Hough circle :

روش **Hough Circle Transform**یا همان **تشخیص دایره به کمک هاف** یکی از الگوریتم‌های پردازش تصویر و بینایی ماشین است که برای **شناسایی دایره‌ها در تصاویر** به کار می‌رود. این روش نسخه‌ای توسعه‌یافته از **Hough Transform** است که در ابتدا برای شناسایی خطوط طراحی شده بود.

ایده‌ی اصلی

هر دایره در فضای تصویر توسط سه پارامتر تعریف می‌شود:

- cx مختصات مرکز دایره در راستای افقی
- cy مختصات مرکز دایره در راستای عمودی
- r شعاع دایره

بنابراین، فضای هاف برای دایره‌ها سه‌بعدی است .(xc,yc,r)

مراحل کلی الگوریتم

1. **تشخیص لبه‌ها**

معمولاً با الگوریتمی مثل Canny لبه‌های تصویر استخراج می‌شود. چون دایره‌ها در تصویر روی لبه‌ها نمایان هستند.

2. **انتقال به فضای هاف**

برای هر نقطه لبه در تصویر، مجموعه‌ای از دایره‌هایی که می‌توانند از آن نقطه عبور کنند در نظر گرفته می‌شود. این دایره‌ها با تغییر مقادیر مختلف r و محاسبه‌ی مرکزها (cx,cy) ایجاد می‌شوند.

3. **رأی‌گیری(Voting)**

هر دایره‌ی احتمالی یک رأی به مختصات مربوطه در فضای پارامتر می‌دهد. نقاطی که بیشترین رأی را جمع کنند، به احتمال زیاد دایره‌های واقعی در تصویر هستند.

مزایا

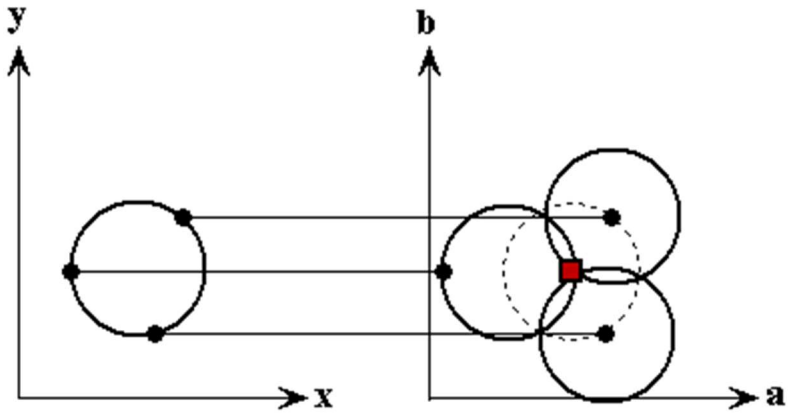
- نسبت به نویز و اشکال اضافی تا حد خوبی مقاوم است.
- توانایی تشخیص چندین دایره با شعاع‌های مختلف را دارد.

معایب

- به دلیل سه‌بعدی بودن فضای پارامتر، از نظر محاسباتی پرهزینه است.
- به کیفیت لبه‌ها و تنظیم دقیق پارامترها (مثل محدوده‌ی شعاع) حساس است.

کاربردها

- تشخیص چشم در تصاویر چهره
- تشخیص سکه‌ها یا اجسام دایره‌ای در صنایع
- بینایی ماشین در رباتیک
- پردازش تصاویر پزشکی (مثل سلول‌ها یا ساختارهای دایره‌ای)



الگوریتم **Hough Circle** کارش این است که **دایره‌ها را داخل یک تصویر پیدا کند.**

به زبان خیلی ساده:

1. اول از تصویر می‌پرسد: «کجاها لبه وجود دارد؟» (مثل جاهایی که رنگ یا شدت تصویر تغییر کرده).
2. بعد برای هر نقطه‌ی لبه فکر می‌کند: اگر این نقطه روی محیط یک دایره بود، مرکز آن دایره کجا می‌تواند باشد؟
3. همه‌ی حدس‌ها را در یک جدول بزرگ (به نام فضای هاف) ذخیره می‌کند.
4. جایی از جدول که بیشترین رأی را گرفته باشد ⟵ همان‌جا مرکز یک دایره است و با شعاع مشخص می‌شود.

