数 字 信 号 处 理

课程设计

题目 FFT算法的matlab实现及应用研究

学院 电子信息工程学院

专业 通信工程

姓名 沈洋

学号 P21414031

**2016年12月29日**

**安徽大学电子学院通信工程系**

**目录**

**[一、设计目的 1](#_Toc29370)**

**[二、设计原理 1](#_Toc4542)**

**[三、步骤 1](#_Toc8747)**

**[四、自己的工作 2](#_Toc14889)**

**[1、fft函数的编写 2](#_Toc8525)**

**[1.1 原位计算 2](#_Toc32184)**

**[1.2 蝶形运算 2](#_Toc5476)**

**[1.3序列倒序 3](#_Toc24646)**

**[2、音频信号的处理 4](#_Toc16352)**

**[五、设计结果及总结分析 5](#_Toc15997)**

**[1.图像对比分析 5](#_Toc4054)**

**[2.数据分析 7](#_Toc1005)**

**[六、程序源代码 9](#_Toc16441)**

**[1.图像处理部分代码 9](#_Toc3791)**

**[2.音频信号处理代码 1](#_Toc28940)5**

**一、设计目的**

1. 加深对DFT算法原理和基本性质的理解；    
2. 熟悉FFT的算法原理和FFT子程序的算法流程和应用；   
3. 学习DSP中FFT的设计和编程思想；    
4. 熟悉FFT和IFFT在实际图像处理和音频处理中的效果；  
5. 掌握MATLAB编程，熟悉MATLAB的基本语法以及函数的调用；  
6. 学会运用MATLAB的信号处理功能，能够进行语音信号采集，并对语音信号进行变换以及还原处理，观察其时域和频域特性；  
7. 学会运用MATLAB进行图像信号采集，并对图像信号进行变换以及还原处理。

1. **设计原理**

本次课程设计我们的任务是分别用Matlab系统自带的基2fft函数和我们自己编写的基2fft函数来对图像进行处理和还原。

首先，我们知道，一张彩色图片是由一个三维数组组成，它们可以分成三个二维数组，分别代表图片每个点的R,G,B的值。所以，我们只需分别对每个二维数组进行fft处理即可。

然后，每个二维数组可以看成由x行个一维数组构成，每个一维数组的元素个数为y个（假设图片的长度为y，宽度为x）。所以我们对一个二维数组求基2fft变换，实际上只需对每个二维数组的每一行和每一列分别求一次一维的基2fft变换即可。

在进行一维数组的基2fft求解之前，必须保证数组的元素个数为2的n次方个，n为整数，所以在进行一维数组的基2fft变换前，应先将数组进行补0处理。所以，我们在对一幅图片进行fft变换前，先将图片进行了补0处理。即把原图片补成长和宽都为2的n次方个点，再进行fft变换。

1. **步骤**
2. 建立一个GUI界面，并填充相应的元素，如按钮，静态文本框等。构思好整

体效果，规划好每个按钮要实现的功能。

1. 右击打开图片按钮，查看回调函数编写打开图片的代码。
2. 同第二步，编写用系统自带fft函数处理图像并显示的代码。
3. 编写对经过fft函数处理过的图像进行ifft变换并显示的代码。
4. 开始自己编写一维数组的基2fft函数并测试所写代码的正确性。
5. 编写一维数组的ifft函数并测试代码的正确性。
6. 运行整个程序，查看程序的实际实现效果。
7. 添加对音频处理的界面及代码。
8. 不断改进所编代码。

**四、自己的工作**

我在本次课程设计中主要负责编写基2fft算法和编写调用自编的基2fft函数来对二维数组进行fft的操作的函数，以及对音频信号的处理。

**1、fft函数的编写**

要编写基2fft算法的运算程序，首先要分析它的运算规律，总结编程思想并画出程序框图。

**1.1 原位计算**

对N=2M点的FFT共进行M级运算，每级由N/2个蝶形运算组成。在同一级中，每个蝶的输入数据只对本蝶有用，所以每算完一个蝶后，所得数据可立即存入原输入数据所占用的存储单元。

**1.2 蝶形运算**

蝶形运算是分级进行的，每级的蝶形运算可以按旋转因子的指数大小排序进行。如果指数大小一样则可以从上往下依次蝶形运算。对N=2M点的FFT共有M级蝶形运算，用L表示从左到右的运算级数（L=1,2,...,M）。第L级共有B=2L-1个不同指数的旋转因子，用R表示这些不同指数旋转因子从上到下的顺序（R=0,1,...,B-1）。第R个旋转因子的指数P=2M-LR，旋转因子指数为P的第一个蝶的第一节点标号k从R开始，由于本级中旋转因子共有2M-L个，且这些蝶的相邻间距为2L，所以旋转因子指数为P的最后一个蝶的第一节点标号k为：(2M-L-1)\*2L+R=N-2L+R，本级中各蝶的第二个节点与第一个节点都相距B点。应用原位计算，蝶形运算可表示成如下形式：



