Color Space

RGB to HSV

Nama : Muhamad Hilmy Haidar
NIM : G64170030
Tanggal : 27 Februari 2020
Nama Asisten : 1. Hilmi Farhan Ramadhani G64160048 2. Kautsar Ibrahim Hilmi G64160073

1. Buatlah fungsi konversi gambar RGB ke HSV !

```
# RGB to HSV
def rgb_to_hsv(image):
   row, col, ch = image.shape
   canvas = np.zeros((row, col, ch), np.uint8)
   for i in range(row):
       for j in range(col):
           b, g, r = image[i, j]
           b, g, r = b / 255, g / 255, r / 255
           v = max(r, g, b)

           h = 0
           if v == min(r, g, b):
               h = 0
           elif v == r:
               h = (60 * (g - b) / (v - min(r, g, b)) + 360) % 360
           elif v == g:
               h = (60 * (b - r) / (v - min(r, g, b)) + 120) % 360
           elif v == b:
               h = (60 * (r - g) / (v - min(r, g, b)) + 240) % 360

           s = 0 if v == 0 else 255 * (v - min(r, g, b)) / v
           v *= 255
           canvas[i, j] = h, s, v

   return canvas
```

2. Lakukan Face detection pada citra ‘FACE DETECTION.png’ !
   2.1. Konversi Gambar RGB ke HSV

Pada Gambar ‘FACE DETECTION.png’ di atas, wajah-wajah yang terdapat di gambar tersebut memiliki warna coklat muda sampai tua dengan kisaran range warna antara #F8E7E1 sampai #6B3A2E. Sangat sulit untuk kita melakukan seleksi warna dengan RGB. Diperlukan adanya perubahan color space dari RGB menjadi HSV untuk membuat perbedaan mencolok antara wajah dan objek lain nya. Kalkulasi yang digunakan untuk mengubah warna RGB ke HSV adalah sebagai berikut :

$$R' = R/255$$
$$G' = G/255$$
$$B' = B/255$$

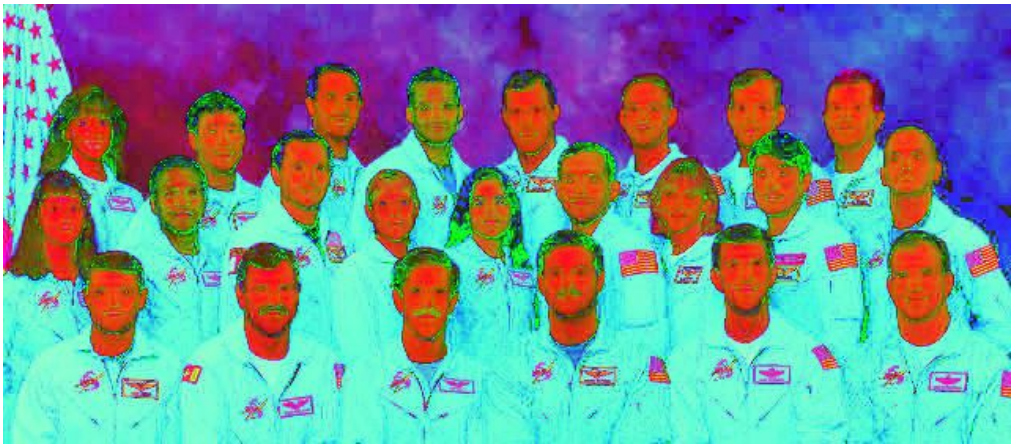
$$Cmax = max(R', G', B')$$
$$Cmin = min(R', G', B')$$

$$\Delta = Cmax - Cmin$$

$$H = \begin{cases} 0^\circ & \Delta = 0 \\ 60^\circ \times \left(\frac{G'-B'}{\Delta} mod 6\right) & , C_{max} = R' \\ 60^\circ \times \left(\frac{B'-R'}{\Delta} + 2\right) & , C_{max} = G' \\ 60^\circ \times \left(\frac{R'-G'}{\Delta} + 4\right) & , C_{max} = B' \end{cases}$$

$$S = \begin{cases} 0 & , C_{max} = 0 \\ \frac{\Delta}{C_{max}} & , C_{max} \neq 0 \end{cases}$$

Hasil dari Konversi gambar menjadi HSV adalah sebagai berikut :



Warna Coklat pada wajah berubah menjadi kemerah-merahan pada color space HSV.

2.2. Threshold Warna Merah

Kita lakukan threshold untuk mengekstraksi wajah dari background nya. Setelah mencoba mencari range nya cocok untuk mengambil warna pada wajah, Nilai yang akan kita gunakan adalah :

$$(H > 0) \text{ and } (H < 40) \text{ and}$$
$$(S > 30) \text{ and } (S < 160) \text{ and}$$
$$(V > 150) \text{ and } (V < 255)$$

Dengan melakukan threshold dimana jika kondisi di atas dipenuhi maka akan ditandai dengan warna putih dan jika tidak dipenuhi akan berwarna hitam diperoleh gambar :

Terlihat cukup banyak noise yang bertebaran pada gambar. Selanjutnya kita akan melakukan filter untuk mengurangi noise.

