Features

نسیم فانی

اطلاعات گزارش

تاریخ: 99/1/4

واژگان کلیدی:
گوشه یاب هریس
ویژگی
geometric transformation

چکیده

ابتدا تخمین هندسی برای حالات مختلف را بررسی می‌کنیم.

با الگوریتم گوشه یاب هریس، گوشه های یک تصویر را می‌یابیم.

1-مقدمه

ویژگی، یک مشخصه بصری از تصویر است که میتواند از رنگ، بافت، شکل یا لبه‌های تصویر استخراج شود. در واقع، استخراج ویژگی، فرایند تبدیل مقدارهای خام پیکسل‌های یک تصویر به اطلاعات مفید و معنادار است. در گذشته، طراحی این فرایند توسط متخصصان این حوزه انجام می‌شد و باید ویژگی‌ها را به صورت دستی تعریف و استخراج می‌کردیم ولی امروزه با کمک شبکه‌های عصبی و یادگیری عمیق، این عمل به صورت خودکار در لایه‌های کانولوشنی شبکه‌های عصبی انجام می‌شود. از ویژگی‌ها در وظایفی مثل بخش‌بندی (segmentation) یا تشخیص اشیاء (object recognition) استفاده می‌شود.

ویژگی‌های استخراج شده از تصاویر، در قالب مقادیر عددی نشان داده می‌شوند و معمولا نسبت به تصویر اصلی، ابعاد بسیار کمتری دارند.

به صورت کلی، ویژگی‌های استخراج شده از تصاویر به دو دسته تقسیم می‌شوند: سراسری و محلی

۱. ویژگی های سراسری

این ویژگی‌ها را می‌توانیم از تصویر خام، در گام اول پردازش تصویر استخراج کنیم و معمولا نسبت به مواردی مانند تغییر زاویه نور پایدار نیستند. در بازیابی تصویر از این ویژگی‌ها برای «شناسایی اشیاء» (بررسی حضور یا عدم حضور یک شی خاص در تصویر) استفاده می‌شود. البته صرفا استفاده از این ویژگی‌ها برای دریافت محتوا و معنای یک تصویر کافی نیست.

۲. ویژگی های محلی

بعد از استخراج ویژگی‌های سراسری و شناسایی قسمتی از تصویر که احتمال حضور یک شی در اون وجود دارد، تشکیل کادرهای Region of Interest= RoI، با تقسیم آن بخش‌ها، به بلوک‌هایی کوچکتر، به بررسی جزئی‌تر

ویژگی‌ها در هر یک از بلوک‌ها می‌پردازیم تا ماهیت آن شی تشخیص داده شود. استخراج این ویژگی‌ها، برای توصیف محتوای تصویر ضروری است.

نقش ویژگی‌ها در شناسایی و تشخیص اشیاء تصویر

در پردازش تصاویر، ابتدا ویژگی‌های سطح پایین (low-level features) موجود در تصویر مانند رنگ، بافت و شکل شناسایی می‌شوند. پس از شناسایی بخشی از تصویر، که احتمالا یک شی در آن قرار دارد، تشخیص هویت آن شی انجام می‌شود تا نام آن مشخص شود. ویژگی‌های استخراج شده از تصاویر، می‌توانند سطح پایین یا سطح بالا باشند. ویژگی‌های سطح بالا(high-level features) به کمک ویژگی‌های سطح پایین کشف می‌شوند.

در تشخیص اشیاء، هدف طبقه‌بندی اشیاء موجود در تصویر، در مجموعه‌ای از کلاس‌های از پیش تعیین شده است. هرکدام از کلاس‌ها، مجموعه‌ای از ویژگی‌های مختص به خود دارند که به طبقه‌بندی اشیاء متعلق به این کلاس‌ها کمک می‌کند. برای مشخص کردن اینکه چه چیزهایی در یک تصویر و در کجای آن قرار دارند، از مدل‌های یادگیری ماشین استفاده می‌شود.

2-توضیحات تکنیکال

Estimate Geometry:

در ابتدا باید مفهوم geometric transformation را درک کنیم.

با geometric transformation ، موقعیت پیکسل های یک تصویر را اصلاح می کنیم ، اما رنگ آن‌ها را بدون تغییر نگه می داریم.

