به نام خدا

پردازش تصویر

۱. پیشگفتار

۲. بهبود دامنه رنگی و فیلترها

تهیه و تنظیم: صالح جمالی (Salehjg)

forum.iranled.com

مقدمه:

متاسفانه با وجود کاربرد بسیار گسترده و مهم پردازش تصویر در امور مختلف و تاسیس رشته‌های جداگانه ی پردازش تصویر در دانشگاه‌های خارج از کشور، بحث پردازش تصویر در ایران هنوز چنان که باید، مطرح نشده است و جای خالی آن هر روز بیشتر حس می شود.

مغز انسان همواره بر روی تصاویر دریافتی از چشم، پردازش‌های گوناگونی انجام می دهد، پردازشی که باعث می شود آگاهانه در دنیای پیرامون خود زندگی کنیم، بخوانیم، بنویسیم، فرار کنیم، بترسیم، ترسیم کنیم، با دو چشم ببینیم، قدرت انتخاب هر چشم را داشته باشیم و تمرکز کنیم و ... ، در واقع مغز انسان ها یا سایر موجوداتی که چشم دارند، ناخودآگاه پردازش تصویر انجام می دهند. با اختراع سنسور دیجیتالی تصویر، بحث پردازش تصویر بیشتر مطرح شد.پردازشی که به سرعت پردازنده وابستگی وافری دارد.البته این مسئله در بعضی از موجودات زنده نیز صادق است.آیا تابه حال سعی کرده اید تا با نگاه کردن به قسمتی که بالای آن نوشته ای وجود دارد، آنرا بخوانید.درست است، نمی توانید، چون مغز شما تنها قسمت مرکزی تصویر را برای پردازش انتخاب کرده است تا سرعت بیشتری در پردازش تصاویر ورودی داشته باشد. پس الگوریتم پردازش، عامل مهمی است، عاملی که همیشه باید آن را در نظر بگیرید.

امیدوارم با برداشتن قدمی کوتاه بوسیله ی نوشتن این مقاله، شما عزیزان را به نحوه‌ی پردازش تصویر که واقعا علمی گسترده است و در چند صفحه نمی گنجد، علاقه مند کرده باشم.

اگر نقص و اشکالی در این مقاله دیدید، لطفا آن را از طریق ایمیل یا انجمن ایرانویج مطرح کنید.

در این مقاله سعی شده تا بیشتر درباره‌ی الگوریتم‌های پردازش تصویر بحث شود و لازم به ذکر است که از نرم‌افزارهای آماده برای پردازش تصویر(مانند MATLAB) استفاده نشده است.

*«این مقاله متعلق به انجمن **ایران ویج** است و ادامه ی آن در انجمن **ایران ویج** قابل دسترس خواهد بود.»*

صالح جمالی گلزار– تابستان ۸۹

Email: saleh_jamali@ymail.com

forum.iranled.com

بخش اول

پیشگفتار

forum.iranled.com

۱ پیش نیازها

برای جلوگیری از هرگونه ابهام آشنایی مقدماتی با موارد زیر لازم است:

- تصاویر و ساختار آنها
- زبان برنامه نویسی و آرایه ها

در زیر توضیح مختصری بر هر دو مورد بالا داده شده است.

۱.۱ پیش درآمدی بر تصویر و ساختار آن:

تصاویری که ما قصد اعمال پردازش بر رویشان را خواهیم کرد ، تصاویر رنگی ۲۴ بیت هستند.

در سیستم رنگی RGB ، تمام رنگ ها از ترکیب ۳ رنگ قرمز ، سبز ، آبی تشکیل شده است پس می توان نتیجه گرفت که در تصاویر رنگی ۲۴ بیت ، ۸ بیت برای هر کانال Red ,Green ,Blue فضا تعیین شده است ، پس دامنه ی تغییرات هر کانال رنگ در سیستم RGB بین ۰ و ۲۵۵ خواهد بود($۲^۸$ = ۲۵۶). پس اگر بخواهیم آرایه ای برای گنجاندن تصویری رنگی ۲۴ بیت با ابعاد ۳۰۰ x ۴۰۰ پیکسل ، ایجاد نمائیم ، بطوریکه بخواهیم سیستم رنگی از نوع RGB باشد ، مجبوربه ساختن آرایه ای از جنس Byte هستیم.

در مورد ابعاد آرایه شما می توانید به طور دلخواه عمل کنید یعنی بسته به این که برای هر کانال رنگ (قرمز – سبز– آبی) می خواهید آرایه ای جدا یا برای هر ۳ کانال رنگ ، تنها یک آرایه ایجاد نمایید ابعاد تغییر خواهد کرد(با ضریب ۳). برای مثال اگر بخواهیم تصویر فرضی ۴۰۰ در ۳۰۰ پیکسل را با سیستم RGB درون **یک** آرایه قرار بدهیم، با توجه به این که تصویر ما ۲۴ بیت است پس برای هر کانال رنگ ۱ بایت فضا نیاز است که با توجه به این مطلب، آرایه دارای ابعاد ۳×۴۰۰×۳۰۰ باشد.(آرایه را تک بعدی تعریف کرده ایم)

forum.iranled.com

نمونه ی آرایه ی تعریف شده در C#.Net :

```
Byte [] Array = new byte[400*300*3];
```

آرایه ی Array، آرایه ای تک بعدی با 400x 300x 3 خانه است.(تعداد کانال‌های رنگ×عرض×طول) و چون تصویر مدنظر ۲۴-بیت است، و تعداد کانال‌های رنگ، ۳ می باشد(پس برای هرکانال رنگ ۸-بیت فضا نیاز است)، جنس آرایه Byte انتخاب شده است.

تصویر ۱.۱.۱

۱.۲ خواندن رنگ هر پیکسل و گنجاندن آنها در آرایه:

برای راحتی کار ، فایل کلاس principle ضمیمه شده است.(برای زبان (C#.Net (Frame Work 3

در این کلاس توابع آماده جهت خواندن و نوشتن بر روی عکس و چند تابع دیگر که در ادامه به توضیح بیشتر در مورد هر یک خواهیم پرداخت.

توابعی که بیشتر کارمان با آن ها گره خورده است در زیر توضیح داده شده است.

```
read_image(Image img, byte[] array)
```

این تابع تصویر img را می خواند و مقادیر آن را در آرایه ی array که از جنس بایت است ، قرار می دهد.

