**Nama : Rizky Nurhadhi**

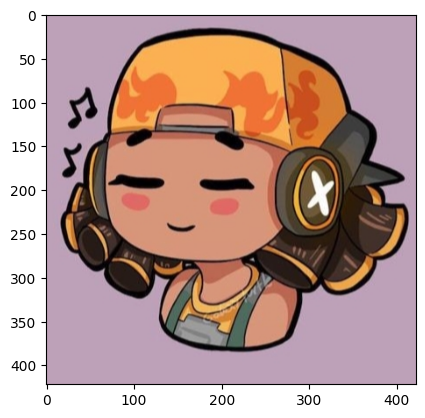
**Nim : 1207070113**

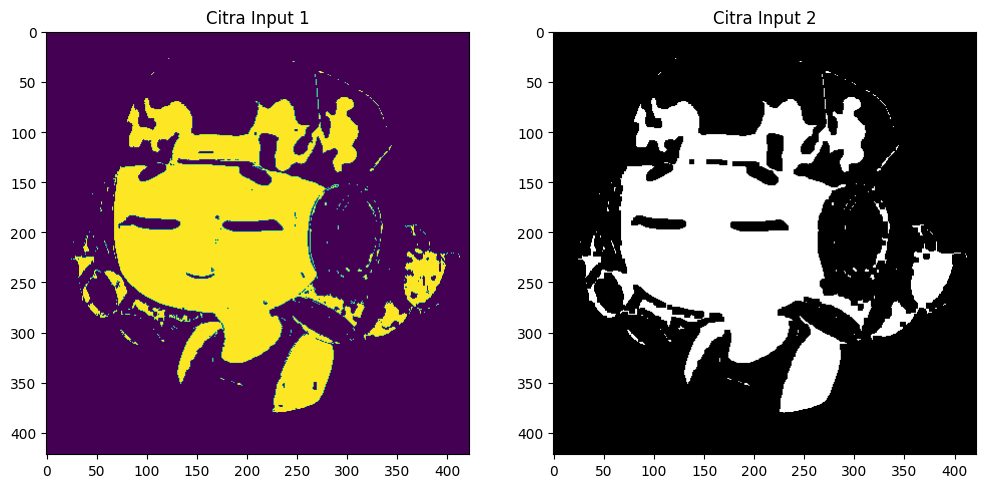
**Kelas : Teknik Elektro - TKK**

**Praktikum 11 Pengolahan Citra Digital – Morfologi dan CNN**

<https://github.com/CrossOne0708/prakPCD_11.git>

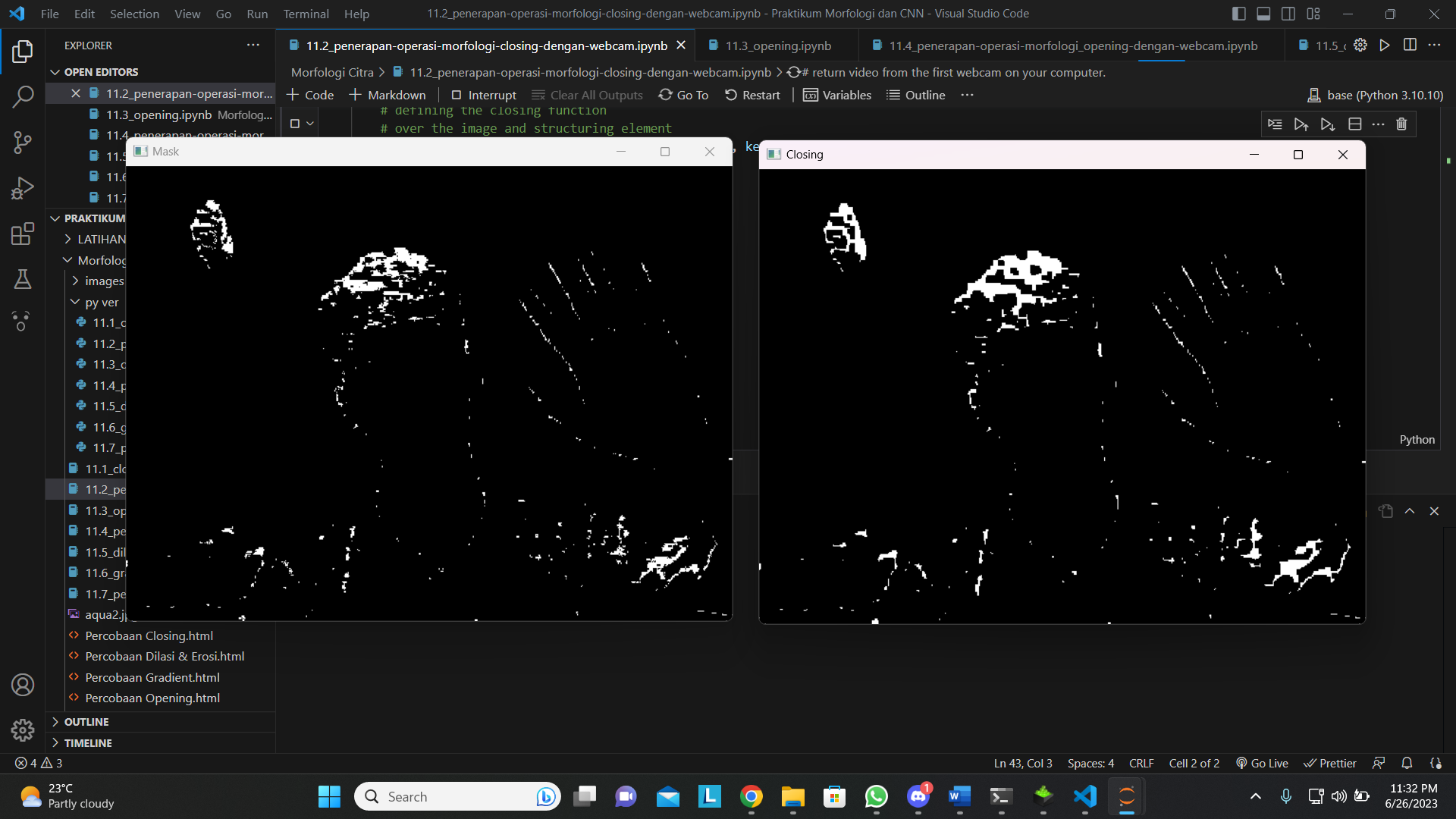
1. **MORFOLOGI**
2. **Closing**

****

****

Program yang telah dicoba dalam latihan pertama menggunakan library OpenCV, NumPy, skimage, dan matplotlib untuk memproses citra. Setelah membaca gambar input, program melakukan konversi ruang warna dari BGR ke HSV. Selanjutnya, program menggunakan rentang warna tertentu untuk membuat mask dan melakukan operasi bitwise pada gambar input. Kemudian, program menerapkan operasi morfologi closing dengan menggunakan kernel yang telah ditentukan. Akhirnya, citra input dan citra hasil closing ditampilkan dalam plot yang dibuat menggunakan matplotlib.

1. **Penerapan Operasi Morfologi Closing Dengan Webcam**



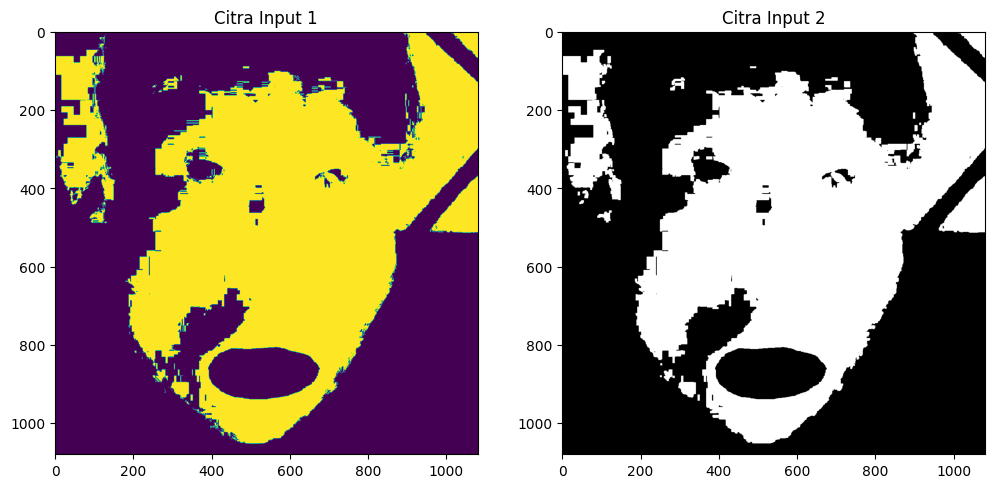
Program yg telah dicoba merupakan contoh penerapan operasi morfologi closing pada video yang diambil menggunakan webcam. Program ini menggunakan library OpenCV dan NumPy.

Pada awal program, video dari webcam pertama diinisialisasi dan dilakukan looping untuk terus membaca frame dari kamera. Setiap frame kemudian diubah ke ruang warna HSV menggunakan fungsi cv2.cvtColor(). Kemudian, program menentukan rentang warna tertentu (dalam hal ini, warna biru) untuk membuat mask menggunakan fungsi cv2.inRange(). Mask ini kemudian digunakan untuk melakukan operasi bitwise pada frame asli menggunakan fungsi cv2.bitwise\_and(), menghasilkan gambar hasil filtrasi warna.

Selanjutnya, program mendefinisikan kernel atau elemen struktural yang digunakan dalam operasi morfologi closing. Kernel yang digunakan adalah kernel persegi berukuran 5x5. Operasi morfologi closing dilakukan pada mask menggunakan fungsi cv2.morphologyEx() dengan parameter cv2.MORPH\_CLOSE. Hasil dari operasi closing ini ditampilkan dalam jendela yang terpisah menggunakan fungsi cv2.imshow(). Program ini akan berjalan terus menerus sampai tombol 'a' ditekan. Setelah itu, program akan menutup semua jendela yang terbuka dan melepas sumber daya yang digunakan.

