مقدمه‌ای بر پردازش تصویر

سید کاوه احمدی

منابع مطرح در پردازش تصویر

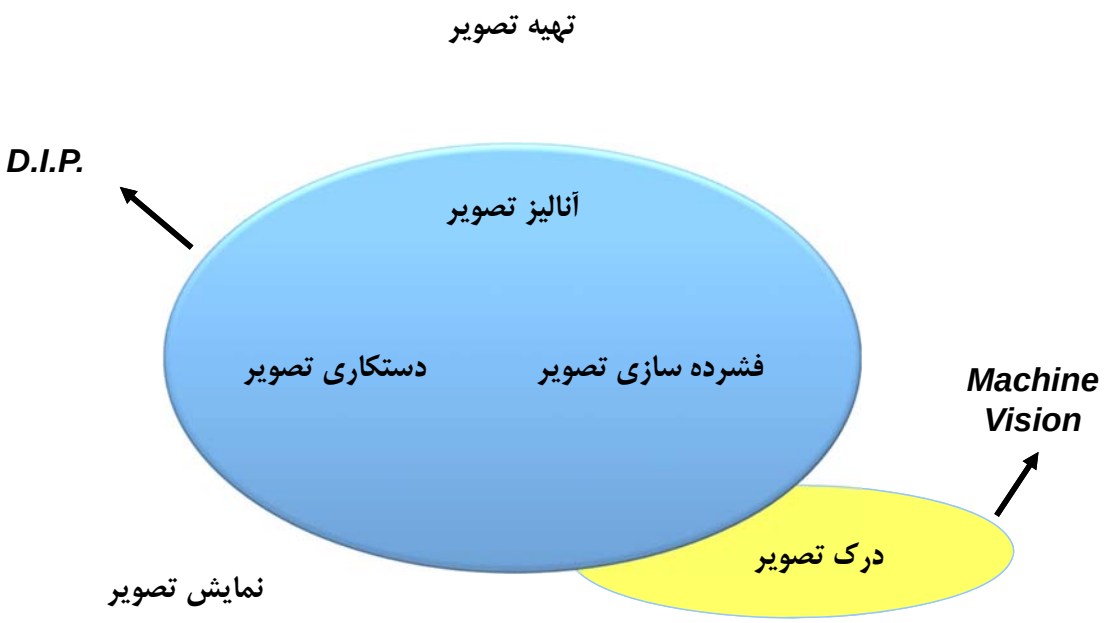
- R. C. Gonzalez and R. E. Woods, **Digital Image Processing**, 3rd edition, Prentice Hall, 2009
- R. C. Gonzalez and R. E. Woods, **Digital Image Processing Using MATLAB**, 2nd edition, Gatesmark Publishing, 2009
- A. K. Jain, **Fundamentals of Digital Image Processing**, Prentice Hall, 1989
- G. X. Pratt, **Digital Image Processing**, 3rd Edition, John Wiley & Sons, 2001

پردازش تصویر با چی؟

- Matlab DIP Tool Box
- OpenCV (http://opencv.org/)
  - It has C++, C, Python and Java interfaces
  - Supports Windows, Linux, Mac OS, iOS and Android
- OpenCV .NET Wrapper:
  - Emgu CV (http://www.emgu.com)

- Aforge.NET(http://code.google.com/p/aforge/)
  - C# framework designed for developers and researchers in the fields of Computer Vision and Artificial Intelligence - image processing, neural networks, genetic algorithms, machine learning, robotics, etc.
    - **AForge.Imaging** - library with image processing routines and filters; **AForge.Vision** - computer vision library;

راهنمای تور



زمینه های مختلف کاربرد پردازش تصویر

- صنعت
- هواشناسی
- پزشکی
- صنایع نظامی و امنیتی
- فناوری های علمی
- کشاورزی
- باستان شناسی
- نجوم و فضا نوردی
- و...

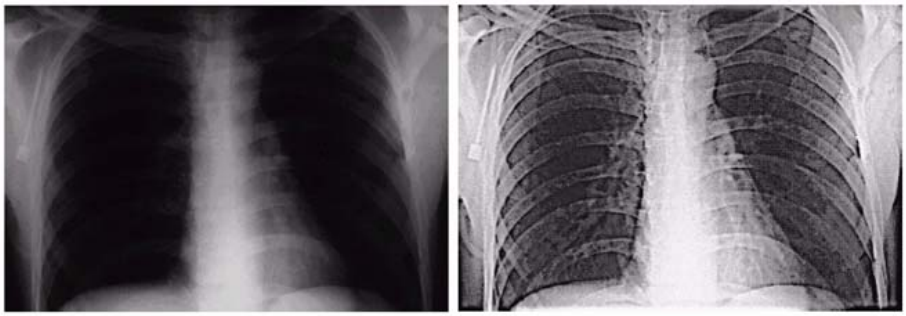
پردازش تصویر چیست؟

- به شکل خیلی ساده می‌توان گفت پردازش تصویر ایجاد تغییرات مناسب در تصویر به منظور بهبود اطلاعات جهت قابل فهم بودن برای انسان یا درک صحنه توسط کامپیوتر است.
  - بهبود اطلاعات تصویر برای دیدن انسان
  - پردازش داده های صحنه برای ادراک توسط ماشین
    - درک صحنه توسط کامپیوتر (Machine Vision)

تفاوت پردازش و تصمیم گیری در مورد تصویر بین انسان و کامپیوتر

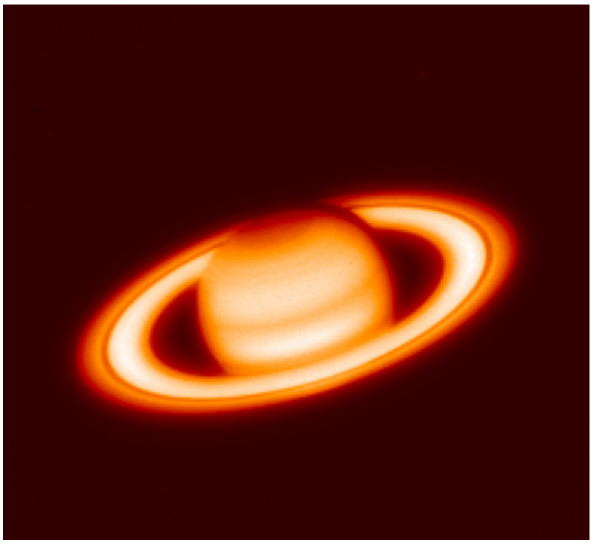
1. تصمیم گیری در سیستم بینایی انسان بستگی به شرایط مختلف نوری ، زاویه ، خستگی و ... دارد ولی در پردازش تصویر دیجیتال، نتیجه بر اساس محاسبات انجام می شود.
2. محدوده بینایی انسان فقط در نور مرئی است ولی در پردازش تصویر دیجیتال می توان تصویر را از هر سنسوری در محدوده های مختلف امواج دریافت کرد.
3. وفق پذیری چشم انسان با شرایط محیطی بسیار بیشتر است. ولی انسان تفاوت رنگها و سطوح روشنایی را بسیار کمتر از کامپیوتر تشخیص می دهد.
4. سرعت در پردازش تصویر دیجیتال بسیار وابسته به نوع سخت افزار و الگوریتم بکار رفته است.

