**TUGAS (PENGOLAHAN CITRA)**

Nama : **Narinda Genta Rosasia**

Nim : **320200401019**

Prodi : **T. Informatika**

Pengolahan analisis diambil berdasarkan referensi dari :

* Hidayat, Jhony dkk. 2002. IMPLEMENTASI METODE OTSU THRESHOLDING PADA BINERISASI CITRA WAJAH. Diakses pada 25 Februari 2023, pada

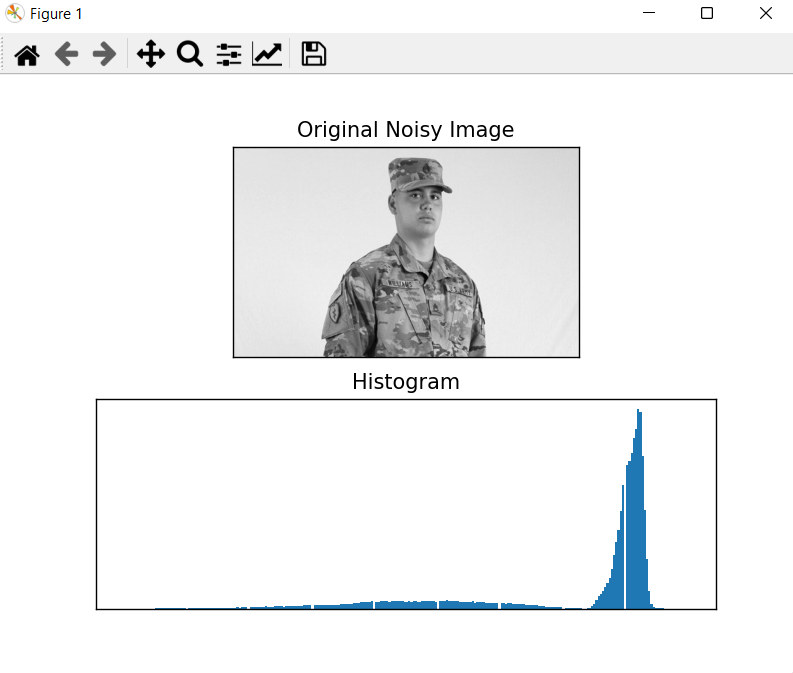
<https://journal.utnd.ac.id/index.php/jot/article/view/246/165>

