

基于 Zynq 的视频实时拼接系统*

陈子为, 陈 龙, 朱美吉, 苏鲁阳

(成都信息工程大学 电子工程学院, 四川 成都 610225)

摘 要: 针对目前视频拼接系统实时性不高、拼接效果不理想、成本高等问题, 对传统的 ORB 算法进行了改进, 并采用软硬件协同设计的方法, 在 Zynq 平台上实现了一款视频实时拼接系统。该系统利用 Zynq 的 PS 搭建嵌入式 Linux 系统, 进行用户界面开发, 实现任务调度; 利用 Vivado HLS 工具将改进的视频拼接算法进行硬件加速并部署到 Zynq 的 PL 中。实验结果表明, 该视频拼接系统拼接后的图像清晰度高、无拼接缝隙, 而且基于 Zynq 和改进 ORB 算法的视频拼接方案使视频拼接速度获得了很大的提升, 满足实时性的要求。

关键词: Zynq; 视频拼接; ORB 算法; HLS

中图分类号: TN911.73; TP368

文献标识码: A

DOI: 10.16157/j.issn.0258-7998.200806

中文引用格式: 陈子为, 陈龙, 朱美吉, 等. 基于 Zynq 的视频实时拼接系统[J]. 电子技术应用, 2021, 47(2): 67-71.

英文引用格式: Chen Ziwei, Chen Long, Zhu Meiji, et al. Real-time video stitching system based on Zynq [J]. Application of Electronic Technique, 2021, 47(2): 67-71.

Real-time video stitching system based on Zynq

Chen Ziwei, Chen Long, Zhu Meiji, Su Luyang

(School of Electronic Engineering, Chengdu University of Information Technology, Chengdu 610225, China)

Abstract: Aiming at the existing problems of low real-time performance, unsatisfactory stitching effect and high cost in video stitching system at present, a real-time video stitching system was implemented on Xilinx Zynq platform using software and hardware collaborative design method after improving traditional ORB algorithm. The system used PS on Zynq to build an embedded Linux system, develop user interface programs, and implement task scheduling. The improving video stitching algorithm was packaged into an IP core that can be deployed into the PL on Zynq through Vivado HLS tools to achieve hardware accelerated processing. Experimental results showed that stitched video image had high clarity and no stitching gap, while this Zynq-based software and hardware collaborative processing architecture had greatly improved the speed of video stitching, which can meet the requirements of real-time processing.

Key words: ZYNQ; video stitching; ORB algorithm; HLS

0 引言

随着生产生活和科学技术的进步, 数码摄像设备得到了广泛地普及与应用。但是由于数码摄像设备自身物理条件的限制, 普通数码摄像设备获取视频的视野范围较小, 超广角镜头或者鱼镜头又会对图像产生畸变^[1], 不能满足实际的应用需求。利用单一摄像头捕捉多张视频画面合成宽视野图像的办法不仅费时费力, 而且获得的全景图像的质量一般不佳。视频拼接技术很好地解决了这一问题。所谓视频拼接就是将几个存在内容相关性的窄视野视频的每一帧图像进行拼接融合处理, 得到一幅宽视野的甚至是全景的视频图像。视频拼接本质上就是图像拼接, 但由于其对于实时性的要求很高, 使得在该领域的技术实践更需要关注平台的处理性能和算法

的复杂性。

目前市面上普遍采用基于 ARM、DSP、FPGA 的嵌入式平台方案, 以及多通道图像采集拼接形成全景的通用计算机平台方案来实现视频拼接。这些实现方案存在一定的缺陷, 基于 ARM 的实现方案处理能力较弱, 无法进行复杂视频算法; 基于 DSP 的实现方案处理速度比 FPGA 慢^[2]; 基于 FPGA 的实现方案灵活性较差且成本较高; 基于计算机平台的实现方案凭借性能优势获得好的效果, 但设备体积庞大、功耗高且价格昂贵^[3]。

针对以上问题, 本文首先对特征点检测效率表现优异的 ORB 算法进行改进, 并利用 Vivado HLS 工具将改进的视频拼接算法进行硬件加速并部署到 Zynq 的 PL 中, 然后利用 Zynq 的 PS 搭建嵌入式 Linux 系统, 进行用户界面开发, 实现任务调度, 最后提出了一种基于 Zynq 平台对多通道视频进行无缝拼接处理的解决方案。

* 基金项目: 四川省教育厅科学研究重点项目(18ZA0088); 四川省大学生创新创业训练计划项目(S201910621118)

