**به نام خدا**

**گزارش پروژه پایانی سیگنال ها و سیستم ها**

**عنوان پروژه : پردازش تصویر**

**نام و نام خانوادگی : غزال ادریس آبادی**

**استاد درس : دکتر اکبری**

**پاییز ۹۹**

* **روش های ذخیره تصاویر :**

در متلب چهار نوع تصویر پشتیبانی می شود :

۱- تصاویر سطح خاکستری ( Grayscale | Grey-level Images )

۲- تصاویر باینری ( ‌‌Binary Images )

۳- تصاویر شاخص گذاری شده ( Indexed Images )

۴- تصاویر RGB

* **تصاویر سطح خاکستری ( Grayscale | Grey-level Images ) :** در این تصاویر از ۸ بیت برای نمایش مقدار شدت رنگ هر پیکسل استفاده می شود. پیکسل با مقدار شدت صفر، رنگ سیاه و پیکسل با مقدار شدت ۲۵۵، رنگ سفید را نمایش می دهد. همچنین پیکسل هایی که مقادیر بین ۰ و ۲۵۵ دارند، طیف های خاکستری را نمایش می دهند. به این تصاویر، تصاویر تک رنگ یا مونوکروم نیز گفته می شود. تصاویر سطح خاکستری، توسط آرایه های دو بعدی و مقادیر پیکسل ها توسط یک عدد ۸ بیتی نمایش داده می شوند.
* **تصاویر باینری ( ‌‌Binary Images ) :** در این تصاویراز ۱ بیت برای نمایش مقدار پیکسل ها استفاده می شود. مقدار ۱ به معنای رنگ سفید و مقدار ۰ به معنای رنگ سیاه است. تصاویر سیاه و سفید توسط آرایه های دو بعدی نمایش داده می شوند. این تصاویر حجم کمی دارند که از مهم ترین مزیت آن ها محسوب می شود.
* **تصاویر شاخص گذاری شده ( Indexed Images ) :** این تصاویر ماتریسی از مقادیر صحیح یا integer هستند. در این ماتریس، هر مقدار صحیح به یک سطر خاص از مقادیر RGB، در یک ماتریس ( Map یا نقشه ) ثانویه به نام نقشه رنگ ( Colour Map ) اشاره دارد.
* **تصاویر RGB :** در این تصاویر هر پیکسل رنگی توسط سه مولفه قرمز ( R )، سبز ( G ) و آبی ( B ) نمایش داده می شود. هر تصویر RGB در متلب، متناظر با یک آرایه سه بعدی به ابعاد MxNx3 است که M بیانگر ارتفاع، N بیانگر پهنا و عدد ۳ بیانگر تعداد مولفه های رنگی تصویر RGB است.در تصاویر RGB و از نوع double، محدوده مقادیر شدت پیکسل ها، بازه [1,0] است. در تصاویر RGB از نوع Unit8 و Unit16 محدوده مقادیر به ترتیب بازه ی [255,0] و [65535,0] است.

**بیشتر الگوریتم های پردازش تصویر در متلب از تصاویر باینری یا سطح خاکستری برای انجام عملیات پردازشی استفاده می کنند.**

* **روش های تشخیص لبه :**

این دسته از روش ها که به آن ها روش های تشخیص لبه ( Edge Detection ) نیز گفته می شود، یکی از مهم ترین روش های پردازش تصویر در متلب محسوب می شوند. از روش های استخراج لبه، برای جداسازی اشیا ( Objects ) موجود در تصویر استفاده می شود. در روش های تشخیص لبه، هدف شناسایی نقاطی در تصاویر دیجیتال است که در این نقاط، روشنایی تصویر (Image Brightness ) نسبت به نقاط همسایه، به شدت تغییر می کند. روش های تشخیص لبه، به دو دسته روش های مبتنی بر جستجو ( Search-Based ) و مبتنی بر عبور از صفر ( Zero-Crossing-Based ) تقسیم بندی می شوند.

**روش مبتنی بر جستجو ( Search-Based ) :** در این روش ابتدا قدرت لبه ها با استفاده از معیارهایی نظیر اندازه گرادیان ( Gradient Magnitude ) محاسبه می شود که معمولا از طریق محاسبه مشتق مرتبه اول صورت می گیرد. سپس با استفاده از تخمین محاسبه شده از جهت محلی ( Local Orientation ) لبه ها ( جهت گرادیان )، بیشینه جهت دار محلی (Local Directional Maxima ) متناظر با اندازه گرادیان جستجو و لبه ها شناسایی می شوند. الگوریتم هایی نظیر الگوریتم Canny، Prewitt و Sobel، جزء الگوریتم های تشخیص لبه مبتنی بر مشتق مرتبه اول هستند.