总结上述规律，可采用如下算法。因为在调用此函数之前，我们已经给数组补过零保证了数组的长度为2的n（n为整数）次方，所以这里省去了数组补零操作。首先读入数据，得到数组长度N和运算级数M（M=log2（N））。

然后对数组进行倒序处理。倒序后从第1级开始逐级进行，共进行M级运算。在进行第L级运算时，先算出该级不同旋转因子的个数B=2L-1,同时，B也是该级中各个蝶形运算两输入数据的间距。再从R=0开始按序计算，直到R=B-1结束。每个R对应的旋转因子指数P=2M-LR，旋转因子指数相同的蝶从上往下依次逐个运算，各个蝶的第一节点标号k都是从R开始，以2L为步长，到N-2L+R结束，也可简取极值N-2结束。所以，数据倒序后的运算可以用一个三重循环程序来实现，图1为编写FFT2的程序框图。

**1.3序列倒序**

用MATLAB等高级语言实现倒序时，直接倒置二进制数位的方法不可取，需要找出产生倒序的十进制规律。将十进制顺序数用I表示，与之对应的二进制数用IB表示。十进制倒序数用J表示，与之对应的二进制数用JB表示，所以JB是IB的为倒置结果。若十进制顺序数加1，则相当于IB最低位加1且逢2向高位进1，即相当于JB最高位加1且逢2向低位进1.JB的变化规律反映到J的变化分两种情况：如果JB的最高位是0（J<N/2），则最高位直接加1（J=J+N/2）得到下一个倒序值；如果JB的最高位为1（J>=N/2），则要先将最高位变成0（J=J-N/2），再在次高位加1（J=J+N/4）。但次高位加1时，同样也要判断该位的0,1值，重复上面的过程，依次类推，直到完成最低位加1，逢2向右进位的运算。此方法可实现按顺序数I的递增顺序，依次求得与之对应的倒序数J。另外，当I=J时不需要交换，当I≠J时需要交换数据。为了避免再次调换前面已经调换过的一对数据，只对I<J的情况进行数据交换即可实现数据倒序操作。图2为数据倒序算法的程序框图。

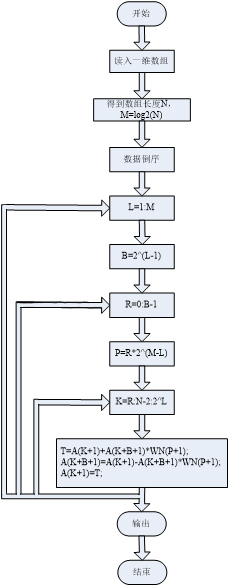
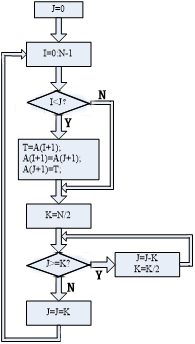
 

图1 基2fft算法程序框图 图2 数据倒序算法的程序框图

**2、音频信号的处理**

在本次课程设计的音频处理中，我又在原来的图像处理的界面上新建了一个音频界面，主要功能有播放和显示原音频信号的声音和波形，播放和显示经过调用系统fft函数和自编fft函数处理过的音频的声音和波形以及播放和显示调用系统自带ifft函数以及自编ifft函数处理过的音频的声音和波形。

主要流程就是先将音频信号存至一一维数组，再根据数组长度进行补0处理，接着就是调用系统fft函数以及自编fft函数对此一位数组处理，之后播放声音以及描绘它的波形。最后再调用系统ifft函数以及自编ifft函数对数组进行处理，再播放声音及描绘相应的波形图。

