Program Studi Teknik Informatika
Institut Teknologi Sumatera

Nama:
Eric Daniel Hutabarat (121140204)
Farhan Rizky Gunawan (121140200)

Tugas Ke: **Final Term Report**

Mata Kuliah: **Sistem/Teknologi Multimedia (IF4025)**

Tanggal: 23 Desember 2024

Toonify

Toonify adalah filter berbasis kamera yang mengubah tampilan wajah pengguna menjadi gaya kartun yang unik. Dengan memanfaatkan OpenCV dan MediaPipe, filter ini mendeteksi wajah dan menciptakan efek kartun halus dengan garis tepi yang tajam dan warna yang lembut. Mudah digunakan, cepat, dan memberikan pengalaman menyenangkan layaknya filter media sosial populer.

Penjelasan Kode Program

0.1 Penjelasan Kode dalam `app.py`

0.1.1 `import os` dan `import sys`

Modul `os` digunakan untuk manipulasi jalur file dan direktori, sementara `sys` digunakan untuk berinteraksi dengan interpreter Python.

0.1.2 Menambahkan direktori `src` ke Python path

```
import os
import sys

sys.path.append(os.path.abspath(os.path.join(os.path.dirname(__file__), "src")))
```

Kode 1: Menambahkan direktori src ke Python path

Tujuan: Menambahkan folder `src` (di mana file sumber atau modul terkait disimpan) ke `sys.path`. Ini memungkinkan Python mengenali modul di folder `src` agar dapat diimpor ke dalam program.
Peran: Memastikan aplikasi dapat mengakses modul yang berada dalam struktur direktori proyek.

0.1.3 Mengimpor `CameraApp` dari modul `main`

```
from main import CameraApp
```

Kode 2: Mengimpor CameraApp dari modul main

Tujuan: Mengimpor kelas atau fungsi utama (`CameraApp`) dari file `main.py` di dalam folder `src`.
Peran: `CameraApp` kemungkinan adalah komponen inti yang mengatur antarmuka pengguna dan logika aplikasi.

0.1.4 Memulai aplikasi PyQt

```
if __name__ == "__main__":
    import sys
    from PyQt5.QtWidgets import QApplication

    app = QApplication(sys.argv)
    window = CameraApp()
    window.show()
    sys.exit(app.exec_())
```

Kode 3: Memulai aplikasi PyQt

Tujuan:

- Memastikan blok kode ini hanya dijalankan jika file `app.py` dijalankan secara langsung, bukan diimpor sebagai modul.
- Membuat instance aplikasi berbasis GUI dengan PyQt5.

Peran:

- `QApplication`: Komponen utama PyQt yang memulai loop event GUI.
- `CameraApp`: Mewakili jendela utama aplikasi.
- `sys.exit(app.exec_())`: Memastikan aplikasi berjalan hingga pengguna keluar, dan memastikan keluar dengan kode status yang sesuai.

0.2 Penjelasan Kode dalam `main.py`

Kode `main.py` memiliki satu tugas utama: mengimpor kelas `CameraApp` dari modul `modules.camera_app`. Berikut adalah penjelasan lebih rinci:

0.2.1 Mengimpor `CameraApp`

```
from modules.camera_app import CameraApp
```

Kode 4: Mengimpor CameraApp dari modul modules.camera$_{a}pp$

Tujuan: Mengimpor kelas `CameraApp` dari file Python bernama `camera_app.py`, yang berada di dalam folder `modules`.
Peran: `CameraApp` adalah inti dari aplikasi ini. File `main.py` dirancang untuk menyediakan akses ke kelas ini tanpa harus merujuk ke lokasi modul secara eksplisit berulang kali.

0.3 Penjelasan Kode dalam `cartoon_filter.py`

File `cartoon_filter.py` digunakan untuk menerapkan efek kartun pada frame video dengan penghilangan latar belakang.

0.3.1 Import Modul

- **`cv2`:** Bagian dari pustaka OpenCV, digunakan untuk manipulasi gambar dan video seperti pengolahan frame, konversi warna, filter, dll.
- **`numpy (np)`:** Pustaka untuk manipulasi array multidimensi, yang digunakan untuk representasi gambar dalam OpenCV.
- **`cvzone.SelfiSegmentationModule`:** Mengimpor `SelfiSegmentation`, modul dari pustaka CVZone untuk segmentasi gambar. Ini digunakan untuk menghapus latar belakang dari gambar atau video secara efisien.

0.3.2 Inisialisasi Objek Segmentasi

```
segmentor = SelfiSegmentation()
```

Kode 5: Inisialisasi Objek Segmentasi

Tujuan: Membuat objek segmentasi untuk menghapus latar belakang menggunakan `SelfiSegmentation`.
Fungsi Utama: `removeBG` (digunakan nanti dalam kode).

