فرادرس
فراتر از یک کلاس درس
www.faradars.org

جلسه دوم:
کار با تصاویر

مدرس: الهام شعبانی نیا

دانشجوی دکترای هوش مصنوعی

دانشگاه اصفهان

فرادرس
FaraDars.org

مقدمه

- برای ساخت برنامه‌های بینایی کامپیوتر باید بتوانیدبه محتوای تصاویر دسترسی داشته باشید و تصاویر را ویرایش یا ایجاد کنید. این جلسه به شما یاد خواهد داد که چطور عناصر تصویر را (یعنی همان پیکسل‌ها را) دستکاری کنید. شما یاد خواهید گرفت که چطور یک تصویر را پویش کرده و هریک از پیکسل‌هایش را پردازش کنید.
- همچنین یاد می‌گیرید که چطور این کار را به صورت موثر انجام دهید. چرا که حتی تصاویری که ابعاد متوسطی دارند، می‌توانند شامل دهها هزار پیکسل باشند.
- اساسا یک تصویر ماتریسی از مقادیر عددی است. به همین دلیل است که OpenCV 2 آن‌ها را با استفاده از داده ساختار cv : :Matدستکاری می‌کند. هر عنصر ماتریس بیانگر یک پیکسل است. برای یک تصویر خاکستری پیکسلها مقادیر ۸ بیتی بدون علامت هستند که صفر متناظر با سیاه و ۲۵۵ متناظر با سفید است. برای یک تصویر رنگی برای هر پیکسل به سه مقدارنیاز است تا بتوان سه کانال اصلی رنگ (قرمز، سبز، آبی) را نمایش داد. بنابراین در این حالت هر عنصر ماتریس یک سه تایی است.

فرادرس
FaraDars.org

دسترسی به مقادیر پیکسل‌ها

- برای دسترسی به هر عتصر ماتریس، تنها باید شماره سطر و ستون آن را مشخص کنید. آن‌گاه عنصر مربوطه که می‌تواند یک مقدار عددی تنها و یا برای یک تصویر چند کانالی یک بردار از مقادیر باشد، برگردانده می‌شود.

- کلاس cv: :Mat متدهای مختلفی برای دسترسی به ویژگی‌های مختلف یک تصویر دارد.
- متغیرهای عمومی عضو، cols و rows به شما تعداد سطرها و ستون‌های تصویر را می‌دهند.
- برای دسترسی به عنصر cv: : Mat متد at(int y, int x) وجود دارد. با این حال نوع بازگشتی این متد باید در زمان کامپایل مشخص باشد. و از آن‌جا که cv: : Mat می‌تواند عناصری از هر نوع را نگه دارد، برنامه نویس باید نوع بازگشتی مورد انتظار را مشخص نماید. بنابراین زمانی که آن را فراخوانی می‌کنید، باید نوع عنصر تصویر را مشخص کنید. مثلا :

```
image.at<uchar>(j,i)= 255;
```

- از این متد تنها برای دسترسی تصادفی به پیکسل‌های تصویر باید استفاده کرد. اما هرگز نباید برای پویش یک تصویر استفاده شود.

فرادرس
FaraDars.org

دسترسی به مقادیر پیکسل‌ها

- در تصاویر رنگی، هر پیکسل با سه جزء مشخص می‌شود : کانال‌های قرمز، سبز و آبی. بنابراین یک cv: :Mat برای یک تصویر رنگی برداری از سه مقدار ۸ بیتی را باز خواهد گرداند. OpenCV نوع تعریف شده ای به نام cv::Vec3b را برای چنین برداری دارد. که مشخص کننده برداری از سه کارکتر بدون علامت است. به همین دلیل است که دسترسی به پیکسل‌های رنگی به صورت زیر انجام می‌شود :

```
image.at<cv::Vec3b>(j,i)[channel]= value;
```

- اندیس کانال یکی از سه کانال رنگی را مشخص می کند
- انواع مشابهی هم برای بردارهای ۲ عنصری و ۴ عنصری (cv::Vec2b و cv::Vec4b) و برای دیگر انواع وجود دارند. که برای انواع دیگر حرف آخر برای short با s، برای عدد صحیح با i، برای اعشاری با f، و برای عدد اعشاری مضاعف با d جایگزین می‌شود. همه این انواع با استفاده از کلاس الگوی cv::Vec<T,N> تعریف می‌شوند. که T نوع و N تعداد عناصر بردار است.