1. **Opening**

****

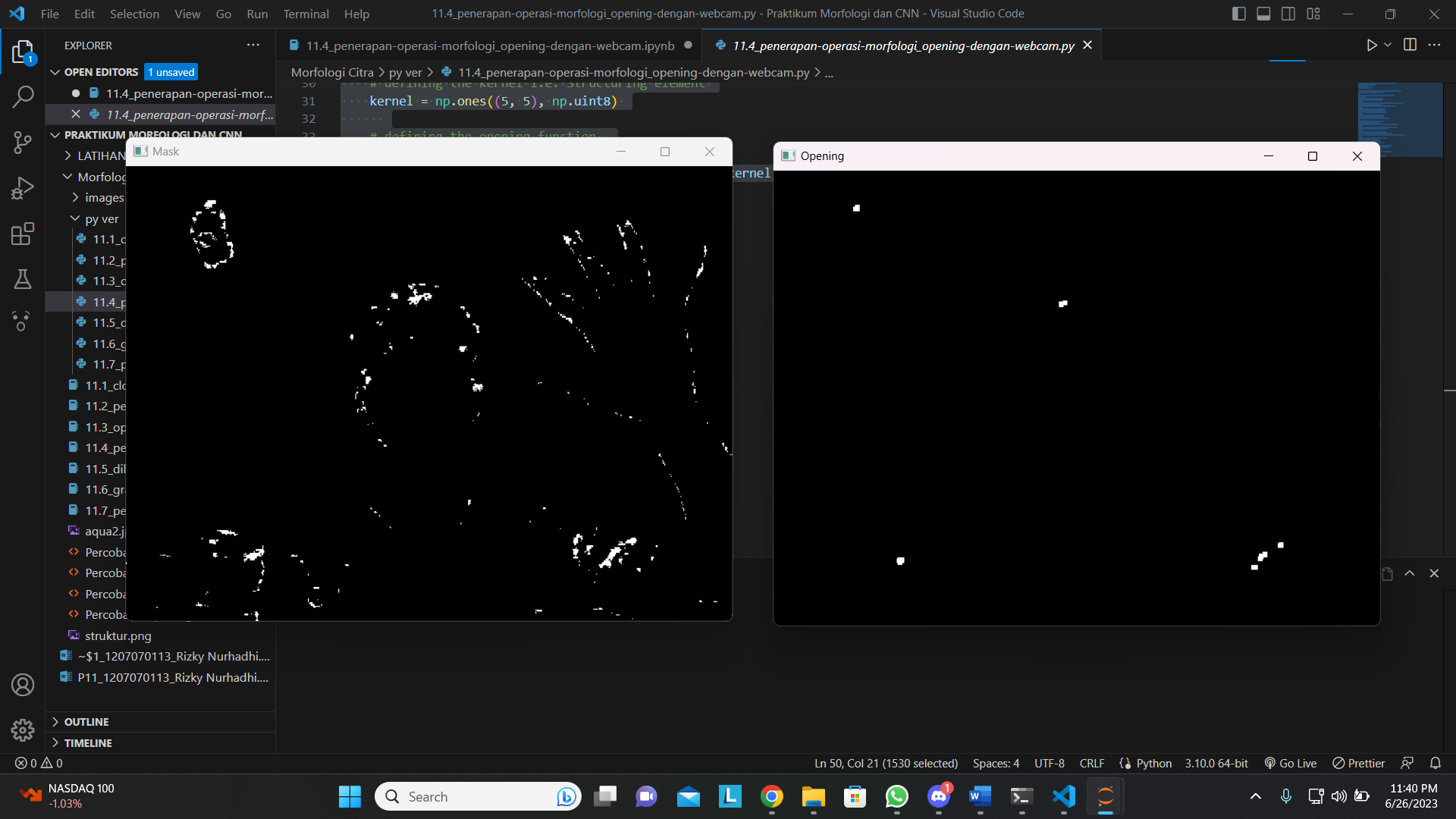
****

Program yang telah dicoba ini merupakan contoh penerapan operasi morfologi opening pada sebuah gambar. Proses morfologi opening dimulai dengan mengubah ruang warna gambar ke HSV menggunakan fungsi cv2.cvtColor(). Selanjutnya, ditentukan rentang warna tertentu (dalam hal ini, rentang warna biru) untuk membuat mask menggunakan fungsi cv2.inRange(). Mask ini kemudian digunakan untuk melakukan operasi bitwise pada gambar asli menggunakan fungsi cv2.bitwise\_and(), menghasilkan gambar hasil filtrasi warna.

Selanjutnya, program mendefinisikan kernel atau elemen struktural yang digunakan dalam operasi morfologi opening. Kernel yang digunakan adalah kernel persegi berukuran 5x5. Operasi morfologi opening dilakukan pada mask menggunakan fungsi cv2.morphologyEx() dengan parameter cv2.MORPH\_OPEN. Hasil dari operasi opening ini ditampilkan dalam jendela yang terpisah menggunakan fungsi plt.imshow().

Program ini menampilkan dua citra dalam satu jendela. Citra pertama adalah citra mask yang menunjukkan area di mana rentang warna biru ditemukan. Citra kedua adalah citra hasil dari operasi opening, yang menghasilkan gambar dengan noise yang lebih sedikit dan objek biru yang lebih terdefinisi.

