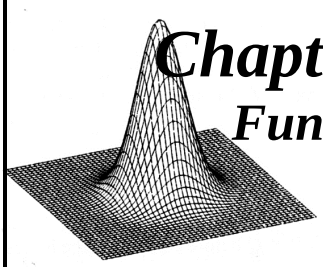


Chapter 2
Fundumentals of Digital Image Processing:

۲.۱.مفاهیم اولیه در پردازش تصویر

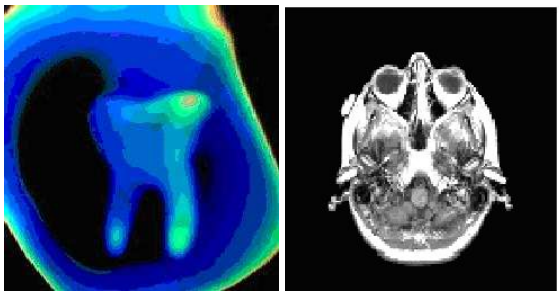
کاربردهای علم پردازش تصویر

علم پردازش تصویر از جمله علوم پرکاربرد و مفید در صنعت می باشد، که در زیر نمونه ای از کاربردهای پردازش تصویر در زمینه های مختلف آورده شده است.

الف) کاربردهای صنعتی مانند کنترل کیفیت بسته بندی دارو در یک کارخانه.

ب)کاربردهای امنیتی مانند تشخیص حرکت، تشخیص اثر انگشت، تشخیص چهره و تشخیص دست خط یا امضاء

ج) کاربردهای پزشکی مانند ارتقای ویژگی‌های تصاویر اشعهX، تولید تصاویر MRI از مغز و یا تصاویر مربوط به CTScan



ز) کاربردهای نظامی مانند تشخیص و هدف یابی خودکار اهداف متحرک یا ثابت توسط موشک های هوا به زمین

آشنایی با مفاهیم اولیه در پردازش تصویر

۱. آشنایی با مفهوم پیکسل در یک تصویر

پیکسل (Pixel) شکل مختصر Picture Elemens نقطه های بسیار ریز مربع شکلی هستند که از تجمع آنها، تصویر روی صفحه نمایش یا روی کاغذ (توسط چاپگر) شکل می گیرد. همان طور که بیت، کوچک ترین واحد اطلاعات قابل پردازش توسط کامپیوتر است، پیکسل نیز کوچک ترین عنصر سخت افزار و نرم افزار نمایشی یا چاپی است که برای شکل گرفتن تصاویر مورد استفاده قرار می گیرد. اگر برای هر پیکسل تنها دو رنگ (معمولاً سیاه و سفید) در نظر گرفته شود، توسط یک بیت از اطلاعات قابل کددهی است و در صورتی که بیش از دو بیت برای ارائه یک پیکسل استفاده شود، محدودهٔ رنگ ها یا سایه های خاکستری وسیع تری، قابل ارائه خواهد بود.

۲. آشنایی با مفهوم عمق بیتی:

شکل زیر نشان دهنده ردیف ها و ستون هایی از نقاط تصویر گرافیکی موجود در حافظه کامپیوتر است.

مقدار هر نقطه (پر یا خالی) در یک یا چند بیت اطلاعات ذخیره می شود. برای تصاویر سادهٔ تک رنگ، یک بیت برای نشان دادن هر نقطه کافیست، اما در تصاویر رنگی و سایه های خاکستری، هر نقطه نیاز به بیش از یک بیت اطلاعات دارد.

هرچه از بیت های بیشتری برای نشان دادن یک نقطه استفاده شود، رنگ ها و سایه های خاکستری بیشتری را می توان نشان داد.غلظت نقاط یا همان Resolation، وضوح تصویر را تعیین می کند. این ویژگی با واحد نقطه در اینچ یا همان bit-map بر روی مانیتور یا چاپ آن با پرینتر، کامپیوتر bit-map را به pixel برای نمایش بر روی مانتیور یا به ink dots برای چاپ تبدیل می کند.

اساس کار اسکنرهای Optical و دستگاه فاکس، تبدیل متن یا تصویر به bit-map می باشد. تصاویر bit-map را اغلب به عنوان تصاویری با خطوط موازی (raster) می شناسند. راه های دیگری نیز برای نشان دادن تصاویر وجود دارد، vector graphics یا object oriented graphics – است. با روش Cector graphics تصاویر با فرمول های ریاضی که تمام اشکال تصویر را تعریف می کند، نشان داده می شوند. این روش انعطاف پذیرتر از bit-map می باشد، چرا که اگر آنها را با اندازه های مختلفی نیز بسنجید، یکسان به نظر خواهند رسید.

تصاویر bit-map در صورت کوچک یا بزرگ شدن، تکه تکه خواهند شد. فونت هایی با اشکال Nector، Scalable fonts، outline fonts یا vector fonts نامیده می شوند.

فونت bit-map را raster نامیده و تنها می توان آنها را برای یک طرح مشخص با size و وضوح تعیین شده طراحی کرد.

۳. آشنایی با مفهوم بعد یک تصویر

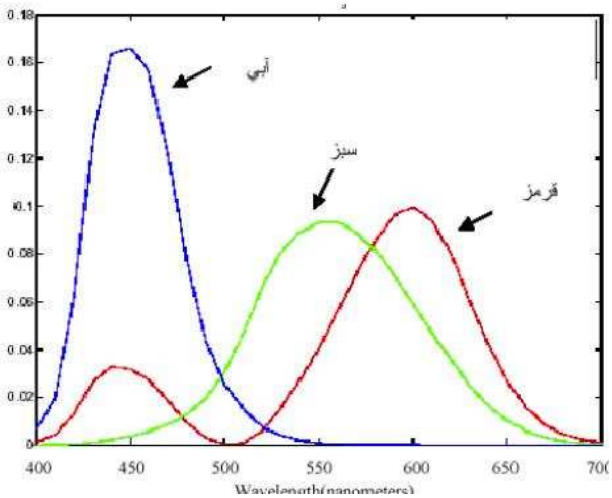
تصویرهای مبتنی بر نقشه بیتی، همواره به صورت شبکه های مربع شکل بزرگ می باشند. این شبکه مانند صفحهٔ شطرنج یا موزاییک های کف آشپزخانه می باشند. این شبکه های مربع شکل بزرگ از مربع های کوچکتری تشکیل شده اند. یکی از مشخصه هایی که همواره می

توان در مورد شبکه ها بیان نمود، این است، که دارای ابعاد می باشند. صفحهٔ شطرنج همواره ۸×۸ است، اما شبکهٔ پیکسل های تشکیل دهندهٔ صفحهٔ نمایش کامپیوتر، ۸۰۰×۶۰۰ است.

ابعاد شبکه در واقع معادل تعداد مربع هایی است، که طول و عرض تصویر را تشکیل داده اند و ربطی به اندازهٔ واقعی تصویر ندارند.

چگونگی تشکیل رنگ در چشم انسان

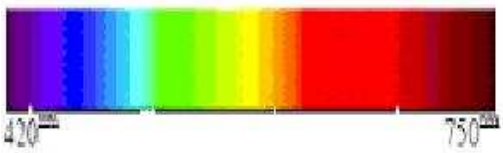
منحنی حساسیت چشم برای مشاهده کنندهٔ استاندارد در مقابل دریافت نور رنگی به صورت شکل زیر می باشد.



در واقع جهت مشاهده و درک هر رنگ، سه انرژی جداگانه مربوط به سه رنگ اصلی دریافت شده و از ترکیب آنان با یکدیگر رنگ تصویر ایجاد خواهد شد. لازم به ذکر است که هر سیستم گیرنده، منحنی حساسیت رنگی مختص به خود را داشته و لذا ترکیب سه انرژی رنگی $R(x,y)$ انرژی رنگ قرمز، $G(x,y)$ انرژی رنگ سبز، $B(x,y)$ انرژی رنگ آبی دریافتی می باشند.

بدین ترتیب جهت مشخص نمودن هر تصویر رنگی، باید از سه ماتریس جهت مقادیر قرمز (Red)، سبز(Green) و آبی (Blue) هر نقطه از تصویر (پیکسل) استفاده نمود. رنگ هر پیکسل توسط ترکیب سه رنگ اصلی در سه ماتریس داده شده، به دست می آید.

طیف رنگ ها را نسبت به طول موج می توان در شکل زیر مشاهده کرد.



پردازش تصویر رنگی

استفاده از رنگ در پردازش تصویر، دو انگیزهٔ اصلی دارد: دلیل اول تحلیل خودکار تصویر رنگ، توصیف گر توانایی است که در اغلب موارد شناسایی و استخراج شیء از صحنه را ساده می سازد.

دلیل دوم در مواردی که تحلیل تصویر به وسیله انسان انجام می شود، چشم انسان قادر است هزاران سایه و شدت رنگ را، در مقایسه با حدود ۲۴ سایه خاکستری دهد.

پردازش تصویر رنگی در حوزهٔ اصلی به دو دسته تقسمی می شود: پردازش تمام رنگ و پردازش شبه رنگی.

در گروه اول، تصاویر موردنظر معمولاً با یک Sensor تمام رنگی نظیر دوربین تلویزیون رنگی با پیمایش گر رنگی (color scanner) برداشته می شوند.

درگروه دوم، به هر شدت تک رنگ خاص یا محدوده ای از شدت ها یک سایه رنگی منتسب می شود تقریباً تا همین اواخر، بیشتر پردازش تصویر رنگی به صورت شبه رنگی انجام می شد.

پیشرفت قابل توجهی که در دهه ۱۹۸۰ انجام شد، باعث گردید Sensor های رنگی و سخت افزار لازم برای پردازش تصویر رنگی با قیمت های قابل قبولی در دسترس قرار گیرند. در نتیجهٔ این پیشرفت ها استفاده از روش های پردازش های تصویر تمام رنگی در محدوده وسیعی از کاربردها در حال افزایش است.

آشنایی با انواع مدل های رنگ

هدف از انتخاب مدل رنگ، تسهیل مشخص سازی رنگ ها در یک استاندارد است، که معمولاً روش مورد قبولی می باشد. در اصل مدل رنگ، تعیین یک سامانهٔ مختصات سه بعدی و زیر فضایی، درون آن سامانه است که در آن سامانه هر رنگ تنها با یک نقطه بیان می شود.

بیشتر مدل های رنگی که اکنون استفاده می شوند، به سمت سخت افزار (مانند نمایشگرها و چاپگرهای رنگ) یا کاربردهایی گرایش دارند، که هدف آنها کار با رنگ می باشد، نظیر تولید گرافیک های رنگی برای (Animation).

عمومی ترین مدل های سخت افزار گرا عبارت اند از:

مدل RGB (آبی، قرمز، سبز) برای نمایشگرهای رنگی و یک گروه وسیع از دوربین های وسیع، مدل CMY (آبی فیروزه ای، بنفش، زرد) برای چاپگرهای رنگی و مدل YIO، یک استاندارد پخش تلویزیون رنگی است.

در مدل سوم، Y متناظر با لومیناتوس است، I و Q دو مؤلفهٔ رنگ هستند، که به ترتیب هم فاز (Inphase) و متعامد (Quadrature) خوانده می شود. مدل HIS (اصل رنگ، اشباع، مقدار) از مدل هایی هستند که به طور مکرر برای کار با تصویر رنگی استفاده می شوند.

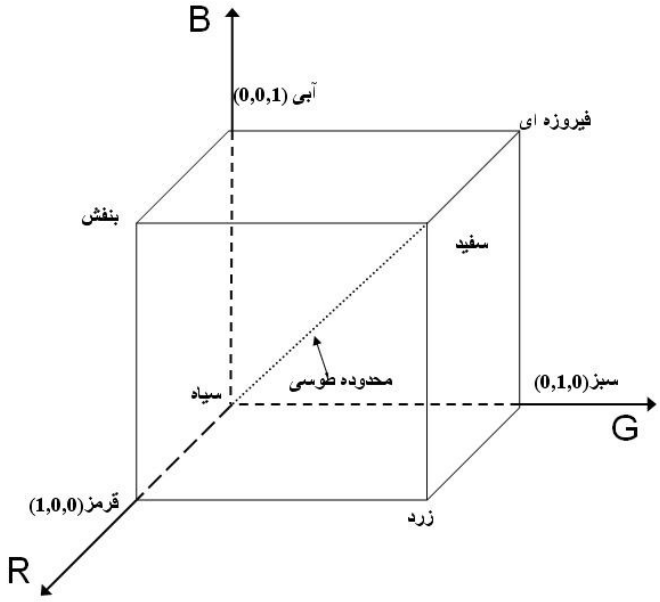
مدل های رنگی RGB، YIQ و HIS متداول ترین مدل هایی هستند، که برای پردازش تصویر استفاده می شوند. در بخش های بعد ویژگی های اصلی این سه مل معرفی خواهند شد.

گرچه مدل CMY به جای استفاده در پردازش تصویر واقعِی، در چاپ استفاده می شود اما به دلیل اهمیت آن در به دست آوردن خروجی های چاپی، در این جا نیز بررسی می شود.

مدل رنگ RGB

این مدل براساس سامانهٔ مختصات کارتزین است. زیر فضای رنگی مورد علاقه، مکعب تصویر زیر می باشد، که در آن مقادیر RGB در سه گوشه، آبی فیروزه ای، بنفش و زرد در سه گوشه دیگر، سیاه در مبدأ و سفید در دورترین گوشه از مبدأ قرار دارد.

در این مدل، محدودهٔ خاکستری از سیاه تا سفید در طول خط و اصل این دو نقطه قرار دارد و سایر رنگ ها نیز نقاطی روی یا درون مکعب هستند، که با بردارهایی که از مبدأ می گذرد تعریف می شود. برای تسهیل مدل، فرض بر این است که تمام مقادیر رنگ تراز شده اند طوری که مکعب تصویر زیر مکعب واحد باشد، یعنی فرض می شود که تمام مقادیرR، G و B در محدود [1,0] قرار دارند.



هر تصویر در مدل رنگ RGB سه صفحه مستقل دارد، که هر صفحه برای رنگ اولیه می باشد. وقتی این سه صفحه به نمایشگر RGB داده شوند، روی صفحهٔ فسفری ترکیب می شوند تا یک تصویر رنگی از مدل RGB جهت پردازش تصویرمعنا می یابد. همچنین اغلب دوربین های رنگی که برای تصویربرداری رقمی به کار می روند، از این قالب استفاده می کنند که این موضوع خود به تنهایی مدل RGB را مدل مهمی در مبحث پردازش تصویر می سازد.

یکی از بهترین مثال های کاربرد مدل RGB، پردازش داده های تصویری چند طیفی هوایی یا ماهواره ای است.

تصاویر به وسیله حس گرهای تصویربرداری که در محدودهٔ طیفی مختلف کار می کنند، گرفته می شوند، برای نمونه، هر قاب خروجی تصویربردار LANDSAT دارای ۴ تصویر رقمی است. (LANDSAT مخفف عبارت Land Sattellite است). (این نام را NASA به ماهواره هایی داده که جهت نظارت سطح زمین ساخته می شوند). همهٔ تصویرها از یک صحنه هستند، که در محدودهٔ پنجرهٔطیفی متفاوتی گرفته می شوند.

دو پنجره از چهار پنجرهٔ فوق، در بخش مرئی طیف هستند، که تقریباً متناظر با سبز و قرمز می باشند. دو پنجره دیگر در بخش مادون قرمز طیف هستند. بنابراین هر صفحه تصویر (Image Plane) معنای فیزیکی دارد و ترکیبات رنگی که با استفاده از مدل RGB برای پردازش و نمایش به دست می آیند، معمولاً وقتی معنا پیدا می کنند، که روی یک صفحهٔ رنگی دیده شوند، یا هنگام بخش بندی تصویر رنگی براساس مؤلفه های طیفی آن معنا دارند.

فرض کنید، که مسئله، ارتقای تصویر رنگی صورت انسان که بخشی از آن در سایه مخفی است باشد. تعدیل بافت نگار ابزار ایده آلی برای حل این نوع مسائل می باشد. به دلیل وجود سه تصویر و به دلیل آن که تعدیل بافت نگار با مقادیر شدت کار می کند، روال این است که هر یک از صفحه تصویرها به طور مستقل تحت تعدیل بافت نگار قرار گیرد. به احتمال زیاد، بخشی که در سایه مخفی است، ارتقا می یابد. در هر حال شدت ها در سه صفحه تصویر به طور متفاوتی تغییر داده می شوند، که این عمل باعث تغییر شدت های نسبی آنها می شود. نتیجه آن که، خواص رنگی مهم در تصویر، نظیر تنهای نرم (Flesh tone) روی نمایشگر RGB به طور طبیعی ظاهر نمی شوند.

مدل رنگ CMY

فیروزه ای، بنفش و زرد رنگ های ثانویه نوری یا رنگ های اولیه مادی هستند مثال وقتی بر سطح پوشیده از ماده رنگی فیروزه ای، نور سفید تابیده می شود، هیچ نور قرمزی از آن منعکس نمی شود، یعنی فیروزه ای نور قرمز را از نور سفید تفریق می کند.

بیشتر وسایلی که مواد رنگی را بر روی کاغذ می نشانند، نظیر چاپگرهای و کپی بردارهای رنگی، به داده های CMY نیاز دارند، یا این که در داخل خود، داده های RGB را به CMY تبدیل می کنند. این تبدیل با استفاده از عمل ساده ؟؟؟ انجام می شود.

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

دوباره فرض بر این است که تمام مقادیر رنگی به محدوده [0,1] تراز شده اند.

مدل رنگ YIQ

مدل YIQ در پخش عمومی تلویزیون رنگی تجارتی استفاده می شود. در واقع YIQ تغییر تصویر RGB به منظور افزایش بازده انتقال و حفظ سازگاری با استانداردهای تلویزیون تک رنگ می باشد. مؤلفهٔ Y در سامانهٔ YIQ، تمام اطلاعات ویدیوئی مورد نیاز تلویزیون تک رنگ می باشد. تبدیل RGB به YIQ به صورت رابطه زیر تعریف می شود.

جهت به دست آوردن مقادری RGB، از مقادیر YIQ به عرض باند (یا در حالت دیجیتالی) بیشتری برای نمایش Y و عرض باند (یا بیت های) کمتری برای نمایش I و Q نیازدارد.

مزیت اصلی مدل رنگ YIQ در پردازش تصویر ناشی از این خاصیت است که اطلاعات لومیناتوس (Y) و اطلاعات رنگی I و Q ناهمبسته (Decoupled) هستند.

مدل رنگی HIS

اصل رنگ است، رنگ خالص را توصیف می کند (زرد، نارنجی یا قرمز خالص)، در حالی که اشباع میزانی که یک رنگ خالص با نور سفید ترقیق شده است، را می دهد به سودمندی مدل HIA مدیون دو عامل اصلی است.

۱. مؤلفهٔ شدت I، از اطلاعات رنگ تصویر مجزا است.

۲. مؤلفه های اصل رنگ و اشباع، رابطهٔ نزدیکی با روش دریافت رنگ توسط انسان دارند.

این ویژگی ها، مدل HIS را به ابزار ایده آلی برای تولید الگوریتم های پردازش تصویر که مبتنی بر بعضی خواص احساس رنگ سامانهٔ بینایی انسان هستند، تبدیل می کند.

۲.۲.پیش پردازش

فراخوانی تصاویر توسط نرم افزار Matlab

با توجه به این مسئله که موضوع اصلی بحث ما در این تحقیق، پردازش تصویر می باشد، از ابتدایی ترین و مقدماتی ترین عملیاتی که در زمینهٔ تصویر باید بدان اشاره نمود، مبحث فراخوانی کردن و نمایش یک تصویر می باشد.

به طور کلی تمامی تصاویر در نرم افزار Matlab به صورت یک ماتریس شناخته می شوند، که برای فراخوانی کردن آن از تابعی به نام imread استفاده می شود.

می توان مقدار بازگشتی این دستور را که ماتریس یک تصویر می باشد، به یک متغیر نسبت داد و در فراخوانی های بعدی این تصویر، کافیست نام متغیر را صدا کرده و تصویر مربوطه را مشاهده نمود.

فرم کلی استفاده از دستور imread به صورت زیر می باشد:

نام متغیر = imread ('مسیر تصویر \ نام تصویر. پسوند تصویر');

برا ی نمونه برای فراخوانی تصویری به نام 020.jpg که در درایو C کامپیوتر و در folder به نام picture موجود می باشد به صورت زیر عمل می شود:

```
p=imread('c:\picture\020.jpg');
```

مثال

هدف فراخوانی ماتریس تصویر flower موجود در درایو e و در پوشه aaa با پسوند jpg می باشد. برای این کار به صورت زیر عمل می شود:

```
I=imread('e:\aaa\fower.jpg');
```

در زیر قسمتی از ماتریس مربوط به تصویر فراخوانی شده مشاهده می گردد:

95	93	54	52	57	69	78	79	80	82	70	72	74
88	86	51	48	53	67	76	76	76	79	70	72	74
86	85	50	48	56	72	80	78	78	82	70	72	74
82	80	56	53	61	77	85	82	82	86	75	76	76
77	74	65	62	70	84	92	89	88	92	81	81	80
74	70	75	72	78	92	99	95	94	97	86	86	85
73	71	82	79	85	98	104	100	97	100	90	89	88
72	71	87	83	89	01	106	101	98	101	92	91	90
70	71	88	85	89	01	106	100	96	99	94	93	91
68	69	89	85	89	01	105	99	95	98	95	94	91
80	65	98	95	94	98	99	95	91	92	90	90	86
73	66	91	88	89	95	98	96	94	95	91	89	83
66	67	82	80	83	91	97	97	97	99	90	83	74
64	68	77	76	80	89	96	97	98	101	92	83	72
65	68	77	75	79	87	93	94	95	97	96	86	75
66	67	76	74	77	85	90	89	89	91	91	82	74
63	65	73	71	73	81	85	84	84	86	85	78	73

نکاتی در مورد فراخوانی تصاویر در Matlab

الف) با قرار دادن سیمی کالن (;) د رانتهای دستورات نوشته شده، جواب نهایی به کاربر داده نمی شود.

برای نمونه اگر در انتهای توابعی نظیر imread سیم کالن (;) استفاده شود، ماتریس تصویر فراخوانده شده نمایش داده نمی شود، اما پردازش های صورت گرفته شده در حافظه Matlab ثبت می شود.

ب) در هنگام نوشتن دستورات و نام متغیرها به کوچک و بزرگ بودن حروف توجه شود.

مثال

کلمه NAME، name، NaMe، name با هم فرق می کنند.

ج) اسامی متغیر ها حتماً باید یک با یک حرف شروع شوند و بعد از آن می توان از حروف، اعداد و یا کاراکتر زیرخط (-) استفاده نمود.

مثال

نام Hw-23 صحیح است. نام 2h اشتباه است.

توجه شود که نمی توان از علائم نقطه گذاری در نام گذاری متغیرها استفاده کرد، زیرا اکثر آنها در Matlab دارای معنای خاصی هستند .

د) اسامی حداکثر ۳۱ کاراکتری اند. کارکترهای ۳۲ به بعد در نظر گرفته نمی شوند.

و) برای دسترسی به مرجع کامل پسوند تصاویر پشتیبانی شده، در محیط Command Windows، دستور زیر را تایپ کنید:

Help imread OR help (‘imread’)

در زیر تعدادی از پسوندهای پشتیبانی شده مربوط به تصاویر در نرم افزار Matlab آورده شده است:

Jpg(jpeg)_Tif(Tiff)_Bmp_Png

هـ) استفده ازکلمات کلیدی برای نام گاری، غیر مجاز می باشد. در Matlab این اسامی عبارتند از:

For,end,if,while,function,return,elseif,case,otherwise,swich,continue,

Else,try,cath,global,persistent,break

گرچه مبا توجه به نکته ب در سطور بالا می توان از کلمات کلیدی استفاده نمود، اما باید به حروف بزرگ نوشته شوند.

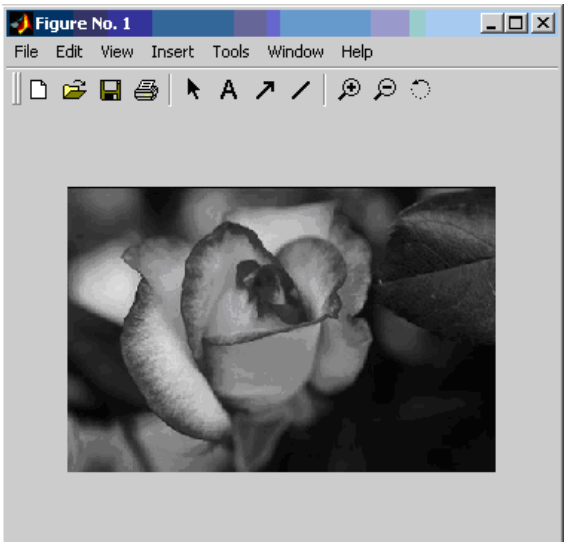
نمایش تصاویر در نرم افزار Matlab

پس از مشاهدهٔ ماتریس مربوط به یک تصویر حال، جهت مشاهده تصویر فراخوانی شده از تابعی با نام imshow استفاده می شود، که فرم کلی استفاده از این دستور به صورت زیر می باشد:

توجه شود در فرم کلی بالا نام متغیر در واقع همان نامی است، که تصویر فراخوانی شده توسط تابع imread به آن نسبت داده شده است.

مثال نمایش تصویر فراخوانی شده در مثال قبل:

```
i=imread('e:\aaa\fower.jpg');
imshow (i)
```



تصویر۱-۲-۲

همان گونه که در این مثال مشاهده می گردد، با قراردادن علامت (;) در انتهای دستور حاوی تابع imread، پردازش صورت گرفته روی تصویر که در واقع ماتریس شامل آن تصویر می باشد، نمایش داده نمی شود.

تعیین وضعیت یک تصویر در حافظه

در بسیاری از موارد ضروری است قبل از انجام هرگونه پردازش بر روی تصویر مروبوطه، در ابتدا اطلاعات سریع و اجمالی دربارهٔ مشخصات مربوط به تصویر به دست آوریم. نرم افراز Matlab این امکان را به ما می دهد که جهت انجام اینکار از تابع whos استفاده شود. این تابع مشخصاتی به صورت زیر را به ما ارائه می دهد:

نام تصویر	سایز ماتریس	میزان فضای اشغالی	نوع کلاس
image	600 * 800	132000	Uint 16

نکته: از این قسمت به بعد با فرض، پیش فرض بودن تصاویر در Matlab، تصاویر فراخوانی شده است.

مثال: دریافت اطلاعاتی از مشخصات کلی تصویر به نام Flower

```
i=imread('flower.jpg') ;
imshow (i)
whos
```

Name	size	Byte Class
I	600*800	1440000 uint8 array

Grand total is 1440000 elements using 1440000 byte

در انجام برخی از پردازش های پیشرفته، اطلاعات حاصل از تابع whos کافی نمی باشد، که جهت رفع این مشکل در مباحث بعد، توابع دیگری جهت دریافت اطلاعات با جزییات بیشتر از تصویر، بیان خواهد شد.

نمودار Histogram بررسی شدت نور یک تصویر

یکی از مسائل بسیار مهم و تأثیر گذار د رعلم پردازش، تصویر شدت نور تابیده شده به یک تصویر می باشد.

نرم افزار Matlab این امکان را به برنامه نویس می دهد که به کمک تابعی به نام imhist بتواند نمودار هیستوگرام یک تصویر را مشاهده و بررسی نماید.

نحوۀ استفاده از تابع ذکر شده، به صورت زیر می باشد:

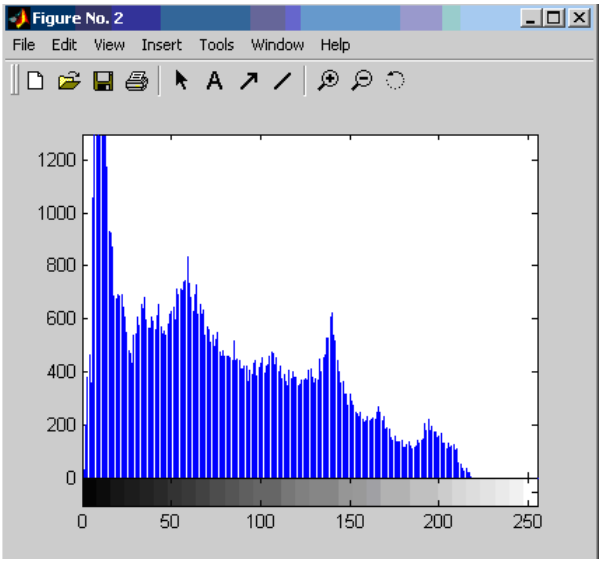
imhist(نام متغیر)

از دیگر کاربردهای این تابع نکته است، که توسط این تابع می توان شدت نور مربوط به دو تصویر را با یکدیگر مقایسه نمود، که از این روش جهت تعیین میزان تفاوت و تشابه دو تصویر استفاده می گردد.

مثال

جهت بررسی نمودار شدت نور مربوط به تصویر فراخوانی شده در مثال قبل که به متغیر i نسبت داده شده است، از دستور زیر استفاده می شود:

```
imhist(i)
```



نکاتی در مورد نمودار هیستوگرام شدت نور تصاویر

۱. محدودهٔ افقی نمودار بر مبنای شدت رنگ (Contrast) و محدودهٔ عمودی آن بیانگر شدت نور (Intensity) می باشد.

۲. در اکثر تصاویر، معمولاً برای محدودهٔ افقی نمودار، عدد 250 ثابت است.

۳. نمودار شدت نور تصاویر تنها برای تصاویر دو بعدی قابل استفاده می باشد و در تصاویر سه بعدی تابع imhist کاربردی نخواهد داشت. توضیحات بیشتر دربارهٔ انواع تصاویر و ابعاد مربوط به آنها در فصل های بعد بیان شده است.

تغییر پسوند مربوط به یک تصویر

برخی از پردازش ها و همچنین برخی از توابع و دستورات در نرم افزار Matlab بر روی فرمت خاصی از تصاویر اعمال نمی شوند. می توان با تغییر آنها به فرمت های دیگر (با تغییر پسوند آنها) پردازش دلخواه را انجام داد.

جهت انجام این کار از تابعی به نام imwrite استفاده می شود و فرم کلی استفاده از این تابع، بدین صورت می باشد:

;

imwrite(آرایه خروجی، 'نام تصویر جدید.سوند')

مثال

جهت تغییر پسوند تصویر flower.jpg به صورت زیر عمل می کنیم:

```
i=imread('flower.jpg') ;
imwrite(I,'flower.png');
```

از این مرحله به بعد، پسوند تصویر مربوطه، از jpg به فرمت png تغییریافته سات. جهت اطمینان از درستی تغییر انجام شده از تابعی به نام imfinfo استفاده می شود، که در مباحث بعد به طور کامل توضیح داده خواهد شد.

دریافت اطلاعات کامل از مشخصات یک تصویر

برای دریافت مشخصات کامل مربوط به تصاویر در نرم افزار Matlab، از تابع imfinfo استفاده می شود.

خروجی تابع مذکور، شامل اطلاعاتی به صورت زیر می باشد:

Name of the file (نام فایل)

File format (فرمت تصویر)

Version number of the file format (شماره ورژن فایل)

File modification data (زمان تغییر فایل)

File size in byte (سایز فایل)

Image width in pixel (مقدار پیکسل در عرض یک تصویر)

Number of bit per pixel (عدد پیکسل در بیت)

اطلاعات کسب شده توسط این تابع به مراتب کامل تر از اطلاعات حاصل از تابع whos می باشد، که در مباحث قبل بیان شد. همین امر باعث شده است که استفاده از این تابع کاربرد بیشتری نسبت به تابع whos داشته باشد.

