آستانه گذاری به روش Otsu:

روش های سراسری آستانه گذاری که در جلسه قبل با آنها آشنا شیدیم معایبی دارند که عبارتند از:

- **حساسیت به تغییرات نور :**روش‌های آستانه‌گذاری سراسری از یک مقدار آستانه ثابت برای کل تصویر استفاده می‌کنند. این موضوع باعث می‌شود در تصاویری که نورپردازی یکنواخت نیست، عملکرد ضعیفی داشته باشند. مناطقی که روشن‌تر یا تاریک‌تر هستند ممکن است به درستی از پس‌زمینه جدا نشوند .
- **نیاز به انتخاب دستی آستانه :**در بسیاری از روش‌های آستانه‌گذاری ساده، کاربر باید مقدار آستانه را به صورت دستی تعیین کند. این کار می‌تواند زمان‌بر و نیازمند تجربه و آزمایش باشد. انتخاب یک آستانه نامناسب می‌تواند منجر به از دست رفتن جزئیات مهم یا جداسازی نادرست اشیاء از پس‌زمینه شود .
- **عدم تطابق با محتوای تصویر :**یک آستانه ثابت ممکن است برای تمام قسمت‌های تصویر مناسب نباشد، زیرا ممکن است توزیع شدت پیکسل‌ها در نواحی مختلف تصویر متفاوت باشد .
- **عملکرد ضعیف در تصاویر با کنتراست پایین :**اگر اختلاف شدت بین اشیاء و پس‌زمینه کم باشد، انتخاب یک آستانه مناسب برای جداسازی آن‌ها دشوار خواهد بود .
- **حساسیت به نویز :**نویز موجود در تصویر می‌تواند بر نتایج آستانه‌گذاری تاثیر منفی بگذارد و منجر به ایجاد پیکسل‌های اشتباه در تصویر باینری شود.

آستانه‌گذاری به روش Otsu در OpenCV

روش آستانه‌گذاری Otsu یک روش خودکار برای تعیین مقدار آستانه بهینه جهت جداسازی پیکسل‌های تصویر به دو دسته (مثلاً پیش‌زمینه و پس‌زمینه) بر اساس شدت روشنایی آن‌ها است. این روش فرض می‌کند که تصویر خاکستری دارای دو ناحیه با توزیع شدت روشنایی نسبتاً مجزا است (هیستوگرام دووجهی).

الگوریتم Otsu به دنبال یافتن آستانه‌ای می‌گردد که **واریانس درون-کلاسی** (within-class variance)را به حداقل برساند یا به طور معادل، **واریانس بین-کلاسی** (between-class variance)را به حداکثر برساند. واریانس درون-کلاسی میانگین وزنی واریانس هر یک از دو دسته پیکسل (روشن و تاریک) است. واریانس بین-کلاسی میزان جدایی میانگین شدت روشنایی دو دسته را اندازه‌گیری می‌کند.

مراحل کلی الگوریتمOtsu

1. **محاسبه هیستوگرام :**ابتدا هیستوگرام تصویر خاکستری محاسبه می‌شود. هیستوگرام تعداد پیکسل‌ها را در هر سطح شدت روشنایی (معمولاً از 0 تا 255) نشان می‌دهد.
2. **نرمال‌سازی هیستوگرام :**هیستوگرام نرمال‌سازی می‌شود تا احتمال وقوع هر سطح شدت روشنایی به دست آید.
3. **تلاش برای یافتن آستانه بهینه :**الگوریتم به صورت تکراری تمام مقادیر آستانه ممکن (از 0 تا 255) را امتحان می‌کند. برای هر آستانه، پیکسل‌ها به دو دسته تقسیم می‌شوند: پیکسل‌هایی با شدت روشنایی کمتر یا مساوی آستانه (فرضاً پس‌زمینه) و پیکسل‌هایی با شدت روشنایی بیشتر از آستانه (فرضاً پیش‌زمینه).
4. **محاسبه واریانس‌ها :**برای هر آستانه، واریانس درون-کلاسی و واریانس بین-کلاسی محاسبه می‌شود.
5. **انتخاب آستانه بهینه :**آستانه‌ای به عنوان آستانه بهینه انتخاب می‌شود که واریانس درون-کلاسی را به حداقل یا واریانس بین-کلاسی را به حداکثر برساند.
6. **آستانه‌گذاری تصویر :**در نهایت، تصویر اصلی با استفاده از آستانه بهینه به یک تصویر باینری (سیاه و سفید) تبدیل می‌شود. پیکسل‌هایی که شدت روشنایی آن‌ها از آستانه بیشتر باشد، به یک مقدار (مثلاً 255 برای سفید) و پیکسل‌هایی که کمتر یا مساوی آستانه باشند، به مقدار دیگری (مثلاً 0 برای سیاه) تبدیل می‌شوند.