لازم به ذکر است که ردیف قرار گیری رنگ های قرمز ، سبز و آبی در آرایه به صورت زیر است:

```
public Image write_image(Image img, byte[] array)
```

این تابع مقادیر موجود بر روی آرایه ی array که از جنس بایت است(تصویر ۲۴ بیت) را در تصویری با ابعاد مشابهِ img ، باز میگرداند.

```
public int get_image_array_count(Image img)
```

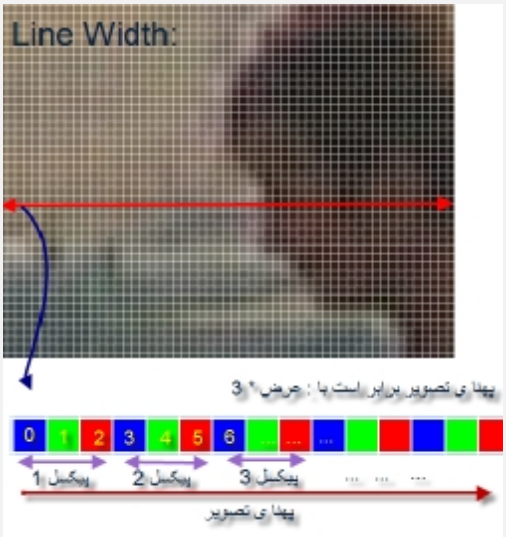
این تابع ابعاد آرایه را با توجه به تصویر img بر می گرداند.

مثال: مقدار بازگشتی این تابع برای تصویری ۲۴ بیت RGB ، ۳۶۰۰۰۰ است : 400x300x3=360000

```
public int get_line_width(Image img)
```

این تابع مقدار پهنا ی عکس را با احتساب ۳ کانال قرمز و سبز و آبی باز می گرداند.

برای درک بهتر موضوع به تصویر زیر نگاه کنید:

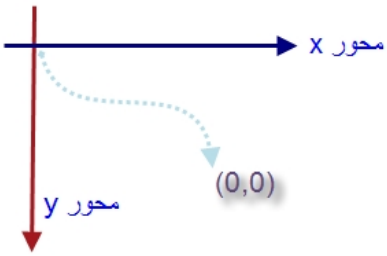


به بیانی دیگر پهنای تصویر(Line Width) عرض تصویری است که هر پیکسل عکس اولیه در آن به ۳ پیکسل مجزای رنگ های قرمز و سبز و آبی تبدیل شده است.

نکته :

index پیکسلی در نقطه ی (x= 10 , y = 40) به صورت زیر محاسبه می شود: (در صورتی که هر سه کانال رنگ در یک آرایه قرار گرفته باشد)

```
Index = y * line width + x *3
```



تصویر ۱.۲.۱

بخش دوم

بهبود دامنه رنگی

توجه: در تمام مقاله، منظور از تصویر، تصویر ۲۴ بیت می باشد. یعنی قرار است فقط بر روی تصاویر رنگی ۲۴ بیتی کار کنیم.

پیش درآمد:

در بخش اول توضیحات اجمالی درباره تصاویر و نحوه ی بارگذاری آنها در آرایه و ... آورده شده بودند. اما در این قسمت شما با color adjustment های مختلف و نحوه ی کار آنها آشنا خواهید شد.

فرض کنیم اگر بخواهید برنامه ای برای پردازش تصویر طراحی کنید تا مانند برنامه ی Photoshop اما در مقیاس کوچک‌تر بر روی تصاویر افکت ها و تغییرات گوناگون اعمال کند به چند مورد اساسی نیاز خواهید داشت که در اکثر برنامه‌های پردازش تصویر یافت می‌شوند. ازجمله:

Contrast, Brightness, Gamma Correction, RGB Channels, Blur, Edge Detection

در این بخش سعی شده تا خوانندگان با Color Adjustment های رایج و نحوه ی کار آنها آشنا شوند.

forum.iranled.com

۲.۱ روشنایی(Brightness):

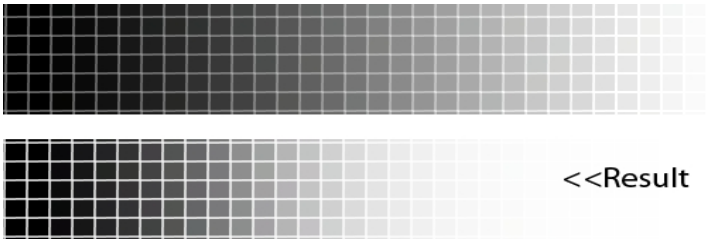
Brightness وظیفه ی اعمال تغییر بر روشنایی تصویر را دارد، همان طور که می دانیم رنگ سفید در سیستم RGB معادل قرمز=۲۵۵، سبز=۲۵۵، آبی=۲۵۵ و رنگ سیاه معادل قرمز=۰، سبز=۰، آبی=۰ می باشد.

پس هرچه از مقدار صفر به ۲۵۵ نزدیک تر میشویم رنگ روشن تری خواهیم داشت. پس کافی است برای اعمال Brightness بر همه ی کانال‌های قرمز و سبز و آبی مقدار دلخواهی را اضافه کنیم البته چون مقدار دلخواه در هر حالت به تمام کانال‌های رنگ اضافه خواهد شد ، باید مقدار حاصل را کنترل کرد تا در دامنه ی صفر تا ۲۵۵ باشد و چون در تصاویر ۲۴ بیت حداکثر هر کانال رنگ ۲۵۵ و حداقلش ۰ است پس حداکثر میزان دلخواه باید ۲۵۵(چون اگر رنگ کانالی صفر باشد آنرا ۲۵۵ کند) و حداقلش ۲۵۵- است.(تا رنگ کانالی که ۲۵۵ است را صفر کند.)

تا اینجا با نحوه ی کار brightness آشنا شدید. حال به کد فرضی زیر توجه کنید تا قضیه برایتان ملموس تر شود.

```
for(int x=0;x<width;x++)
{
        for(int y=0;y<height;y++)
        {
                red = red + value;
                if(red > 255){red = 255;}
                if(red < 0){red = 0;}
                //================
                green = green + value;
                if(green > 255){green = 255;}
                if(green < 0){green = 0;}
                //================
                blue = blue + value;
                if(blue > 255){blue = 255;}
                if(blue < 0){blue = 0;}
        }
```

um.iranled.com



تصویر ۲.۱.۱ - نتیجه ی اعمال Brightness

forum.iranled.com

۲.۲ بهبود کانال‌های قرمز، سبز و آبی(RGB Channels Adjustment):

همان طور که از نامش پیداست، وظیفه ی جمع کردن مقادیر ثابت متناظر با هر یک از کانال‌های قرمز، سبز و آبی را دارد. همانطور که میدانید دامنه ی تغییر هر کانال در سیستم RGB ۲۴- بیت، صفر تا ۲۵۵ میباشد.پس حداکثر مقدار ثابت ۲۵۵(برای اینکه صفر را به ۲۵۵ برساند) و حداقل آن ۲۵۵- خواهد بود(برای اینکه رنگ ۲۵۵ را به صفر برساند).