فرادرس
FaraDars.org

دسترسی به مقادیر پیکسل‌ها

- استفاده از متد at کلاس cv: : Mat گاهی می‌تواند دست و پاگیر باشد. زیرا نوع بازگشتی باید برای هر فراخونی مشخص شود.
- در مواردی که نوع ماتریس مشخص است، می توان از کلاس cv::Mat_استفاده کرد که یک زیرکلاس الگو از cv::Mat است. این کلاس بدون اینکه هیچ ویژگی داده جدیدی اضافه کند، چند متد اضافه تعریف می‌کند. به طوری که اشاره‌گرها یا ارجاعات به یک کلاس بتواند مستقیما به کلاس دیگر تبدیل شوند. در بین این متدهای اضافه، اپراتور() وجود دارد که اجازه دسترسی مستقیم به عناصر ماتریس را می‌دهد. بنابراین اگر imageارجاعی به یک ماتریس از نوع uchar باشد، می‌توان نوشت :

```
cv::Mat_<uchar> im2= image; // im2 refers to image
im2(50,100)= 0; // access to row 50 and column 100
```

- از آن جا که نوع عناصر cv::Mat_ در زمان ایجاد متغیر اعلان می‌شوند، اپراتور () در زمان کامپایل می‌داند که چه نوعی را باز گرداند. در واقع با استفاده از اپراتور () دقیقا همان نتیجه متد at فراهم می‌شود.

فرادرس
FaraDars.org

مثال

- برای اینکه دسترسی مستقیم به مقدار پیکسل‌ها را ببینیم، ما یک تابع ساده ایجاد خواهیم کرد که نویز سیاه و سفید (یا فلفل و نمک) را به تصویر اضافه کند.
- همانطور که از اسمش پیداست، نویز سیاه و سفید نوع خاصی از نویز است که در آن مقدار برخی از پیکسل‌ها با سفید و برخی دیگر با سیاه عوض می‌شود.
- این نوع نویز می‌تواند در اثر خطاهای ارتباطی، زمانی که مقدار برخی از پیکسل‌ها در حین انتقال گم می‌شود، اتفاق بیافتد.
- در این مثال ابتدا به صورت تصادفی چند پیکسل را انتخاب کرده و سپس آن‌ها را سفید می‌کنیم.

فرادرس
FaraDars.org

پویش تصویر با اشاره گرها

فرادرس
FaraDars.org

چرا اشاره گرها

- در بسیاری از کارهای پردازش تصویر، نیاز است تا همه پیکسل‌های تصویر برای انجام یکسری محاسبات پویش شوند.
- با در نظر گرفتن تعداد بسیار زیاد پیکسل‌هایی که باید ملاقات شوند، بسیار ضروری است که این‌کار به شکلی کارا انجام شود.
- این دستورالعمل و دستورالعمل بعدی به شما روش‌های متفاوت پیاده سازی یک حلقه پویش تصویر را نشان خواهند داد.
- در این دستورالعمل از محاسبه اشاره‌گر استفاده می‌کنیم.

