

به نام خدا

## مقدمه ای بر پردازش تصویر

نویسنده

محمد نحوی

### کلمات کلیدی

پردازش تصویر، تشخیص الگو، لبه گیری، مرکز ثقل.

### چکیده

در این مقاله مقدمه بر پردازش تصویر ارائه شده است. در این مقاله مبانی پردازش تصویر و چند الگوریتم کاربردی در زمینه های لبه گیری و تشخیص الگو ارائه شده است.



## مقدمه :

در هر سیستمی و با هر عملکردی برای تصمیم گیری به داده های ورودی احتیاج داریم. این ورودی ها می تواند از یک سنسور صوتی ، سنسور فاصله سنج ، سنسور مادون قرمز ، میکروفن یا تصاویر ارسالی از یک دوربین باشند .

امروزه پردازش تصویر بهترین ابزار برای استخراج ویژگی ها و تحلیل موقعیت و در نهایت تصمیم گیری صحیح می باشد . در مورد انسان نیز به همین صورت است ، اطلاعات از طریق چشم به مغز ارسال می شود و مغز با پردازش این اطلاعات تصمیم نهایی را گرفته و فرمان را صادر می کند .

هدف از پردازش تصویر پیاده سازی عملکرد ذهن انسان در قبال داده های دریافتی از چشم نمی باشد زیرا این عملیات بسیار پیچیده و حجیم است ، بلکه هدف پیش پردازش ها و انجام پردازش های خاصی برای استخراج ویژگی مورد نیاز برای رسیدن به اهداف از پیش تعیین شده می باشد.

## ۱. چند تعریف

		<b>j</b>				
<b>i</b>		0,0	0,1	0,2	0,3	0,4
		1,0	1,1	1,2	1,3	1,4
		2,0	2,1	2,2	2,3	2,4
		3,0	3,1	3,2	3,3	3,4
		4,0	4,1	4,2	4,3	4,4

**(i,j)**

### ۱-۱. پیکسل

کوچکترین جزء یک تصویر PIXEL است. پیکسل مخفف Picture element به معنی المان تصویر است. یک تصویر متشکل از تعداد زیادی پیکسل است که در کنار یکدیگر قرار گرفته اند. در واقع زمانی که با یک دوربین دیجیتال عکس می گیرید اگر رزولوشن دوربین شما  $640 \times 480$  باشد به این معنی است که ماتریس با ابعاد  $640 \times 480$  در اختیار شماست که ۶۴۰ پیکسل در طول و ۴۸۰ پیکسل در عرض دارد. به ازای هر پیکسل یک سلول نوری در دوربین وجود دارد. شدت نور این سلول نوری مقدار عددی را برای این پیکسل تعیین می کند به طور مثال به ازای رنگ سیاه مقدار صفر در پیکسل ذخیره می شود و به ازای رنگ سفید مقدار ۲۵۵ در آن ذخیره می شود.

همسایگی :

۲ نوع همسایگی وجود دارد :

۱ - همسایگی ۸ تایی

۲ - همسایگی ۴ تایی

همسایگی ۸ تایی به صورت

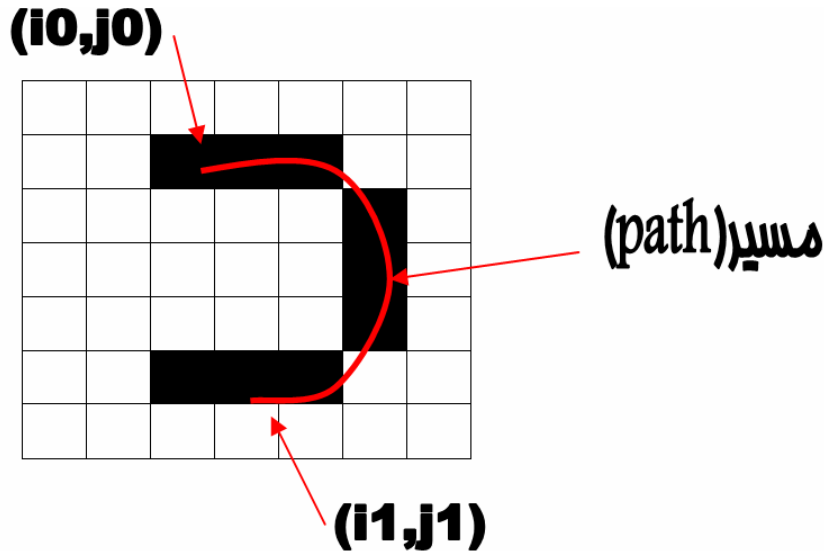
$(i-1,j-1), (i,j-1), (i+1,j-1), (i-1,j), (i+1,j), (i-1,j+1), (i,j+1), (i+1,j+1)$

و همسایگی ۴ تایی به صورت

$(i-1,j), (i+1,j), (i,j-1), (i,j+1)$

## ۱-۲. مسیر (path) :

۲ پیکسل را در یک تصویر باینری در نظر بگیرید . اگر راهی بین این ۲ پیکسل از طریق پیکسل هایی که حاوی آبجکت هستند وجود داشته باشد ، به پیکسل های رابط مسیر گفته می شود .