1 算法设计

视频拼接流程如图 1 所示。首先通过摄像头采集两路视频图像, 然后进行视频图像帧同步, 接着再进行预处理, 预处理主要用于解决视频图像采集过程中引入的噪声、亮度差异调整等问题。预处理完成后进行图像的配准和变换, 其中图像配准是视频拼接中最关键、最耗时的步骤, 可以采用基于图像灰度统计特性、基于图像特征点和基于图像理解三种图像配准方式。综合考虑算法复杂度和配准准确率, 本设计采用基于特征点的方式进行图像配准。图像变换主要是解决图像配准后图像的内容拉伸变形的问题。为了消除图像拼接产生的拼接缝隙, 使其更加平滑、过渡更自然, 需要在图像配准和变换后进行图像融合。最后把处理过后的视频图像进行输出。

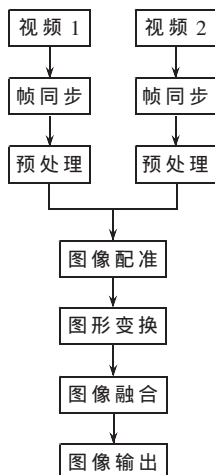


图 1 视频图像拼接流程图

1.1 基于 ORB 的图像配准算法

基于特征点的图像配准基本流程如图 2 所示, 视频图像采集完成后先进行特征点的检测, 然后对特征点进行描述, 得到特征点的集合后再进行特征点的匹配。在特征点的匹配过程中, 一般用特征点间的“距离”(例如欧式距离、汉明距离)来估算不同特征点之间的相似性度量(Similarity Measurement), 然后采用最短距离的方法寻找最佳匹配点对^[4]。完成特征点的检测和匹配后就完成了图像的配准工作。

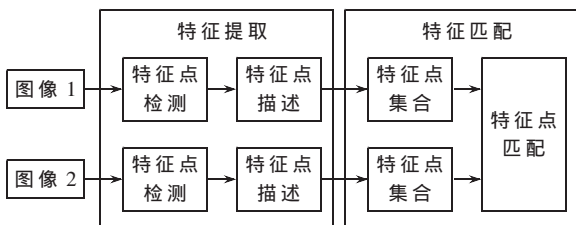


图 2 基于特征点的图像配准流程图

使用特征点来实现图像配准的方法比较多, 目前使用比较广泛的方法有 SIFT、SURF、ORB 等, 由于 ORB 速度上比 SIFT 和 SURF 快得多, 而性能上与两者不相上下^[5], 因而本文采用 ORB 算法作为视频拼接的配准算法。ORB(Oriented FAST and Rotated BRIEF)算法^[5]是一种快速的特征点检测与描述算法, 它基于 FAST(Features from Accelerated Segment Test)关键点检测^[6]和 BRIEF(Binary Robust Independent Elementary Features)^[7]描述符的组合, 具有旋转不变性和对噪声的鲁棒性。

1.1.1 ORB 特征点检测子

ORB 特征点检测采用的是 FAST 角点检测算法, 它

的核心思想就是在众多的像素点里找出与众不同的像素点。在灰度图像中, FAST 角点定义为: 任取一个像素点, 使其与邻域内的其他像素点进行比较, 如果有一定数量的点与其差别很大, 那么这个像素点就被选为角点^[8]。FAST 角点检测示意图如图 3 所示, 选取一个像素点 p (其灰度值为 I_p), 然后在像素点 p 的邻域以 r 为半径的圆周上找到 n 个像素点 I_k , 其中 $k=1, 2, 3, \dots, n$ 。那么 FAST 角点的判别表达式如式(1)所示:

$$CRF = \begin{cases} 1, & |I_p - I_k| > t \\ 0, & \text{其他} \end{cases} \quad (1)$$

式中 t 表示阈值。通常 $r=3, n=16$ 。

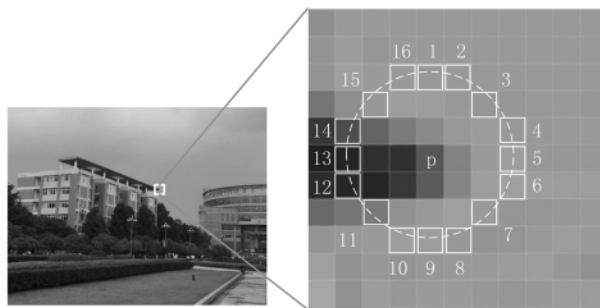


图 3 FAST 角点检测示意图

FAST 角点的具体计算过程是:

- (1) 在图像上选择一个像素点 p , 设其灰度值为 I_p 。
- (2) 设置一个合适的阈值 t , 如果两个像素点的差值大于阈值, 则认为这两个像素点不同。
- (3) 比较以 $r=3$ 时的圆周上的 16 个像素点。
- (4) 如果这 16 个像素点与像素点 p 有 m 个点不同, 这里 $m=12$, 那么这个点就是角点。
- (5) 使用一个高效的排除算法, 比如, 仅检查圆周上位置为 1、5、9、13 这四个位置的像素点, 可以先检查 1、9, 看 $CRF=1$ 是否成立, 然后再检查 5、13。如果是角点, 那么上述的四个像素点至少有 3 个应该满足 $CRF=1$, 如果不是这样, 那么这个点肯定不是一个角点。

FAST 角点检测要对比的像素点太多, 处理起来速度很慢, 其实检测一个点是不是角点, 没有必要把所有的点都进行对比。文献[9]中使用决策树算法构造一个角点分类器, 判断一个点是不是角点只需平均检测周围 3.8 个像素点即可。

1.1.2 ORB 特征点描述子

特征点描述子就是在特征点邻域内随机选取若干点对, 将这些点对灰度值的大小组合得到一个确定的二进制串, 这个二进制串就是该特征点的描述子^[10]。特征点随机点对如图 4 所示, 特征点 p 的邻域内存在若干点对, 图中以不同颜色作为不同点对的区分。ORB 特征点描述使用的是基于像素的二进制位比较的 BRIEF 特征描述子^[7], 求解 BRIEF 描述子的具体过程如下:

- (1) 选择其中一个特征点 p , 以特征点 p 为圆心, 在

其邻域内以 d 为半径画圆;

(2)在圆周内按照某一模式取个 n 点对,为了方便叙述这里取 $n=5$;

(3)标记这 5 个点,分别是 $p_1(A, B), p_2(A, B), p_3(A, B), p_4(A, B), p_5(A, B)$, A 和 B 分别表示点对两端的像素点;

(4)定义一种运算关系 T ,其表达式如式(2)所示:

$$T(P(A, B)) = \begin{cases} 1, I_A > I_B \\ 0, I_A \leq I_B \end{cases} \quad (2)$$

式中, I_A, I_B 分别表示 A 和 B 点的灰度值。

(5)对所有点对进行 T 运算,将得到的结果按照式(3)进行组合:

$$f_n(w) = \sum_{1 \leq i \leq n} 2^{i-1} T(w; x, y) \quad (3)$$

组合结果如式(4)所示:

$$\begin{cases} T(P_1(A, B))=0 \\ T(P_2(A, B))=1 \\ T(P_3(A, B))=1 \\ T(P_4(A, B))=0 \\ T(P_5(A, B))=1 \end{cases} \quad (4)$$

上式中得到的二进制串 01101 就是需要的 ORB 特征描述子。

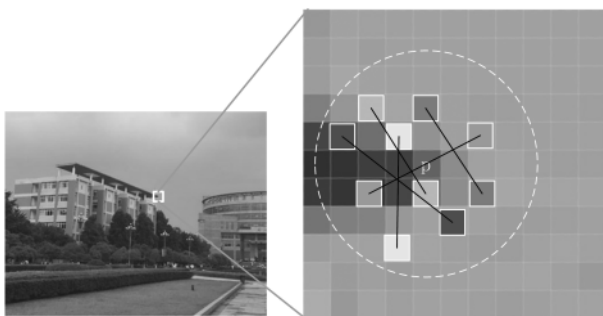


图 4 特征点随机点对示意图

1.2 算法改进

ORB 算法作为一种特征点检测与描述算法,在图像较小时效果还比较好,但是图像较大时消耗的时间较多,不适于视频配准的需求,因此对其进行改进。已知摄像头的安装位置,即输入视频流的相对位置是已知的和固定的。ORB 算法检测的特征点是针对于整幅图像的,运算量比较大,运算时间很长。既然知道了两幅视频帧图像的相对位置,就可以找出 ROI 区域(感兴趣区域),即如图 5 所示的两幅图中虚线内的图像部分,取该 ROI 区域的图像进行 ORB 运算,可大大减少运算量(此时的运算量可减少为原来的一半)^[11-12]。

