به نام خدا

آزمایش پنجم

آشنایی با پردازش تصویر

هدف آزمایش:

- آشنایی با آشکارسازی لبه‌های تصویر
- آشنایی با یک روش کاهش نویز در تصویر

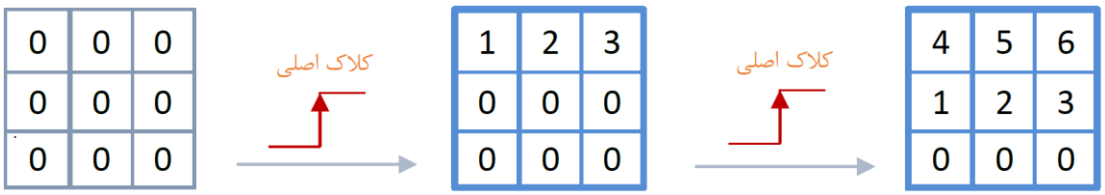
پیش گزارش:

برنامه‌ای بنویسید که بتواند با در نظر گرفتن ملاحظاتی که در ادامه گفته می‌شود، یک ماتریس ۳ در ۳ که شامل بخشی از مقادیر تصویر سیاه و سفید اصلی است را با خواندن از حافظه‌ی ROM بازسازی کند. مطمئن شوید که تمام داده‌های حافظه‌ی ROM در کلاک های اصلی مختلف در این ماتریس ۳ در ۳ به صورت مناسب قرار می‌گیرند. هدف از این کار این است که بتوانیم یک پنجره‌ی ۳ در ۳ روی تصویر بلغزانیم و پردازش لازم را بر روی آن انجام دهیم. نتایج حاصل از شبیه‌سازی را به عنوان پیش گزارش ارسال کنید.

- کلاک ورودی اصلی را ۲۵ مگاهرتز در نظر گرفته و ابتدا کلاکی سه برابر کلاک اصلی تولید کنید به گونه‌ای که در هر لبه از این کلاک تولید شده بتوانید یک خانه از حافظه را خوانده و مقدار آن را ذخیره کنید. بدین ترتیب در هر لبه از کلاک اصلی محتویات سه خانه از حافظه در دسترس است.
- این مقادیر خوانده شده را در یک سطر از حافظه ۳ در ۳ به گونه‌ای ذخیره کنید که بتوان در کلاک اصلی بعدی مقادیر جدید خوانده شده را به این حافظه اضافه نمود. به این ترتیب می‌توانیم یک

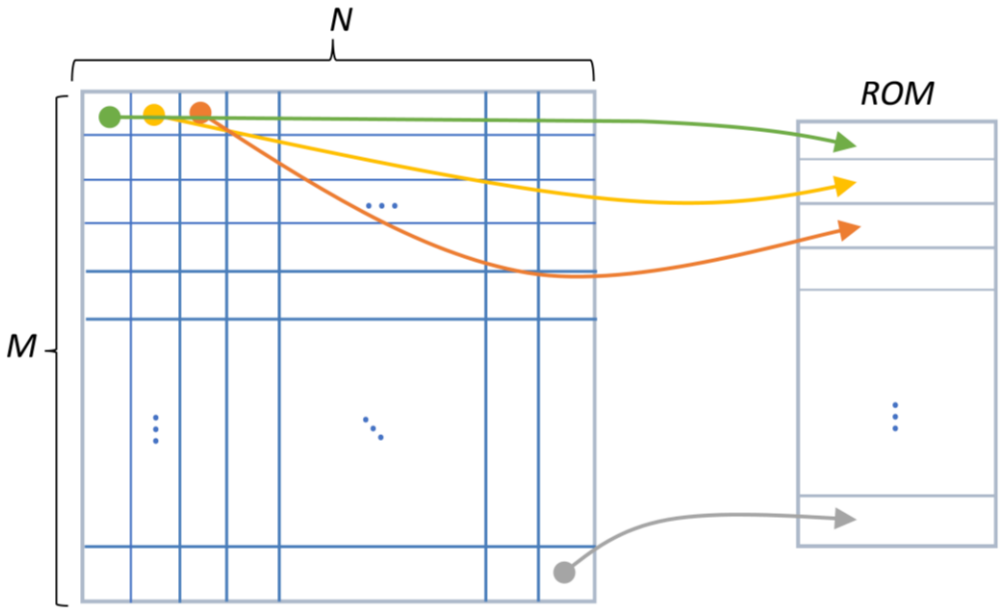
حافظه ۳ در ۳ شامل مقادیر بخشی از تصویر داشته باشیم. به عبارت دیگر می‌خواهیم در هر لحظه اطلاعات بخشی از تصویر (به صورت بلوک ۳ در ۳) در دسترس باشد.