فرم کلی استفاده از تابع imfinfo به صورت زیر می باشد:

imfinfo(‘مسیر فایل’)

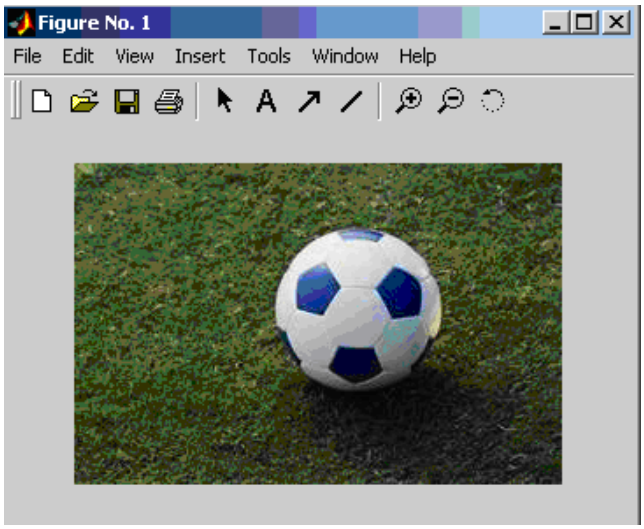
مثال

در این مثال هدف بررسی مشخصات کامل و دقیق تصویری به نام ball می باشد:

```
i=imread('ball.jpg') ;
imshow(i
```

```
imfinfo('ball.jpg')
```

```
ans=
      File name: 'g:\ax\ball.jpg'
    FileModData: 'll-Feb-1999 08:34:50'
       Filesize: 11469
         Format: 'jpg'
  FormatVersion: ''
          width: 278
         Height: 183
       BitDtpth: 24
     Color Type: 'truecolor'
FormatSignature: ''
        Comment: {}
```



اتصالات پیکسل ها در تصاویر

اتصال بین پیکسل ها مفهوم مهمی است که در تعیین مرز اشیاء یا اجزای نواحی تصاویر استفاده می شود. جهت تعیین اتصال دو پیکسل، باید نوع همسایگی پیکسل ها مشخص گردد. برای نمونه، آیا همسایه های چهارگانه هستند و آیا سطوح خاکستری آنها معیار شباهت مشخص شده ای را برآورده می کنند؟

برای مثال در تصویر دودویی با مقادیر صفر و یک ممکن است دو پیکسل، همسایه چهارگانه باشند، اما متصل گفته نمی شوند، مگر اینکه دارای مقادیر یکسانی باشند.

اتصالات پیکسل ها، تعیین کنندهٔ اتصال هر پیکسل مجاور و همچنین تعیین نوع اتصالات هر یک از پیکسل ها می باشد.

به تنظیمات گروهی پیکسل ها و اتصالات آنهادر تصاویر Binary، Object یا Connected Component گفته می شود.

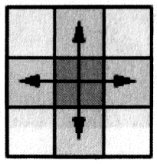
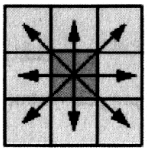
مثال

Foreground تصاویر Binary زیر به صورت Object های 1 تنظیم شده است اگر Foreground، 4 اتصال داشته باشد، تمام Object های Background به صورت صفر تنظیم می شود. همین طور اگر اتصال Foreground، 8 تایی باشد، پیکسل Background ، دارای حالت Loop وOutside Loop می شوند.

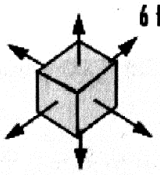
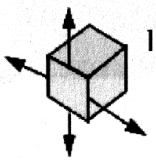
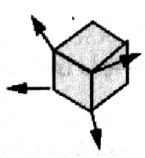
```
0  0  0  0  0  0  0  0
0  1  1  1  1  1  0  0
0  1  0  0  0  1  0  0
0  1  0  0  0  1  0  0
0  1  0  0  0  1  0  0
0  1  1  1  1  0  0  0
0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0
```

تعیین اتصالات تصویر

در زیر لیستی از اتصالات تصاویر دو بعدی آورده شده است:

اتصال ۴ تایی	
اتصال ۸ تایی	

لیست زیر شامل اتصالات تصاویر سه بعدی است:

اتصال ۶ تایی	
اتصال ۸ تایی	
اتصال ۲۶ تایی	

انتخاب اتصال

نوع همسایگی پیکسل ها و نحوهٔ مرزبندی آنها د رانتخاب نوع اتصالات تأثیر گذار است. برای نمونه، اگر تعداد همسایه های متصل به هم، چهارتایی باشد، تصویر دارای دو Object می باشد و اگر هشت تایی باشد، دارای یک object است.

در زیر اتصال North /South تعریف می شود، که تصویر به دو ستون شکسته شده است.

```
CONN=[010;010;010]
CONN=
0  1  0
0  1  0
0  1  0
```

آرایه ها باید نسبت به مرکز قرینه باشند.

آشنایی با تابع Imopen

یکی از توابع پرکاربرد و مفید در علم پردازش تصویر می باشد.

جهت آشنایی بیشتر با چگونگی کارکرد این تابع، به syntax های مربوط به این تابع توجه شود

```
IM2=imopen(IM,SE)

IM2=imopen(IM,NHOOD)
```

توضیحات مربوط به IM2=imopen(IM,SE)

کاربرد این فرمت جهت استفاده در تصاویر Bainery و Grayscate می باشد، که آن را IM با ساختار المنت SE می نامند.

قابل ذکر است، که آرگومان مربوط به این المنت به صورت تک بعدی می باشد.

توضیحات IM2=imopen(IM,NHOOD)

کاربرد این فرمت با ساختار المنت Strel (NHOOD) می باشد. NHOOD در واقع یک آرایه به صورت 1,0 است، که وظیفهٔ آن تعیین ساختار همسایگی می باشد.

کلاس های پشتیبانی کننده

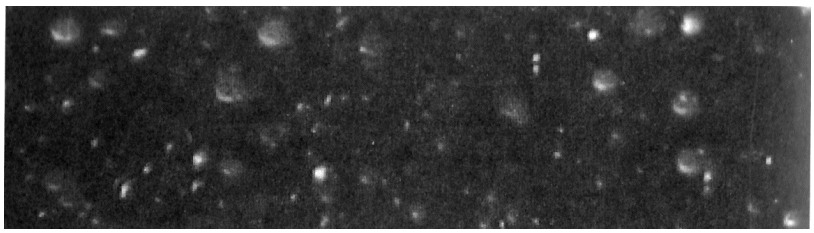
IM به صورت فرمت عددی، منطقی است و می تواند هر بعدی را پشتیبانی کند.

مثال

تصویری با نام snowflakes فرا خوانی شده است و در ادامه توسط تابع imopen و فیلتر کوچک کننده عملیات مربوطه روی آن انجام خواهد شد.

I=imread(‘snowflakes.png’);

Imshow(I)



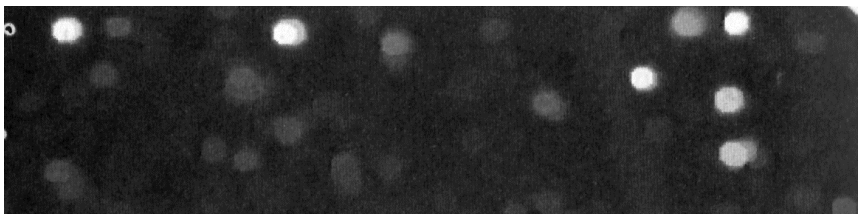
مرحلهٔ بعد، ساختن disk – shaped با شعاع 5 پیکسیل است. برای انجام این کار بدین صورت عمل می شود:

Se=strel(‘disk’,5);

در مرحلهٔ سوم، هدف، حذف پوسته های برقی کوچک تر از شعاع 5 پیکسل است.

I-opened=imopen(I,se);

Figure,imshow(I_opened,[])



در ادامه، از تابع Strel برای ساخت المنت ساختمانی (structure) استفاده خواهد شد. جهت آشنایی با نحوهٔ استفاده از این تابع، syntax های مربوط به این تابع بررسی می شود.

Se=strel(shape,parameters)

این syntax یک المنت ساختمانی flat به نام SE می سازد، که می تواند هریک از فرمت های زیر را داشته باشد.با توجه به فرمت های مختلف Strel، پارامترهای گوناگونی نیز تعریف شده است. جهت دریافت اطلاعات کافی در این زمینه، به جداول زیر و help نرم افزار Matlab مراجعه شود.

Flat structuring Elements	
‘arbitrary’	‘pair’
‘diamond’	‘periodicline’
‘disk’	‘rectangle’
‘line’	‘square’
‘octagon’	

Nonflat structuring Elementa	
‘arbitrary’	‘ball’

توضیحات مربوط به هر یک از پارامترهای موجود در جداول فوق به طور کامل در help نرم افزار Matlab آورده شده است که توصیه می شود جهت دریافت اطلاعات جامع تر در این زمینه به نرم افزار Matlab مراجعه شود.

1)SE=strel('arbitrary',NHOOD)

گزینهٔ NHOOD برای همسایگی پیکسل ها مورد استفاده واقع می شود. NHOOD در واقع ماتریسی شامل 0 و 1 می باشد که در آن مکان یک ها تعیین کنندهٔ همسایگی است.

مرکز NHOOD در واقع مرکز کل المنت می باشد، که توسط فرمول زیر محاسبه می شود.

Floor(size(NHOOD)+1)/2)

قابل ذکر است، که می توان مقداررشته ای "arbitrary" را حذف نمود و تنها از strel (NHOOD) استفاده کرد.

2)SE=strel('arbitrary',NHOODEIGHT)

SE=

1	0	0
1	0	0
1	0	1

Origin

این حالت یک ساختار غیر مسطح (Nonflat) می سازد HEIGHT ماتریس هم اندازه با NHOOD می باشد، که بالاترین مقدار آن به مقادیر غیرصفر در NHOOD وابسته است.

مقادیر ماتریس HEIGHT باید از نوع real و محدود باشد.

قابل ذکر است که این امکان وجود دارد، که مقدار رشته ای "arbitrary" را حذف نمود و تنها از strel (NHOOD,HEIGHT) استفاده شود.

3)SE=strel('ball',R,H,N)

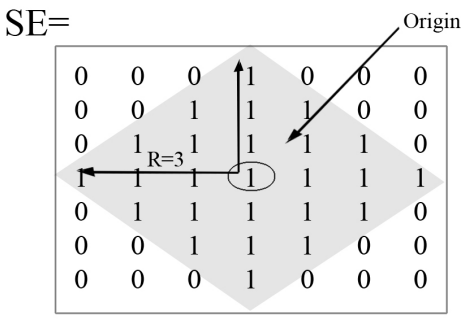
به کمک Ball-shaped یک المنت ساختمانی غیرمسطح ساخته می شود، که عملاً بیضی شکل می باشد. شعاع در صفحهٔ X-Y به صورت R می باشد و ارتفاع آن H نامیده می شود.

توجه شود مقادیر مربوط به شعاع و N باید اعدادی صحیح و غیر صفر باشند و مقدار H نیز عددی از نوع real است. اگر مقدار N عددی بزرگتر از 0 باشد، المنت ball- shaped، ترتیبی از N غیر مسطح و به صورت line- shaped خواهد بود. هنگامی که N مساوی با مقدار 0 باشد. هیچ محدودیتی به کار نرفته است و شامل تمام پیکسل های موجود در مرکز می باشد.

قابل توجه است که پیش فرض، N، عدد هشت می باشد.

4)SE=strel('diamond',R)

این حالت در واقع یک المنت مسطح . diamond –shaped می سازد که در آن R تعیین کنندهٔ فاصلهٔ مبدأ تا نقطهٔ لوزی (diamond) می باشد.(R به صورت عددی صحیح و غیر صفر می باشد).

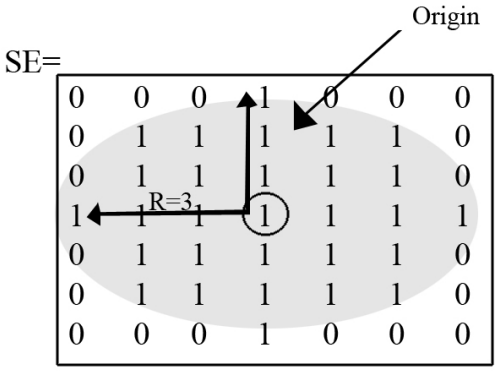


5)SE=strel(‘desk’,R,N)

در این فرمت، یک المنت مسطح disk-shaped ساخته می شود. شعاع (R) به صورت عدد صحیح و غیر صفر می باشد و N باید به صورت یکی از اعداد 6،4،0 یا 8 باشد.

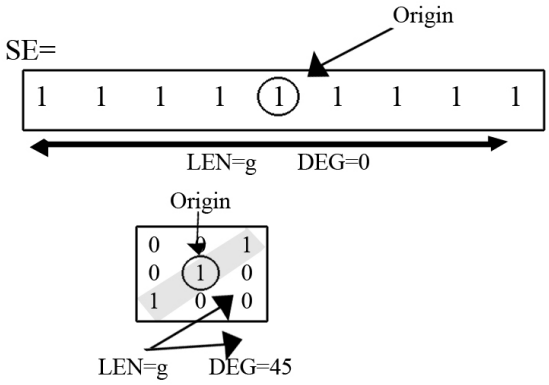
اگر مقدار N عددی بیشتر از صفر باشد، disk –shhped مشابه periodic – line عمل می کند، که در ادامه توضیح داده خواهد شد.

اگر N مساوی صفر باشد، هیچ محدودیتی به کار نرفته و شامل تمام پیکسل هایی موجود در مرکز و کوچک تر از شعاع می باشد. قابل توجه است که در این حالت پیش فرض N به صورت عدد 4 می باشد.



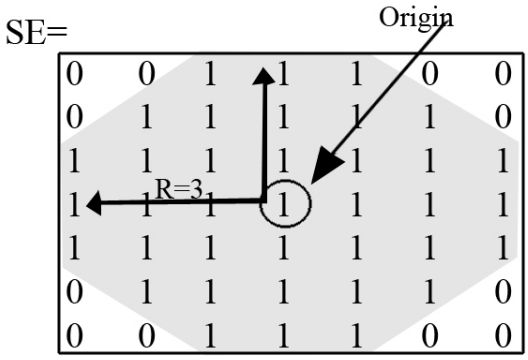
6)SE=strel(‘line’,LEN,DEG)

یک المنت خطی و مسطح می سازد، که در آن LEN تعیین کنندۀ طول و DEG تعیین کنندۀ زاویۀ خط برای اندازه گیری در جهت خلاف حرکت سرعت از بردار افقی می باشد. LEN در واقع مشابه فاصلۀ بین مرکز تا اعضای المنت ساختمان در جهت مخالف خط می باشد.



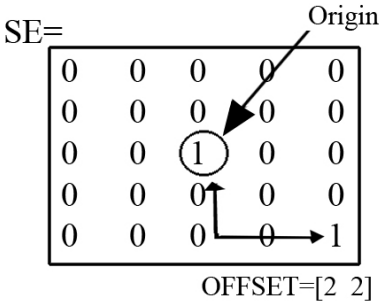
7)SE=strel(‘otagon’,R)

به کمک این فرمت، یک المنت octagonal و مسطح ساخته می شود. شعاع، تعیین کنندۀ فاصله از مبدأ تا کناره های octangle می باشد و برای اندازه گیری بردارهای عمودی و افقی مورد استفاده واقع می شود. شعاع باید مضربی از عدد 3 انتخاب شود.



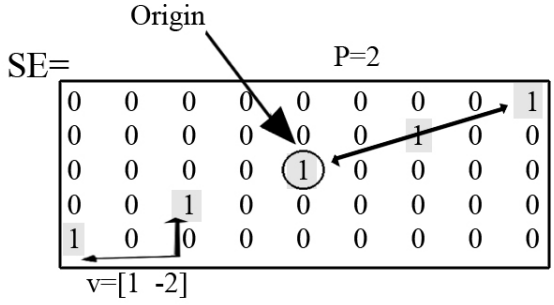
8)SE=strel(‘pair’,OFFSET)

به کمک این فرمت، یک المنت مسطح دو عضوی ساخته می شود، که یکی از اعضاء در مبدأ واقع شده است و دومین عضو معین شده برای بردار OFFSET می باشد. OFFSET یک بردار دو عضوی با مقدار عدد صحیح می باشد.



9)SE=strel(‘periodicline’,P,V)

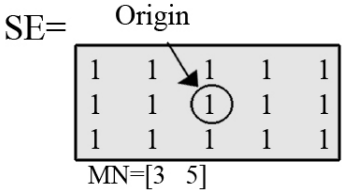
توسط این فرمت، المنتی مسطح ساخته می شود که دارای P+1×۲ تعداد اعضا می باشد. V به صورت یک بردار دو عضوی تعریف می شود، که ردیف های آن به صورت مقادیر عددی صحیح و ستون های آن OFFSET می باشند، یکی از این اعضا در مبدأ است و مکان دومین عضو ۱× V، V×1-، V×2، V×2-،...، P×V، P×V- می باشد.



10)SE=strel('rectangle',MN)

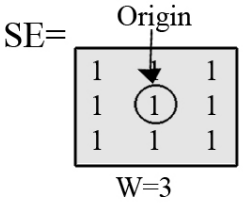
این حالت یک المنت rectangle – shaped و مسطح می سازد، که در آن MN مشخصهٔ اندازه است، دارای دو بردار غیر منفی و صحیح است.

اولین المنت MN شماره ای از ردیف همسایگی ها و دومین المنت، شماره ای از ستون آن همسایگی می باشد.



11)SE=strel('square',W)

این حالت یک المنت مربع شکل است، که ارتفاع آن با W تعیین می شود. مقدار W به صورت عددی غیرمنفی و صحیح می باشد.



توجه: برای تمام حالات و فرمت های مختلف، "arbitrary" به صورت استثناء می باشد. المنت ها با تکنیک های شناخته شده ای به نام structuring element decomposition ساخته شده اند.

لیست زیر بیانگر روش های پشتیبانی شده توسط strel می باشد:

Method	**Description**
Getheight	Get height of structuring element
Getneighbors	Get structuring element neighbor location and height
Getnhood	Get structuring element neighboehood
Getsequence	Extract sequence of decomposed structuring elemens
Isflat	Return true for flat structuring element
Reflect	Reflect structuring element
Translate	Translate structuring element

به نمونه های زیر در این زمینه توجه شود:

```
Se1=strel('square',11)     %11-by-11 square
Se2=strel('line',10,45)    %line,lengh10,angle45degrees
Se3=strel('disk',15)       %disk,radius15
Se4=strel('ball',15,5)     %ball,radius15,height5
```

انواع تصاویر در نرم افزار Matlab

به طور کلی تصاویر در نرم افزار Matlab، به چهار دستهٔ اصلی تقسیم می شوند:

1.Binary

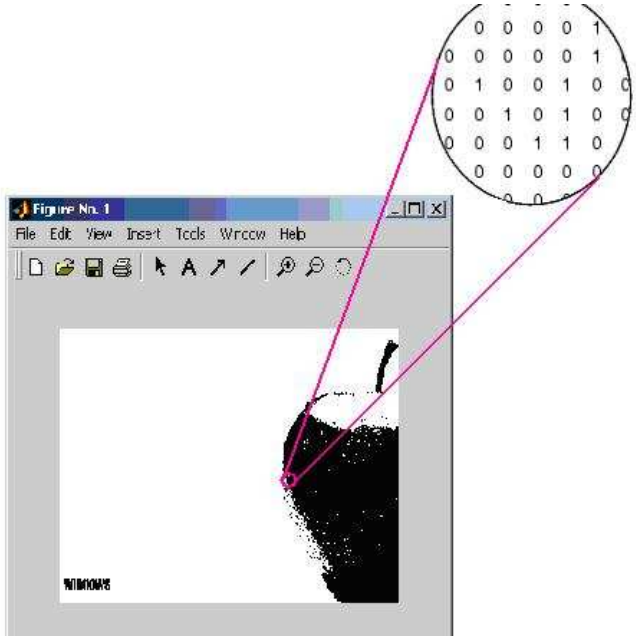
2.Intensity

3.Indexed

4.RGB

۱. تصاویر Binery

این دسته از تصاویر تنها دارای دو ارزش مجزا می باشند، که شامل یک آرایهٔ منطقی به صورت 0 یا عنوان off pixel و آرایه منطقی دیگری به صورت 1 با عنوان on pixel می باشد، که نحوهٔ نمایش این دسته از تصاویر به صورت زیر می باشد:



تصویر۶-۲-۲

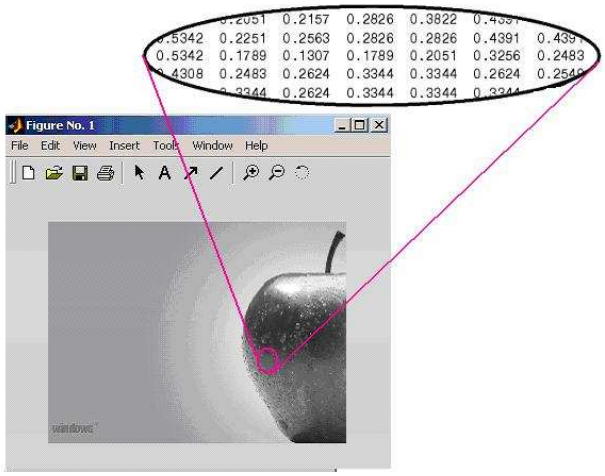
همان گونه که در تصویر۶-۲-۲مشخص است، تمام ارایه های مربوط به این ماتریس تنها به صورت 0، یا 1 می باشند، پس توجه شود که در عضوهای ماتریس فراخوانی شده مربوط به یک تصویر Bainery عددی به جز 0 و 1 وجود ندارد.

۲. تصاویر Intensity

کلاس های پشتیبانی شده توسط این نوع از تصاویر به صورت Double، Uint 8 و Uint 16 می باشد.

فرمت این دسته از تصاویر به صورت خاکستری است و آرایه های ماتریس بین صفر و یک می باشد. دراین روش، یک ماتریس m,n به تعداد پیکسل ها تشکیل می گردد، که عناصر این ماتریس اعداد یک جعبه رنگ تک بعدی است.

به عنوان مثال در حالت خاکستری (Gray) و کلاس Double، عدد صفر نشان دهندۀ رنگ سیاه و عدد یک نمایندۀ رنگ سفید و اعداد بین صفر و یک تا دقت سه رقم اعشار به شدت رنگ های خاکستری اشاره می کند.



تصویر۷-۲-۲

همان گونه که در ماتریس نشان داده شده مربوط به تصاویر Intensity مشاهده می گردد، تمام آرایه های مربوط به این ماتریس، شامل اعدای بین 0 و 1 می باشد و اعداد بزرگ تر از 1 و یا کوچک تر از 0، شامل محدودۀ این دسته از تصاویر نمی باشد.

۳. تصاویر RGB

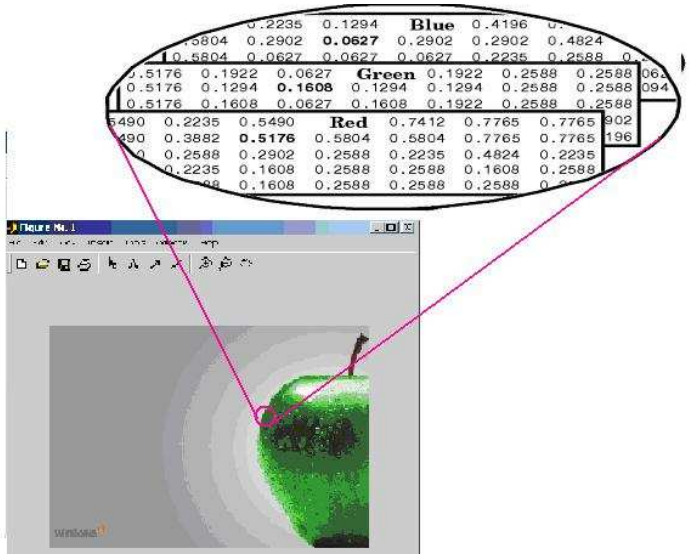
برای ذخیره نمودن اطلاعات مربوط به این نوع تصاویر، از یک ماتریس m×n×3 که مقدار هر یک از رنگ های قرمز (R)، سبز(G) و آبی (B) را برای هر پیکسل نمایش می دهد استفاده می شود. در این حالت نیز ماتریس می تواند کلاس هایی به صورت Uint8، Uint16 و Double داشته باشد.

به عنوان مثال اطلاعات مربوط به پیکسل (۲،۳) در آرایه های RGB (۲،۳،۱) برای رنگ قرمز (۲،۳،۲) برای رنگ سبز و (۳،۳،۲) برای رنگ آبی ذخیره می گردد.

پیکسل (0,0,0) نشان دهنده رنگ سیاه و پیکسل (1,1,1) مشخص کننده رنگ سفید می باشد و ترکیب سه رنگ برای هر پیکسل نشان دهنده آرایۀ سه بعدی است.

به عنوان نمونه ترکیب رنگی قرمز، سبز و آبی با پیکسل دارای مختصات (5,10) به ترتیب به صورت زیر نگهداری می شوند:

1)RGB(1,5,10) 2)RGB(2,5,10) 3)RGB(3,5,10)

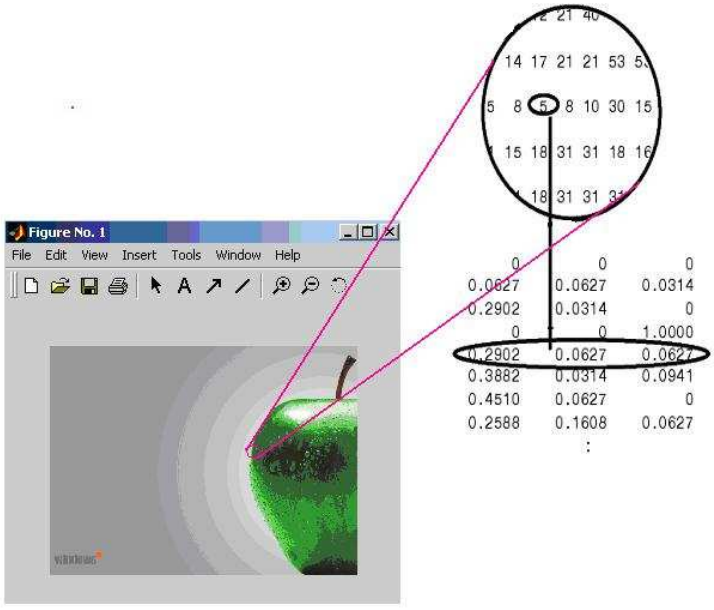


تصویر۸-۲-۲

۴. تصاویر Index

در این دسته از تصاویر، نرم افزار Matlab یک ماتریس m×n را که می تواند هر یک از کلاس های uint8، uint16،Double را دارا باشد، به عنوان ماتریس شاخص استفاده می کند و از یک ماتریس m×3 از کلاس Double برای جعبهٔ رنگ مربوطه استفاده می کند، که هر یک از عناصر ماتریس شاخص به یکی از سطرهای ماتریس جعبهٔ رنگ اشاره دارد.

قابل ذکر است، که هریک از سه ستون ماتریس جعبهٔ رنگ، در واقع تعیین کنندهٔ شدت رنگ های قرمز، سبز و آبی می باشد.



تصویر۹-۲-۲

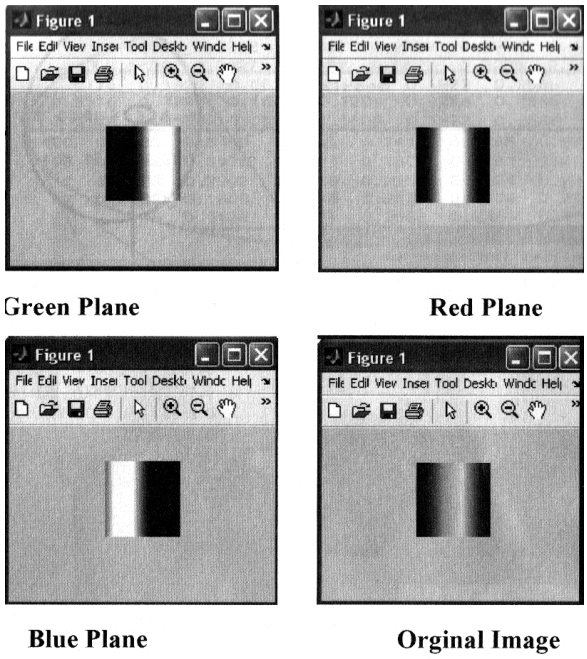
مثال

در زیر تصویر رنگی RGB ایجاد شده است، که در آن، هر سه رنگ قرمز، سبزو آبی با یکدیگر ترکیب هستند.

برای دیدن هر طیف رنگ، RGB را فراخوانی کرده و طیف های رنگی مربوط به این تصویر را در صفحهٔ جداگانه نشان می دهد.

```
RGB=reshape(ones(64,1)*reshape(jet(64),1,192),[64,64,3];
R=RGB(:,:,1);
G=RGB(:,:,2);
B=RGB(:,:,3);
Figure,imshow(R )
Figure,imshow(G)
Figure,imshow(B)
Figure,imshow(RGB)
```

همان گونه که مشاهده می شود، در plan های تصویر۱۰-۲-۲، محدوده های سفید نشان دهندهٔ طیف رنگ های قرمز، سبز و آبی می باشد و ارزش رنگی محدوده های سیاه معادل صفر است.



تصویر۱۰-۲-۲

خلاصه ای از انواع تصاویر و کلاس های مربوط به هر یک

Image Type	Storage Class	Interpetation
Binary	Logical	An array of zeros and ones
Indexed	Double Uint 8 OR Uint 16	An array of integers in range [1, p] An array of integers in range [0, P- 1]
Intensity	Double	An array of floating – point value the typical range of values is [0,1]
RGB	Uint 8 or Uint 16 Double Uint 8 or Uint 16	An array of integers the typical range of values is [0,255] or [0, 65535] An m- by–n-by–3 array of floating – point value in the range [0, 1] An m-by- n- by- 3 array of integers-point value int the range [0,255] or [0,65535]

تبدیل انواع تصاویر به یکدیگر

جهت کاربرد برخی از دستورات موجود در Toolbox Image Processing، در نرم افزار Matlab، گاهی لازم است که فرمت تصاویر را به فرمت دلخواه تغییر دهید. بدین دلیل که برخی از دستورات و توابع موجود در Matlab تنها بر روی نوع خاصی از تصاویر قابل اجرا می باشند

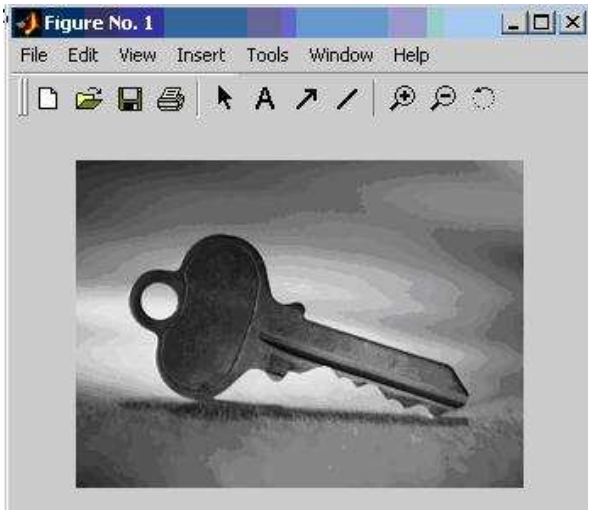
.برای نمونه، عملیات Thersholing تنها بر روی تصاویر به فرمت Graycale قابل اجرا می باشد.

جهت تبدیل انواع تصاویر به یکدیگر از توابع تبدیل موجود در جدول زیر استفاده می شود.

