《视频图像处理》

[《视频图像处理》 1](#_Toc183600352)

[第一节:视频图像基础 3](#_Toc183600353)

[1.1图像 3](#_Toc183600354)

[1.1.1像素点 3](#_Toc183600355)

[1.1.2像素时钟 4](#_Toc183600356)

[1.2视频 5](#_Toc183600357)

[1.2.1帧率 5](#_Toc183600358)

[1.3视频图像处理 5](#_Toc183600359)

[1.3.1邻域 6](#_Toc183600360)

[1.3.2邻域选取 6](#_Toc183600361)

[第二节：图像滤波 7](#_Toc183600362)

[2.1均值滤波 8](#_Toc183600363)

[2.1.1均值滤波算法验证 8](#_Toc183600364)

[2.1.2验证结果 9](#_Toc183600365)

[2.1.3均值算法优化 10](#_Toc183600366)

[2.2中值滤波 11](#_Toc183600367)

[2.2.1冒泡排序算法 12](#_Toc183600368)

[2.2.2冒牌排序算法HDL实现 12](#_Toc183600369)

[2.2.3中值滤波验证 13](#_Toc183600370)

# 第一节:视频图像基础

生活中音像数据是一个庞大的数据，自然离不开对影像数据的 各种处理，比如GPU是视频图像处理专用的ASIC数字电路，它和DSP，cpu等数字电路在实际应用中扮演者不同的角色，视频图像处理的应用是非常广泛。

## 1.1图像

图像就是**影像数据**在计**算机或者电路中**的一种表示，在一个平面上能够展现出连续画面，在电路或者计算机中表示连续的画面是不现实的，通过离散的表示方法来展现连续的画面。

通过以上可以看出，任何影像数据在平面上展现，都是离散表示，被整个平面分割成较小的区域，只要区域足够的小，肉眼看到的是连续的，我们把这个较小的区域叫做像素点。

### 1.1.1像素点

任何图像在计算机和电路中都是以像素点为最小的表示单位，换句话说像素点是图像的最小单位，任何像素点作为最基本的单位，像素点展现出来的是色彩。

像素点在实际中，它的值不同表示出不同图像，下来我们探讨像素值由哪些分类

1. 二值图像

二值图像值的是每个像素点的值只有2个值，一般情况下二至图像的像素点值用1比特表示，比如点阵图。

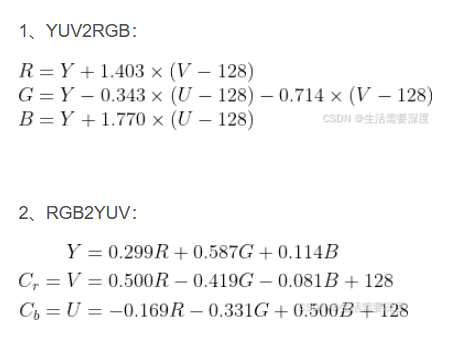
除了以上标准的二值图像以外，黑白图像也可以认为是二值图像

1. 灰度图

灰度图每个像素点用8比特表示，取值范围[0:255]，叫做灰度值，黑白图也可以认为是灰度图的一种特殊情况。

1. 彩色图像

彩色图像通常一个像素点需要3个值表示，这三个值表示颜色，构成了色彩空间，比较常见的色彩空间由RGB,YUV,HSV,CMY，其中RGB和YUV是2中最基本的色彩空间，这个空间转换如下：



**YUV色彩空间中Y分量构成的图像就是灰度图像，当我们需要把彩色图像转换为灰度图，我们只需要转换到YUV色彩空间获取Y分量即可得到灰度图**。

### 1.1.2像素时钟

在实际中图像在平面上连续的展示是通过离散的方式表示，在平面上所有像素点不是同时出现，而是分时出现，整个平面所有像素点显示需要的时间，一般用它的倒数，即频率表示，叫做刷新频率，时间叫做刷新时间，每个像素点显示需要的时间的倒数叫做像素时钟。

## 1.2视频

视频本质上是多福图像在时间轴上动态显示的过程，每幅图像之间的时间间隔足够小，肉眼是分别不出来，展现出来的是连续的动态过程。

视频本质上还有由图像构成，图像的基本单位是像素，视频也是以像素作为基本单位。

### 1.2.1帧率

视频中一副图像我们通常叫做一帧图像，每一帧图像动态显示的时间间隔我们用帧率表示，应用有的场景用fps表示，有的用hz表示。

帧率越高对硬件需求越高，图像的刷新率也越高。

## 1.3视频图像处理

视频图像处理是一个典型的应用，直观的说就是对视频或者图像数据进行各种操作，都叫做视频图像处理，归根结底是对像素点的操作，可以对一副图像的像素点操作，也可以对多福图像的像素操作。