به وسیله ی این تنظیم(RGB Channels) میتوانیم تاثیر میزان هر کانال R,G,B را در تصویر، کنترل نمائیم.

تصویر ۲.۲.۱ – تصویر اصلی – تصاویری که با RGB Channels تغییر یافته اند.

به کد فرضی زیر توجه کنید: (مقادیر ثابت برای هر کانال به صورت زیر در نظر گرفته شده اند)

Red_const, Green_const, Blue_const

```
Red = Red + Red_const;

Green = Green + Green_const;

Blue = Blue + Blue_const;
```

اگر مقادیر ثابت برای همه ی کانال‌های R,G,B با هم برابر باشند، RGB Channels دقیقاً مانند Brightness عمل خواهد نمود.(پیدا کردن دلیلش بر عهده ی خودتان!)

۲.۳ کنتراست(Contrast):

کنتراست چیست؟ چه زمانی می گوییم که تباین (contrast) رنگ پیکسل‌های یک تصویر زیاد یا کم است؟ از جمله‌ی آخر مفهوم کمی آشکار تر شده است. بله زمانی که تفاوت زیادتر باشد اما تفاوت چه چیزی؟ به دو تصویر زیر توجه کنید:

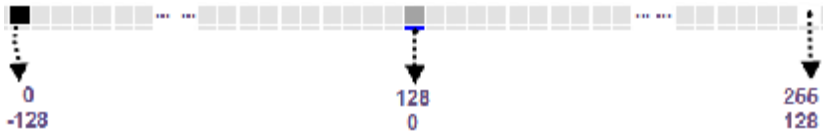
تصویر ۲.۳.۱ تصویر ۲.۳.۲

تصویر ۲.۳.۲ دارای کنتراست بیشتری نسبت تصویر ۲.۳.۱ است.

اگر به تصاویر بالا دقت کنید تصویر(۲.۳.۲) که دارای کنتراست زیادتری است دارای خاصیتی است که تصویر (۲.۳.۱) ندارد. واضح است که در تصویر ۲.۳.۲، مناطقی که در تصویر ۲.۳.۱ تیره بودند، تیره تر شده و مناطقی که روشن بوده اند، روشن‌تر شده‌اند. پس برای اعمال contrast نیازمند اعمال یک مرز داریم. مرزی که رنگ روشن را از تیره جدا کند. در کد فرضی که در صفحات بعد آورده شده است، مقدار مرز ۱۲۸ می‌باشد. ۱۲۸ میانه‌ی ۰-۲۵۵ است. شما می توانید مقدارش را تغییر دهید تا نتایج را مشاهده کنید.

forum.ir

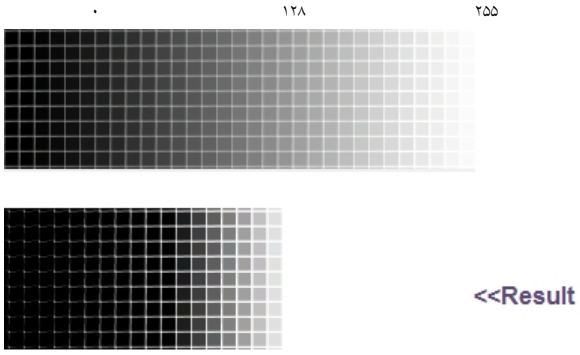
بیایید در ذهنمان مسائل را تجزیه کنیم. مقدار مرز روشنی و تیرگی را ۱۲۸ قرار داده ایم. حال پیکسلی را در نظر بگیرید که رنگش R=200,G=200,B=200 است. چگونه می‌توانیم تئوری بالا را رویش پیاده کنیم؟ (کمی فکر کنید بعدا ادامه را بخوانید.)



تصویر ۲.۳.۳

رنگ هایی را در نظر بگیرید که در سمت چپ ۱۲۸ قرار میگیرد(سمت چپ تصویر۲.۳.۳). برای اینکه بتوانیم این رنگ ها را تیره تر کنیم باید به نحوی به سمت صفر نزدیک ترشان کنیم. اگر از تمام رنگ ها مقدار مرز(۱۲۸) را کم کنیم دامنه ی مقادیر حاصل بین ۱۲۸ و ۱۲۸- خواهد بود. مقادیر منفی رنگ‌های تیره و مقادیر مثبت رنگ‌های روشن تر هستند. حال یک مقدار ثابت که مثبت است را در رنگ هایی ضرب کنیم که ۱۲۸ از آنها کم شده بود. با این عمل ما همان تئوری‌ای که در بالا به توضیح آن پرداختیم را پیاده کرده‌ایم. یعنی رنگ‌های روشن‌تر(مقادیر مثبت) روشن‌تر شده-اند (به طرف مثبت بی نهایت سوق یافته‌اند) و رنگ‌های تیره (مقادیر منفی) تیره تر شده اند.(به طرف منفی بی نهایت سوق یافته اند.) حال باید ۱۲۸را به تمام مقادیر حاصل اضافه کنیم. با این کار ۱۲۸ ای که در اول کم کرده بودیم را حذف می کنیم. تنها عملی که باید انجام شود محدود کردن مقادیر نسبت به عمق رنگ است ، چون تصاویر ما ۲۴ بیت است پس مقادیر باید بین ۲۵۵ و ۰ محدود شوند.نتیجه در تصویر ۲.۳.۴ آورده شده است.

forum.iranled.com



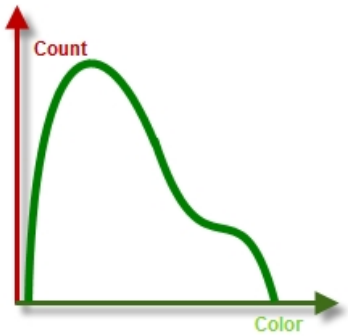
تصویر۲.۳.۴

این روش یکی از روش‌های گوناگون اعمال contrast است. روش معمول دیگر، روشی است که بر پایه ی histogram بنا شده است.

forum.iranled.com

۲.۴ هیستوگرام(Histogram):

نمودارها در نمایش اطلاعات مربوط به رنگ‌های استفاده شده و پراکندگی آن ها کاربرد دارد. محور افقی مقادیر رنگ ها و محور عمودی تعداد استفاده از آنها می باشد.