1. **Penerapan Operasi Morfologi Opening Dengan Webcam**

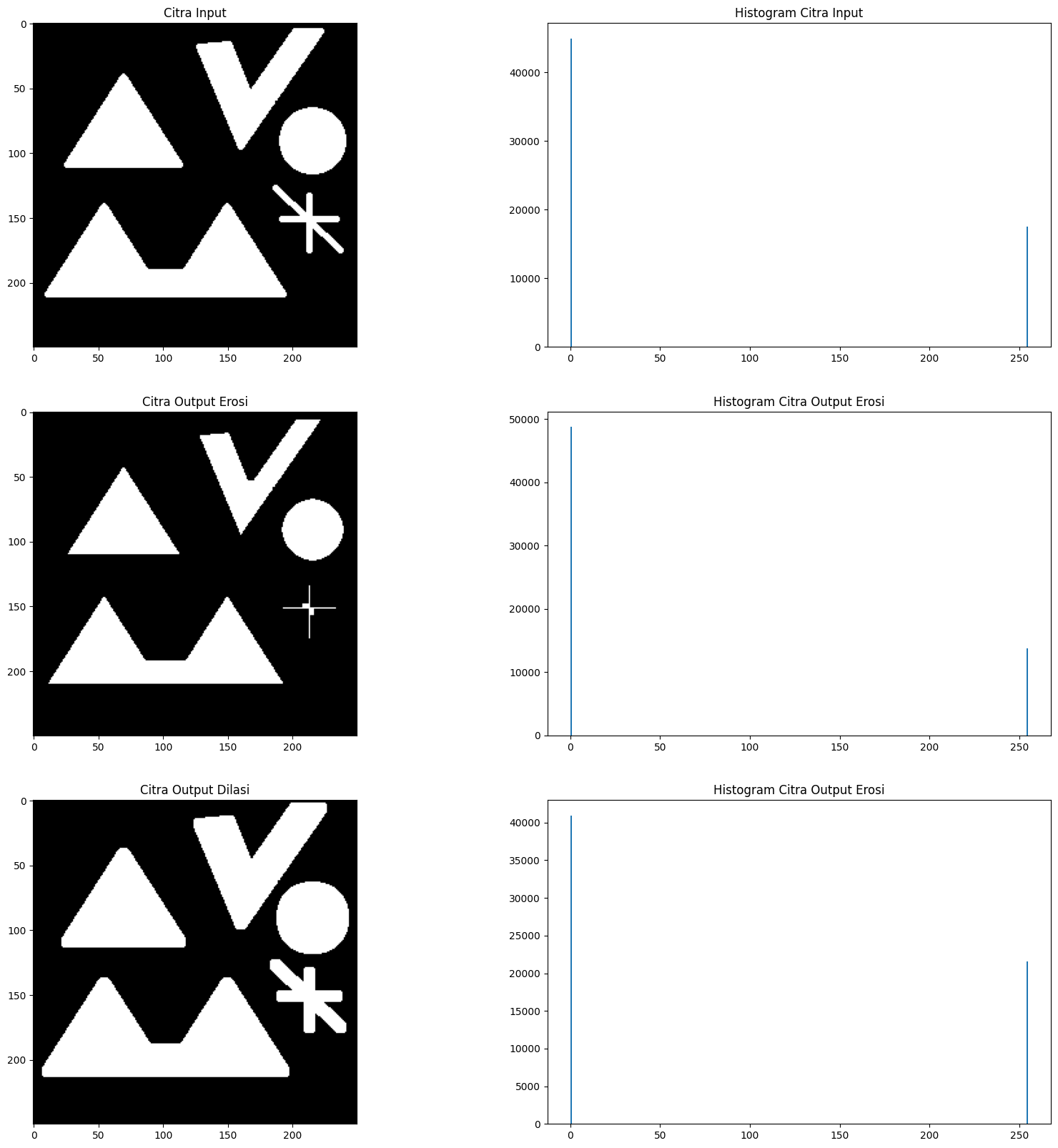


Pada program yang telah dicoba ini merupakan contoh penerapan operasi morfologi opening pada video yang diambil menggunakan webcam. Program ini menggunakan library OpenCV dan NumPy. Pada awal program, dilakukan import library yang diperlukan. Kemudian, video dari webcam pertama pada komputer diambil menggunakan fungsi cv2.VideoCapture(0).

Program ini memiliki loop while yang berjalan selama inisialisasi pengambilan video telah dilakukan. Di dalam loop ini, dilakukan pembacaan frame dari kamera menggunakan fungsi screenRead.read(). Frame yang didapatkan kemudian diubah ke dalam ruang warna HSV menggunakan fungsi cv2.cvtColor().Selanjutnya, ditentukan rentang warna tertentu (dalam hal ini, rentang warna biru) untuk membuat mask menggunakan fungsi cv2.inRange(). Mask ini kemudian digunakan untuk melakukan operasi bitwise pada frame asli menggunakan fungsi cv2.bitwise\_and(), menghasilkan frame hasil filtrasi warna.

Kemudian, program mendefinisikan kernel atau elemen struktural yang digunakan dalam operasi morfologi opening. Kernel yang digunakan adalah kernel persegi berukuran 5x5. Operasi morfologi opening dilakukan pada mask menggunakan fungsi cv2.morphologyEx() dengan parameter cv2.MORPH\_OPEN. Hasil dari operasi opening ini ditampilkan dalam jendela terpisah menggunakan fungsi cv2.imshow(). Dalam program ini, jendela menampilkan dua citra secara bersamaan. Citra pertama adalah citra mask yang menunjukkan area di mana rentang warna biru ditemukan. Citra kedua adalah citra hasil dari operasi opening, yang menghasilkan gambar dengan noise yang lebih sedikit dan objek biru yang lebih terdefinisi.

