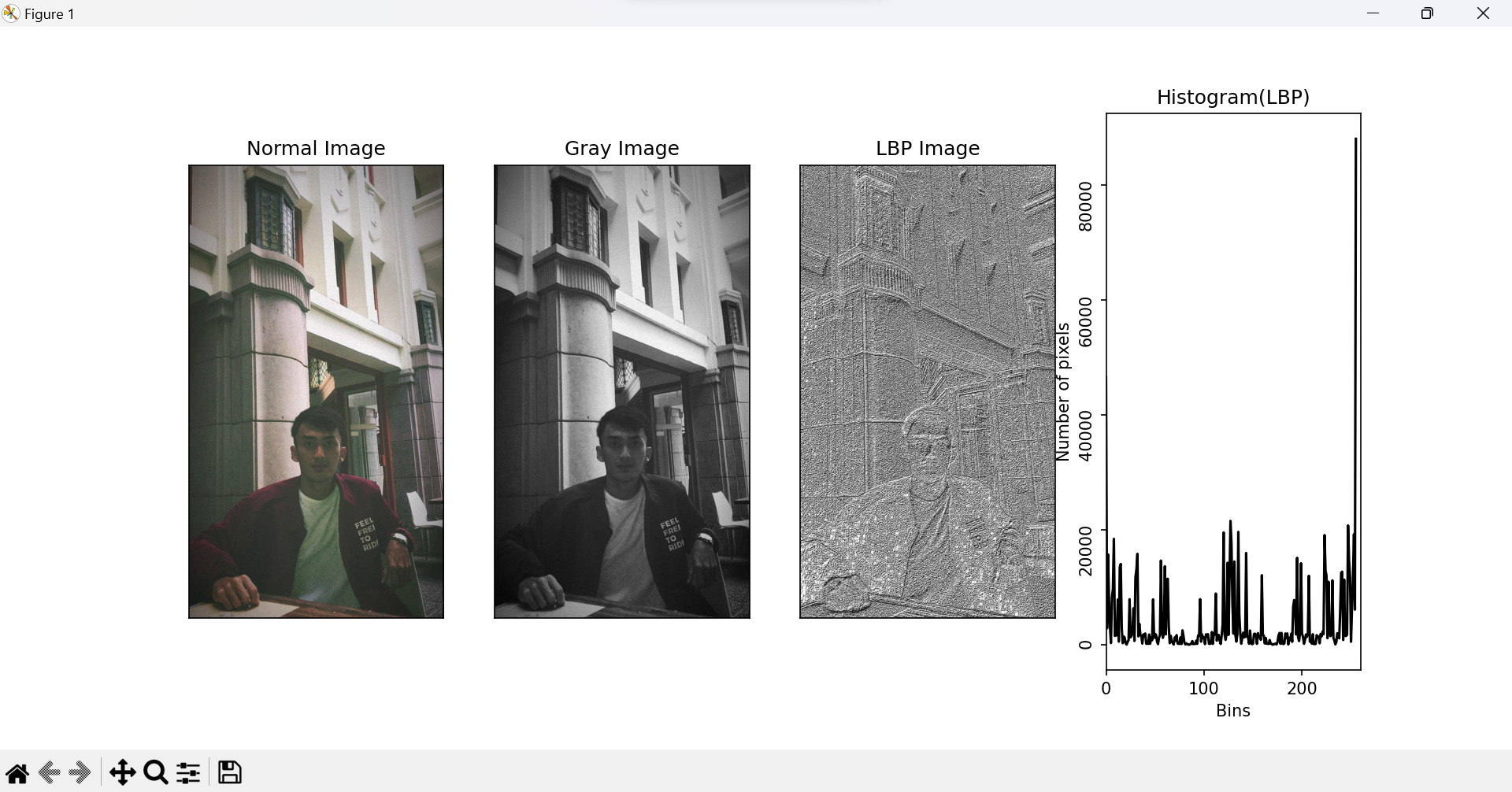
Praktikum 10

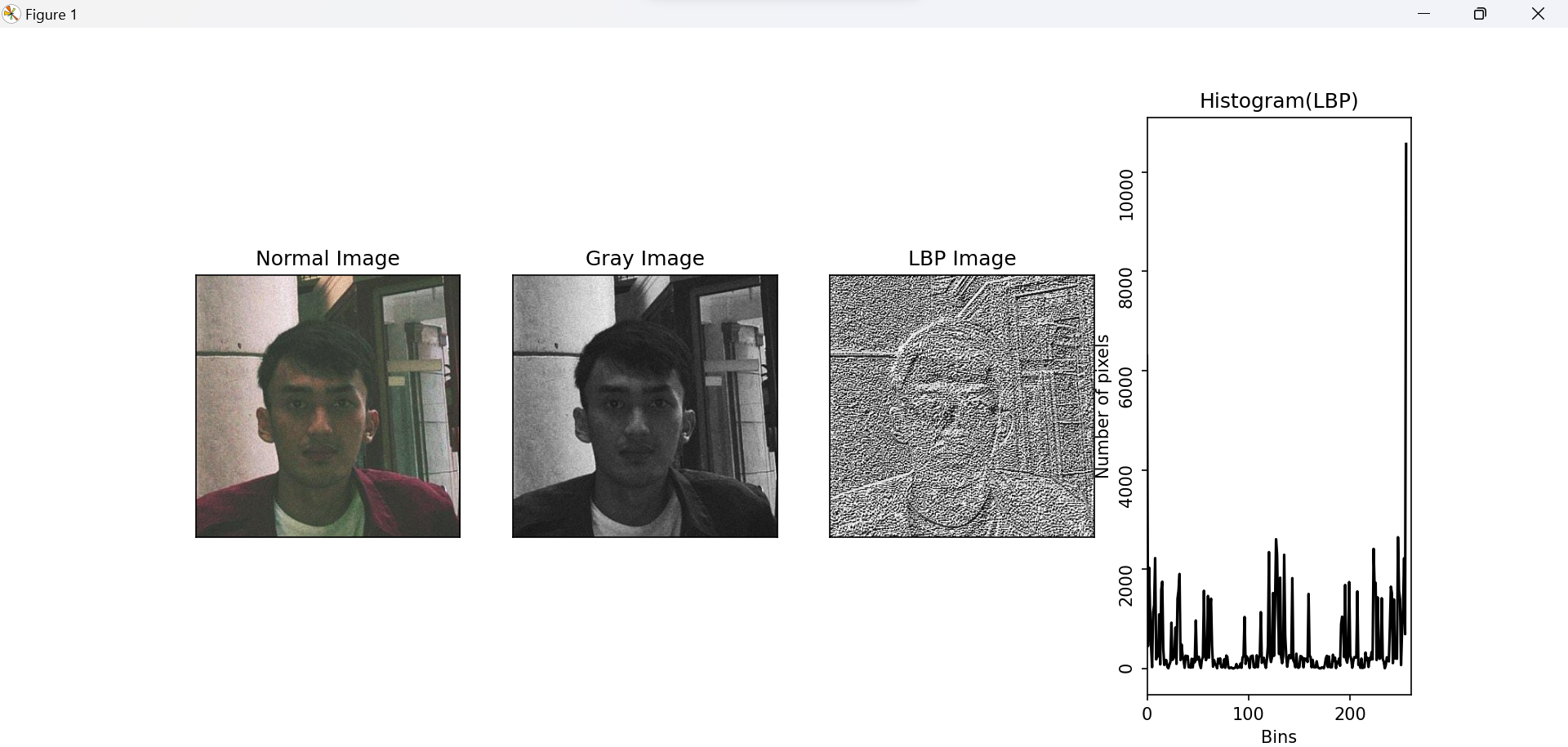
Nama : Reilham Riziq Algani

Kelas : PCD TT

NIM : 1207070101

1. Ekstraksi LBP





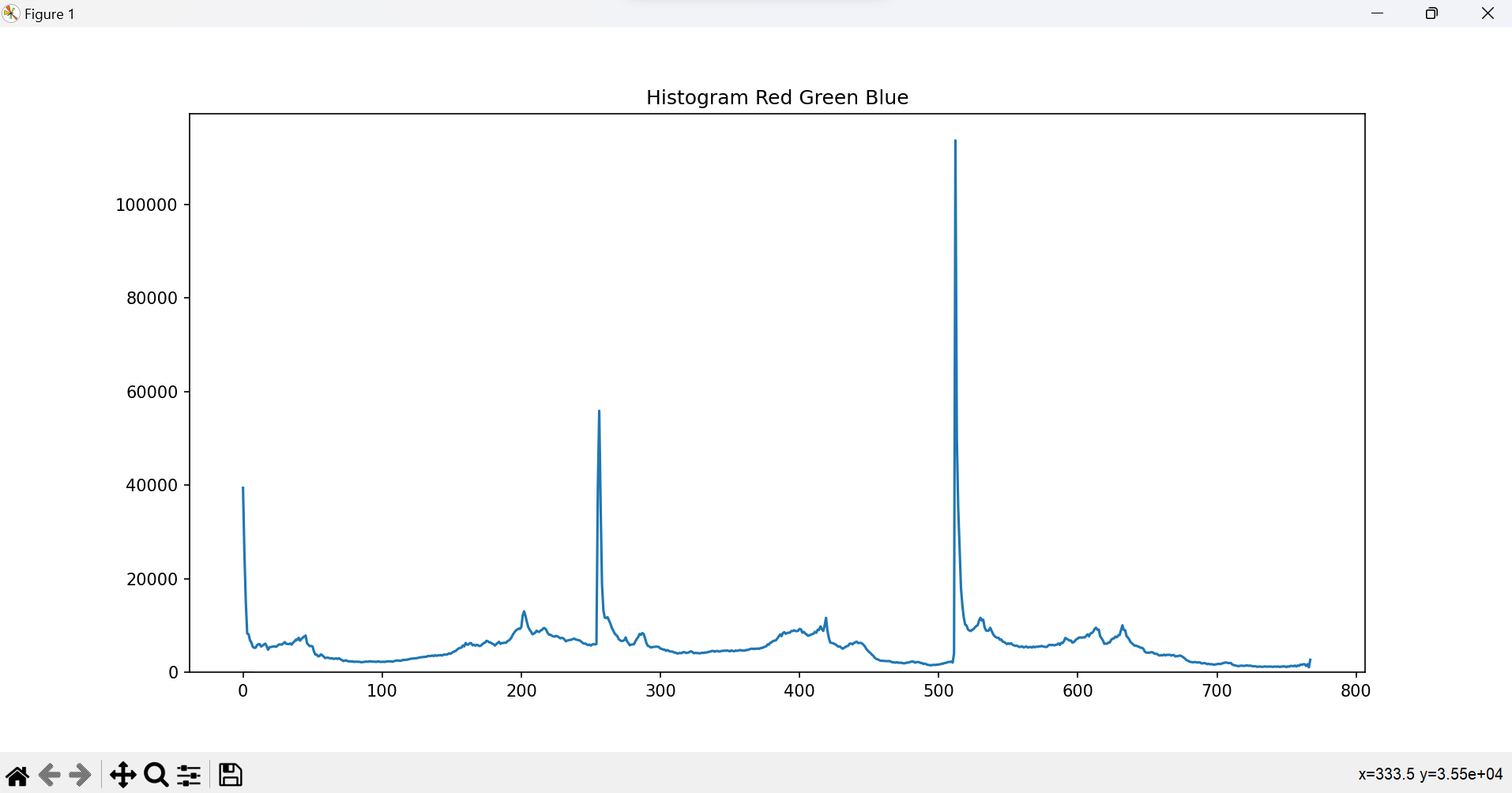
Dalam program ini, digunakan pustaka OpenCV (cv2) untuk membaca dan mengubah gambar. Fungsi get\_pixel digunakan untuk mendapatkan nilai pixel dari tetangganya dengan membandingkannya dengan nilai pixel pusat. Fungsi lbp\_calculated\_pixel