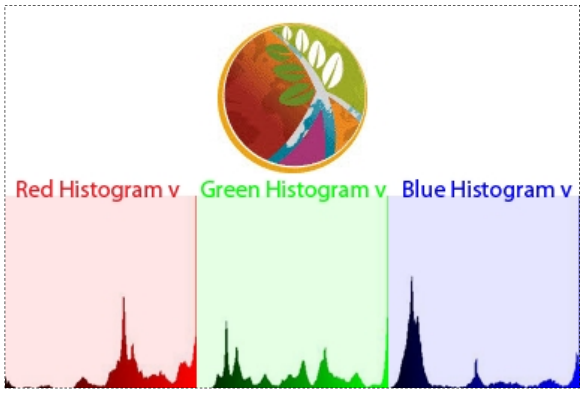
صفر ۲۵۵

تصویر ۲.۴.۱

در اغلب نرم افزار هایی که به نوعی با تصویر سر و کار دارند، می توان نمودارهای مختلفی را مشاهده کرد.

مانند نمودار کانال‌های رنگی RGB در تصاویر رنگی و نمودار مقیاس خاکستری (gray) در تصاویر رنگی و مقیاس خاکستری(Grayscale).

نمونه ای از نمودار کانال‌های RGB را در تصویر زیر مشاهده می کنید:



تصویر ۲.۴.۲

forum.iranled.com

روش کار همچنان که مشاهده کردید، فوق‌العاده ساده است. برای شروع نیاز به ۳ آرایه که دارای ۲۵۶ خانه باشند، نیاز است. جون مقادیر منفی برایمان اهمیتی ندارد و از طرفی گنجایش مهم است، ترجیحا جنس آرایه ها را از unsigned int قرار میدهیم. سپس شروع به بررسی تمام پیکسل‌های تصویر میکنیم. بر فرض پیکسلی دارای رنگ قرمز=۱۰۰، سبز=۱۵۰و آبی=۲۰۰ باشد. چون مقدار کانال قرمز ۱۰۰ است، به ۱۰۰ اُمین index آرایه‌ای که برای کانال قرمز انتخاب کرده‌ایم، یک واحد اضافه می کنیم. در ادامه چون مقدار کانال سبز ۱۵۰ است به ۱۵۰ امین index آرایه‌ای که برای کانال سبز انتخاب کرده ایم، یک واحد اضافه می کنیم. برای کانال آبی هم همین روال را ادامه می‌دهیم.

به کد فرضی زیر توجه کنید:

آرایه‌ها به صورت زیر تعریف شده‌اند:

برای کانال قرمز `Int []Red_Array = new int[256];`

برای کانال سبز `Int []Green_Array = new int[256];`

برای کانال آبی `Int []Blue_Array = new int[256];`

```
For(int x=0;x<width;x++)
{
        For(int y=0;y<height;y++)
        {
                Red_array[Red]++;
                Green_array[Green]++;
                Blue_array[Blue]++;
        }
}
```

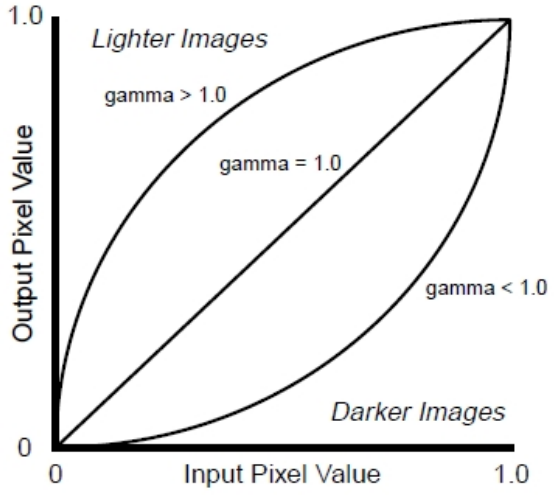
و در انتها، آرایه‌ها را با توجه به سلیقه شخصی به تصویر می‌کشیم.

m.iranled.com

۲.۵ اصلاح گاما(Gamma correction):

شاید تا به حال برایتان پیش آمده باشد که از تصویری پرینت بگیرید، اما با تصویری که در مانیتور با شرایط استاندارد دیده بودید، در روشنایی متفاوت باشد. دلیل این تفاوت، اختلاف در گاما است.(بین پرینتر و مانیتور)

برای حل این اختلاف، از اصلاح گاما استفاده میشود. بطوری که مقداری بین ۰ تا ۱۰ به عنوان مقدار ثابت به تابع اصلاح گاما داده می‌شود و تابع با توجه به مقدار ثابت، رنگ ورودی را که مقداری در دامنه ی [0,1] است، تغییر می-دهد. به تصویر زیر توجه کنید:



تصویر ۲.۵.۱

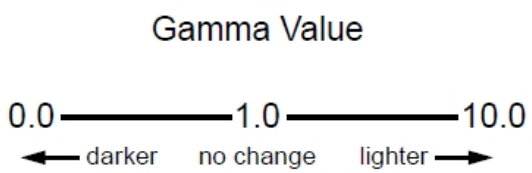
همان طور که در منحنی بالا مشاهده می کنید، اگر مقدار ثابت :

- کوچکتر از یک باشد، تصویر تیره تر میشود.
- برابر یک باشد، تصویر ثابت باقی خواهد ماند.
- بزرگتر از یک باشد، تصویر روشن تر می شود.

تا اینجا با عملکرد کلی اصلاح گاما، آشنا شدید. حال می خواهیم در مورد خود تابع اصلاح گاما بحث کنیم.

$$newval(x) = x^{\left(\frac{1}{gammaval}\right)}$$

تعریف بالا، تابع اصلاح گاما را نشان میدهد که مقدار ثابت برابر gammaVal می باشد. رنگ ورودی نیز همـان x است و رنگ خروجی newVal(x).



تصویر ۲.۵.۲

نمونه:

تصویر ۲.۵.۳ – تصویر اصلی – Gamma Correction

در تصویر بالا، ۲ نمونه از تصویر اصلی (سمت راست) را مشاهده می کنید که به ترتیب، دارای مقدار ثابت ۲.۵(تصویر میانی) و ۰.۵ (تصویر سمت چپ) برای اصلاح گاما، می باشند.

