autonomous cars: past, present and future a review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology. In: International Conference on Informatics in Control, Automation and Robotics, pp. 191–198. IEEE (2015)
5. Song, W., Yang, Y., Fu, M., et al.: Lane detection and classification for forward collision warning system based on stereo vision. IEEE Sens. J. **PP**(99), 1 (2018)
6. Jin, S., Cho, J., Xuan, D.P., et al.: FPGA design and implementation of a real-time stereo vision system. IEEE Trans. Circ. Syst. Video Technol. **20**(1), 15–26 (2010)
7. Pérez-Patricio, M., Aguilar-González, A.: FPGA implementation of an efficient similarity-based adaptive window algorithm for real-time stereo matching. J. Real-Time Image Process. 1–17 (2015)
8. Perri, S., Frustaci, F., Spagnolo, F., et al.: Design of real-time FPGA-based embedded system for stereo vision. In: 2018 IEEE International Symposium on Circuits and Systems (ISCAS), pp.1–5. IEEE (2018)
9. De Michell, G., Gupta, R.K.: Hardware/software co-design. Proc. IEEE **85**(3), 349–365 (1997)
10. Wolf, W.H.: Hardware-software co-design of embedded systems. Proc. IEEE **82**(7), 967–989 (1994)
11. Adams, J.K., Thomas, D.E.: The design of mixed hardware/software systems. In: Design Automation Conference Proceedings, pp. 515–520. IEEE (1996)
12. Edwards, S., Lavagno, L., Lee, E.A., et al.: Design of embedded systems: formal models, validation, and synthesis. Proc. IEEE **85**(3), 366–390 (1997)
13. Lu, J., Jiang, D., Ma, M.: Embedded system software and hardware cooperative design practical guide based on Xilinx ZYNQ. China Machine Press, Beijing (2013, Chinese)
14. Yan, Y., et al.: Camera calibration in binocular stereo vision of moving robot. In: The Sixth World Congress on Intelligent Control and Automation, WCICA 2006, pp. 9257–9261. IEEE (2006)
15. Zhang, Z.: Flexible camera calibration by viewing a plane from unknown orientations. In: The Proceedings of the Seventh IEEE International Conference on Computer Vision, vol. 1, pp. 666–673. IEEE (2002)
16. Bouguet, J.: Camera calibration toolbox for matlab (2004). http://www.vision.caltech.edu/bouguetj/calibdoc/
17. Fetić, A., Jurić, D., Osmanković, D.: The procedure of a camera calibration using camera calibration toolbox for MATLAB. In: 2012 Proceedings of the International Convention Mipro, pp. 1752–1757. IEEE (2012)

18. Tang, Y.P., Pang, C.J., Zhou, Z.S., et al.: Binocular omni-directional vision sensor and epipolar rectification in its omni-directional images. J. Zhejiang Univ. Technol. **1**, 20 (2011)
19. Bouguet, J.: The calibration toolbox for matlab, example 5: stereo rectification algorithm. Code and instructions only. http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/example5.html
20. Mei, X., Sun, X., Zhou, M., et al.: On building an accurate stereo matching system on graphics hardware. In: IEEE International Conference on Computer Vision Workshops, pp. 467–474. IEEE Computer Society (2011)
21. Zynq-7000 All Programmable SoC Overview DS190 (v1.2). Xilinx (2012)