digunakan untuk menghitung nilai Local Binary Patterns (LBP) pada suatu pixel berdasarkan nilai pixel tetangganya. Fungsi show\_output digunakan untuk menampilkan gambar hasil dan histogram menggunakan pustaka Matplotlib.

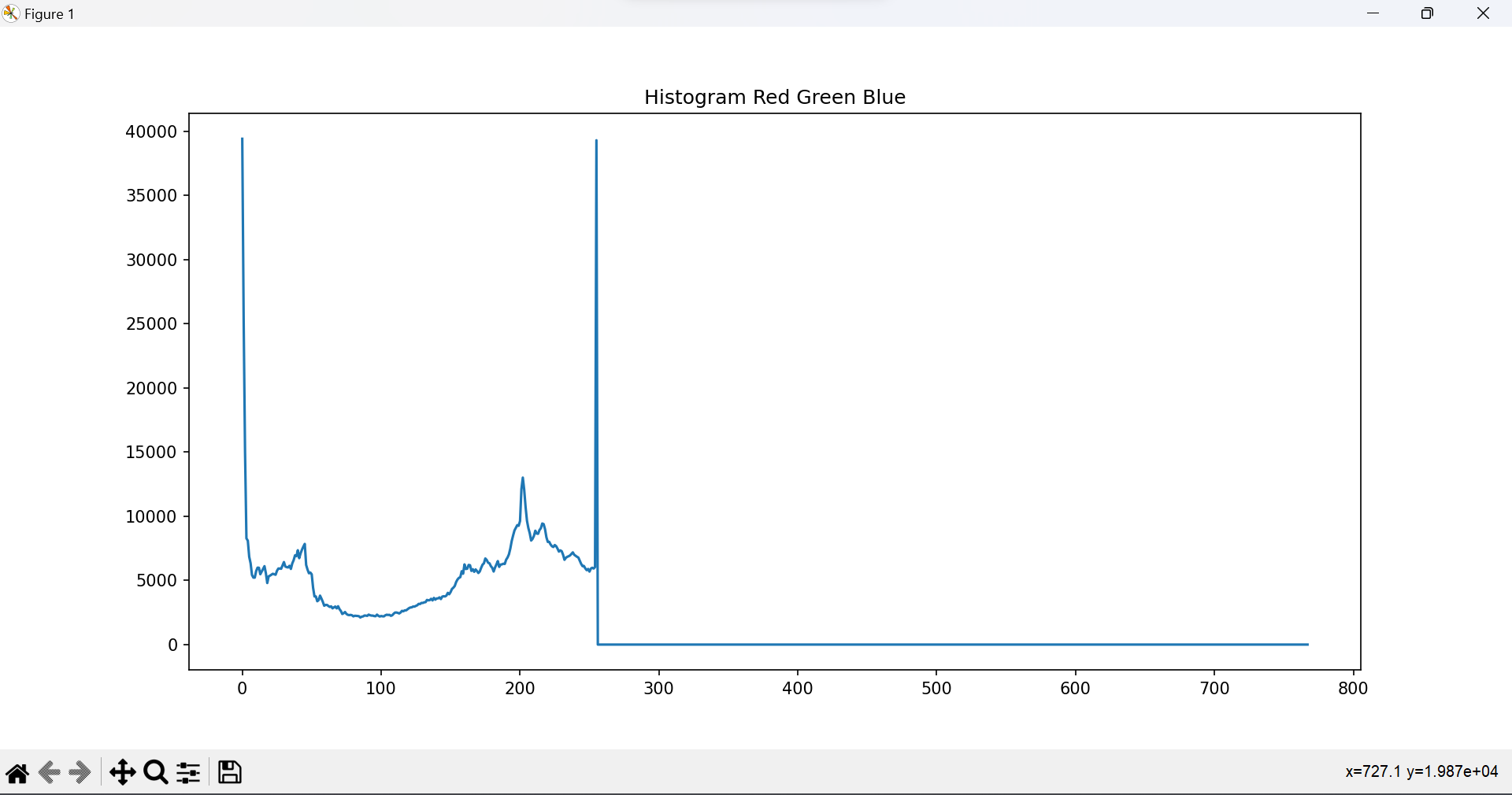
Proses inti dari program ini melibatkan langkah-langkah berikut: membaca gambar, mengubahnya menjadi citra grayscale, menghitung LBP untuk setiap pixel, menghitung histogram LBP, dan menampilkan hasilnya.