1. Pengolahan citra menggunakan 3 teknik segmentasi untuk Gambar pertama

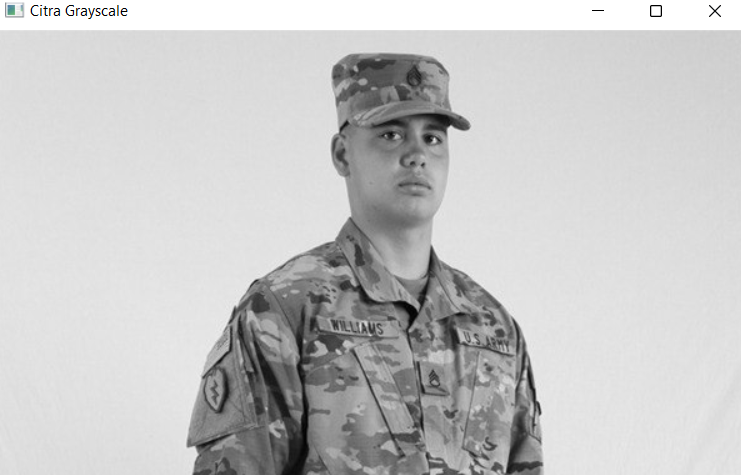
* Gambar 1



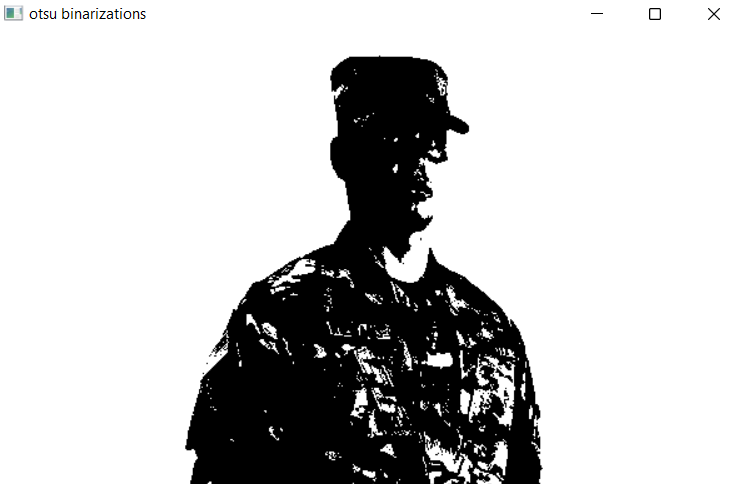
* Historigram



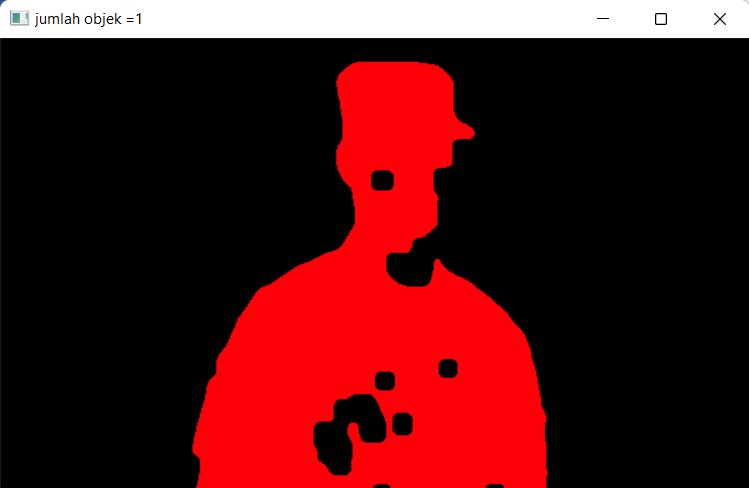
* Citra Grayscale



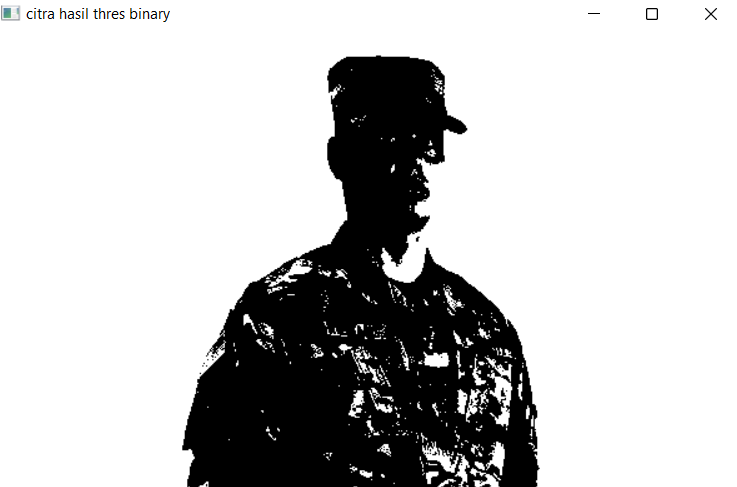
* Citra Otzu Binarization



* Teknik Pencitraan CCL

Code

* import cv2
* import numpy as np
* image = cv2.imread('task1.jpg')
* gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)
* gray\_correct = np.array(200 \* (gray/255) \*\* 1.2 , dtype='uint8')
* gray\_equ = cv2.equalizeHist(gray)
* thresh = cv2.adaptiveThreshold(gray\_correct, 255, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C, cv2.THRESH\_BINARY\_INV, 255 , 10)
* kernel = np.ones((15,15), np.uint8)
* img\_dilation = cv2.dilate(thresh, kernel, iterations=1)
* img\_erode = cv2.erode(img\_dilation, kernel, iterations=1)
* img\_erode = cv2.medianBlur(img\_erode, 7)
* ret, labels = cv2.connectedComponents(img\_erode)
* label\_hue = np.uint8(179 \* labels/np.max(labels))
* blank\_ch = 255 \* np.ones\_like(label\_hue)