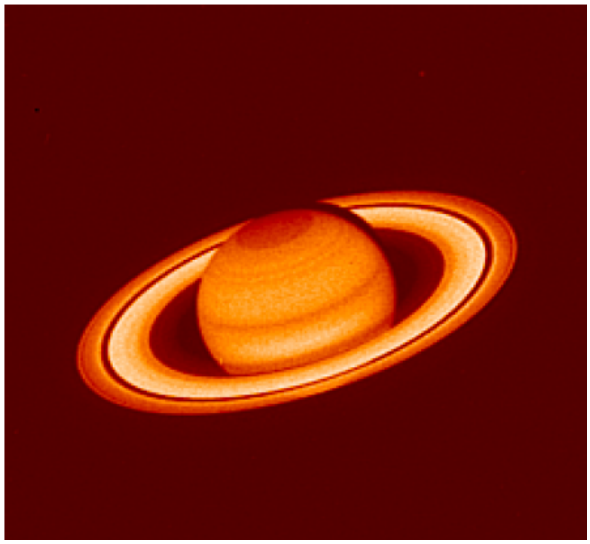
مثال‌هایی از پردازش تصویر دیجیتال (بهبود اطلاعات تصویر برای دیدن انسان)



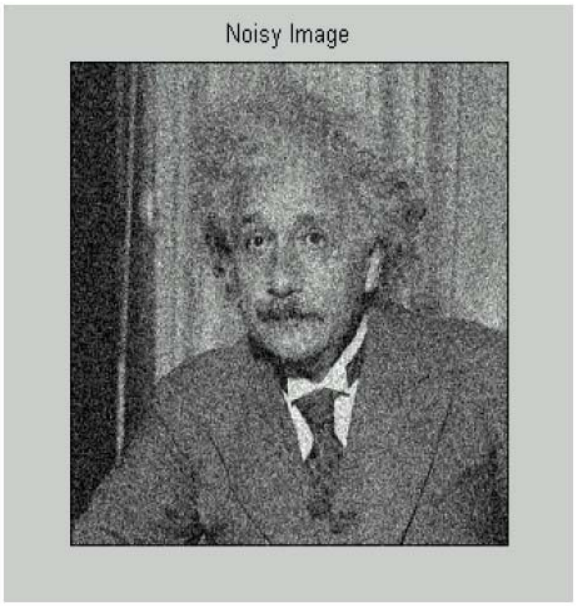
مثال‌هایی از پردازش تصویر دیجیتال (افزایش تباین)

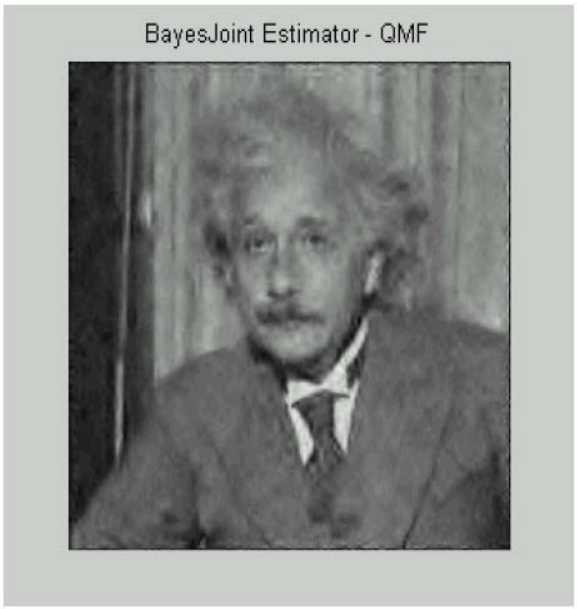
مثال‌هایی از پردازش تصویر دیجیتال (ترمیم)





مثال‌هایی از پردازش تصویر دیجیتال (کاهش نویز)





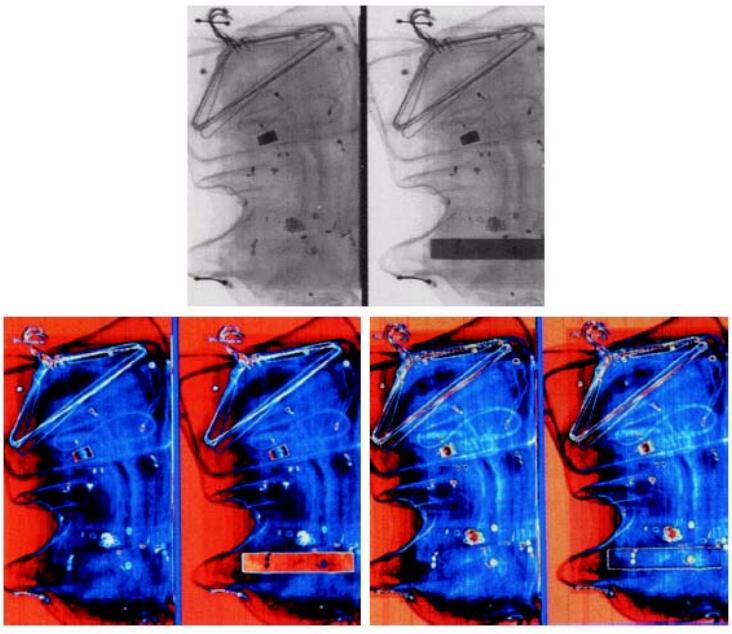
مثال‌هایی از پردازش تصویر دیجیتال (جلوه‌های ویژه)

Photo

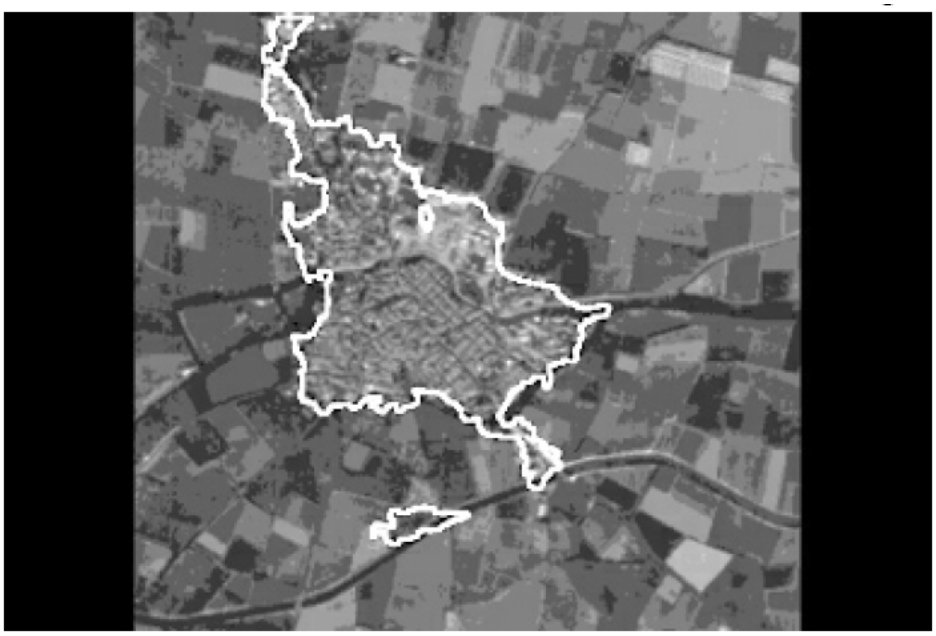
Simulated
color pencils

Simulated
oil painting

مثال‌هایی از پردازش تصویر دیجیتال (کاربرد امنیتی)



