Rotation:

چرخاندن تصویر (**Rotation**) یکی از مهم‌ترین تبدیلات هندسی در پردازش تصویر است که برای تغییر زاویه‌ی دید تصویر به‌کار می‌رود. در این تبدیل، تمام نقاط تصویر حول یک نقطه‌ی مرجع (معمولاً مرکز تصویر) با زاویه‌ی مشخصی دوران داده می‌شوند. برای مثال، اگر بخواهیم تصویری را ۹۰ درجه بچرخانیم، مختصات هر پیکسل جدید بر اساس رابطه‌های مثلثاتی (سینوس و کسینوس زاویه) محاسبه می‌شود. این کار باعث می‌شود که تصویر جهت جدیدی پیدا کند، بدون اینکه اجزای آن تغییر شکل دهند.

در OpenCV برای چرخش تصویر ابتدا باید **ماتریس چرخش** ساخته شود. این کار با تابع `cv2.getRotationMatrix2D(center, angle, scale)` انجام می‌شود. ورودی‌های این تابع عبارت‌اند از: نقطه‌ی مرکز دوران(`center`) ، زاویه دوران به درجه (`angle`) و ضریب بزرگ‌نمایی یا کوچک‌نمایی (`scale`).برای مثال اگر بخواهیم تصویری را ۴۵ درجه حول مرکز خودش بچرخانیم و اندازه‌ی آن ثابت بماند، می‌توانیم `scale=1` بگذاریم. خروجی این تابع یک ماتریس ۲×۳ است که آماده‌ی اعمال روی تصویر است.

پس از ساختن ماتریس چرخش، برای اعمال آن روی تصویر از تابع `cv2.warpAffine` استفاده می‌کنیم. این تابع تصویر اصلی و ماتریس چرخش را می‌گیرد و تصویر چرخیده‌شده را برمی‌گرداند. باید توجه داشت که هنگام چرخش، بخش‌هایی از تصویر ممکن است از قاب خارج شوند یا فضاهای خالی در اطراف تصویر ایجاد شود. این فضاهای خالی به‌طور پیش‌فرض با رنگ سیاه پر می‌شوند، اما می‌توان آن‌ها را با رنگ‌های دلخواه پر کرد. چرخش تصاویر در زمینه‌هایی مثل تشخیص الگو، اصلاح زاویه‌ی اسناد اسکن‌شده و افزایش داده‌ها (Data Augmentation) در یادگیری ماشین کاربرد فراوانی دارد.

برای اینجام این عملیات می توان از کد زیر استفاده نمود:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

rotation_amount_degree = -20
#  convert rotation amount to radian
theta = rotation_amount_degree * np.pi / 180.0


image = cv2.imread('images/input.jpg')
height, width, _ = image.shape

#       | cos(theta) -sin(theta) Tx |
#  T  = | sin(theta) cos(theta)  Ty |
```

```
# T is our rotation matrix
T = np.float32([[np.cos(theta), -np.sin(theta), 0], [np.sin(theta), np.cos(theta),0]])
# We use warpAffine to transform the image using the matrix, T
img_translation = cv2.warpAffine(image, T, (width, height))
plt.imshow(img_translation[...,::-1])
```

در این کد:

```
rotation_amount_degree = -20
#  convert rotation amount to radian
theta = rotation_amount_degree * np.pi / 180.0
```

در اینجا زاویه‌ی چرخش برابر **۲۰ -درجه** انتخاب شده علامت منفی یعنی چرخش **ساعتگرد**.

وقتی در برنامه‌نویسی از توابع مثلثاتی مانند `np.sin()` و `np.cos()` استفاده می‌کنیم، این توابع انتظار دارند زاویه بر حسب **رادیان** داده شود، در حالی‌که ما معمولاً زاویه‌ها را به صورت **درجه** بیان می‌کنیم (مثلاً 30 درجه یا 90 درجه). به همین دلیل، قبل از آنکه زاویه را وارد این توابع کنیم باید آن را از درجه به رادیان تبدیل کنیم. رابطه‌ی بین درجه و رادیان این است که **۳۶۰ درجه برابر با 2π رادیان** است، بنابراین هر درجه معادل $\pi/180$ رادیان خواهد بود.

در این کد، خط `theta = rotation_amount_degree * np.pi / 180.0` دقیقاً همین تبدیل را انجام می‌دهد. به عنوان مثال، اگر مقدار زاویه `20-`درجه باشد، این خط آن را به `0.349- 180` رادیان $-20\times\pi/180\approx-0.349$ تبدیل می‌کند. این مقدار رادیانی سپس در فرمول‌های ماتریس چرخش (برای محاسبه‌ی سینوس و کسینوس زاویه) استفاده می‌شود. به این ترتیب، تصویر می‌تواند با دقت درست و بر اساس محاسبات ریاضی صحیح دوران پیدا کند.

```
T = np.float32([[np.cos(theta), -np.sin(theta), 0], [np.sin(theta), np.cos(theta),0]])
```

ساختار کلی ماتریس چرخش

برای چرخاندن یک نقطه (x,y) در صفحه، از این فرمول استفاده می‌کنیم:

$$\begin{bmatrix} x \\ y \end{bmatrix} \begin{bmatrix} \sin\theta - & \cos\theta \\ \cos\theta & \sin\theta \end{bmatrix} = \begin{bmatrix} 'x \\ 'y \end{bmatrix}$$

- θزاویه‌ی چرخش است (بر حسب رادیان).
- x' و y'مختصات جدید نقطه بعد از چرخش هستند.
- این ماتریس باعث میشود تمام نقاط تصویر با همان زاویه بچرخند.

چرا ۲×۳ است؟

درOpenCV ، برای اینکه بتوان همزمان **جابجایی (Translation)** را هم اعمال کرد، به ماتریس یک ستون اضافه میشود و ماتریس نهایی ۲×۳ خواهد بود:

$$\begin{bmatrix} Tx & \sin\theta - & \cos\theta \\ Ty & \cos\theta & \sin\theta \end{bmatrix} = T$$

- `Tx` و `Ty`مقدار جابجایی در محور x و y هستند.
- در این کد هر دو برابر ۰ هستن ⟵ یعنی تصویر فقط می‌چرخد و جابجا نمی‌شود.

در این کد:

`np.cos(theta)` کسینوس زاویه‌ی دوران (`theta`) را حساب می‌کند.

`np.sin(theta)` سینوس همان زاویه را حساب می‌کند.

این دو مقدار در کنار هم یک **ماتریس چرخش دوبعدی** را تشکیل می‌دهند

این ماتریس تعیین می‌کند که هر نقطه‌ی تصویر بعد از چرخش، به کجا منتقل شود.

در انتهای هر سطر یک ۰ اضافه شده است (ستون سوم). این مقادیر `Tx` و `Ty`هستند، که اگر غیرصفر باشند تصویر علاوه بر چرخش، جابه‌جا هم می‌شود. چون اینجا صفرند، فقط چرخش انجام می‌شود.

`np.float32(...)` تضمین می‌کند که ماتریس از نوع داده‌ی ۳۲بیتی اعشاری باشد، چون توابع OpenCV (مثل `cv2.warpAffine`) فقط این نوع داده را قبول می‌کنند.