۲.۶ اصلاح گاما برای کانال‌های قرمز، سبز، آبی(RGB gamma correction):

اصلاح گاما برای کانال‌های قرمز، سبز، آبی تفاوت چندانی با اصلاح گاما ندارد. تنها تفاوت میان این دو، قابلیت اعمال مقادیر ثابت مختص هر کانال رنگ در سیستم RGB است.

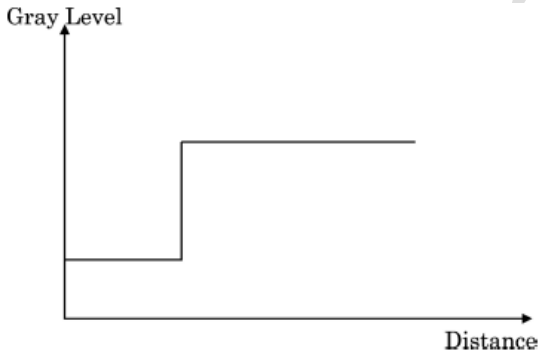
نمونه ای از RGB Gamma Correction:

تصویر ۲.۶.۱ – تصویر اصلی – RGB Gamma Correction

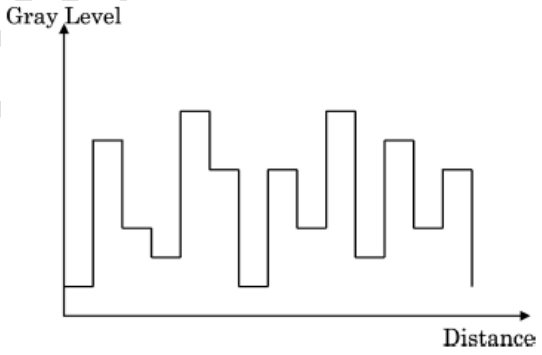
forum.iranled.com

۲.۷.۱ Spatial Frequency Filtering:

تمام عکس ها و تصاویر دارای فرکانس فاصله ای هستند. ما با بعضی از فرکانس ها آشنایی داریم، مانند برق ۲۲۰ ولت **۵۰ هرتز**. در برق، فرکانس با زمان اما در تصاویر فرکانس با فاصله تعریف میشود. فرکانس فاصله‌ای (Spatial Frequency) را میتوان به تغییرات رنگ مقیاس خاکستری (Grayscale) نسبت داد. تصویر(۲.۷.۱.۱)، نمایی از تغییرات تصویر را نشان میدهد که دارای فرکانس فاصله ای کمی است و تصویر(۲.۷.۱.۲) نمایی از تغییرات را در فاصله ای که یکسان با تصویر قبلی است که دارای فرکانس فاصله ای زیادی است (high spatial frequency).



تصویر ۲.۷.۱.۱ – نمای کناری از تصویر با فرکانس فاصله ای کم(low spatial frequency)



تصویر ۲.۷.۱.۲– نمای کناری از تصویر با فرکانس فاصله ای زیاد(high spatial frequency)

۲.۷.۲ کاربردهای فرکانس فاصله ای در پردازش تصویر:

با استفاده از فرکانس فاصله ای در تصاویر میتوان پردازش‌های گوناگونی اعم از حذف نویز ، محو کردن تصویر(smoothing) ، edge detection و... را انجام داد. در ادامه بیشتر با کاربردها و نحوه ی عمل آنها آشنا خواهید شد.

۲.۷.۳ فیلتر ها:

کاربرد فیلترها بسیار گسترده است. به عنوان مثال به فیلترهای بالا گذرو پائین گذر اشاره می کنم. فیلتر High-Pass همان طوری که از نامش پیداست، فرکانس‌های بالا را از خود عبور می دهد و مانع عبور فرکانس‌های پائین میشود در حالی که فیلتر Low-Pass فرکانس‌های پائین را از خود عبور می دهد و مانع عبور فرکانس‌های بالا می‌شود. همان طور که در بالا گفته شده فیلترها کاربرد گسترده ای دارند و این گستردگی بوسیله ی فرکانس فاصله ای، در تصاویر نیز داخل شده است.

۲.۷.۴ فرکانس در مقابل Spatial Filtering:

صدایی با فرکانس ۱۰۰۰ هرتز را در نظر بگیرید(1KHz).در قلرو فرکانسی(Frequency Domain) این یک مقدار برابر با ۱۰۰۰ است.حال در این قلمرو اگر بخواهید فرکانس این موج را به ۹۰۰ هرتز کاهش دهید ساده ترین راه ضرب کردن در یک فیلتر پائین گذر که تا ۹۰۰ هرتز را از خود عبور میدهد است.ضرب کردن در یک فیلتر پائین گذر فکر خوبی است اما یک مشکل وجود دارد. مردم صدا ها را در قلمرو زمان می شنوند (time domain)، اگرچه قبل از ضرب کردن ،سیگنال را به قلرو فرکانسی خواهیم برد. یکی از راه‌های تبدیل، استفاده از Fourier Transform میباشد. اما استفاده از Fourier Transform نیازمند انجام محاسبات سنگین است که در اکثر موارد ارزش تلاش را ندارد.

forum.iranled.com

ضرب کردن در قلمرو فرکانسی مانند پیچیدگی در قلمرو زمان و فاصله ای(یا فضایی یا spatial) است. استفاده از یک ماسک پیچیدگی کوچک(small convolution mask) مانند ۳×۳ بسیار آسان تر و سریع تر از پیاده کردن Fourier Transform است.

$$
\begin{array}{rccc}
 & 0 & 1 & 0 \\
1/6 * & 1 & 2 & 1 \\
 & 0 & 1 & 0
\end{array}
$$

$$
\begin{array}{rccc}
 & 1 & 1 & 1 \\
1/9 * & 1 & 1 & 1 \\
 & 1 & 1 & 1
\end{array}
$$

$$
\begin{array}{rccc}
 & 1 & 1 & 1 \\
1/10 * & 1 & 2 & 1 \\
 & 1 & 1 & 1
\end{array}
$$

$$
\begin{array}{rccc}
 & 1 & 2 & 1 \\
1/16 * & 2 & 4 & 2 \\
 & 1 & 2 & 1
\end{array}
$$

تصویر ۲.۷.۴.۱ – ماسک‌های convolution کوچک ۳×۳

۲.۷.۵ Low-Pass Filter:

در قسمت قبل توضیح مختصری درباره ی دو فیلتر high-pass وlow-pass دادیم. حال میخواهیم فیلتر low-pass(پائین گذر) را بطور دقیق تر در تصاویر بررسی کنیم.