**五、设计结果及总结分析**

**1.图像对比分析**

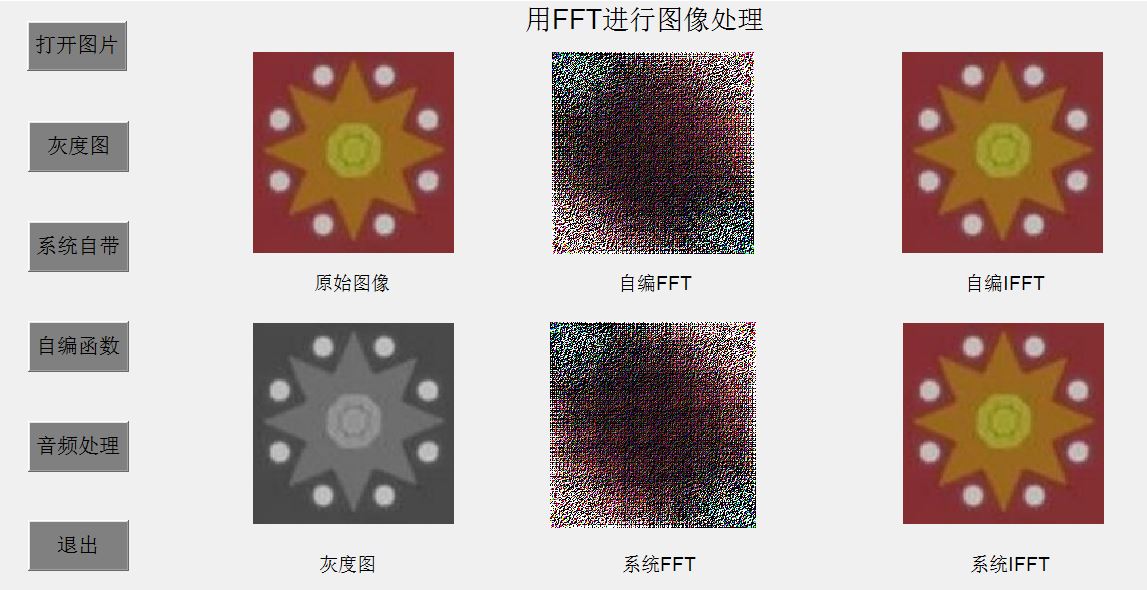


图3 图像处理整体运行结果

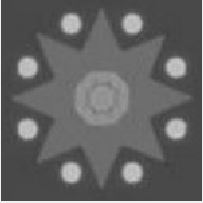
 

图4 原图 图5 灰度图

图6 自编fft效果 图7 系统自带fft效果

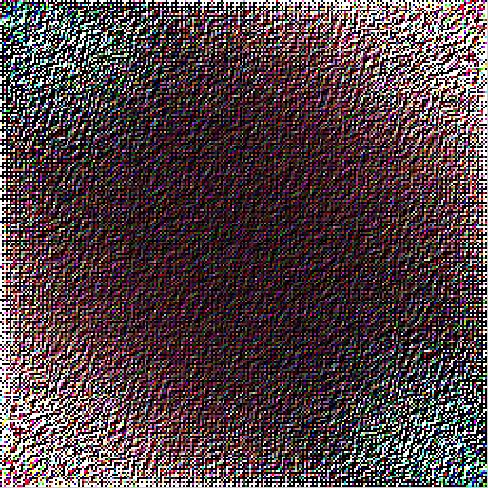
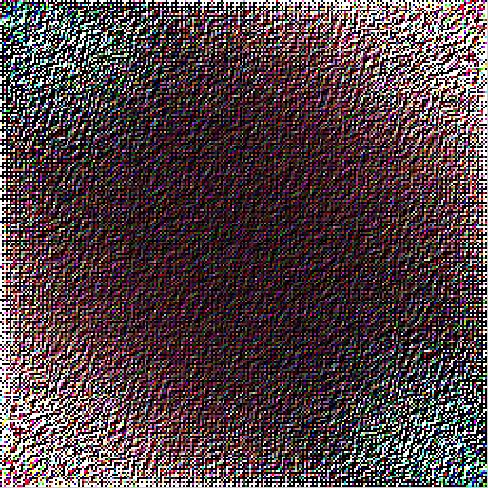
 

图8 自编ifft效果 图9 系统自带ifft效果

从运行结果来看，自编fft函数对图像的处理和系统自带的fft函数对图像的处理效果相同，说明程序编写的正确。在看自编的ifft函数和系统自带ifft函数对图像的还原，两者还原的图片也没有区别。说明编写的ifft函数也是正确的。