因此改进的 ORB 检测流程如图 6 所示。

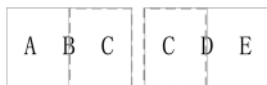


图 5 视频帧图像示意图

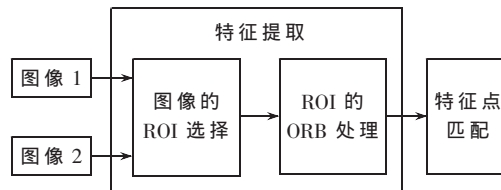


图 6 改进的 ORB 检测流程图

1.3 算法仿真

首先拍摄两张具有重叠区域的图片,如图 7 所示。



图 7 算法验证实验图

对图 7 中左右两幅图进行 ORB 算法检测特征点的匹配操作,效果图如图 8 所示。

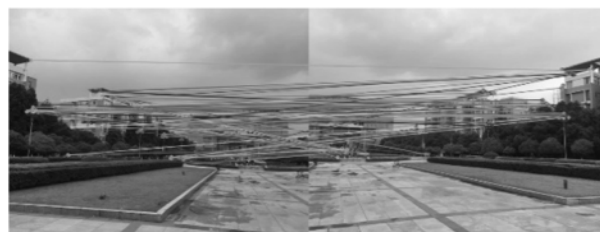


图 8 ORB 算法检测特征点匹配图

再利用改进的 ORB 算法进行仿真,实验效果如图 9 所示。

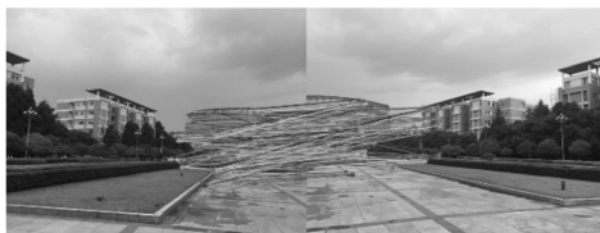


图 9 改进 ORB 算法检测特征点匹配图

从上面的实验结果对比可以看出,使用改进的 ORB 检测算法可以大大压缩传统的图像检测时间,提高了算法的检测效率,使得图像检测拼接技术可以应用于视频拼接。

2 系统设计及实现

2.1 系统构成

本系统基于 Xilinx 公司的 Zynq-7000 系列 AP SoC (All Programmable System on Chip, 全可编程的片上系统)

平台进行设计,在 FPGA 架构中集成了一个处理系统 PS (Processing System,即双核 ARM Cortex-A9 硬核)和一个等价于一片 FPGA 的可编程逻辑 PL(Programmable Logic)。本系统硬件上主要由视频采集、视频拼接处理和视频显示三部分构成,如图 10 所示。视频采集设备使用的是两路 500 万像素的 OV5640 摄像头。由于多个摄像头采集数据会有很大的吞吐量,因此利用 Vivado HLS 工具在 Zynq 的 PL 部分采用若干个 IP 核对视频数据进行流水采集(#pragma HLS dataflow 和 #pragma HLS PIPELINE II=1),并循环流水线核展开嵌套循环(#pragma HLS LOOP_FLATTEN),达到并行读取摄像头数据的目的,同时还利用 Vivado HLS 对视频拼接算法中的特征点检测进行硬件加速 IP 核封装设计,实现算法的高速并行处理。视频显示部分通过 VDMA 传输待显示的图像数据到 HDMI 接口进而传送到 LCD 显示器上进行显示。

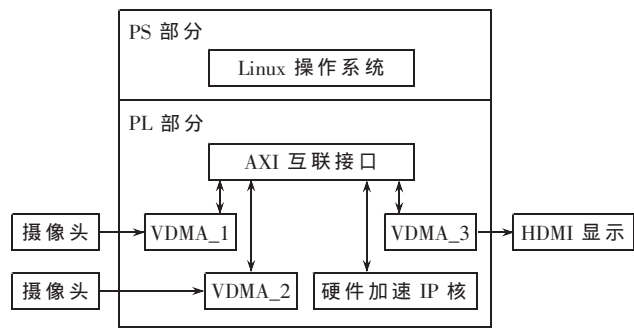


图 10 系统功能结构图

为了实现良好的人机交互和提升系统的灵活性,在 Zynq 的 PS 部分,搭建嵌入式 Linux 系统,移植部分 OpenCV 库和 Qt 库并进行程序的编译、配置等工作,完成视频图像拼接系统的软硬件协调及人机交互用户界面。

