基于 FPGA 的二维 FFT 图像边缘增强设计

任勇峰， 武昊男， 储成群， 焦新泉

(中北大学电子测试技术国家重点实验室，山西 太原 030051)

摘 要：在处理图像类信息时，图像细节往往能传达更多信息，是人们较为关注部分。针对在光照不理想的条件下，传感器采集到的图像对比度低、细节难以分辨的问题，提出一种基于现场可编程门阵列(FPGA)的二维快速傅立叶变换的图像边缘提取及增强方法。通过模块化设计，完成 4 路并行 512×512 点快速傅里叶变换(FFT)运算处理器设计，并通过 FFT 模块复用减少 FPGA 内资源消耗，同时实现图像频谱的高通滤波算法及傅立叶逆变换算法。经过仿真与实验，确定该方法有效可靠，实时性强，可以满足工业上图像处理的需求。

关 键 词：现场可编程门阵列；快速傅里叶变换；边缘增强；频域；滤波

中图分类号：TP 791 **DOI：**10.11996/JG.j.2095-302X.2019010137

文献标识码：A **文 章 编 号：**2095-302X(2019)01-0137-06

2D-FFT Image Edge Enhancement Design Based on FPGA

REN Yong-feng, WU Hao-nan, CHU Cheng-qun, JIAO Xin-quan

(National Key Laboratory for Electronic Measurement Technology, North University of China, Taiyuan Shanxi 030051, China)

Abstract: When processing the information like images, we find that image details can often convey more information, and that is why people are more concerned with images details. However, under the condition of unsatisfactory illumination, the images collected by sensors are usually with low contrast and difficult to distinguish the details. To solve the above problems, an image edge extraction and enhancement method based on field-programmable gate array (FPGA) is proposed. Four parallel 512×512-point fast Fourier transform (FFT) processors are designed by modular design, and the resource consumption in FPGA is reduced by multiplexing FFT modules. At the same time, high-pass filtering algorithm and inverse fourier transform algorithm of image spectrum are realized. The simulation and experiment show that the method is effective, reliable and real-time, and it can also meet the needs of industrial image processing.

Keywords: field-programmable gate array **(**FPGA); fast Fourier transformation (FFT); edge enhancement; frequency domain; wave filtering

图像增强是数字图像处理的重要组成部分，其目的是增强图像对比度，从而优化视觉效果。人眼对灰度变化的敏感度不是线性的，就 8 bit 灰度图片而言，人眼只对灰度值位于[48,206]区间的变化较为敏感[1]。为提高灰度图像的人眼视觉效果，人们提出许多相应算法。传统的提高灰度图像的对比度和可视效果的算法有许多，大致可分为空域类和频域类。空域类包括 Sobel、灰度变换、Canny 等[2]；频域类则包括小波变换、短时傅立叶变换、快速傅立叶变换等[3]。

收稿日期：2018-06-05；**定稿日期：**2018-09-12

基金项目：国家自然科学基金项目(61727804)

第一作者：任勇峰(1968–)，男，山西中阳人，教授，博士，博士生导师。主要研究方向为测试计量技术、电路与系统。E-mail：renyongfeng@nuc.edun.cn

通信作者：焦新泉(1968–)，男，江苏泰州人，副教授，博士，硕士生导师。主要研究方向为高速数据采集存储。E-mail：jiaoxinquan@nuc.edu.cn

通常情况下图像的特征在空域难以描述，但在频域却十分明显[4]，且就提高图像对比度而言，时频分析与频域分析并无太大区别，因此选择快速傅里叶变换(fast Fourier transform，FFT)算法作为本设计的核心算法。通常图像的边缘及细节部分由灰度值相差较大的部分构成，在频域中表现为高频部分[5]。因此用傅立叶变换的方法对图像进行边缘加强时，只需要提取图像频谱高频部分并加强即可。

所有的图像边缘增强算法有一个共同点——运算量巨大，通常处理时间较长，使用 CPU 进行运算耗时少则几百毫秒，多则几十秒，难以满足实时性的要求，而现场可编程门阵列(field-programmable gate array，FPGA)具有实时性强，数据吞吐量大的特点，非常适合处理图像类数据[6-7]，并且由于其并行运算的特点，可以成倍地提高 FFT 的运算速度[8-9]。因此，选择 FPGA 作为本设计的硬件平台，通过模块化设计，完成 4 路 2D-FFT 处理器的设计。