اگر (u، v) مختصات تصویر را در تصویر اصلی و (x، y) را در یک تصویر تغییر شکل یافته (یا تاب خورده) نشان دهد، برای ارتباط پیکسل های مربوطه در استفاده از یک جفت تابع استفاده می کنیم:

Forward mapping:

$$\begin{cases} x = x(u,v) \\ y = y(u,v) \end{cases}, \quad or \quad \mathbf{x} = x(\mathbf{u})$$

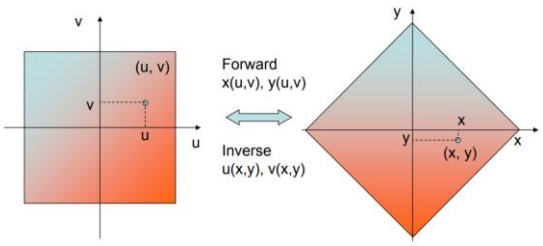
Inverse mapping:

$$\begin{cases} u = u(x,y) \\ v = v(x,y) \end{cases}, \quad or \quad \mathbf{u} = u(\mathbf{x})$$

اگر f (u,v) یا **f (u)** نشانگر تصویر اصلی و g (x,y) یا g **(x)** تصویر تغییر شکل یافته باشد. آن‌ها به این صورت با هم مرتبط می شوند:

$$\begin{cases} g(x,y) = f(u(x,y), v(x,y)) \\ f(u,v) = g(x(u,v), y(u,v)) \end{cases}, \quad or \quad \begin{cases} g(\mathbf{x}) = f(u(\mathbf{x})) \\ f(\mathbf{u}) = g(x(\mathbf{u})) \end{cases}$$

به عنوان مثال:



geometric transformation ترکیبی از translation، scaling و rotation با فرم کلی زیر است:

$$\mathbf{x} = \mathbf{RS}(\mathbf{u} + \mathbf{t}) = \mathbf{Au} + \mathbf{b},$$
$$\mathbf{u} = \mathbf{A}^{-1}(\mathbf{x} - \mathbf{b}) = \mathbf{A}^{-1}\mathbf{x} + \mathbf{c},$$
$$with\ \mathbf{A} = \mathbf{RS}, \mathbf{b} = \mathbf{RSt}, \mathbf{c} = -\mathbf{t}.$$

برای اینکه بتوانیم هندسه یک تصویر را تخمین بزنیم نیاز داریم که مقادیر این تابع را پیدا کنیم.

برای این منظور از نقاط ویژگی استفاده می‌کنیم. ابتدا نقاط ویژگی را در هر دو تصویر پیدا می‌کنیم. برای این منظور الگوریتم‌های زیادی وجود دارد که ما در اینجا از الگوریتم surf استفاده می‌کنیم.

SURF از فیلترهای مربع شکل به عنوان تقریب صاف کننده گاوسی استفاده می کند. (رویکرد SIFT از فیلترهای آبشار برای تشخیص نقاط مشخصه غیرقابل تغییر در مقیاس استفاده می کند ، جایی که اختلاف Gaussians (DoG) بر روی تصاویر تغییر یافته به صورت تدریجی محاسبه می شود.) اگر از تصویر انتگرال استفاده شود ، فیلتر کردن تصویر با یک مربع بسیار سریعتر است:

$$S(x,y) = \sum_{i=0}^{x}\sum_{j=0}^{y} I(i,j)$$

مجموع تصویر اصلی درون یک مستطیل را می توان با استفاده از تصویر انتگرال به سرعت ارزیابی کرد و نیاز به ارزیابی در چهار گوشه مستطیل دارد.

SURF از یک blob detector بر اساس ماتریس هسیان برای یافتن نقاط مورد علاقه استفاده می کند. تعیین کننده ماتریس هسیان به عنوان معیاری برای تغییر محلی در اطراف نقطه استفاده می شود و نقاطی انتخاب می شوند که این تعیین کننده حداکثر باشد.

$$H(p,\sigma) = \begin{pmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{yx}(p,\sigma) & L_{yy}(p,\sigma) \end{pmatrix}$$

سپس این نقاط را باید دیگر match کنیم. با مقایسه توصیفگرهای بدست آمده از تصاویر مختلف ، می توان جفت های منطبق را پیدا کرد. در نهایت برای نقاطی که با هم match شده اند یک تابع تبدیل مطابق آنچه گفته شد می‌یابیم این ماتریس‌ها را در کنار هم قرار داده و ماتریس تبدیل را به دست می‌آوریم.

به کمک ماتریس تبدیل حاصل می‌توانیم تصویر تغییر یافته را به تصویر اولیه مپ کنیم.