## ۱-۳. زمینه

به مجموعه پیکسل هایی که شامل آبجکت نیستند ، یا مجموعه پیکسل هایی که شامل  $\bar{S}$  هستند گفته میشود .

اتصال :

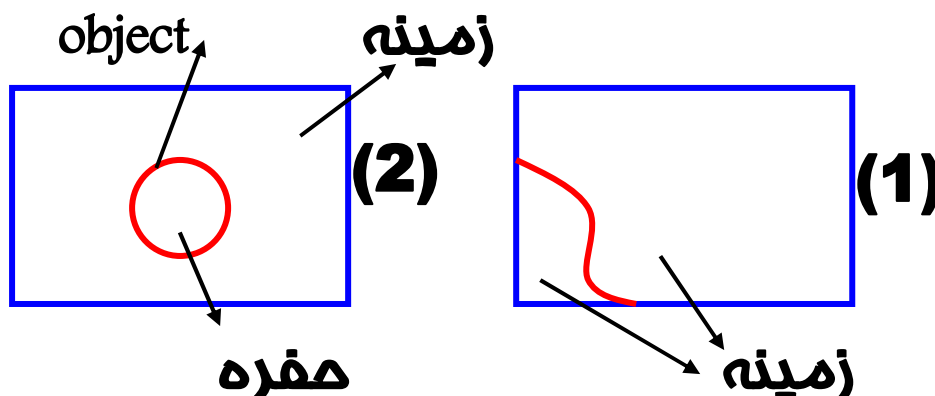
اگر  $S$  مجموع پیکسل هایی باشد که شامل آبجکت هستند پس  $\bar{S}$  شامل پیکسل های بک گراند است .  
 ۲ پیکسل  $p$  و  $q$  که هر ۲ متعلق به  $S$  هستند متصل به یکدیگر هستند اگر حداقل یک مسیر بین  $p$  و  $q$  وجود داشته باشد .

مرز تصویر (border) :

کلیه پیکسل هایی که در اطراف آبجکت قرار گرفته اند .

## ۱-۴. حفره

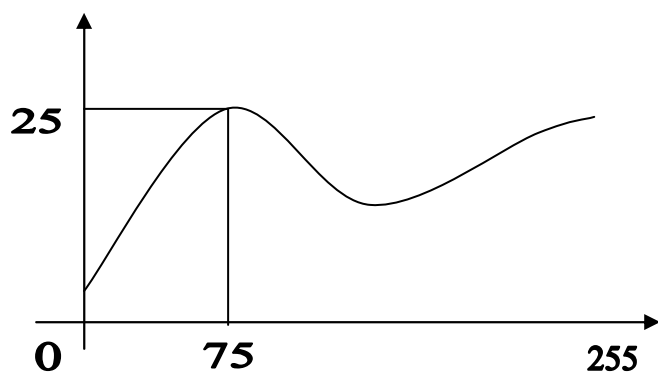
مجموعه ای از  $\bar{S}$  که حتی یک پیکسل هم در border نداشته باشد ، حفره نامیده می شود .



در شکل یک زمینه ۱ تکه است .

## ۱-۵. هیستوگرام

هیستوگرام در واقع همان نمودار میله ای افزونی هر رنگ می باشد .



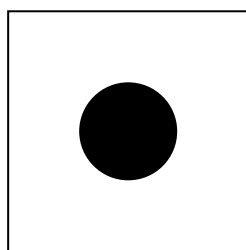
به عنوان مثال تعداد پیکسل هایی که حاوی رنگ شماره ۷۵ هستند ۲۵ عدد است . از هیستوگرام برای آستانه گیری استفاده می شود .

آستانه گیری (threshold):

فرض کنید می خواهید از یک عکس رنگی ۱۶ بیتی یک تصویر سیاه و سفید بسازید . برای این کار باید تک تک پیکسل ها را با مقدار مشخصی مقایسه کنید اگر بزرگتر از آن مقدار مشخص بود به جای آن رنگ سفید و اگر کوچکتر بود به جای آن مقدار سیاه را قرار می دهیم . به این کار آستانه گیری یا threshold می گویند .