فیلتر پائین گذر(در بحث پردازش تصویر) را میتوان برای از بین برد نویز در تصویری سیاه سفید بکار برد. تغییری که این فیلتر روی عکس میگذارد با توجه به ماسک پیچیدگی(convolution) عموما کم است. اگر به ماسک‌های convolution ای که در تصویر ۲.۷.۴.۱ آورده شده اند دقت کنید ، متوجه خواهید شد که مقادیر بیشتر، در قسمت میانی ماسک قرار دارد و قسمت‌های کناری عموما مقادیر کمتری را شامل اند.(اگر مقادیر کناری بیشتر از مقدار مرکزی باشد تغییر محسوسی در تصویری که فیلتر به آن اعمال شده قابل مشاهده خواهد بود، در ادامه علت را خواهید فهمید.)

forum.iranled.com

در این جا رنگ پیکسل‌های دو تصویر با مقیاس خاکستری را بر روی کاغذ آورده ایم که تصویر سمت راست دارای فرکانس فاصله ای کم (Low Spatial Frequency) و تصویر سمت چپ دارای فرکانس فاصله ای زیاد (High Spatial Frequency) می باشد.

150	150	150	150	150
1	1	1	1	1
150	150	150	150	150
1	1	1	1	1
150	150	150	150	150
1	1	1	1	1
150	150	150	150	150
1	1	1	1	1
150	150	150	150	150
1	1	1	1	1

تصویر ۲.۷.۵.۲

150	150	150	150	150
150	150	150	150	150
150	150	150	150	150
150	150	150	150	150
150	150	150	150	150
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

تصویر ۲.۷.۵.۱

حال می خواهیم نتیجه ی اعمال فیلتر پائین گذر را در تصویر سمت راست با ماسک پیچیدگی (Convolution) ۳×۳ ای که در تصویر ۲.۷.۴.۱ صفحه ی قبل آمده(اولین ماسک-بالای تصویر ۲.۷.۴.۱) بررسی کنیم.

150	150	150	150	150
150	150	150	150	150
150	150	150	150	150
150	150	150	150	150
125	125	125	125	125
25	25	25	25	25
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

تصویر ۲.۷.۵.۳ – نتیجه ی اعمال فیلتر پائین گذر بر روی تصویر ۲.۷.۵.۱

در مقایسه ی نتیجه و تصویر اصلی ، نتیجه در حالت واقعی به صورت محو دیده می‌شود.

forum.iranled.com

۲.۷.۶ Median Filter:

فیلتر میانگین(median) ، همانطور که از نامش پیداست میانگین رنگ ۳×۳(یا ۵×۵ یا ...) پیکسل را در پیکسل مرکزی قرار میدهد. برای این فیلتر ، نیازی به ماسک پیچیدگی(convolution) نیست چون تمام ضرایب ۱ است.

100	100	100	100	100
50	50	50	50	50
100	100	100	100	100
50	50	50	50	50
100	100	100	100	100
50	50	50	50	50
100	100	100	100	100
50	50	50	50	50
100	100	100	100	100
50	50	50	50	50

تصویر ۲.۷.۶.۱ – نتیجه ی اعمال median بر روی تصویر ۲.۷.۵.۲(سمت چپ)

۲.۷.۷ High-Pass Filter:

فیلتر بالاگذر یا High-Pass (همانطور که قبلا گفته شد)، لبه‌های (edges) تصویر را شارپ تر می کنید. تقریبا فیلترهای بالاگذر و پائین گذر نقطه ی مقابل هم هستند.

0	-1	0
-1	5	-1
0	-1	0

-1	-1	-1
-1	9	-1
-1	-1	-1

1	-2	1
-2	5	-2
1	-2	1

تصویر ۲.۷.۷.۱ – ماسک‌های پیچیدگی(convolution) برای فیلتر بالاگذر

forum.iranleu.com

فیلتر بالاگذر قسمت هایی از تصویر که پیکسل هایش رنگی مشابه هم دارند را تغییر نخواهد داد.

150	150	150	150	150
150	150	150	150	150
150	150	150	150	150
150	150	150	150	150
255	255	255	255	255
0	0	0	0	0
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

تصویر ۲.۷.۷.۲ – نتیجه ی اعمال فیلتر بالاگذر بر روی تصویر ۲.۷.۵.۱ (سمت راست)

255	255	255	255	255
0	0	0	0	0
255	255	255	255	255
0	0	0	0	0
255	255	255	255	255
0	0	0	0	0
255	255	255	255	255
0	0	0	0	0
255	255	255	255	255
0	0	0	0	0

تصویر ۲.۷.۷.۳ – نتیجه ی اعمال فیلتر بالاگذر بر روی تصویر ۲.۷.۵.۲ (سمت چپ)

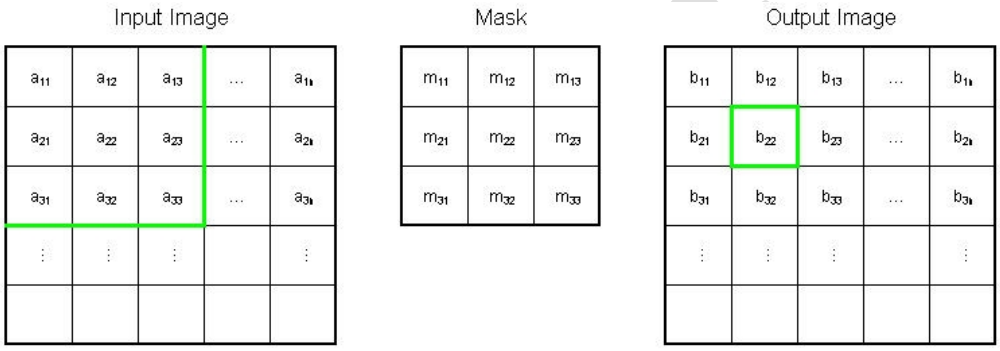
forum.ir.com

۲.۷.۸ بررسی نحوه ی اعمال ماسک‌های Convolution:

۲.۷.۸.۱ ماسک تک‌بُعدی

در فیلتر هایی که تنها از یک ماسک convolution برای محورهای xوy استفاده کرده‌اند، برای اعمال ماسک از روش زیر استفاده می‌کنند. این نوع از ماسک‌های convolution در فیلتر Edge Detection گروه دوم(Laplacian) و در موارد دیگر کاربرد دارد.البته توجه داشته باشید که ممکن است ابعادش بزرگتر باشد، مانند ۵×۵ ، ۷×۷ و... .(به دلیل نیاز به پیکسل مرکزی، باید ابعاد از اعداد فرد باشد.)