对图像的操作，即视频图像处理我们归纳起来有以下2种：

变换：针对一副或者多福图像的像素点经过科学运算，实现相应变换。

控制：针对一副或者多福图像的的相关控制，比如DDRx读写控制

### 1.3.1邻域

在视频图像处理中，有个关键的概念，我们称之为领域，在诸多应用中都离不开邻域，所谓邻域就是视频图像处理中参与运算的所有像素点的集合。

当我们在对视频图像数据进行运算时候，我们需要考虑哪些像素点参与运算，有的运算需要单个像素点参与运算，有的时候需要多个像素点参与运算，这里我们多个像素点就构成了邻域。

邻域

分辨率=K1\*K2

### 1.3.2邻域选取

在应用中关于邻域的选取也有要求，由于运算基本都是基于中心像素点运算，因此一般选取完全对称邻域。

|  |  |
| --- | --- |
| P00 | P01 |
| P10 | P11 |

2\*2邻域（非对称邻域，共4个像素点）

|  |  |  |
| --- | --- | --- |
| P00 | P01 | P02 |
| P10 | P11 | P12 |

2\*3邻域（左右对称邻域，共6个像素点）

|  |  |
| --- | --- |
| P01 | P01 |
| P10 | P11 |
| P20 | P21 |

3\*2邻域（上下对称邻域，共6个像素点）

|  |  |  |
| --- | --- | --- |
| P00 | P01 | P02 |
| P10 | P11 | P12 |
| P20 | P21 | P22 |

3\*3邻域（完全对称邻域，共9个像素点）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P00 | P01 | P02 | P03 | P04 |
| P10 | P11 | P12 | P13 | P14 |
| P20 | P21 | P22 | P23 | P24 |
| P30 | P31 | P32 | P33 | P34 |
| P40 | P41 | P42 | P43 | P44 |

5\*5邻域（完全对称邻域，共9个像素点）

在实际中邻域选取以完全对称邻域作为邻域，对称中心对应的像素点叫做中心像素点，也叫做目标像素点，我们往往以最小完成对称邻域3\*3作为我们的邻域。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

注意：边缘像素点没有邻域

# 第二节：图像滤波

图像滤波是视频图像处理中常见的一种处理变化算法，主要是对图像左平滑处理。

## 2.1均值滤波

用邻域的平均值代替源目标像素点的值，构成新的图像，这个图像就是滤波后的图像。

Oval = (p00+p01+p02+p10+p11+p12+p20+p21+p22)/9

### 2.1.1均值滤波算法验证

img\_filter

P11[7:0]

P10[7:0]

p02[7:0]

p01[7:0]

p00[7:0]

oval[7:0]

P20[7:0]

P12[7:0]

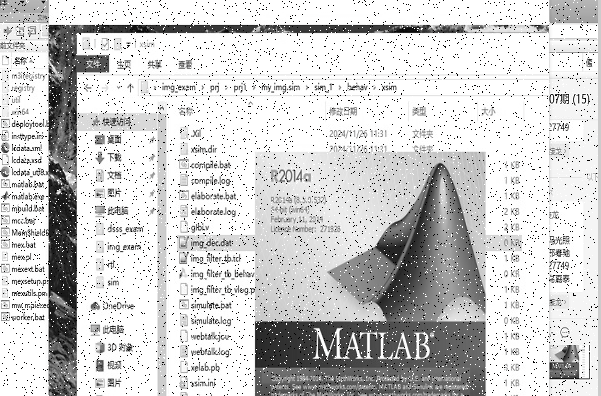
P22[7:0]

P21[7:0]

端口说明

|  |  |
| --- | --- |
| 信号名 | 描述 |
| 邻域像素点 | |
| p00[7:0] | 邻域1\*1像素点 |
| p01[7:0] | 邻域1\*2像素点 |
| p02[7:0] | 邻域1\*3像素点 |
| p10[7:0] | 领域2\*1像素点 |
| p11[7:0] | 领域2\*2像素点，中心像素点 |
| p12[7:0] | 领域2\*3像素点 |
| P20[7:0] | 领域3\*1像素点 |
| P21[7:0] | 领域3\*2像素点 |
| P22[7:0] | 领域3\*3像素点 |
| 输出平均值 | |
| oval[7:0] | 邻域平均值 |

### 2.1.2验证结果



滤波后

滤波前



### 2.1.3均值算法优化

在硬件设计中设计按照抽象层次分为系统级，算法级，RTL级，门极开关级，当我们在进行算法实现的时候，能够把算法级的算法转化为RTL级则转换，在设计中尽量避免使用乘法器，触发器。