توجه: حافظه ۳ در ۳ در نظر گرفته شده در ابتدا با صفر پر شده است و باید در هر کلاک اصلی مقادیر جدید در یک سطر(یا ستون) جدید نوشته شوند یا مقادیر قبلی شیفت داده شده (مثلاً به اندازه یک سطر یا ستون) و مقادیر جدید جایگزین گردند.



شکل ۱: نحوه قرار دادن بیت‌های خوانده شده از حافظه

توجه: اطلاعات ذخیره شده از تصویر در ROM به صورت زیر هستند. بنابراین اگر ابعاد تصویر برای مثال ۱۰ در ۱۰ پیکسل باشد، در اولین بلوک ۳ در ۳ مقادیر موجود در آدرس‌های ۱و۲و۳ و ۱۱و۱۲و۱۳ و ۲۱و۲۲و۲۳ از حافظه‌ی ROM قرار داده می‌شود.



شکل ۲: تصویر ذخیره شده در ROM

دستور کار:

این آزمایش شامل دو بخش است. در بخش اول قرار است لبه‌های تصویر را با استفاده از الگوریتم sobel آشکارسازی کنید و در بخش دوم با استفاده از روش میانه، نویز موجود در تصویر (نویز نمک و فلفل) را کاهش خواهید داد. ابعاد هر دو تصویر ۲۲۴×۲۲۴ است. برای هر بخش در یک پروژه‌ی جداگانه ابتدا فایل coe. مربوط به تصویر اولیه را که به پیوست ارسال شده، به یک حافظه‌ی ROM بدهید. سپس همان‌طور که در دستور کار توضیح داده شده عملیات پردازش تصویر را انجام دهید.

برای دیدن خروجی عملیات پردازش تصویر، باید دیتای خروجی پردازش را در یک فایل ذخیره کرده و مشاهده کنید. یک راه ساده این است که دیتای پردازش شده را در testbench به صورت یک فایلtxt. ذخیره نمایید. سپس این فایل را در نرم‌افزار MATLAB باز کرده و آن را به صورت تصویر مشاهده کنید. برای ذخیره‌ی دیتا می‌توانید از دستوراتی که در کد زیر آمده است استفاده کنید. در این دستورات ابتدا اطلاعات تصویر در یک آرایه به نام outfile ریخته شده و پس از اتمام این کار در فایل output.txt ذخیره شده است.

```
60    reg [7:0] outfile  [0 : 224*224 - 1];
61    integer fd;
62    integer i;
63
64    initial
65    begin
66     fd = $fopen("output.txt", "w");
67    end
68
69    always @(posedge flag)
70     begin
71     for(i=0; i<224*224; i=i+1)
72     begin
73     $fwrite(fd, "%d\n", outfile[i][7:0]);
74     end
75     $fclose(fd);
76    end
77
78    always @(posedge clk)
79    begin
80    if (flag==0)
81    outfile[addr]<=dout;
82    end
83
```

سپس با استفاده از نرم‌افزار MATLAB این فایل به صورت تصویر نمایش داده شده است.

```
1 -    clc
2 -    clear all
3 -    A=importdata ('output.txt');
4 -    B=reshape(A,[224 224]);
5 -    imshow(B',[]);
6
```

روش فوق صرفاً یک پیشنهاد ساده است و می‌توانید از شیوه‌های دیگر نیز برای ذخیره و مشاهده‌ی تصویر استفاده کنید. برای مثال می‌توان در testbench خروجی را مستقیماً به صورت یک فایل تصویر با فرمت bmp. نیز ذخیره نمود که می‌توانید به صورت اختیاری در مورد جزئیات آن تحقیق کرده و انجام دهید.

این آزمایش با استفاده از testbench شبیه‌سازی می‌شود و بر روی برد پیاده‌سازی نخواهد شد.

