الجلسة الرابعة

Feature Extraction

Corner, line and circle detection (Harris and Hough transform)

كشف الزوايا والخطوط (كاشف هاريس و تحويل هاف)

1.1 كاشف Harris

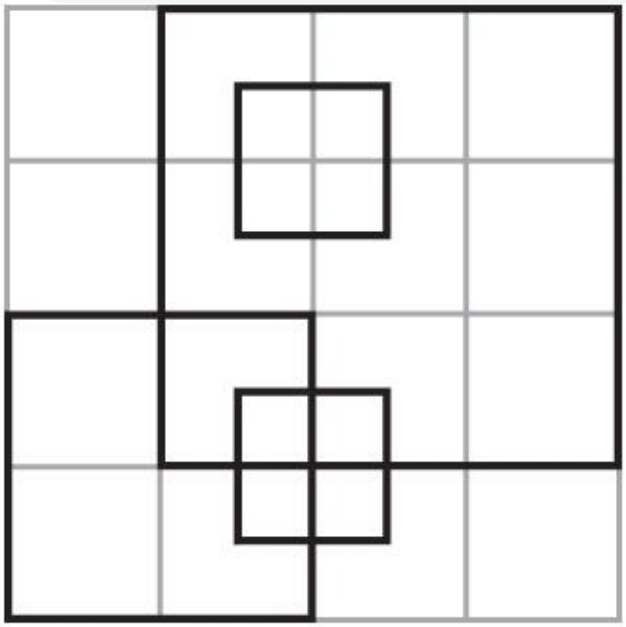
- يستخدم لكشف الزوايا ويعطي استجابة عالية عند وجود قيمة عالية للتدرج في اتجاهين مختلفين عند نفس البكسل.

نفذ الكود:

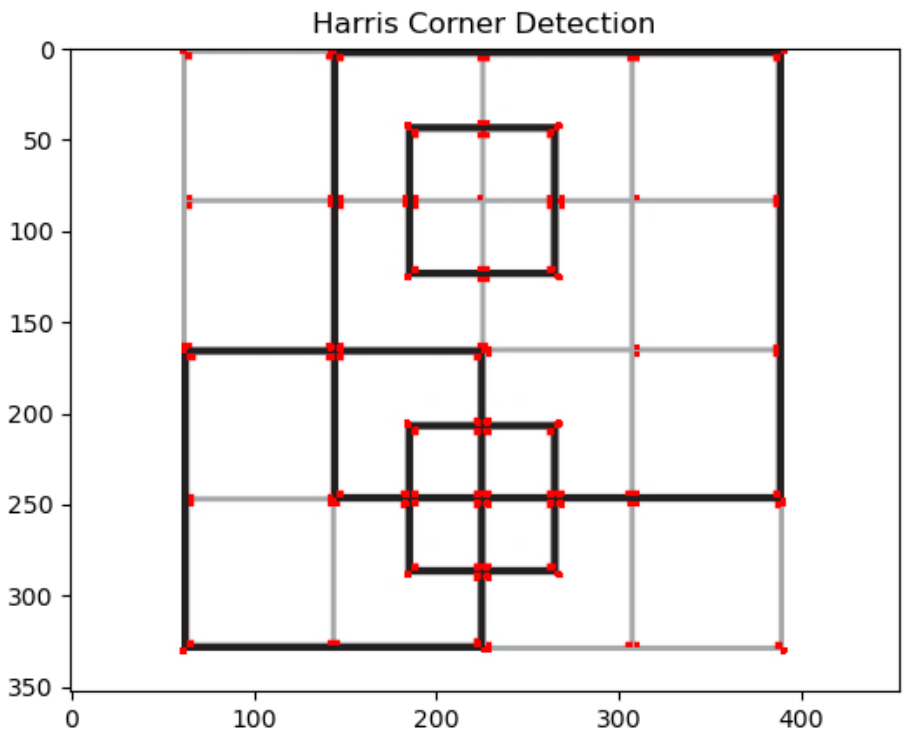
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
# Read the image
image=cv2.imread("box.jpg",1)
# Convert to grayscale
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
# Apply Harris Corner Detection
dst = cv2.cornerHarris(gray, 2, 3, 0.04)
# Dilate the result for better visualization
dst = cv2.dilate(dst, None)
# Threshold for corner points
threshold = 0.01 * dst.max()
image[dst > threshold] = [0, 0, 255]  # Mark corners in red
# Display the result
plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
plt.title("Harris Corner Detection")
plt.show()
```

تطبيق كاشف Harris باستخدام تابع جاهز من مكتبة OpenCV نمرر للتابع النسخة الرمادية من الصورة، و عدد البكسلات المجاورة للبكسل التي تدرس من قبل الكاشف، وحجم قناع سوبل وهو هنا 3*3، ومعامل k معامل حساسية هاريس لكشف الزوايا (عتبة قبول الزاوية أو رفضها)

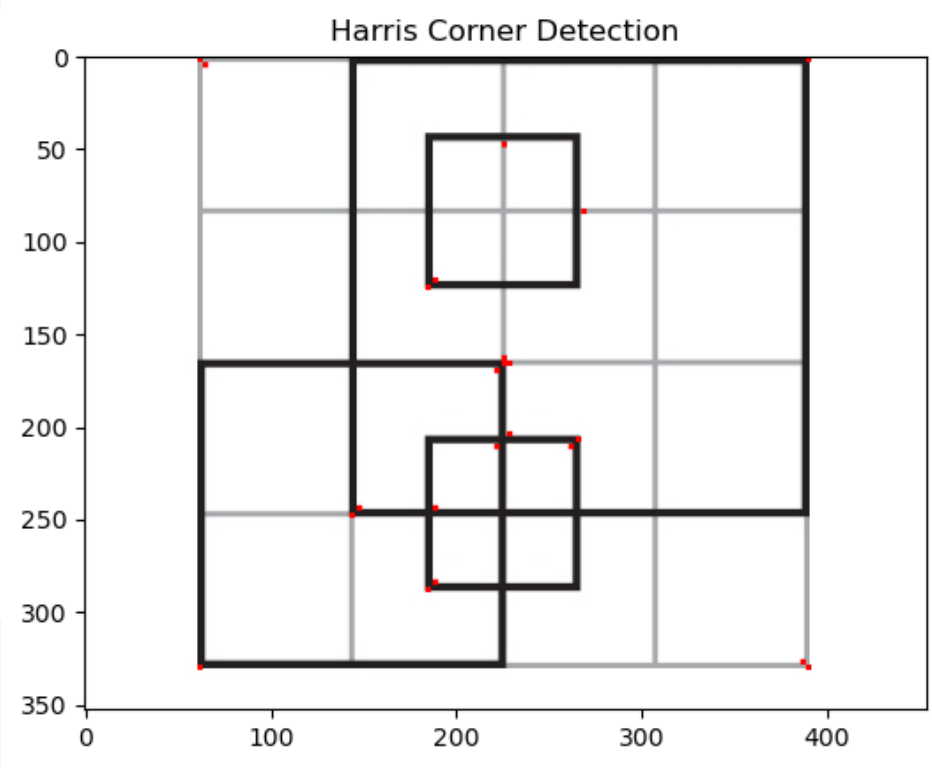
الصورة الأصلية