在硬件设计中求均值可通过加权平均法实现求平均值，加权平均法是可以通过加减移位等操作实现。

假如一学年的考试分为3次，第一季度考试，其中考试，期末考试，小明第一季度的数学成绩是85分，其中开始数学成绩是60分，期末考试数学成绩是70分，求小明1学年的数学平均成绩。

数学平均成绩数学平均 = （85分+60分+70分）/3 = 71.67分

数学平均可以反应小明这一学年的数学学习的基本状况，但是不是绝对很全面，为了全面反应小明这学年的数学学习基本情况，可采用如下的方式计算：

数学平均成绩 = （1/4 \* 85分） + （1/4\*60分） + (2/4)70分

= 21.25+15+35 = 71.25

**数学平均成绩加权平均 = （（1 \* 85分） + （1\*60分） + 2\*70分 ）/4**

权值：根据实际应用情况合理分配

权值：根据实际应用情况合理分配

权值：根据实际应用情况合理分配

**数学平均成绩加权平均 = （（1 \* 85分） + （1\*60分） + 1\*70分 ）/3**

**数学平均成绩加权平均 = （（1 \* 85分） + （2\*60分） + 3\*70分 ）/6**

通过以上的加权平均法分析，我们可以将均值滤波算法按照加权平均的方式实现，合理的分配权值便可以更好的通过硬件实现。

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 2 | 4 | 2 |
| 1 | 2 | 1 |

|  |  |  |
| --- | --- | --- |
| P00 | P01 | P02 |
| P10 | P11 | P12 |
| P20 | P21 | P22 |

Oval = (p00\*1 + p01\*2 +p02\*1 +

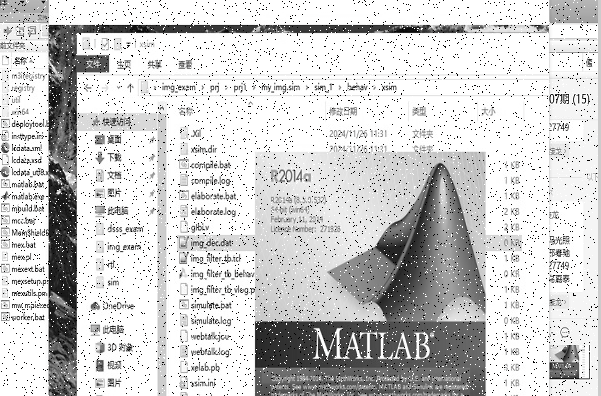
P10\*2 + p11\*4 +p12\*2 +

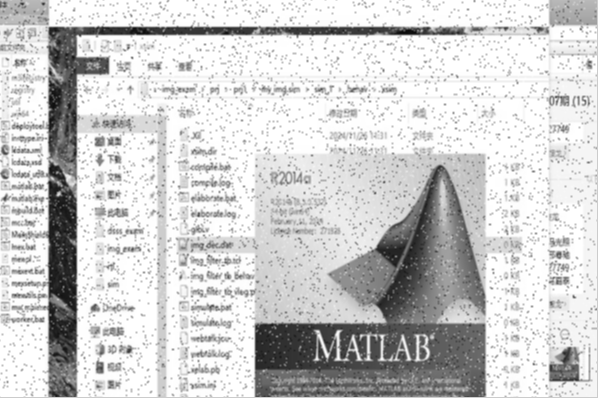
P20\*1 + p21\*2 +p22\*1 )/16;

= **(p00\*<<0 + p01<<1 +p02<<0 +**

**P10<<1 + p11<<2 +p12<<1 +**

**P20<<0 + p21<<1 +p22<<0 )>>4;**





滤波后

滤波前

## 2.2中值滤波

所谓中值滤波就是用邻域的中间值代替原来中心目标像素点的值，构成的图像就是滤波后的图像。

在硬件实现中值滤波的时候，核心就是邻域的数据的排序问题，排序的算法有很多，选择一种适合硬件实现且占用硬件资源不多的算法即可，常见的排序算法有;



以上排序算法是计算机数据结构原理中的基本排序算法，各种排序算法所需要的时间（计算器软件运行），稳定性各不相同，在硬件设计中常使用冒泡排序算法，也经典排序算法，适合硬件实现。

### 2.2.1冒泡排序算法

MLB

MSB

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| N-1 | N-2 | N-3 | N-4 | … | 1 | 0 |