1 FFT 算法的 FPGA 基础

1.1 一维 FFT 及 IFFT

对离散信号 $x(n)$ $(0\leqslant n\leqslant N-1)$，其傅立叶变换为

$$X(k)=\sum_{n=0}^{N-1}x(n)e^{-i\frac{2\pi}{N}kn} \tag{1}$$

从式(1)可以看到，$x(n)$求傅立叶变换的过程即是对一组数据求积再求和的过程。FPGA 中有大量的加法器资源，可以方便地实现求和过程。但是乘法运算却会消耗大量的布线和时钟资源。因此要通过 FPGA 实现高效的傅立叶变换，必须减少其中的乘法运算。

蝶形运算的提出解决了上述的问题。蝶形运算可以将 N 点傅立叶变换转化为 2 个 $N/2$ 点的傅立叶变换，变换结果再求积与和的形式。如图 1 所示，将 1 个长序列分解为 2 个短序列的 FFT 运算被称为 Radix-2 FFT。

对于 $N=2^n$ 点序列，进行傅立叶变换需要进行 N^2 次乘法与加法，如果进行 n 次蝶形运算后，就可以将傅立叶变换转化为 $N\log_2 N$ 次求加法和乘法运算，极大减少了运算量。运算过程中任何一个节点的数值，仅与其前一级的 2 个节点有关，而与其他节点无关。因此 N 点傅立叶变换过程仅需 N 个寄存器，用来存储当前节点的数据，当下一个节点数据计算完毕后，覆盖寄存器即可。

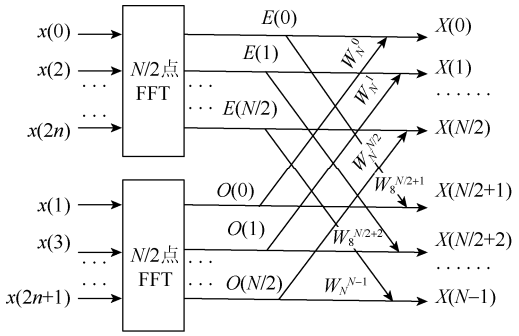


图 1 N 点蝶形运算

IFFT 是 FFT 的逆变换，即

$$x(n)=\frac{1}{N}\sum_{k=0}^{N-1}X(K)W_N^{-kn} \tag{2}$$

对比 FFT 的定义式，只有旋转因子不同，并且多了一个系数。Radix-2 FFT 序列长度 N 是 2 的整数次幂，在 FPGA 里通过右移寄存器操作可以简单实现。因此，在 FFT 的基础上实现 IFFT，只需要将逆变换的旋转因子更新进 ROM 中，再运算出结果后，将结果右移 $\log_2 N$ 位即可。

1.2 二维 FFT

分辨率 $N\times M$ 的图像，可以用一个 $N\times M$ 的矩阵 $\boldsymbol{X}(n,m)$表示，再经过二维 FFT 运算提取图像频谱。二维 FFT 可以看成 2 组一维 FFT 的组合，即

$$\begin{aligned}\boldsymbol{X}(k,l)&=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\boldsymbol{x}(n,m)e^{-i\frac{2\pi}{N}kn}e^{-i\frac{2\pi}{M}lm}\\&=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\boldsymbol{x}(n,m)W_N^{nk}W_M^{ml}\end{aligned} \tag{3}$$

式(3)可以理解为先对像素矩阵 $\boldsymbol{X}(n,m)$的每一行进行 M 点 FFT 运算，运算 N 次后得到 $\boldsymbol{F}(k,m)$，再对每一列进行 N 点 FFT 运算，运算 M 次最后得到 $\boldsymbol{X}(k,l)$。二维 FFT 运算满足可分性，即先进行行变换与先进行列变换对 FFT 运算结果没有影响。

2 硬件设计

选择 Xilinx XC6SLX150-3FGG676I FPGA 作为主控芯片。一帧 512×512 分辨率的 256 阶灰度图像数据量为 256 KB，图像缓存及运算中间数据缓存使用 DDR2 SDRAM 芯片 MT47H256M8。该芯片内部分为 8 个 Bank，每个 Bank 的容量为 256 Mbit，数据位为 8 位，地址位为 15 位。行列地址总线复用，行地址 15 位，列地址 10 位通过 RAS 和 CAS 信号激活行地址和列地址。部分电路如图 2 所示。