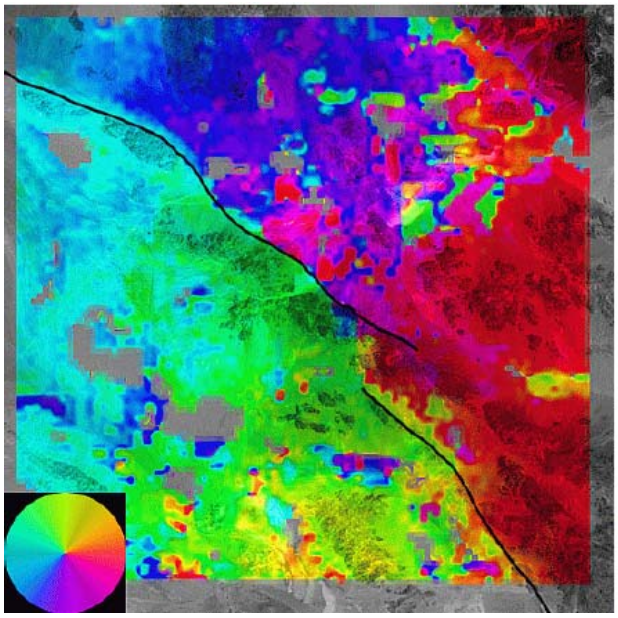
مثال‌هایی از پردازش تصویر دیجیتال



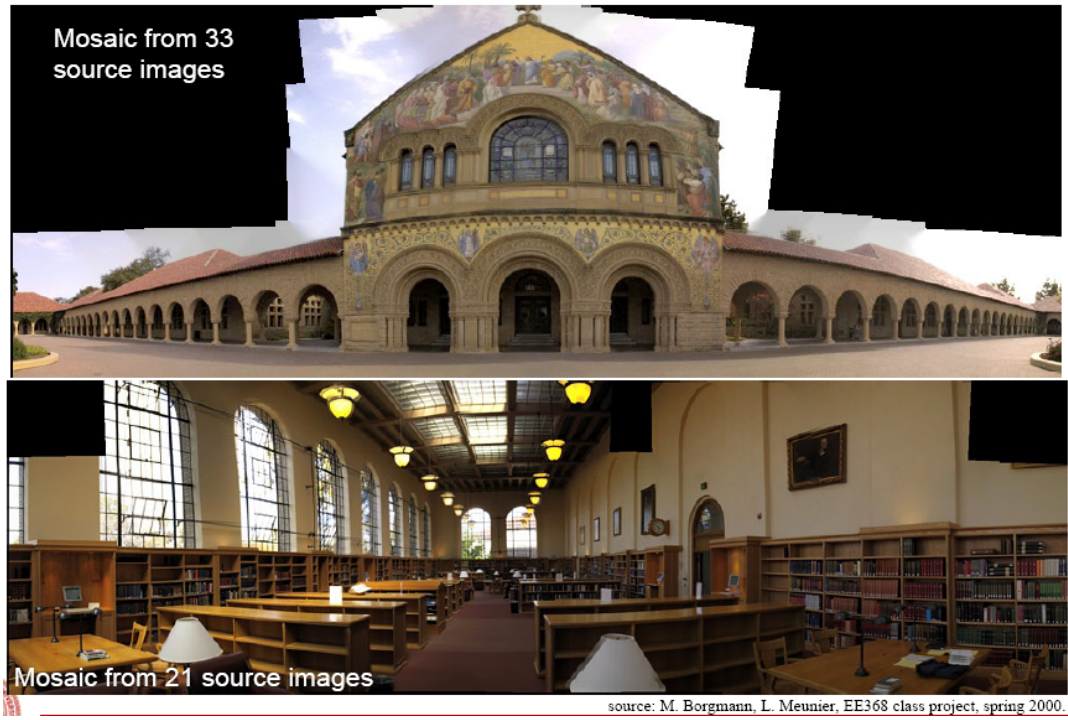
source: INRIA, Sophia-Antipolis,

مثال‌هایی از پردازش تصویر دیجیتال

- تجزیه و تحلیل زلزله از فضا
- تصویر جابجایی زمین در اثر زلزله لندرز را نشان می‌دهد

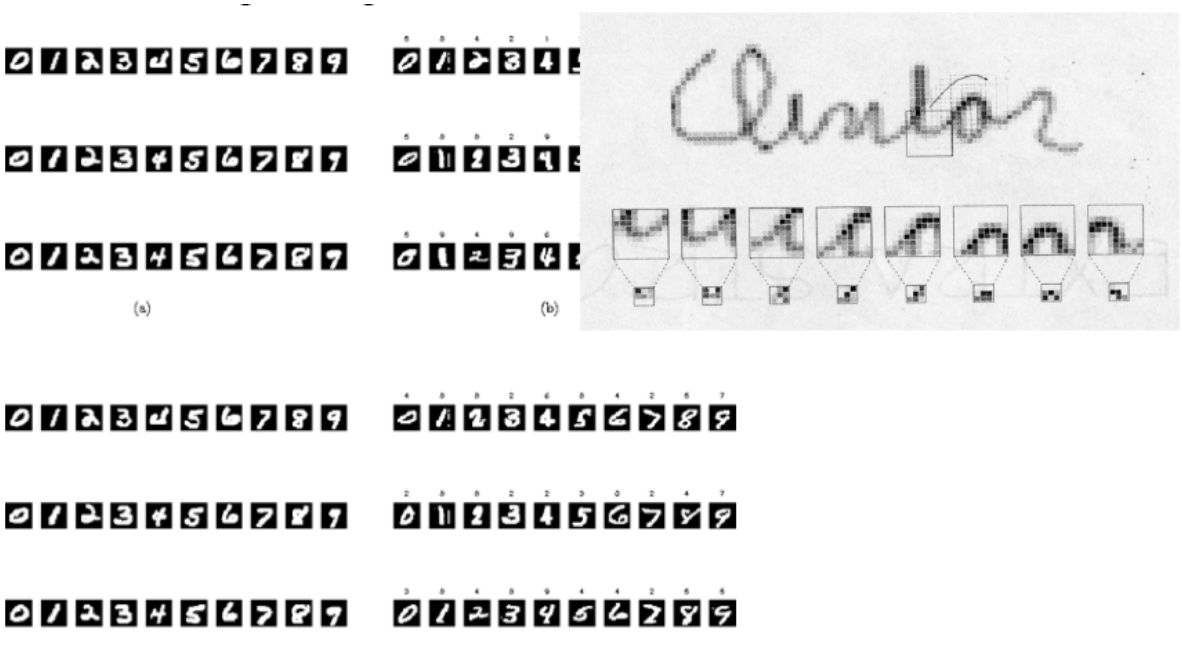


مثال‌هایی از پردازش تصویر دیجیتال (ایجاد تصاویر پانوراما)



source: M. Borgmann, L. Meunier, EE368 class project, spring 2000.

مثال‌هایی از پردازش تصویر دیجیتال (تشخیص دست خط)

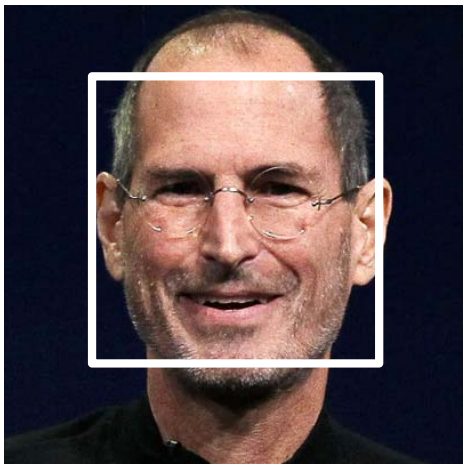


مثال‌هایی از پردازش تصویر دیجیتال (تشخیص چهره)

- Face Detection

مثال‌هایی از پردازش تصویر دیجیتال (احراز هویت چهره)

- Face Authentication



Steve Jobs

مشکلات پیش رو

عملیات اصلی در پردازش تصویر (بینایی ماشین)

Scene recognition

Object recognition

Object detection

عملیات اصلی در پردازش تصویر (بینایی ماشین)

Scene recognition

Object recognition

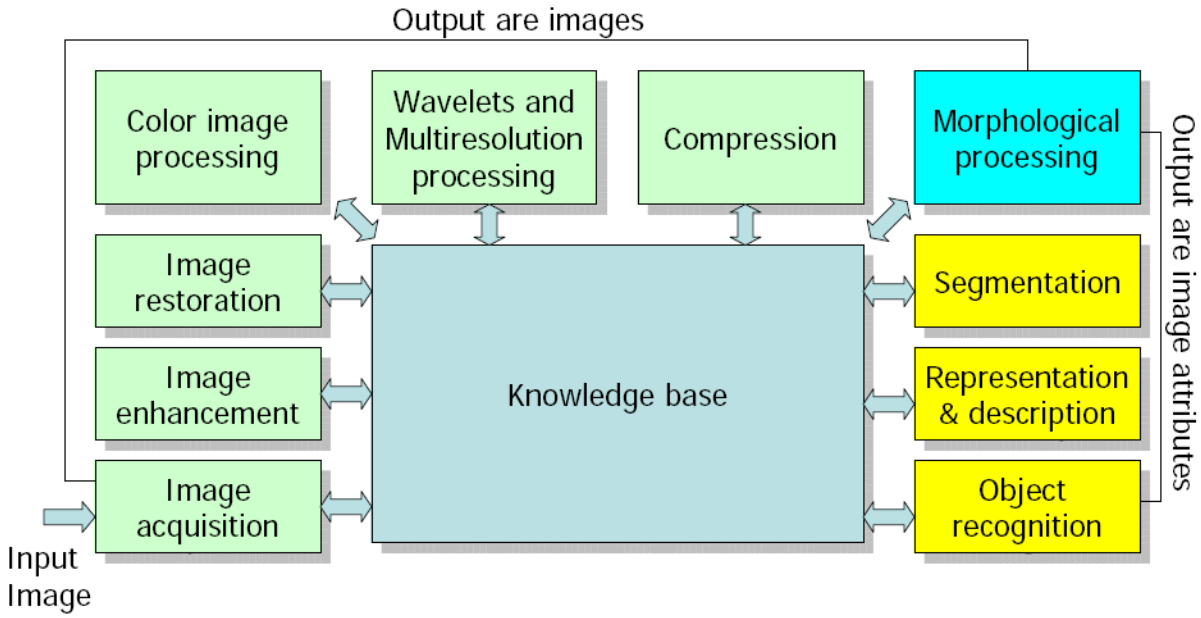
Object detection

Segmentation

عملیات اصلی در پردازش تصویر

- ترمیم تصویر: تصاویر اخذ شده از رادار ها و ماهواره ها
- حذف نویز: پیکسلی که اختلاف رنگ زیادی با پیکسل های مجاور دارد
- بالا بردن دقت تصویر: افزایش کنتراست (تباین) تصویر از طریق کشیدن هیستوگرام و امتداد آن
- آشکار سازی لبه‌ها
- تبدیلات هندسی: همانند تغییر اندازه، چرخش و...
- رنگ: همانند کاهش رنگ‌ها
- فشرده‌سازی تصویر: کاهش حجم تصویر

اجزا پردازش تصویر دیجیتال



بدانید!