**روش مبتنی بر عبور از صفر ( Zero-Crossing-Based ) :** در این روش نقاط عبور از صفر در مشتق مرتبه دوم محاسبه شده از تصویر، جستجو خواهند شد و از این طریق، لبه های موجود در تصاویر شناسایی می شوند. الگوریتم های Zerocross و LOG در این دسته قرار می گیرند.

* **الگوریتم های تشخیص لبه :**
* **الگوریتم Sobel :** یک الگوریتم مبتنی بر جستجو است. همانطور که در بالا اشاره کردیم، لبه ها را با استفاده از تخمین مشتق پیدا می کند و آن ها را در نقاطی برمی گرداند که گرادیان تصویر max، I است. در این فیلتر دو mask عمودی و افقی وجود دارد که sobel mask افقی، بیشتر لبه های افقی و sobel mask عمودی، لبه های عمودی را مشخص می کند.

اگر و به ترتیب تصاویر فیلتر شده به وسیله mask افقی و عمودی باشند، آنگاه تصویر لبه های تصویر را بهتر نشان می دهد.

\*\* برای کاهش هزینه محاسبات به جای رابطه بالا از تقریب استفاده می شود.

* **الگوریتم Prewitt :** این الگوریتم شبیه به الگوریتم Sobel است با این تفاوت که ضرایب mask آن ها باهم فرق می کند.
* **الگوریتم Canny :** یک الگوریتم مبتنی بر جستجو است. این الگوریتم به خوبی با نویز مقابله و لبه های ضعیف را کشف می کند. سه هدف اصلی این الگوریتم عبارتند از :

۱- تضعیف نویز

۲- پیدا کردن نقاطی که بتوان آن ها را به عنوان لبه درنظر گرفت.

۳- جذب نقاطی که احتمال لبه بودن آن ها کم است.

**ورودی ها :**

۱- یک تصویر ورودی

۲- یک پارامتر به نام سیگما جهت مقدار نرم کنندگی تصویر

۳- یک حد آستانه بالا Th

۴- یک حد آستانه پایین Tl

**مراحل :**

۱- ابتدا تصویر رنگی جهت لبه یابی بهتر به یک تصویر سطح خاکستری تبدیل می شود.

۲- نویز از تصویر دریافتی حذف می شود. ( از آنجایی که فیلتر gaussian از یک simple mask برای حذف نویز استفاده می کند، الگوریتم canny آن را به کار می گیرد ).

۳- در یک تصویر سطح خاکستری مکانی که بیشترین تغییرات را داشته باشد، به عنوان لبه درنظر گرفته می شود. این مکان ها با استفاده از الگوریتم sobel به دست می آیند. سپس لبه های مات یافت شده به لبه های تیزتر تبدیل می شوند.

۴- بعضی از لبه های کشف شده در مرحله قبل، واقعا لبه نیستند و نویزند. این نویزها باید توسط حد آستانه هیسترزیس فیلتر شوند. هیسترزیس از دو حد آستانه بالاتر ( Th ) و حد آستانه پایین تر ( Tl ) استفاده می کند. در canny بهتر است نسبت Th به Tl سه به یک باشد.

* **الگوریتم Roberts :** این الگوریتم شبیه به الگوریتم canny است با این تفاوت که پیکسل های کمتری را برای تقریب گرادیان به کار می گیرد. همچنین نسبت به canny حساسیت کمتری به نویز دارد.
* **الگوریتم Zerocross :** یک الگوریتم مبتنی بر عبور از صفر است. این الگوریتم قسمت هایی از لاپلاس یک تصویر را جستجو می کند که مقدار لاپلاس از صفر می گذرد. یعنی نقاطی که لاپلاس، علامت را تغییر می دهد.
* **الگوریتم Laplacian of gaussian(LOG)** : یک الگوریتم مبتنی بر عبور از صفر است. لاپلاس یک اندازه گیری ایزوتروپیک دوبعدی از مشتق فضایی مرتبه دوم از یک تصویر است. لاپلاس یک تصویر، مکان هایی با تغییرات شدید را سریع نشان می دهد و به همین دلیل معمولا از آن برای تشخیص لبه ها استفاده می کنند.