فرادرس
FaraDars.org

یادآوری

- در یک تصویر سه بایت اول بافر داده تصویر، مقادیر سه کانال رنگی پیکسل بالا چپ را می‌دهد، سه بایت بعدی مقادیر پیکسل دوم سطر اول و الی آخر.
- دقت کنید که OpenCV به طور پیش‌فرض ترتیب BGR کانال‌ها را استفاده می‌کند، بنابراین معمولا اولین کانال، کانال آبی است) .
- تصویری به عرض W و ارتفاع H به بلوک حافظه ای به اندازه WxHx3 کارکتر نیاز دارد. با این حال ممکن است برای کارایی بیشتر به طول یک سطر چند پیکسل بیشتر اضافه شود. این به این دلیل است که برخی از تراشه‌های پردازنده مالتی مدیا (برای مثال، معماری MMX اینتل) زمانی که سطرها مضربی از ۴ یا ۸ هستند، می‌توانند تصاویر را بسیار کاراتر پردازش کنند. واضح است که این پیکسل‌های اضافه نمایش داده نمی‌شوند و یا ذخیره نمی‌شوند. و مقدار دقیق آن‌ها نادیده گرفته می‌شود.

فرادرس
FaraDars.org

چند نکته

- ویژگی داده cols، عرض تصویر (تعداد ستون‌ها) و ویژگی rows ارتفاع تصویر را به شما می‌دهند. در حالی که ویژگی داده step پهنای موثر بر حسب بایت را می‌دهد.
- حتی اگر تصویر شما از نوعی غیر از uchar است، باز هم step تعداد بایت های یک سطر را به شما می‌دهد.
- اندازه یک پیکسل توسط متد elemSize بدست می‌آید(برای مثال برای یک ماتریس سه کاناله از نوع short integer (CV_16SC3)، متد elemSize عدد ۶ را برمی‌گرداند).
- تعداد کانال‌ها در تصویر توسط متد nchanels بدست می‌آید(که برای تصویر خاکستری یک و برای یک تصویر رنگی سه است).
- در نهایت متد total تعداد کل پیکسل‌های ماتریس را برمی‌گرداند.

فرادرس
FaraDars.org

چند نکته

- بنابراین تعداد پیکسل‌های یک سطر از فرمول زیر بدست می‌آید:

```
int nc= image.cols * image.channels();
```

- برای ساده کردن محاسبات اشاره‌گری، کلاس cv: : Mat متدی را ارائه می‌دهد که مستقیما به شما آدرس یک سطر تصویر را می‌دهد. این متد که ptr نام دارد، یک متد الگو است که آدرس سطر j ام را برمی‌گرداند:

```
uchar* data= image.ptr<uchar>(j);
```

فرادرس
FaraDars.org

مثال

- بیایید یک تصویر رنگی ۲۵۶ در ۲۵۶ ایجاد کرده و
- مقادیر پیکسل هایی که اندیس سطر و ستون آنها بین ۰ تا ۸۰ باشد را برابر (0,0,255)
- مقادیر پیکسل هایی که اندیس سطر و ستون آنها بین ۸۰ تا ۱۶۰ باشد را برابر (0,255,0)
- مقادیر پیکسل هایی که اندیس سطر و ستون آنها بین ۱۶۰ تا ۲۵۶ باشد را برابر (255,0,0) قرار دهیم.

فرادرس
FaraDars.org

پویش تصویر با تکرارگرها

فرادرس
FaraDars.org

مقدمه

- در برنامه نویسی شیئ گرا، حلقه زدن روی یک مجموعه داده اغلب با استفاده از تکرارگرها انجام می‌شود.
- تکرارگرها کلاس‌های خاصی هستند که برای مرور تک تک عناصر یک مجموعه ساخته می‌شوند. به طوری که جزئیات چکونگی تکرار روی هر عنصر برای یک مجموعه داده را می‌پوشانند.
- این کاربرد اصل پنهان سازی اطلاعات، پویش یک مجموعه را ساده‌تر می‌کند.
- علاوه بر این فرم پویش مجموعه ها را بدون توجه به اینکه چه نوع مجموعه ای در حال استفاده است، به یک شکل مشابه تبدیل می‌کند.
- OpenCV یک کلاس تکرارگر cv: : Mat مطابق با تکرارگرهای استاندارد C++ ارائه می‌دهد.
- هدف اصلی استفاده از تکرارگرها ساده سازی فرآیند پویش تصویر و کاهش احتمال خطا ست. و اساسا برای بهینه سازی این فرآیند نیست.