2.2 系统设计与实现

两个 OV5640 摄像头首先连接到 Zynq 的 PL 端,采用 Verilog 语言按照 SCCB 协议构造摄像头驱动模块,然后在 Zynq 的 PS 端按照标准的字符设备驱动方式采用 C 语言编写图像采集驱动程序,最后将摄像头模块连接在 VDMA 上通过 AXI4 总线传输数据。对于图像显示部分,VDMA 传输的显示数据连接到 HDMI 驱动显示模块,进而显示在显示器上。系统硬件实物如图 11 所示。

本文所提到的应用程序的用户界面是利用 Qt 软件开发的,包括显示源视频图像的窗口、输出结果的窗口、必要的功能按键等。开发的应用程序运行在基于 Zynq 平台的嵌入式 Linux 系统上。硬件设备上电操作之后,在控制台电脑上可以看到嵌入式 Linux 系统的启动信息。当 Linux 系统启动完成,可以通过命令对应用程序进行启动,之后使用鼠标对应用程序进行操作,右上角有相应的按钮,可以打开和关闭摄像头,可以完成视频流的图片截取,可以完成视频的拼接。应用程序进行视频拼接时的运行效果如图 12 所示。

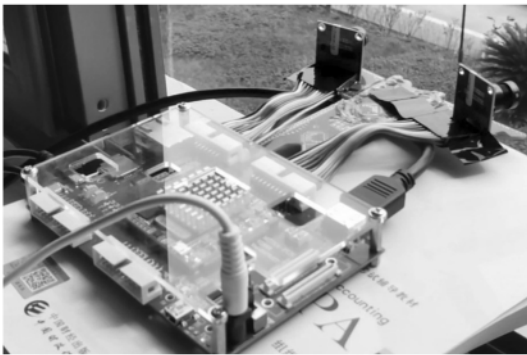


图 11 系统硬件实物图



图 12 系统应用程序运行图

3 性能分析

本文使用清晰度和空间频率作为视频拼接质量的客观评价标准^[13]。不同重合率拼接质量客观评价分析如表 1 所示。

表 1 不同重合率拼接质量客观评价表

重合率/%	清晰度	空间频率
20	5 301.4	15.134 5
40	4 487.83	17.080 7
60	3 825.57	16.810 0
80	4 999.77	15.739 3

从表 1 可以得出,不同重合率的视频图像的拼接对于拼接质量影响不同,重合率过小会导致图像无法配准,而重合率过大则导致配准后的图像信息量较少,拼接后的效果反而降低。因此要选择合适的重合率,一般以 30%~50%为宜。

再对拼接时效进行分析。实验输入的视频图像的分辨率为 640×480。每隔 0.5 s 取一帧图像,连续取 10 帧。记录了两组视频拼接时间的数据,一组是未使用 ROI 和未硬件加速的处理时间,如表 2 所示;另一组是在使用 ROI 改进算法并利用 FPGA 逻辑资源对耗时最多的特征点检测算法进行硬件加速的情况下的拼接处理时间,如表 3 所示。

通过对比两张表中的数据,可知使用 ROI 和硬件加速的拼接处理时间远远小于未使用 ROI 和未硬件加速的处理时间。主要体现在特征点检测和特征点匹配上,

表2 未使用 ROI 和未硬件加速的拼接时间

(ms)				
图像帧数	特征点检测	特征点匹配	图像变换	图像融合
第1帧	74.75	81.55	109.23	162.84
第2帧	73.17	75.20	107.71	154.76
第3帧	69.03	79.22	107.01	159.77
第4帧	76.55	77.87	111.69	156.19
第5帧	67.03	84.84	107.45	152.39
第6帧	66.14	84.30	114.10	160.60
第7帧	62.25	84.31	109.54	153.78
第8帧	66.46	80.51	113.26	161.84
第9帧	72.49	78.71	109.69	158.33
第10帧	64.56	79.77	117.96	160.32
均值	69.24	80.63	110.76	158.08

表3 使用 ROI 和硬件加速的拼接时间

(ms)				
图像帧数	特征点检测	特征点匹配	图像变换	图像融合
第1帧	4.15	66.36	110.34	159.89
第2帧	4.17	62.69	108.13	156.15
第3帧	3.83	63.15	106.96	157.14
第4帧	4.51	62.31	110.78	155.64
第5帧	3.78	65.64	108.11	157.25
第6帧	3.63	64.45	115.09	159.64
第7帧	3.82	65.14	108.89	157.01
第8帧	3.58	63.67	110.96	159.04
第9帧	4.13	63.96	111.12	156.95
第10帧	3.79	61.36	118.08	159.94
均值	3.94	63.87	110.85	157.87