شرح مراحل:

1) پیدا نقاط correspond بین دو تصویر $K \geq N$

$$(u_i, v_i) \leftrightarrow (x_i, y_i), i = 1,2\ldots, K.$$

2) پیدا کردن ضرایب a_i ، b_i .... (i=0,..,N-1)

$$\begin{cases} x(u_i, v_i) = a_0 + a_1 u_i + a_2 v_i + \cdots = x_i, \\ y(u_i, v_i) = b_0 + b_1 u_i + b_2 v_i + \cdots = y_i, \end{cases} \quad i = 1,2,\ldots, K$$

برای حل این معادله معادله را به معادله ماتریسی تبدیل می‌کنیم:

$$\mathbf{Aa} = \mathbf{x}, \quad \mathbf{Ab} = \mathbf{y}$$
$$\textit{where}$$
$$\mathbf{A} = \begin{bmatrix} 1 & u_1 & v_1 & \cdots \\ 1 & u_2 & v_2 & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ 1 & u_K & v_K & \cdots \end{bmatrix}, \mathbf{a} = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{N-1} \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{N-1} \end{bmatrix}, \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_K \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_K \end{bmatrix}$$

اگر k=N و ماتریس A یک ماتریس non-singular است:

$$\mathbf{a} = \mathbf{A}^{-1}\mathbf{x}, \quad \mathbf{b} = \mathbf{A}^{-1}\mathbf{y}$$

اگر k>N از زوش least square استفاده می‌کنیم:

$$\mathbf{a} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{x}, \quad b = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T y$$

در غیر اینصورت باید نقاط correspond بیشتری پیدا کنیم!

در متلب از توابع detectSURFFeatures ، extractFeatures و matchFeatures استفاده کرده‌ایم.

Corner Detection:

معمولاً الگوریتم آشکارساز گوشه هریس را می‌توان به پنج مرحله تقسیم کرد:

1- Color to grayscale
2- Spatial derivative calculation
3- Structure tensor setup
4- Harris response calculation
5- Non-maximum suppression

Color to grayscale:

اگـر از آشکارسـاز گوشـه هـریس در یـک تصـویر رنگـی اسـتفاده کنـیم ، اولـین قـدم تبـدیل آن بـه یـک تصـویر در مقیاس خاکسـتری اسـت کـه باعـث افـزایش سـرعت پردازش می شود.

مقـدار پیکسـل مقیـاس خاکسـتری را مـی تـوان بـه عنـوان مجمـوع‌هـای وزن‌دار از مقـادیر R، B و G تصویر رنگ محاسبه کرد :

$$\sum_{C \in \{R,B,G\}} w_C \cdot C,$$

where, e.g.,

$$w_R = 0.299,\ w_B = 0.587,\ w_G = 1 - (w_R + w_B).$$

Spatial derivative calculation:

بعـد، مـی خـواهیم $I_x(x,y)$ و $I_y(x,y)$ را محاسـبه کنیم.

Structure tensor setup:

بـا $I_x(x,y)$ و $I_y(x,y)$ مـی‌تـوانیم structure tensor M را بسازیم.

Harris response calculation:

در ایـن مرحلـه ، بـا اسـتفاده تقریـب زیـر ، کـوچکترین مقدار ویژه تنسور سازه را محاسبه می کنیم:

برای x<<y : $\frac{x \cdot y}{x+y} = x\frac{1}{1+x/y} \approx x$

با trace: $\mathrm{tr}(M) = m_{11} + m_{22}$

یکی دیگر از محاسـبات معمـول پاسـخ هـریس بـه شـرح زیر است:

$$R = \lambda_1 \lambda_2 \quad k \cdot (\lambda_1 + \lambda_2)^2 = \det(M) \quad k \cdot \mathrm{tr}(M)^2$$

Non-maximum suppression:

بـه منظـور انتخـاب مقـادیر بهینـه بـرای نشـان دادن گوشـه هـا ، local maxima را بـه عنـوان گوشـه هـای داخـل پنجـره پیـدا مـی کنـیم کـه یـک فیلتـر 3 در 3 است.

3-شکل‌ها، جدول‌ها و روابط (فرمول‌ها)

تصویر 1 ، attack 1

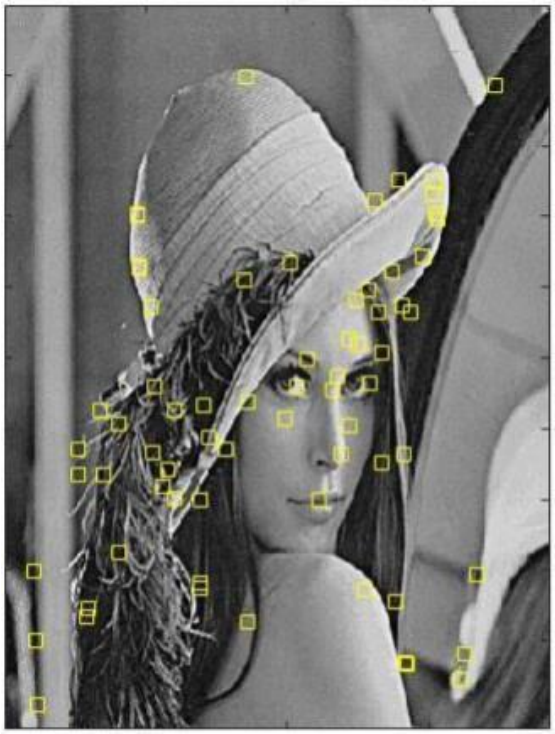
فیچرهای match شده:

تصویر نتیجه:

تصویر نتیجه:

تصویر attack1، 2

فیچرهای match شده:



تصویر نتیجه:

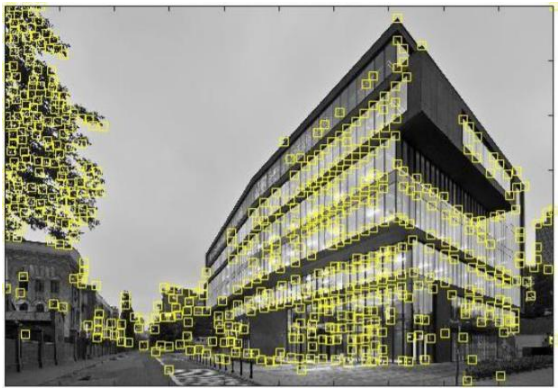
تصویر3، attack1

فیچرهای match شده:

تصویر4، attack1

فیچرهای match شده:

تصویر نتیجه:



4- کدها:

تخمین هندسی:

```
function[output]= geoEst(img,ref,img1)
feature_img = detectSURFFeatures(img,'MetricThreshold',6500);
feature_ref = detectSURFFeatures(ref,'MetricThreshold',6500);
[f1,v1] = extractFeatures(img,feature_img);
[f2,v2] = extractFeatures(ref,feature_ref);
indexPairs = matchFeatures(f1,f2);
mp1 = v1(indexPairs(:,1));
mp2 = v2(indexPairs(:,2));
x1 = mp1.Location(:,1);
y1 = mp1.Location(:,2);
x2 = mp2.Location(:,1);
y2 = mp2.Location(:,2);
z1 = [x1 y1];
z2 = [x2 y2];
T = fitgeotrans(z1,z2,'nonreflectivesimilarity');
output = imwarp(img1,T,'cubic','OutputView',imref2d(size(img1)));
output = output(1:512,1:512);
end
```

گوشه یاب:

```
function [c,r,R] = hc(img,sigma,t)
dx = [-1 0 1;-1 0 1;-1 0 1]/6;
dy = [-1 -1 -1;0 0 0 ; 1 1 1]/6;
g = fspecial('gaussian',max(1,fix(6*sigma)), sigma);
x = filt(img,dx);
y = filt(img,dy);
x2 = conv2(x.^2, g);
y2 = conv2(y.^2, g);
xy = conv2(x.*y, g);
k = 0.04;
R11 = (x2.*y2 - xy.^2)- k*(x2 + y2).^2;
cond = R11>t;
R = R11.*cond;
X = mat2gray(R);
R = (255/(max(max(X))-min(min(X))))*X;
sze = 2*20+1;
mx =ordfilt2(R,sze^2,ones(sze));
R =(R==mx)&(R>t);
[r,c] = find(R);
```

```
function output=filt(img,mask)
    [A,C] = size(img);
    [a,b] = size(mask);
    img = padarray(img,[1,1]);
    output = zeros(A,C,'double');
    for i=1:A
        for j=1:C
            part = img(i:i+a-1,j:j+b-1);
            mult = part.*mask;
            out = sum(mult,'all');
            output(i,j) = out;
        end
    end
end
```

نتایج:

	SSIM	MSE	MP
1	0.9	17.2	14
2	0.6	617.3	68
3	0.6	403	14
4	0.9	63.1	28
mean	0.75	275.15	31
std	0.15	247.8	22.11

گوشه یابی:

مراجع

- https://eeweb.engineering.nyu.edu/~yao/EL5123/lecture12_ImageWarping.pdf
- https://virgool.io/@za.haghgu/%D9%88%DB%8C%DA%98%DA%AF%DB%8C%D9%87%D8%A7%DB%8C-%D8%AA%D8%B5%D9%88%DB%8C%D8%B1-ifwdtdf3s3pq
- https://en.wikipedia.org/wiki/Harris_Corner_Detector