محاسبه مساحت و محیط:

فرض کنید که تصویری از یک دایره در یک زمینه سفید رنگ دارید مانند شکل زیر



برای بدست آوردن مساحت این دایره فقط کافی است که تعداد پیکسل هایی که مقدار عددی آن صفر است را بشمارید و آن را کالیبره کنید ( به این معنی که با توجه به فاصله دوربین تا آبجکت ، نسبتی بین تعداد پیکسل های واقعی آبجکت و تعداد پیکسل های تصویر بدست آورد) . به این ترتیب به راحتی می توان مساحت هر آبجکتی را بدست آورد حتی اگر منتظم نباشد .

برای بدست آوردن محیط دایره می توان ابتدا شعاع دایره را با استفاده از مساحت آن به دست آورد (  $R = \sqrt{S/\pi}$  سپس با استفاده از فرمول  $P=2\pi R$  محیط را محاسبه کرد.

نحوه بدست آوردن مرکز تصویر و آبجکت :

برای بدست آوردن مرکز تصویر از روش زیر استفاده می شود :

$$I = \frac{I_b - I_e}{2} \quad J = \frac{J_b - J_e}{2}$$

به این معنی که تعداد پیکسل های ماتریس در سطر و تعداد پیکسل های ماتریس در ستون تقسیم بر ۲ دقیقاً  $i, j$  پیکسل مرکز تصویر خواهد بود .

برای بدست آوردن مرکز آبجکت باید به صورت زیر عمل می کنیم :

مساحت آبجکت رو بدست می آوریم . برای به دست آوردن مساحت تعداد پیکسل های آبجکت را جمع می کنیم . حال برای بدست آوردن  $I$  مرکز آبجکت مقدار  $i$  های پیکسل ها را با هم جمع و تقسیم به مساحت می کنیم . برای بدست آوردن  $J$  مرکز آبجکت مقدار  $j$  های پیکسل ها را با هم جمع و تقسیم به مساحت می کنیم .

به عنوان مثال برای بدست آوردن مرکز آبجکت زیر:

۰,۰	0,1	0,2	0,3	0,4	0,5	0,6	0,7
1,0							1,7
2,0							2,7
3,0							3,7
4,0							4,7
5,0							5,7
6,0							6,7
7,0	7,1	7,2	7,3	7,4	7,5	7,6	۷,۷

مساحت آبجکت زیر  $S=9$  است و برای بدست آوردن  $I$  مرکز آبجکت :

$$\frac{(2+3+4+2+3+4+2+3+4)}{S} = I = 3$$

و برای بدست آوردن  $J$  مرکز آبجکت :

$$\frac{(4+5+6+4+5+6+4+5+6)}{S} = J = 5$$

همانطور که در شکل مشاهده می کنید مرکز آبجکت (۳ و ۵) است .

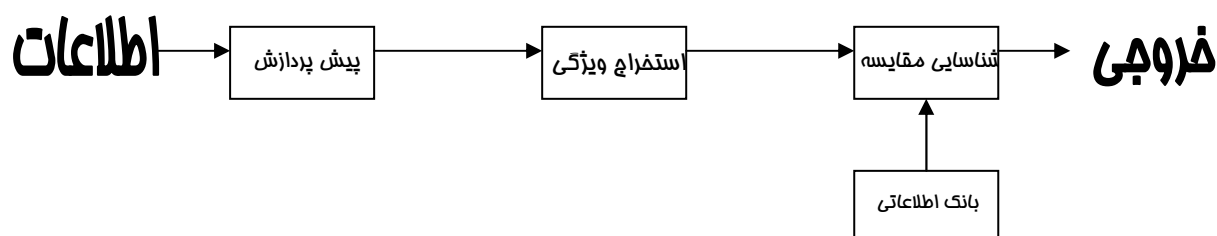
برای بدست آوردن فاصله آبجکت تا مرکز تصویر یا تا مرکز آبجکت دیگر از فرمول آشنای زیر استفاده می شود .