تابع	توضیحات
dither	تبدیل RGB به index و همچنین intensity به binary به روش Dither
gray 2 ind	تبدیل فرمت Intensity (Grayscale) به index
grayslice	تبدیل فرمت intensity (Grayscale) به index به کمک عملیات Thresholding
In2bw	تبدیل RGB, index, intensity به Binary
ind2gray	تبدیل index به intensity
Ind2rgb	تبدیل index به RGB
Mat2gray	تبدیل داده درون ماتریس به Intensity (Grayscale)
Rgb2gray	تبدیل RGB به Gray

درمثال صفحهٔ بعد، هدف تغییر فرمت تصویر رنگی key به فرمت Bainary می باشد.

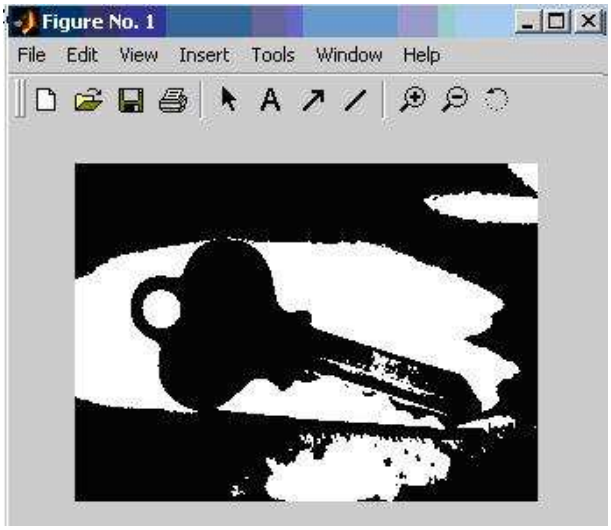
```
i=imread('key.jpg');
imview(i)
```



تصویر ۱۱-۲-۲

حالت جهت تبدیل به فرمت Bainery، از دستور زیر استفاده می شود.

```
p=im2bw(i);
imview(p)
```

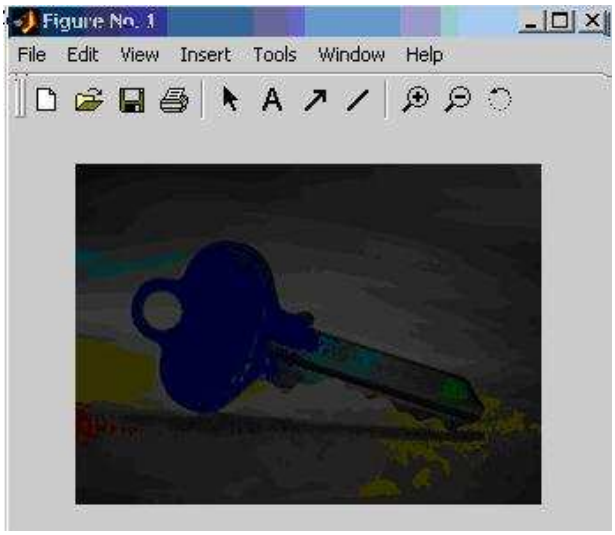


تصویر ۱۲-۲-۲

مثال

جهت تبدیل تصویر orginal مثال قبل به فرمت index، به صورت زیر عمل می شود:

```
y=gray2ind(i);
imview(y)
```

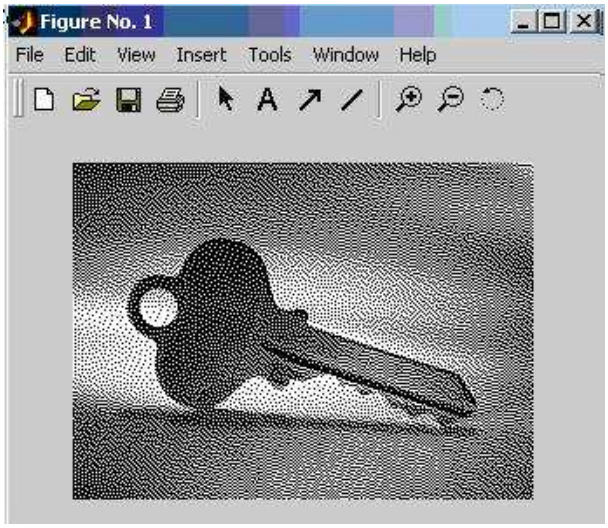


تصویر۱۳-۲-۲

مثال

برای آشنایی با تابع تبدیل dither تغییرات اعمال شده بر روی تصویر اولیهٔ مثال قبل، به کمک این تابع به صورت زیر مشاهده می گردد.

```
w=dither(i);
imshow(w)
```



تصویر۱۴-۲-۲

همان گونه که در سه مثال قبل مشاهده شد، در این مثال ها برای نمایش تصاویر از تابع جدیدی با عنوان imvew استفاده شده است.

نحوهٔ کارکرد این تابع، تا حدودی مشابه تابع imshow می باشد.

اطلاعات کامل تر در زمینهٔ این تابع در فصل های بعد بیان خواهد شد.

نمایش تصاویر چند فریمی

یک تصویر چند فریمی، در واقع یک فایل با بیش از یک تصویر است. فرمت های پشتیبانی کننده توسط تصاویر چند فریمی عبارتند از:

HDF and TIFF

روش های نمایش تصاویر چند فریمی

نمایش به صورت جداگانه به کمک تابع imshow

۱. نمایش به یکباره تمام فریم ها به کمک تابع montage

۲. تبدیل فریم ها به فیلم توسط تابع immovie

آرایه های تصاویر چند فریمی

ممکن است در برخی از کاربرها به مجموعه تصاویری، که از لحاظ زمانی به آنها تأخیر داده شده است، نیاز پیدا شود.

مثال

فریم های فیلم slices های MRI در toolbox image processing جهت نگهداری تصاویر چند فریمی در آرایه، امکاناتی را فراهم کرده است.

هر قسمت از تصویر را به طور جداگانه فریم می نامند. اگر هر آرایه چندین فریم را نگهداری کند، هر کدام از آنها به تنهایی دارای چهار بعد هستند.

برای نمونه احتمال دارد یک آرایه با 5 تصویر رنگی 400 در 300 به تصویر زیر در آید:

400 by 300 by 1 by 5

استفاده از دستور cat برای نگهداری تصاویر جدا در یک فایل چند فریمی

برای مثال اگر یک گروه از تصویر a2،a1 ..... a5 را داشته باشید، می توان همه آنها را به تنهایی درون یک آرایه نگهداری کرد.

$A=cat(4,a_1,a_2,a_3,...a_5);$

همچنین در نرم افزار Matlab می توان به سادگی فریم ها را از یک تصویر چند فریمی خارج نمود. برای نمونه در صورت داشتن چند تصویر چند فریمی، برای خارج کردن سومین فریم، به صورت زیر عمل می شود:

Frm3=mult(:,:,3)

شرایط پشتیبای چند فریمی ها

بسیاری از توابع image processing تنها برای تصاویر دو، سه و یا چهار بعدی کار می کنند، اما شما باید به طور جداگانه هر فریم را مورد پردازش قرار دهید.

برای نمونه دستور زیر، هفتمین فریم از آرایه ای به نام MULTI را نمایش می دهد:

```
Imshow(:,:,3)
```

نمایش فریم های تصویر چند فریمی به طور جداگانه

مثال

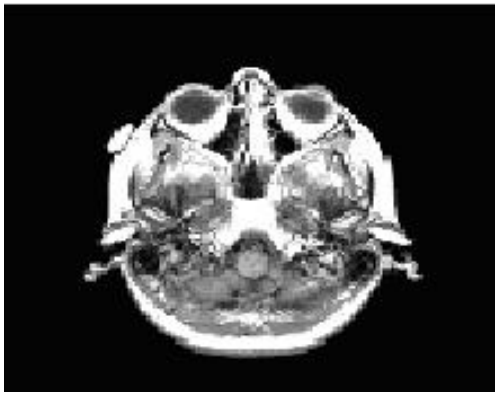
در این مثال، تصویر mri.tif فراخوانی شده و سومین فریم آن نمایش داده شده است.

تعریف آرایه ای برای نگهداری 27 فریم از تصویر mri.tif به صورت زیر می باشد.

```
mri=uint8(zeros(128,128,1,27) ;
for frame=1:27
[mri(:,:,:,frame), map]=imread('mri.tif',frame) ;
end
imshow((:,:,:,3), map) ;
```

تصاویر چند فریمی Intesnity، Index، Binary، دارای آرایهٔ چند بعدی به صورت زیر هستند.

M by N by I by J by k



تصویر۱۵-۲-۲

بیان چند نکته

۱. قابل ذکر است که k نماینده جمع تعداد فریم ها و عدد یک بدین معناست، که تصویر تنها دارای یک رنگ است.

۲. توجه شود، که هر دو دستور زیر معادل یکدیگر هستند:

```
Imshow(mri(:,:,1,3),map)

Imshow(mri(:,:,:,3),map)
```

۳. تصاویر چند فریمی RGB دارای آرایه (M by N by 3 by K) هستند، که در آن عدد 3 بدان معناست، که تصویر از پالت 6 رنگی استفاده می کند.

مثال

```
Imshow(rgb(:,:,:,7))
```

در بالا هفتمین فریم را نشان داده و معادل زیر است:

```
Imshow(rgb(:,:,3,7))
```

عدد 3، نمایندهٔ یکی از سه رنگ RGB است.

نمایش یکباره تمام فریم های یک تصویرچند فریمی

جهت به وقوع پیوستن این کار از تابعی بنام montage استفاده می شود، که این تابع فریم را به چند قسمت جداگانه تقسیم کرده و به طور جداگانه هر تصویر را نمایش می دهد. نکتهٔ جالب توجه مربوط به این دستور، شباهت Syntax های این تابع با تابع imshow می باشد.

```
Montage(x,map)
```

توجه شود که تمام فریم های یک چند فریمی در آرایهٔ تصویر index، باید از Colormap مشابه استفاده کنند.

تبدیل چند فریمی به فیلم

جهت انجام این کار از تابعی به نام immovie استفاده می شود، که در زیر با یک مثال، نحوهٔ استفاده از این تابع بیان شده است.

مثال

```
Mov=immovie(x,map)
```

X نام تصویر مربوطه است و قابل ذکر است، که تصویر X، آرایهٔ چهاربعدی است که تبدیل به فیلم شده است، معادل این دستور در محیط Matlab بدین صورت می باشد:

```
Movie(mov)
```

در مثال زیر تصویر چهار بعدی و چند فریمی mri.tif به فیلم تبدیل شده است.

مثال

```
mri=uint8(zeros(128,128,1,27) ;
for frame=1:27
[mri(:,:,:,frame),map]=imread('mri.tif',frame) ;
end
mov=immovie(mri,map) ;
movie(mov) ;
```

نمایش تصاویر متعدد

نرم افزارMatlab، قادر به نمایش بدن قید و شرط تعداد تصاویر نمی باشد، اما در figure های جداگانه این کار امکان پذیر است.

محدودیت اصلی در تعداد رنگ های نمایش داده شده توسط کامپیوتر می باشد.

سیستم مربوطه باید 8 و 16 یا 24 بیت درهر هر پیکسل داشته باشد، یعنی دارای عمق رنگ 8 و 16 یا 24 باشد.

در صورتی که از سیستمی با عمق رنگ 16 یا 24 استفاده شود، تا ماکزیمم 256 رنگ مختلف، نمایش داده می شود.

نحوۀ تعیین عمق پیکسل در سیستم

```
Get(0,'screen depth');
```

تمایش هر تصویر در figure های جداگانه

برای این کار از تابع imshow استفاده می شود، که درمباحث قبل به طور کامل بیان شده است.

```
Imshow(i)

Figure,imshow(i2)

Figure,imshow(i3)
```

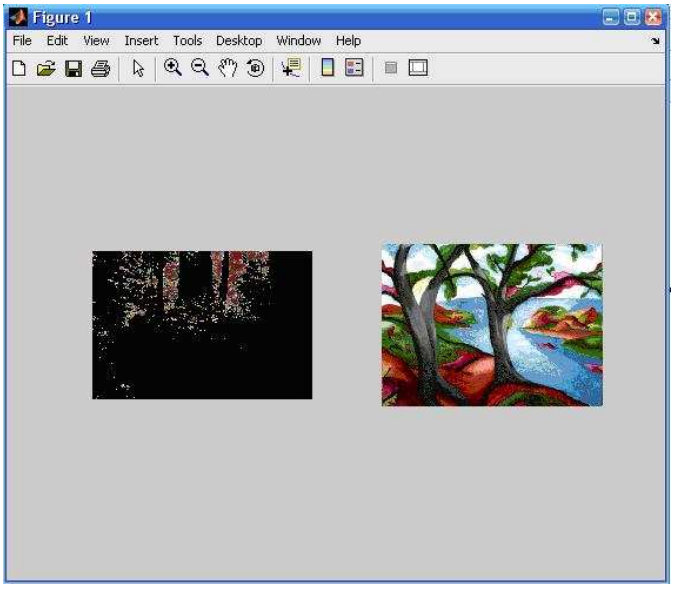
با انجام مراحل فوق، تصاویر i، i2 و i3 در پنجره های جداگانه نمایش داده می شوند.

مثال

```
[x1 ,map1]=imread('forest.tif');
[x2 ,map2]=imread('trees.tif');
imshow(x1,map1),figureimshow(x2,map2);
```

در این مثال دو تصویر از نوع Index نشان داده شده، که هشت بیتی اند و colormap و وضوح تصویر آنها با یکدیگر متفاوت است. همین امر باعث می شود که تصویر اول از colormap تصویر دوم استفاده کند.

همان گونه که در تصویر ۱۶-۲ به وضوح مشاهده می شود، این کار نتیجۀ چندان جالبی نمی دهد.



تصویر۱۶-۲-۲

از جمله ترفندهای جلوگیری از بروز مشکل فوق، استفاه از رنگ های کمتر است که به روش های متعددی انجام می پذیرد.

یکی از این روش ها استفاده از تابع imapprox و روش دوم تبدیل تصویر به فرمت RGB یا Truecolor می باشد.

وارد کردن چند تصویر در یک Figure

۱. توسط تابع imshow در ترکیب با subplot

۲. توسط تابع subimage در ترکیب با subplot

تابع subpot در واقع یک Figure را به چند ناحیه نمایشی تقسیم می کند. (به صورت یک ماتریس n در m)

مثال

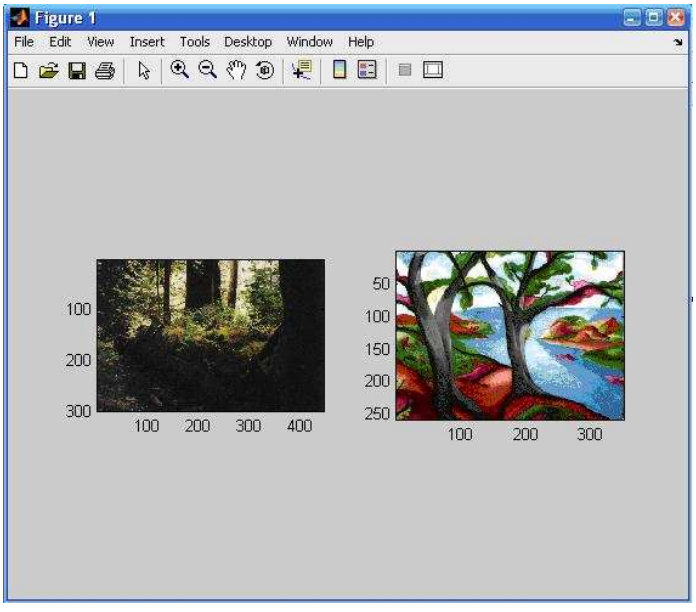
در زیر نحوهٔ نمایش دو تصویر در کنار یکدیگر، در یک پنجره آورده شده است:

```
[x1 map1]=imread('forest.tif');
[x2 map2]=imread('trees.tif');
Subplot(1,2,1),imshow(x1,map2)
Subplot(1,2,2),imshow(x2,map2)
```

در صورتی که colormap ها مشترک باشند، استفاده از تابع subpot نتیجهٔ غیرقابل قبولی رادر برخواهد داشت، که جهت رفع مشکل فوق، از تابعی به نام subimage استفاده خواهد شد.

تابع subimage در واقع تصاویر را به فرمت RGB تبدیل می کند. البته این کار قبل از نمایش تصویر صورت می پذیرد. آن گاه از به وجود آمدن مشکلات share (به اشتراک گذاشتن) colormap ها جلوگیری می شود.

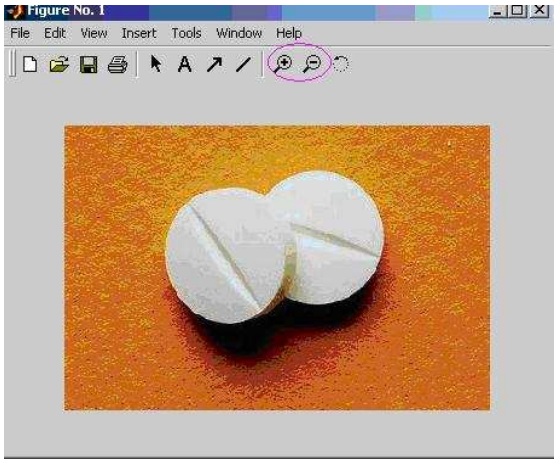
```
[x1map1]=imread('forest.tif') ;
[x2map2]=imread('trees.tif') ;
Subplot(1,2,1) ,subimage(x1,map1)
Subplot(1,2,2) , subimage(x2,map2)
```



تصویر۱۷-۲-۲

Zoom کردن تصاویر توسط نرم افزار Matlab

به طور کلی در محیط Matlab، ساده ترین روش جهت عملیات zoom بر روی تصاویر، استفاده از گزینهٔ zoom در بالای پنجره مربوط به هر تصویر می باشد.



تصویر۱۸-۲-۲

عملیات Zoom محدودهٔ بردارهای تصویر را تغییر می دهد و هیچ گونه تغییری در داده های تصویر و اطلاعات مربوط به آن ایجاد نمی کند. تابع Zoom در واقع تنها اندازهٔ یک تصویر را تغییر می دهد.

به دلیل سادگی بیش از حد این مبحث، از توضیحات اضافه خودداری می شود. در همین راستا نیز می توان از توابع زیر استفاده نمود.

Zoom on AND zoom out

همچنین تابع zoom off برای برگرداندن سایز تصویر به حالت اولیهٔ آن مورد استفاده قرار می گیرد.

عملیات Texture Mapping در نرم افزار Matlab

در گرافیک سه بعدی به فرایند افزودن جزئیات موردنظر به یک شیء گویند، که به کمک ایجاد تصویر یا یک الگو ایجاد می شود که می توان "به دور" شیء پیچید.

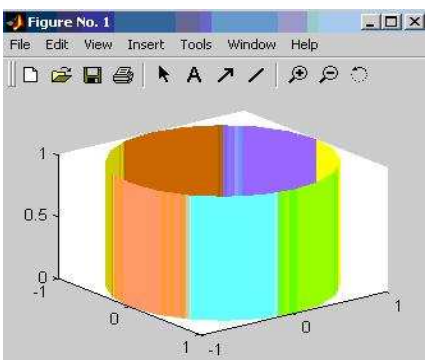
هنگام استفاده از تابع imshow نرم افزار Matlab تصاویر را به صورت دو بعدی نمایش می دهد .

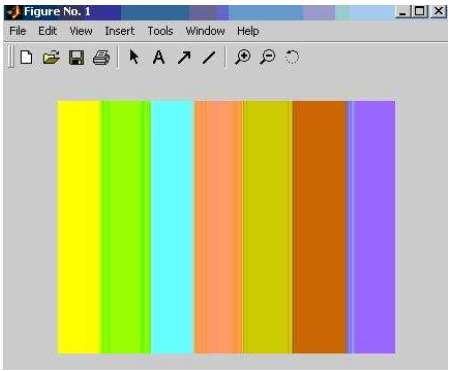
هرچند این کار همیشه ممکن نیست اما برای پارامترهای surface و یا سطحی با استفاده از plot یا sphere امکان پذیر است.

تابع wrap این نوع نمایش را توسط texture mapping ایجاد می کند. قابل ذکر است که عملیات texture mapping در واقع به عنوان یک پردازش برای نقشه ها مورد استفاده واقع می شود.

مثال

```
[x y z]=cylinder ;
i=imread('color.jpg ') ;
imshow(i)
wrap (x y z i) ;
```





معرفی تابع improfile و کاربرد این تابع در نرم افزار Matlab

این تابع جهت محاسبه ارزش پیکس ها (Pixel Value) به طور مقطعی و به کمک خط مورد استفاده قرار می گیرد.

در ابتدا به بررسی Syntax های مربوط به این تابع پرداخته می شود:

C=improfile

C= improfile(n)

C= improfile(I,xi,yi)

C= improfile(I,xi,yi,n)

[cx,cy,c]= improfile(…)

[cx,cy,c,xi,yi]= improfile(…)

[…]=improfile(x,y,I,xi,yi)

[…]=improfile(x,y,I,xi,yi,n)

[…]=improfile(…,method)

تابع مذکور مقدار شدت نور را به وسیلهٔ مسیر یک یا چندین خط در تصویر محاسبه می کند. این تابع، نقاطی با فاصله های مساوی در مسیری که توسط کاربر تعیین شده انتخاب کرده، سپس آنها را جهت یافتن ارزش شدت نور پیکسل ها در هر نقطه درج می کند.

توجه: تابع improfile تنها برای تصاویر RGB . GrayScale قابل استفاده می باشد.

اگر تابع فوق را توسط هر یک از Syntax های زیر فراخوانی کنید، به طور خودکار بردارهایی در تصویر ترسیم می گردند.

C= improfile

C= improfile(n)

مقدار n در واقع تعیین کنندهٔ تعداد نقاط محاسبه شده برای شدت نور می باشد و در صورت عدم استفاده از این آرگومان، improfile مقداری برای n انتخاب می کند، که تقریباً برابر با تعداد پیکسل هایی است، که در مسیر عبور می باشد.

تعیین مسیر خط درتصویر با کلیک موس روی نقاطی در تصویر انجام می شود و با فشردن کلیدهای Backspace یا Delete نقاطی که از قبل انتخاب شده اند، پاک می شوند.

قابل ذکر است که Shift-click، right-click، double – click، باعث افزودن خودکار آخرین انتخاب خواهند شد و فشردن Return عمل انتخاب را به اتمام می رساند. با اتمام عمل انتخاب، تابع improfile ، نتیجه را به متغیر c بر می گرداند.

اگر ورودی یک تصویر، GrayScale باشد، c به صورت یک بردار 1 در n خواهد بود و برای تصاویر RGB، c یک آرایه 1 در n در سه (n-by-1-by-3) می باشد.

از نتیجه را به کاربر نمایش می دهد. اگر مسیر مشخص شده در تصویر تک خط باشد؛

قابل توجه است که در صورتی که آرگومان خروجی حذف شود، improfile یک plot improfile یک plot دو بعدی می سازد و در صورتی که مسیر انتخابی دو یا بیشتر از دو خط باشد تابع improfile یک plot سه بعدی می سازد.

جهت اینکه تعیین تعداد پیکسل ها به طور خودکار انجام نگیرد و کاربر در تعیین تعداد پیکسل ها داشته باشد، syntax های زیر مورد استفاده واقع می شوند:

C=improfile(I,xi,yi)

C=improfile(I,xi,yi,n)

Xi و yi هر دو بردارهایی مساوی اند که فضایی با مختصات خاصی ایجاد می کنند ونقاط درون این دو بردار تعیین کننده مقدار پیکسل هایی است، که باید عملیات profile بر روی آنها انجام می گیرد.

جهت دریافت اطلاعات اطلاعات بیشتر از تصویر Syntax های زیر مورد استفاده قرار می گیرند:

[cx,cy,c]= improfile(…)

[cx,cy,c,xi,yi]= improfile(…)

Cx و cy بردارهایی به طول n می باشند، که فضایی با مختصات خاصی ایجاد می کنند و نقاط درون این دوبردار تعیین کننده مقدار پیکسل هایی است که باید profile شوند.

در صورتی که تمایل نداشته باشیم، فضایی که نقاط درون آنها برای profile استفاده می شود، به طور پیش فرض تعیین گردد، از Syntax های زیر استفاده می گردد.

[..]=improfile(x,y,I,xi,yi)

[…]=improfile(x,y,I,xi,yi,n)

X و y بردارهایی هستند که جهت تعیین مقادیر XData و YData مورد استفاده واقع می شوند.

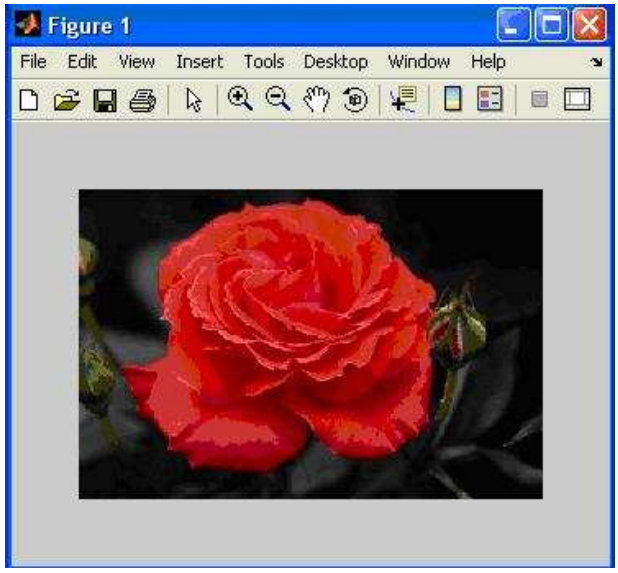
کلاس های پشتیبانی کننده توسط تابع improfile

تصویر ورودی می تواند به صورت یکی از فرمت های uint8، uint16، single، double، logical باشد.

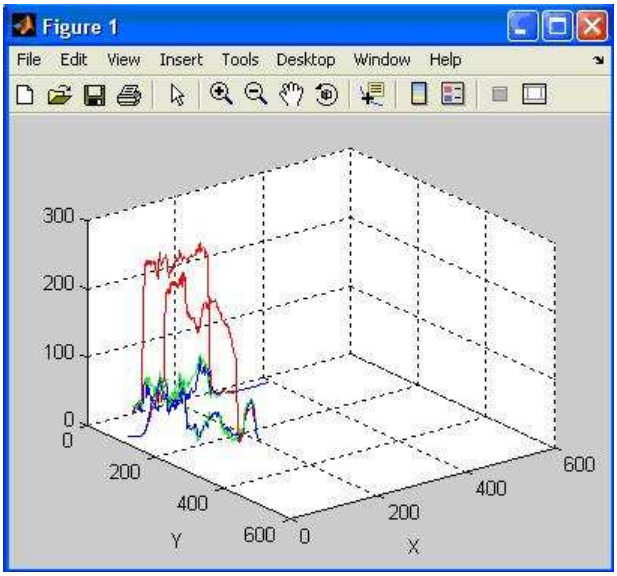
مثال

هدف از این مثال اعمال تابع improfile بر روی تصویر رنگی flower می باشد:

```
I = imread('flower. jpg');
imshow(I)
x = [19 427 416 77];
y = [96 462 37 33];
improfile(I¡x¡y)¡grid on;
```



تصویر ۲۱-۲-۲



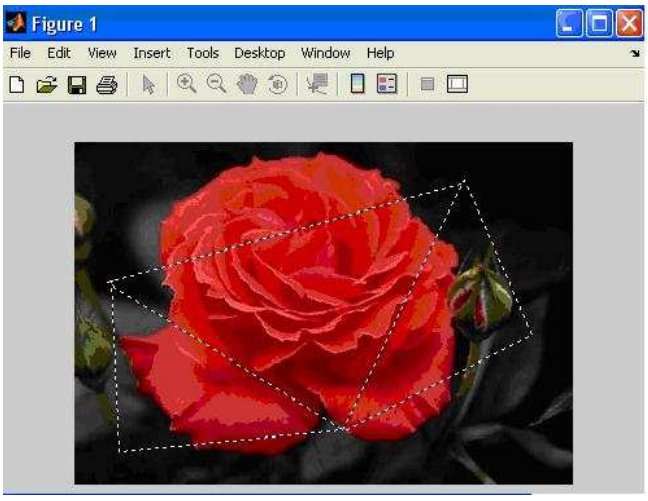
تصویر ۲۲-۲-۲

مثال

در این مثال نوع دیگری از اعمال تابع improfile را برروی تصویر مثال قبل بررسی می کنیم.

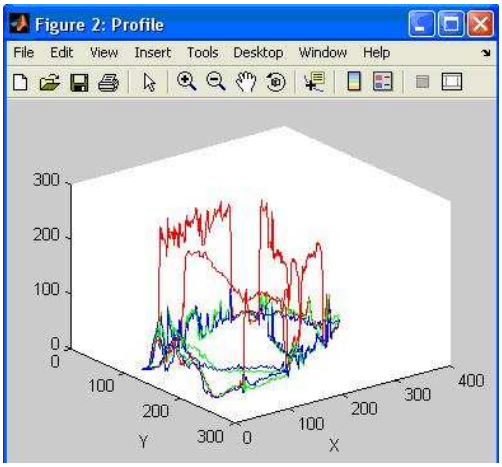
```
imshow flower.jpg
improfile
```

پس از فراخوانی، تصویر بالا توسط موس خطی به دلخواه به صورت زیر، بر روی تصویر رسم شده است:



تصویر ۲۳-۲-۲

Plot رسم شده برای تصویر بالا به صورت زیر می باشد:

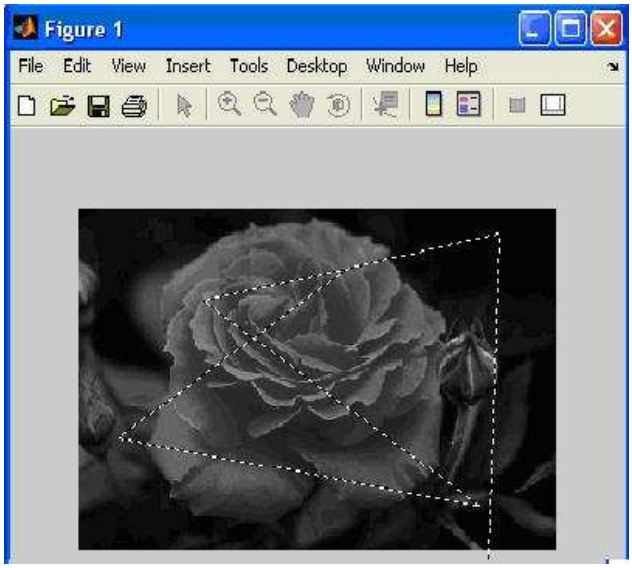


تصویر۲۴-۲-۲

مثال

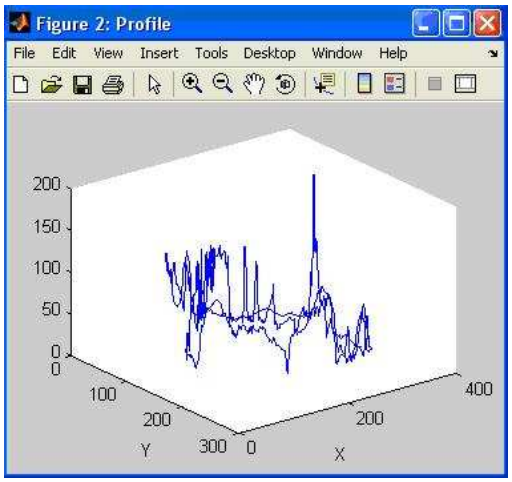
در این مثال هدف، بررسی تابع improfile بر روی تصاویر به فرمت GrayScale می باشد.

```
I = fitsread('solarspectra.fts');
imshow(I¡[]);
improfile
```



تصویر۲۵-۲-۲

Plot رسم شده برای تصویر بالا به تصویر زیر می باشد:



تصویر۲۶-۲-۲

تغییر سایز تصویر خروجی در نرم افزار Matlab

در نرم افزار قدرتمند Matlab این امکان وجود دارد، که بتوان به کمک توابعی به سادگی size تصاویر را به دلخواه تغییر داد.