- تمامی پردازش هایی که بر روی تصویر انجام می گیرد با استفاده از قواعد و روابط ریاضی است .
  - مشتقات مرتبه اول و دوم
  - عملگر سوبل
  - محاسبه گرادیان x و y و آستانه‌گیری پیکسلی.
  - تبدیل فوریه
  - عملگر های لاپلاسی و مار هیلدرث
  - ویولت
  - عملگر میانگین

مراحل پردازش تصویر

1. دریافت تصویر ورودی

- نمونه برداری (Sampling): شکسته شدن تصویر به پیکسل های مجزا
  - تبدیل آنالوگ به دیجیتال

مراحل پردازش تصویر

2. پیش پردازش (بهبود تصویر)

- پیش پردازش:
  - هدف از پیش‌پردازش، بهبود داده‌های تصویر به شکلی که اعوجاج ناخواسته در تصویر را از بین ببرد و/یا بهبود ویژگی‌های تصویر با توجه به پردازش مورد نیاز روی تصاویر است.
  - پیش پردازش محتوای اطلاعات تصویر را افزایش نمی‌دهد.
  - پیش‌پردازش در شرایطی متفاوت و برای حذف اطلاعاتی که برای پردازش مورد نظر ما ضروری نیست می‌تواند مفید باشد.

مراحل پردازش تصویر

3. پردازش تصویر (استخراج ویژگی)

- پیدا کردن ویژگی‌هایی از تصویر که برای پردازش مورد نیاز مفید باشد. این مرحله تا حد زیادی وابسته به مسئله است.
- نمونه‌هایی از ویژگی‌ها: لبه‌ها، گوشه‌ها، اشیا و...

4. آنالیز تصویر

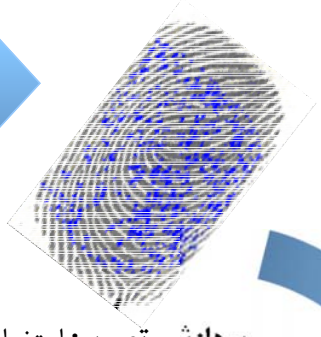
- استفاده از تصویر بهبود یافته و ویژگی‌های استراج شده برای آنالیز تصویر و رسیدن به مقصود مورد نظر از پردازش تصویر.

مراحل پردازش تصویر

دریافت تصویر ورودی

پیش پردازش:

- بهبود تصویر
- حذف ویژگی‌های غیر ضروری
- حذف مشکلاتی از قبیل جراحت و... از تصویر



پردازش تصویر: استخراج ویژگی‌هایی که در تشخیص اثر انگشت اهمیت دارند (شناسایی نقاط ویژه، انحناها و...).

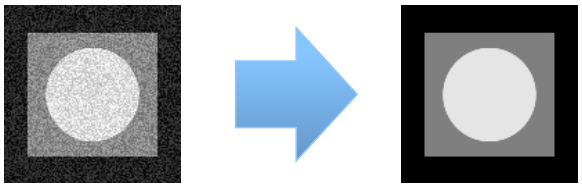
آنالیز تصویر:
پیدا کردن شخص متناظر با اثر انگشت با استفاده از ویژگی‌های استخراج شده

بهبود تصویر (پیش پردازش)

- همانطور که اشاره شد این عملیات (پیش پردازش) معمولا پیش از عملیات پردازش اصلی یا عملیات آنالیز تصویر انجام می‌شود.
- در این عملیات بهبودهایی بر روی داده‌های تصویر اعمال می‌شود تا امکان استخراج دقیقتر و صحیح‌تر اطلاعات میسر گردد.
- این عملیات در سه بخش زیر شرح داده خواهد شد:
  - افزایش کیفی و تنظیم شدت
  - افزایش تباین (کنتراست)
  - حذف نویز
  - فیلترهای نرم/تیز کننده
  - حذف مولفه‌های غیر ضروری

بهبود تصویر (پیش پردازش)

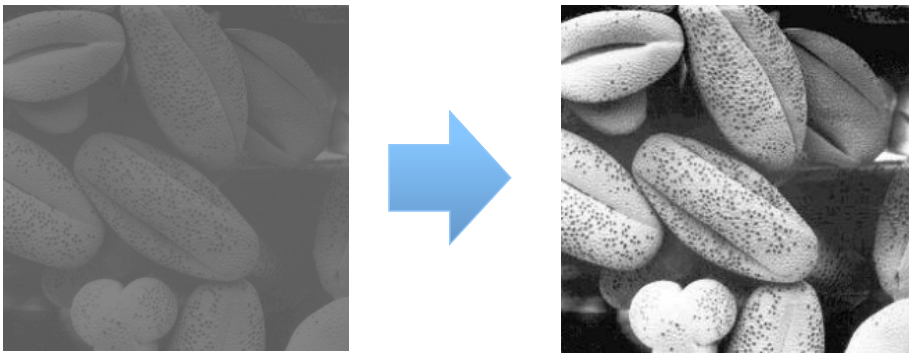
- حذف نویز



- افزایش کیفیت

بهبود تصویر (پیش پردازش)

- افزایش تباین



بهبود تصویر

تصویر smooth شده

تصویر sharp شده

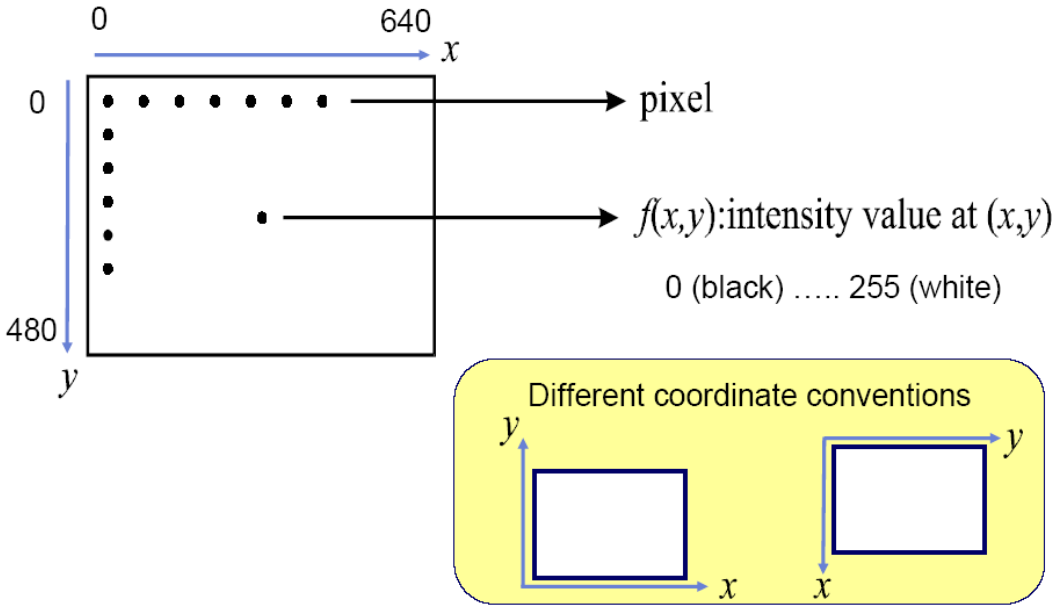
تصویر و دیجیتال

- تصویر
  - تابع دو بعدی `f(x, y)`
    - `x, y`: مختصات مکانی
    - مقدار `f`: شدت یا سطح خاکستری

- تصویر دیجیتال
  - هر دو مختصات و مقدارها گسسته هستند
  - یک مجموعه از پیکسل‌ها
  - منظور از پیکسل
    - مختصات پیکسل
    - مقدار پیکسل
    - یا هر دو