FaraDars.org

مقدمه

- یک شیئ تکرارگر برای یک cv:: Mat را می‌توان در ابتدا با ایجاد یک cv::MatIterator_ object بدست آورد.
- خط فاصله به معنای این است که این یک متد الگوست. در حقیقت از آن‌جا که از تکرارگرهای تصویر برای دسترسی به عناصر تصویر استفاده می‌شوند، نوع بازگشتی بایستی در زمان کامپایل مشخص باشد. سپس تکرارگر به صورت زیر اعلان می‌شود:

```
cv::MatIterator_<cv::Vec3b> it;
```

فرادرس
FaraDars.org

مقدمه

- و یا اینکه می‌توانید نوع iterator تعریف شده در کلاس Mat_template را هم استفاده کنید:

```
cv::Mat_<cv::Vec3b>::iterator it;
```

- آنگاه روی پیکسل‌ها با استفاده از متدهای تکرارگر begin وend معمولی حلقه می‌زنید.

فرادرس
FaraDars.org

مقدمه

کار با تکرارگرها صرفنظر از اینکه چه نوع مجموعه‌ای را پویش می‌کنیم، همیشه از یک مدل پیروی می‌کند.

۱

ابتدا شیئ تکرارگر خود را با استفاده از کلاس مناسب که در مثال ما cv::Mat_<cv::Vec3b>::iterator (یا cv::MatIterator_<cv::Vec3b>) است، ایجاد می‌کنید.

۲

سپس با استفاده از متد begin تکرارگری که در نقطه شروع قرار دارد را بدست می‌آورید (در مثال ما، گوشه بالا چپ تصویر). برای یک مورد cv: : Matاین کار به صورت image.begin<cv::Vec3b>() انجام می‌شود.

۳

موقعیت پایانی مجموعه شما هم به صورت مشابه اما با استفاده از متد end بدست می‌آید.

فرادرس
FaraDars.org

مقدمه

- پس از مقداردهی اولیه تکرارگر، یک حلقه ایجاد می‌کنید به طوری که روی همه عناصر تا رسیدن به end تکرار شود. یک حلقه while نوعی به شکل زیر خواهد بود :

```
while (it!= itend) {

   // process each pixel --------------------
    …
   // end of pixel processing ----------------
   ++it;
}
```

- از اپراتور ++ برای حرکت به عنصر بعدی استفاده می‌شود. همچنین می‌توانید از اندازه گام بلندتری استفاده کنید. برای مثال it+=10 هر ده پیکسل را مورد پردازش قرا ر می‌دهد.
- در نهایت درون حلقه پردازش، می‌توانید از اپراتور * برای دسترسی به عنصر کنونی استفاده کنید. که با استفاده از این اپراتور می‌توانید عنصر را خوانده (برای مثال element= *it) و یا بنویسید (برای مثال *it= element;).

فرادرس
FaraDars.org

مثال

- یک تصویر را با استفاده از تکرارگرها پویش کرده و مقادیر کانال های مختلف آن را تغییر دهید.

فرادرس
FaraDars.org

دسترسی به پیکسل های مجاور در هنگام پویش تصویر

فرادرس
FaraDars.org

مقدمه

- در پردازش تصویر، توابع پردازشی که یک مقدار را در هر پیکسل بر اساس مقادیر پیکسل های همسایه محاسبه می کنند، بسیار متداول هستند.
- زمانی که این پیکسل های همسایه شامل پیکسل های سطرهای قبل و بعد نیز هستند، باید بتوانید همزمان چند سطر تصویر را پویش کنید. این دستورالعمل به شما نحوه انجام کار را نشان می دهد.