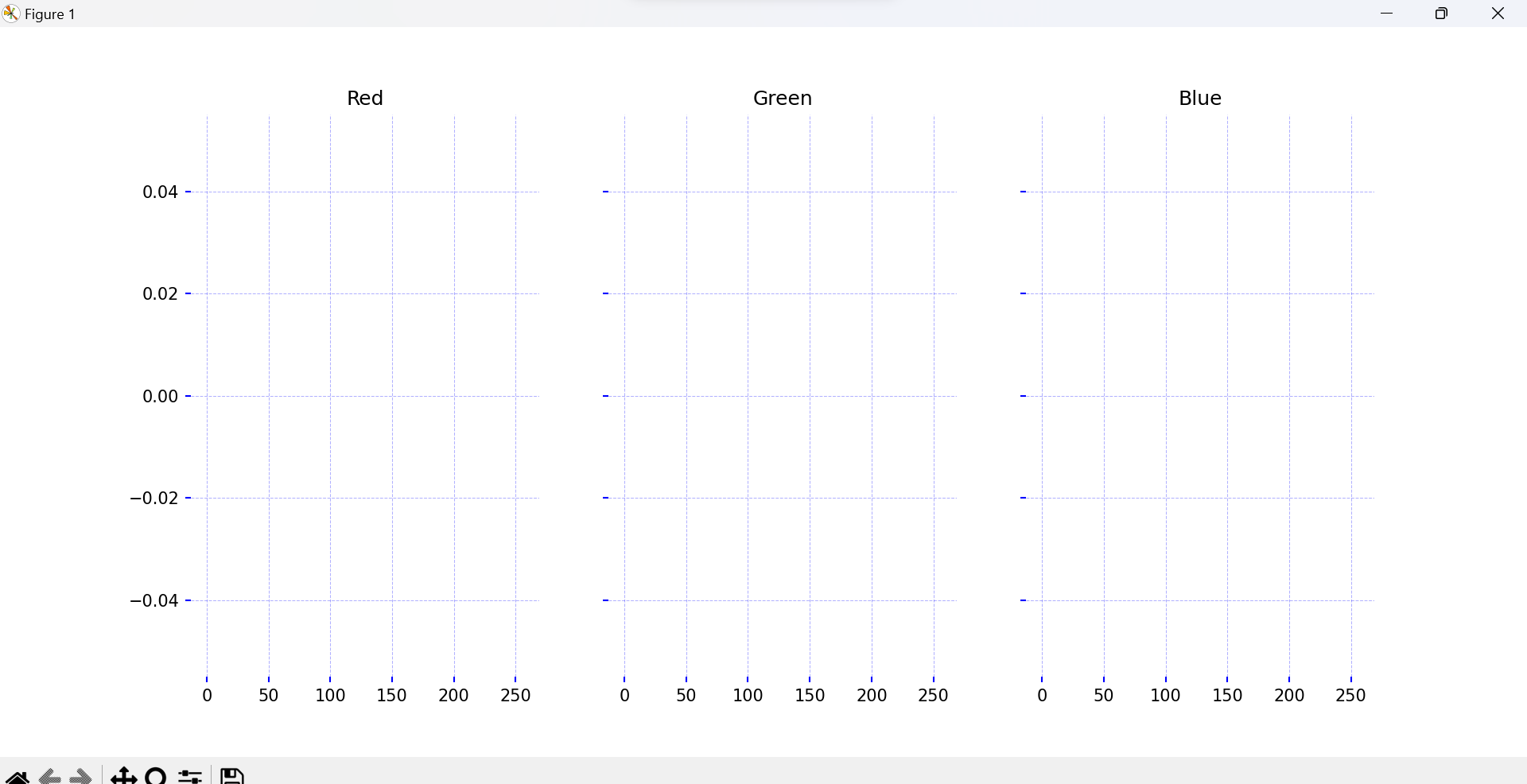
Program ini juga menyertakan contoh penggunaan dengan menggunakan gambar yang memiliki tingkat kecerahan yang berbeda.

Pada akhir program, hasil LBP ditampilkan dalam bentuk gambar dan histogram.

1. Etraksi fitur berwarna







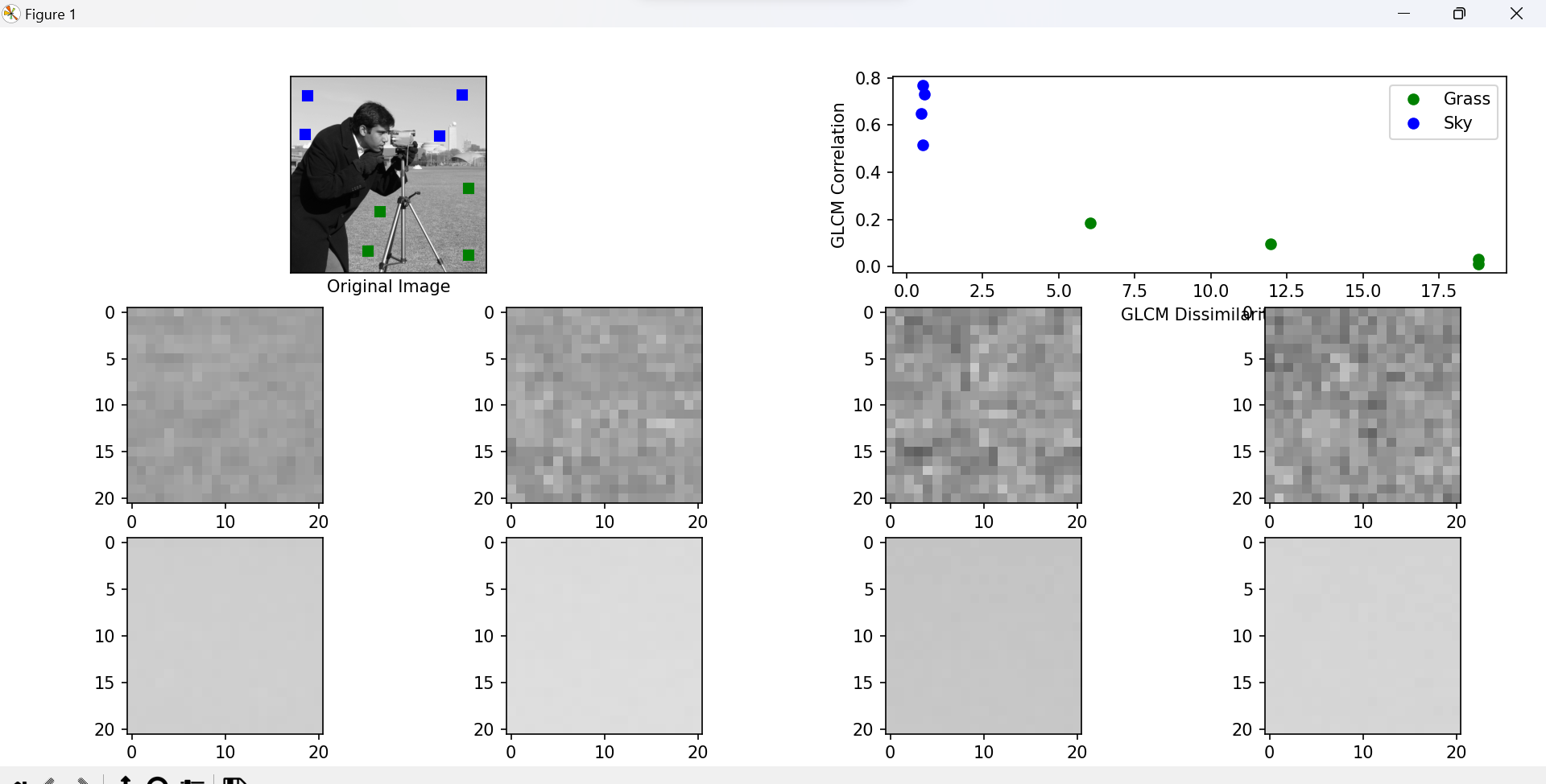
Dalam program ini, digunakan pustaka NumPy, imageio, Matplotlib, dan OpenCV (cv2) untuk membaca, memanipulasi, dan menampilkan gambar.