برای نمونه درمواردی برای واضح تر شدن تصویر مربوطه نیاز است تصویر را بزرگ تر کرد و یا برعکس در جایی نیاز است، که جهت جلوگیری از فضای زیاد اشغال شده، سایز تصویر را کاهش داد.

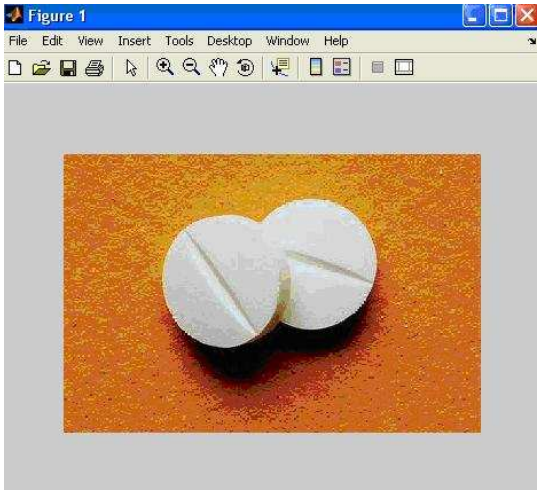
برای انجام این کار از تابعی بنام imre size استفاده می شود که در این تابع عدد نسبت داده شده در واقع تعیین کننده میزان بزرگ نمایی یا کوچک نمایی تصویر می باشد.

مثال: تصویر به نامpaslilles را فراخوانی کرده، سپس به کمک تابع imshow تصویر فراخوانی شده را نمایش داده و در انتها توسط تابع معرفی شده size تصویر را به دلخواه تغییر خواهیم داد.

```
i=imread(' pastilles.tif ') ;
imshow(i)
```

تصویر۲۷-۲-۲

```
u=imresize(I ,1.5) ;
imshow(u)
```



تصویر۲۸-۲-۲

همان گونه که در تصویر بالا مشاهده می گردد، با نسبت دادن عدد 1.5 به تابع imresize تصویر از حالت اولیه بزرگ تر شده است، حال جهت کوچک تر نمودن تصویر اولیه، لازم است که عدد نسبتاً کوچک تری را به تابع مربوطه نسبت داد.

مثال

فرض شود مقدار بزرگ نمایی تصویر مثال قبل به مقدار عددی 0.7 کاهش داده شده است، تصویر حاصل به صورت زیر تغییر می یابد.

```
u2=imresize(I,0.7) ;
imshow(u2)
```

تصویر۲۹-۲-۲

در عملیات مربوط به پردازش تصویر بزرگ نمایی تصاویر را zoom in و عملیات کوچک نمایی تصاویر را zoom out گویند.

چرخاندن یک تصویر در نرم افزار Matlab

در مواردی نیاز است، که یک تصویر به زاویه و در جهت دلخواه چرخانده شود و از تصویر حاصل جایگزین تصویر اولیه استفاده شود. جهت انجام این کار از تابعی بنام imrotate استفاده می شود.

این تابع به دو صورت مورد استفاده قرار می گیرد که به شرح زیر می باشد:

الف) چرخاندن تصویر(Imrotate Of Image)

ب) چرخاندن زاویه (Imrotate Of Angle)

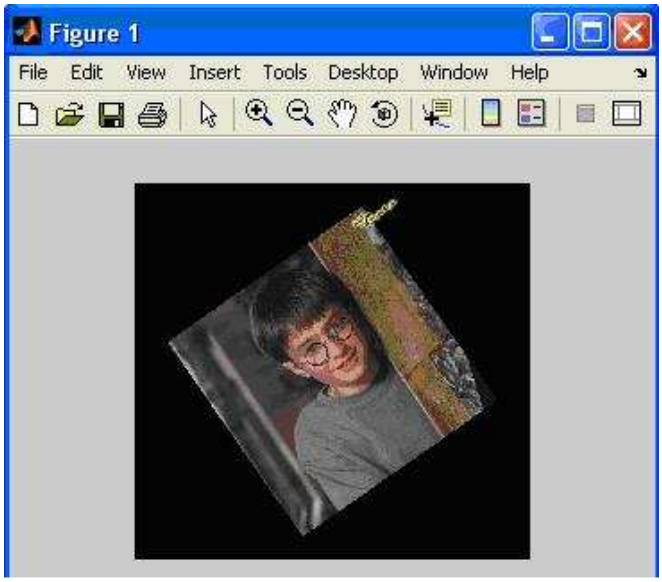
چرخش زاویه برحسب درجه به دلخواه صورت می پذیرد. در صورتی که مقدار عددی نسبت داده شده به تابع imrotate مقداری مثبت باشد، میزان درجه چرخش تصویر به صورت (ساعتگرد) می باشد و در صورتی که مقدار عددی نسبت داده شده به تابع منفی باشد، میزان چرخش به صورت پادساعتگرد خواهد بود.

مثال

در این مثال هدف چرخاندن تصویر boy با زاویه ۳۵ درجه مثبت (ساعتگرد) می باشد.

برای انجام این کار بدین صورت از تابع imrotate استفاده می شود.

```
i=imread('boy.jpg') ;
imshow(i)
j = imrotate ( i , 35 )
```

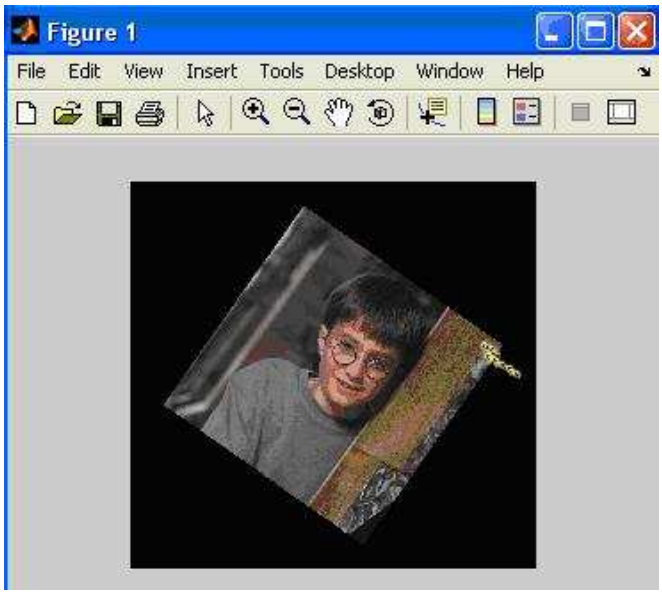


تصویر ۳۰-۲-۲

همانگونه که به وضوح مشخص است در تصویر 2-30 تصویر نسبت به حالت اولیه دقیقاً به مقدار 35 درجه ساعتگر چرخش یافته است.

مثال

جهت چرخش پاد ساعتگرد یک تصویر، درجهٔ چرخش باید به صورت عددی منفی اعمال شود. برای نمونه در مثال قبل، در صورتی که عدد ۳۵- جایگزین عدد ۳۵ شود، تصویر حاصل به صورت زیر خواهد بود.



تصویر ۳۱-۲-۲

چیدن ناحیهٔ دلخواه از تصویر در نرم افزار Matlab

در بسیاری از موارد، جهت انجام عملیات پردازش بر روی یک تصویر، ضروری است که قسمت مورد نظر از تصویر اصلی جدا شود. برای انجام این کار از تابع imcrop استفاده می شود، که به دو روش زیر قابل استفاده می باشد.

روش اول: چیدن ناحیهٔ دلخواه از تصویر به کمک mouse

بااستفاده از mouse، قسمتی از تصویر را click کرده، آن را نگه داشته، سپس محدوده ای که قصد چیدن آن ناحیه را دارید drag کرده و دکمهٔ mouse را رها کنید.

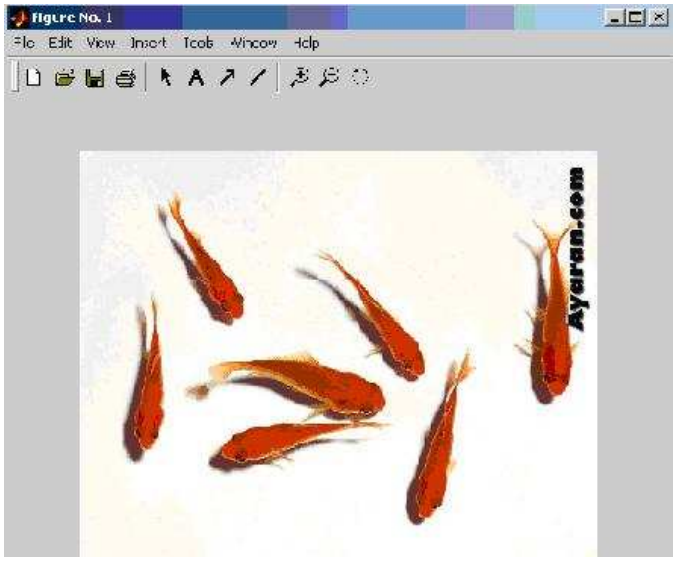
قابل ذکر است که با نجام این کار، ماتریس قسمت انتخاب شده در محیط Matlab نشان داده می شود.

مثال

تصویری به نام fish را فرخوانی کرده، محدودهٔ دلخواه را جهت انجام عملیات پردازش مربوطه از تصویر جدا می‌کنیم.

به کمک تابع imshow و صدا زدن متغیری که ماتریس ناحیهٔ crop شده در آن ریخته شده است، تصویر محدودهٔ crop شده، نمایش داده می شود.

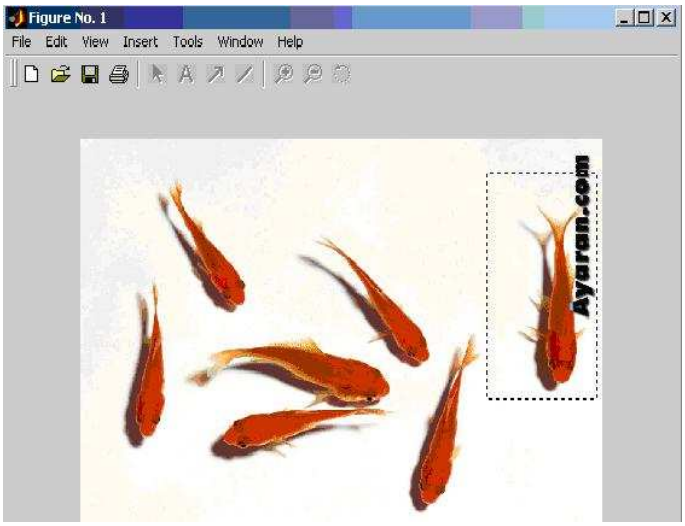
```
i=imread('fish.jpg') ;
imshow (i)
```



تصویر۳۲-۲-۲

حال جهت انتخاب محدودهٔ مورد نظر به کمک mouse، تابع imcrop را صدا می زنیم:

```
r=imcrop(i)
```



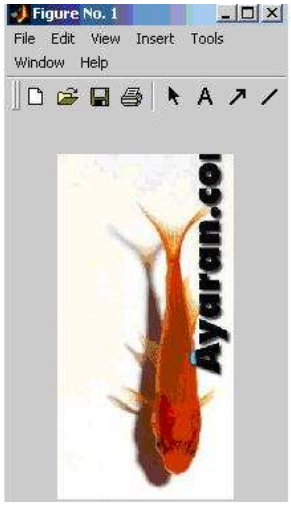
تصویر۳۳-۲-۲

در زیر قسمتی از ماتریس محدودهٔ crop شده از تصویر نشان داده شده است.

```
108   108   107   106   98
106   105    99    97   95
103   102    97    94   93
 98    98    94    93   91
 95    95    91    89   89
 93    93    93    93   89
 93    93    93    93   94
 93    93    95    94   93
 93    94    97    98   98
```

جهت مشاهدهٔ محدودهٔ crop شده، از دستور زیر استفاده می کنیم:

```
imshow(r)
```



تصویر۳۴-۲-۲

روش دوم: چیدن ناحیهٔ دلخواه از تصویر، توسط مختصات نسبت داده شده به تابع imcrop

با نسبت دادن مقادیر (y,x) محدودهٔ دلخواه را انتخاب نموده، سپس نتیجهٔ حاصل با استفاده از تابع imcrop مشاهده می شود.

نحوهٔ خواندن مختصات تصویر، به ترتیب از چپ به راست به صورت زیر است: y1، x1، x2، y2 براساس دیاگرام زیر

X1	X2
Y1	Y2

برای نسبت دادن اعداد مناسب برای crop کردن محدودهٔ مناسبی از تصویر، به نکات زیر توجه شود:

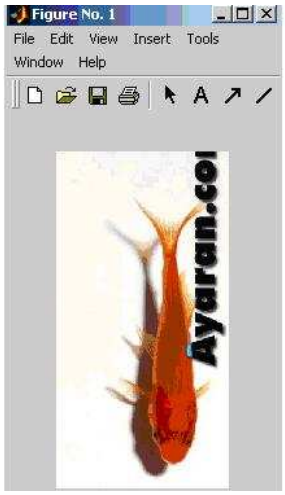
الف) برای انجام برش افقی بر روی تصویر لازم است، که مقادیر عددی مربوط به پارامتر xها را تغییر داد. (جهت بالابردن تصویر افقی باید مقدار x کاهش یابد).

ب) برای انجام برش های عمودی، x1 و y1 تغییر داده می شوند.

مثال

برای آشنایی بیشتر با روش فوق، تصویر fish به کمک روش بیان شده crop شده است که تصویر حاصل، بدین صورت مشاهده می گردد.

```
i=imread('fish.jpg');
i2=imcrop(I,[51 87 350 118]) ;
imshow (i)
figure,imshow(i2)
```



تصویر۳۵-۲-۲

محاسبهٔ مساحت تصاویر در نرم افزار Matlab

محاسبهٔ مساحت مربوط به یک تصویر در حالات مختلف، از جمله موارد پرکاربرد در علم پردازش تصویر می باشد. برای محاسبهٔ مساحت مربوط به یک تصویر، از تابعی با عنوان bwarea استفاده می شود.

Total در واقع یک مقدار عددی‌است، که ارزش آن معادل جمع پیکسل های on می باشد، اما ممکن است مقدار Total چندان دقیق نباشد، که علت این امر وجود الگوهای متفاوت از پیکسل های سفید در تصویر می باشد.

کلاس های پشتیبانی شده توسط این تابع

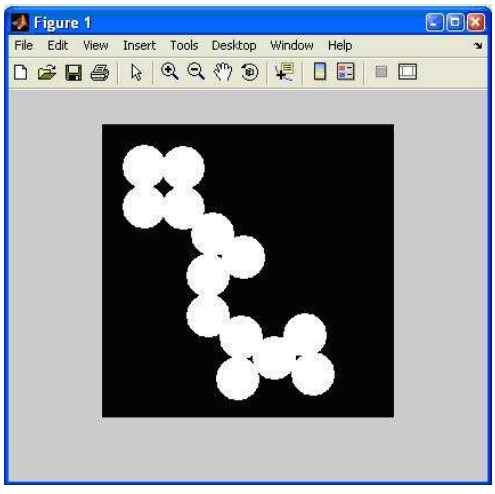
تصاویری که به کمک این دستور بر روی آنها عملیات پردازش صورت می گیرد، به فرمت عددی یا منطقی هستند. در برخی حالات ورودی هر پیکسل عددی غیر صفر است، که معادل حالت on می باشد.

همچنین قابل ذکر است، که Total از نوع double است.

مثال

در این مثال از یکی از تصاویر موجود در نرم افزار Matlab استفاده شده است. در ابتدا تصویر circles که در محیط Matlab تعریف شده است را فراخوانی کرده و سپس تابع bwarea را بر روی آن اعمال کنید.

```
i=imread('circles.png');
imshow(i)
bwarea (i)
```



تصویر۳۶-۲-۲

همان گونه که در بالا مشاهده شد، مقدار عددی مربوط به مساحت تصویر بالا به سادگی توسط تابع bwarea به دست آمده است.

تعیین تعداد Object های درون یک تصویر

پس از تبدیل تصویر به فرمت Binary، می توان از تابع bwlabel برای تعیین Object های درون یک تصویر استفاده نمود. برای نمونه، برای تعیین تعداد دانه های برنج در تصویر rice که در نرم افزار Matlab تعریف شده است این تابع به تمام ترکیبات متصل در تصویر BW برچسب زده و تعداد آنها را در متغیری به نام num Object قرار می دهد.

```
[labeled,numobjects]=bwlabed(bw,4);
Numobjects
Ans=
  101
```

بدین شکل نیز می توان نوشت:

```
Max(labeled(: ))
Ans=
 101
```

درستی نتایج به شرایط زیر وابسته است:

۱. اندازهٔ Object.
۲. Object ها تا چه حد ملموس و قابل تفکیک هستند.
۳. صحت و درستی تقریبی Background و تفکیک آن از Foreground تصویر مربوطه.
۴. چگونگی اتصالات انتخاب شده.

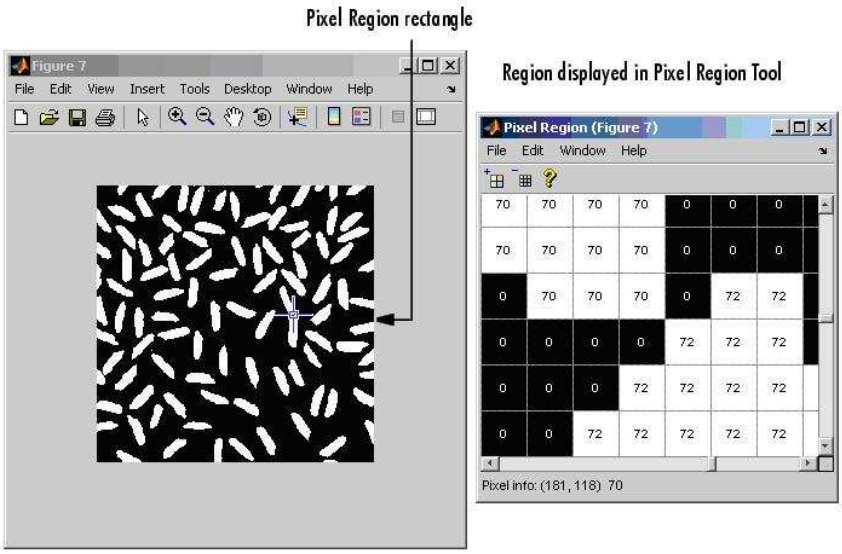
جهت دانستن ارزش مربوط به هر پیکسل، راه های متعددی وجود دارد. برای نمونه در صورتی که هدف، تعیین ارزش بخش کوچکی از تصویر باشد، می توان از تابع imcrop استفاده کرد.

روش دیگر، استفاده از ابزارهای محاسبه نواحی تصویر (Pixel Region Tools) می باشد که در این حالت، ماتریس برچسب ها به کمک تابع imshow نشان داده می شوند.

```
Figure,imshow(labeled);
```

در حالت impixelregion در واقع pixel region مستطیلی را به نام pixel region rectangle ایجاد می کند که در مرکز بخش انتخاب شده در تصویر، اگر این مستطیل حذف گردد، مقادیر جدید پیکسل ها در پنجره ای جدید به روز می شوند.

تصویر زیر ارزش پیکسل های دو دانه برنج را نشان می دهد که در آن پیکسل های صفر، نمایندهٔ Backgound می باشند.

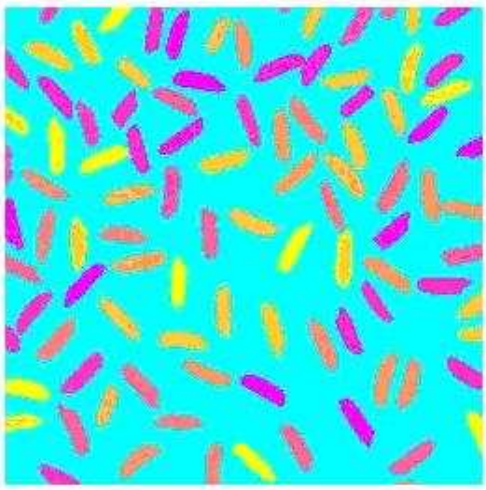


جهت آشنایی با چگونگی کار با تابع imcrop به مبحث مربوط به این تابع مراجعه شود.

روش مناسب جهت درک هرچه بیشتر ماتریس برچسب های تصویر، نمایش تصاویر index به صورت رنگی می باشد. در این ماتریس هر Object دارای رنگ متفاوتی است.

برای انجام این کار از تابعی به نام labe12rgb استفاده می شود. با استفاده از این تابع می توان Colormap متفاوت، رنگ Backgound دلخواه و Object های رنگی متفاوتی را حاصل نمود.

```
Pseudo_color=label2rgb(labeled,@spring,’c’,’shuffle’);
Imshow(pseudo_color);
```



تصویر۳۷-۲-۲

در زیر کاربرد هر یک از آلمان های موجود در دستور بالا به اختصار بیان شده است:

@: برای ساخت توابع به صورت دستی مورد استفاده واقع می شود، که این المنت قبل از اسم تابع آورده می شود.

Spring: نام یکی از Colormap های نرم افزار Matlab می باشد. از دیگر Colormap های نرم افزار Matlab می توان به مواردی نظیر Hsr, Pink, Gray همچنین می توان از Color map مشخصی که توسط برنامه نویس نوشته می شود استفاده نمود.

اگر نوع Color map انتخابی تعیین نشده باشد. نرم افزارMatlab به طور پیش فرض از Colormap مخصوصی با عنوان ColorCube استفاده می کند.

رشته C استفاده شده در دستور بالا نام اول رنگ دلخواه می باشد که توسط دستور Shuffle یا Colormap صدا زده شده است که در این جا (Spring) است، مخلوط می شود.

جهت راحتی استفاده در توابع مربوطه در زیر مخفف نام رنگ های مختلف، آورده شده است:

C:cyne

K:black

B:blue

G:green

M:magenta

R:red

W:white

Y:yellow

خصوصیات مربوط به Object های درون تصویر و اندازه گیری هر یک

ازجمله عملیات پرکاربرد و پردازش های صورت گرفته بر روی یک تصویر، تعیین و بررسی خصوصیات مربوط به Object های درون تصویر می باشد.

تابع regionprops در واقع خصوصیات Object های درون هر تصویر را اندازه گیری کرده و ماتریس هر یک را درون یک آرایه از نوع Structure قرار می دهد.

هنگامی که دستورات برچسب ها (Labels) با این تابع ترکیب شود، برای هر خصیصه یک Structure ساخته می شود.

برای نمونه هدف استفاده از regionprops برای ساخت آرایه Structure که شامل برخی خصوصیات اصلی برای Labels ها می باشد.

برای انجام این کار کافی است پارامتر رشته ای basic به آرگومان های regionprops اضافه گردد.

با انجام این کار، سه مفهوم اصلی به کاربر بازگردانده می شود که شامل مساحت (area)، مرکزیت center (of mass) و bounding box می باشد.

bounding box نماینده بخش کوچکی از تصویر می باشد، که در مثال زیر شامل قسمتی از دانه های برنج مربوط به تصویر rice می باشد.

مثال

در این مثال، هدف، بررسی کاربرد Basic در تابع regionprops می باشد.

```
Graindata=regionprops(labeled,'basic')
Ans=
 Graindata=
   101x1 struct array with field:
   Area
   Centroid
   boundingBox
```

Regionprops تصویر را به نواحی مختلف تقسیم می کند (بلوک بندی) که هریک از نواحی تقسیم بندی شده دارای مرکز، مساحت و مشخص هستند. برای نمونه جهت تعیین مساحت پنجاه و یکمین ناحیه یا به عبارتی 51 امین برچسب باید بدین طریق عمل کرد.

```
Graindata(51).Area
Ans=
   140
```

همچنین جهت یافتن مرکز و Boundingboxاین ناحیه به صورت زیر عمل می شود.

```
Graindata(51).BoundBox, Graindata(51).Centroid
Ans=
 107.5000    4.5000   13.0000    20.0000
Ans=
 114.5000   15.4500
```

در زیر حالات مختلف مربوط به تابع regionprops آورده شده است:

```
STATS=regionprops(L,properties)
STATS=regionprops(L,properties)
```

Regionprops تصویر را به نواحی مختلف تقسیم کرده، که تنظیمات خصوصیات مربوط به هر ناحیه در ماتریس Label به نام L اندازه گیری می شود، که برای هر ناحیه یک مقدار صحیح مثبت برای L در نظر گرفته می شود.

می توان چندین پارامتر برای خصیصه های متفاوت در این دستور استفاده کرد که با(،) ازهم جدا می شوند و به جای اندیکس properties در فرم کلی دستورات بالا نوشته می شوند.

برای دریافت و مشاهده تمام خصوصیات مربوط به هر تصویر، از پارامتر رشته‌ای all استفاده می شود.

پارامترهای این تابع عبارتند از:

'Area'	'EulerNumber'	'Orientation'
'BoundingBox'	'Extent'	'Perimeter'
'Centroid'	'Extrema'	'PixelIdxList'
'ConvexArea'	'FilledArea'	'PixelList'
'ConvexHull'	'FilledImage'	'Solidity'
'ConvexImage'	'Image'	'SubarrayIdx'
'Eccentricity'	'MajorAxisLength'	
'EquivDiameter'	'MinorAxisLength'	

تعریف

۱. "area": جهت تعیین مساحت هر ناحیه مورد استفاده واقع می شود.

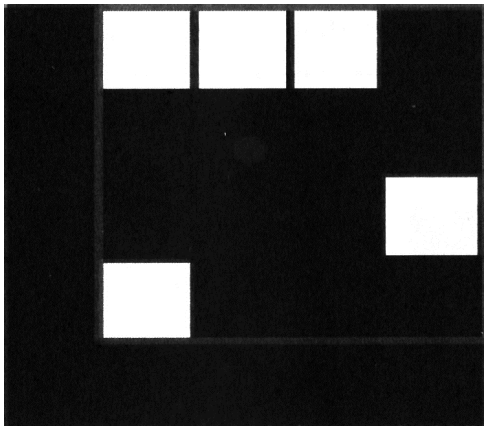
*

۲. "BoundingBox": برداری به صورت 1-by-ndims(L)*2 می باشد که تصویر را به نواحی کوچکی به شکل مربع تصویر تقسیم می کند. در این جا BoundingBox به معنی [ul-corner width] می باشد.

ul-corner به فرم [x y z…] بوده و تعیین کنندهٔ بالاترین گوشه سمت چپ در Bounding Box می باشد Width به فرم [x-widthy – width…] است و تعیین کننده پهنای BoundingBox می باشد.

۳. "Centroid": در این حالت بردار به صورت 1-by-ndims(L)*2 است و به عنوان مرکز هر ناحیه محسوب می شودو.

قابل ذکر است که اولین المنت، بیانگرCenteriod (مختصات افقی) x- coordinate و دومین المنت نشان دهنده (مختصات عمودی) یا y-coordinate را مشخص می کند.



تصویر۳۸-۲-۲نشان دهنده مرکزیت است و bounding box را نشان می دهد.

۴. "ConvexHull": ماتریسی به صورت p-by-2 می باشد به صورت کوچک ترین چندضلعی (plygon) محدب شامل نواحی تصویر که در آن هر ردیف از ماتریس شامل x-coordiate و y-coordinate از هر رأس Polygon می باشد.

این مشخصه تنها قابل استفاده در ماتریس های دو بعدی است.

۵. "ConvexImage": این حالت مربوط به تصاویر به فرمت منطقی یا Binary می باشد که در این حالت پوسته ای محدب از پیکسل ها، جهت تعیین مکان پیکسل مورد استفاده قرار می گیرد.

۶. "ConvArea": مقدار عددی است و در واقع نشان دهندۀ شمارۀ مربوط به هر پیکسل در ConvexImge می باشد، که این مشخصه تنها برای وروردی های دو بعدی قابل استفاده می باشد.

۷. "Eccentricity": مقدار عددی است و حالتی برای گریز از مرکز بیضی، که این حالت، نسبتی از فاصلۀ بین بیضی و طول قطب های آن است و دارای مقداری بین صفر و یک می باشد. قابل ذکر است که این مشخصه نیز تنها برای ورودی های دو بعدی کاربرد دارد

۸. "Equiv Diameter": این حالت نیز به صورت مقداری عددی تعریف می شود که در آن از قطر دایره برای یافتن مساحت ناحیه استفاده می گردد و با فرمول (Area/pi*4) محاسبه می شود.

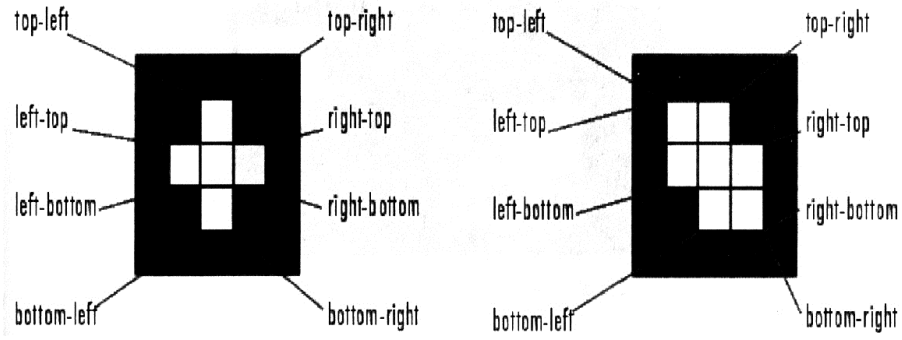
۹. "EulerNumber": مقدار عددی به صورت نسبتی از پیکسل ها که در bounding box هر ناحیه است را شامل می شود، که boundingbox ها و تقسیمات مربوط به هر یک جهت تعیین مساحت استفاده می شود.

این مشخصه نیز مانند قبل تنها در ورودی های دو بعدی کاربرد دارد.

۱۰. "Extrema" : ماتریس 8 در 2 می باشد که شامل نقاط Extrema در ناحیه است. محتویات ردیف و ستون به ترتیب شامل x و y-coofdinates هر نقطه می باشد.

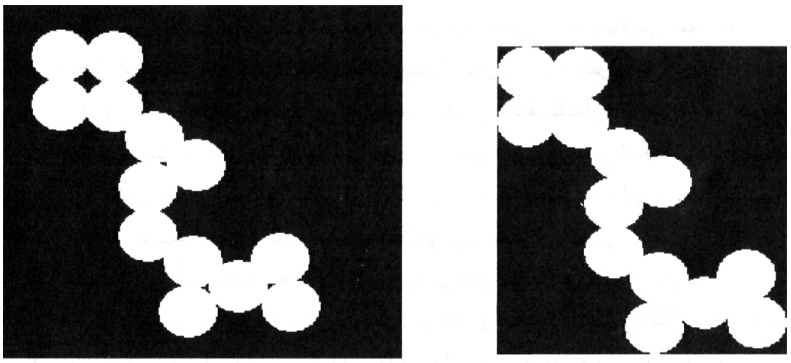
فرمت بردارها به صورت زیر تعریف شده است:

این مشخصه تنها برای ورودی های دو بعدی کاربرد دارد. تصاویر زیر نشان دهندۀ Extrema در دو ناحیۀ متفاوت است، که همان گونه که مشخص است در ناحیۀ سمت چپ، نقاط Extrema از هم مجزا بوده و در ناحیۀ سمت راست، نقاط Extrema مرکزی، (top-left and left-top.e.g) مشابه یکدیگر هستند.



۱۱. "filedArea": شامل تعداد پیکسل های فعال (on) در FilledImage می باشد.

۱۲. "FilledImage" : شامل تصاویر Binary هم اندازه با bounding box می باشد، که نواحی معادل پیکسل های روشن به همراه حفره های درون آنها هستند.