فرادرس
FaraDars.org

مثال

- برای بیان این عمل، ما از یک تابع پردازشی که تصویر را واضح (sharp) می کند، استفاده می کنیم. اینکار براساس عملگر لاپلاسین است. در حقیقت اگر لاپلاسین یک تصویر را از خود تصویر کم کنید، لبه های تصویر تقویت شده و تصویر واضح تری تولید می شود. این عملگر واضح سازی به صورت زیر محاسبه می شود:

```
sharpened_pixel= 5*current-left-right-up-down;
```

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

فرادرس
FaraDars.org

انجام محاسبات ساده تصویر

فرادرس
FaraDars.org

cv:: add

- همه توابع محاسباتی به روش مشابهی عمل میکنند. دو ورودی مهیا شده و یک پارامتر سوم خروجی را مشخص میکند. در برخی مواقع میتوان اوزانی را مشخص نمود که به عنوان ضرایب اسکالردر عملگر مورد استفاده قرار میگیرند.

```
// c[i]= a[i]+b[i];
cv::add(imageA,imageB,resultC);
// c[i]= a[i]+k;
cv::add(imageA,cv::Scalar(k),resultC);
// c[i]= k1*a[1]+k2*b[i]+k3;
cv::addWeighted(imageA,k1,imageB,k2,k3,resultC);
// c[i]= k*a[1]+b[i];
cv::scaleAdd(imageA,k,imageB,resultC);
```

فرادرس
FaraDars.org

cv:: add

- برای برخی از توابع می‌توانید ماسک هم تعریف کنید:

```
// if (mask[i]) c[i]= a[i]+b[i];
cv::add(imageA,imageB,resultC,mask);
```

- اگر ماسک را اعمال کنید، عملیات تنها روی پیکسل‌هایی که برای آن‌ها مقدار ماسک نال نیست، انجام می‌شود(ماسک باید تک کانال باشد).
- در همه حالات تابع cv::saturate_cast همیشه برای اطمینان از اینکه نتایج در حوزه مقدار تعریف شده باقی می ماند، مورد استفاده قرار می‌گیرد(یعنی برای اجتناب از سرریز و underflow).

فرادرس
FaraDars.org

دیگر توابع محاسباتی

- به فرم‌های مختلف توابع cv::subtract ،cv::absdiff ،cv::multiply و cv::divide هم نگاهی بیاندازید.
- عملگرهای بیتی نیز موجود هستند : cv::bitwise_or, cv::bitwise_and, cv::bitwise_xor و cv::bitwise_not .
- عملگرهای cv::min و cv::max هم که مقادیر بیشینه و کمینه پیکسل های متناظر را پیدا می‌کنند نیز بسیار مفید هستند.

فرادرس
FaraDars.org

دیگر توابع محاسباتی

- تصاویر بایستی اندازه و نوع یکسانی داشته باشند. همچنین از آن جا که عملیات برای هر عنصر انجام می‌شود، یکی از تصاویر ورودی می تواند به عنوان خروجی مورد استفاده قرار گیرد.
- تعدادی عملگر نیز که یک تصویر را به عنوان ورودی می‌گیرند، نیز وجود دارند:
  cv::sqrt
  cv::pow
  cv::abs
  cv::exp
  و cv::log
- در واقع برای تقریبا هر کاری که باید روی تصاویر اعمال شود، یک تابع OpenCV وجود دارد.

فرادرس
FaraDars.org

سربارگذاری عملگرها

- در OpenCV 2 بسیاری از توابع محاسباتی عملگر متناظر با خود دارند که سربارگذاری شده اند. در نتیجه فراخوانی cv::addWeighted را می توان به صورت زیر نوشت:

```
result= 0.7*image1+0.9*image2;
```

- بسیاری از عملگرهای C++ سربارگذاری شده هستند.

فرادرس
FaraDars.org

سربارگذاری عملگرها

- در بین آن‌ها عملگرهای بیتی ~, ^, |, &، توابع min, max و abs و عملگرهای مقایسه ای <, <=, ==,!=, >, >= وجود دارند. که این دسته آخر یک تصویر ۸ بیتی باینری برمی گرداند.