2.3. Median Filter

Median filter dipilih karena hasil nya yang lebih baik dari pada filter lain dalam mengurangi noise. Kernel yang kita gunakan berukuran 3 x 3. Ukuran tersebut dipilih untuk mengurangi nilai border sehingga tidak perlu dilakukan perlakuan khusus pada border nya nanti.

Pseudo Code Median Filter :

```
allocate outputPixelValue[image width][image height]
allocate window[window width × window height]
edgex := (window width / 2) rounded down
edgey := (window height / 2) rounded down
for x from edgex to image width - edgex do
    for y from edgey to image height - edgey do
        i = 0
        for fx from 0 to window width do
            for fy from 0 to window height do
                window[i] := inputPixelValue[x + fx - edgex][y + fy - edgey]
                i := i + 1
        sort entries in window[]
        outputPixelValue[x][y] := window[window width * window height / 2]
```

Hasil dari median filter :

Dari Gambar di atas terlihat bahwa di beberapa wajah terdapat lubang-lubang pixel hasil kekurangan dari threshold awal tadi. Untuk mengurangi ukuran lubang pada wajah lakukan filter dengan low pass agar kita mendapatkan gambar yang blur. Mengapa gambar blur ? dengan gambar blur maka pinggiran lubang akan tersamarkan.

2.4. Low Pass Filter

Low pass filter memiliki metode yang cukup mudah untuk membuat suatu gambar menjadi blur. Kernel yang digunakan adalah :

$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

Hasil gambarnya :

Dari gambar diatas, terlihat bahwa pinggiran lubang sudah tidak tajam lagi, artinya nilai pixel yang tadinya 0 menjadi bernilai tidak 0. Sekarang kita bisa lakukan thresholding lagi untuk mengecilkan lubang nya.

2.5. Threshold Warna Putih

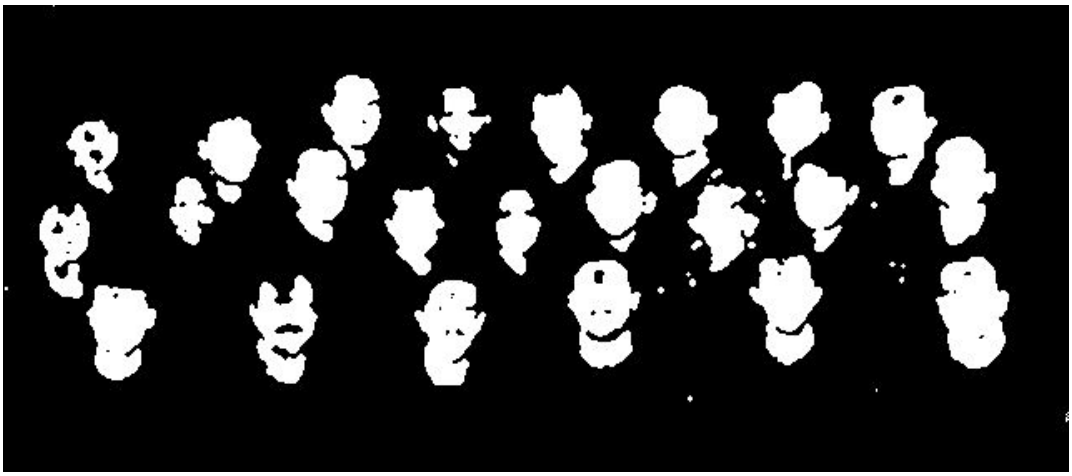
Threshold warna putih ini berguna untuk memperbesar nilai yang sebelumnya blur menjadi lebih putih sehingga akan diperoleh warna putih yang lebih banyak dari sebelumnya berkat blur yang di threshold warna putihnya.

Hasil threshold :

Lubang pada wajah yang semula besar sekarang terlah tertutupi. Namun pinggiran wajah masih tampak kasar sehingga kita lakukan median filter (dengan kondisi kernel yang sama) untuk yang kedua kalinya.

2.6. Median Filter

Hasil dari median filter yang kedua :



Kita bisa menyebut gambar di atas sebagai mask atau subtractor karena gambar di atas akan kita gunakan untuk subtracting gambar awal yaitu gambar ‘FACE DETECTION.png’.

2.7. Pengurangan Image

Dengan melakukan subtracting diperoleh hasilnya sebagai berikut :

Beberapa wajah masih memiliki lubang tetapi sudah lebih baik menyeleksi wajah dibandingkan pada hasil dari gambar ‘contoh_hasil.png’.

Perbandingan

contoh_hasil.png	Hasil (yang diperoleh saat ini)

Gambar sebelah kanan memiliki akurasi seleksi wajah yang lebih baik dan lebih bersih dari pada yang sebelah kiri. Gambar contoh_hasil masih menyeleksi rambut dan juga terdapat noise. Perlu diingat bahwa hasil yang kita dapat adalah seleksi wajah saja tanpa menyeleksi rambutnya.

Referensi :

1. Konversi dari RGB ke HSV
2. Threshold Warna Kulit
3. Median Filter
4. Low Pass Filter