* labeled\_img = cv2.merge ([label\_hue, blank\_ch, blank\_ch])
* labeled\_img =cv2.cvtColor (labeled\_img, cv2.COLOR\_HSV2BGR)
* labeled\_img[label\_hue ==0] = 0
* cv2.imshow("citra rgb", image)
* cv2.imshow("citra grayscale", thresh)
* cv2.imshow("dilation + erosin", img\_erode)
* cv2.imshow("jumlah objek =" + str(ret-1), labeled\_img)
* cv2.waitKey(0)
* cv2.destroyAllWindows()
* Teknik Thres Binary



Code

ret, thresholding = cv2.threshold (gray,170,255,cv2.THRESH\_BINARY)

adaptif\_th\_mean = cv2.adaptiveThreshold (blur,255,cv2.ADAPTIVE\_THRESH\_MEAN\_C,cv2.THRESH\_BINARY,11,2)

adaptif\_th\_gaussian  = cv2.adaptiveThreshold (blur,255,cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C,cv2.THRESH\_BINARY,11,2)

**Hasil Analisis Gambar 1**

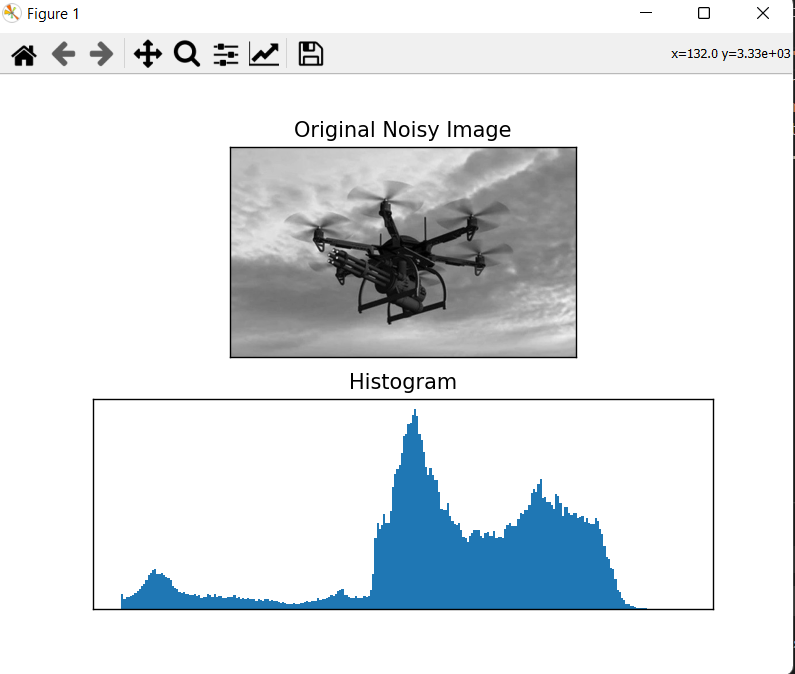
Berdasarkan hasil uji coba citra yang digunakan, metode CCL digunakan agar menjalankan pendeteksian jumlah benda dengan jelas. Pencitraan dengan metode Otsu binarization mampu membentuk pengenalan objek citra, begitu juga untuk metode Thres binary.