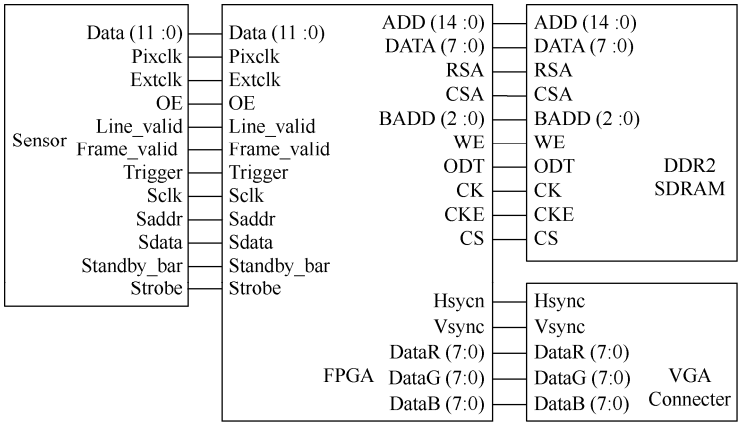


图 2　整体硬件电路

3　逻辑设计

本设计使用模块化设计方案，针对 512×512 分辨率的 256 阶灰度图像进行处理。图像的灰度由深到浅用十六进制可表示为 00~FF。图像传感器采集回一帧数据，由 FPGA 先存入 SDRAM 中，再读回 FPGA 进行处理，处理完毕后，直接通过 VGA 接口显示，流程如图 3 所示。

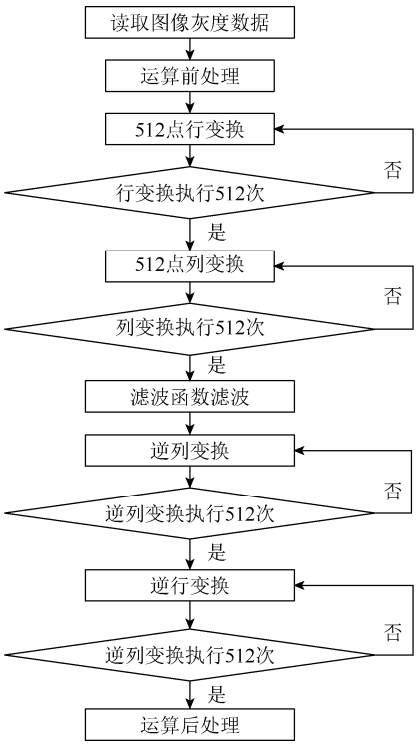


图 3　流程图

3.1　图像数据处理及滤波函数设计

通常图像经过傅立叶变换后，频谱的零点集中在频谱图的四角，不利于后续的滤波处理。为了将频谱原点移至图像中心，可在运算前对图像数据按式(4)进行预处理。得到一组新的图像数据 $\boldsymbol{Y}(m,n)$，即

$$\boldsymbol{Y}(m,n)=-1^{(n+m)}\boldsymbol{X}(n,m) \tag{4}$$

该方法在 FPGA 中只需一个加法器即可实现。步骤如下：

(1) 读取一个 8 bit 图像数据 $\boldsymbol{X}(n,m)$。

(2) 利用加法器计算 $n+m$。

(3) 将加法器输出结果的最低位作为该 8 bit 数据的符号位。

FFT 运算完成后，对频谱进行滤波，使用滤波函数 $\boldsymbol{H}(u,v)$乘以频谱，得到滤波后的频谱。

$$H(k,l)=\begin{cases}1, & d>400\\ 0, & d<400\end{cases} \tag{5}$$

式(5)中 d 为点 $\boldsymbol{F}(k,l)$到频谱中心点的距离的平方。滤波函数逻辑电路如图 4 所示。

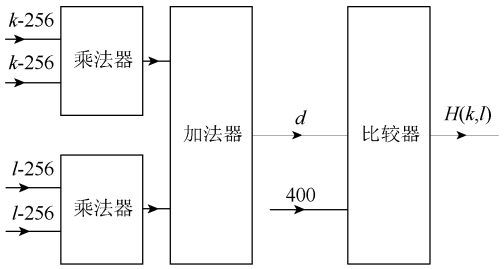


图 4　滤波函数逻辑电路

3.2　FFT 模块逻辑设计

FFT 运算单元主要控制信号及标志信号有 Start、Busy、DV、及 Xn_Index、Xk_Index。Start 信号表示一行或一列图像数据开始输入；Busy 表示 FFT 模块正在运算；DV 表示开始输出有效数据；Xn_Index、Xk_Index 信号是输入输出数据的计数，时序如图 5 所示。