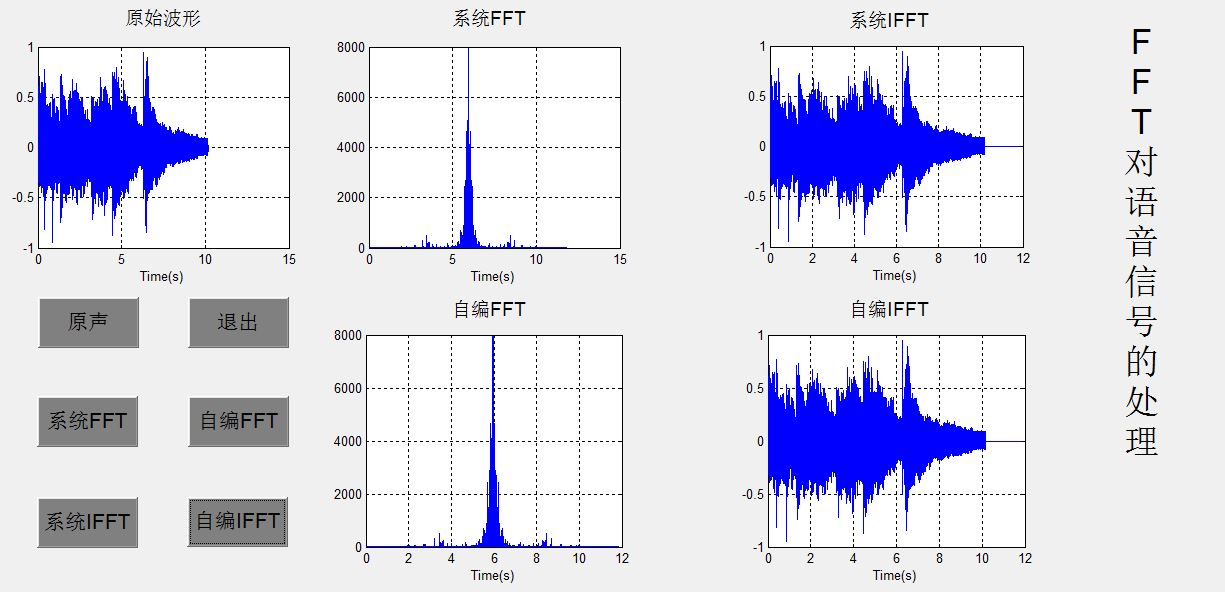


图9 音频处理整体运行结果

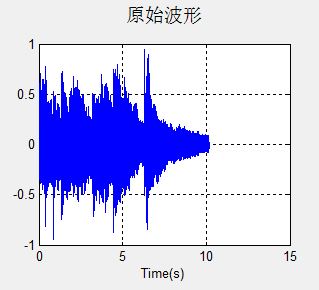


图10 原始波形

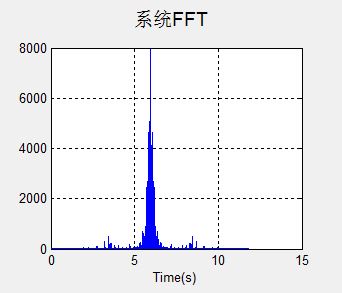
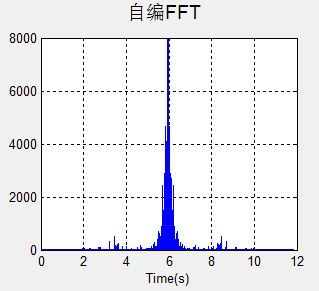
 

图11 系统FFT处理后波形 图12 自编FFT处理后波形

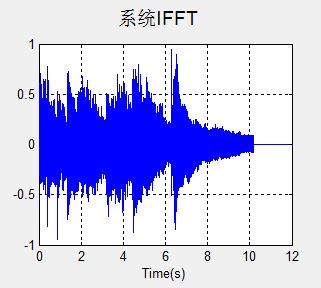
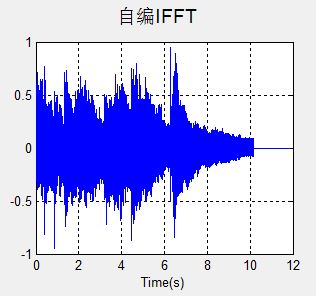
 