1. Pengolahan citra menggunakan 3 teknik segmentasi untuk Gambar Kedua

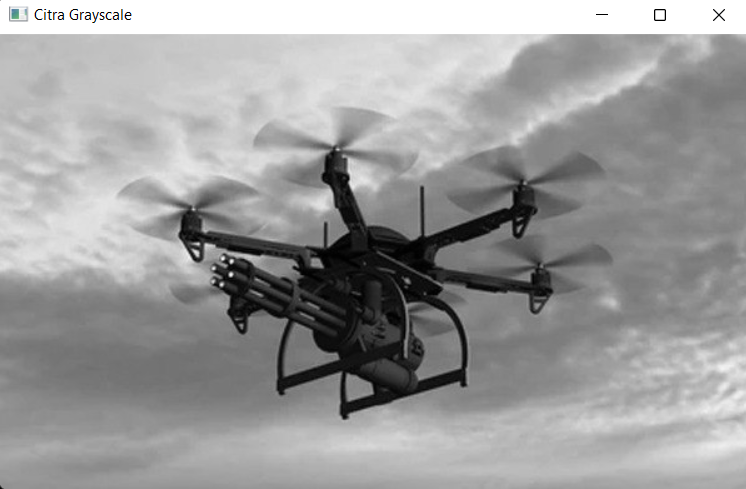
* Gambar 1



* Historigram



* Citra Greyscale



* Teknik Otsu Binarization



* Teknik CCL



Code

…

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

gray\_correct = np.array(52 \* (gray/255) \*\* 1.2 , dtype='uint8')

gray\_equ = cv2.equalizeHist(gray)

thresh = cv2.adaptiveThreshold(gray\_correct, 255, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C, cv2.THRESH\_BINARY\_INV, 255 , 10)

kernel = np.ones((15,15), np.uint8)

img\_dilation = cv2.dilate(thresh, kernel, iterations=1)

img\_erode = cv2.erode(img\_dilation, kernel, iterations=1)

img\_erode = cv2.medianBlur(img\_erode, 7)

ret, labels = cv2.connectedComponents(img\_erode)

label\_hue = np.uint8(179 \* labels/np.max(labels))

blank\_ch = 255 \* np.ones\_like(label\_hue)

labeled\_img = cv2.merge ([label\_hue, blank\_ch, blank\_ch])

labeled\_img =cv2.cvtColor (labeled\_img, cv2.COLOR\_HSV2BGR)

labeled\_img[label\_hue ==0] = 0

…

* Teknik Thres Binary



Code

...

ret, thresholding = cv2.threshold (gray,110,255,cv2.THRESH\_BINARY)

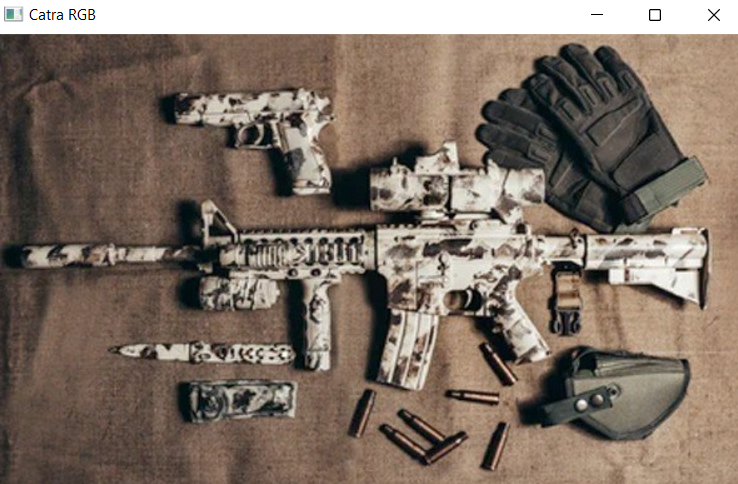
adaptif\_th\_mean = cv2.adaptiveThreshold (blur,255,cv2.ADAPTIVE\_THRESH\_MEAN\_C,cv2.THRESH\_BINARY,11,2)

adaptif\_th\_gaussian  = cv2.adaptiveThreshold (blur,255,cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C,cv2.THRESH\_BINARY,11,2)

...