پیکسل

- `640 x 480 8-bit image`



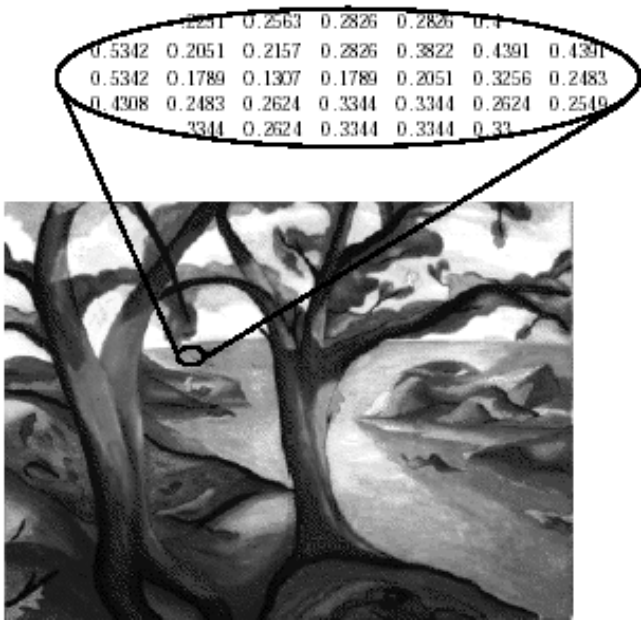
انواع تصاویر

- تصاویر اندیس‌دار
- تصاویر شدت
- تصاویر باینری
- تصاویر RGB
- تصاویر چندفریمی
- فرمتهای گرافیکی: تصاویر با فرمتهای مختلفی می‌توانند بر روی دیسک ذخیره شوند. مهمترین فرمتهای گرافیکی در زمان حاضر عبارتند از: PNG، JPG، BMP، TIFF، GIF که تمامی آنها بعلاوه چندین فرمت دیگر توسط متلب پشتیبانی می‌شوند.

تصاویر شدت/خاکستری (Intensity Image)

- تنها دارای مقادیر شدت روشنایی هستند.
  - فاقد خصوصیات رنگ مانند: فام و خلوص
- در متلب این تصاویر توسط ماتریس‌های دو بعدی تعریف می‌شوند.
  - مقدار هر عنصر از این ماتریس معرف میزان روشنایی پیکسل متناظرش در تصویر می‌باشد.
  - دامنه تغییرات عناصر این ماتریس ممکن است بین ۰ تا ۱ و یا بین ۰ تا ۲۵۵ تغییر کند.
  - در حالت اول داده‌های ماتریس از نوع `double` و در حالت دوم از نوع `uint8` خواهد بود
    - میزان حافظه مورد نیاز `double` چهار برابر نوع `uint8` است.

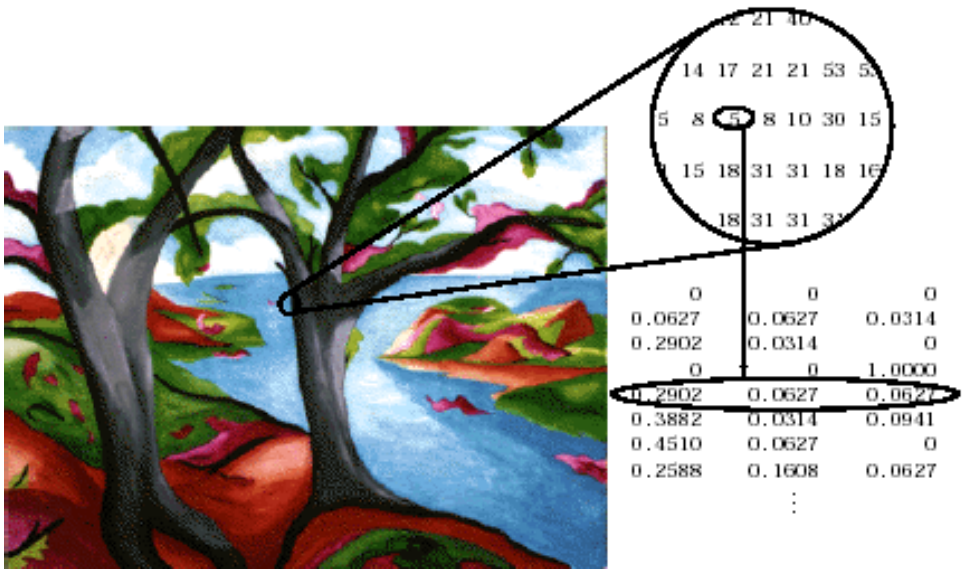
تصاویر شدت/خاکستری (Intensity Image)



تصاویر اندیس‌شده (Indexed Image)

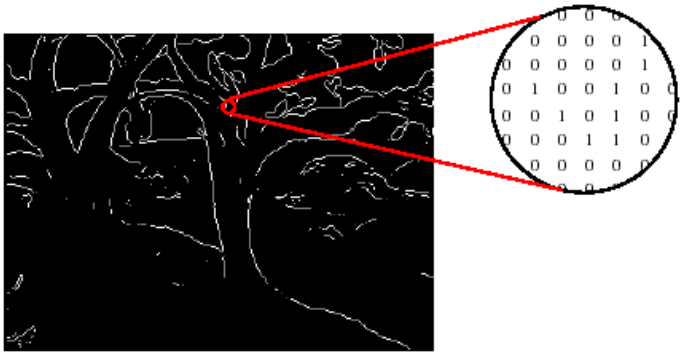
- این تصاویر توسط دو ماتریس زیر مشخص می‌شوند:
  - ماتریس اندیس: ماتریسی با ابعادی برابر با ابعاد تصویر
    - مقادیر این ماتریس معمولا بین ۱ تا ۲۵۶ تغییر می‌کند که این مقادیر، معرف شماره سطری از ماتریس نقشه‌رنگ است.
  - ماتریس نقشه‌ی رنگ (map): ماتریسی دارای ۳ ستون که هر سطر از آن معرف یکی از رنگ‌های موجود در تصویر است.
    - عنصر اول هر سطر معرف نسبت اولیه قرمز، عنصر دوم معرف نسبت اولیه سبز و عنصر سوم معرف نسبت اولیه آبی است.
- یک تصویر اندیس‌شده بسته به مقادیر ماتریس نقشه‌ی رنگ، ممکن است رنگی یا خاکستری باشد.

تصاویر اندیس‌شده (Indexed Image)



تصاویر باینری

- تصویری که هر پیکسل از آن تنها بتواند دارای یکی از دو مقدار ممکن (معمولا ۰ و ۱) باشد.
  - در متلب این تصاویر می‌توانند با فرمت `double` و یا `uint8` ذخیره شوند.
  - بطور پیش‌فرض متلب فرمت `uint8` را به کار می‌برد که مقادیر آن می‌تواند، ۰ و ۱ و یا ۰ و ۲۵۵ باشد.



تصاویر RGB (True Color)

- تصاویر رنگی که به ازای هر پیکسل از آن، سه نسبت اولیه از رنگ‌های اصلی ذخیره می‌شود بین ۰ تا ۲۵۵).
  - برای یک پیکسل سفید (255, 255, 255): بیشترین شدت روشنایی R و G و B
  - برای یک پیکسل سبز (0, 255, 0): بیشترین شدت روشنایی G و کمترین شدت روشنایی R و B
- برای هر نقطه از تصویر بیش از ۱۶ میلیون (۲۵۶*۲۵۶*۲۵۶) حالت رنگی مختلف امکانپذیر خواهد بود.
  - یک تصویر RGB سه برابر یک تصویر خاکستری هم‌اندازه حافظه اشغال می‌کند و به همان نسبت هم به زمان پردازش بیشتری نیاز دارد.
- در متلب هر تصویر RGB بصورت یک ماتریس سه‌بعدی تعریف می‌شود که در بعد سوم آن مقادیر اولیه‌های رنگی هر نقطه (r, g, b) ذخیره می‌شوند. عناصر این ماتریس ممکن است بین ۰ تا ۱ (double) و یا بین ۰ تا ۲۵۵ (uint8) تغییر کند
- دقت شود که یک تصویر RGB لزوما رنگی نیست اما قابلیت رنگی بودن را دارد.