والنتيجة عند عتبة 0.04:

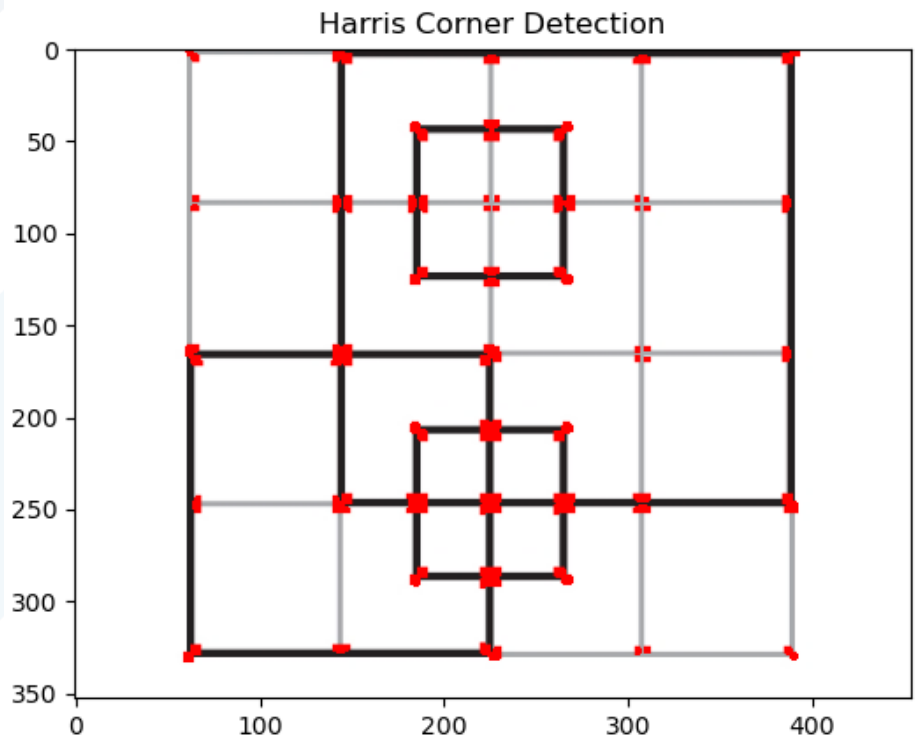


والنتيجة عن عتبة أكبر 0.1



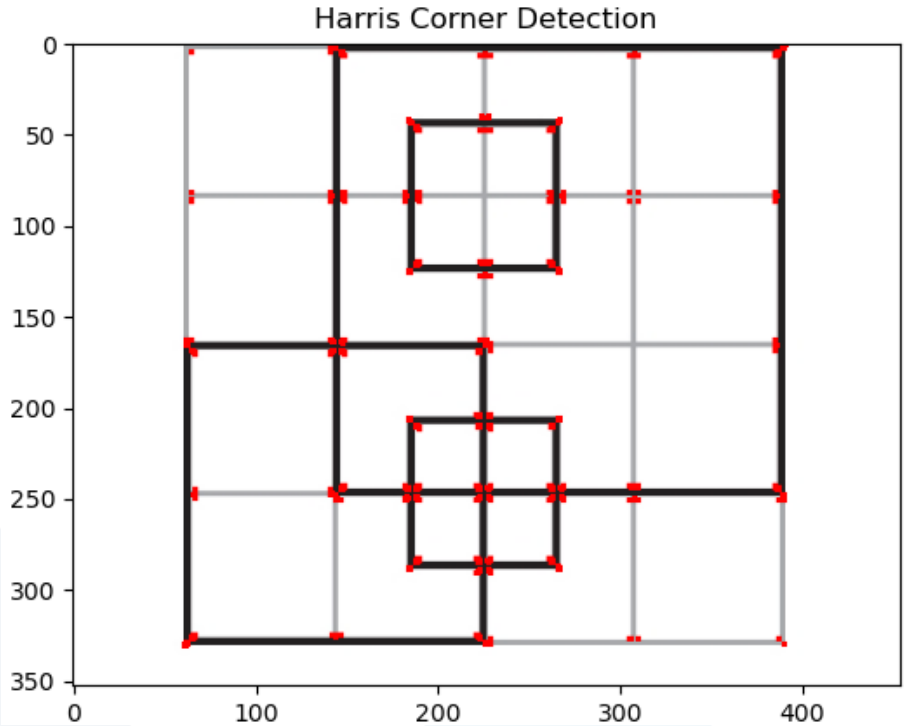
نلاحظ أن الزوايا الأقوى تم كشفها أما الزوايا الأضعف أهملت.

عند تغيير عدد البكسلات المجاورة التي تؤخذ بالحسبان عند حساب مشتقات سوبل من 2 إلى 4:



نلاحظ أنَّ الاستجابة كانت أقوى نظراً لزيادة عدد البكسلات المجاورة للبكسل التي يتم ضمنها الاشتقاق.

أما تثبيت العدد على 2 والعتبة على 0.04 مع زيادة أبعاد مرشح سوبل نحصل على الاستجابة التالية:



نلاحظ كيف أن استجابة الزاوية أصبحت أكثر دقة (مثلا نقطة تلاق منصفات المربع فيها 4 زوايا واضحة بدلاً من زاوية واحدة).

1.2 كشف الخطوط باستخدام تحويل هاف Hough Transform

يمكن استخدام تحويل Hough لكشف الخطوط وفقاً لزوايا محددة.

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
# Read the image
image=cv2.imread("box.jpg",1)
# Convert to grayscale
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
edges = cv2.Canny(gray, 150, 250, apertureSize=3)
```