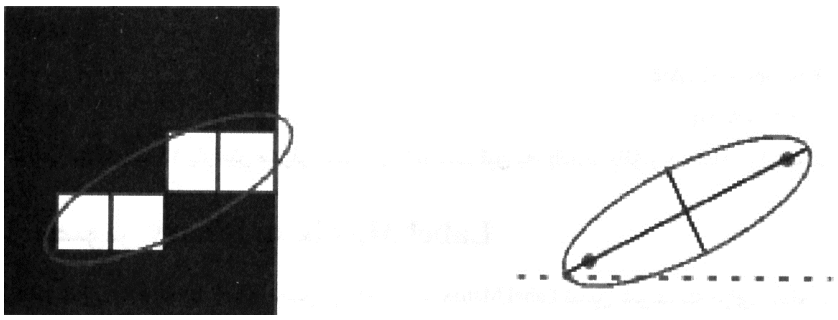
Original Image, Containing a Single Region

Image Returned

۱۳. "Image": شامل تصاویر Binary هم اندازه که در آن bounding box های ناحیه به عنوان پیکسل های فعال و سایر پیکسل ها به عنوان غیر فعال محسوب می گردند.

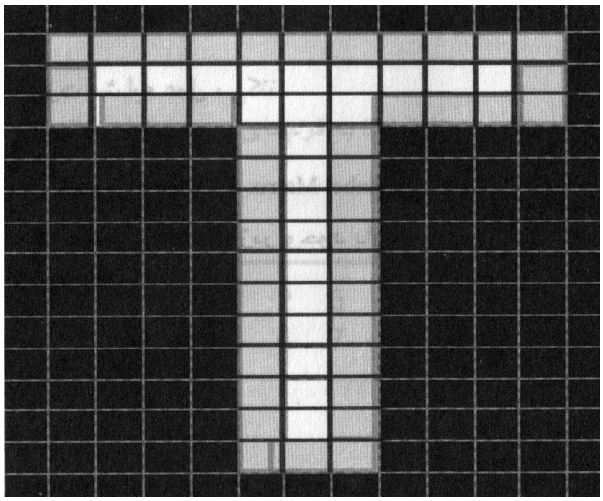
۱۴. "Orientation": مقدار عددی است که در واقع شامل زاویه ای (برحسب درجه) بردار x بیضی و بزرگترین بردارییضی که هم اندازه با دومین مقدار در ناحیه است. این مشخصه تنها برای ورودی های دو بعدی است.

۱۵. "perimeter": بردار p-element که شامل مجموعه پیکسل هایی است، که در مرزهای هر ناحیه در تصویر وجود دارد، P تعداد ناحیه هاست.



تصویر زیر، پیکسل هایی که شامل تعداد پیکسل های محیط نواحی هستند را نشان می دهد.

ذکر این نکته حائز اهمیت است که نواحی حتماً باید دارای هماسیگی باشند و در غیر این صورت نتیجه غیرقابل قبولی مشاهده می شود.



تصویر۳۹-۲-۲

۱۶. "PixelIdxList": بردار p-element که شامل Index های خطی پیکسل های ناحیه می باشد.

۱۷. "PixelList": یک ماتریس p-by-ndims(L) است. پیکسل های حقیقی در هر ناحیه و هر ردیف از ماتریس به فرمت [x y z ...] می باشد، که تعیین کنندهٔ هماهنگی پیکسل ها در ناحیه هستند.

۱۸. "Solidity": تناسب پیکسل ها در لایه های محبی که تقریباً در ناحیه ها هستند و Area/Convex Area را محاسبه می کند.

قابل ذکر است که ماتریس ورودی L هر کلاس عددی را می تواند پشتیبانی کند.

دستورات زیر برای محاسبهٔ مساحت استفاده می شوند، که معادل یکدیگر هستند.

Stats(1).Area,stats(2).Area,…,stats(end).Area

And

Stats.Area

و یا اینکه:

```
Stats=regionprops(L,'Area');
allArea=[stats.Area];
```

جهت آشنایی بیشتر با ۱۸ پارامتر معرفی شده در صفحات قبل به help نرم افزار Matlab مراجعه شود.

تبدیل تصویر Binary به Label Matrix

قبل از اعمال تابع Regionprops باید تصاویر Bainary به Lavbel Maxtrix تبدیل شوند، که برای انجام این کار به دو طریق زیر می توان اقدام کرد:

۱. استفاده از تابع bwfunction

```
L=bwlabel(BW);
```

۲. استفاده از تابع double

```
L=double(BW);
```

هر دو تابع فوق به صورت مشابه عمل می کنند.

مثال

هدف از این مثال، تبدیل تصویر Bainary به Lavbel Maxtrix می باشد.

فرض شود Label Maxtrix منطقی(Bainary) زیر وجود دارد:

```
1 1 0 0 0 0
1 1 0 0 0 0
0 0 0 0 0 0
0 0 0 0 1 1
0 0 0 0 1 1
```

Bwlable در واقع یک Label Maxtrix را با دو ناحیه همسایه ساخته، که با مقادیر صحیح 1 و 2 برچسب (Lable) شده اند.

```
Mylabel=bwlabel(BW)

Mylabel=
1  1  0  0  0  0
1  1  0  0  0  0
0  0  0  0  0  0
0  0  0  0  2  2
0  0  0  0  2  2
```

تابع double یک Lavbel Maxtrix را ساخته، که شامل یک ناحِیهٔ غیرهمسایه می باشد و با مقدار صحیح یک برچسب شده است.

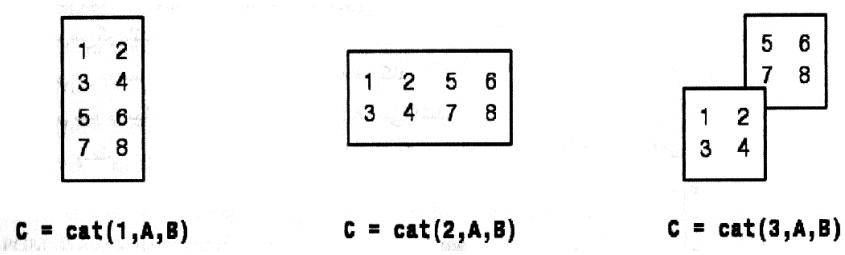
```
Mylabel2=
1  1  0  0  0  0
1  1  0  0  0  0
0  0  0  0  0  0
0  0  0  0  1  1
0  0  0  0  1  1
```

اتصال آرایه ها در نرم افزار Matlab

فرض کنید دو ماتریس A و B با مقادیری به صورت زیر وجود دارد:

```
A=          B=
  2   1       5   6
  3   4       7   8
```

در نرم افزار Matlab با استفاده از تابع Cat به فرم های زیر می توان تریس های A و B را به یکدیگر متصل نمود.



مثال

دستورات زیر نواحی برچسب زده مرکزی را محاسبه کرده و سپس آن را رسم می کند:

```
bw = imread('text.png');
L = bwlabel(bw);
s  = regionprops(L¡ 'centroid');
centroids = cat(1¡ s.Centroid);
imshow(bw)
hold on
plot(centroids(:¡1)¡ centroids(:¡2)¡ 'b*')
hold off
```

محاسبهٔ خصوصیات آماری هر Object در تصاویر

برای انجام این کار، مراحل زیر به ترتیب دنیال می شود.

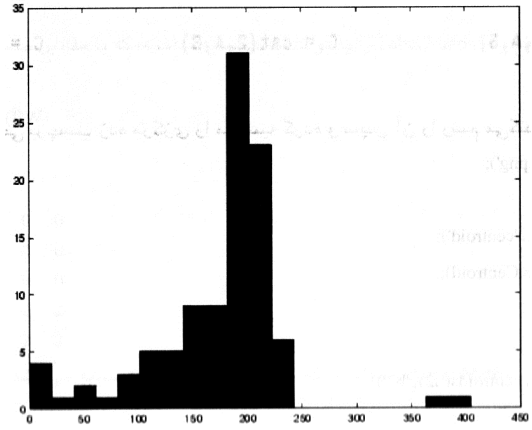
```
1)max([graindata.Area])

Ans=

    404

2)biggrain=find([graindata.Area]==404)

Biggrain=

    59

3)mean([graindata.Area])

Ans=

    175.0396

4)hist([graindata.Area],20)
```

دستور اول، مقدار ناحیهٔ اشغال شده توسط اتصالات Object ها (نواحی اشغال شده توسط دانه های برنج) را بر می گرداند.

دستور دوم، مساحت هر یک را محاسبه می کند.

دستور سوم، بیان کنندهٔ میانگین مساحت ها می باشد.

دستور چهارم، نمودار آماری مربوطه را رسم کرده، که به صورت زیر می باشد.



تصویر ۴۰-۲-۲

عملیات روی تصاویر در نرم افزار Matlab

در این قسمت به تعدادی از عملیات مهم و پرکاربرد، که بر روی تصاویر قابل اجرا می باشد، اشاره شده است.

1-Reading Images

2-Writing Image

3-Querying Graphicimage file for information

4-Converting Image between image storage classes

حال به بررسی هر یک از عملیات بیان شده برروی تصاویر می پردازیم.

پسوندهای پشتیبانی شدهٔ تصاویر در نرم افزار Matlab

تابع imread برای فراخوانی کردن تصاویر قابل استفاده می باشد، که توضیحات مربوط به این تابع در مباحث قبل آورده شده است. در این قسمت به بررسی پسوندهای پشتیبانی توسط این تابع می پردازیم:

>>BMP(Microsoft Window Bitmap)

>>CUR(Microsoft Window Cursor resource)

>>HDF(Hierarchical Data Format)

>>ICO(Window Icon resource)

>>JPEG(Joint Photographic Expert Group)

>>PBM(Portable Bitmap)

>>PCX(Window Painbrush)

>>PGM(Portable graymap)

>>PNG(Portable Network Graphics)

>>PPM(Portable pixmap)

>>RAS(Sun Raster image)

>>TIFF(Tagged Image File Format)

XWD(X Window Dump)

۱. تغییر پسوند تصاویر در نرم افزار Matlab

در برخی از موارد لازم است جهت انجام عملیات پردازش بر روی یک تصویر پسوند مربوط به آن تصویر را به صورت یکی از پسوندهای پشتیبانی شدهٔ دیگر در نرم افزار Matlab، تبدیل نمود.

برای انجام این کار از تابع imwrite استفاده می شود. برای آشنایی بیشتر با نحوهٔ استفاده از این تابع، مثال زیر بررسی شود.

مثال

تصویری با عنوان pout که از نمونهٔ تصاویر تعریف شده در Matlab می باشد را فراخوانی کرده، پسوند پیش فرض تصویر مذکور tif می باشد. هدف تبدیل پسوند تصویر بالا به فرمت png می باشد، که از دیگر پسوندهای پشتیبانی کنندهٔ تصاویر می باشد.

برای انجام این کار بدین صورت عمل می شود.

```
i=imread('pout.tif');
imshow(i)
imwrite(i,'pout.png');
imfinfo('i.png');
```

```
File name: 'pout.png'
    FileModData: '02-Dec-2005  17:04:32'
       Filesize: 31559
         Format: 'png'
  FormatVersion: []
          width: 240
         Height: 291
      BitDtpth: 8
```

پس از تبدیل پسوند تصویر، به فرمت png جهت اطمینان از تبدیل صورت گرفته شده از دستور imfinfo استفاده شده است، که همان گونه که مشخص است، تبدیل به صورت صحیح انجام گرفته است.

۲. دریافت اطلاعات کامل مربوط به تصویر در نرم افزار Matlab

برای دریافت اطلاعات کامل مربوط به یک تصویر، imfinfo استفاده می شود که شرح کامل این تابع در مباحث قبل آورده شده است.

۳. تبدیل کلاس های پشتیبانی شده به یکدیگر در نرم افزار Matlab

در بعضی موارد لازم است برای اجرای برخی از توابع، نوع کلاس آن تصویر را به فرمت دیگری از کلاس های پشتیبانی شده تبدیل نمود، که به سادگی می توان نوع کلاس تصاویر ذخیره شده در Matlab را به فرمت دیگری تبدیل نمود.

از قبل می دانیم که انواع کلاس های پشتیبانی شدهٔ مربوط به تصاویر، به فرمت (Double,Uint16,Uint8) می باشند.

جهت تبدیل هر یک از انواع کلاس های بالا به یکدیگر از توابع زیر استفاده می شود، که کاربرد هر یک متناسب با نام آنها مشخص است.

Im2double

Im2uint8

Im2uint16

جهت آشنایی بیشتر با توابع مطرح شده، به مثال زیر توجه شود.

مثال

در این مثال نیز مجدداً با فراخوانی تصویر pout در محیط Matlab و استفاده از تابع whos برای تعیین نوع کلاس اولیه این تصویر نوشته می شود.

```
I=imread('pout.tif');
Whos('i')
```

Name	size	Byte Class
i	291*240	69840 uint8 array

Grand total is 69840 elements using 69840 byte

همان گونه که مشاهده می شود کلاس تصویر فراخوانی شده به صورت پیش فرض به فرمت uint 8 می باشد.

حال قصد داریم تصویر مذکور را به فرمت double تبدیل می کنیم.

```
U=im2double(i);
Whos('u');
```

Name	size	Byte Class
u	291*240	558720 uint8 array

Grand total is 69840 elements using 558720 byte

همان گونه که مشاهده می شود، فرمت کلاس تصویر فراخوانی شده از uint8 به double تبدیل یافت.

نکاتی در مورد تبدیل انواع تصاویر و فرمت کلاس های مربوطه

۱. معمولاً تبدیل انواع تصاویر (RGB-Index-Intensity-Binary) و یا تبدیل انواع کلاس ها به یکدیگر بدین منظور انجام می گیرد، که برخی از دستورات Matlab تنها برای نوع خاصی از تصاویر و یا نوع خاصی از کلاس ها قابل اجرا می باشند.

برای نمونه در مبحث لبه برداری تصاویر (edge detection)، که در فصول بعد به طور کامل بیان خواهد شد. توضیح خواهیم داد که عملیات لبه برداری تنها بر روی تصاویر به فرمت Grayscale پاسخ دلخواه را می دهد وبر روی انواع دیگر تصاویر قابل اجرا نمی باشد.

۲. معمولاً هنگام تبدیل کلاس های مربوط به تصاویر، مقداری از اطلاعات مربوط به آن تصویر گم می شود، که اصطلاحاً در علم پردازش تصویر، وضعیت فوق را Losing information گویند.

برای نمونه زمانی که تصویری به فرمت uint16 از نوع intensity را که حداکثر ظرفیت برای ذخیره شدن آن 65.536 است به فرمت uint8 تبدیل می گردد بدین دلیل که تصور کلاس Uint 8 جهت ذخیره سازی به 256 حالت خاکستری فضا نیاز دارد، مقداری از اطلاعات تصویر گم خواهد شد. تمام ارزش های بین 0 تا 127 در تصویر orginal به 0 و بین 128 تا 385 به 1 تبدیل می گردد. بنابراین گم شدن اطلاعات، مشکل چندانی برای تصویر به وجود نخواهد آورد.

۳. نکتهٔ دیگری که درمورد کلاس های یک تصویر حائز اهمیت است، این مسئله است که همیشه برای برنامه نویس مقدور نیست، که بتواند تصویر نوع index را به سایر انواع کلاس ها تبدیل کند.

برای نمونه یک تصویر uint 8 or double به فرمت index را با پالت رنگ 300 رنگی را به هیچ عنوان نمی توان به uint8 تبدیل کرد. علت آن است که uint8 تنها دارای 256 حالت است.

قابل توجه است که در صورتی که نیاز به تبدیل یک تصویر uint8 یا double به فرمت index شد، ابتدا باید به کمک تابع imapprox تعداد رنگ های تصویر orginal را کاهش داده و سپس تبدیلات لازم را اعمال کرد.

جهت کسب اطلاعات بیشتر در مورد تابع imapprox در محیط Matlab دستور زیر را تایپ کنید. (help imappax)

تبدیل فرمت تصاویر باینری

عمق ذخیرهٔ این فایل ها به صورت یک بیتی می باشد.

مثال :فرمت tif، باینری را پشتیبانی می کند و در مثال زیر فایل باینری BW تبدیل به فایل tif شده و هم زمان نیز نام آن به test تغییر یافته است.

```
Imwrite(BW,'test.tif');
```

برای دیدن تغییرات از تابع imfinfo استفاده می شود:

```
Imfinfo('test.tif')
```

Ans=

```
      File name: 'd:\mystuff\grid.tif'
    FileModData: '25-Nov-1998  11:36:17'
       Filesize:  340
         Format: 'tif'
  FormatVersion: []
          width: 20
         Height: 20
       BitDtpth: 1
     Color Type: 'grayscale'
Formatsignature: [73 73 42 0]
      Byteorder: 'little-endian'

 NewsubfileType: 0
  Bitspersample: 1

    compression:'CCITT ID'
```

نمایش تصاویر به کمک تابع Imview

تابع imvew از توابع پرکاربرد در نرم افزار Matlab و به ویژه در toolbox image processing می باشد. کاربرد اصلی این دستور برای نمایش تصاویر می باشد، که در واقع کاربردی مشابه تابع imshow دارد.

امکانات ویژه و منحصر به فرد مربوط به این تابع باعث به وجود آمدن مزایایی نسبت به imshow شده است.

برای آشنایی بیشتر با این تابع به مثال زیر توجه شود.

مثال

جهت بررسی تابع imvew جهت نمایش تصویر pout بدین صورت عمل می شود.

I=imread(‘pout.tif’);

Imveiw(i)

تصویر ۴۱-۲-۲

همان گونه که در تصویر ۴۱-۲ مشاهده می شود، علاوه بر تصویر اصلی در پنجره کوچک تری به نام oveview نیز تصویر نمایش داده شده است.

سایر امکانات کاربردی مربوط تابع Imview

از دیگر امکانات این تابع می توان به موارد زیر اشاره نمود:

۱. با قراردادن mouse بر روی هر پیکسل از تصویر مربوطه، مختصات رنگی آن تصویر در پایین تصویر نشان داده می شود. در صورتی که تصویر به فرمت رنگی (RGB) باشد. سه عدد که به ترتیب نماینده مختصات رنگ های قرمز، سبز و آبی می باشد، نمایش داده می شود و در صورتی که تصویر کلاس Binary باشد تنها یک عدد نمایش داده می شود. که اگر آن عدد 0 باشد، نمایندهٔ مشکی بودن رنگ آن محدوده و اگر 1 باشد به معنای سفید بودن رنگ آن محدوده می باشد.

همان گونه که در تصویر ۴۱-۲ مشاهده می شود نقطه ای که mouse بر روی آن قراردارد نمایندهٔ عدد 83 می باشد.

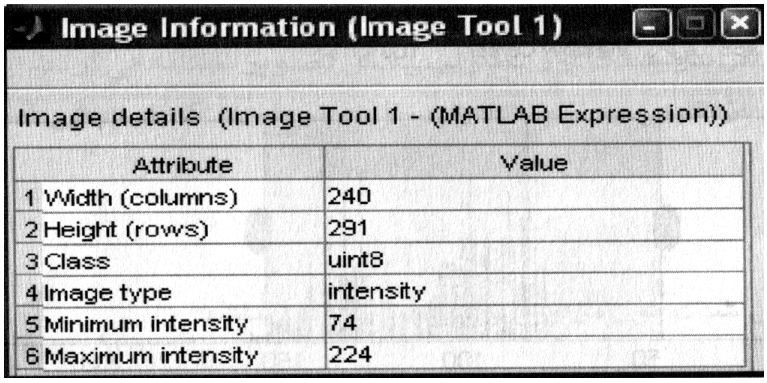
۲. پس از اجرای این تابع در تصویر حاصل نقطه ای کهmouse بر روی آن قرار دارد مختصات position (x,y) آن نقطه نیز در زیر تصویر نوشته می شود.

در تصویر فوق نقطه ای که mouse بر روی آن قرار داشته است، مختصات x آن عدد 120 و مختصات y آن عدد 23 می باشد.

۳. از دیگر مزایای بسیار مفید این تابع، آن است که به کمک این تابع می توان نوع فرمت تصویر نمایش داده شده را تشخیص داد که به فرمت (Intensity,RGB و ...) می باشد و با قراردادن mouse در بیرون از ناحیهٔ اصلی تصویر، در زیر تصویر مربوطه نوع آن نوشته خواهد شد.

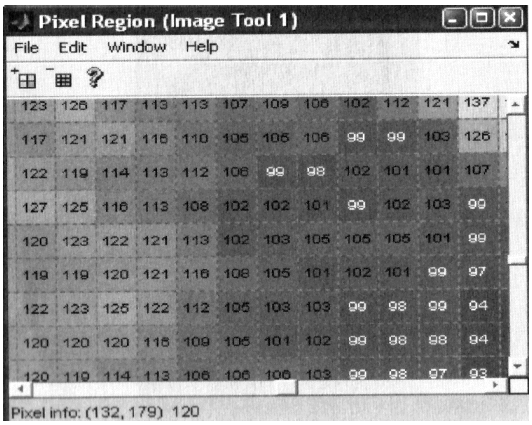
۴. از منوی tool و گزینهٔ Image information، اطلاعات دقیق مربوط به تصویر را می توان مشاهده نمود، که شامل مواردی نظیر نوع تصویر، Class، وزن و ارتفاع تصویر، min وmax شدت نور و مواردی از این قبیل می باشد.

در تصویر ۴۲-۲ پنجرهٔ اطلاعات مربوط به تصویر موجوئ در مثال قبل قابل مشاهده است.



تصویر۴۲-۲-۲

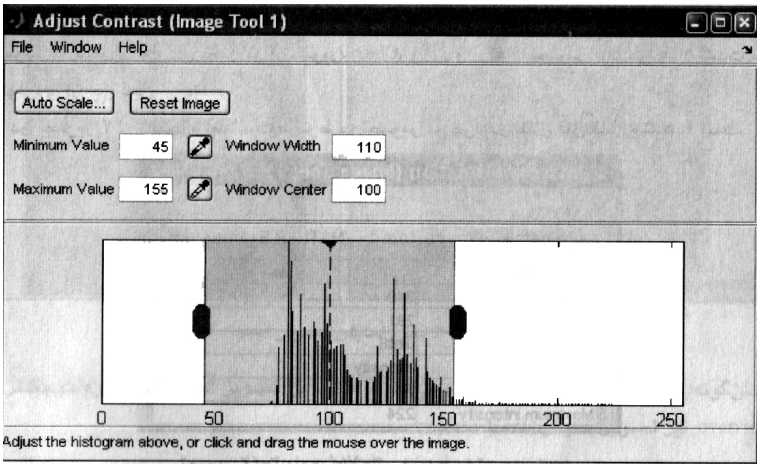
۵.از منوی tools، گزینه pixel Region را انتخاب کرده، به کمک این قسمت می توان به صورت بلوکی و با رنگ های دقیق تصویر تمام مشخصات هر بلوک و هر پیکسل را اعم از مختصات و اعداد مربوط به رنگ هر پیکسل مشاهده نمود.



تصویر۴۳-۲-۲

۶. از دیگر امکانات جالب توجه این تابع، افزودن و کاهش شدت نور تصویر به صورت دستی می باشد، که در مباحث قبل، تابع مخصوص این کار بیان شده است.

جهت انجام این کار از منوی tools گزینه Adjust Contrast را انتخاب کرده و پنجره ای مشابه تصویر ۴۴-۲ باز می شود.



تصویر۴۴-۲-۲

همان گونه که در پنجره بالا مشاهده می شود، با

تغییر اعداد مربوطه، می توان شدت نور تصویر را کم یا زیاد کرد.

2.3بهینه سازی تصاویر

افزایش شدت نور مربوط به یک تصویر در نرم افزار Matlab

در برخی موارد نیاز است به دلایلی مانند کیفیت تصویر، شدت نور یک تصویر افزایش یابد و در واقع ارزش عددی جهت افزایش شدت نور تصویر، به کل تصویر اضافه می گردد.

جهت انجام ای کار از تابعی به نام histeq استفاده می شودف که فرم کلی استفاده از این تابع به صورت زیر می باشد:

histeq(نام متغیر)

مثال

در این مثال هدف افزایش شدت نور مربوط به تصویر food می باشد:

```
i=imread('food.tif ') ;
imshow(i)
i2=histeq(i);figure, imshow(i2)
```

تصویر۱-۳-۲

پس از استفاده از تابع imhist و افزایش شدت نور تصویر بالا، نتیجهٔ حاصل به صورت تصویر۲-۳-۲ مشاهده می گردد:

تصویر۲-۳-۲

توجه: با استفاده از figure در پشت توابعی نظیر imshow، تصویر نهایی در پنجره ای جدید نمایش داده می شود.

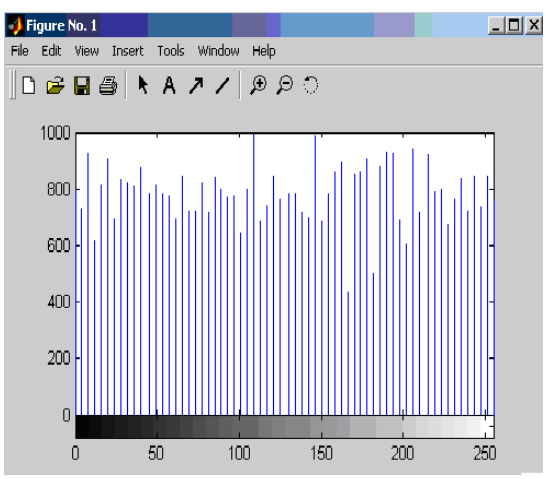
هدف استفاده از این تابع، نمایش چند تصویر به صورت هم زمان می باشد.

جهت مشاهدهٔ تغییرات مربوط به شدت نور تصاویر، پس از استفاده از تابع histeq می توان نمودار مربوط به شدت نور هر دو تصویر را بررسی و تغییرات مربوط به هریک را مشاهده نمود.

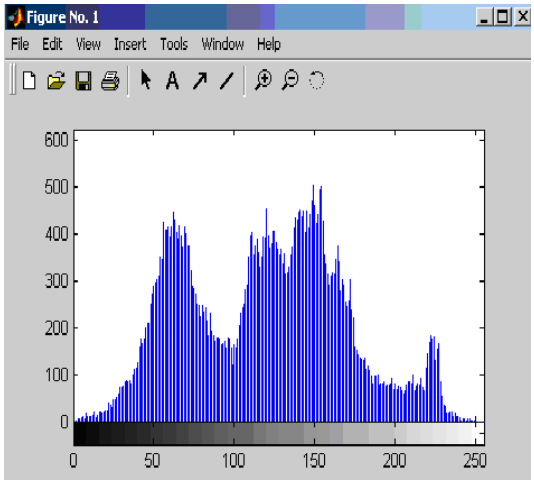
مثال

جهت بررسی تغییرات اعمال شده بر روی شدت نور تصاویر مربوط به متغیر های i و i2 د رمثال قبل به صروت زیر عمل می شود:

imhist(i)
imhist(i2)

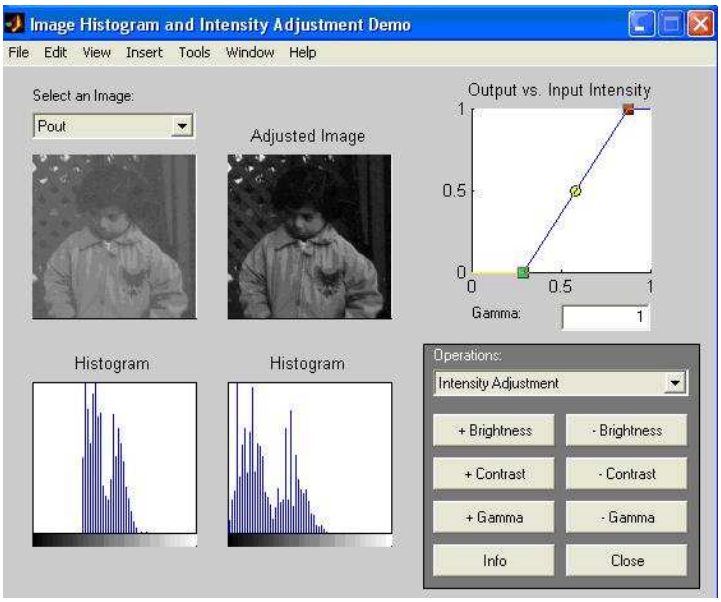


تصویر۳-۳-۲



تصویر۴-۳-۲

همان گونه که در نمودارهای بالا مشخص است، شکل شماره ۳-۳ نمودار شدت نور مربوط به تصویل اصلی food در حالت اولیه می باشد و پس از افزایش شدت نور، نمودار شدت نور آن به صورت شکل ۴-۳ مشاهده می گردد.



انجام عملیات چهارگانه محاسباتی بر روی تصاویر

انجام عملیات چهارگانه محاسباتی (جمع، تفریق، ضرب و تقسیم) از جمله عملیات مهم و پرکاربرد در علم پردازش تصویر می باشد و جهت پردازش های مختلف بر روی تصاویر، کاربردهای متعددی دارد. به طور کلی تقسیم بندی این عملیات به صورت زیر می باشد:

1- Adding Image 2- Substracting Imag

3- Multi playing Image 4- Diriding Image

نکته: توجه شود انجام عملیات چهارگانه محاسباتی بر روی تصاویر، حتماً باید تصاویر از یک کلاس و به یک اندازه باشند، در غیر این صورت نرم افزار مطلب قادربه انجام عملیات پردازش بر روی آنها نخواهد بود.

۱. عملیات جمع بر روی تصاویر

همان گونه که مشخص است، در این قسمت هدف اصلی شامل دو مرحله است:

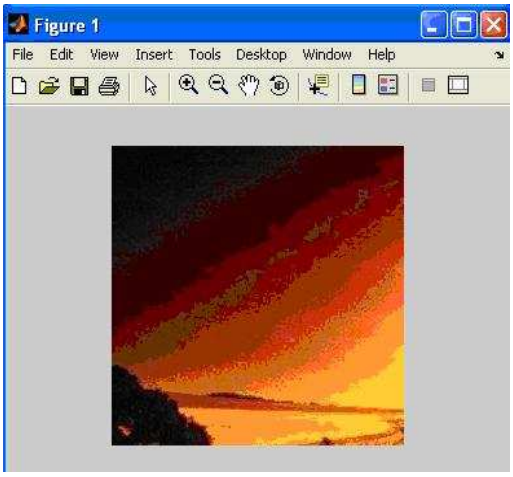
مرحلهٔ اول: جمع کردن دو تصویر با یکدیگر

مرحلهٔ دوم: جمع کردن یک ارزش عددی به تصویر

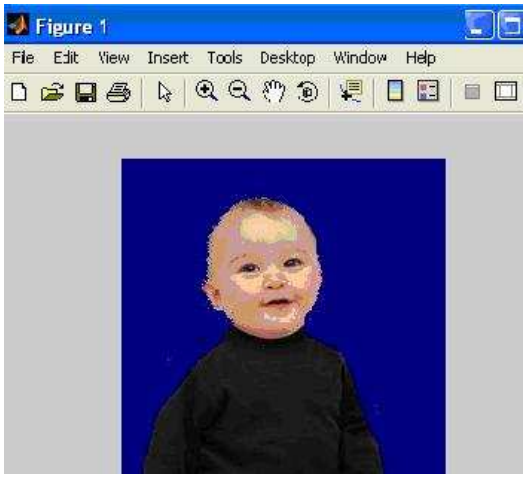
در حالت اول به طور مستقیم، هر دو تصویر با یکدیگر جمع شده است و در حالت دوم ارزش عددی به تک تک پیکسل های تصویر اضافه شده است.