Fungsi makeItZero digunakan untuk menginisialisasi sebuah array dengan nilai 0.

Program ini melakukan ekstraksi fitur berdasarkan warna dengan cara menghitung histogram dari komponen warna (merah, hijau, biru) dan menampilkan histogram RGB menggunakan Matplotlib.

Selain itu, program juga menggunakan fungsi cv2.calcHist dari OpenCV untuk menghitung histogram warna dan menampilkannya.

1. Etraksi fitur GLCM



Dalam program ini, digunakan pustaka Matplotlib dan scikit-image (skimage) untuk memanipulasi dan menampilkan gambar.

Program ini menggunakan gambar "camera" dari pustaka scikit-image sebagai input.

Program ini melakukan ekstraksi fitur GLCM (Gray-Level Co-occurrence Matrix) dengan mengambil beberapa area di dalam gambar (seperti rumput dan langit) dan menghitung GLCM untuk setiap area. GLCM dihitung untuk setiap bagian menggunakan fungsi graycomatrix dari pustaka scikit-image.

Fitur-fitur GLCM seperti dissimilarity dan correlation dihitung menggunakan fungsi graycoprops dari pustaka scikit-image.

Hasilnya ditampilkan dalam bentuk scatter plot, dengan sumbu x dan y masing-masing mewakili dissimilarity dan correlation dari setiap bagian.

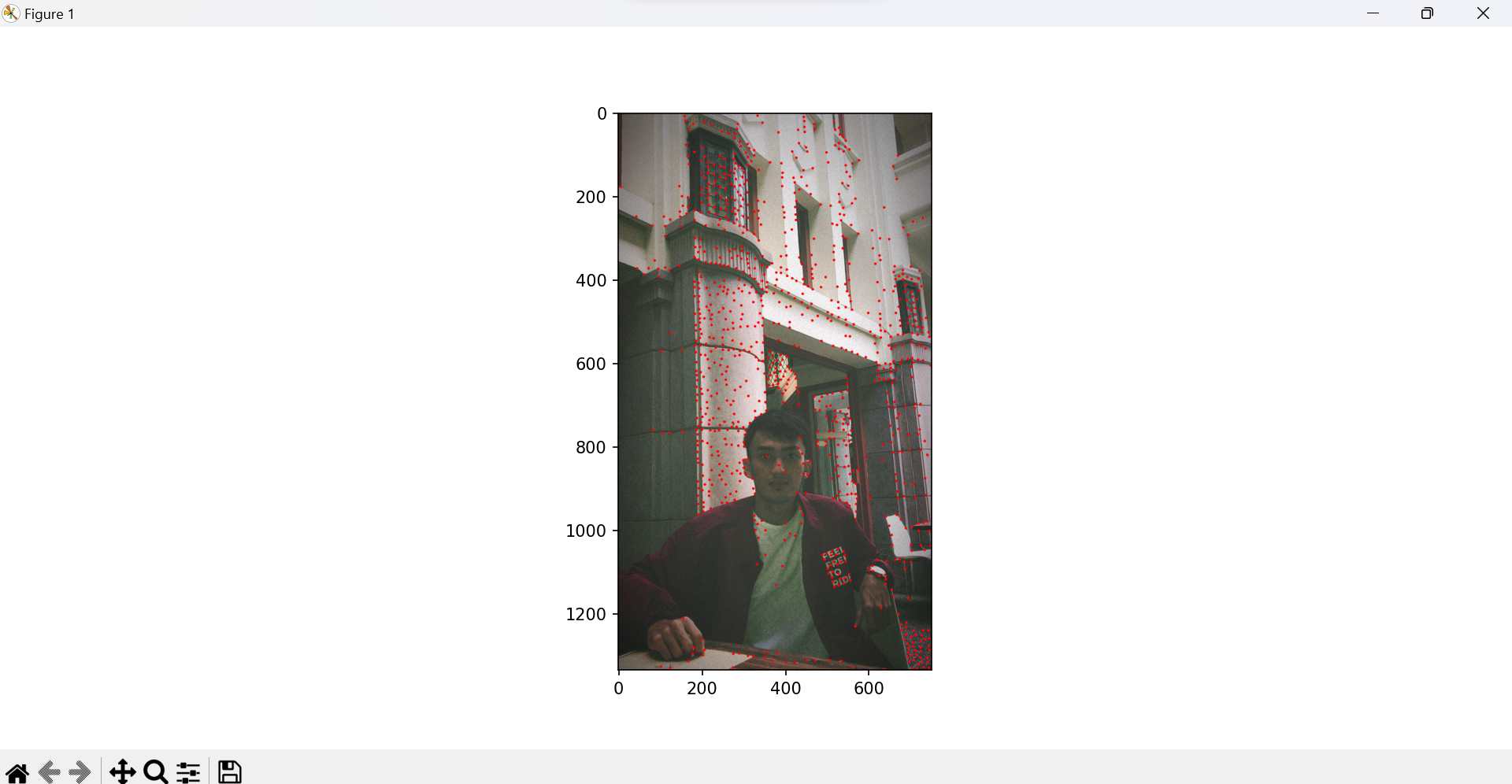
1. **Kesimpulan Perbandingan Ekstraksi Fitur LBP, Berdasarkan Warna, dan GLCM :**

Dalam perbandingan ketiga program tersebut, terdapat perbedaan dalam metode ekstraksi fitur yang digunakan. Program LBP berfokus pada ekstraksi fitur tekstur menggunakan LBP, sedangkan program berbasis warna berfokus pada ekstraksi fitur berdasarkan distribusi warna dalam gambar. Program GLCM berfokus pada ekstraksi fitur berdasarkan hubungan spasial antara piksel dalam gambar.

Pemilihan metode ekstraksi fitur yang sesuai tergantung pada jenis analisis yang ingin dilakukan dan karakteristik gambar yang akan diproses. LBP cocok untuk analisis tekstur atau menggambarkan pola tekstur gambar, ekstraksi fitur berdasarkan warna cocok untuk menganalisis distribusi warna dalam gambar, dan GLCM cocok untuk menganalisis hubungan intensitas piksel dalam gambar.