1. Pengolahan citra menggunakan 3 teknik segmentasi untuk Gambar Kedua

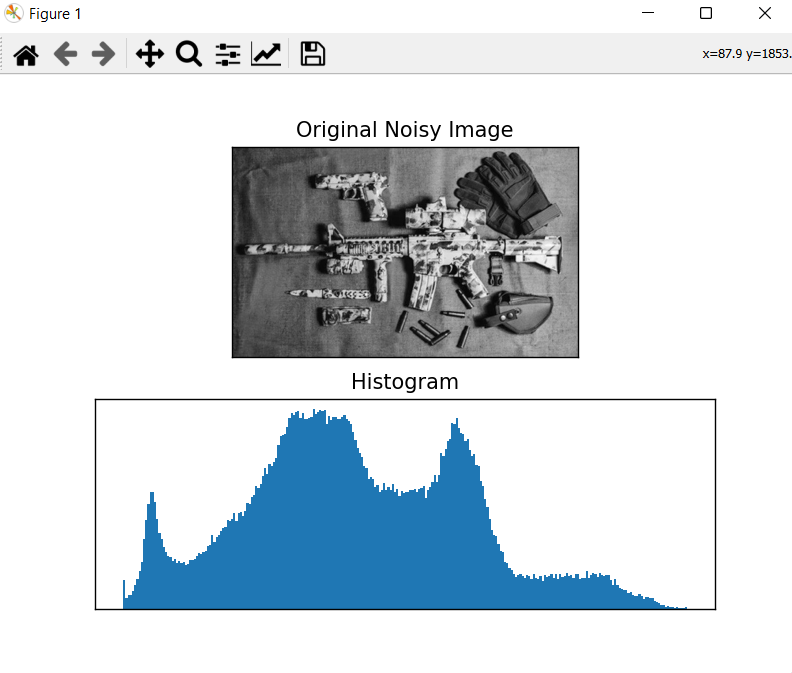
* Gambar Asli



* Citra Greyscale



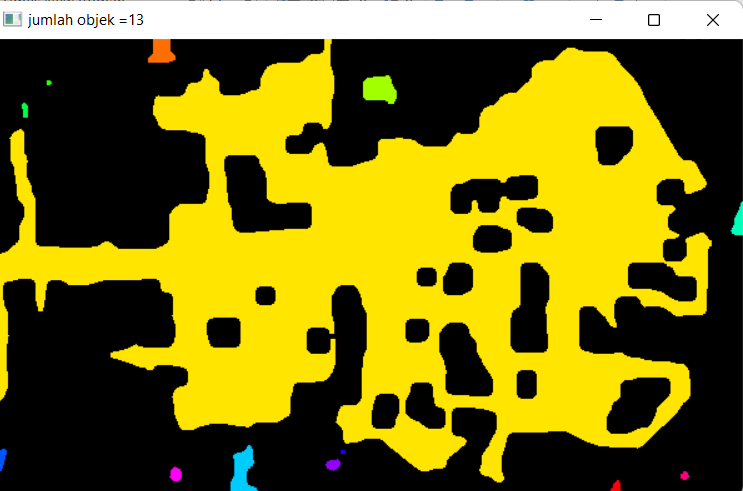
* Historigram



* Citra Otsu Binarization



* Citra Hasil CCL



Code

...

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

gray\_correct = np.array(100 \* (gray/255) \*\* 1.2 , dtype='uint8')

gray\_equ = cv2.equalizeHist(gray)

thresh = cv2.adaptiveThreshold(gray\_correct, 255, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C, cv2.THRESH\_BINARY\_INV, 255 , 10)

kernel = np.ones((15,15), np.uint8)

img\_dilation = cv2.dilate(thresh, kernel, iterations=1)

img\_erode = cv2.erode(img\_dilation, kernel, iterations=1)

img\_erode = cv2.medianBlur(img\_erode, 7)

ret, labels = cv2.connectedComponents(img\_erode)

label\_hue = np.uint8(179 \* labels/np.max(labels))

blank\_ch = 255 \* np.ones\_like(label\_hue)

labeled\_img = cv2.merge ([label\_hue, blank\_ch, blank\_ch])

labeled\_img =cv2.cvtColor (labeled\_img, cv2.COLOR\_HSV2BGR)

labeled\_img[label\_hue ==0] = 0

...

* Citra Hasil Thres Binary



**Hasil Analisis Gambar 3**

Berdasarkan hasil uji coba citra yang digunakan, metode CCL, Otsu Binarization, dan Thres Binary tidak dapat menentukan jumlah benda dengan jelas.