مثال :جهت درک مفهوم جمع تصاویر به مثال زیر توجه کنید:

```
i=imread('child.tif') ;
j=imread('sunset.tif') ;
imshow(i)
imshow(j)
```



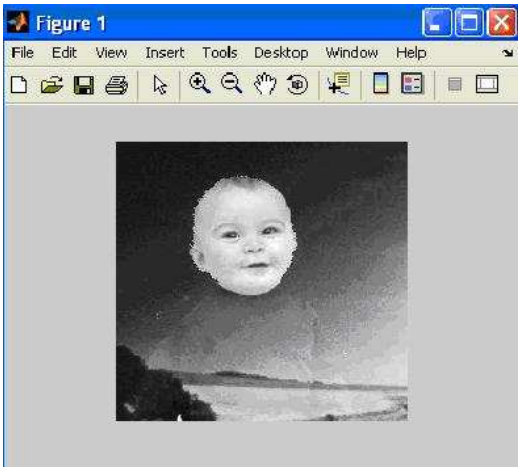
تصویر۶-۳-۲



تصویر۵-۳-۲

حال جهت جمع کردن تصاویر فوق، بدین صورت اقدام می کنیم:

```
k=imadd(i , j) ;
 imshow(k)
```



تصویر۷-۳-۲

همچنین می توان با استفاده از تابع جمع، ارزش عددی را به تمام پیکسل های یک تصویر اضافه نمود، که معمولاً از این کار جهت افزایش شدت روشنایی یک تصویر استفاده می شود.

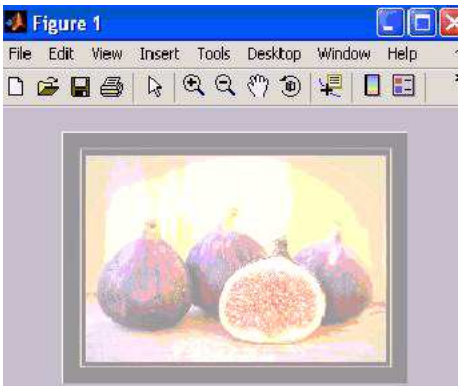
مثال

در این مثال، هدف، بررسی نتیجهٔ جمع تصویر ۸-۳ با ارزش عددی ۱۰۰ می باشد.

```
J=imread('fig.tif');
imshow(J)
```

```
T=imadd(J.150) ;
imshow(T)
```

تصویر۸-۳-۲



تصویر۹-۳-۲

همان گونه که در تصویر بالا مشاهده می گردد، مقدار عددی ۱۰۰ به تمام پیکسل های تصویر اصلی اضافه شده است و در نتیجه شدت روشنایی تصویر، حاصل به میزان عدد انتخابی افزایش یافته است.

۲. عملیات تفریق بر روی تصاویر

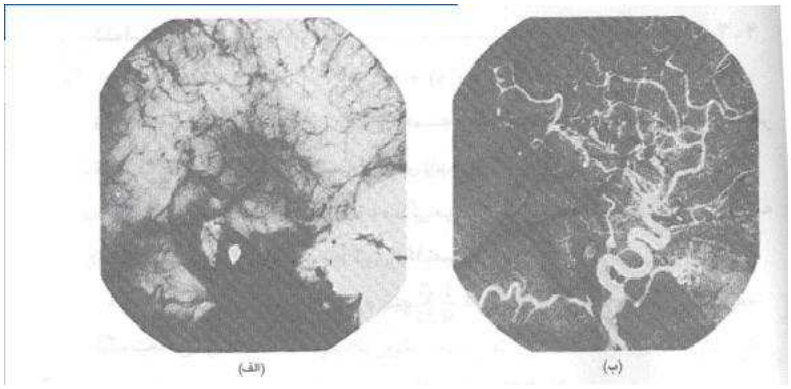
تفاضل دو تصویر $f(x,y)$ و $h(x,y)$ که به صورت $g(x,y) = F(x, y) – h(x, y)$

بیان می شود، با محاسبهٔ تفاضل بین هر یک از زوج پیکسل های متناظر در f و h به دست می آید.

از کاربردهای معادلهٔ بالا در مسئلهٔ ارتقاء مبحثی از تصویر برداری پزشکی به نام پرتوپردازی حالت نقاب می باشد. در این مورد نقاب $h(x,y)$ تصویر اشعه x قسمتی از بدن بیمار است که با قرار گرفتن یک تشدید کننده و یک دوربین تلویزیونی (به جای فیلم اشعه x معمولی) در مقابل منبع مولد اشعه x برداشته شده است.

تصویر $f(x,y)$ نمونه ای از مجموعه تصاویر مشابه از یک ناحیهٔ بدن انسان است، که پس از تزریق مادهٔ رنگی به شریان آن ناحیه برداشته شده است.

تفریق نقاب از هر نمونه در دنبالهٔ تصاویر تلویزیونی ورودی باعث خواهد شد، که تنها جزئیات ان دسته از نواحی که بین $f(x,y)$ و $h(x,y)$ متفاوت هستند، در تصویر به صورت جزئیات ارتقاء یافته دیده شود. بدان دلیل که دریافت تصاویر در نرخ های تلویزیونی امکان پذیر است، این روش تصویر متحرکی از چگونگی انتشار ماده رنگی در رنگ ها، تولید می کند.



تصویر ۱۰-۳-۲ (الف)　　تصویر ۱۰-۳-۲(ب)

مثال

تصویر الف، در واقع تصویر اشعهٔ X از بالای سر بیمار است، که قبل از تزریق مادهٔ رنگی ید (Iodine) dye به شریان برداشته شده است. دوربینی که این تصویر را تولید کرده است. بالای سر بیمار و رو به پایین جاسازی شده است.

به عنوان مرجع، مقایسهٔ نقطه روش در یک سوم پایین تصویر، هستهٔ ستون فقرات جاسازی شده است. تصویر ب، تفاضل بین نقاب تصویر الف و تصویری که مدت کوتاهی پس از ورود مادهٔ رنگی به شریان گرفته شده است را نشان می دهد. قابل توج است، که رگ های روشن که مادهٔ رنگی را حمل می کنند به صورت بارزی در تصویر ب ارتقاء یافته اند.

این رنگ ها کاملاً روشن ظاهر می شوند، زیرا تفریق نمی شوند (چرا که بخشی از تصویر اولیه نیستند).

زمینه در کل بسیار تیره تر از تصویر الف است، زیرا حاصل تفاضل در نواحی کم تغییر، مقادیر کوچکی را می دهد، که این مقادیر به صورت سایه های خاکستری تیره در تصویر تفاضل ظاهر می شود.

به عنوان مثال توجه شود نخاع در شکل الف روشن است، درحالی که در تصویر ب، بر اثر تفریق کاملاً تیره به نظر می رسد.

همان گونه که مشخص است، در این مرحله، هدف اصلی شامل دو قسمت عمده است:

قسمت اول: تفریق کردن دو تصویر از یکدیگر

قسمت دوم: تفریق کردن یک ارزش عددی از تمام پیکسل های مربوط به تصویر

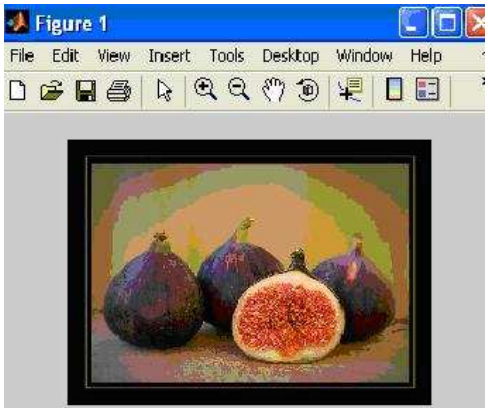
در مرحلهٔ اول، ارزش هر پیکسل از تصویر اول را، از همان شماره پیکسل از تصویر دوم تفریق شده و حاصل تصویر نهایی در خروجی نمایش داده می شود.

همچنین می توان یک ارزش عددی را ازتمام پیکسل های آن تصویر کم کرد، که در این صورت به دلیل کوچکتر شدن ارزش پیکسل ها، شدت نور تصویر نهایی کاهش می یابد.

مثال

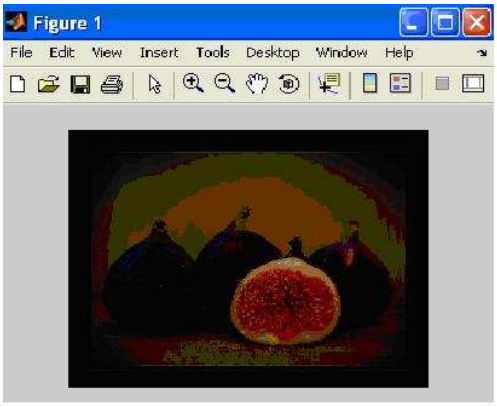
در ابتدا تصویر مربوطه را فرخوانی کرده و ارزش عددی ۱۰۰ را ازتمام پیکسل های آن کم می کنیم. تصویر تهایی به صورت زیر مشاهده می شود:

```
i=imread('fig.tif') ;
imshow(i)
```



تصویر ۱۱-۳-۲

```
r=imsubtract (i ,100) ;
imshow(r)
```



تصویر ۱۲-۳-۲

همان گونه که در تصویر بالا مشخص است، با توجه به میزان ارزش عددی کم شده نسبت به تصویر اولیه، میزان روشنایی تصویر کاهش یافته و تصویر به صورت تیره تر نمایش داده می شود.

۳. عملیات ضرب بر روی تصاویر

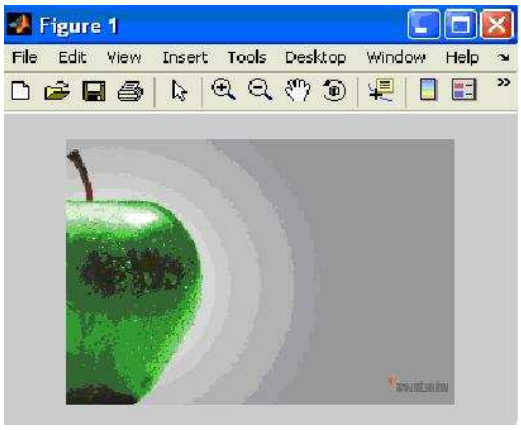
دراین قسمت، ضرب نظیر به نظیر تصاویر مورد بررسی قرار می گیرد. نماد استفاده شده در این مرحله به صورت (*.) می باشد. در حالت ضرب نظیر به نظیر تصاویر، تک تک پیکسل های هم شماره از هر دو تصویر انتخابی در یکدیگر ضرب شده و تصویر حاصل در خروجی نمایش داده می شود. از این حالت معمولاً جهت افزایش شدت روشنایی یک تصویر استفاده می گردد.

علت افزایش روشنایی آن است، که در ضرب نظیر به نظیر، ارزش تمام پیکسل های هم شماره از هر دو تصویر در یکدیگر ضرب شده و در نتیجه تصویر نهایی تصویر با ارزش پیکسل بالاتری خواهد بود. جهت انجام عملیات ضرب از تابعی با عنوان immultipy استفاده می شود.

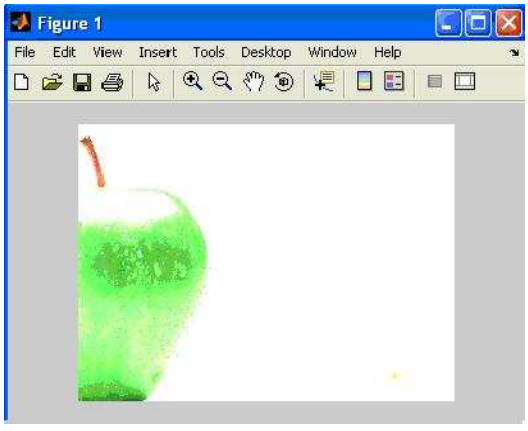
نکته: در حالت ضرب تصاویر نیز مانند حالات مربوط به جمع و تفریق می توان ارزش عددی را در تصاویر مربوطه ضرب نمود.

مثال :دراین مثال، هدف ضرب ارزش عددی ۳ در تصویر فراخوانی شده در زیر می باشد، که برای انجام این کار بدین صورت عمل می شود.

```
i=imread('apples.jpg') ;
imshow(i)
p=immultiply(I,3) ;
imshow(p)
```



تصویر ۱۳-۳-۲



تصویر ۱۴-۳-۲

۴. عملیات تقسیم بر روی تصاویر

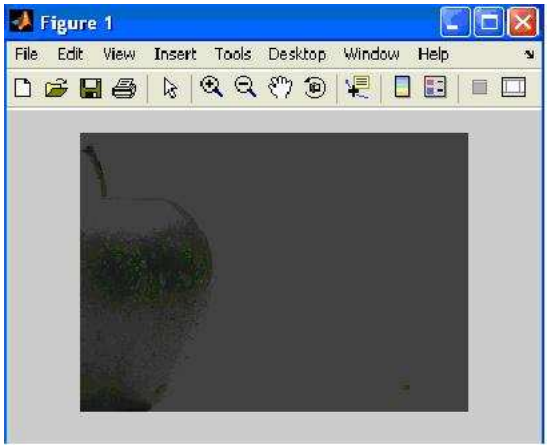
در این قسمت، هدف اصلی، تقسیم نظیر به نظیر تصاویر می باشد. نماد استفاده شده در این مرحله به صورت (./) می باشد که در آن تک تک پیکسل های هم شماره از هردو تصویر انتخابی به یکدیگر تقسیم شده و تصویر حاصل در خروجی نمایش داده می شود که از این حالت معمولاً جهت کاهش شدت روشنایی یک تصویر استفاده می شود.

علت کاهش روشنایی بدین دلیل است، که در تقسیم نظیر به نظیر تصاویر، ارزش تمام پیکسل های هم شماره از هر تصویر به یکدیگر تقسیم شده و در نتیجه تصویر نهایی، تصویری با ارزش پیکسل کمتری خواهد بود.

تابع استفاده شده در این قسمت imdivide نام دارد.

مثال :در این مثال، هدف، انجام عملیات تقسیم بر روی تصویر اولیه مثال قبل می باشد، که برای انجام این کار به صورت زیر عمل می شود.

```
s=imdivide(i,8) ;
imshow(s)
```



تصویر۱۵-۳-۲

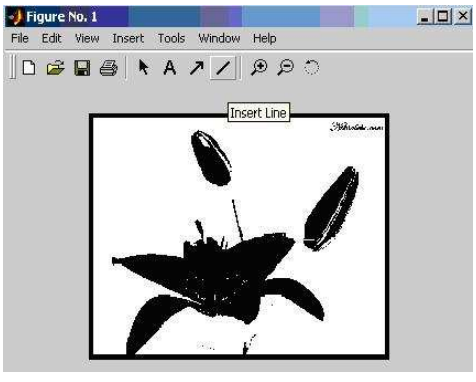
همان گونه که در تصویر۱۵-۳-۲مشاهده می گردد، تقسیم ارزش عددی ۸ به پیکسل های تصویر باعث کاهش مقدار روشنایی تصویر اولیه شده است و تصویر نهایی نسبت به تصویر اولیه از روشنایی کمتری برخوردارد است.

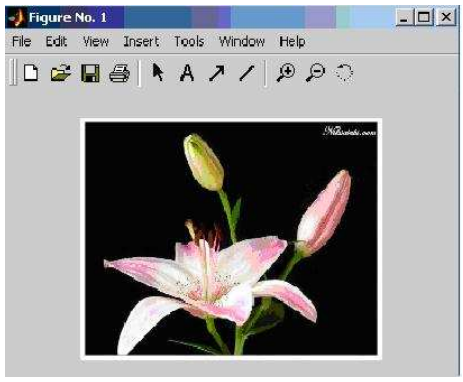
تغییر نمایش تصاویر رنگی به فرمت باینری در نرم افزار Matlab

جهت معکوس کردن رنگ های سیاه و سفید یک تصویر باینری، کافی است آن تصویر را Not کرد.

مثال :برای Not کردن تصویر ۱۶-۳ بدین صورت عمل می شود.

```
i=imread('flower4.tif');
imshow(i)
BW=rgb2gray(i)
imshow (~ BW)
```

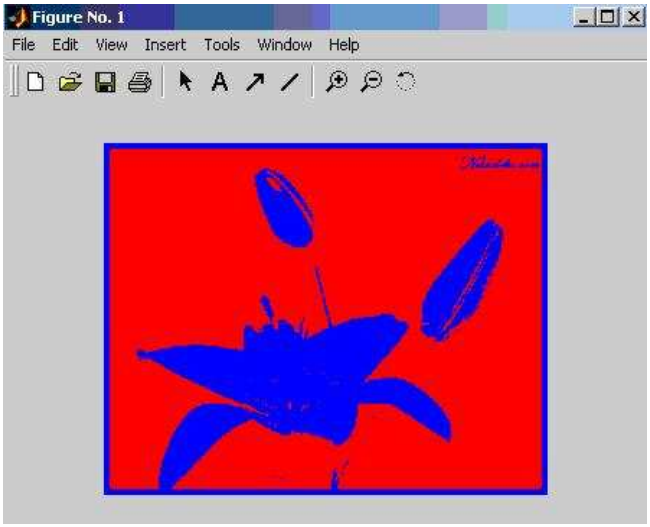




عملیات دیگری که توسط نرم افزار قدرتمند مطلب می توان انجام داد، رنگی نمودن تصاویر باینری می باشد.

دستور زیر پیکسل های صفر را به قرمز و پیکسل های شامل یک را به رنگ آبی تبدیل می کند.

Imshow (BW,[100,001])



تصویر۱۸-۳-۲

با تغییر صفرها و یک ها، می توان رنگ های دیگر را نیز به دست آورد.

[010] = سبز ، [1 0 1] = بنفش، [0 0 0] = سیاه ، [1 1 1] = سفید، [0 1 1] = زرد

[001] = آبی کم رنگ ، [0.5 0.5 0.5] = طوسی ، [0.5 0 0] = قرمز تیره ، [1 1 1] = سفید، [0.40 0.62 1] = نارنجی

[0.49 1 0.83] = لیمویی

نمایش تصاویر رنگی

نام دیگر این نوع تصاویر، truecolor است و syntax کلی نمایش رنگی به صورت زیر می باشد:

imshow (RGB)

RGB به صورت یک آرایهٔ سه بعدی به صورت (m-by-n-by-3) می باشد، که مختصات پیکسل ها به صورت (r,c) نمایش داده می شوند. تابع imshow در واقع رنگ ها را در سه بیت یا سه جز نمایش می دهد (r,c,1:3)

سیستم هایی که 24 پیکسل بر بیت در صفحه اند، می توانند RGB را نشان دهند، زیرا آنها 8 بیت (256 حالت) برای هر کدام از رنگ ها سبز و آبی تخصیص می دهند.در سیستم هایی که کمتر از 24 پیکسل در صفحه اند، Matlab رنگ های مشابه و نزدیک به آنها را با ترکیب کردن رنگ ها نشان می دهد.مقدار پیش فرض تعداد پیکسل ها 24 عدد در یک بیت است، که می توان 256 حالت رنگی از آن به دست آورد و همچنین این امکان وجود دارد، که عدد 24 را به 32 و 64 تغییر داد.بدین صورت:

imshow (i , 32) OR imshow (I, 64)

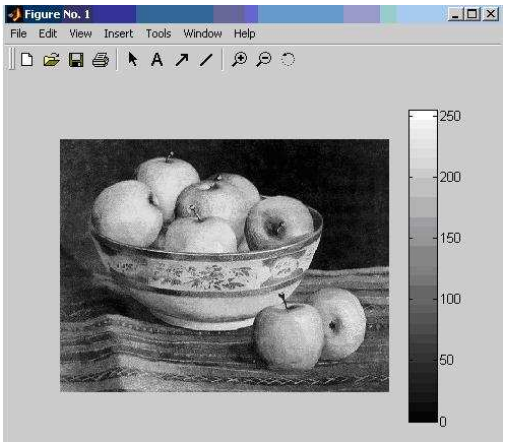
افزودن نوار رنگی

از تابع colorbar جهت افزودن نوار رنگی به کناره تصاویر استفاده می شود، که در آن ارزش داده های تصویر و رنگ ها توسط colorbar تطبیق داده می شود.

مثال :در مثال زیر، افزودن نوار رنگی به یک تصویر Grayscale با کلاس 8 فیلتر شده را بررسی می کنیم.

(مبحث مربوط به فیلتر و چگونگی کاربرد آنها در مباحث بعد توضیح داده خواهد شد.)

```
i=imread( 'apple.tif') ;
imshow(i)

imshow(i2),colorbar
```



تصویر۱۹-۳-۲

یافتن اطلاعات رنگی دربارهٔ هر پیکسل از تصویر

معمولاً برای تعیین مقادیر رنگی مربوط به یک نقطه از تصویر، از تابعی به نام impixel استفاده می شود. نحوهٔ کارکرد این تابع بدین صورت خواهد بود، که پس از فراخوانی تصویر دلخواه با صدا زدن تابع impixel به کمک موس، نقاطی از تصویر را انتخاب کرده و بازدن کلید enter، مقادیر رنگی مربوط به R و G و B پیکسل آن نقطه درخروجی دریافت خواهد شد.

در واقع توسط این تابع می توان ارزش رنگی هر پیکسل از تصویر را به دست آورد.

Syntax های مربوط به این تابع در زیر آورده شده است:

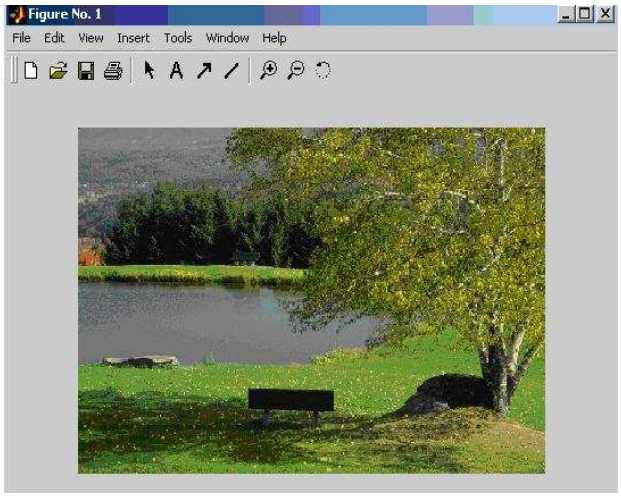
```
P=impixel(I)
P=impixel(X,map)
P=impixel(RGB)
P=impixel(I,c,r)
(X,map,c,r)P=impixel
P=impixel(RGB,c,r)
[c,r,P]=impixel(…)
P=impixel(x,y,I,xi,yi)
P=impixel(x,y,X,map,xi,yi)
P=impixel(x,y,RGB,xi,yi)
[xi,yi,P]=impixel(x,y,…)
```

مثال

تصویری به نام trees را فراخوانی کرده و به کمک تابع impixel، مقادیر رنگ مربوط به چند نقطهٔ دلخواه را به دست آورید.

برای انجام این کار بدین صورت عمل می شود:

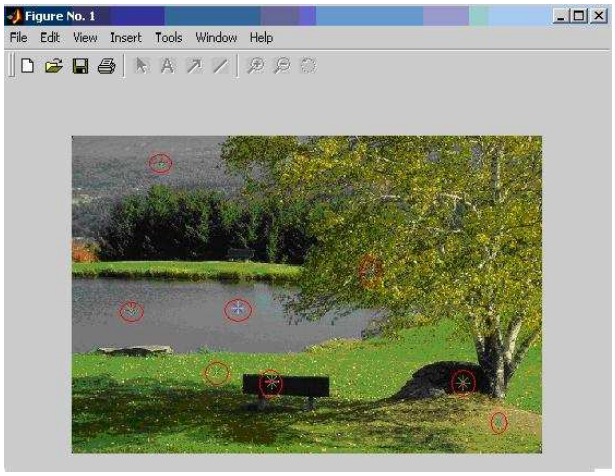
```
i=imread('trees.png');
imshow(i)
```



تصویر ۲۰-۳-۲

```
impixel (i)
```

نقاط انتخاب شده توسط موس، در تصویر زیر مشاهده می گردد و ماتریس مشاهده شده در مرحلهٔ بعد در واقع به ترتیب، ارزش رنگی هر نقطه نسبت به رنگ های قرمز، سبز و آبی می باشد.



تصویر ۲۱-۳-۲

```
129   131   101
10    18    11
20    68    72
112   111   85
123   211   195
136   141   114
10    17    14
81    149   162
```

در صورتی که تصویر اصلی به صورت سیاه – سفید باشد و تابع impixel بر روی آن تصویر اجرا شود، مقادیر R و G و B مربوط به رنگ سیاه هر سه مقدار عددی صفر و برای رنگ سفید مقدار عددی ۲۵۵ را نمایش می دهد.

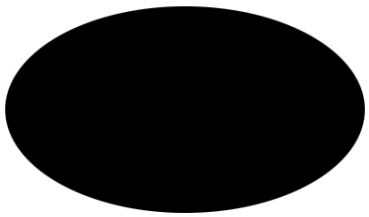
برای درک بهتر این مطلب، به مثال زیر توجه شود.

مثال

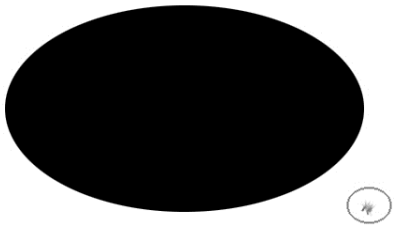
پس از فراخوانی تصویر مذکور، که تنها شامل رنگ سیاه و سفید است، تابع impixel بر روی آن اعمال می شود و نتیجهٔ مربوطه بررسی خواهد شد.

I=imread(‘image.bmp’);

Imshow(i)



Impixel(i)



0 0 0

255 255 255

همان گونه که مشاهده می شود، ارزش رنگی مربوط به نقطهٔ سیاه، عدد 0 و برای نقاط سفید عدد 255 می باشد.

اضافه نمودن noise به تصاویر در نرم Matlab

برای افزودن noise به یک تصویر، از تابعی imnoise استفاده می شود.

به طور کلی noise های کاربردی برای تصاویر به پنج دسته تقسیم بندی می شوند، که شرح مربوط به هریک، در جدول زیر آورده شده است.

Value	**Description**
‘gaussian’	Gaussian white noise with constant mean and variance
‘Localvar’	Zero-mean Gaussian white an intensity-dependent variance
‘Poisson’	Poisson noise
‘salt & pepper’	On &off pixel
‘speckle’	Multiplictive noise

برای آشنایی و درک تفاوت انواع مختلف noiseها، هر یک از noise های بالا بر روی یک تصویر آزمایش و بررسی می شود.

قابل ذکر است، که فرمت کلی استفاده از دستورات مربوط به افزودن noise به تصاویر، به صورت زیر می باشد:

J = imnoise (I, type , parameters)

در فرمت بالا type، در واقع نوع noise اضافه شده به تصویر را بیان می کند و parameters عددی است که نشانگر میزان درصد noise اضافی به تصویر می باشد، که هرچه این مقدار عدد بزرگ تر باشد، مقدار noise اضافه شده به تصویر نیز بیشتر خواهد بود.

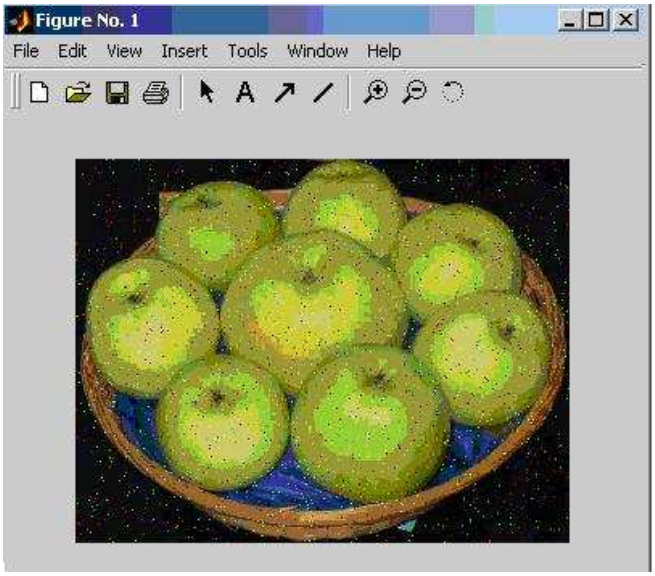
مثال

فرض شود تصویری به نام apple را فراخوانی کرده و انواع noiseها بر روی آن بررسی شده است. نتیجهٔ حاصل به صورت زیر می باشد.

```
i = imread('apple.tif ') ;
imshow(i)
```

تصویر۲۴-۳-۲

```
j = imnoise(I,'salt & pepper',0.02 );
imshow(j)
```



تصویر۲۵-۳-۲

همان گونه که در تصویر ۲۵-۳ مشخص است، با افزودن salt&pepper noise تصویر حاصل به صورت بالا در آمده است.

حال همان تصویر اولیهٔ این بار با افزودن Gaussian noise بررسی می شود.

```
j = imnoise(I,' gaussian',0.02 );
imshow(j)
```

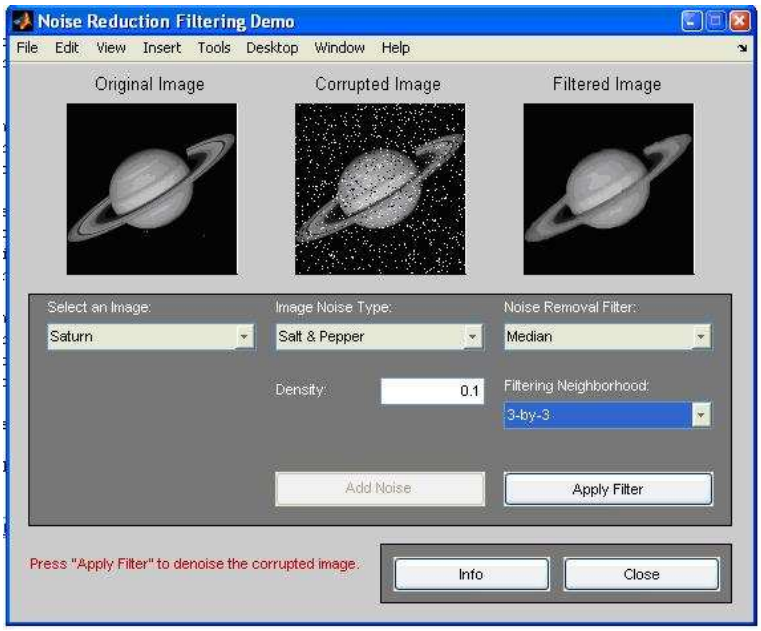
تصویر۲۶-۳-۲

تصویر اصلی را این بار با افزودن Speckle noise آزمایش کرده و تصویر حاصل را مشاهده کنید.

```
j = imnoise(I,'speckle',0.02 );
Imshow(j)
```

تصویر۲۷-۳-۲

برای مشاهدهٔ مثال های بیشتر در این زمینه، به Demo مربوط به مبحث نویزها در Image Processing Toolbox مراجعه شود.



تصویر۲۸-۳-۲

با تغییر نوع از قسمت select an image و تغییر نوع noise دلخواه توسط قسمت image noisetype می توان تأثیر noise دلخواه را بر روی تصاویر متعدد مشاهده نمود.

حذف noise موجود بر روی تصاویر در نرم افزار Matlab

همان گونه که در قسمت قبل بیان شد، به کمک تابع imnoise و پارامترهای مربوط به آن می توان به یک تصویر noise اضافه نمود.

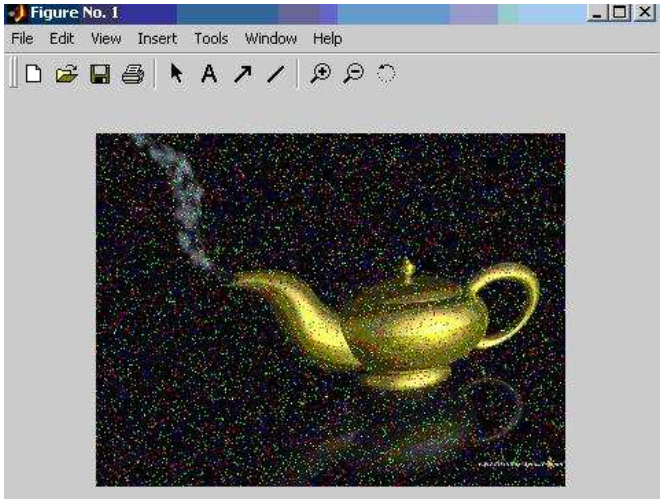
حال که نحوهٔ افزودن نویز به تصاویر را فرا گرفتیم، نوبت آن است که چگونگی حذف نویزهای اضافه شده به تصاویر را بررسی کنیم.