图13 系统IFFT处理后波形 图14 自编IFFT处理后波形

1. **数据分析**

因只观察图像并不能在理论上支持此算法编写的正确性，故我们在答辩后又在代码中加入了数据分析，结果截图如下：

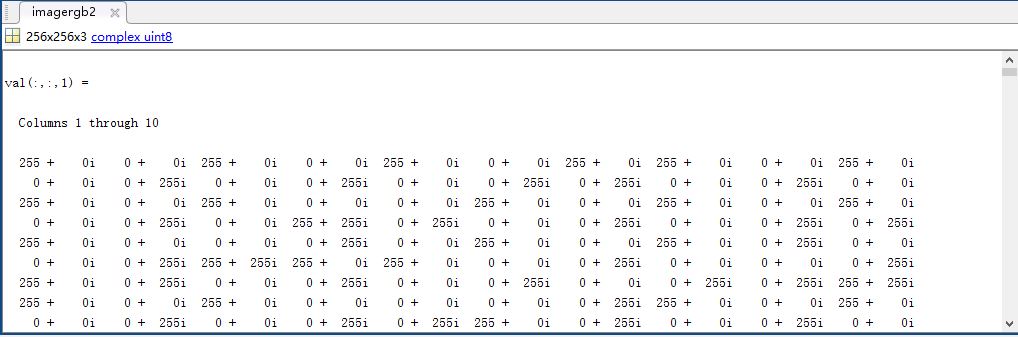


图15 系统fft数据截图

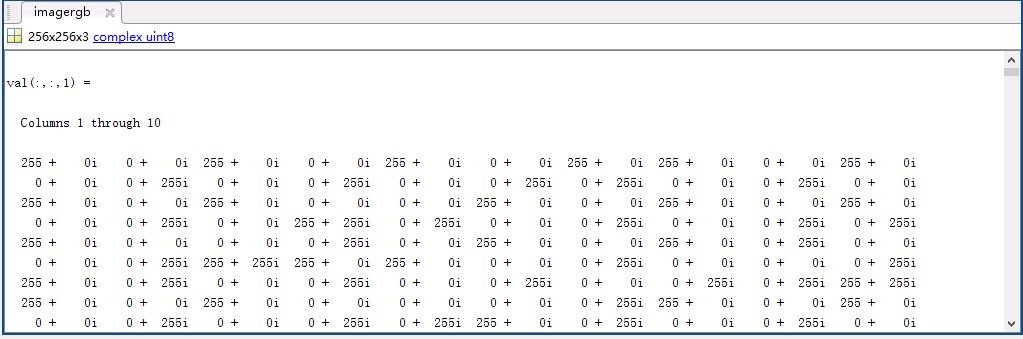


图16 自编fft数据截图

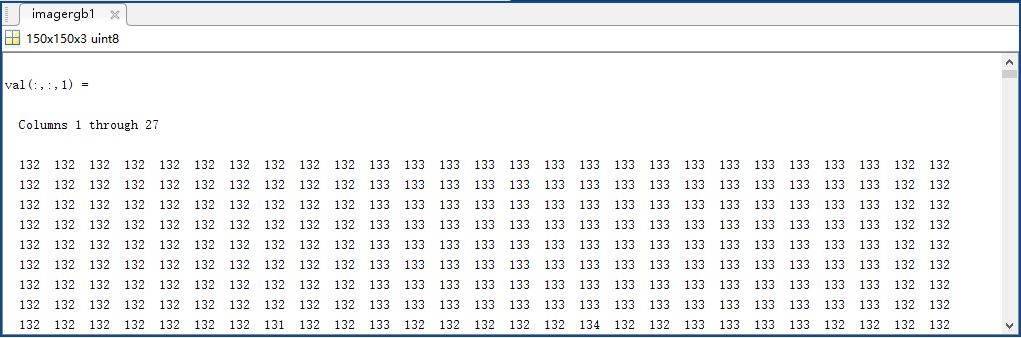


图17 系统ifft数据截图

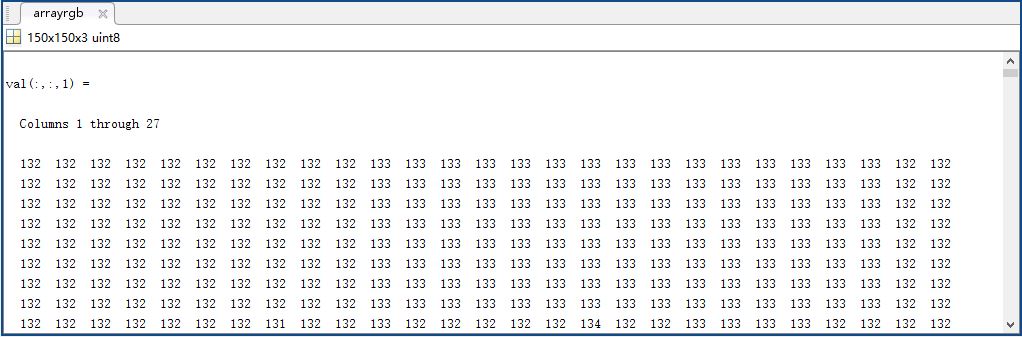


图18 自编ifft数据截图

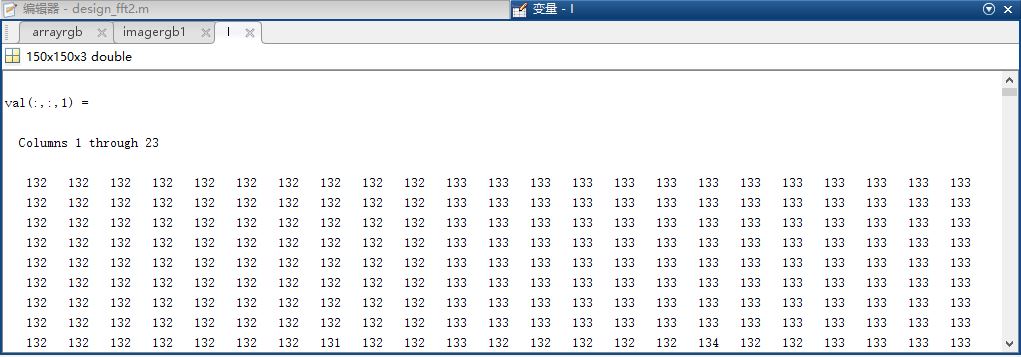


图19 原图数据截图

通过数据对比，我们发现自编的fft算法并没有问题，与系统自带的函数编出来的结果数据相同。

具体.mat数据文件会在附件中给出。

1. **程序源代码**

**1.图像处理部分代码**

function varargout = design\_fft2(varargin)

gui\_Singleton = 1;

gui\_State = struct('gui\_Name', mfilename, ...

'gui\_Singleton', gui\_Singleton, ...

'gui\_OpeningFcn', @untitled\_OpeningFcn, ...

'gui\_OutputFcn', @untitled\_OutputFcn, ...

'gui\_LayoutFcn', [] , ...

'gui\_Callback', []);

if nargin && ischar(varargin{1})

gui\_State.gui\_Callback = str2func(varargin{1});

end

if nargout

[varargout{1:nargout}] = gui\_mainfcn(gui\_State, varargin{:});

else

gui\_mainfcn(gui\_State, varargin{:});

end

function untitled\_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;

guidata(hObject, handles);

function varargout = untitled\_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

function pushbutton1\_Callback(hObject, eventdata, handles) %显示原图

[filename, pathname]=uigetfile({'\*.jpg;\*.tif;\*.bmp;\*.gif' },'File Selector');

image=imread(strcat(pathname,filename));

I=image;

I=double(I);

save image.mat I;

axes(handles.axes1);

imshow(image);