تصاویر RGB (true color)

خواندن تصاویر در Matlab

- **imread**
  - بسته به نوع تصویر، فرمت کلی استفاده از این تابع به یکی از صورتهای زیر است:
    - برای تصاویر خاکستری، RGB و باینری:

```
I = imread('filename')
```

      - برای تصاویر خاکستری و باینری، I یک آرایه دو بعدی و برای تصاویر RGB یک آرایه ۳ بعدی خواهد بود.
    - برای تصاویر اندیس‌شده:

```
[I, map] = imread('filename')
```

      - I ماتریس اندیس و map ماتریس نقشه‌رنگ است
  - نکته: تابع imread را با تعداد آرگومانهای بیشتری نیز می‌توان فراخوانی کرد. جهت اطلاع بیشتر به راهنمای متلب رجوع کنید.

نمایش تصاویر در Matlab

- **imshow**
  - نمایش تصاویر خاکستری، RGB و باینری:

```
imshow(I);
```

  - نمایش تصویر اندیس‌شده

```
imshow(I, map)
```

  - نمایش فایل گرافیکی

```
imshow('filename');
```

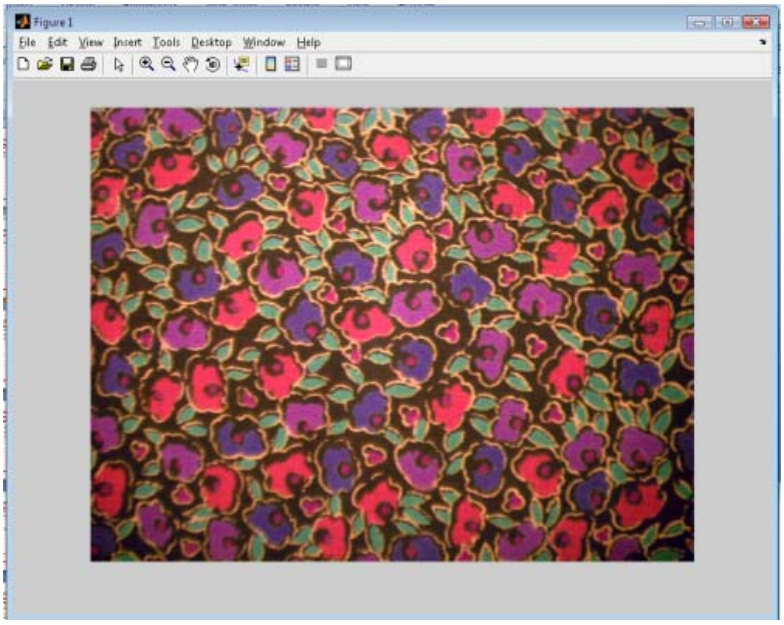
  - مثال:

```
imshow('fabric.png')
```

    - یا:

```
I = imread('fabric.png');
imshow(m)
```

نمایش تصاویر در Matlab



ذخیره تصاویر در Matlab

- **imwrite**
  - `imwrite(I, 'filename');`
  - `imwrite(I, map , 'filename');`

مشاهده مشخصات تصویر در Matlab

- **imfinfo**
  - اطلاعاتی از فایل گرافیکی مانند ابعاد تصویر، دقت ابعادی و دقت عمقی، نحوه فشرده‌سازی و... را ارائه می‌دهد

```
info=imfinfo('filename')
```


تبدیل نوع تصاویر در Matlab

- با استفاده از توابع زیر می‌توان نوع یک تصویر را تغییر داد:
  - `I = rgb2gray(I);`
  - `bw = im2bw(I, level);`
  - `bw = im2bw(I, map, level)`
    - `level` سطح آستانه می‌باشد که باید بین 0 تا 1 باشد.
  - `I = ind2gray(I, map);`
  - `[I, map] = gray2ind(I);`
  - `[I, map] = rgb2ind(I);`
  - `I = ind2rgb(I, map);`
- برای کسب اطلاعات بیشتر به راهنمای متلب مراجعه کنید.

عملیات ریاضی بر روی تصاویر در Matlab

- امکان بکاربردن عملگرهای ریاضی و بسیاری از توابع متلب بر روی تصاویر نوع `uint8` وجود ندارد.
  - پیش از انجام عملیات ریاضی باید نوع داده‌ها را به `double` تبدیل کرد.
    - `I = double(I);`
  - پس از انجام عملیات ریاضی در صورت نیاز می‌توان نوع متغیر را به `uint8` بازگرداند:
    - `I = im2uint8(I);`

عملیات هندسی بر روی تصاویر در Matlab

- **`imresize`**
  - تغییر ابعاد تصویر:
    - این تابع به یکی از دو صورت زیر قابل استفاده است:
      - `Y = imresize(I, a);`
        - `a` نسبت تغییر در ابعاد تصویر است. `a=2` تصویر را دو برابر می‌کند
      - `Y = imresize(I, [m , n]);`
        - `m` و `n` ابعاد جدید تصویر (تعداد سطرها و ستون‌ها) را مشخص می‌کند. بدیهی است که باید اعداد صحیح مثبت باشند.

عملیات هندسی بر روی تصاویر در Matlab

- **imrotate**
  - چرخش تصویر:
    - `Ir = imrotate(I, d, ['Option'], ['crop'])`
      - آرگومان دوم میزان چرخش تصویر برحسب درجه می‌باشد.
      - آرگومان سوم اختیاری بوده و می‌تواند یکی از مقادیر `bilinear, nearest` یا `bicubic` باشد. این آرگومان الگوریتم بکار رفته در فرآیند چرخش را مشخص می‌کند.
        - در صورتیکه این آرگومان بکار برده نشود، مقدار پیش‌فرض `nearest` خواهد بود.
      - آرگومان چهارم نیز اختیاری می‌باشد و تنها می‌تواند مقدار `'crop'` را داشته باشد. درصورتیکه بکار برده شود، ابعاد تصویر پس از چرخش تغییر نمی‌کند و بخشی از تصویر برش داده و حذف می‌شود.

عملیات هندسی بر روی تصاویر در Matlab

- **imrotate**

```
I = imread('cameraman.tif');
Ir = imrotate(I, 35);
Irc = imrotate(I, 35,'crop');
imshow(I);
figure, imshow(Ir);
figure, imshow(Irc);
```

عملیات هندسی بر روی تصاویر در Matlab

- **imrotate**

I

Ic

Irc

عملیات هندسی بر روی تصاویر در Matlab

- **imcrop**
  - برش تصویر
    - این تابع به یکی از شکلهای زیر قابل استفاده است :
      - `I2 = imcrop(I, rect)`
      - `I2 = imcrop(I, map, rect)`
      - `I2 = imcrop(I, rect)`
    - که در این روابط `rect` یک بردار سطری است که مختصات ناحیه مستطیلی شکلی از تصویر که می‌خواهیم برش دهیم رامشخص می‌کند.
    - درصورتیکه این آرگومان در ورودی مشخص نشود، تصویر نمایش داده شده و متلب منتظر می‌ماند تا کاربر یک ناحیه مستطیلی را با ماوس انتخاب کند.

عملیات هندسی بر روی تصاویر در Matlab

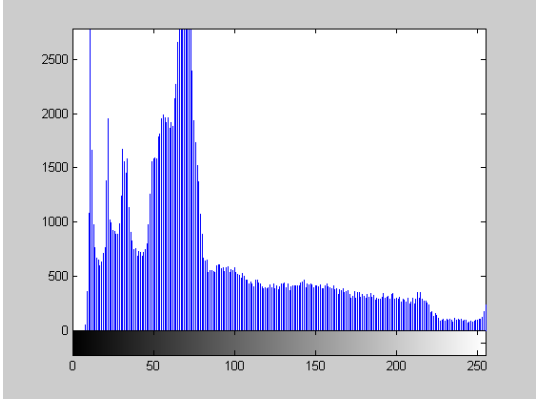
- **imcrop**

```
I = imread('pout.tif');
imshow(I);
I2 = imcrop(I, [size(I)/4, size(I)/2])
figure, imshow(I2);
```

هیستوگرام تصویر

- imhist
  - رسم نمودار فراوانی نقاط تصویر خاکستری

```
I = imread('flowers.tif');
I = rgb2gray(I);
imshow(I); figure, imhist(I);
```



فیلترهای خطی و طراحی فیلتر

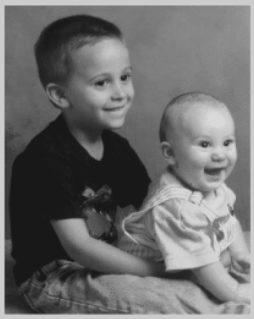
- برای اعمال یک فیلتر بر روی تصویر می‌توان از تابع `filter2` استفاده کرد:
- `I = filter2(h, m)`
  - در رابطه `h` ماتریس فیلتر و `m` ماتریس تصویر اولیه است.
  - `h` می‌تواند هر ماتریس با ابعاد دل‌خواه باشد، اما معمولا یک ماتریس ۳*۳ یا ۵*۵ است.