1. **Dilasi dan Erosi**

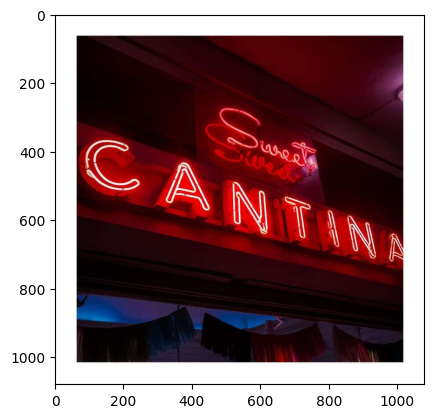


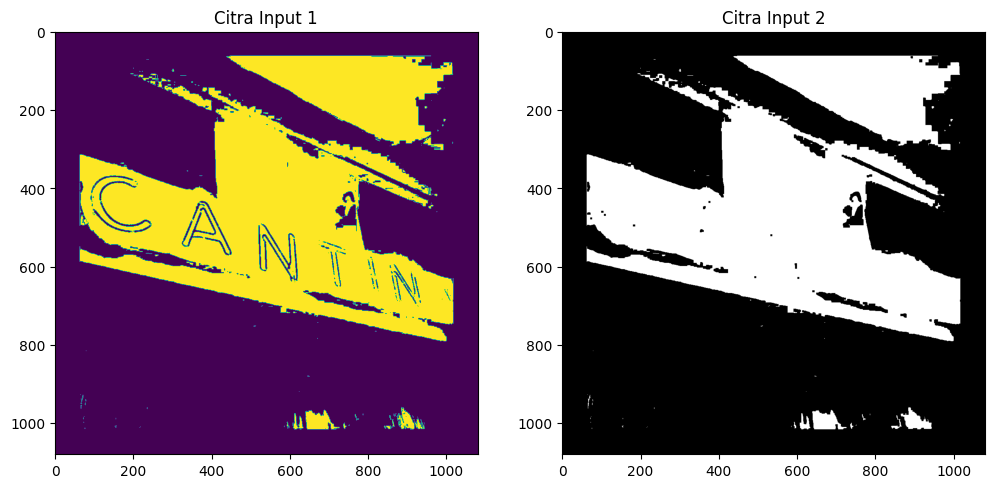
Percobaan dilasi (erosion) dan erosi (dilation) pada citra dalam program ini menggunakan library OpenCV, NumPy, dan Matplotlib. Citra input dibaca menggunakan fungsi cv2.imread() dengan mode grayscale (0) dan disimpan dalam variabel 'img'. Selanjutnya, dilakukan inisialisasi kernel sebagai matriks berukuran 5x5 dengan nilai elemen 1 menggunakan np.ones(). Kernel ini akan digunakan dalam operasi dilasi dan erosi.

Proses dilasi dan erosi dilakukan menggunakan fungsi cv2.erode() dan cv2.dilate(). Pada proses erosi, citra input 'img' dierosi dengan menggunakan kernel yang telah ditentukan sebanyak satu kali iterasi. Hasilnya disimpan dalam variabel 'img\_erosion'. Pada proses dilasi, citra input 'img' didilasi dengan menggunakan kernel yang sama sebanyak satu kali iterasi. Hasilnya disimpan dalam variabel 'img\_dilation'.

Kemudian, dilakukan plotting citra dan histogram menggunakan Matplotlib. Dalam program ini, dibuat subplot dengan ukuran 3x2 untuk menampilkan citra input, histogram citra input, citra output erosi, histogram citra output erosi, citra output dilasi, dan histogram citra output dilasi. Citra dan histogram ditampilkan menggunakan fungsi plt.imshow() dan plt.hist(), masing-masing dengan parameter citra yang akan ditampilkan dan jumlah bin histogram.

1. **Gradient**

****

****

Program yang telah dicoba ini merupakan contoh penerapan operasi morfologi gradient pada citra menggunakan library OpenCV, NumPy, dan Matplotlib. Program ini membaca citra input. Kemudian citra input diubah menjadi format HSV menggunakan fungsi cv2.cvtColor(). Lalu, rentang masking atau batasan nilai warna untuk operasi morfologi ditentukan dengan inisialisasi variabel blue1 dan blue2. Selanjutnya, dilakukan pembuatan mask menggunakan cv2.inRange() untuk memfilter piksel berdasarkan rentang warna yang ditentukan.

Setelah itu, dilakukan operasi bitwise\_and antara citra input dan mask untuk mempertahankan piksel yang sesuai dengan rentang warna. Kemudian, ditentukan kernel sebagai elemen struktur menggunakan np.ones(). Kernel ini digunakan dalam operasi morfologi. Pada tahap berikutnya, dilakukan operasi morfologi closing pada mask menggunakan cv2.morphologyEx(). Hasilnya disimpan dalam variabel 'closing'.  
 Terakhir, dilakukan plotting citra menggunakan Matplotlib dengan menampilkan citra mask (Citra Input 1) dan citra hasil morfologi closing (Citra Input 2) dalam subplot. Visualisasi citra ini membantu dalam memahami konsep dan efek dari operasi morfologi gradient pada citra.