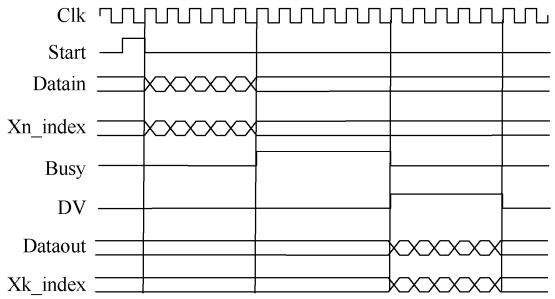


图 5 时序图

512 点 Radix-2 FFT 模块需要进行 9 级的蝶形运算，用到 256 个旋转因子。由于旋转因子在数据运算过程中可造成数据位增加并且出现小数，因此需对每级蝶形运算结果进行修正，提取有效数据的整数部分。为防止数据溢出，每级蝶形运算完成后将运算结果乘以缩减因子，缩减因子设置以数据不溢出为标准。旋转因子的比特数是决定快速傅立叶变换精度的重要因素，但是过高的比特数同样会造成运算量的增加，从而减慢运算速度，应当考虑精度和速度折中选择，本设计使用 16 bit 旋转因子。采用查找表的方式将其存储在FPGA片内ROM中，系统上电时自动进行初始化。单通道 FFT 模块原理如图 6 所示。

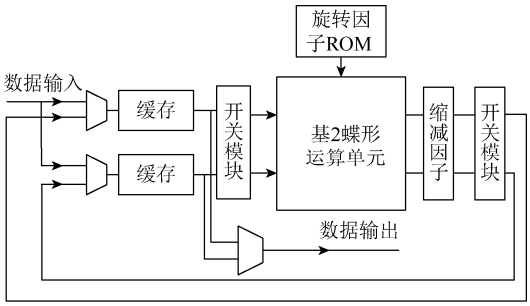


图 6 单通道 FFT

4 通道 FFT 由 4 个单通道 FFT 并行组成，4 通道同步执行 512 点 FFT 运算，输入数据为 64 位，其中每路 16 位，8 位实部、8 位虚部，对图像原始数据来说，虚部对应数值为 00。

当 IFFT 变换时，将 IFFT 的旋转因子加载入旋转因子 ROM 中。频谱数据乘以高通滤波函数后输入 FFT 模块，实现 FFT 模块的 IFFT 运算功能。极大节约了 FPGA 片内资源。在 IFFT 运算完成后，取结果的实部，进行后处理，处理方法与前处理相同，即可得到由频谱恢复后的图像。

3.3 DDR2 模块逻辑设计

在 HDL 设计中生成 8 个宽度为 8 bit，深度为 512 的 BRAM 作为待运算数据的缓存。8 个 BRAM 分为 2 组，一组缓存待运算数据实部，一组缓存待运算数据虚部，8 个 BRAM 由同一个使能信号控制，从而保证 4 路数据同步传输。并且由于数据缓存时间远小于 FFT 运算时间，每次 BRAM 中的数据输入 FFT 模块后立即从 DDR2 中读取下一行(列)的数据，用以节约时钟开销。

设计中 FFT 行与列变换处理单元复用，系统工作时先进行 4 路行变换，将计算结果 $F(n,l)$回传至 DDR2 中。128 次行变换执行完毕后，通过控制 DDR2 读数地址，读取 $F(n,l)$列数据缓存入 BRAM 中，重复 4 次后，将 4 组列数据同步输入 FFT 运算单元进行运算。每次 FFT 运算完成后，将 4 路结果经由一个 2 KB 的 FIFO 缓存至 DDR2 中，覆盖对应中间变量 $F(x,l)$，运算 128 次后得到二维 FFT 变换最终结果 $F(k,l)$。