Pemilihan program yang tepat tergantung pada kebutuhan analisis dan sifat gambar yang akan diolah. Semua program tersebut memiliki kegunaan yang berbeda dan dapat memberikan informasi yang berbeda pula tergantung pada jenis data yang diproses.

1. Corner detector

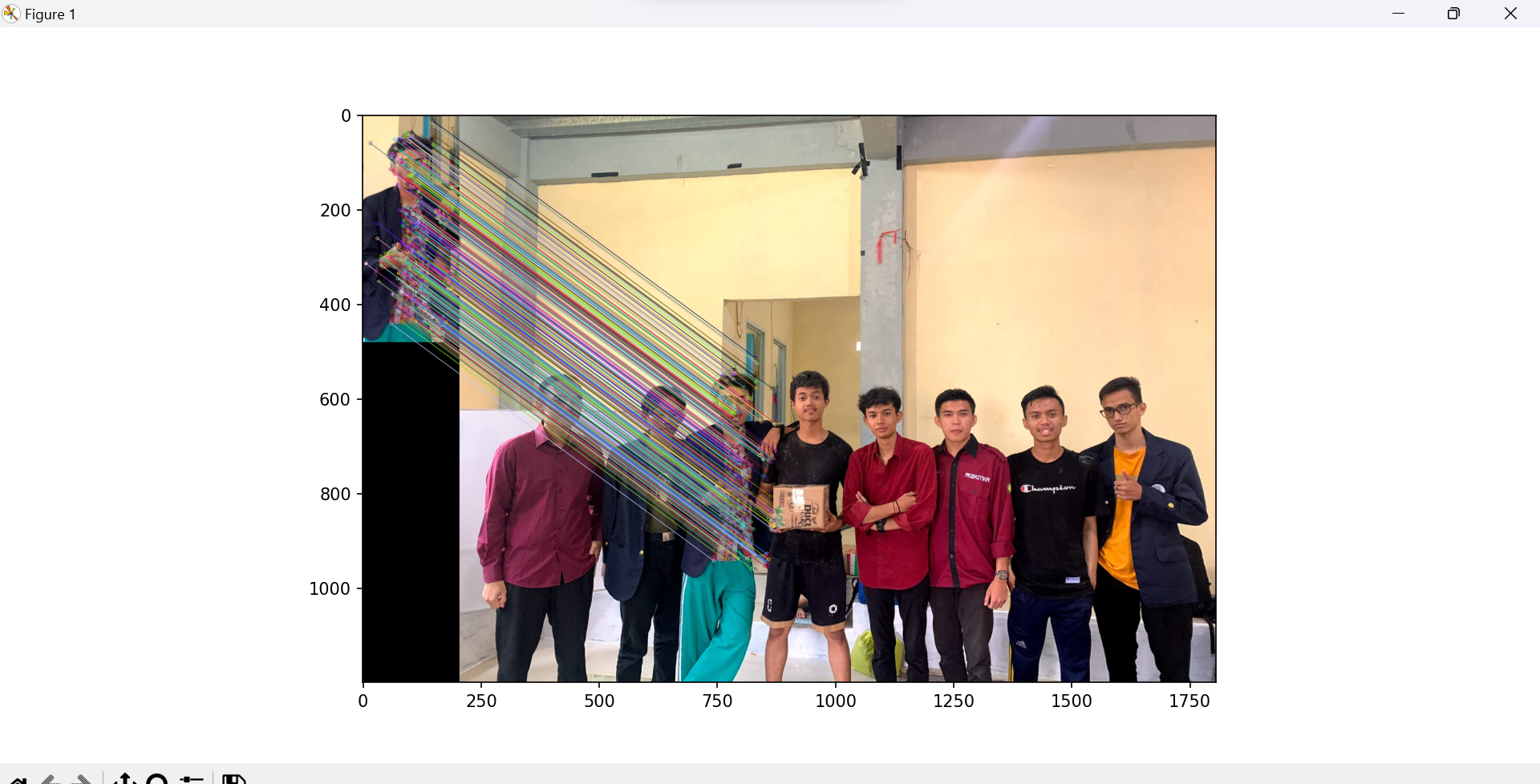


Program di atas merupakan implementasi dari metode deteksi sudut (corner detection) menggunakan algoritma Shi-Tomasi Good Features to Track (GFTT) pada gambar "ei1.jpg". Fungsi cv2.goodFeaturesToTrack() dari pustaka OpenCV digunakan untuk menerapkan metode Shi-Tomasi GFTT guna mendeteksi sudut pada citra grayscale. Argumen pertama (gray) adalah citra masukan, argumen kedua (1000) adalah jumlah maksimum sudut yang ingin dideteksi, argumen ketiga (0.01) adalah kualitas sudut yang diinginkan, dan argumen keempat (10) adalah jarak minimum antara sudut yang dideteksi. Fungsi NumPy np.int0() digunakan untuk mengubah tipe data array menjadi tipe data integer.

Dalam perulangan for i in corners, dilakukan iterasi untuk setiap sudut yang terdeteksi. Melalui penggunaan i.ravel(), dimensi array sudut dibuka dan disimpan dalam variabel x dan y. Fungsi cv2.circle() dari pustaka OpenCV digunakan untuk menggambar lingkaran pada citra. Lingkaran tersebut memiliki pusat (x, y), radius 3 pixel, warna putih (255), dan ketebalan -1 (mengisi lingkaran).

Dengan demikian, program ini membaca gambar "rektorat.jpg", mendeteksi sudut menggunakan metode Shi-Tomasi GFTT, menampilkan jumlah sudut yang terdeteksi (sebanyak 970 sudut), dan menampilkan citra tersebut dengan sudut yang ditandai menggunakan lingkaran.