هیستوگرام‌ها

هیستوگرام تصویر یک نمودار ستونی است که توزیع شدت روشنایی پیکسل‌های یک تصویر را نشان می‌دهد. محور افقی (x) نشان‌دهنده سطوح شدت روشنایی (معمولاً از 0 برای سیاه تا 255 برای سفید در تصاویر 8 بیتی) و محور عمودی (y) نشان‌دهنده تعداد پیکسل‌هایی است که در آن سطح شدت روشنایی قرار دارند.

اهمیت هیستوگرام در آستانه‌گذاریOtsu

- روش Otsu به طور مستقیم از هیستوگرام تصویر برای تعیین آستانه بهینه استفاده می‌کند.
- شکل هیستوگرام می‌تواند اطلاعات مهمی در مورد تصویر ارائه دهد. برای مثال، یک هیستوگرام دووجهی با دو قله مجزا نشان می‌دهد که احتمالاً تصویر دارای دو ناحیه با شدت روشنایی متفاوت است (مانند یک شیء روشن در یک پس‌زمینه تاریک). در چنین مواردی، روش Otsu معمولاً عملکرد خوبی دارد و آستانه‌ای را در دره بین دو قله پیدا می‌کند.
- اگر هیستوگرام دارای یک قله یا چند قله باشد، ممکن است روش Otsu نتایج مطلوبی به دست ندهد.

نمونه کد با توضیحات:

با استفاده از قطعه کد زیر می توانیم روش آستانه گذاری Otsu را انجام دهیم:

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


img = cv2.imread('images/plates.jpg', cv2.IMREAD_GRAYSCALE)

if img is None:
    print("Error")
else:
    #Otsu
    thresh_value, thresh_img = cv2.threshold(img, 0, 255, cv2.THRESH_BINARY +
cv2.THRESH_OTSU)

    # histogram
    hist = cv2.calcHist([img], [0], None, [256], [0, 256])

    # show images
    plt.figure(figsize=(12, 6))

    plt.subplot(1, 3, 1)
    plt.imshow(img, cmap='gray')
    plt.title('Original')
    plt.axis('off')

    plt.subplot(1, 3, 2)
    plt.imshow(thresh_img, cmap='gray')
    plt.title(f'Otsu: {thresh_value})')
    plt.axis('off')

    plt.subplot(1, 3, 3)
    plt.plot(hist)
    plt.title('Histogram')
    plt.xlabel('Brightness level')
    plt.ylabel('Piaxls')
    plt.axvline(thresh_value, color='r', linestyle='--', label=f'Otsu: {thresh_value}')
    plt.legend()