- همچنین می توانید عملگرهای ضرب ماتریسی m1*m2، وارون ماتریس m1.inv()، ترانهاده ماتریس m1.t()، دترمینان m1.determinant() ، نرم برداری v1.norm() ,ضرب خارجی v1.cross(v2)،ضرب داخلی v1.dot(v2) و از این قبیل را می‌توانید داشته باشید.

- همچنین می توانید عملگر op= (مثلا +=) را تعریف کنید.

فرادرس
FaraDars.org

جدا کردن کانال‌های تصویر

فرادرس
FaraDars.org

جدا کردن کانال‌های تصویر

- خیلی وقت‌ها می‌خواهیم کانال‌های متفاوت یک تصویر را به صورت مستقل پردازش کنیم.
- برای مثال ممکن است بخواهید که کاری را تنها روی یک کانال تصویر انجام دهید. قطعا می‌توانید این کار را از طریق یک حلقه پویش تصویر هم انجام دهید.
- اما می‌توانید از تابع cv::split هم که سه کانال یک تصویر رنگی را در سه نمونه مجزای cv::Mat کپی می‌کند، را هم استفاده کنید.

- تابع cv::merge دوگان عمل قبل است. یعنی این تابع یک تصویر رنگی را از سه تصویر تک کانال می‌سازد.

فرادرس
FaraDars.org

مثال

- فرض کنید می خواهیم تصویر بارانی خود را تنها به کانال آبی تصویر اضافه کنیم. در این صورت به روش زیر باید عمل کرد:

مثال

- فرض کنید می خواهیم تصویر بارانی خود را تنها به کانال آبی تصویر اضافه کنیم. در این صورت به روش زیر باید عمل کرد:

```
// create vector of 3 images
std::vector<cv::Mat> planes;
// split 1 3-channel image into 3 1-channel images
cv::split(image1,planes);
// add to blue channel
planes[0]+= image2;
// merge the 3 1-channel images into 1 3-channel image
cv::merge(planes,result);
```

فرادرس
FaraDars.org

تعریف نواحی مورد علاقه

فرادرس
FaraDars.org

ROI

- گاهی اوقات یک تابع پردازشی بایستی تنها روی یک بخش تصویر اعمال شود. در اینجا به شما نشان می‌دهیم که چگونه یک ناحیه مورد علاقه را درون یک تصویر تعریف کنید.

- اولین مرحله شامل تعریف ROI است. با یکبار تعریف ROI، می‌توان با آن مانند یک نمونه cv::Mat معمولی کار کرد. نکته مهم آن است که نقاط ROI به همان بافر داده به عنوان تصویر والد اشاره می‌کنند.

فرادرس
FaraDars.org

ROI

- ناحیه مورد علاقه یک ناحیه مستطیلی را با مشخص کردن موقعیت گوشه بالا- چپ (دو پارامتر اول سازنده آن) و اندازه مستطیل (طول و عرض دو پارامتر بعدی هستند)توصیف می‌کند.
- یک راه تعریف ROI، استفاده از یک مورد cv::Rect است.

```
// define image ROI
cv::Mat imageROI;
imageROI= image(cv::Rect(385,270,logo.cols,logo.rows));
```

فرادرس
FaraDars.org

ROI

- همچنین ROI می‌تواند با استفاده از محدوده های سطر وستون نیز توصیف شود. یک محدوده دنباله ای پیوسته است که از یک اندیس شروع و به اندیسی دیگر خاتمه می‌یابد(اندیس آخر جزء محدوده محسوب نمی‌شود). ساختار cv::Range برای نمایش این مفهوم استفاده می‌شود. بنابراین یک ROI می‌تواند از دو محدوده تعریف شود.

```
cv::Mat imageROI= image(cv::Range(270,270+logo.rows),
                        cv::Range(385,385+logo.cols))
```

