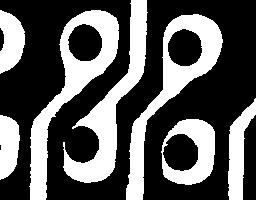
Diberikan sebuah gambar :



Dari gambar tersebut, kita akan mendeteksi ada berapa banyak lubang berbentuk lingkarang yang ada pada gambar tersebut. Untuk itu, akan digunakan fungsi Hough Circle dari OpenCV karena fungsi tersebut paling baik dalam mendeteksi lingkaran.

|  |
| --- |
| import cv2 as cv |

Untuk deteksi lingkaran, yang akan kita gunakan hanyalah package OpenCV saja. Pertama-tama, seperti biasa kita akan membaca gambar terlebih dahulu.

|  |
| --- |
| image = cv.imread("image-soalno-1.jpeg") |

Ubah color space gambar menjadi grayscale agar channel yang digunakan hanyalah satu.

|  |
| --- |
| gray = cv.cvtColor(image, cv.COLOR\_BGR2GRAY) |

Setelah itu threshold nilai grayscale nya. Sebenarnya kita tidak diwajibkan untuk men-threshold nilai, hanya saja dengan melakukan threshold maka nilai yang muncul akan menjadi binary. Hal tersebut membuat pixel putih dan hitam memiliki perbedaan yang signifikan terutama pada sisi-sisi (boundaries). Kita akan melakukan threshold pada nilai tengah dari range 0-255 yaitu 127. Nilai tersebut dipilih agar pembagian terjadi merata.

|  |
| --- |
| \_, thresh = cv.threshold(gray, 255//2, 255, cv.THRESH\_BINARY) |

Setelah threshold, aplikasikan fungsi Hough Circle dengan beberapa parameter yang diperlukan yaitu :

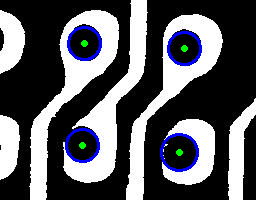
* ***gray***: Input image (grayscale).
* ***HOUGH\_GRADIENT***: metode yang digunakan untuk mendeteksi circle yaitu dengan melihat gradien yang didapat dari canny edge detection.
* ***dp*** *= 1*: The inverse ratio of resolution. Gunakan nilai satu apabila ingin rasio gambar tidak berubah.
* ***minDist*** *=* Jarak minimum antar lingkaran
* ***param1*** *= 100*: Upper threshold untuk Canny edge detector. Setelah melakukan beberapa kali percobaan dengan nilai param1 yang berbeda pada gambar yang diberikan, ternyata tidak berpengaruh. Nilai yang diuji yaitu 1-100 memberikan hasil yang sama.
* ***param2*** = 20: Threshold for center detection. Semakin kecil nilai center, semakin banyak lingkaran yang ditemukan. Nilai yang paling sesuai yang ditemukan adalah 20
* ***minRadius*** *= 0*: Minimum radius to be detected. Radius terkecil dari lingkaran, kita tidak menggunakan parameter ini sebenarnya sehingga bisa gunakan parameter 0 sebagai tanda jika tidak ada batasan nilai. Dapat diasumsikan pula lingkaran yang muncul pada gambar bisa jadi berukuran kecil jadi tidak perlu didefinisikan minimum radius.
* ***maxRadius*** *= 40*: Maximum radius to be detected. Radius maksimum dari lingkaran yang boleh dideteksi. Kita menggunakan nilai ini karena kita tidak mau menyeleksi lingkaran luar. Kita hanya ingin menyeleksi lingkaran dalam saja sehingga nilai nya harus dibatasi. Dan setelah melakukan percobaan pada beberapa nilai dan melihat secara manual rata-rata radius lingkaran (30 < r < 40) maka ditentukan batas maksimum nya 40.

|  |
| --- |
| circles = cv.HoughCircles(thresh, cv.HOUGH\_GRADIENT, 1, image.shape[0] // 8, param1=100, param2=20, minRadius=0, maxRadius=40) |

Karena algoritma HoughCircles menggunakan Canny Edge Detection di dalam nya, maka kernel yang digunakan berasal dari Gaussian filter yang secara otomatis digunakan. Dari fungsi tersebut akan diperoleh array lingkaran yang terdeteksi. Gambarkan lingkaran-lingkaran yang terdeteksi dengan mewarnai keliling nya dengan warna biru dan pusatnya berwarna hijau.

|  |
| --- |
| if circles is not None:  for (x, y, r) in circles[0, :]:  cv.circle(image, (x, y), r, (255, 0, 0), 2)  cv.circle(image, (x, y), 2, (0, 255, 0), 2) |

Berikut adalah hasil yang diperoleh :



Hasil yang diperoleh cukup memuaskan karena Hough Circle dapat mendeteksi 4 lingkaran dengan ukuran yang sangat mendekati. Kita juga memperoleh titik center dan panjang jari-jarinya.

1. (x = 84.5, y = 43.5) -> r = 16.799999237060547
2. (x = 184.5, y = 48.5) -> r = 16.799999237060547
3. (x = 179.5, y = 152.5) -> r = 18.100000381469727
4. (x = 82.5, y = 145.5) -> r = 16.0