فیلترهای خطی و طراحی فیلتر

- با استفاده از تابع `fspecial` می‌توان فیلترهای معمول در پردازش تصویر را برای استفاده در تابع `filter2` ایجاد کرد.
- روش استفاده از این تابع بصورت زیر است:
- `h = fspecial(‘نام فیلتر‘ , ابعاد فیلتر)`
  - بسته به نوع آرگومان اول ممکن است این تابع با یک یا بیش دو آرگومان نیز بکار برده شود.
  - نام فیلتر می‌تواند یکی از پارامترهای زیر باشد:
    - `gaussian:` پایین گذر
    - `sobel:` بالا گذر
    - `prewitt:` بالا گذر
    - `laplacian:` فیلتر لاپلاس
    - `log:` اعمال فیلتر گوسی و پس از آن لاپلاس
    - `average:` فیلتر میانگین
    - `unsharp:` پایین گذر

فیلترهای خطی و طراحی فیلتر

```
[I, map] = imread('kids.tif');
SobelFilter = fspecial('sobel');
I = ind2gray(I, map);
I2 = filter2(SobelFilter, I);
imshow(I);
figure;
imshow(I2);
```



http://homepages.inf.ed.ac.uk/rbf/HIPRfiltops.htm/2

فیلترهای خطی و طراحی فیلتر

	-1	
-1	+5	-1
	-1	

scale= 1 *// the divisor for the sum of the coefficients*
Offset = 0 *// how far away the neighboring pixels are from the center pixel*

100	100	100
100	200	200
100	200	200

center pixel = –100 + –200 + –200 + –100 + 1000
= 1000 – 600
= ~~400~~ = 255, i.e., maximum allowed brightness.

آنالیز و بهسازی تصویر

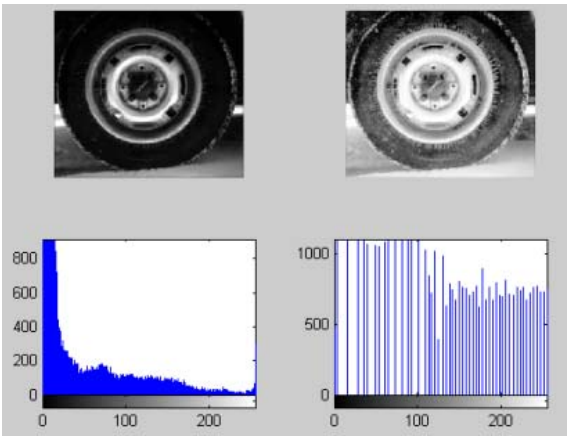
- آنالیز و بهسازی تصویر شامل سه عملیات زیر است:
  - بدست آوردن ارزش نقاط تصویر و اعمال عملیات آماری بر روی آنها
  - آنالیز تصویر به منظور استخراج اطلاعات در مورد ساختار کلی آن
  - بهسازی تصویر به منظور واضح‌تر شدن جزییات تصویر و حذف نویز به منظور آماده‌سازی برای عملیات پردازشی بعدی که در ادامه به هر یک خواهیم پرداخت
    - تنظیم شدت
    - متعادل کردن هیستوگرام یا بهسازی تباین
    - حذف نویز

بهبود تصویر: افزایش تباین (کنتراست) در Matlab

- **`histeq`**
  - افزایش تباین با متعادل کردن هیستوگرام
  - تابع `histeq` بصورت خودکار بهترین تنظیم هیستوگرام را بر روی تصویر انجام می‌دهد و معمولا کیفیت روشنایی تصویر را به میزان زیادی بهبود می‌بخشد.

بهبود تصویر: افزایش تباین (کنتراست) در Matlab

```
I = imread('tire.tif');
J = histeq(I);
figure;
subplot(2,2,1);imshow(I);
subplot(2,2,2);imshow(J);
subplot(2,2,3);imhist(I);
subplot(2,2,4);imhist(J);
```



بهبود تصویر: تنظیم شدت تصویر در Matlab

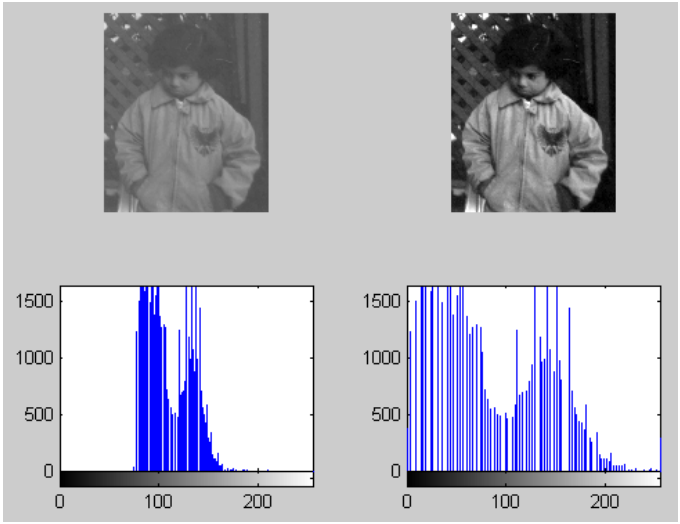
- **`imadjust`**
  - با استفاده از این تابع می‌توان دامنه تغییرات روشنایی یک تصویر را تغییر داد.
  - شکل کلی کاربرد این تابع بصورت زیر است:

```
J=imadjust(I , [low , high] , [bottom , top])
```

  - آرگومان دوم برداری دو عنصری است که بیانگر دامنه حاوی روشنایی‌هایی از تصویر است که عملیات تنظیم شدت بر روی آنها باید اعمال گردد.
  - آرگومان سوم، دامنه تغییرات جدید روشنایی برای نقاط فوق است.

بهبود تصویر: تنظیم شدت تصویر در Matlab

```
I = imread('pout.tif');
J = imadjust(I, [0.3 , 0.7], [0 ,1]);
figure,
subplot(2,2,1); imshow(I);
subplot(2,2,2); imshow(J);
subplot(2,2,3); imhist(I);
subplot(2,2,4); imhist(J);
```



بهبود تصویر: حذف نویز

- تصاویر دیجیتال کم و بیش دارای نویز هستند.
- حذف نویز قبل از هرگونه عملیات پردازشی باید انجام گیرد.
- فیلترهای متعددی برای حذف نویز طراحی شده‌اند.
- در متلب نیز چندین فیلتر برای حذف نویز وجود دارد که از این میان به ساده‌ترین آنها اشاره خواهیم کرد:
  - فیلتر میانگین
  - فیلتر میانه
- برای ایجاد فیلتر میانگین از تابع `fspecial` که قبلا توضیح داده شد و تابع `filter2` می‌توان استفاده کرد.
- برای اعمال فیلتر میانه از تابع `medfilt2` استفاده کنید.
- بطورکلی تمامی فیلترهای حذف نویز از وضوح (`sharpness`) تصویر می‌کاهند. در میان دو فیلتر میانگین و میانه، فیلتر میانه معمولا نتیجه بهتری ایجاد می‌کند و وضوح تصویر را نیز کمتر تحت تاثیر قرار می‌دهد.