0.3.3 Fungsi `cartoonize_frame`

Fungsi ini menerima sebuah frame video (gambar dalam format array NumPy) dan menghasilkan gambar yang telah diberi efek kartun.

Langkah-Langkah Pemrosesan

```
white = (255, 255, 255)  # Warna putih untuk latar belakang
frame_no_bg = segmentor.removeBG(frame, white)
```

Kode 6: Menghapus Latar Belakang

Apa yang Terjadi:

- Latar belakang dihapus menggunakan metode `removeBG`. [1]
- Latar belakang diganti dengan warna putih.

```
gray = cv2.cvtColor(frame_no_bg, cv2.COLOR_BGR2GRAY)
```

Kode 7: Konversi ke Grayscale

Apa yang Terjadi: Frame dikonversi menjadi gambar skala abu-abu untuk pengolahan tepi.

```
gray = cv2.medianBlur(gray, 1)
```

Kode 8: Mengurangi Noise

Apa yang Terjadi: Menggunakan filter median untuk mengurangi noise pada gambar abu-abu.

```
edges1 = cv2.adaptiveThreshold(gray, 255, cv2.ADAPTIVE_THRESH_MEAN_C, cv2.THRESH_BINARY, 9, 9)
```

Kode 9: Deteksi Tepi

Apa yang Terjadi: `adaptiveThreshold` mendeteksi tepi dengan adaptif, menghasilkan gambar biner (hitam-putih) yang merepresentasikan tepi. [2]

```
color = cv2.bilateralFilter(frame_no_bg, 9, 250, 250)
```

Kode 10: Efek Halus dengan Bilateral Filter

Apa yang Terjadi: Filter bilateral diterapkan untuk menghasilkan gambar dengan efek seperti lukisan, mempertahankan tepi sambil menghaluskan warna.

```
cartoon1 = cv2.bitwise_and(color, color, mask=edges1)
```

Kode 11: Menggabungkan Warna dan Tepi

Apa yang Terjadi: Kombinasi gambar berwarna halus dengan tepi yang terdeteksi, menciptakan efek kartun.

```
return cartoon1
```

Kode 12: Mengembalikan Gambar Kartun

0.4 Penjelasan Kode dalam `camera_app.py`

File `camera_app.py` mendefinisikan kelas `CameraApp`, yang merupakan komponen utama dari antarmuka pengguna aplikasi *Toonify*. Kelas ini menggunakan PyQt5 untuk membuat jendela GUI dan OpenCV untuk menangkap serta memproses video secara real-time. Berikut adalah penjelasan detail setiap bagian kode:

0.4.1 Import Modul

- **`cv2`:** Digunakan untuk menangkap frame dari kamera dan memprosesnya.
- **`PyQt5.QtWidgets`:** Digunakan untuk membuat jendela GUI, label untuk menampilkan video, tombol untuk interaksi, dan pengelolaan tata letak.
- **`PyQt5.QtCore.QTimer`:** Digunakan untuk mengatur loop timer agar aplikasi dapat terus-menerus menangkap frame kamera dan memperbaruinya.
- **`PyQt5.QtGui`:** Digunakan untuk mengonversi frame video menjadi format gambar (`QImage`) yang dapat ditampilkan di label GUI.
- **`utils.cartoon_filter`:** Mengimpor fungsi `cartoonize_frame`, yang digunakan untuk menerapkan efek kartun pada frame video.

0.4.2 Kelas `CameraApp`

1. Inisialisasi (`__init__`)

- `self.setWindowTitle("Toonify")`: Menentukan judul jendela aplikasi.
- `self.setGeometry(100, 100, 800, 600)`: Menentukan ukuran dan posisi awal jendela.

Widget dan Layout:

- `QLabel`: Label untuk menampilkan video dari kamera.
- `QPushButton`: Tombol untuk mengaktifkan/menonaktifkan efek kartun.
- `QVBoxLayout`: Tata letak vertikal untuk menyusun widget.

- `QWidget`: Wadah untuk mengatur layout dan menampilkannya di jendela utama.

Timer untuk Mengatur Frame:

```
self.timer = QTimer()
```

Kode 13: Timer untuk Mengatur Frame

Timer ini akan memicu fungsi `update_frame` setiap interval tertentu (30ms untuk kecepatan sekitar 33 FPS).

Kamera:

```
self.cap = cv2.VideoCapture(0)
```

Kode 14: Kamera

Membuka kamera default (indeks 0).

Status Filtering:

```
self.filtering = False
```

Kode 15: Status Filtering

Digunakan untuk menyimpan status apakah efek kartun aktif atau tidak.