1. **Penerapan Operasi Morfologi Gradient dengan Webcam**



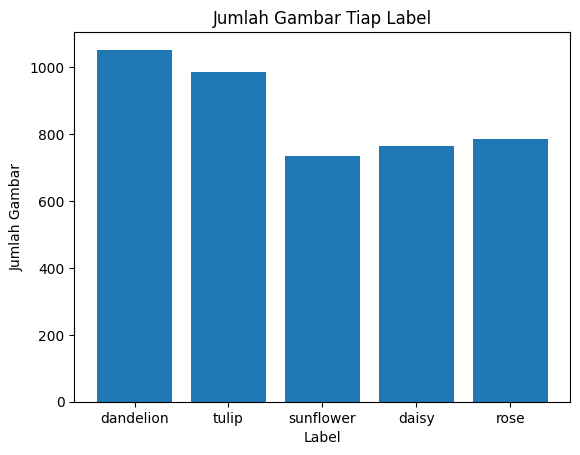
Pada percobaan ini yaitu contoh penerapan operasi morfologi gradient pada citra yang diambil melalui webcam menggunakan library OpenCV dan NumPy. Program ini berfungsi untuk membaca citra dari webcam, mengkonversinya ke dalam ruang warna HSV, dan melakukan operasi morfologi gradient pada citra tersebut.

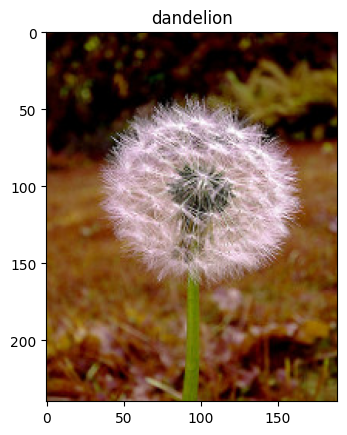
Pertama, program mengakses video dari webcam menggunakan cv2.VideoCapture(). Selanjutnya, dilakukan loop untuk membaca setiap frame dari kamera. Setiap frame yang dibaca diubah ke ruang warna HSV menggunakan cv2.cvtColor(). Kemudian, ditentukan rentang warna yang akan digunakan dalam masking dengan menginisialisasi variabel blue1 dan blue2.

Selanjutnya, dilakukan pembuatan mask dengan cv2.inRange() untuk memfilter piksel berdasarkan rentang warna yang ditentukan sebelumnya. Mask ini akan digunakan dalam operasi morfologi gradient. Selanjutnya, dilakukan operasi bitwise\_and antara citra asli dan mask untuk mempertahankan piksel yang sesuai dengan rentang warna dan ditentukan kernel sebagai elemen struktur dengan menggunakan np.ones(). Kernel ini akan digunakan dalam operasi morfologi.

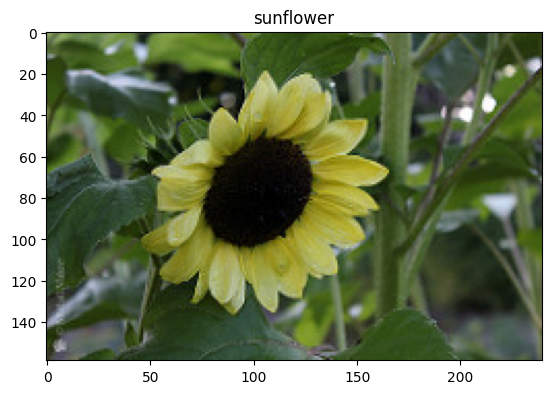
Pada tahap berikutnya, dilakukan operasi morfologi gradient pada mask menggunakan cv2.morphologyEx() dengan parameter cv2.MORPH\_GRADIENT. Hasilnya disimpan dalam variabel 'gradient'. Terakhir, citra hasil operasi morfologi gradient ditampilkan dalam jendela menggunakan cv2.imshow(). Program akan terus berjalan sampai tombol 'a' ditekan.

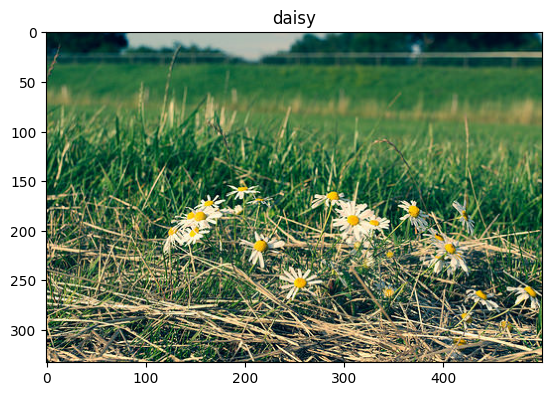
1. **CNN (Convolutional Neural Network)**
2. **Template Matching**