$$d = \sqrt{(i_1 - i_0)^2 + (j_1 - j_0)^2}$$

## ۲. تشخیص الگو (pattern recognition)

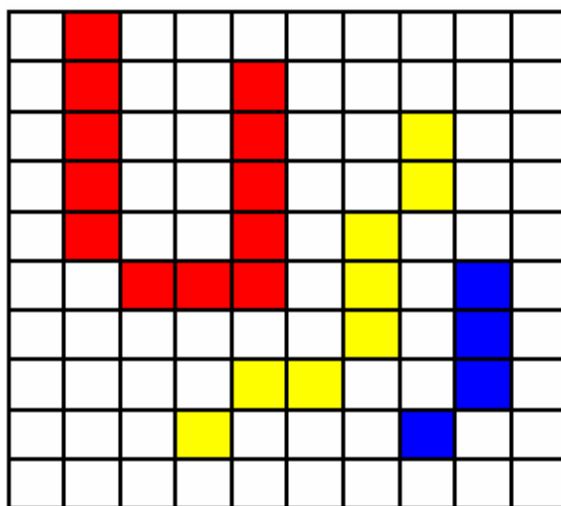
فرض کنید می خواهید تشخیص دهید که آبجکتی که در تصویر است دایره است یا خیر . به این عمل که توسط کامپیوتر انجام می شود تشخیصی الگو گفته می شود .  
برای این منظور باید به دنبال ویژگی منحصر به فردی در دایره ها باشیم . به عنوان مثلا در مغز انسان ، دایره مکان هندسی نقاطی است که فاصله آنها از یک نقطه مشخص یکسان است . انسان نیز برای تشخیص هر شی یک ویژگی مشخص از آن در ذهن دارد .  
در دایره این ویژگی منحصر به فرد  $P2/S=4R$  است . البته باید به این نکته توجه کرد که منحصر به فرد بودن این ویژگی در اشکال هندسی است و در ابجکت های بی شکل صادق نیست .

روند تشخیص الگو ها در کامپیوتر به صورت زیر است :



## ۲-۱. شماره گذاری آبجکت های درون تصویر

برای تشخیص اینکه چه تعداد آبجکت در تصویر وجود دارد و هر کدام از پیکسل ها متعلق به کدام آبجکت است از روش های مختلفی استفاده می شود و به این عمل component labeling یا شماره گذاری اجزاء گفته می شود .  
برای اختصار آخرین و سریع ترین الگوریتم موجود را بررسی می کنیم :



ماتریسی که در بالا مشاهده می کنید یک تصویر به فرض سیاه و سفید است (رنگ هایی که در تصویر مشاهده می کنید برای جدا کردن آبجکت های است) در خانه هایی که آبجکتی وجود ندارد مقدار صفر و در پیکسل هایی که آبجکت وجود دارد عدد یک وجود دارد.

از سطر اول شروع می کنیم و انقدر در سطر حرکت می کنیم تا به اولین پیکسلی که حاوی مقدار ۱ است برسیم، این پیکسل را به عنوان اولین آبجکت در نظر می گیریم. کار را ادامه می دهیم در سطر بعد به اولین مقدار ۱ که رسیدیم همسایه های ۸ گانه آنرا نگاه می کنیم اگر پیکسلی را قبلا نشانه گذاری کرده بوده ایم چون مسیری بین این پیکسل و پیکسل قبلی وجود دارد پس این ۲ پیکسل هر دو از یک آبجکت هستند پس به این پیکسل نیز برچسب همان پیکسل قبلی را می زنیم. کار را ادامه می دهیم به پیکسلی می رسیم که در شکل مشخص است که متعلق به آبجکت یک است ولی در همسایگی آن پیکسلی برچسب گذاری نشده است. بدون در نظر گرفتن این موضوع برچسب جدیدی را به آن می زنیم. کار را به همین صورت ادامه می دهیم تا پایان ماتریس برسیم.

نکته قابل توجه این است که آبجکتی هایی که با ۲ برچسب مشخص شده اند در جایی این ۲ پیکسل به هم می رسند. در بار دوم که ماتریس تصویر را ریفرش می کنیم به محل تلاقی برچسب ها که رسیدیم مقدار برچسب کوچکتر را در برچسب های بزرگتر قرار می دهیم. حال فقط با ۲ بار ریفرش کردن تصویر به فرض سیاه و سفید بودن تصویر می توان تمامی آبجکت ها را مشخص و جدا کرد.

## ۲-۱. فاصله بین ۲ آبجکت

مینیمم پیکسل هایی موجود بین ۲ آبجکت.

کاربرد این کار به عنوان مثال سرعت سنج اتومبیل ها در اتوبان ها است.

## ۲-۱-۱. تشخیص لبه در یک تصویر

دلیل ایجاد لبه در یک تصویر اختلاف شدت نور در ۲ طرف لبه است. لبه ها قسمتی از تصویر هستند که استخوان بندی تصویر را مشخص می کند.