برای اعمال ماسک بر روی تصاویر مقیاس خاکستری(Grayscale) باید بصورت زیر عمل کنیم.



تصویر ۲.۷.۸.۱

با توجه به تصویر بالا، ماسک را در هر نقطه از تصویر ورودی قرار میدهیم(مختصات x, y فرض شود.). حال تمام نقاط متناظر را به هم ضرب میکنیم، سپس تمام حواصل را با هم جمع می کنیم، در طرف دیگر تمام مقادیر ماسک را باهم جمع می کنیم(اگر حاصل صفر شد، آنرا یک فرض می کنیم)، حال این دو مقدار را بر هم تقسیم می کنیم و حاصل را در پیکسل به مختصات x, y قرار میدهیم(به صورت زیر).همین کار را برای تک تک پیکسل‌های تصویر به جز حواشی آن انجام می دهیم.

```
Variable1 = (a11*m11) + (a12*m12) + (a13*m13) + (a21*m21) + … + (a33*m33)
Variable2 = m11+m12+m13+m21+m22+…+m33
If (Variable2 = 0) {Variable2=1}
Result=Variable1/Variable2
```

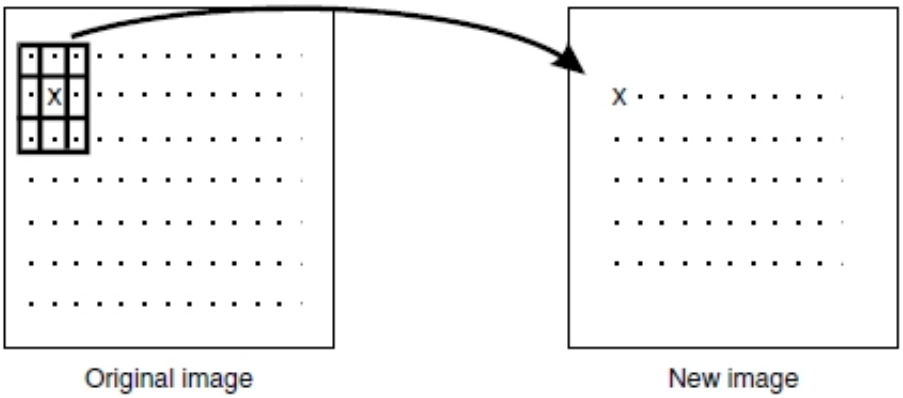
۲.۷.۸.۲ ماسک دوبُعدی

برای پیاده سازی ماسک دو بُعدی برای فیلتر Edge Detection (روش Sobel)، باید از روشی که در ادامه توضیح داده میشود استفاده کنید. در اینجا منظور از ماسک دو بُعدی ، این است که فیلتر مدّ نظر از دو ماسک مجزا، برای محورهای x, y استفاده کرده است. برای رسیدن به هدف، باید هر کدام از ماسک ها را بطور جداگانه طبق روش قبل، اعمال نمود با این تفاوت که مقدار حاصل از هر ماسک را (ResultX, ResultY) بنا بر یکی از روش‌های زیر(یا روش دلخواه دیگر) در پیکسل x, y قرار داد.

$$|Result| = (ResultX^2 + ResultY^2)^{1/2}$$

$$|Result'| = |ResultX| + |ResultY|$$

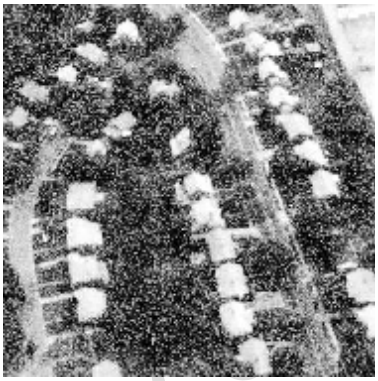
تمام این توضیحات برای تصاویر مقیاس خاکستری است، برای تصاویر رنگی هم کافی است تا تمام عملیات را بطور جداگانه برای هرکانال رنگ هر پیکسل، انجام دهید.



تصویر ۲.۷.۸.۲

۲.۷.۹ نتایج اعمال فیلترهای پائین و بالا گذر و فیلتر میانگین:

۱. فیلتر پائین گذر:



تصویر ۲.۷.۹.۱ – تصویر اصلی نویز دار

تصویر ۲.۷.۹.۲ – فیلتر پائین گذر با ماسک #6

تصویر ۲.۷.۹.۳ – فیلتر پائین گذر با ماسک #10

تصویر ۲.۷.۹.۴ – فیلتر پائین گذر با ماسک #9

تصویر ۲.۷.۹.۵ – فیلتر پائین گذر با ماسک #16

تصویر ۲.۷.۹.۶ – فیلتر میانگین ۳x۳

forum.anhd.com

۲. فیلتر میانگین:

تصویر ۲.۷.۹.۷ – تصویر اصلی

تصویر ۲.۷.۹.۸ – فیلتر میانگین ۳×۳

تصویر ۲.۷.۹.۹ – فیلتر میانگین ۵×۵

تصویر ۲.۷.۹.۱۰ – فیلتر میانگین ۷×۷

۳. فیلتر بالاگذر

تصویر ۲.۷.۹.۱۱ – فیلتر بالاگذر با ماسک #1

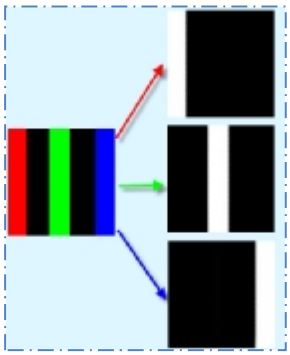
تصویر ۲.۷.۹.۱۲ – فیلتر بالاگذر با ماسک #2

forum.iran

تصویر ۲.۷.۹.۱۳ – فیلتر بالاگذر با ماسک #3

۲.۷.۱۰ اعمال عملیات بر روی تصاویر رنگی

ممکن است بگوئید که تمام این تصاویر مقیاس خاکستری است(grayscale) پس تصویر رنگی چه می شود؟ راه حل بسیار ساده است. همان طور که میدانیم تصویر رنگی متشکل از رنگ پیکسل ها است. این رنگ ممکن است سیستم‌های رنگی متفاوتی داشته باشد. سیستمی که ما بر روی آن بحث می کنیم و خواهیم کرد ، RGB است. سیستم RGB متشکل است از ۳ کانال کاملا مجزا ، بطوری که رنگ هر پیکسل دارای مقادیر مجزای قرمز ، سبز و آبی است. پس برای اعمال تمام عملیات فوق بر روی تصاویر رنگی با سیستم RGB ، میتوان تمام عملیات را به طور مجزا بر روی تک تک کانال‌های رنگ انجام داد.(یعنی به جای عکس مقیاس خاکستری، تک تک کانال‌های رنگ تصویر را قرار میدهیم)