۱- در این آزمایش ابتدا می‌خواهیم فایل تصویر شکل ۳ (داده شده در فایل ImgGray_Sobel_8bit.coe) را از یک حافظه ROM خوانده و با انجام یک پردازش ساده بر روی آن، لبه‌های موجود در تصویر را آشکارسازی نماییم.

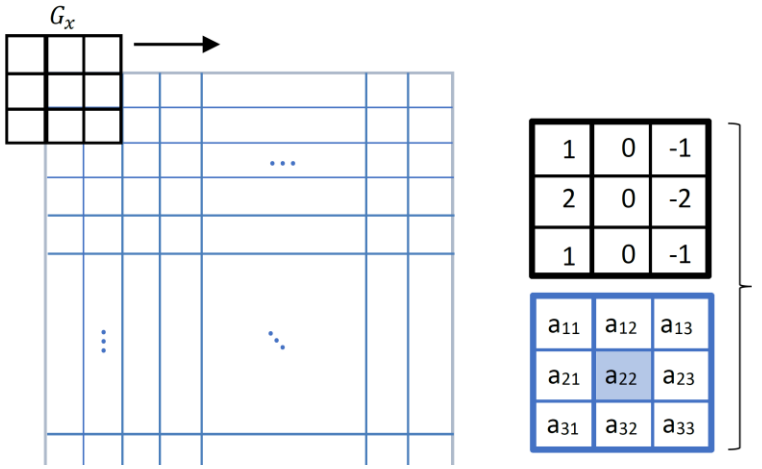
شکل ۳

برای پیاده‌سازی این آشکارساز لبه، از الگوریتم ساده‌ی sobel استفاده می‌کنیم. در این الگوریتم ابتدا ماتریس‌های عملگر در راستای محور افقی و عمودی را به فرم زیر تعریف می‌کنیم:

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \qquad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

که در آن ماتریس Gx (Gy) تغییرات موجود در تصویر را در راستای افقی (عمودی) استخراج می‌کند. برای به دست آوردن این تغییرات کافی است این ماتریس‌ها را بر روی تصویر اصلی در راستای افقی و عمودی حرکت داده و با جمع کردن حاصل ضرب مقادیر همپوشانی شده از تصویر اصلی با مقادیر ماتریس‌های فوق، به هر پیکسل یک **ضریب وزنی** اختصاص دهیم (شکل ۴). به این ترتیب درنهایت دو عدد Ax و Ay خواهیم داشت که با تشکیل مقدار $A=|Ax|+|Ay|$ می‌توان لبه‌های موجود در تصویر را با انتخاب یک سطح آستانه‌ی مناسب به صورت تقریبی پیدا نمود (مقدار A با سطح آستانه مقایسه می‌شود، اگر بیشتر بود آن پیکسل به عنوان لبه شناسایی شده و در خروجی رنگ سیاه به آن اختصاص می‌دهیم. اگر کمتر بود در

خروجی رنگ سفید به آن پیکسل اختصاص داده می‌شود). در شکل۴ روند کلی الگوریتم برای محور افقی نمایش داده شده است. توجه شود که هنگامی که ماتریس Gx به انتهای هر ستون از تصویر می‌رسد آن را به اول سطر بعد انتقال داده و مجدداً حرکت می‌دهیم. این عمل ممکن است باعث شود که در سطرها و ستون‌های ابتدایی نتوانیم لبه‌ها را آشکارسازی نماییم که البته چندان اهمیتی نخواهد داشت.

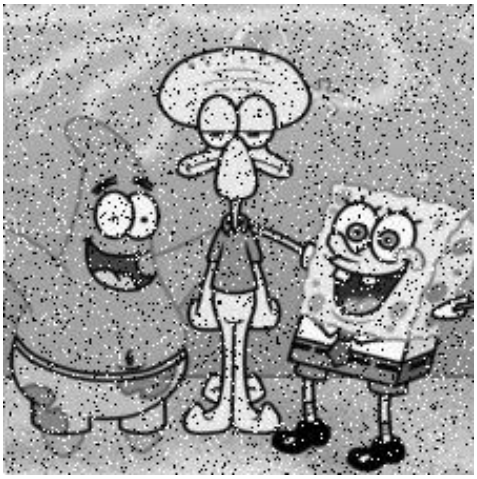


شکل ۴: نحوه‌ی تعیین وزن هر پیکسل

از آنجایی که نمی‌خواهیم این کار را به صورت offline انجام دهیم و می‌خواهیم در هر لحظه مشخص کنیم که هر پیکسل آیا یک لبه است یا خیر، به صورتی که در پیش گزارش گفته شده عمل می‌کنیم.