function pushbutton2\_Callback(hObject, eventdata, handles) %自编fft图像

load image.mat

image=I;

image1=image(:,:,1);

image2=image(:,:,2);

image3=image(:,:,3);

[r,c]=size(image1);

%补零

t=log2(r);

t1=floor(t);%t1为小于等于t的最大整数

t2=ceil(t); %t2为大于等于t的最小整数

if t1~=t2 %t1不等于t2

image1(2^t2,c)=0;

image2(2^t2,c)=0;

image3(2^t2,c)=0;

end

[r1,c1]=size(image1);

t=log2(c1);

t3=floor(t);

t4=ceil(t);

if t3~=t4

image1(r1,2^t4)=0;

image2(r1,2^t4)=0;

image3(r1,2^t4)=0;

end

%自编fft计算

image1s=transform\_fft2(image1);

image2s=transform\_fft2(image2);

image3s=transform\_fft2(image3);

image1a=uint8(image1s);

image2a=uint8(image2s);

image3a=uint8(image3s);

imagergb(:,:,1)=image1a;

imagergb(:,:,2)=image2a;

imagergb(:,:,3)=image3a;

save imagefftzibian.mat imagergb;

axes(handles.axes2);

imshow(imagergb);

figure(1);

imshow(imagergb);

array1=transform\_ifft2(image1s);

array2=transform\_ifft2(image2s);

array3=transform\_ifft2(image3s);

%matlab为图像提供了特殊的数据类型uint8(8位无符号整数），以此方式存储的图像称作8位图像。

array1=uint8(array1);

array2=uint8(array2);

array3=uint8(array3);

%截取原图片长度和宽度

array1=array1(1:r,1:c);

array2=array2(1:r,1:c);

array3=array3(1:r,1:c);

arrayrgb(:,:,1)=array1;

arrayrgb(:,:,2)=array2;

arrayrgb(:,:,3)=array3;

save imageifftzibian.mat arrayrgb;

axes(handles.axes3);

imshow(arrayrgb,[]);

figure(3);

imshow(arrayrgb);

function pushbutton3\_Callback(hObject, eventdata, handles) %系统fft图像

load image.mat

image=I;

image1=image(:,:,1);

image2=image(:,:,2);

image3=image(:,:,3);