## ۲-۲. انواع لبه

۱- لبه عمودی

۲- لبه افقی

۳- لبه مورب

برای یافتن یک لبه عمودی به روش زیر استفاده می کنیم :

از سطر اول به صورت افقی شروع به حرکت می کنیم و هر پیکسل را با پیکسل قبل از خودش مقایسه می کنیم اگر تفاوت آنها از عدد مشخصی بیشتر باشد آنجا لبه است ، در غیر این صورت به آن کاری نداریم .

$$I_{(i,j)} = n_{(i,j+1)} - n_{(i,j)}$$

$$\text{if}(I_{(i,j)} > T)$$

$$E_{(i,j)} = 1$$

else

$$E_{(i,j)} = 0$$

برای یافتن یک لبه افقی به روش زیر استفاده می کنیم :

از ستون اول به صورت عمودی شروع به حرکت می کنیم و هر پیکسل را با پیکسل قبل از خودش مقایسه می کنیم اگر تفاوت آنها از عدد مشخصی بیشتر باشد آنجا لبه است ، در غیر این صورت به آن کاری نداریم .

$$J_{(i,j)} = n_{(i+1,j)} - n_{(i,j)}$$

$$\text{if}(J_{(i,j)} > T)$$

$$E_{(i,j)} = 1$$

else

$$E_{(i,j)} = 0$$

برای یافتن یک لبه مورب به روش زیر عمل می کنیم :

$$R_{(i,j)} = \sqrt{(I_{(i,j)})^2 + (J_{(i,j)})^2}$$

$$\text{if}(R_{(i,j)} > T)$$

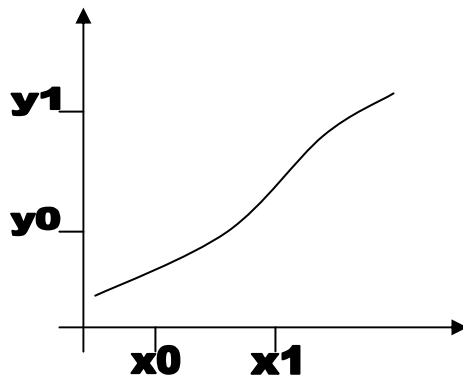
$$E_{(i,j)} = 1$$

else

$$E_{(i,j)} = 0$$

## ۲-۲-۱. رابطه مشتق تصویر و لبه گیری

در واقع روشی که در بالا برای یافتن لبه ها به کار رفته است ، همان مشتق تصویر است . برای اثبات این مدعا به شکل زیر توجه کنید :



$$\frac{y_{j+1} - y_j}{x_{i+1} - x_i}$$

ما در واقع شیب منحنی را محاسبه کرده ایم طول  $x_{j+1} - x_j$  که همواره یک پیکسل است و  $y_{j+1} - y_j$  هم اختلاف مقدار ۲ پیکسل است .

و در فرمول :

$$R_{(i,j)} = \sqrt{(I_{(i,j)})^2 + (J_{(i,j)})^2}$$

در واقع یک مشتق ضمنی از تصویر است .

نتیجه اینکه لبه ، همان مشتق تصویر است .

برای پیاده سازی فرمول لبه گیری به عنوان مثال عمودی در واقع پنجره ای به صورت روبرو داریم :

-۱	+۱
----	----

حال این ماتریس را بر روی ماتریس تصویر می غلتانیم و خروجی برای هر پیکسل در جایگاه خود قرار می دهیم .

برای محاسبه لبه روش های یا در واقع ویندوهای مختلفی طراحی شده است که به چند نمونه از آنها اشاره می کنیم :

-۱	-۲	-۱
۰	×	۰
-۱	-۲	-۱

-۱	۰	۱
-۲	×	۲
-۱	۰	۱

اثبات می شود که ماتریس زیر از تصویر لاپلاس می گیرد :

۰	۱	۰
۱	-۴	۱
۰	۱	۰

## ۲-۳. Zoom :

برای بزرگ نمایی یا کوچک نمایی تصویر الگوریتم های مختلفی وجود دارد که ما به ساده ترین آنها اشاره می کنیم .

برای ۲ برابر کردن تصویر بین هر سطر و ستون یک سطر و ستون اضافه قرار می دهیم و مقدار آن را میانگین ۲ پیکسل کناری آن قرار می دهیم .

برای ۱/۲ برابر کردن تصویر به صورت یکی در میان سطر و ستون ها را حذف می کنیم .  
همین روش را برای ۴ برابر کردن یا ۱/۴ برابر کردن ویا ... استفاده می کنیم .