در این پروژه به دو صورت لبه یابی انجام شده است :

۱- پیاده سازی دستی الگوریتم های لبه یابی

۲- استفاده از توابع آماده متلب جهت تشخیص لبه ها

**پیاده سازی دستی :**

در این بخش از الگوریتم های sobel، roberts برای تشخیص لبه های تصویر استفاده شده است که عملکرد هر کدام را مختصرا توضیح می دهیم.

الگوریتمsobel : در پیاده سازی این الگوریتم همان طور که در توضیحات بالا گفته شد، ابتدا مقادیر و با مشتق گیری محاسبه شده اند که لبه های افقی و عمودی را مشخص می کنند. در مرحله بعد مقدار S که همان جذر مجموع توان دو مقادیر است محاسبه شده است که تصویر نهایی را مشخص می کند.

الگوریتم roberts : عملکرد این الگوریتم شبیه الگوریتم canny است که قبل تر به طور کامل توضیح داده شد.

**توابع آماده متلب :**

در این بخش از الگوریتم های sobel، roberts، canny و zerocross برای تشخیص لبه ها استفاده کردیم. در مرحله اول عکس رنگی را تبدیل به عکس خاکستری می کنیم. سپس با استفاده از تابع edge الگوریتم موردنظر را روی عکس خاکستری اعمال کرده و نتیجه را نمایش می دهیم.

* **اضافه کردن نویز :**

قبل از معرفی نویزها لازم است با اصطلاح Geometric Enhancement آشنایی پیدا کنیم.

**Geometric Enhancement :** در پروسه های بارزسازی تصاویر از آن استفاده می شود. در حقیقت به فرآیندهایی که در هنگام بارزسازی تصاویر از اطلاعات مجاورتی پیکسل ها استفاده می شود، Geometric Enhancement می گویند. در کاربرد باعث می شود که لبه ها واضح دیده شوند و مرز مشخصی بین آن ها وجود داشته باشد. همچنین استفاده از آن جهت حذف نویزها الزامی است.

**نویز چیست؟** نویز پدیده ای است که یک سیگنال واقعی ( سیگنال مورد انتظار ) را تغییر می دهد و چیزی غیر از سیگنال مورد نظر ما ثبت می کند. معمولا نویزها دارای ماهیت اتفاقی می باشند و از قاعده خاصی تبعیت نمی کنند، اما برخی از نویزها قابل پیش بینی اند. عمدتا نویزها به صورت جمع شونده هستند به این معنی که نویز با جمع جبری به سیگنال اضافه می شود. به این نوع نویزها، نویزهای عمومی گفته می شود. البته باید دقت کنیم که خود نویزها می توانند مقادیر مثبت یا منفی داشته باشند.

در متلب با استفاده از دستور imnoise می توان نویزهای مختلفی به تصویر اضافه کرد. ورودی اول و دوم این تابع به ترتیب تصویر مورد نظر برای اعمال نویز و نوع نویز را تعیین می کند. به طور کلی پنج نوع نویز توسط این دستور می توان ایجاد کرد :

* **نویز white(uniform) :** از جنس نویزهای جمع شونده است و تابع نویز uniform می باشد. uniform به این معنی که اگر کل دامنه تغییرات درجه خاکستری را (۱۲۸+ و ۱۲۸-) در نظر بگیریم، احتمال ظهور نویزها با درجات مختلف باهم برابر است.
* **نویز gaussian :** از جنس نویزهای جمع شونده است و تابع نویز gaussian می باشد. یعنی نویزهای کوچکتر احتمال رخداد بیشتر و نویزهای بزرگتر احتمال رخداد کمتری دارند. برای استفاده از این نویز باید به عنوان ورودی، مقدار انحراف معیار و میانگین نویز تعیین شود. ورودی سوم تابع imnoise، میانگین نویز را تعیین می کند که مقدار پیش فرض آن برابر با ۰ است. ورودی چهارم نیز انحراف معیار را مشخص می کند که مقدار پیش فرض آن برابر ۰.۰۱ است.
* **نویز salt & pepper :** این نویز که به نویز فلفل نمکی معروف است در تصویر تعدادی پیکسل سفید و سیاه رنگ ایجاد می کند. احتمال رخداد این نویز در دو مقدار است : ۰ یا ۲۵۵. به طوری که یا سیگنال را از بین می برد یا کاملا آن را یک می کند و مقدار نویز را با مقدار سیگنال جایگزین می کند. ورودی سوم تابع imnoise، شدت نویز را مشخص می کند.
* **نویز poisson :** این نویز هیچ پارامتری برای تنظیم یا زیاد و کم کردن شدت نویز از ورودی نمی گیرد و فقط می تواند رندوم تصویر را نویزی کند.
* **نویز speckle :** از جنس نویزهای ضرب شونده است ( در سیگنال ضرب می شود ). اگر صفر باشد سیگنال را از بین می برد. میانگین این نویز صفر است اما انحراف معیار را به عنوان ورودی سوم می توان تنظیم کرد.
* **نویز periodic :** نویزهایی هستند که تکرار می شوند و متناوب اند.
* **حذف کردن نویز :**