1. Feature detection dan matching

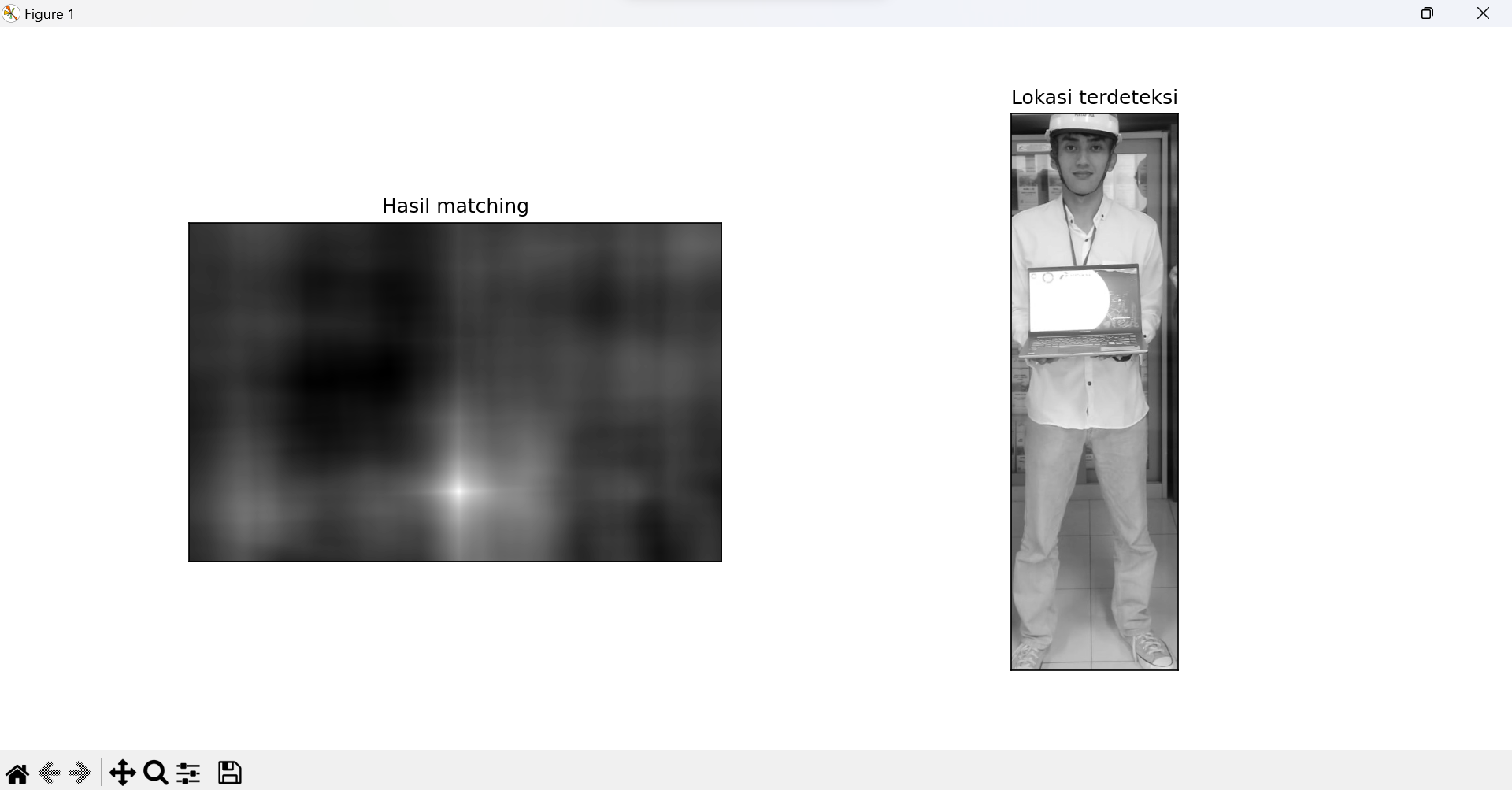


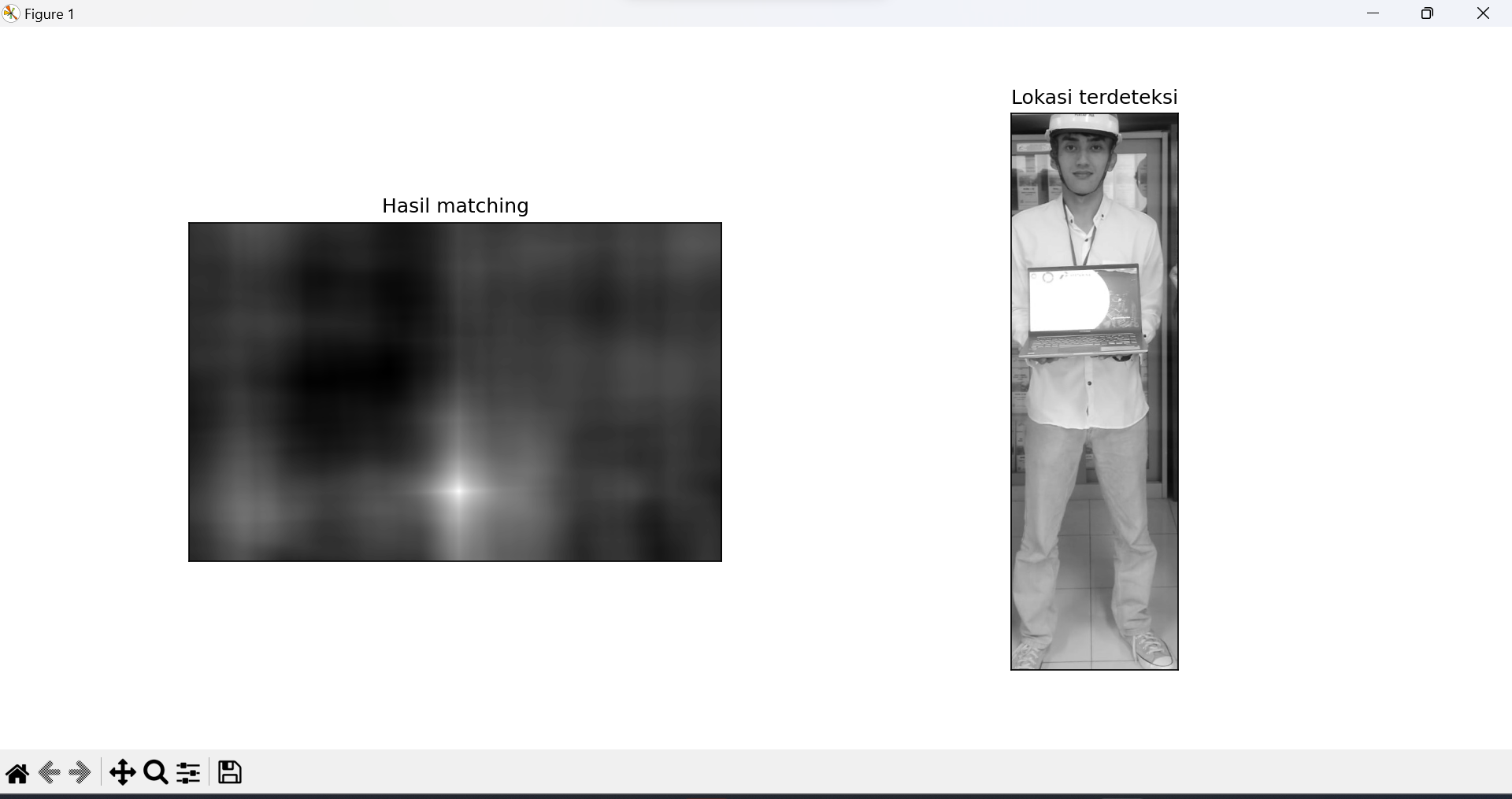
Program di atas merupakan contoh skrip untuk melakukan deteksi fitur dan pencocokan antara dua gambar menggunakan algoritma SIFT (Scale-Invariant Feature Transform) dan Brute-Force Matcher. Dalam OpenCV, kita dapat membuat objek detektor SIFT menggunakan metode cv2.SIFT\_create(). Objek ini digunakan untuk mendeteksi titik kunci (key points) dan menghitung deskriptor menggunakan metode SIFT. Deskriptor merupakan representasi numerik yang menggambarkan fitur dari setiap titik kunci dalam citra. Selain itu, kita dapat menggunakan fungsi sift.detectAndCompute() untuk mendeteksi titik kunci dan menghitung deskriptor. Argumen pertama adalah citra grayscale yang digunakan, sementara argumen kedua adalah maska opsional yang dapat digunakan untuk membatasi area di mana deteksi titik kunci dilakukan. Dengan menggunakan objek detektor SIFT dan fungsi detectAndCompute(), kita dapat melakukan analisis dan pemrosesan citra menggunakan metode SIFT dari OpenCV.