توسط تابع medfilt2 ، می توان نویز اضافه شده به یک تصویر را حذف نمود.

مثال

تصویری به نام eight فراخوانی و یکی از نوع noise ها را به آن اضافه کرده و سپس توسط تابع medfilt2 نویز اضافه به تصویر را حذف کنید.

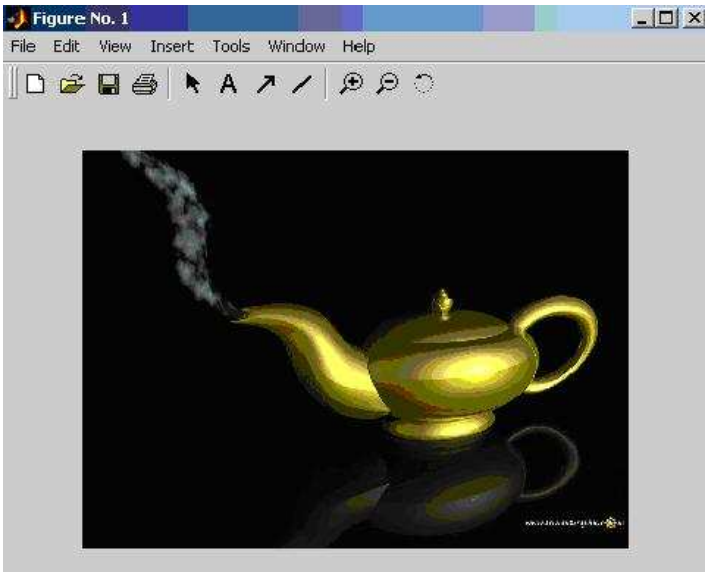
```
i = imread('eight.tif') ;
j = imnoise(i,'salt & pepper',0.02);
imshow(j)
```



تصویر۲۹-۳-۲

حال جهت حذف noise موجود در تصویر بالا، از تابع medfilt2 استفاده می شود.

```
k = medfilt2(j);
imshow(k)
```



تصویر ۳۰-۳-۲

همان گونه که در تصویر ۳۰-۳-۲ مشاهده می شود، پس از استفاده از تابع medfilt2، noise موجود روی تصویر به طور کامل حذف شده است.

از این تابع می توان برای حذف انواع noise های موجود در تصاویر استفاده نمود.

عملیات لبه برداری بر روی تصاویر در نرم افزار Matlab

آشکار سازی لبه (Edge Detection)، یکی از عملیات مهم و پرکاربرد در علم پردازش تصویر می باشد و معمولاً برای تشخیص لبه های یک شی از بین چند شی دیگر مورد استفاده قرار می گیرد، که برای انجام این کار از تابعی به نام edge استفاده می شود.

تغییرات فیزیکی به صورت تغییر رنگ و تغییر شدت روشنایی به صورت لبه در تصویر نمایان می شوند. در محیط با مقادیر پیوسته، مشتق، تغییرات ناگهانی و شدت آن را مشخص می کند.

در محیط گسسته محاسبه تغییرات نسبت به پیکسل های مجاور، تقریبی، از مشتق را نمایان می سازد. در محاسبه بردارگردایان، دو مؤلفهٔ اندازه و جهت بسیار مهم می باشند.

در عملیات لبه برداری ورودی یک تصویر به فرمت intensity می باشد و در خروجی تصویر Binary داده می شود، که در تصویر حاصل مرزهای بیرونی تصویر به صورت 1 و مرزهای داخل به صورت 0 نشان داده می شوند.

الگوریتم های لبه برداری

به طور کلی عملیات لبه برداری به چند الگوریتم صورت می پذیرد، کهدر زیر به شرح هر یک از آنها پرداخته شده است:

۱- الگوریتم Soble

۲- الگوریتم Canny

۳- الگوریتم Roberts

۴- الگوریتم Prewitt

۵- الگوریتم Laplacian of Gaussian

۶- الگوریتم Zero- Cross

در ابتدا جهت آشنایی بیشتر با هریک از انواع الگوریتم های بالا، syntax های مربوط به هریک بیان شده است و در ادامه، مثال مربوط به استفاده هر یک از الگوریتم های ذکر شده، بررسی خواهد شد:

```
BW=edge(I,'sobel')
BW=edge(I,''sobel',thresh)
BW=edge(I,''sobel',thresh,direction)

BW=edge(I,'prewitt')
BW=edge(I,'prewitt',thresh)
BW=edge(I,'prewitt',thresh,direction)
[BW,thresh]=edge(I,'prewitt',…)

BW=edge(I,'roberts')
BW=edge(I,'roberts',thresh)
[BW,thresh]=edge(I,'robert',…)

BW=edge(I,'log')
BW=edge(I,'log',thresh)
BW=edge(I,'log',thresh,sigma)
[BW,threshold]=edge(I,'log',…)

BW=edge(I,'zerocross',thresh,h)
BW=edge(I,'zerocross',…)
BW=edge(I,'canny')
BW=edge(I,'canny',thresh)
```

BW=edge(I,’canny’,thresh,sigma)

[BW,threshold]=edge(I,’canny’,…)

مثال

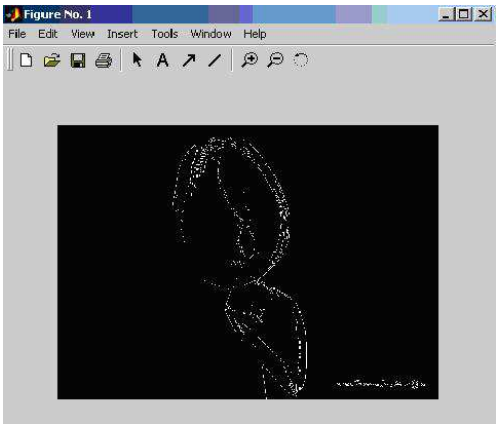
در این مثال تصویری به نام flower2.png را فراخوانی کرده و سپس نتیجهٔ استفاده هر یک از متدهای لبه برداری بررسی شود.

```
i=imread('flower2.png');
imshow(i)
```

تصویر ۳۱-۳-۲

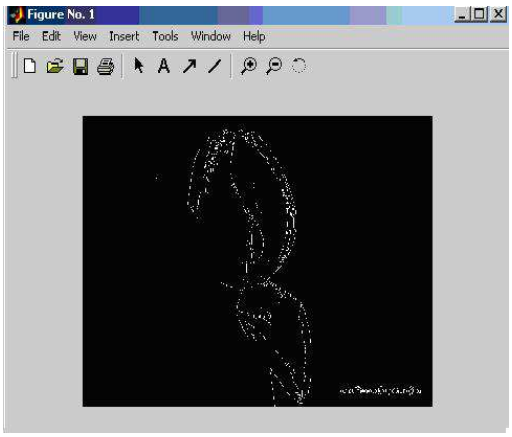
تصویر اصلی

```
R=rgb2gray(i)
BW1 = edge(R,'prewitt');
figure, imshow(BW1)
```



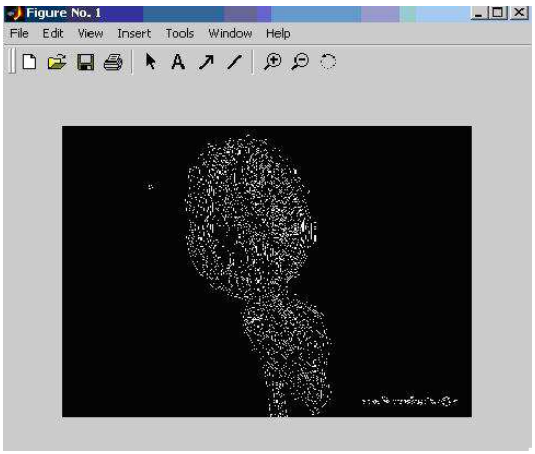
تصویر ۳۲-۳-۲

```
BW2 = edge(i,'sobel');
figure,imshow(BW2)
```



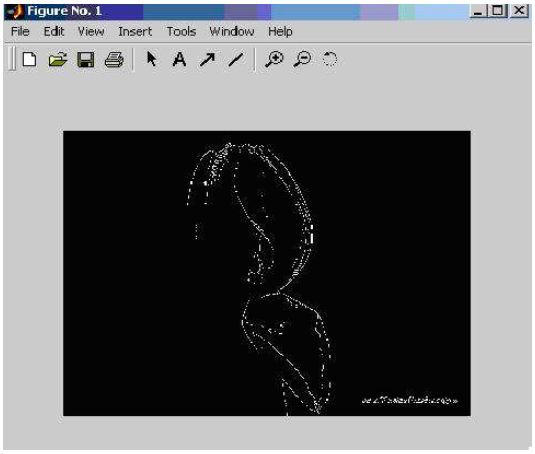
تصویر33-۳-۲

```
BW3 = edge(i,'canny');
figure,imshow(BW3)
```



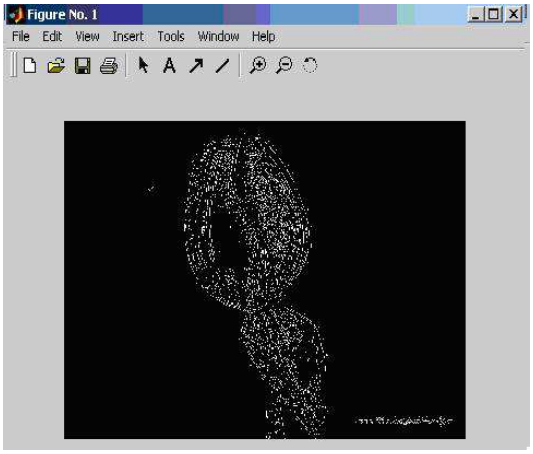
تصویر34-۳-۲

```
BW4 = edge(i, 'roberts ');
figure,imshow(BW4)
```



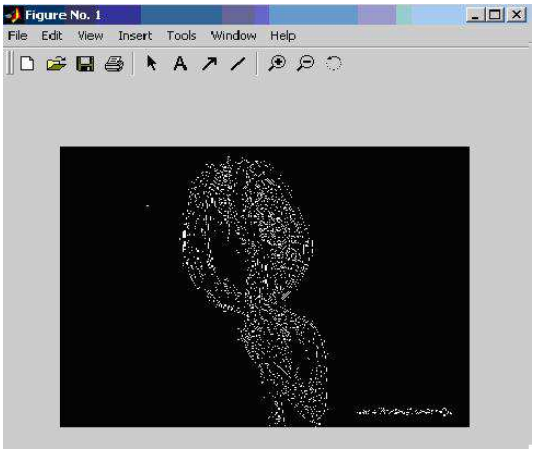
تصویر35-۳-۲

```
BW5 = edge(i, ' zerocross ');
figure,imshow(BW5)
```



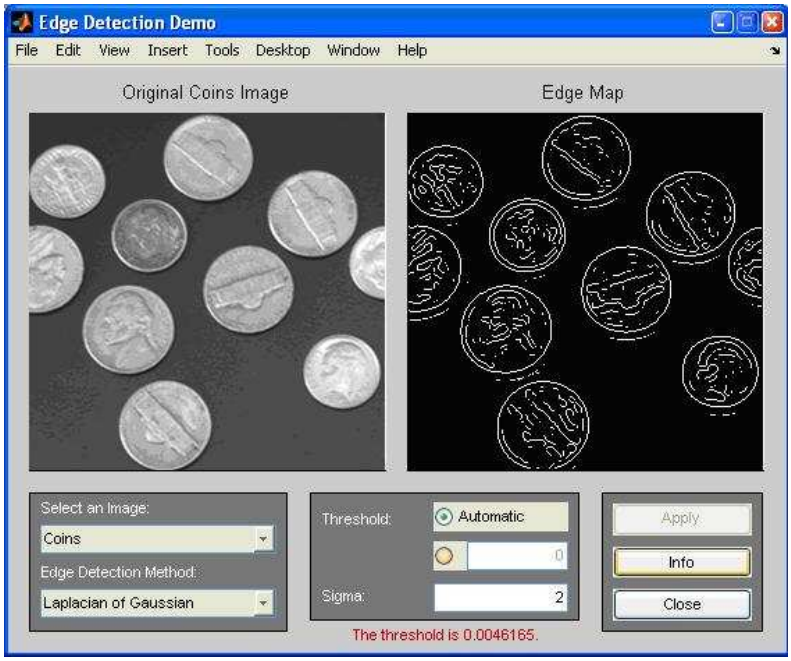
تصویر36-۳-۲

```
BW6 = edge(i,' log ');
figure,imshow(BW6)
```



تصویر37-۳-۲

جهت مشاهدهٔ مثالهای بیشتر در زمینهٔ لبه برداری تصاویر، به Demo مربوط به این قسمت در نرم افزار Matlab مراجعه شود . جهت مشاهدهٔ بیشتر در زمینهٔ لبه برداری تصاویر، به Demo مربوط به این قسمت در نرم افزار Matlab مراجعه شود.



تصویر38-۳-۲

توجه شود که الگوریتم ها و متدهای دیگری نیز در زمینه لبه برداری ابداع شده است که به دلیل ازدیاد مطالب از ذکر آنها در این کتاب خودداری شده است که می توان جهت بررسی هر یک به مقالات موجود در این زمینه مراجعه شود.

حذف ناحیهٔ دلخواه از تصاویر در نرم افزار Matlab

تابع roifill ازجمله توابع پرکاربرد و مفید در علم پردازش تصویر می باشد، که از این تابع جهت حذف ناحیهٔ زائد از تصویر به صورت دلخواه استفاده می شود.

به طور کلی به دو روش می توان از این تابع استفاده نمود:

روش اول: به صورت دستی و به کمک موس

روش دوم: انتخاب محدودهٔ دلخواه توسط مقادیر سطر و ستون

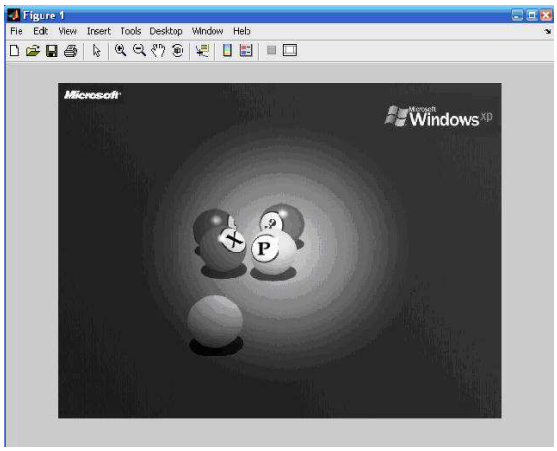
روش اول

در این روش به کمک این تابع، قسمت دلخواه از تصویر اصلی را محو کرد.

نحوهٔ کارکرد این تابع بدین صورت است، که پس از فراخوانی تصویر اصلی با اجرای تابع roifill و انتخاب محدودهٔ دلخواه توسط رسم یک یا چند ضلعی به کمک mouse، محدودهٔ موردنظر را تعیین کرده و تصویر نهایی که قسمت انتخاب شده توسط mouse از ان ناحیه محو شده است، مشاهده می شود.

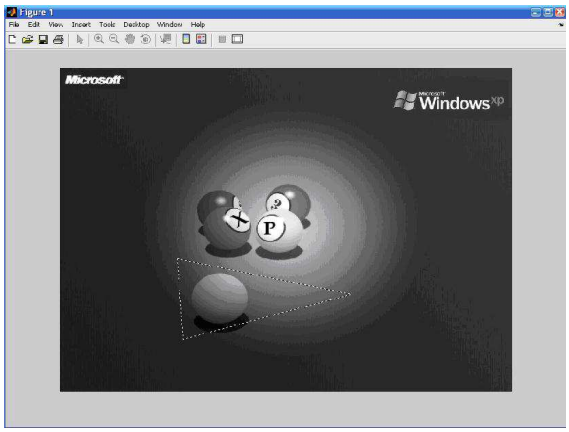
مثال:در این مثال تصویری به نام game را فراخوانی کرده و تغییرات دلخواه به کمک دستور roifill بر روی آن اعمال می شود.

```
i=imread('game.tif');
imshow(i)
roifill(i)
```



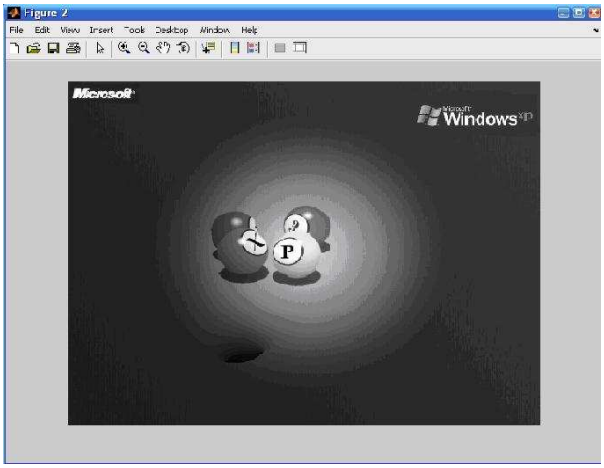
تصویر39-۳-۲

در تصویر ۴۰-۳-2، توسط mouse و رسم یک سه ضلعی محدودهٔ موردنظر و دلخواه انتخاب شده است.



تصویر40-۳-۲

همان گونه که در تصویر بعد به وضوح مشاهده می شود، توپ انتخاب شده از تصویر اصلی حذف شده است:



تصویر41-۳-۲

تذکر:

۱. تصویری که تابع roifill بر روی آن اعمال می شود، حتماً باید به فرمت GrayScale باشد. دستورات تبدیل انواع فرمت تصاویر به یکدیگر در فصول قبل بیان شده‌است.

۲. هنگام استفاده از روش اول، یعنی انتخاب محدودهٔ دلخواه توسط رسم چند ضلعی به کمک mouse، در صورتی که چند ضلعی انتخاب شده مناسب نبوده برای انتخاب محدودهٔ دیگر و حذف محدودهٔ انتخاب شدهٔ قبلی، از کلیدهای Delet و یا Back Space استفاده می شود.

۳. پس از انتخاب محدودهٔ دلخواه توسط رسم چند ضلعی به کمک right click یا double click، انتخاب خود را تثبت کرده و تصویر حاصل شده مشاهده می گردد.

روش دوم

دومین روش استفاده از تابع roifill بدین صورت می باشد، که تصویر موردنظر را فرخوانی کرده و سپس شمارهٔ ردیف و ستون های محدوده ای که باید از تصویر اصلی محو شود، به فرمت کلی زیر نوشته می شود.

```
Row=[a,b,c,d,…]
Coloum=[t,y,u,l,…]
J=roifill(TheName ofImage,Row,Coloum);
```

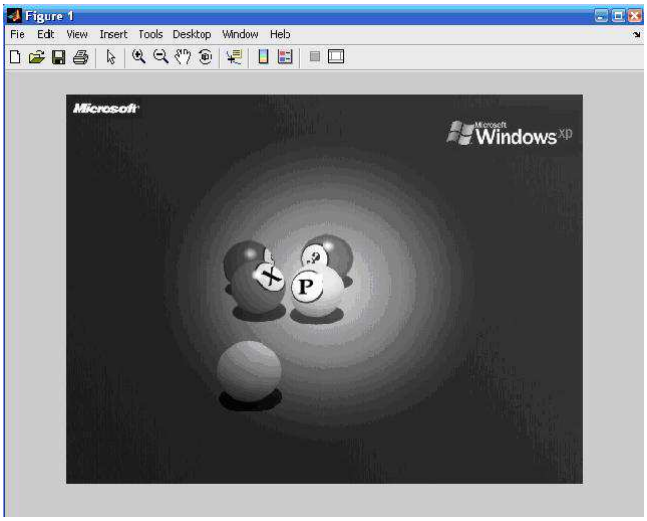
در انتها توسط دستور imshow، نتیجهٔ کاربررسی می شود.

جهت آشنایی بیشتر با نحوهٔ کاربرد روش دوم استفاده از این تابع به مثال زیر توجه شود.

مثال

هدف، حذف محدودهٔ دلخواه از تصویر مثال قبل به روش تعیین ماتریس آن محدوده می باشد.

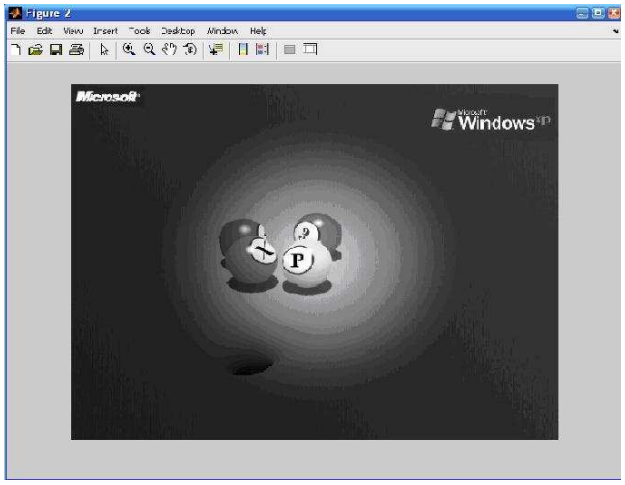
```
i=imread('game.tif');
imshow(i)
```



تصویر42-۳-۲

در این قسمت، شمارهٔ ردیف و ستون‌های ماتریس محدوده ای از تصویر، که قصد حذف آن محدوده از تصویر اولیه را داریم، مشخص شده است.

```
c = [222 272 300 270 221 194 ] ;
r = [21 21 75 121 121 75 ] ;
j = roifill (i,c,r );
imshow(j)
```



تصویر43-۳-۲

همان گونه که مشاهده می شود، با دادن سطر و ستون محدودهٔ دلخواه و استفاده از دستور roifill، توپ انتخاب شده از تصویر اولیه، به طور کامل محو شده‌است.

حذف ناحیهٔ دلخواه از تصویر به کمک تابع roipoly

به طور کلی به دو روش می توان از این تابع استفاده کرد:

روش اول: به صورت دستی و با استفاده از موس

روش دوم: انتخاب محدودهٔ دلخواه توسط مقادیر مناسب سطر و ستون

روش اول

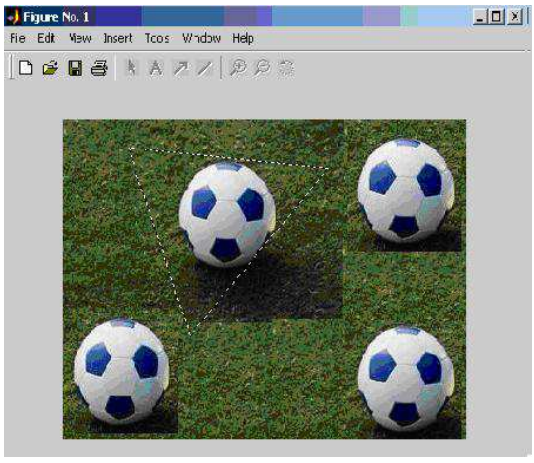
نحوۀ کارکرد این تابع، تا حدی شبیه تابع roifill می باشد. تنها تفاوت میان این دو تابع در این است که در تابع roifill محدوده انتخاب شده از تصویر محو می شود، بدون اینکه تغییری در سایر قسمت های تصویر اصلی ایجاد شود، اما تابع roipoly در واقع محدودۀ انتخاب شده را به صورت 1 (رنگ سفید) کنار گذاشته و بقیۀ محدودۀ تصویر را به صورت 0 (رنگ سیاه) نمایش می دهد. برای آشنایی بیشتر با نحوۀ کارکرد این تابع به مثال زیر توجه شود.

مثال :مشاهده و بررسی کارکرد تابع roipoly برروی تصویر football

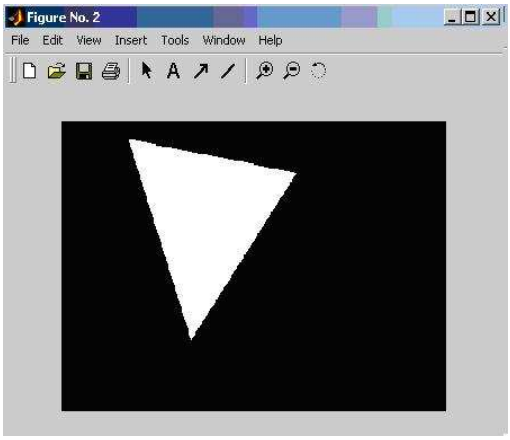
```
i=imread('football.tif');
imshow(i)

roipoly(i)
```

تصویر44-۳-۲

در این قسمت پس از اجرای تابع، محدودۀ دلخواه با رسم چند ضلعی توسط mouse انتخاب می گردد.



تصویر45-۳-۲



تصویر حاصل پس از اجرای تابع roipoly و انتخاب محدودۀ دلخواه به صورت تصویر ۴۶-۳-2می باشد. همانگونه که مشاهده می گردد، با اعمال تابع roipoly، محدودۀ انتخاب شده به صورت یک و بقیۀ محدودۀ تصویر، به صورت صفر نمایش داده شده است.

روش دوم

روش دوم استفاده از تابع roipoly، بدین صورت می باشد، که تصویر موردنظر را فراخوانی کرده و سپس شمارهٔ ردیف و ستون های مورد نظر برای حذف از تصویر اصلی، به فرم کلی زیر نوشته می شود. BW = roipoly (I, c, r)

برای آشنایی بیشتر با نحوهٔ کاربرد روش دوم استفاده از این تابع، به مثل زیر توجه شود.

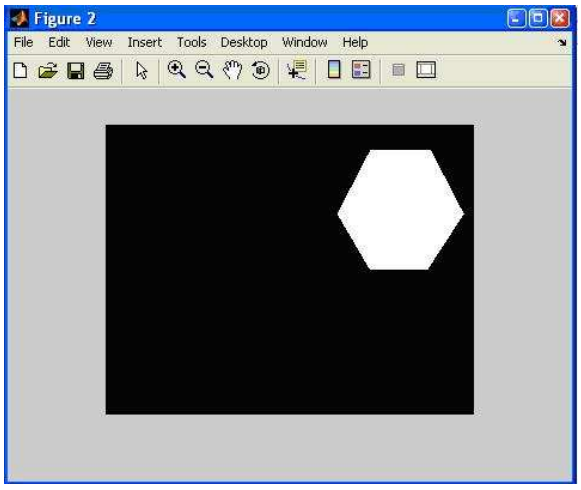
مثال

تصویر مثال قبل، این بار با انتخاب محدودهٔ دلخواه و تعیین شماره های سطر و ستون مناسب محدودهٔ موردنظر بررسی شود.

```
i = imread('football.tif');
c = [222 272 300 270 221 194];
r = [21 21 75 121 121 75];
BW = roipoly(i,c,r);
imshow(i)
figure, imshow(BW)
```

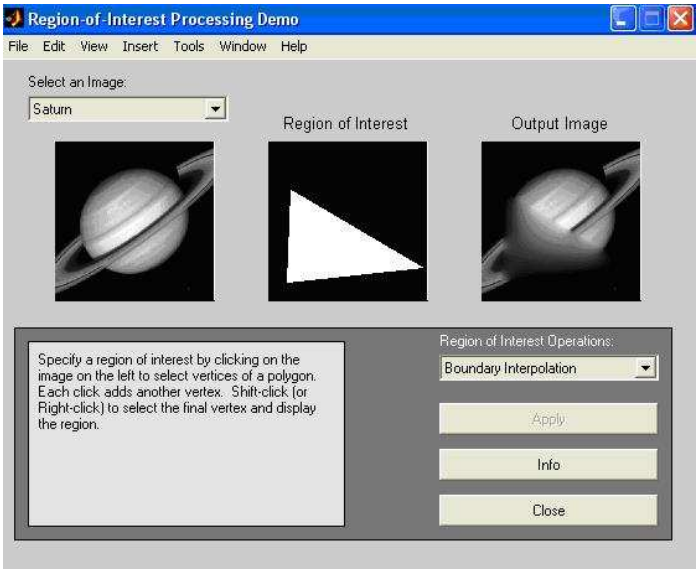
تصویر47-۳-۲

پس از اعمال سطر و ستون ماتریس دلخواه از تصویر اصلِی، نتیجه به صورت تصویر ۴۸-۳ -2خواهد شد.



تصویر48-۳-۲

جهت مشاهدهٔ مثال های بیشتر در زمینهٔ حذف ناحیهٔ دلخواه از تصاویر به Demo مربوط به این قسمت در نرم افزار Matlab مراجعه شود.



پرکردن ناحیهٔ دلخواه از تصویر در نرم افزار Matlab

در برخی موارد، جهت پردازش بهتر بر روی تصاویر وافزایش عملیات پردازش، لازم است که نواحی و حفره های موجود در تصویر را پر کرده و هم سطح محدودهٔ اصلی تصویر کرد، به کمک تابع bwfill، می توان ناحیهٔ Backgroud تصاویر به فرمت Binary پرکرد.

تابع bwfill در واقع یک تابع پدربزرگ (grandfather) یا ریشه نامیده می شود و در واقع جایگزینی مناسب تابع imfill می باشد.

BW2 = bwfill (bwl, R, C, N)

شروع عملکرد تابع bwfill از پیکسل (R,C) می باشد و در صورتی که R,C بردارهایی با طول مساوی باشند، عمل پرکردن به صورت موازی انجام خواهد گرفت.

(R (k), C (k))

مقدار عددی N می تواند به صورت 4 یا 8 باشد که تعیین کننده connectivity می باشد.

زمانی که عدد چهار استفاده می شود، چهار اتصال در تصویر Foreground وجود دارد و در صورتی که عدد هشت استفاده شود، هشت اتصال در تصویر Foreground وجود دارد.

BW2 = bwfill (BW 1, N)

این دستور تصویر را در صفحه نمایش نشان می دهد و اجازه می دهد که نقطهٔ دلخواه به کمک mouse تعیین شود.

نحوهٔ عملکرد این تابع بدین ترتیب است، که تابع bwfill بر روی تصاویر برداری جاری عمل می کند، بدین صورت که توسط کلیک mouse آن قسمت تعیین شده و پس از اعمال تابع bwfill، محدودهٔ انتخاب شده، توسط mouse پر می شود.

تذکرات

۱. فشار دادن کلیدهای Backspace و یا delete باعث حذف نقاط انتخاب شده، توسط mouse می گردد.

۲. Shift click و Right click و double click نقطهٔ پایانی را انتخاب کرده و آنگاه عمل پرکردن آغاز می شود و با فشردن کلید enter عمل انتخاب به پایان می رسد.

[BW2 , ID] = bwfill (….)

این دستور مشخصات خطی تمام پیکسل های پرشده توسط این تابع را بر می گرداند.

BW2 = bwfill (BW1 , 'holes ' N)

تابع bwfill در واقع تعیین می کند، که کدام پیکسل جز حفره ها باشد و آنگاه ارزش عددی پیکسل های حفره را از صفر به یک تغییر می دهد.

قابل ذکر است که پیش فرض مقدار N عدد 8 است.

[BW2, [Dx] bwfill (BW1 , ' holes' , N)

این دستور در واقع مشخصات خطی تمام پیکسل های پرشده در bwfill را بر می گرداند.

اگر از تابع bwfill برای آرگومان های خروجی استفاده شود، تصویر خروجی در figure جداگانه نمایش داده می شود.

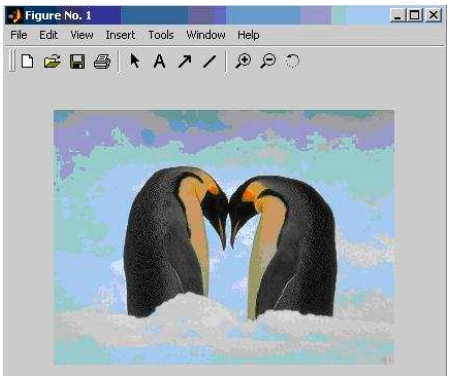
کلاس های پشتیبانی شده توسط این تابع

تصویر ورودی حتماً باید به فرمت عددی یا منطقی باشد و تصویر خروجی به فرمت منطقی خواهد بود.