特征点检测部分未使用 ROI 和未硬件加速的处理时间是使用 ROI 和硬件加速的处理时间的 17.57 倍,特征点匹配部分未使用 ROI 和未硬件加速的处理时间是使用 ROI 和硬件加速的处理时间的 1.26 倍。对比其他部分,没有明显差异。这是因为使用 ROI 算法后,特征点的搜索区域变为原来的 50%,减少了检测时间,降低了误匹配率。特征点匹配上得益于 ROI 算法,减少了特征点的分布区域,提高了特征点匹配的质量。

实验结果表明采用合适的重合率可以得到质量较好的拼接效果,使用 FPGA 硬件加速算法大大提高了算法的效率,减少了拼接的时间。

4 结论

本文针对 ORB 算法特征点检测进行了改进创新,实验证明改进后的算法在运行时间上减少了很多,满足了视频处理的实时性。对于利用 Vivado HLS 设计工具将算法进行 IP 核封装,实现硬件加速处理予以了验证,证明可以较大地提高算法的实时性。同时,本文采用 Zynq 平台设计视频拼接系统,对系统中各个模块进行了合理的任务分配,使得系统同时拥有 ARM 的灵活性以及 FPGA 的并行处理能力,充分地展现了并行异构架构的优势,

为机器视觉的小型化应用提供了重要参考。

参考文献

- [1] 王智.超广角镜头设计与畸变校正算法研究[D].杭州:浙江大学,2018.
- [2] 朱美能,李德华,金良海,等.基于多 DSP 的并行实时视频处理系统[J].计算机与数字工程,2007,35(8):41-44.
- [3] XU W, MULLIGAN J. Panoramic video stitching from commodity HDTV cameras[J]. Multimedia Systems, 2013, 19(5): 407-426.
- [4] 余道明.图像配准技术研究及应用[D].成都:西南交通大学,2011.
- [5] RUBLEE E, RABAU D V, KONOLIGE K, et al. ORB: an efficient alternative to SIFT or SURF[C]. 2011 IEEE International Conference on Computer Vision (ICCV). IEEE, 2011: 2564-2571.
- [6] ROSTEN E, DRUMMOND T. Machine learning for high-speed corner detection[C]. European Conference on Computer Vision. Springer-Verlag, 2006: 430-443.
- [7] CALONDER M, LEPETIT V, STRECHA C, et al. BRIEF: Binary robust independent elementary features[C]. European Conference on Computer Vision. Springer-Verlag, 2010: 778-792.
- [8] 刘亮,王平,孙亮.基于区域灰度变化的自适应 FAST 角点检测算法[J].微电子学与计算机,2017,34(3):20-24.
- [9] ALAHI A, ORTIZ R, VANDERGHEYNST P. Freak: fast retina keypoint[C]. 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012: 510-517.
- [10] 吉冬玉.基于 ORB 算法的图像匹配方法研究[D].西安:陕西师范大学,2016.
- [11] 姚泽烽,程显毅,谢璐.基于 ORB 和最小凸包的感兴趣区域检测方法研究[J].计算机应用研究,2018,35(10):3186-3188.
- [12] 应光林.基于双目视觉感兴趣区域的行人检测[J].信息通信,2018(3):48-50.
- [13] 韩德强,杨利平,王宗侠.基于 Zynq-7000 的宽幅视频处理系统设计与实现[J].电子技术应用,2016,42(6):59-62,66.

(收稿日期:2020-08-03)

作者简介:

陈子为(1978-),男,硕士,副教授,主要研究方向:嵌入式系统与 DSP、图像处理与识别、计算机视觉。

陈龙(1995-),男,硕士,主要研究方向:图像信息处理技术。

朱美吉(1994-),女,硕士,主要研究方向:图像信息处理技术。

版权声明

经作者授权，本论文版权和信息网络传播权归属于《电子技术应用》杂志，凡未经本刊书面同意任何机构、组织和个人不得擅自复印、汇编、翻译和进行信息网络传播。未经本刊书面同意，禁止一切互联网论文资源平台非法上传、收录本论文。

截至目前，本论文已经授权被中国期刊全文数据库（CNKI）、万方数据知识服务平台、中文科技期刊数据库（维普网）、DOAJ、美国《乌利希期刊指南》、JST 日本科技技术振兴机构数据库等数据库全文收录。

对于违反上述禁止行为并违法使用本论文的机构、组织和个人，本刊将采取一切必要法律行动来维护正当权益。

特此声明！

《电子技术应用》编辑部

中国电子信息产业集团有限公司第六研究所