[r,c]=size(image1);

t=log2(r);

t1=floor(t);%t1为小于等于t的最大整数

t2=ceil(t); %t2为大于等于t的最小整数

if t1~=t2 %t1不等于t2

image1(2^t2,c)=0;

image2(2^t2,c)=0;

image3(2^t2,c)=0;

end

[r1,c1]=size(image1);

t=log2(c1);

t3=floor(t);

t4=ceil(t);

if t3~=t4

image1(r1,2^t4)=0;

image2(r1,2^t4)=0;

image3(r1,2^t4)=0;

end

image1=fft2(image1);

image2=fft2(image2);

image3=fft2(image3);

image1a=uint8(image1);

image2a=uint8(image2);

image3a=uint8(image3);

imagergb2(:,:,1)=image1a;

imagergb2(:,:,2)=image2a;

imagergb2(:,:,3)=image3a;

save imagefftxitong.mat imagergb2;

axes(handles.axes4);

imshow(imagergb2);

figure(2);

imshow(imagergb2);

image1=ifft2(image1);

image1=uint8(image1);

image2=ifft2(image2);

image2=uint8(image2);

image3=ifft2(image3);

image3=uint8(image3);

image1=image1(1:r,1:c);

image2=image2(1:r,1:c);

image3=image3(1:r,1:c);

imagergb1(:,:,1)=image1;

imagergb1(:,:,2)=image2;

imagergb1(:,:,3)=image3;

save imageifftxitong.mat imagergb1;

axes(handles.axes5);

imshow(imagergb1);

figure(4);

imshow(imagergb1);

return

function pushbutton4\_Callback(hObject, eventdata, handles) %关闭

clc;

close all;

close(gcf);

function pushbutton5\_Callback(hObject, eventdata, handles) %灰度图

load image.mat

image=I;

image=uint8(image);

if ndims(image)==3

image=rgb2gray(image);

end

axes(handles.axes6);

imshow(image);

figure(5);

imshow(image);

function array1=transform\_fft(array) %对N位一维数组进行FFT

N=length(array);

M=log2(N);

WN=zeros(1,N,'double');

for m=0:N/2-1;

WN(m+1)=exp(-1i\*2\*pi/N)^m;

end

J=0;

for I=0:N-1

if I<J

T=array(I+1);

array(I+1)=array(J+1);

array(J+1)=T;

end

K=N/2;

while J>=K

J=J-K;

K=K/2;

end

J=J+K;

end

for L=1:M

B=2^(L-1);

for R=0:B-1

P=2^(M-L)\*R;

for K= R:2^L:N-2

T=array(K+1)+array(K+B+1)\*WN(P+1);

array(K+B+1)=array(K+1)-array(K+B+1)\*WN(P+1);

array(K+1)=T;

end

end

end

array1=array;

return

function array=transform\_fft2(array) %对二维数组进行FFT

array=double(array);

[r1,c1]=size(array);

for j=1:r1 %对每一行进行FFT

array(j,:)=transform\_fft(array(j,:));

end

for j=1:c1

array(:,j)=transform\_fft((array(:,j)));

end

function array=transform\_ifft2(array) %对二维数组进行IFFT

array=conj(array);

[r1,c1]=size(array);

for i=1:c1

array(i,:)=transform\_fft(array(i,:));

end

for i=1:c1

array(:,i)=transform\_fft(array(:,i));

end

array=conj(array);

array=array/(r1\*c1);

function pushbutton15\_Callback(hObject, eventdata, handles) %音频处理

open('yinpin.fig');

1. **音频信号处理代码**

function varargout = yinpin(varargin)

gui\_Singleton = 1;

gui\_State = struct('gui\_Name', mfilename, ...

'gui\_Singleton', gui\_Singleton, ...

'gui\_OpeningFcn', @yinpin\_OpeningFcn, ...

'gui\_OutputFcn', @yinpin\_OutputFcn, ...

'gui\_LayoutFcn', [] , ...