یعنی این الگوریتم مثل یک «رأی‌گیری هوشمند» عمل می‌کند: هر نقطه‌ی لبه، به دایره‌های ممکن رأی می‌دهد و در نهایت دایره‌های واقعی با بیشترین رأی‌ها کشف می‌شوند.

خلاصه **Hough Circle** :**الگوریتمی است که با نگاه کردن به لبه‌ها و امتحان کردن همه‌ی دایره‌های ممکن، دایره‌های واقعی موجود در تصویر را پیدا می‌کند.**

اما بر خلاف این در هاف لاین:

ایده‌ی اصلی

الگوریتم **Hough Line** کارش این است که **خطوط مستقیم موجود در تصویر را پیدا کند.**

توضیح ساده

1. اول تصویر بررسی می‌شود تا **لبه‌ها** مشخص شوند مثلاً با.Canny
2. هر نقطه‌ی لبه می‌گوید:
   - اگر من روی یک خط باشم، آن خط می‌تواند چه شکلی باشد؟
3. هر نقطه برای خط‌های ممکن **رأی** می‌دهد.
4. خط‌هایی که بیشترین رأی را بگیرند ← همان خط‌های واقعی در تصویر هستند.

یک نکته مهم

یک خط را نمی‌شود با شیب و عرض از مبدأ $(y = mx + b)$ راحت نمایش داد چون شیب عمودی مشکل ایجاد می‌کند. به همین دلیل در هاف از فرم **قطبی** استفاده می‌کنیم:

$$y \sin\theta + x \cos\theta = \rho$$

- ρ\rhoفاصله‌ی خط از مبدأ است.
- θ\thetaزاویه‌ی خط نسبت به محور x است.

پس هر نقطه لبه، در فضای (ρ,θ)تبدیل به یک منحنی سینوسی می‌شود.
جایی که منحنی‌های زیادی روی هم بیفتند ← همان خط واقعی در تصویر است.

شباهت با هاف دایره

- در دایره: رأی‌گیری در فضای سه‌بعدی (cx,cy) انجام می‌شود.
- در خط: رأی‌گیری در فضای دوبعدی .(ρ,θ)

متد اصلی برای انجام این کار متد زیر است:

```
HoughCircles(image, method, dp, minDist, param1=?, param2=?, minRadius=?, maxRadius=?)
```

۱ کلیت تابع و ورودی تصویر

1. هدف: پیدا کردن دایره‌ها در یک تصویر تک‌کاناله (گرِیس‌اسکیل).
2. نوع تصویر: ورودی معمولاً باید uint8(۸ بیتی) و تک‌کاناله باشد — اگر رنگی است ابتدا cv2.cvtColor(..., cv2.COLOR_BGR2GRAY)کنید.
3. پیش‌پردازش معمول: قبل از Hough معمولاً نویز را با GaussianBlurیا medianBlurکاهش می‌دهند (مثلاً GaussianBlur(img, (9,9),2))، چون لبه‌های نامنظم باعث نتایج بد می‌شوند.
4. خروجی: آرایه‌ای از دایره‌ها (هر دایره .([x_center, y_center, radius]) = اگر دایره‌ای پیدا نشود خروجی None است. مقادیر معمولاً float هستند و برای رسم معمول آنها را گرد می‌کنند (np.uint16(np.around(circles))).

method۲

1. معمول‌ترین مقدار .cv2.HOUGH_GRADIENT: این روش مبتنی‌بر گرادیان لبه است و رایج‌ترین پیاده‌سازی در OpenCV.
2. نکته: بعضی نسخه‌ها روش‌های دیگری دارند که پارامترها را متفاوت تفسیر می‌کنند؛ اگر از روش غیرمعمول استفاده می‌کنید مستندات مربوطه را چک کنید.