نویزهای جمع شونده به راحتی قابل حذف شدن هستند. در Geometric Enhancement نویزهایی مثل نویزهای فلفل نمکی هم به راحتی حذف می شوند اما حذف کردن نویزهای متناوب با استفاده از این روش به سختی امکان پذیر است. بنابراین برای حذف این نویزها معمولا از فضای فرکانس استفاده می کنند.

برای حذف کردن نویزها ( غیر از نویزهای متناوب ) عموما از فیلترهای averaging استفاده می شود. با وجود نویزهای مختلف، با توجه به نوع نویز فیلتر مناسب انتخاب می گردد. مثلا برای حذف نویز gaussian از فیلتر averaging gaussian استفاده می شود. به این صورت که برای هر پیکسل با توجه به مقدار همسایگانش و وزن هر کدام، میانگین گیری صورت می گیرد. فیلتر دیگر، فیلتر median است که برای حذف نویزهای فلفل نمکی از آن استفاده می شود.

در این پروژه برای حذف نویز gaussian از فیلتر wiener2 استفاده شده است. این فیلتر یک فیلتر خطی است که با استفاده از مقدار واریانس نویزها را حذف می کند. به این صورت که هر چه مقدار واریانس کمتر باشد، تصویر صاف تری حاصل خواهد شد. ابتدا تصویر RGB را به یک تصویر سطح خاکستری تبدیل کرده و آن را نویزی می کنیم. اعداد [5 5] اندازه همسایگانی هستند که از آن ها برا تخمین تصویر اصلی استفاده می شود.

همچنین برای حذف نویز salt & pepperمی توان از فیلترهای median یا average استفاده کرد.

* **منابع :**

[**https://blog.faradars.org/image-processing-in-matlab/**](https://blog.faradars.org/image-processing-in-matlab/)

[**https://howsam.org/image-processing-matlab-toolbox/#%D8%AF%D8%B3%D8%AA%D9%88%D8%B1\_imnoise\_%D8%AF%D8%B1\_%D9%85%D8%AA%D9%84%D8%A8**](https://howsam.org/image-processing-matlab-toolbox/#%D8%AF%D8%B3%D8%AA%D9%88%D8%B1_imnoise_%D8%AF%D8%B1_%D9%85%D8%AA%D9%84%D8%A8)

[**https://mapscale.ir/%d9%85%d8%b9%d8%b1%d9%81%db%8c-%d9%86%d9%88%db%8c%d8%b2%d9%87%d8%a7-%d8%af%d8%b1-%d9%be%d8%b1%d8%af%d8%a7%d8%b2%d8%b4-%d8%aa%d8%b5%d9%88%db%8c%d8%b1/**](https://mapscale.ir/%d9%85%d8%b9%d8%b1%d9%81%db%8c-%d9%86%d9%88%db%8c%d8%b2%d9%87%d8%a7-%d8%af%d8%b1-%d9%be%d8%b1%d8%af%d8%a7%d8%b2%d8%b4-%d8%aa%d8%b5%d9%88%db%8c%d8%b1/)

[**http://www.7khatcode.com/1481/%D8%B1%D9%88%D8%B4-%D9%87%D8%A7%DB%8C-%D8%AD%D8%B0%D9%81-%D9%86%D9%88%DB%8C%D8%B2-%D8%AF%D8%B1-%D9%BE%D8%B1%D8%AF%D8%A7%D8%B2%D8%B4-%D8%AA%D8%B5%D9%88%DB%8C%D8%B1?show=1487#a1487**](http://www.7khatcode.com/1481/%D8%B1%D9%88%D8%B4-%D9%87%D8%A7%DB%8C-%D8%AD%D8%B0%D9%81-%D9%86%D9%88%DB%8C%D8%B2-%D8%AF%D8%B1-%D9%BE%D8%B1%D8%AF%D8%A7%D8%B2%D8%B4-%D8%AA%D8%B5%D9%88%DB%8C%D8%B1?show=1487#a1487)

[**https://www.mathworks.com/help/images/ref/wiener2.html**](https://www.mathworks.com/help/images/ref/wiener2.html)