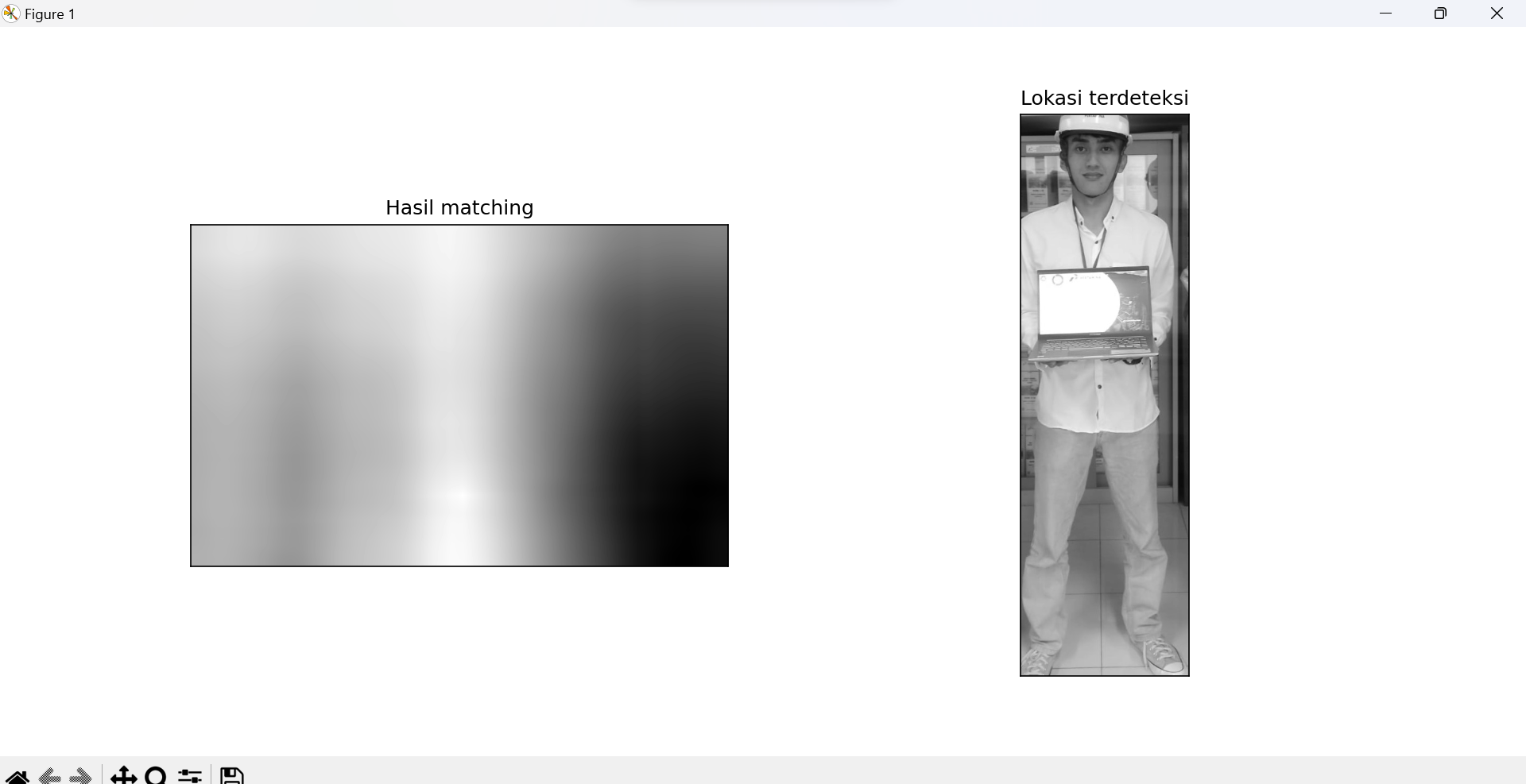
Dalam proses pencocokan, kita dapat menggunakan objek matcher Brute-Force dengan menggunakan metode cv2.BFMatcher() dari OpenCV. Objek ini digunakan untuk mencocokkan fitur dengan menggunakan metode k-NN (k-Nearest Neighbors). Dalam fungsi bf.knnMatch(), kita memberikan dua argumen. Argumen pertama adalah deskriptor fitur dari kedua gambar, yaitu des1 dan des2 dalam contoh ini. Sedangkan argumen kedua, yaitu k=2, menentukan jumlah tetangga terdekat yang akan dicocokkan. Dengan menggunakan objek matcher Brute-Force dan fungsi knnMatch(), kita dapat melakukan pencocokan fitur antara dua gambar menggunakan metode k-NN.

Untuk menggambar hasil pencocokan pada gambar baru (img3), kita dapat menggunakan fungsi cv2.drawMatchesKnn() untuk menghasilkan gambar yang menunjukkan hasil pencocokan antara kedua gambar dengan titik kunci yang cocok. Dengan demikian, hasilnya dapat dilihat pada gambar pertama yang menampilkan banyak garis-garis berwarna sebagai indikator pencocokan antara kedua gambar. Ketika gambar wajah diganti, program tetap menjalankan deteksi dan pencocokan, namun tidak banyak kesamaan atau pencocokan yang tepat, sehingga garis-garis berwarna tidak sebanyak pada gambar pertama.

1. Template Matching









Program di atas merupakan sebuah implementasi dari template matching untuk mencari kemiripan antara sebuah citra template dan citra input. Citra input dan citra template di baca menggunakan fungsi cv2.imread. Citra template ini akan dicocokkan dengan citra input menggunakan metode template matching.

Program menggunakan enam metode yang berbeda untuk perbandingan, yaitu cv2.TM\_CCOEFF, cv2.TM\_CCOEFF\_NORMED, cv2.TM\_CCORR, cv2.TM\_CCORR\_NORMED, cv2.TM\_SQDIFF, dan cv2.TM\_SQDIFF\_NORMED. Metode-metode ini diterapkan secara berurutan pada citra input. Selanjutnya, program mengatur ukuran tampilan hasil plotting menggunakan plt.rcParams["figure.figsize"] agar lebih besar.

Pada setiap iterasi, citra input disalin untuk diolah dengan metode template matching yang sedang digunakan. Hasil template matching disimpan dalam variabel res. Program juga mencari ukuran citra template untuk menggambar kotak persegi pada lokasi yang ditemukan. Selanjutnya, dilakukan pengecekan untuk metode cv2.TM\_SQDIFF dan cv2.TM\_SQDIFF\_NORMED. Jika metode yang sedang digunakan adalah salah satu dari metode tersebut, maka nilai top\_left diambil dari min\_loc, karena semakin kecil nilai hasil, semakin baik kemiripannya. Jika metode yang digunakan adalah selain itu, maka nilai top\_left diambil dari max\_loc, karena semakin besar nilai hasil, semakin baik kemiripannya.