فرادرس
FaraDars.org

ROI

- هر تغییری در ROI روی تصویر اولیه در ناحیه متناظر تاثیر می‌گذارد. چرا که تصویر و ROI از داده تصویر به صورت اشتراکی استفاده می‌کنند. تعریف یک ROI داده ای را کپی نمی‌کند.
- اگر بخواهید یک ROI متشکل از چند خط یک تصویر را تعریف کنید، فراخوانی زیر می‌تواند استفاده شود:

```
cv::Mat imageROI= image.rowRange(start,end) ;
```

- و به صورت مشابه برای یک ROI متشکل از چند ستون تصویر :

```
cv::Mat imageROI= image.colRange(start,end) ;
```

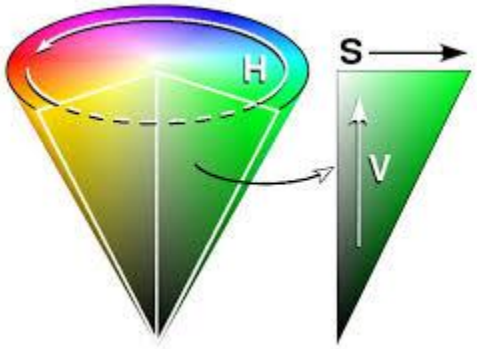
فرادرس
FaraDars.org

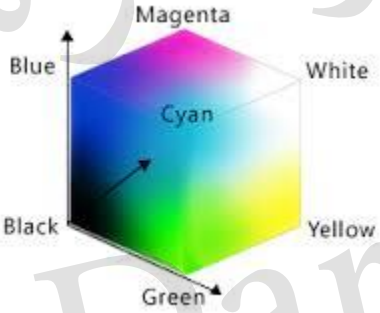
مثال

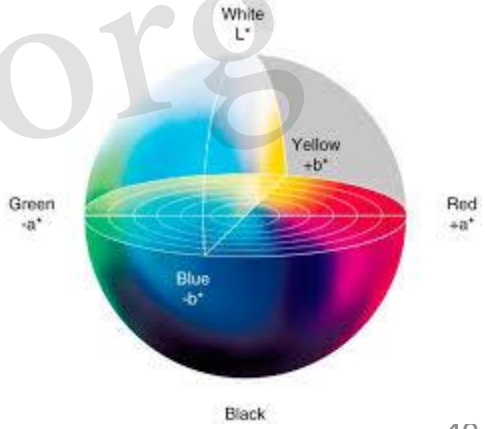
درج لوگو بر روی تصویر

فرادرس
FaraDars.org

تبدیل فضاهای رنگ



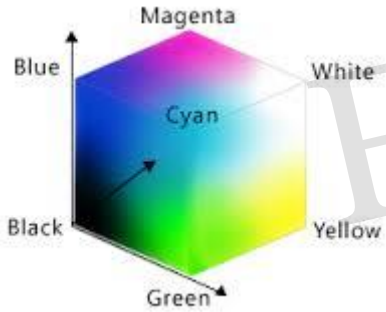




فرادرس
FaraDars.org

حوزه رنگی RGB

- اساس حوزه رنگی RGBاستفاده از سه مؤلفه رنگی اصلی؛ قرمز، سبز و آبی می‌باشد. این سه رنگ به این دلیل انتخاب شده‌اند که با ترکیب آنها می‌توان هر رنگ دلخواهی را ایجاد نمود.
- در حقیقت سیستم بینایی انسان نیز از سلول‌های مخروطی شکلی استفاده می‌کند که نسبت به طیف‌های رنگی قرمز، سبز و آبی حساس است.
- این سیستم رنگی، سیستم پیش‌فرض تصاویر دیجیتالی می‌باشد به این دلیل که تصاویر با استفاده از این سه مؤلفه از محیط گرفته می‌شوند. در زمان گرفتن تصویر از محیط، نور دریافت شده با استفاده از فیلترهای رنگی به سه مؤلفه قرمز، سبز و آبی تجزیه شده و ذخیره می‌شوند.