    plt.tight_layout()
```

```
plt.show()
```

توضیحات کد:

1. **`img = cv2.imread('input.jpg', cv2.IMREAD_GRAYSCALE)`:** تصویر با نام `input.jpg` را در حالت خاکستری می‌خوانیم. شما باید نام تصویر خود را جایگزین کنید.
2. **`cv2.threshold(img, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)`:** این تابع آستانه‌گذاری را اعمال می‌کند .
   - آرگومان اول تصویر خاکستری است.
   - آرگومان دوم مقدار آستانه اولیه است که در روش Otsu نادیده گرفته می‌شود (به همین دلیل 0 قرار داده شده است).
   - آرگومان سوم مقدار ماکزیمم پیکسلی است که به پیکسل‌های بالاتر از آستانه اختصاص داده می‌شود (255 برای سفید).
   - آرگومان چهارم نوع آستانه‌گذاری است `cv2.THRESH_BINARY` .یک نوع آستانه‌گذاری باینری ساده است که با `cv2.THRESH_OTSU` ترکیب شده است تا آستانه به طور خودکار توسط الگوریتم Otsu تعیین شود.
   - این تابع دو مقدار برمی‌گرداند: مقدار آستانه بهینه محاسبه شده (`thresh_value`) و تصویر آستانه‌گذاری شده.(`thresh_img`)
3. **`hist = cv2.calcHist([img], [0], None, [256], [0, 256])`:** این تابع هیستوگرام تصویر خاکستری را محاسبه می‌کند .
   - آرگومان اول تصویر ورودی به صورت یک لیست قرار می‌گیرد.
   - آرگومان دوم کانال‌های مورد نظر برای محاسبه هیستوگرام را مشخص می‌کند (در اینجا [0] برای تصویر خاکستری).
   - آرگومان سوم ماسک است (در اینجا `None` برای کل تصویر.(
   - آرگومان چهارم تعداد سطل‌ها (bins) در هیستوگرام است (256 برای سطوح شدت 0-255).
   - آرگومان پنجم محدوده مقادیر شدت روشنایی است.([0, 256])
4. **نمایش نتایج با `matplotlib.pyplot`:** از `matplotlib` برای نمایش تصویر اصلی، تصویر آستانه‌گذاری شده و هیستوگرام استفاده می‌کنیم. یک خط عمودی قرمز пунк چین در نمودار هیستوگرام، مقدار آستانه Otsu را نشان می‌دهد.

با اجرای این کد، شما می‌توانید تصویر اصلی، نتیجه آستانه‌گذاری به روش Otsu و هیستوگرام تصویر را به همراه مقدار آستانه تعیین شده مشاهده کنید. این به شما درک بهتری از نحوه عملکرد الگوریتم Otsu و ارتباط آن با توزیع شدت روشنایی در تصویر می‌دهد.

استفاده از Gaussian blur در otsu :

استفاده از Gaussian Blur به عنوان یک مرحله **پیش‌پردازش** قبل از اعمال آستانه‌گذاری Otsu می‌تواند در بسیاری از موارد **بسیار مفید** باشد و عملکرد آستانه‌گذاری را بهبود بخشد. دلیل این امر به شرح زیر است:

مزایای استفاده از Gaussian Blur قبل از آستانه‌گذاری Otsu:

1. **کاهش نویز :**تصاویر اغلب دارای نویز هستند که می‌تواند باعث ایجاد تغییرات ناگهانی در شدت روشنایی پیکسل‌ها شود. این نویز می‌تواند هیستوگرام را ناهموار کرده و منجر به تعیین یک آستانه غیربهینه توسط الگوریتم Otsu شود .
   Gaussian Blurبا اعمال یک فیلتر هموارساز، نویز را کاهش داده و هیستوگرام را صاف‌تر می‌کند. این امر می‌تواند به الگوریتم Otsu کمک کند تا دره بین دو قله (در صورت وجود) را بهتر تشخیص دهد و آستانه مناسب‌تری را پیدا کند.
2. **کاهش جزئیات کوچک و ناخواسته** Gaussian Blur **:**می‌تواند جزئیات کوچک و ناخواسته‌ای که ممکن است در فرآیند آستانه‌گذاری اختلال ایجاد کنند را محو کند. این امر می‌تواند به جداسازی بهتر اشیاء اصلی از پس‌زمینه کمک کند.

چه زمانی باید از Gaussian Blur استفاده کرد؟

اگر تصویر شما دارای نویز قابل توجهی است یا شامل جزئیات ریزی است که نمی‌خواهید در تصویر باینری نهایی ظاهر شوند، استفاده از Gaussian Blur قبل از آستانه‌گذاری Otsu توصیه می‌شود.