سپس با داشتن این بلوک از تصویر عملیات دو ماتریس فوق را بر روی این بلوک اعمال کرده و ضریب وزنی پیکسل خروجی را تعیین می‌کنیم. با تعیین یک سطح آستانه، رنگ مناسب برای این پیکسل را تعیین کرده (مشخص کردن پیکسل به عنوان لبه) و رنگ تعیین شده را بر اساس کلاک اصلی به خروجی انتقال می‌دهیم.

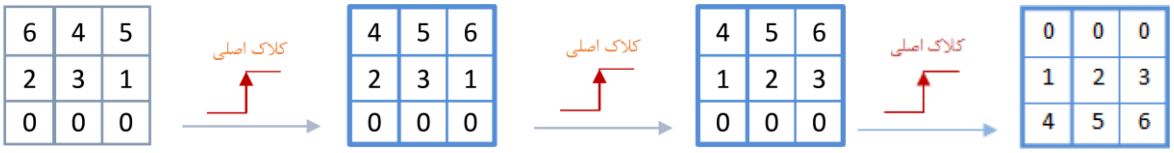
۲- در قسمت دوم این آزمایش می‌خواهیم با استفاده از یک فیلتر میانه ۳ در۳، نویزهای موجود در تصویر شکل ۵ (داده شده در فایل imgGray_8bit_median.coe) را کاهش دهیم.



شکل ۵

برای این کار با در نظر گرفتن همان پنجره‌ی ۳ در ۳ و همان نحوه خواندن داده‌ها، بایستی مابین داده‌های موجود در پنجره، میانه‌ی آن‌ها را به عنوان خروجی عملگر فیلتر میانه یا همان پیکسل خروجی تصویر به دست آورد. همان‌طور که می‌دانیم به دست آوردن میانه مابین ۹ عدد، بسیار طولانی و زمان‌بر است.

برای حل این مشکل از روش میانه یابی سریع استفاده می‌کنیم. این روش به این صورت است که ابتدا ۳ پیکسل سطر اول پنجره ۳ در ۳ را به ترتیب از مقدار کوچک‌تر به بزرگ‌تر مرتب کرده و همین روال را برای دو سطر دیگر انجام می‌دهیم. سپس ستون اول پنجره را نیز از مقدار کوچک‌تر به بزرگ‌تر مرتب کرده و همین روال برای دو ستون دیگر پنجره اجرا می‌شود. درنهایت مقدار میانه کل ۹ عدد برابر خواهد بود با میانه سه عددی که روی قطر اصلی پنجره ۳ در ۳ قرار دارند (شکل ۶).



شکل ۶

۱- برای هر دو پروژه از بخش summary report میزان منابع مصرف شده و حداکثر فرکانس کاری را تعیین کنید. سپس ساختار طراحی شده را با استفاده از نرم‌افزار Plan ahead نمایش دهید.

۲- **اختیاری:** برنامه‌ای بنویسید که یک تصویر رنگی را به یک تصویر سیاه و سفید تبدیل کند.

نحوه‌ی تحویل:

- فایلی که در سامانه آپلود می‌کنید باید یک فایل فشرده حاوی پیش گزارش، گزارش کار و فولدر کامل پروژه باشد.
- فایل ارسالی را با حروف انگلیسی و با فرمتی مشابه myname_9511111_exp1 نام‌گذاری کنید.
- گزارش کار باید حاوی **توضیحات** لازم در مورد کدهای نوشته شده و **مراحل انجام کار** و **نتایج شبیه‌سازی‌ها** باشد. **کیفیت گزارش** کار به طور جدی در نمره اثرگذار است.
- پیش گزارش و گزارش کار را به صورت تایپ شده با قلم B Nazanin اندازه‌ی ۱۴ بنویسید.
- سعی کنید کدهای خود را خوانا و مرتب نوشته و کامنت گذاری کنید.
- آزمایش‌ها باید به صورت انفرادی انجام و تحویل داده شود.