فرادرس
FaraDars.org

مشکلات RGB

- متأسفانه تفاضل رنگ‌ها در حوزه RGBراهکار مناسبی برای اندازه‌گیری میزان شباهت دو رنگ نمی‌باشد. در واقع این حوزه رنگی از **یکنواختی ادراکی** برخوردار نمی‌باشد. این به این معنی است که دو رنگ با تفاضل مشخص، ممکن است به یکدیگر شبیه باشند در صورتی که دو رنگ دیگر با همان میزان تفاضل هیچ‌گونه شباهتی به یکدیگر نداشته باشند.
- در حوزه RGB، مولفه روشنایی از مولفه رنگ جدا نیست.
- با توصیف انسان از رنگ ها مغایرت دارد.
- به منظور حل این مشکل فضاهای رنگی دیگری همچون مثلا HSV،CIE L*a*b*، ... معرفی شده اند که هر یک دارای خواص متفاوتی هستند.

فرادرس
FaraDars.org

فضاهای رنگی دیگر

- حوزه رنگی HSVو HLSدوحوزه رنگی متداول دیگری هستند که مهم‌ترین خاصیت آنها این است که رنگ‌ها رابه صورت ۳ مؤلفه؛ رنگ، درجه اشباع و میزان روشنایی که انسان نیز برای توصیف رنگ‌ها ازآنها استفاده می‌کند، ارائه می‌دهند.

- فضای رنگی دیگری معرفی شده است که دارای خاصیت یکنواختی ادراکی می‌باشد. این فضای رنگی CIE L*a*b*نام دارد. با تبدیل تصاویر به این حوزه رنگی، تفاضل اقلیدسی بین رنگ پیکسل‌ها و رنگ هدف کاملاً معنادار بوده و این تفاضل معرف میزان شباهت بین دو رنگ خواهد بود.

- یکی دیگر از این حوزه‌های رنگی YCrCbاست که در ذخیره سازی تصاویر با فرمت JPEGمورد استفاده قرارمی‌گیرد

فرادرس
FaraDars.org

cvtColor

- تبدیل در حوزه رنگ به سادگی و توسط تابع cv::cvtColor کتابخانه OpenCV قابل انجام است. مثلا بیایید تا حوزه رنگی RGB تصویر ورودی را به CIE L*a*b* تبدیل نماییم:

```
cv::cvtColor(image, converted, CV_BGR2Lab);
```

فرادرس
FaraDars.org

cvtColor

- نوع پیکسل‌ها در تصویر خروجی همانند نوع پیکسل‌ها در تصویر ورودی خواهد بود. اما باید توجه داشت که همواره دامنه مقادیر پیکسل‌ها به حوزه رنگی تصویر وابسته است. برای مثال در تصاویر با حوزه رنگی CIE L*a*b* مقادیر پیکسل‌ها درکانال Lدربازه ۰ تا ۱۰۰ ودرکانال‌های aو bدربازه ۱۲۷- تا ۱۲۷ می‌باشد.
- کتابخانهOpenCV از اکثرحوزه‌های رنگی پشتیبانی می‌کند و تنها در زمان تبدیل لازم است پارامتر مناسب به تابع تبدیل داده شود.
- همچنین می‌توان تصویر را به فرم خاکستری تبدیل نمود که در نتیجه تصویر خروجی شامل یک کانال خواهدبود.

```
cv::cvtColor(color, gray, CV_BGR2Gray);
```

فرادرس
FaraDars.Org

این اسلاید ها بر مبنای نکات مطرح شده در فرادرس
«آموزش پردازش تصویر با OpenCV»
تهیه شده است.

برای کسب اطلاعات بیشتر در مورد این آموزش به لینک زیر مراجعه نمایید.

faradars.org/fvimg9405

فرادرس
FaraDars.org