بداية يتم كشف حواف الصورة باستخدام مرشح كاني وبعتبتين
Low=150 و high=250

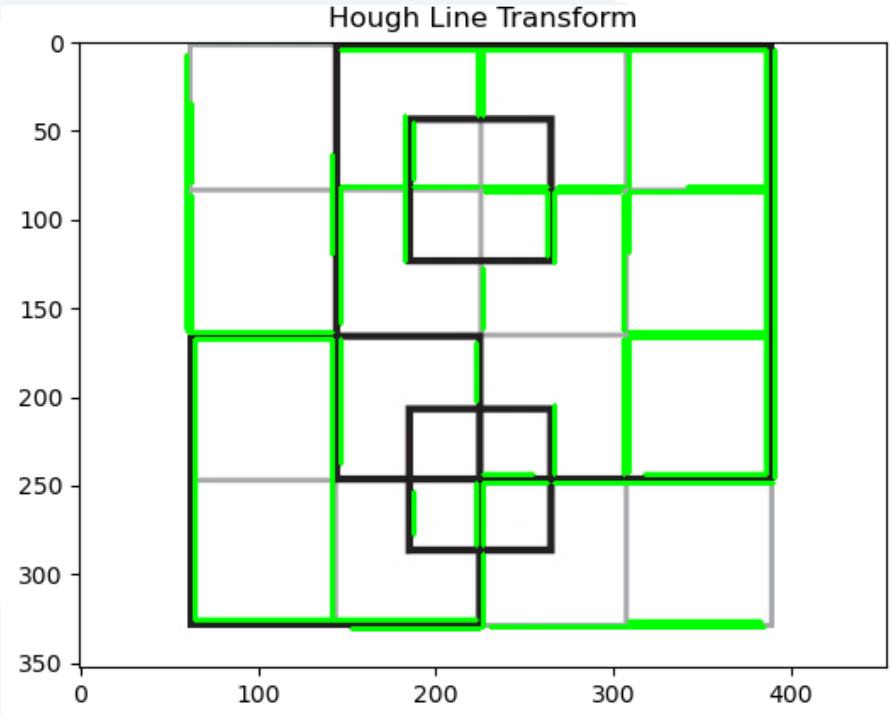
```
minLineLength = 1000

maxLineGap = 20


lines = cv2.HoughLinesP(edges, 1, np.pi / 180, 100, minLineLength, maxLineGap)
```

تطبيق تابع تحويل هاف لكشف الخطوط نمرر له صورة الحواف، وقيمة distance resolution (rho) ودقة الزاوية theta أو مجال الانتقال من زاوية لأخرى للبحث عن الخطوط وهو هنا 180/180 أي 1 ثم يتم تحديد العتبة وهي العدد الأصغري المقبول للأصوات للخط ليتم قبوله
ثم يتم تحديد minLineLength وهو أصغر طول مقبول للخط المراد كشفه وmaxLineGap وهي عدد الثغرات الأعظمي المسموح وجودها بين أجزاء الخط.

```
# Draw the detected lines

for line in lines:

   x1, y1, x2, y2 = line[0]  # Unpack the endpoints

   cv2.line(image, (x1, y1), (x2, y2), (0, 255, 0),  2)  # Draw the line

# Display the result

plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))

plt.title("Hough Line Transform")

plt.show()
```

رسم الخطوط الناتجة عن عملية الكشف بلون أخضر مباشرة فوق الصورة الأصلية ثم عرض النتيجة

النتيجة عند اختيار عدد أصوات Voting numbers مساوي لـ 100



النتيجة عند اختيار عدد أصوات Voting numbers مساوي لـ 300