'gui\_Callback', []);

if nargin && ischar(varargin{1})

gui\_State.gui\_Callback = str2func(varargin{1});

end

if nargout

[varargout{1:nargout}] = gui\_mainfcn(gui\_State, varargin{:});

else

gui\_mainfcn(gui\_State, varargin{:});

end

function yinpin\_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;

guidata(hObject, handles);

function varargout = yinpin\_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

function pushbutton1\_Callback(hObject, eventdata, handles) %原声

handles = guihandles(gcf);

[filename,pathname]=uigetfile({'\*.wav'},'选择语音信号');

str=[pathname filename];

[y,Fs]=wavread(str);

y=y(:,1);

Y=y;

save y.mat Y;

FS=Fs;

save Fs.mat FS;

sound(y,Fs);

t=(0:length(y)-1)/Fs; %计算时间轴

axes(handles.axes1);

plot(t,y);xlabel('Time(s)');%在第一个窗口画波形

grid on;

function pushbutton2\_Callback(hObject, eventdata, handles) %系统FFT

handles = guihandles(gcf);

load y.mat;

y=Y;

load Fs.mat;

Fs=FS;

r=length(y);

m=log2(r);

m1=floor(m);%t1为小于等于t的最大整数

m2=ceil(m); %t2为大于等于t的最小整数

if m1~=m2 %t1不等于t2

y(r:2^m2)=0;

end

r=length(y);

t=(0:r-1)/Fs; %计算时间轴

y=fft2(y);

y=fftshift(y);

y=abs(y);

sound(y,Fs);

axes(handles.axes2);

plot(t,y);xlabel('Time(s)');%在第一个窗口画波形

grid on;

function pushbutton3\_Callback(hObject, eventdata, handles) %系统IFFT

handles = guihandles(gcf);

load y.mat;

y=Y;

load Fs.mat;

Fs=FS;

r=length(y);

m=log2(r);

m1=floor(m);%t1为小于等于t的最大整数

m2=ceil(m); %t2为大于等于t的最小整数

if m1~=m2 %t1不等于t2

y(r:2^m2)=0;

end

r=length(y);

y=fft2(y);

y1=ifft2(y);

t=(0:r-1)/Fs; %计算时间轴

sound(y1,Fs);

axes(handles.axes3);

plot(t,y1);xlabel('Time(s)');%在第一个窗口画波形

grid on;

function pushbutton4\_Callback(hObject, eventdata, handles) %自编FFT

handles = guihandles(gcf);

load y.mat;

y=Y;

load Fs.mat;

Fs=FS;

r=length(y);

m=log2(r);

m1=floor(m);%t1为小于等于t的最大整数

m2=ceil(m); %t2为大于等于t的最小整数

if m1~=m2 %t1不等于t2

y(r:2^m2)=0;

end

r=length(y);

t=(0:r-1)/Fs; %计算时间轴

y=transform\_fft(y);

y=fftshift(y);

y=abs(y);

sound(y,Fs);

axes(handles.axes4);

plot(t,y);xlabel('Time(s)');%在第一个窗口画波形

grid on;

function pushbutton5\_Callback(hObject, eventdata, handles) %自编IFFT

handles = guihandles(gcf);

load y.mat;

y=Y;

load Fs.mat;

Fs=FS;

r=length(y);

m=log2(r);

m1=floor(m);%t1为小于等于t的最大整数

m2=ceil(m); %t2为大于等于t的最小整数

if m1~=m2 %t1不等于t2

y(r:2^m2)=0;

end

r=length(y);

y=fft2(y);

y1=ifft2(y);

t=(0:r-1)/Fs; %计算时间轴

sound(y1,Fs);

axes(handles.axes5);

plot(t,y1);xlabel('Time(s)');%在第一个窗口画波形

grid on;

function array1=transform\_fft(array) %对N位一维数组进行FFT

array=double(array);

N=length(array);

M=log2(N);

for m=0:N/2-1;

WN(m+1)=exp(-1i\*2\*pi/N)^m;

end

J=0;

for I=0:N-1

if I<J

T=array(I+1);

array(I+1)=array(J+1);

array(J+1)=T;

end

K=N/2;

while J>=K

J=J-K;

K=K/2;

end

J=J+K;

end

for L=1:M

B=2^(L-1);

for R=0:B-1

P=2^(M-L)\*R;

for K= R:2^L:N-2

T=array(K+1)+array(K+B+1)\*WN(P+1);

array(K+B+1)=array(K+1)-array(K+B+1)\*WN(P+1);

array(K+1)=T;

end

end

end

array1=array;

return

function array=transform\_ifft(array) %对N位一维数组进行IFFT

N=length(array);

array=conj(array);

array1=transform\_fft(array);

array2=conj(array1);

array3=array2/N;

array=array3;

function pushbutton6\_Callback(hObject, eventdata, handles) %退出

clc;

close all;

close(gcf);