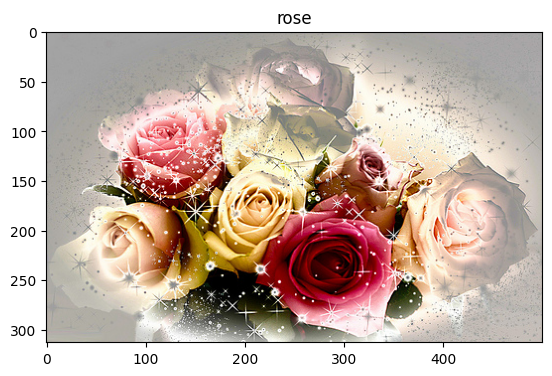
****

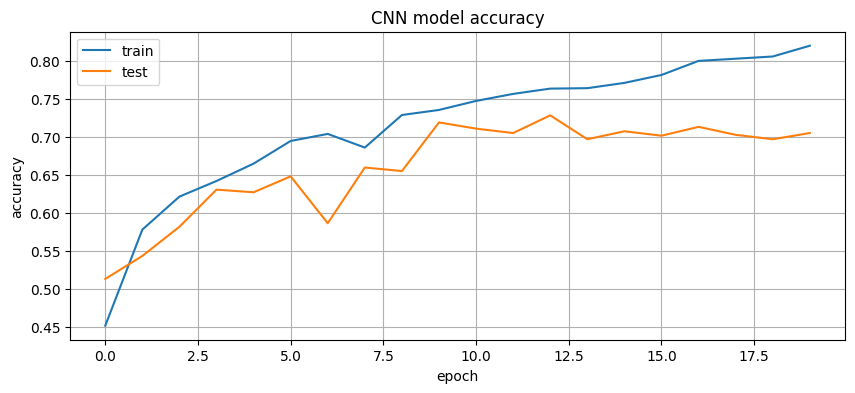
****

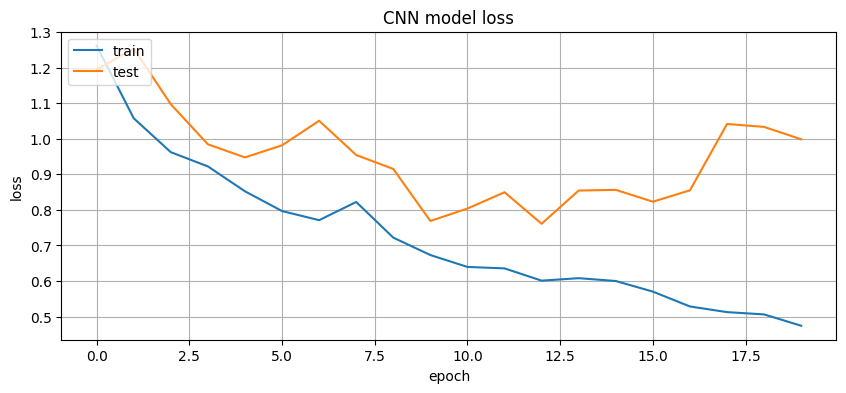
****

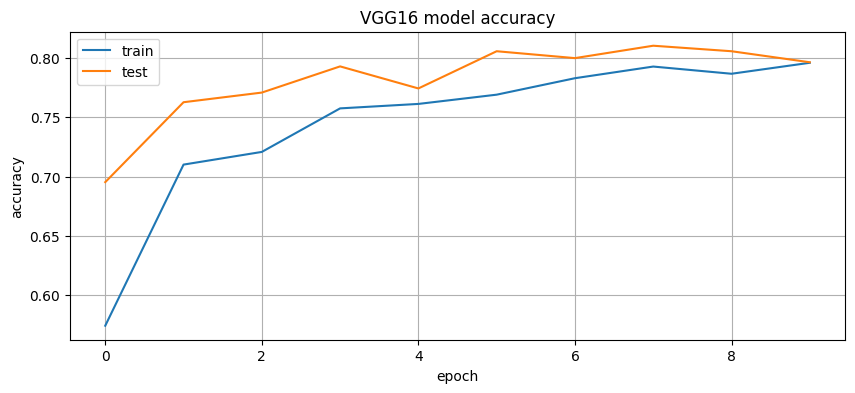
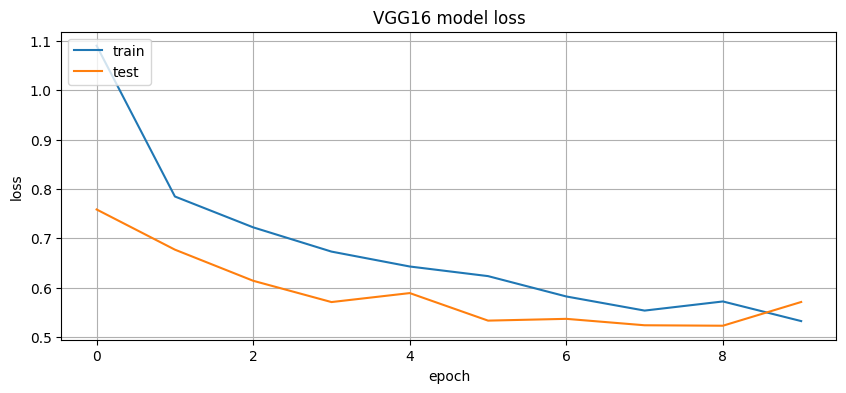
****