2. Fungsi `update_frame` **Tujuan:** Menangkap frame dari kamera, memprosesnya, dan menampilkan hasil di antarmuka.

Langkah-Langkah:

- **Menangkap Frame:**

```
ret, frame = self.cap.read()

```

Kode 16: Menangkap Frame

Jika kamera berhasil membaca frame, variabel `ret` akan bernilai `True` dan `frame` berisi data gambar.

- **Membalik Frame:**

```
frame = cv2.flip(frame, 1)

```

Kode 17: Membalik Frame

Membalik frame secara horizontal untuk memberikan efek mirroring.

- **Menerapkan Efek Kartun (opsional):**

```
if self.filtering:
    frame = cartoonize_frame(frame)

```

Kode 18: Menerapkan Efek Kartun

Jika `self.filtering` diaktifkan, frame akan diproses menggunakan fungsi `cartoonize_frame`.

- **Konversi ke QImage:**

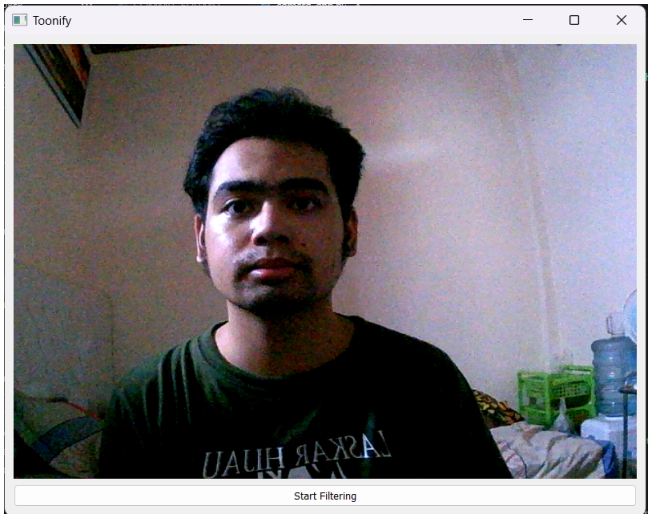
```
qt_image = QImage(frame.data, w, h, bytes_per_line, QImage.Format_BGR888)
self.label.setPixmap(QPixmap.fromImage(qt_image))

```

Kode 19: Konversi ke QImage

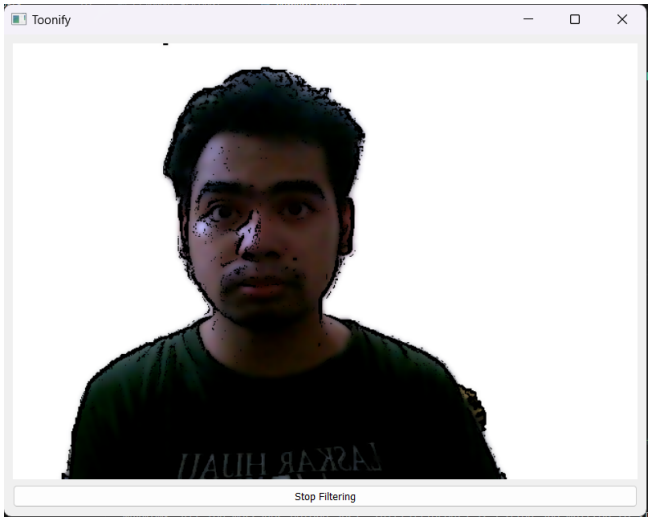
Frame dari OpenCV (format NumPy array) dikonversi menjadi `QImage` agar bisa ditampilkan di label PyQt.

3. **Fungsi `toggle_filtering`** : Mengubah status filtering.
   - Jika `self.filtering` aktif, matikan dan ubah teks tombol menjadi "Start Filtering".
   - Jika tidak aktif, aktifkan dan ubah teks tombol menjadi "Stop Filtering".
4. **Fungsi `closeEvent`** : Membebaskan sumber daya kamera saat aplikasi ditutup.
   - `self.cap.release()`: Menutup koneksi ke kamera.
   - `cv2.destroyAllWindows()`: Menutup semua jendela OpenCV yang mungkin terbuka.

contoh penggunaan program



Gambar 1: sebelum filter



Gambar 2: setelah filter

References

[1] E. Bruno, "Opencv – 10 lines to remove the background in an image ," 2022, microsoft Azure profile image, originally published at elbruno.com on Jun 7, 2022. [Online]. Available: https://dev.to/azure/opencv-10-lines-to-remove-the-background-in-an-image-3m98

[2] Amrutha, "Cartoonify image using opencv and python," 2024, last updated: 16 Oct, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2022/06/cartoonify-image-using-opencv-and-python/

[3] O. ChatGPT, "Chatgpt conversation share," 2024, accessed: 2024-12-23. [Online]. Available: https://chatgpt.com/share/6769275e-c588-8011-9f3a-9f0d151e709b

[3]