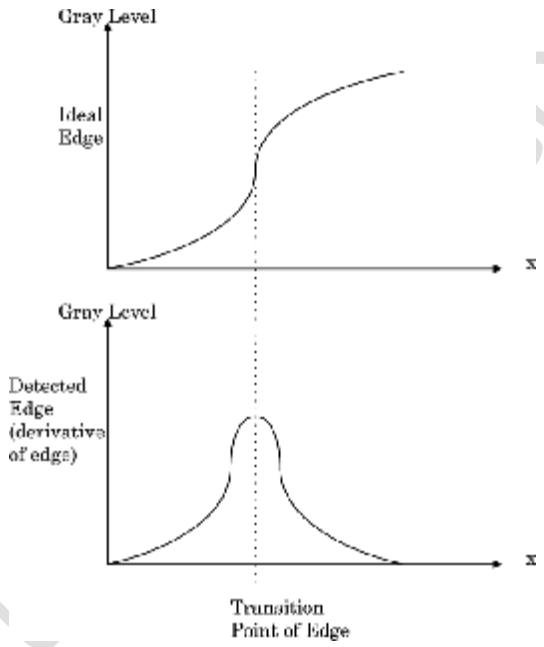


تصویر ۲.۷.۱۰.۱ – استخراج کانال‌های قرمز، سبز و آبی

forum.iranled.com

۲.۷.۱۱ Edge Detection:

یکی از اساسی ترین و مهم ترین فیلترها در پردازش تصویر، فیلتری است که لبه های(Edge) تصویر را تشخیص دهد. روش‌های گوناگونی طراحی شده که هر کدام بسته به نوع بازده در موارد مختلف کاربرد دارد. با استفاده از این فیلتر قادر خواهید بود تا اطلاعاتی از اندازه ی شیء و ابعادش بدست آورید که این اطلاعات در پردازش اشیاء بسیار حائز اهمیت می باشد.



تصویر ۲.۷.۱۱.۱ – نمای کناری از یک edge ایده آل

لبه(Edge) به قسمتی از تصویر گفته می شود که در آن رنگ از مقدار کم به زیاد سوق می یابد یا بلعکس. یک فیلتر Edge Detection باید رنگ هایی را از خود عبود دهد که در نمودار پائینی شکل بالا نمایان شده است و به جز آنها ، بقیه ی تصویر تیره رنگ باشد.(پس زمینه ی تیره)

مشکلی که وجود دارد این است که چگونه لبه ها را در تمام مختصات تصویر تشخیص دهیم. اکثر روش ها جزء دو گروه هستند، گروه اول : Gradient و گروه دوم : Laplacian .

گروه اول:

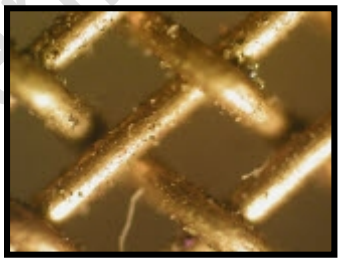
در این روش از هشت عدد ماسک Convolution(برای دقت بیشتر) استفاده می شود، که در جهت‌های اصلی و فرعی بنا شده اند. روش Sobel جزء این گروه است. در این روش از دو ماسک Convolution برای محورهای x و y استفاده میشود، البته نوع ۸ ماسکی روش Sobel نیز وجود دارد.

در Sobel همان طور که گفته شد، یکی از ماسک‌های برای محور Xها و دیگری برای محور Yها می باشد.

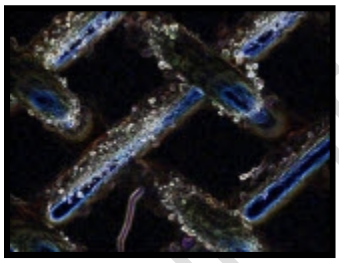
-1	0	+1
-2	0	+2
-1	0	+1

+1	+2	+1
0	0	0
-1	-2	-1

تصویر ۲.۷.۱۱.۲ – ماسک‌های convolution برای Sobel – برای محور y – برای محور x



تصویر ۲.۷.۱۱.۳ – تصویر اصلی



تصویر ۲.۷.۱۱.۴ – روش Sobel بر روی تصویر رنگی

تصویر ۲.۷.۱۱.۵ – تصویر اصلی

تصویر۲.۷.۱۱.۶ – روش Sobel

forum.tanlec.com

گروه دوم:

گروه دوم که به Laplacian معروف است، از یک ماسک Convolution ۵×۵ به جای ماسک ۳×۳ استفاده می شود. لازم به ذکر است که این ماسک برای هردو محور x ها و y ها اعمال میشود.

این روش از Edge Detection دارای حساسیت بیشتری به نویز است که باعث می شود کاربرد گروه اول در تصاویری که نویز دارند، بیشتر شود.

تصویر ۲.۷.۱۱.۷ – Laplace

تصویر ۲.۷.۱۱.۸ – Sobel

forum.iranic.com

منابع:

- http://en.wikipedia.org/wiki/Brightness
- http://en.wikipedia.org/wiki/RGB_color_model
- http://msdn.microsoft.com/enus/library/system.drawing.imaging.bitmapdata.scan0.aspx
- http://www.codeproject.com/KB/GDI-plus/csharpgraphicfilters11.aspx
- http://www.codonics.com/Support/NP/pdf/gamma.pdf
- http://www.pages.drexel.edu/~weg22/edge.html
- Image processing in C-analyzing and enhancing digital images. Second edition. R&D Publications.
- Nixon, M.S. and Aguado, A.S. 2008. Feature extraction and image processing. Second edition. Academic Press.

forum.iranled.com