مثال

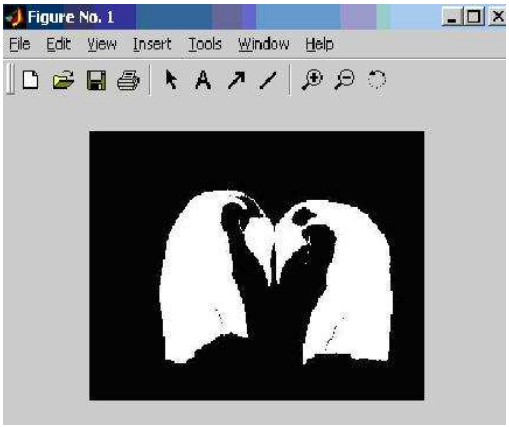
جهت درک بهتر توضیحات مطرح شده، تابع bwfill برای تصویر زیر اجرا می شود.

```
i=imread('pan.png');
imshow(i)
```



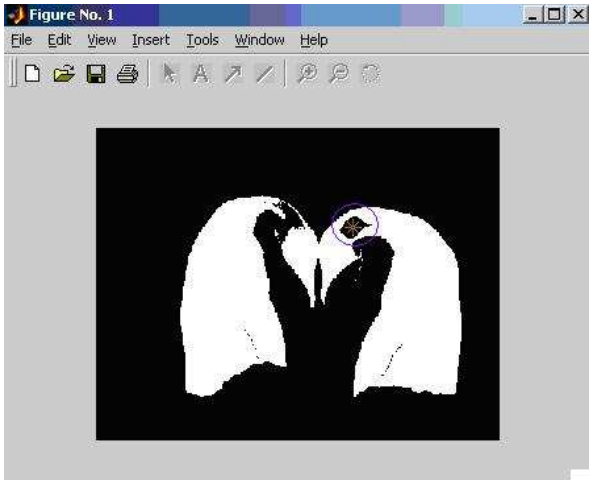
تصویر49-۳-۲

```
bw=im2bw(i)
```

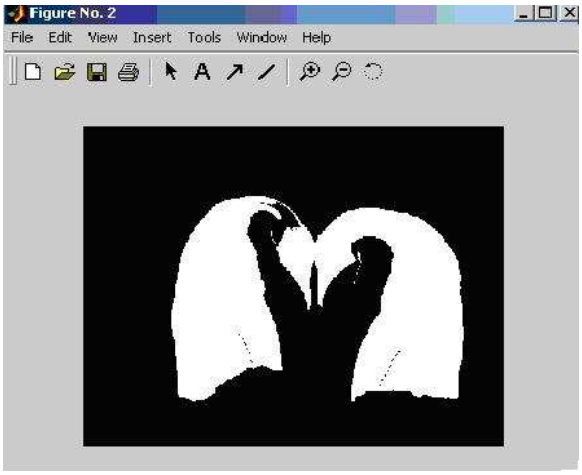


تصویر50-۳-۲

bwfill(bw)



تصویر51-۳-۲ (انتخاب با mouse)



تصویر۵۲-۳-۲ (حاصل از تابع bwfill)

تصویر شماره ۵۰-۳ -2همان تصویر اصلی فراخوانی شده می باشد، تصویر شمارهٔ ۵۱-۳ ۲توسط mouse، محدوده ای که قصد پرکردن آن را داریم تعیین کرده و در شکل شمارهٔ ۵۲-۳-2، نتیجهٔ حاصل که محدودهٔ انتخاب شدهٔ پر شده است، نمایش داده می شود.

برای پر کردن حفره های مربوط به تصاویر رنگی، ازتابعی به نام imfill استفاده می شود.

جهت دریافت اطلاعات بیشتر در مورد این تابع در محیط مطلب، دستور زیر اجرا شود.

Help ('imfill')

فیلتر کردن تصاویر در نرم افزار Matlab

فیلتر کردن تصاویر از موارد مهم و پرکاربرد درعلم پردازش تصویر می باشد. برای عملیات فیلتر کردن یک تصویر، از تابعی با عنوان imfilter استفاده می شود.

Syntax های مورد استفاده در این تابع قه صورت زیر می باشد.

B = imfilter (A, H)

B = imfilter (A, H, option 1, option 2)

توضیحات مربوط به B = imfilter(A,H)

این تابع آرایهٔ چند بعدی تصویر A را فیلتر می کند، که این کار توسط فیلتر چند بعدی H انجام می پذیرد. نتیجه آن که B و A از نظر اندازه و کلاس مشابه یکدیگرند.

توجه شود که برای هر المان از خروجی B دقت double نقاط اعشاری استفاده می شود.

اگر A آرایهٔ integer باشد، پس المان خروجی که اجرا می شود نیز ازنوع integer خواهد بود و مقدار اعشاری مربوط به آن نیز به صورت round می باشد.

توضیحات مربوط به B=imfilter(A,H,option 1 , option 2)

این قسمت فیلتر های چند بعدی مطابق با option ها ی تعیین شده را، نشان می دهد که آرگومان ها و option ها به صورت یکی از حالات زیر می باشد.

X: برای تعیین محدوده و مرز آرایه مورد استفاده قرار می گیرد و در صورتی که مقدار X معادل صفر باشد بدین معناست که مرزی تعیین نشده است.

Symmetric: جهت قرینه کردن مرزها با روش انعکاس آینه ای مورد استفاده واقع می شود.

Replicate: معادل نزدیک ترین ارزش محدودهٔ آرایه می باشد.

Circlure: این حالت به صورت حلقوی برای تعیین محدودهٔ آرایه ها مورد استفاده قرار می گیرد.

تنظیمات مربوط به آرایه های خروجی

آرایه های موجود در خروجی به صورت های مختلف تنظیم می شوند، که در زیر به بررسی هر یک پرداخته شده است.

Same: در این حالت، آرایه های ورودی و خروجی از نظر اندازه با یکدیگر مساوی و برابر می شوند.

قابل ذکر است که به صورت پیش فرض، مقدار خروجی به صورت Same می باشد.

Full: در این حالت، آرایهٔ خروجی به صورت کامل فیلتر می شود و نسبت به آرایهٔ ورودی بزرگ تر خواهد بود.

مثال

بررسی کاربرد motion در تابع imfilter، تصویر رنگی را خوانده نمایش می دهد.

```
Original RGB=imread('peppers.png');
Imshow(originalRGB)
```

فیلتری به نام h با خاصیت motion(فیلتری شبیه حرکت دوربین به صورت خطی)، هر خط به طول 50 پیکسل با زاویه 45 درجه می سازد:

```
H=fspecial('motion',50,45);
```

این فیلتر، روی تصویر رنگی فراخوانی شده، اعمال می شود:

```
filteredRGB=imfilter(originalRGB,h);
figure,imshow(filteredRGB)
whos rgb2
```

Name	Size	BytesClass
H	37*37	10952 double array
Rgb	384*512*3	589824 uint8 array
Rgb2	384*512*3	589824 uint8 array

مثال

بررسی کاربرد replicate در تابع imfilter

مرزها را تکرار می کند (عملیات دوباره سازی):

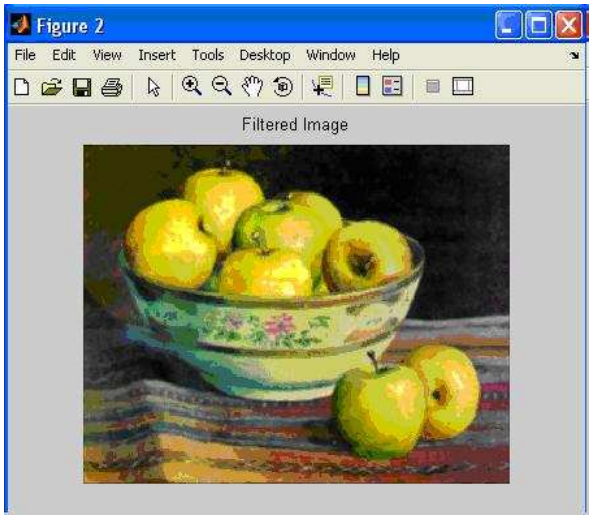
```
bounaryReplicateRGB=imfilter(originalRGB,h,’replicate’);

figure,imshow(boundary ReplicateRGB
```

مثال

بررسی کاربرد avergeing در تابع imfilter

```
I = imread('apples.png');
h = ones(5,5) / 25;
I2 = imfilter(I,h);
imshow(I), title('Original Image');
figure, imshow(I2), title('Filtered Image')
```

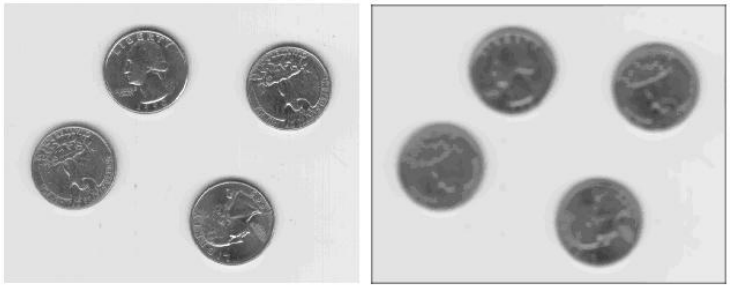
تصویر ۵۲-۳-۲



تصویر ۵۳-۳-۲

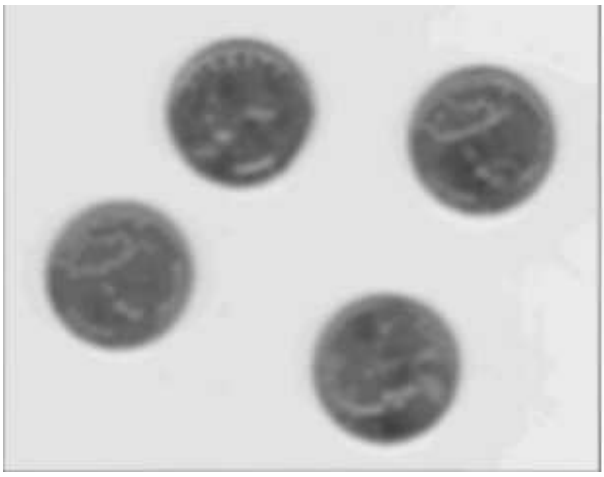
شکل زیر، نمونه ای از تصویر مربوط به فیلتر avergeing می باشد:

```
I = imread('apples.png');
h = ones(5,5) / 25;
I2 = imfilter(I,h);
imshow(I), title('Original Image');

figure, imshow(I2), title('Filtered Image')
```



تصویر۵۴-۳-۲

```
I3 = imfilter(I,h,'replicate');
figure, imshow(I3);
title('Filtered Image with Border Replication')
```



تصویر۵۵-۳-۲

طراحی فیلتر به کمک نرم افزار Matlab

معمولاً برای طراحی یک فیلتر در نرم افزار Matlab، از تابعی با عنوان fspecial استفاده می شود که فرم کلی استفاده از این تابع به صورت زیر می باشد:

h = fspecial ('motion, a, b),

در آن،a طول خطی که براثر حرکت دوربین شبیه سازی می شود و b درواقع مبنای درجه زاویه Tetha و درجه گردش خطوط می باشد.

Syntax های مربوط به دستور fspecial

H = fspecial (type)

H = fspecial (type, parametric)

درحالت اول، type شامل موارد زیر است:

Gaasian – sobel – prewi ++- Lapaeian- log- average – unshape

در حالت دوم، میانگین فیلتر ها را بر می گرداند و در متغیر h قرار می دهد، که با توجه به اندازهٔ تعیین شده در hsize(hsize در واقع تعداد اندازه بردارهایی است که به صورت سطر و ستون در h ریخته می شود) می تواند هم به صورت بردار ثابت و هم به صورت عدد تعیین گردد. قابل ذکر است که پیش فرض آن به صورت [33] می باشد.

Hsfspecial ('disk' , vadius)

این حالت، فیلتر میانگین برای دایرهٔ pillbox است که شامل ماتریس از 2*radius می باشد.

حالت پیش فرض دایره ای، به صورت ثابت، عدد ۵ است.

H= f special ('gaassian' , hsize , sigma)

در این حالت، یک فیلتر پایین گذر گوسی قرینه به صورت چرخشی را بر می گرداند، که اندازهٔ آن با hsize تعیین شده و از تقسیمات استاندارد sigma حاصل می گردد.

Hsize می تواند به صورت یک بردار باشد، که از تعدادی سطر و ستون در h معین می شود و یا می تواند به صورت مقدار عددی باشد که در برخی حالات h یک ماتریس مربع می باشد.

قابل ذکر است، که ارزش مقدار پیش فرض سیگما عدد ثابت 0.5 است.

H=fspecial(‘laplacian’,alpha)

این حالت در واقع یک فیلتر سه در سه را به صورت دو بعدی با پارامتر Alpha کنترل می کند.

پیش فرض مقدار Alpha عدد 0.2 است.

H=fspecial(‘log’,hsize,sigma)

نحوهٔ کارکرد این تابع، نیز به صورت مدل گوسین می باشد، البته تنها تفاوت مدل Log با مدل گوسین این است، که حالت نحوه کارکرد این تابع به صورت مدل گوسین می باشد، البته تنها تفاوت مدل Log با مدل گوسین این است، که حالت پیش فرض Log به صورت [5 5] می باشد. پیش فرض مدل Log به صورت [55] می باشد.

H=fspecial(‘motion’,len,tetha)

در این حالت، شکل فیلتر شبیه حرکت دوربین به صورت خطی می باشد، که طول این خط بستگی به تعداد پیکسل دارد و مقدار Tetha، تعیین کنندهٔ درجهٔ آن می باشد.

حالت پیش فرض ایت خط عدد 9 است.

H=fspecial(‘prewitt’)

این حالت در واقع یک فیلتر سه در سه در h بر می گرداند، که لبه های افقی را تقویت می کند.

[1 1 1

0 0 0

-1 -1 -1]

برای پیدا کردن لبه های عمودی یا مشتقات x(x-derivatives) از h استفاده می شود.

H=fspecial(‘sobel’)

فیلتری سه در سه به نام h برمی گرداند، که بر لبه های افقی تأکید دارد. برای تأکید، بر لبه های عمودی فیلتر h را جا به جا کند.

[1 2 1

0 0 0

-1 -2 -1]

H=fspecial(‘unsharp’,alpha)

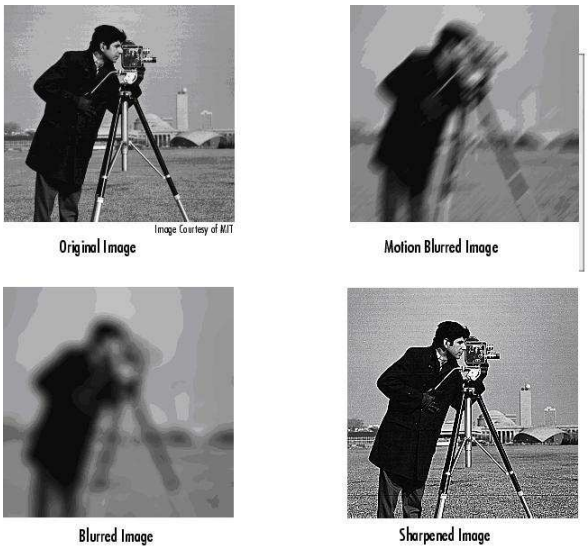
یک فیلتر سه در سه، افزایش دهندهٔ شدت نور unsharp تولید می کند، fspical یک فیلتر unsharp را ساخته، که دارای پارامتر منفی Alpha می باشد.

کلاس های پشتیبانی کننده

کلاس فیلتر h، از نوع Double می باشد.

مثال : اعمال چند نمونه فیلتر بر روی تصویر تعریف شده در Matlab به نام Cameraman

```
I = imread('cameraman.tif');
subplot(2,2,1);
imshow(I); title('Original Image');

H = fspecial('motion',20,45);
MotionBlur = imfilter(I,H,'replicate');
subplot(2,2,2);
imshow(MotionBlur);title('Motion Blurred Image');

H = fspecial('disk',10);
blurred = imfilter(I,H,'replicate');
subplot(2,2,3);
imshow(blurred); title('Blurred Image');

H = fspecial('unsharp');
sharpened = imfilter(I,H,'replicate');
subplot(2,2,4);
imshow(sharpened); title('Sharpened Image');
```



تصویر۵۶-۳-۲

بررسی تعدادی الگوریتم

تابع fspical فیلتر گوسی (Gaussian) را با الگوریتم زیر می سازد:

$$h_g(n_1,n_2)=e^{-\left(n_1^2+n_2^2\right)/\left(2\sigma^2\right)}$$

$$h(n_1,n_2)=\frac{h_g(n_1,n_2)}{\sum_{n_1}\sum_{n_2}h_g}$$

تابع fspical فیلتر Laplacian را با الگوریتم زیر می سازد.

$$\nabla^2=\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}$$

$$\nabla^2\approx\frac{4}{(\alpha+1)}\begin{bmatrix}\frac{\alpha}{4} & \frac{1-\alpha}{4} & \frac{\alpha}{4}\\ \frac{1-\alpha}{4} & -1 & \frac{1-\alpha}{4}\\ \frac{\alpha}{4} & \frac{1-\alpha}{4} & \frac{\alpha}{4}\end{bmatrix}$$

تابع fspical فیلتر Laplacian of Gaussian(LoG) را با الگوریتم زیر می سازد:

$$h_g(n_1,n_2)=e^{-\left(n_1^2+n_2^2\right)/\left(2\sigma^2\right)}$$

$$h(n_1,n_2)=\frac{\left(n_1^2+n_2^2-2\sigma^2\right)h_g(n_1,n_2)}{2\pi\sigma^6\sum_{n_1}\sum_{n_2}h_g}$$

تابع fspical فیلتر averaging را الگوریتم زیر می سازد:

Ones (n (1), n (2)) / (n (1)* n(2))

تعادل Contrast یک تصویر

در برخی از تصاویر، جهت پردازش بهتر و ساده تر بر روی تصویر، لازم است که شدت رنگ آن تصویر افزایش یابد.

تابعی که این کار را انجام می دهد، imadjust نام دارد. این تابع، ورودی را از کاربر گرفته و دو بردار در خروجی می دهد، که به صورت زیر تعریف می شود:

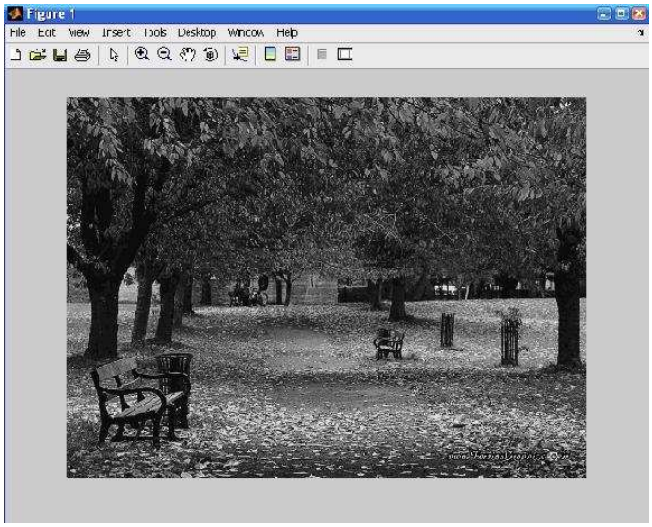
[bottom top] ,[low heigh]

افزایش شدت رنگ مربوط به یک تصویر، کاربردهای فراوانی دارد که جهت آشنایی بیشتر با این تابع، مثال زیر بررسی شود.

مثال

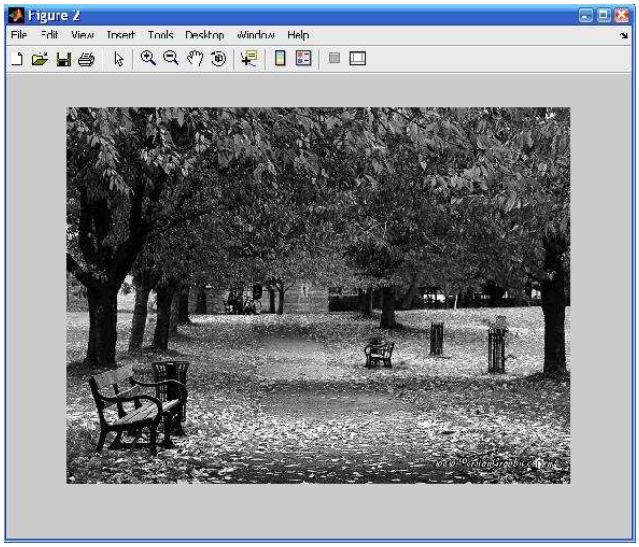
در این مثال، هدف، افزایش شدت رنگ تصویری به نمام park به کمک تابع imadjust می باشد.

```
i=imread('park.tif') ;
imshow(i)
```



تصویر۵۷-۳-۲

```
j=imadjust(i);
imshow(j)
```



تصویر۵۸-۳-۲

همان گونه که در تصاویر (۵۷-۳ ۲) و (۵۸-۲۳) مشاهده می شود، تصویر ۵۷-۳ -2نشان دهنده حالت اولیهٔ تصویر park می باشد و تصویر ۵۸-۳ پس از اعمال تغییرات افزایش شدت رنگ توسط تابع imadjust می باشد.

حالات مختلف آرگومان های تابع imadjust

در ابتدا به بیان چند نکته در این رابطه پرداخته می شود:

۱. در تمام مثال های زیر مقدار گاما (آرگومان چهارم) حذف شده است، که پیش فرض آن [0 1] می باشد.

۲. آرگومان دوم، حد بالاوپایین تصویر ورودی و آرگومان سوم، حد بالاوپایین تصویر خروجی می باشد.

۳. در آرگومان های ورودی و خروجی، باید حد بالا از حد پایین بزرگ تر باشد.

1-I3=imadjust(i2,stretchlim(i2),[0 1])

توضیحات مربوط به تابع stretchim

این تابع به طور اتوماتیک، شدت رنگ تصویر را با تنظیم مقداری به صورت زیر محاسبه می کند.

Right[low high]

2- I3=imadjust(i2, stretchlim(i2),[1 0])

دراین حالت رنگ Foreground و Background معکوس می شود. و کیفیت تصویر مناسب می باشد.

3- I3=imadjust(i2,[0 1],[0 1])

معادل دستور شمارهٔ 1 عمل می کند.

4- I3=imadjust(i2,[1 0],[0 1])

پیغام خطایی به دلیل بالاتر بودن حد پایین تصویر ورودی (low) از حد بالای تصویر ورودی "high) داده می شود.

5- I3=imadjust(i2,[0 1],[1 0])

در این حالت، رنگ Foreground و Background معکوس می شود.

6- I3=imadjust(i2,[1 0],[1 0])

پیغام خطا: در هردو آرگومان، ورودی خروجی، حد پایین ازحد بالا، بزرگ تر می باشد.

7- I3=imadjust(i2,[0 0],[0 0])

8- I3=imadjust(i2,[1 1],[1 1])

پیغام خطا، آرگومان ورودی باید از آرگومان خروجی بیشتر باشد. (برای هر دو شماره7 و8)

9- I3=imadjust(i2,[1 0],[1 1])

در این حالت، خروجی به رنگ سفید است. (رنگ غالب همان رنگ Foreground است).

10- I3=imadjust(i2,[1 0],[0 0])

در این حالت، خروجی به رنگ سیاه می باشد. (رنگ غالب همان رنگ Background است).

عملیات Thresholding بر روی یک تصویر

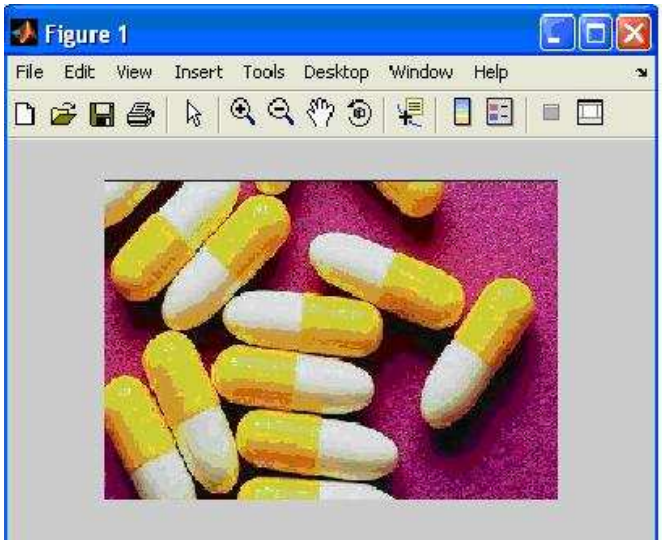
به طور کلی، عملیات Thresholding بر روی تصاویر به چند دسته اساسی تقسیم بندی می شود، که در این قسمت به توضیحات مربوط به هر یک پرداخته خواهد شد.

1- Create a Binary Version of the Image

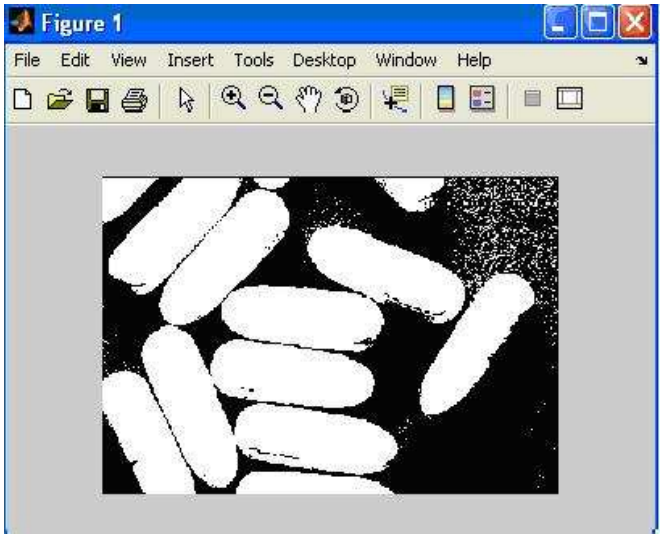
در این فرمت، ساخت نسخهٔ Binary تصویر بررسی خواهد شد. تابع Graythresh به طور خودکار حدود اختصاص داده شده در تصویر را برای معکوس کردن تصاویر Grayscale به فرمت Binary محاسبه می کند، این کار توسط تابع im2bw انجام می شود.

مثال:در این مثال، جهت آشنایی بیشتر با عملیات Thresholding، این عملیات را برروی تصویر cap پس از عملیات فراخوانی انجام می دهیم.

```
Level=graythresh(I3);
Bw=im2bw(I3,level);
```



تصویر۵۹-۳-۲



تصویر۶۰-۳-۲

کلاس این تصویر، به فرمت منطقی می باشد که در زیر با فراخوانی تابع whos مشاهده می گردد.

```
whos
MATLAB respond with
Name          Siza            Byte Class
I           256*256          65536 uint8 array
I2          256*256          65536 uint8 array
I3          256*256          65536 uint8 array
background  256*256          65536 uint8 array
bw          256*256          65536 logical array
level         1*1                8 double array

grand total is 327681 elements using 327688 byte
```

2-grayslice

در این فرمت، ساخت تصاویر Index، از تصایر به فرمت Intensity با استفاده از محدودهٔ چند سطحی (multilevel thresholding) بررسی خواهد شد.

برای آشنایی بیشتر با نحوهٔ کارکرد این تابع، syntax های مربوط به این تابع بررسی می شود.

X=grayslice(I,n)

X=grayslice(I,v)

X=grayslice(I,n)

این دستور، تصویر Intensity با نام I را با مقادیر $\frac{1}{n}, \frac{2}{n}, \ldots \frac{n-1}{n}$ به تصویر به فرمت Index به نام × تبدیل می کند.

X=grayslice(I,v)

قابل ذکراست که V برداری بین اعداد صفر و یک است.

کلاس تصویر ورودی I می تواند uint8، uint16،single یا به صورت double باشد و کلاس تصویر خروجی x بستگی به Threshld value دارد، که به وسیله n یا طول v تعیین می گردد.

در صورتی که Threshld value کمتر از 256 باشد، و محدوده مقادیر X بین 0 تا n باشد، یا به اندازهٔ طول v کلاس X، uint8 است.

اگر Threshld value برابر یا بزرگتر از 256 باشد، کلاس X، double می باشد و محدوده X از n+1 تا 1 t lingth (v) می باشد.

مثال

```
I = imread('snowflakes.png');
X = grayslice(I,16);
imshow(I)

figure, imshow(X,jet(16))
```

به کمک تابع bwlabel، می توان به ترکیبات متصل به هم در تصاویر باینری بر[سب زد. Syntaxهای مربوط به این تابع بدین صورت می باشد.

1-L=bwlabel(BW,n)

2-[L,num]=bwlabel(BW,n)

Syntax شمارهٔ اول، ماتریس L راکه هم اندازهٔ BW می باشد بر می گرداند. محتویات L شامل object های متصل به هم در BW می باشد.

مقدار n می تواند به صورت 4 یا 8 تعریف شود. با انتخاب عدد 4، نحوهٔ اتصال به صورت چهارتایی و با در نظر گرفتن عدد 8، نحوه اتصال به صورت هشت تایی خواهد بود. پیش فرض مقدار n به صورت عدد هشت می باشد.

المنت L، عددی صحیح برابر یا معادل 0 می باشد. منظور از برچسب پیکسل صفر Background است. برچسب یک بدین معناست، که اولین object مدنظر است، برچسب 2 به معنای دومین object و ... Syntax دوم، تعداد object های متصل به هم پیدا شده در BW را به متغیر num برمی گرداند.

Bwlabel تنها ورودی های دو بعدی را پشتیبانی می کند، اما bwlabeln ورودی با هر بعدی را پشتیبانی می کند. پس هرگاه در استفاده از ورودی های دو بعدی با مشکل برخورد کردید، با مراجعه به bwlabeln، مشکل قابل حل است.

کلاس های پشتیبانی شده

BW می تواند به صورت فرمت منطقی یا عددی باشد، اما باید جتماً به صورت عددی حقیقی، دو بعدی و غیر پراکنده باشد.

قابل ذکر است، که کلاس L باید به فرمت double باشد.

در کنار این تابع می توان از تابع جستجو (find) نیز استفاده کرد. برای نمونه جهت برگرداندن مختصات پیکسل ها با object شماره دو می توان بدین صورت نوشت:

```
[r,c]=find(bwlabel(BW==2)
```

جهت آشنایی بیشتر با این تابع، نمونهٔ زیر بررسی شود.

در زیر از عدد اتصالی 4 برای اتصال object ها استفاده شده است. به objectهای 2 و 3 توجه کنید. با استفاده از عدد اتصالی هشت، BW ممکن است به صورت تک object به نظر رسد.

```
BW=[1 1 1 0 0 0 0 0
    1 1 1 0 1 1 0 0
    1 1 1 0 1 1 0 0
    1 1 1 0 0 0 1 0
    1 1 1 0 0 0 1 0
    1 1 1 0 0 0 1 0
    1 1 1 0 0 1 1 0
    1 1 1 0 0 0 0 0];

L=bwlabel(BW,4)
  L=1 1 1 0 0 0 0 0
    1 1 1 0 2 2 0 0
    1 1 1 0 2 2 0 0
    1 1 1 0 0 0 3 0
    1 1 1 0 0 0 3 0
    1 1 1 0 0 0 3 0
    1 1 1 0 0 3 3 0
    1 1 1 0 0 0 0 0


[r,c]=find(L==2);
rc=[r,c]
rc=
  2 5
  3 5
  2 6
  3 6
```