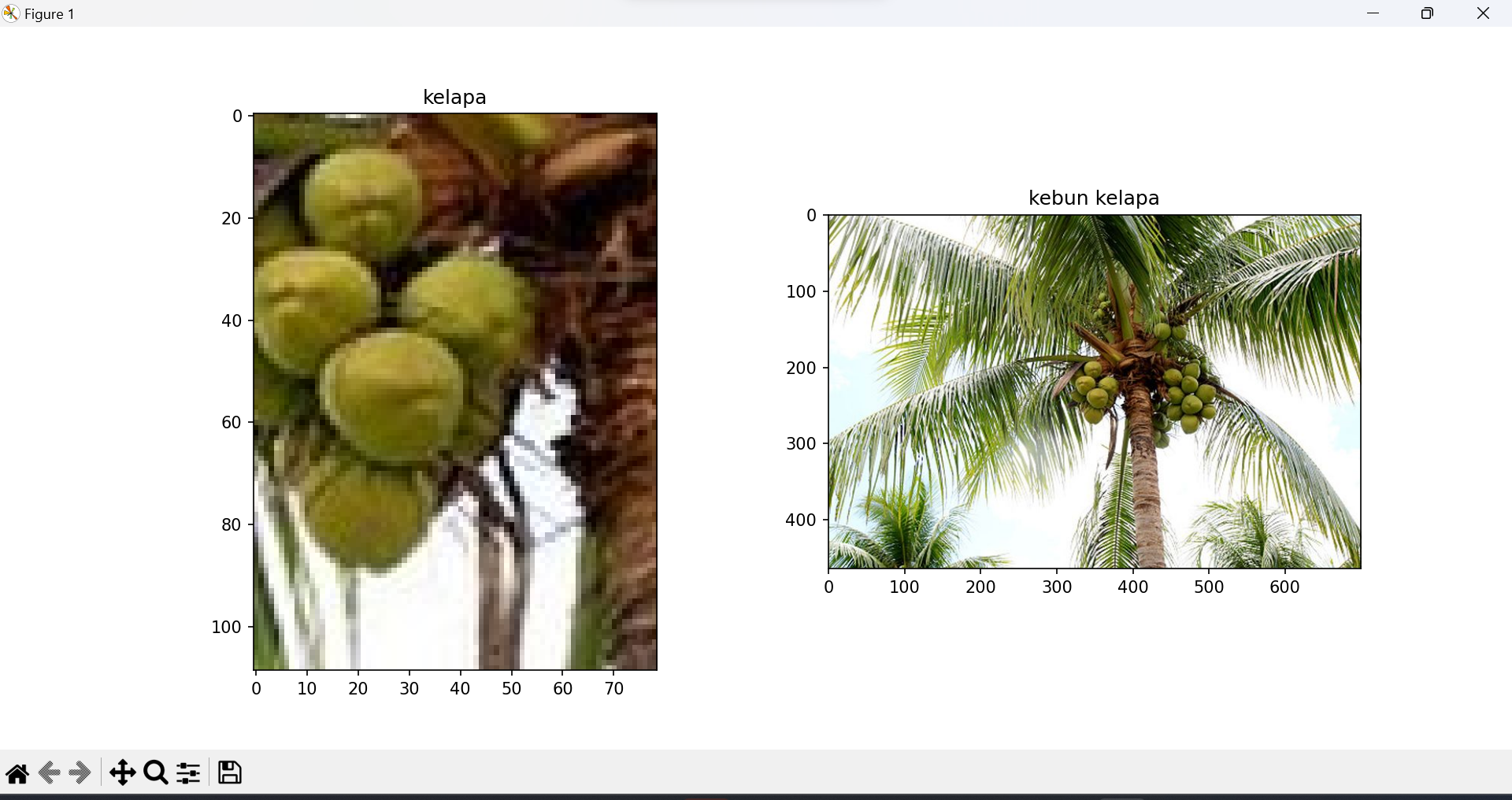
Selanjutnya, program membuat persegi pada lokasi yang ditemukan menggunakan fungsi cv2.rectangle dengan ketebalan garis 2. Program mencetak hasil metode-template matching yang digunakan. Terakhir, program menampilkan citra hasil template matching dan citra dengan persegi yang menunjukkan lokasi yang terdeteksi menggunakan matplotlib.

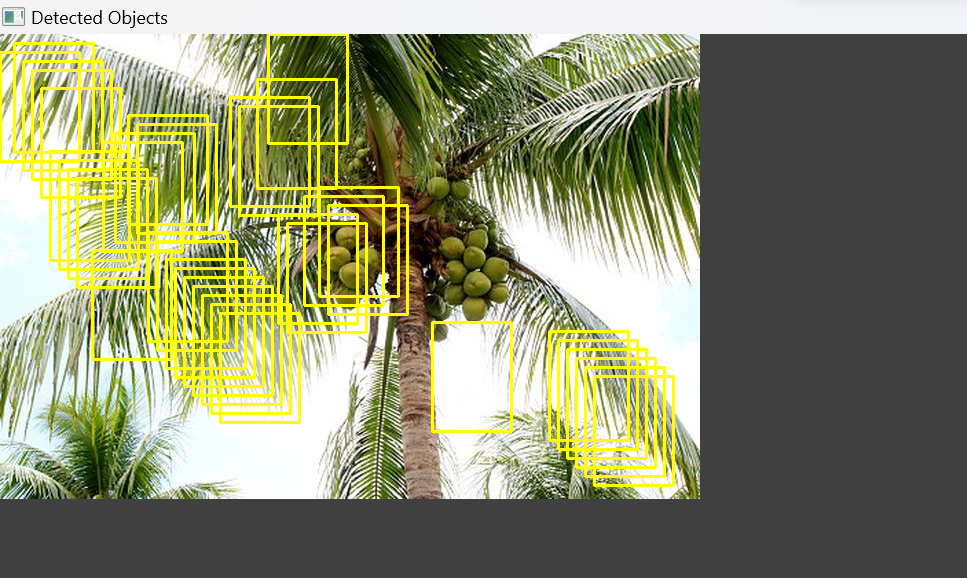
1. Template matching 2



Program di atas menggunakan metode yang sama dengan program sebelumnya, namun mengganti gambar yang berbeda. Hasil yang didapatkan, program tidak mendeteksi wajah dengan baik dan hasilnya tidak sesuai. Sehingga dapat dikatakan program tidak mengenali wajah jika menggunakan gambar yang berbeda.

1. Deteksi kelapa





Program di atas terdiri dari dua bagian utama. Bagian pertama adalah menampilkan dua gambar menggunakan library matplotlib, sementara bagian kedua melakukan deteksi objek menggunakan metode template matching pada gambar.

Bagian pertama program menggunakan library matplotlib untuk menampilkan dua gambar, yaitu "kelapa.jpg" dan "kelapa1.jpg". Gambar-gambar tersebut diimpor menggunakan cv2.imread dan dikonversi ke format warna yang kompatibel dengan matplotlib menggunakan cv2.cvtColor. Gambar pertama ditampilkan pada subplot 1 dengan judul "kelapa", sedangkan gambar kedua ditampilkan pada subplot 2 dengan judul "kebun kelapa".

Bagian kedua program dimulai dengan membaca gambar "kelompokkelapa.jpg" secara keseluruhan dan mengubahnya menjadi skala abu-abu menggunakan cv2.cvtColor. Selanjutnya, gambar template "kelapa.jpg" juga dibaca dan dikonversi ke skala abu-abu.

Setelah itu, ukuran template diambil dan akan digunakan untuk menggambar kotak persegi pada lokasi yang ditemukan. Program melakukan template matching dengan metode cv2.TM\_CCOEFF\_NORMED menggunakan cv2.matchTemplate. Nilai threshold untuk deteksi kemiripan ditentukan sebesar 0.15. Kemudian, program mencari lokasi-lokasi yang memiliki nilai kemiripan di atas atau sama dengan threshold menggunakan np.where.

Program melakukan iterasi pada setiap lokasi yang ditemukan dan melakukan beberapa tindakan. Pertama, program menggambar persegi kuning pada lokasi tersebut menggunakan cv2.rectangle. Kemudian, program menyimpan koordinat x dan y dalam list lspoint dan lspoint2 dengan jarak (+/- 9) dari titik tengah persegi. Tujuan dari ini adalah untuk menghindari deteksi yang berdekatan yang mungkin merupakan duplikat. Jumlah deteksi yang ditemukan juga dihitung menggunakan variabel count.

Terakhir, program menampilkan gambar dengan objek yang terdeteksi menggunakan cv2.imshow. Program akan menunggu hingga tombol keyboard ditekan sebelum menutup jendela tampilan menggunakan cv2.waitKey dan cv2.destroyAllWindows.

Program ini menampilkan dua gambar secara berdampingan menggunakan matplotlib dan melakukan deteksi objek menggunakan metode template matching. Program menggunakan gambar template "kelapa.jpg" untuk mencari objek yang serupa dalam gambar "kelompokkelapa.jpg". Deteksi objek dilakukan dengan menggambar kotak persegi pada lokasi yang memiliki kemiripan di atas atau sama dengan threshold yang ditentukan. Jumlah objek yang ditemukan juga dicetak, dan hasil deteksi objek sebanyak 39.akan tetpi entah mengapa daun di definisikan seperti kelapa entah karena warna nya sama hijau atau kenpaa