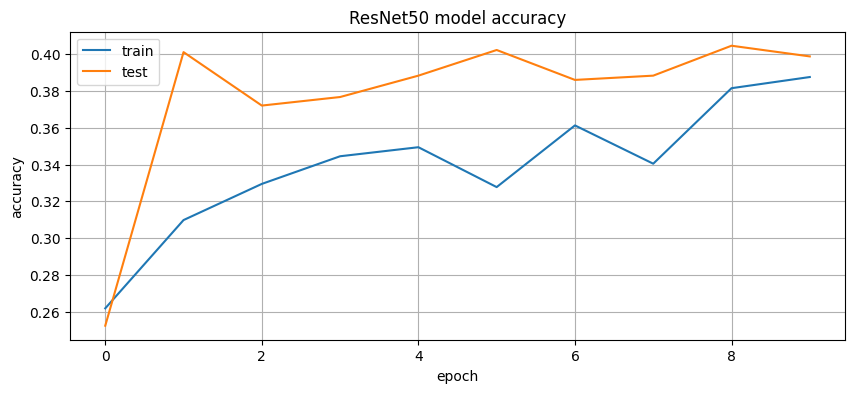
****

****

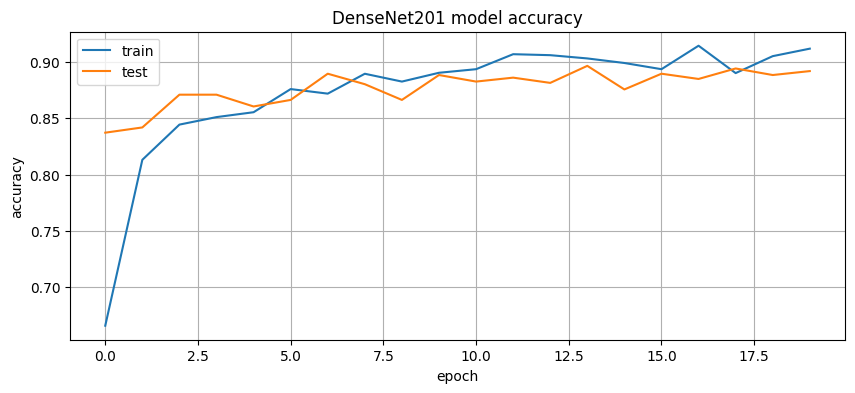
****

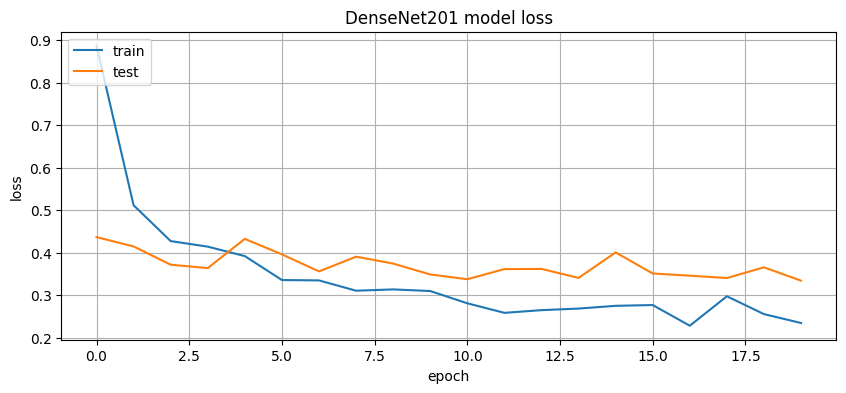
****

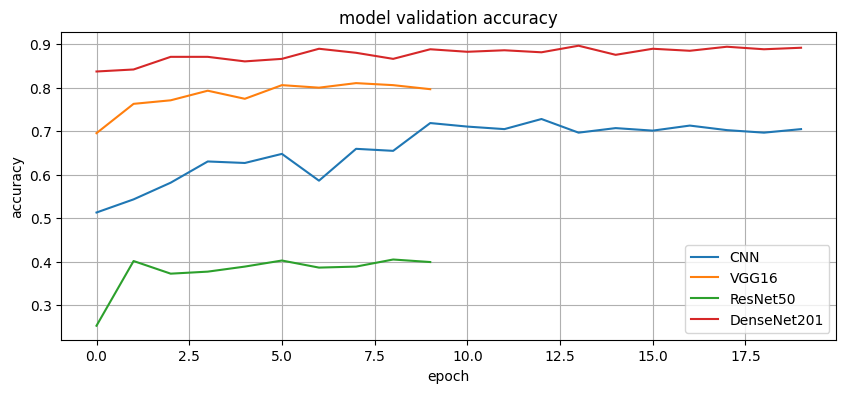
****

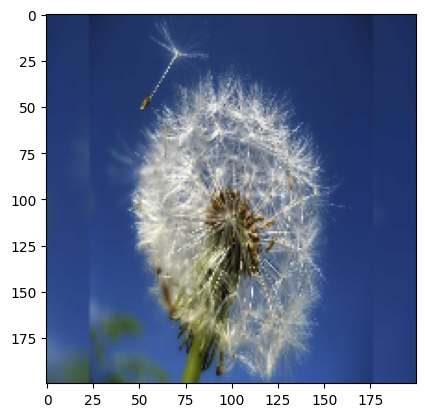
****

****

****

****

****

****

Pada percobaan ini dicoba program untuk implementasi beberapa metode pengolahan citramenggunakan TensorFlow dan Keras. Pertama, program mengimpor library yang diperlukan dan mengunduh dataset "flowers". Setelah itu, program memeriksa jumlah gambar pada setiap kelas dalam dataset dan menampilkan sampel gambar untuk setiap kelas. Kemudian, dilakukan pengolahan data dengan menggunakan ImageDataGenerator untuk preprocessing dan pembagian data menjadi data latih dan data validasi. Selanjutnya, program membuat arsitektur model CNN dan melatih model tersebut. Setelah itu, dilakukan transfer learning dengan menggunakan model VGG16, ResNet50, dan DenseNet201 yang telah dilatih pada dataset ImageNet. Model-model ini di-compile dan dilatih menggunakan metode yang serupa dengan model CNN. Kemudian pada akhirnya, program menampilkan visualisasi akurasi dari model-model yang telah dilatih.