总体逻辑设计如图 7 所示。

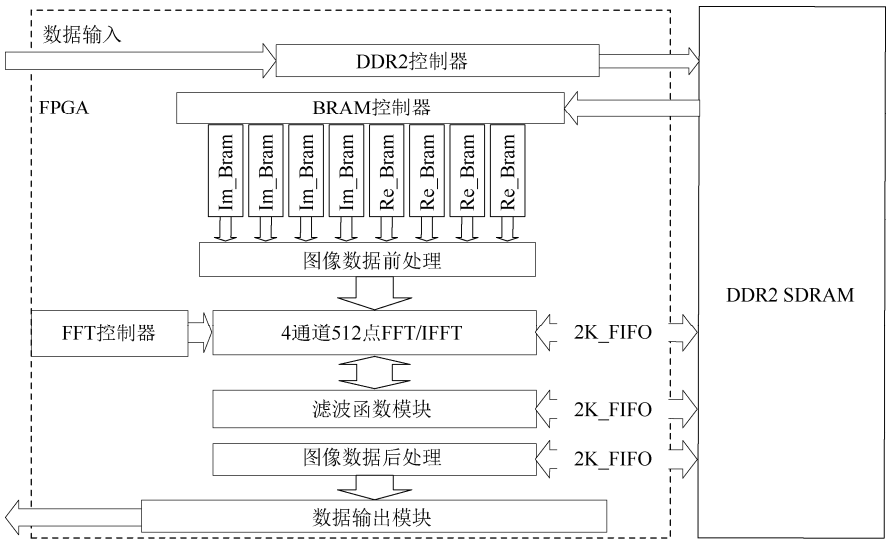


图 7 总体逻辑设计

4　实验结果及分析

为验证本设计的正确性，利用 ISE 对 4 路 FFT 运算单元进行仿真。仿真中通过输入自加数，在控制信号配合下，输出变换后的结果。在 Matlab 上做相同的 FFT 运算，以确定 FFT 运算单元的缩减因子。最后通过实验，实现一帧图像的频谱提取、滤波及反变换。

使用 Matlab 进行傅立叶变换，输入数据为 0~127，重复 4 次共 512 个数据得到部分结果如图 8 所示。

1	2	3	4	5
3.2512e+04+...	0.0000+0.0000i	0.0000+0.0000i	0.0000+0.0000i	−2.5600e+02+1.0428e+04i

图 8　Matlab 进行 FFT 运算部分结果

FPGA 平台的仿真结果如图 9 所示，Matlab 与 FPGA 的结果对比见表 1。

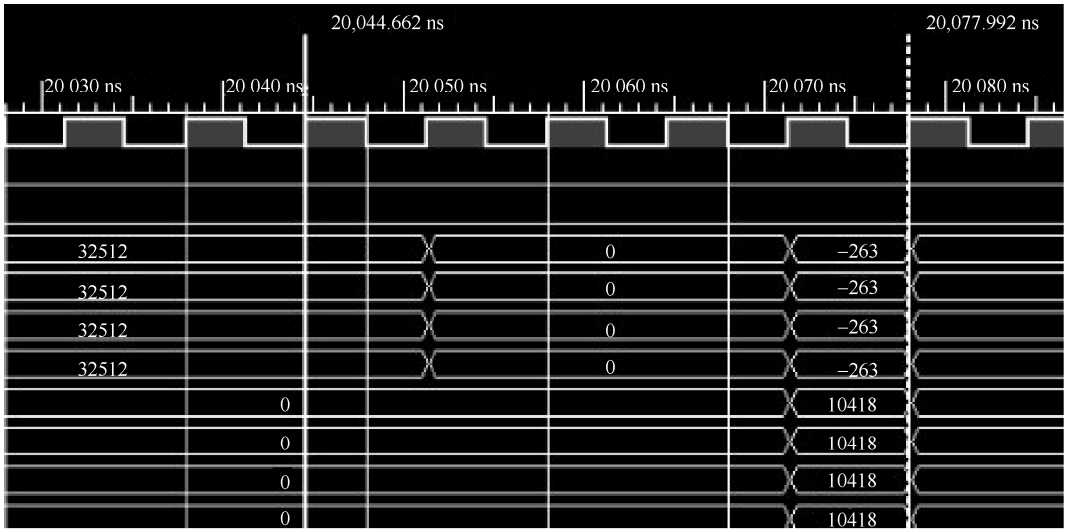


图 9　FFT 运算单元部分运算结果

表 1　部分运算结果

测试平台		运算结果				
FPGA	实部	32 512	0	0	0	−263
	虚部	0	0	0	0	10 418
Matlab	实部	32 512	0	0	0	−256
	虚部	0	0	0	0	10 428

表 2　运算时间

运算规模	使用平台	运算时间(ms)
512 点 FFT	Matlab	1.972 0
	FPGA	0.023 4
512×512 点 2D-FFT	Matlab	2 019
	FPGA	7.0130