**第N比较交换**

D0

将1个数经过1次比较交换

**第N-1比较交换**

D1

将3个数经过2次比较交换

**第4比较交换**

Dn-4

将N-3个数经过N-4次比较交换

**第3比较交换**

Dn-3

将N-2个数经过N-3次比较交换

**第2比较交换**

Dn-2

将N-1个数经过N-2次比较交换

**第1比较交换**

Dn-1

将N个数经过N-1次比较交换

遍历 ：如上图所示，我们把每一次的比较交换过程叫做遍历

交换比较：2个数比较大小，同时交换

### 2.2.2冒牌排序算法HDL实现

比较交换电路模块

p01[7:0]

p00[7:0]

min[7:0]

max[7:0]

D1[7:0]

D0[7:0]

switch

遍历

第1次遍历

max43[7:0]

max42[7:0]

max42[7:0]

max41[7:0]

max41[7:0]

max40[7:0]

max40[7:0]

max33[7:0]

max332[7:0]

max33[7:0]

max32[7:0]

max32[7:0]

max31[7:0]

max31[7:0]

max30[7:0]

max30[7:0]

max25[7:0]

max24[7:0]

max24[7:0]

max23[7:0]

max23[7:0]

max22[7:0]

max22[7:0]

max21[7:0]

max21[7:0]

max20[7:0]

max20[7:0]

max16[7:0]

max15[7:0]

max15[7:0]

max14[7:0]

max14[7:0]

max13[7:0]

max13[7:0]

max12[7:0]

max12[7:0]

max11[7:0]

max11[7:0]

max10[7:0]

max06[7:0]

max07[7:0]

max05[7:0]

min06[7:0]

max06[7:0]

P20 [7:0]

max04[7:0]

min05[7:0]

max05[7:0]

P12 [7:0]

max03[7:0]

min04[7:0]

max04[7:0]

P11 [7:0]

max02[7:0]

min03[7:0]

max03[7:0]

min02[7:0]

max02[7:0]

P10 [7:0]

max01[7:0]

min01[7:0]

max01[7:0]

P02[7:0]

max00[7:0]

max00[7:0]

min00[7:0]

Switch07

Switch06

Switch05

Switch04

Switch03

Switch02

min00[7:0]

Switch01

Switch00

第5次遍历

第4次遍历

第3次遍历

第2次遍历

Switch12

Switch11

Switch10

min43[7:0]

Switch43

min42[7:0]

Switch42

min41[7:0]

Switch41

min40[7:0]

min30[7:0]

Switch40

min34[7:0]

Switch34

min33[7:0]

Switch33

min32[7:0]

Switch32

min31[7:0]

Switch31

min30[7:0]

min20[7:0]

Switch30

min25[7:0]

Switch25

min24[7:0]

min15[7:0]

Switch24

Switch23

min23[7:0]

Switch23

min22[7:0]

Switch22

min21[7:0]

Switch21

min20[7:0]

min10[7:0]

Switch20

min16[7:0]

min07[7:0]

Switch16

Switch15

min14[7:0]

Switch14

min13[7:0]

Switch13

min12[7:0]

min11[7:0]

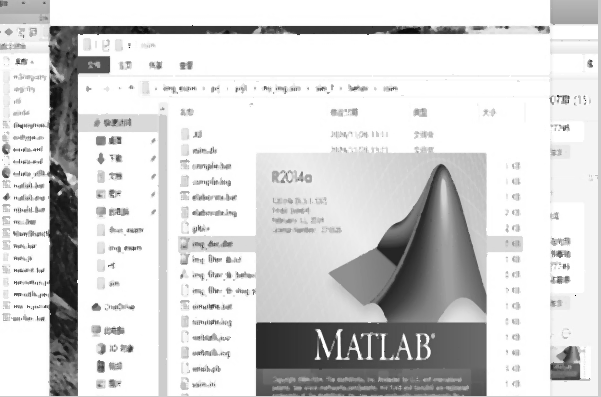
max10[7:0]

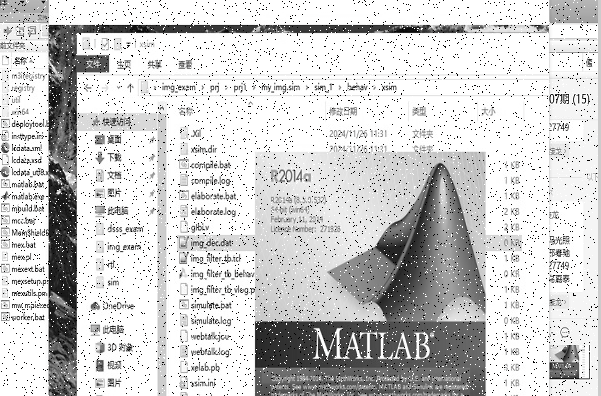
min10[7:0]

P22 [7:0]

P21 [7:0]

### 2.2.3中值滤波验证





滤波后

滤波前

我色粉