۳ — dp (نسبت عکس(inverse ratio

1. تعریف: نسبت رزولوشن تصویر به رزولوشن آکومولاتور.(accumulator)
   - اگر dp = 1 → آکومولاتور هم‌رزولوشن تصویر.

- اگر `dp = 2` → آکومولاتور نصف عرض/ارتفاع تصویر (کم‌تر حافظه، دقت کمتر).

2. اثر: مقدار بزرگ‌تر → سرعت و حافظه کمتر، ولی دقت مرکز و شعاع کمتر. مقدار کوچک (≈1) دقت بهتر ولی هزینه محاسباتی بیشتر.
3. پیشنهاد عملی: از `1.0` یا `1.2` شروع کنید؛ اگر حافظه یا سرعت مشکل داشت `dp` را بالا ببرید.

۴ — `minDist` حداقل فاصله بین مراکز دایره‌ها

1. معنی: کمترین فاصله پیکسلی مجاز بین دو مرکز دایره شناسایی‌شده.
2. نقش: جلوگیری از تشخیص‌های تکراری برای یک دایره یا حذف دایره‌های بسیار نزدیک.
3. قاعده سرانگشتی: آن را بر اساس اندازه شعاع تنظیم کنید؛ مثلاً `minDist ≈ expected_radius` یا `expected_radius/2`. اگر دایره‌های واقعی خیلی نزدیکند مقدار را کوچک‌تر کنید (اما ریسک تکراری شدن بالا می‌رود).

۵ — `param1` (آستانه‌ی اول Canny)

1. برای `HOUGH_GRADIENT` این پارامتر به عنوان **آستانه‌ی بالاتر تابع Canny** برای پیدا کردن لبه‌ها استفاده می‌شود.
2. معمولاً آستانه پایین‌تر Canny برابر `param1/2` در نظر گرفته می‌شود (قابل تنظیم در پیاده‌سازی‌ها).
3. اثر: مقدار بزرگ‌تر → فقط لبه‌های قوی‌تر گرفته می‌شوند (کمتر نویز ولی ممکن است لبه‌های ضعیف حذف شوند).
   مقدار کوچک‌تر → لبه‌های بیشتری استخراج می‌شوند (ممکن است رأی‌های کاذب زیاد شود).
4. مقدار شروع رایج `100` :یا `200` بسته به کنتراست تصویر.

۶ — `param2` (آستانه‌ی آکومولاتور (حساسیت تشخیص مرکز)

1. معنی: آستانه‌ای روی مقدار رأی/آکومولاتور که مشخص می‌کند یک نقطهٔ مرکز باید چقدر «قوی» باشد تا به عنوان دایره قبول شود.
2. اثر:
   - `param2` کوچک → حساسیت بیشتر → دایره‌های بیشتری (شامل کاذب).
   - `param2` بزرگ → حساسیت کمتر → فقط دایره‌های قوی‌تر پذیرفته می‌شوند.
3. مقدار معمول شروع: حدود `50–20` (بسته به تصویر). اگر هیچ دایره‌ای نمی‌گیرید `param2` را کم کنید؛ اگر دایره‌های کاذب زیاد است `param2` را زیاد کنید.

۷ `minRadius` و `maxRadius`

1. معنی: بازه‌ی شعاع‌هایی که جستجو می‌شوند (بر حسب پیکسل).
2. اهمیت: محدود کردن این بازه هم سرعت را بالا می‌برد هم خطا را کم می‌کند.
3. اگر هر دو برابر صفر باشند، تابع همه‌ی شعاع‌ها را بررسی می‌کند — کند و پرخطا. همیشه اگر می‌دانید شعاع‌ها در چه بازه‌ای‌اند، مشخصشان کنید.

۸ فرمت خروجی و نکات هنگام رسم

خروجی معمولاً شکل `(1, N, 3)` یا `(N, 3)` دارد؛ ترتیب هر ورودی `[x_center, y_center, radius]`