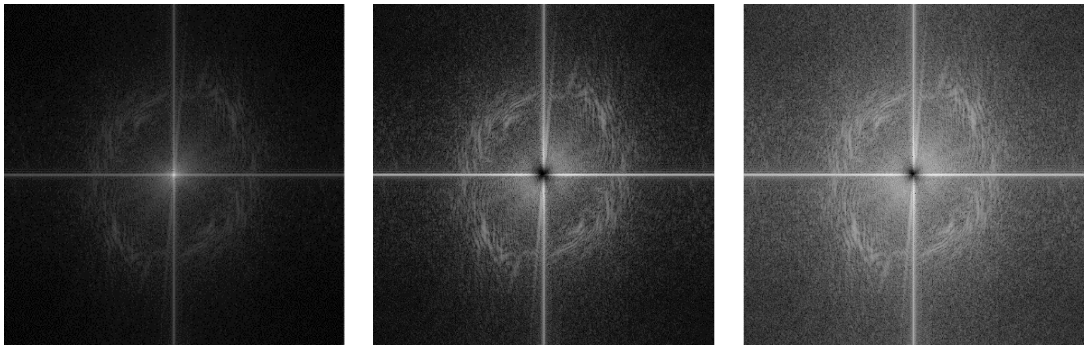
从表 1 可以看到 FFT 运算单元与 Matlab 的运算结果略有不同，这是因为 FFT 运算单元截断了运算过程中出现的小数。

通过 Matlab 的探查计时功能可以看到，进行一帧 512×512 像素图像 FFT 运算的时间约为 2.019 s；而基于 FPGA 的 4 路并行 FFT 运算单元进行一次 512 点 FFT 运算需要 3 505 个时钟周期，工作频率经过 PLL 锁相环后倍频在 150 MHz，进行一次 FFT 所需时间约为 23.367 μs。进行一次相同运算只需要约 7.013 ms 远远快于 Matlab 的运算速度，实时性得到很大提高。运算时间对比见表 2。

实验设计如下：利用 FPGA 平台对一帧指纹图像进行截止频率为 20 的数字高通滤波，输出频谱及其对应的图像。

将原始频谱进行高通滤波后，得到频谱高频部分，如图 10(b)所示。再经过 IFFT 运算后，得到图像纹理特征，如图 11(b)所示。

将图 10(b)的频谱数据左移 4 位后，得到增强频谱，如图 10(c)所示，反变换后得到边缘增强后的图像，如图 11(c)所示。对比图 11(a)和图 11(c)，可以看到处理过后的图像指纹清晰可辨，对比度明显提高。



(a) 原始频谱 (b) 滤波后频谱 (c) 增强后频谱

图 10 频谱对比

(a) 原始图像 (b) 滤波后图像 (c) 增强后图像

图 11 图像对比

5 结 束 语

本文在 FPGA 基础上从底层实现了一种二维 FFT 处理器，并应用在图像边缘增强领域，动态提高图像对比度，实现图像的频谱提取、滤波及由频谱提取图像的过程。结果显示该方法实时性高、稳定可靠，而且处理过的图像更加符合人眼的视觉特性，可以满足对图像处理实时性要求较高场合的应用需求。

参 考 文 献

[1] 康牧. 图像处理中几个关键算法的研究[D]. 西安: 西安电子科技大学, 2009.

[2] 吴诗婳, 吴一全, 周建江. 基于 NSST 和改进数学形态学的遥感图像目标边缘提取[J]. 图学学报, 2017, 38(4): 523-530.

[3] 刘丽, 匡纲要. 图像纹理特征提取方法综述[J]. 中国图象图形学报, 2009, 14(4): 622-635.

[4] 张勇. 傅里叶变换在数字图像处理中的应用[J]. 廊坊师范学院学报: 自然科学版, 2015, 15(3): 25-27.

[5] 周浦城, 周远, 韩裕生. 视频图像去雨技术研究进展[J]. 图学学报, 2017, 38(5): 629-646.

[6] 杨军, 于艳艳, 陈成, 等. 基于 FPGA 的二维 FFT 处理器的研究与设计[J]. 云南大学学报: 自然科学版, 2013, 35(6): 750-755.

[7] 张丽君. 大点数 FFT 的二维算法 FPGA 并行实现[J]. 无线电通信技术, 2013, 39(3): 86-88.

[8] 刘冀川. 实数二维 FFT 及其改进算法的 FPGA 实现[J]. 无线电通信技术, 2014, 40(3): 94-96.

[9] NGUYEN N H, KHAN S A, KIM C H, et al. A high-performance, resource-efficient, reconfigurable parallel-pipelined FFT processor for FPGA platforms [J]. Microprocessors and Microsystems, 2018, 60(7): 96-106.