بهبود تصویر: حذف نویز

- مقایسه فیلتر میانه و فیلتر میانگین

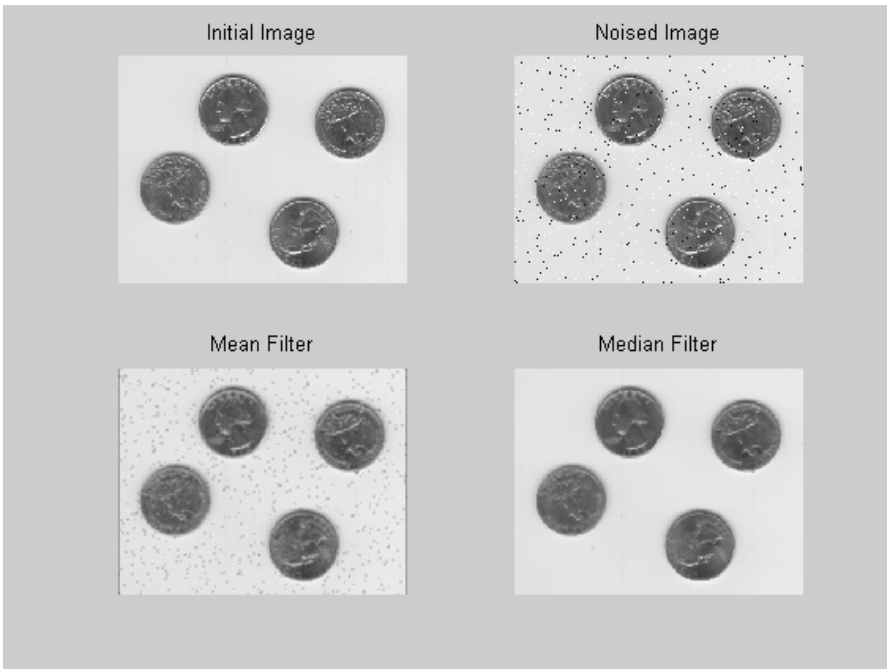
```
I = imread('eight.tif');

% افزودن نویز
J = imnoise(I, 'Salt & pepper', 0.02);

% فیلتر میانگین
K = filter2(fspecial('average', 3), J) / 255;
% فیلتر میانه
L = medfilt2(J, [3, 3]);

subplot(2,2,1); imshow( I ); title('Initial Image')
subplot(2,2,2); imshow( J ); title('Noised Image');
subplot(2,2,3); imshow( K ); title('Mean Filter');
subplot(2,2,4); imshow( L ); title('Median Filter');
```


بهبود تصویر: حذف نویز



عملیات ساختاری (Morphological Operations) روی تصاویر باینری

- عملیات ساختاری به عملیاتی گفته می‌شود که بر روی تصاویر باینری اعمال شده و هدف از آن ایجاد تغییر و یا تصحیح در اجزا داخل یک تصویر باینری باشد.
- این عملیات معمولا یک مرحله قبل از عملیات پردازش نهایی انجام می‌شود.
- منظور از عملیات پردازش نهایی عملیاتی است که در آن اطلاعاتی از تصویر استخراج می‌شود.
- از میان این عملیات چهار عمل مهتر هستند:
  - عملیات افزایش (Dilation)
  - عملیات فرسایش (Erosion)
  - عملیات گشودن
  - عملیات بستن

عملیات ساختاری – افزایش

- عملیاتی است که باعث افزایش ابعاد اجزا داخل تصویر به اندازه یک یا چند پیکسل می‌گردد.
- در اثر این عمل ممکن است نقاطی که از یک تصویر باینری در اثر عواملی چون تاثیر نویز یا اعمال حد آستانه نامطلوب جا افتاده است، تصحیح گردند. مثلا ممکن است دو جزء از تصویر به یکدیگر متصل گردند.
- الگوریتم اعمال فیلتر افزایش بدین صورت است که تمامی نقاط سیاه تصویر بررسی شده در صورتیکه حداقل یکی از همسایگان انتخابی نقطه مورد بررسی سفید باشند، نقطه مزبور نیز سفید خواهد شد در غیر اینصورت سیاه باقی خواهد ماند.

عملیات ساختاری – افزایش

- ابعاد همسایگی و انتخاب همسایه‌ها توسط یک ماتریس ماسک (Mask) مشخص می‌شوند.
- مثلا اگر ماتریس ماسک یک ماتریس ۳*۳ باشد که تمای عناصر آن برابر با ۱ باشد. یعنی یک همسایگی ۳*۳ بکار برده شود و تمامی ۹ همسایه نقطه مورد بررسی برای عملیات افزایش یا فرسایش مد نظر قرار گیرند.

عملیات ساختاری – افزایش

- **imdilate**
  - ایجاد یک دیسک به شکل الماس. همسایگی‌های براساس این دیسک افزایش می‌یابند.

```
se = strel('diamond', 1);
```

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

  - افزایش با استفاده از این دیسک

```
bw2 = imdilate(bw1, se);
```

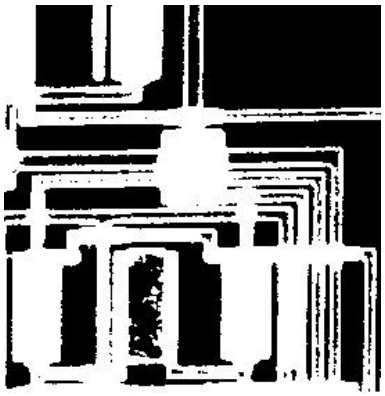
عملیات ساختاری – فرسایش

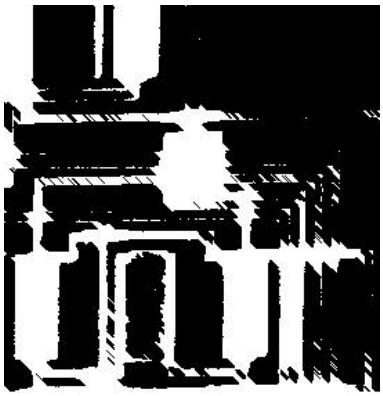
- دقیقا عکس عملیات افزایش است.
- در این عملیات معمولا نقاط ناخواسته تصویر باینری حذف می‌شوند و سایر اجزا تصویر نیز به اندازه یک یا چند پیکسل نازکتر خواهند شد.
- عملا تمامی نقاط سفید تصویر بررسی شده در صورتیکه حداقل یکی از همسایگان انتخابی آن سیاه باشد، آن نقطه نیز سیاه خواهد شد.

```
bw2 = imerode(bw1, se);
```


عملیات ساختاری – فرسایش

```
bw1 = imread('circbw.tif');
SE = eye(5);
bw2 = imerode(bw1, SE);
imshow(bw1);
figure, imshow(bw2);
```





عملیات ساختاری – افزایش/فرسایش

- **bwmorph**
  - هر دوی عملیات افزایش و فرسایش را می‌توان با استفاده از تابع کلی‌تر نیز می‌توان انجام داد.

عملیات ساختاری – گشودن/بستن

- از ترکیب‌های مختلف دو عملیات افزایش و فرسایش می‌توان عملیات دیگری ایجاد کرد.
- مهمترین این عملیات، عملیات گشودن و بستن است.
- در عملیات گشودن اجزایی از تصویر باینری که از یک اندازه تعیین شده کوچکتر باشند حذف می‌شوند بدون آنکه ابعاد سایر اجزا تغییر کند.
- در عملیات بستن نیز نواحی جا افتاده تصویر باینری بدون تغییر در ابعاد سایر اجزا ترمیم می‌گردند.
- عملا در صورتیکه ابتدا عملیات فرسایش و سپس افزایش بر یک تصویر باینری اعمال شود، نتیجه، عملیات گشودن خواهد بود اما اگر ابتدا افزایش و سپس فرسایش اعمال گردد، عملیات بستن حاصل خواهد شد.

عملیات ساختاری – گشودن/بستن

- **bwmorph**
  - برای اعمال عملیات گشودن و بستن می‌توان از این تابع استفاده کرد.
  - می‌توان این دو عملیات را از عملیات فرسایش و افزایش نیز بدست آورد.

عملیات ساختاری – گشودن/بستن

```
bw1 = imread('circbw.tif');
SE = ones(40 , 30);
bw2 = imerode(bw1, se);
bw3 = imdilate(bw2, se);
imshow(bw2);
figure, imshow(bw3);
```

عملیات ساختاری – عملیات از پیش تعریف شده

- **immorph**
  - با استفاده از این تابع می‌توان بسیاری از عملیات ساختاری معروف پردازش تصویر را اعمال نمود.
  - شکل کلی استفاده از این تابع بصورت زیر است:

```
bw2 = bwmorph(bw1, operation, [n]);
```

  - مقدار آرگومان دوم یکی از رشته‌های زیر است:
  - `erode dilate open close fill hbreak skel remove`
  - آرگومان سوم اختیاری بوده و بیانگر ابعاد ماسک مورد استفاده یا فاکتور دیگری با توجه نوع آرگومان دوم در عملیات است.
  - در صورت حذف آرگومان سوم، مقدار پیش فرض آن بکار برده خواهد شد.

عملیات ساختاری – عملیات از پیش تعریف شده

- نتیجه عملیات اسکلتون را بر روی تصویر

```
bw1 = imread('circbw.tif');
bw2 = bwmorph(bw1, 'skel', inf)
imshow(bw1);
figure, imshow(bw2);
```