به نام خدا

گزارش کار پروژه سوم

مبانی بینائی کامپیوتر

دانشکده مهندسی برق و کامپیوتر

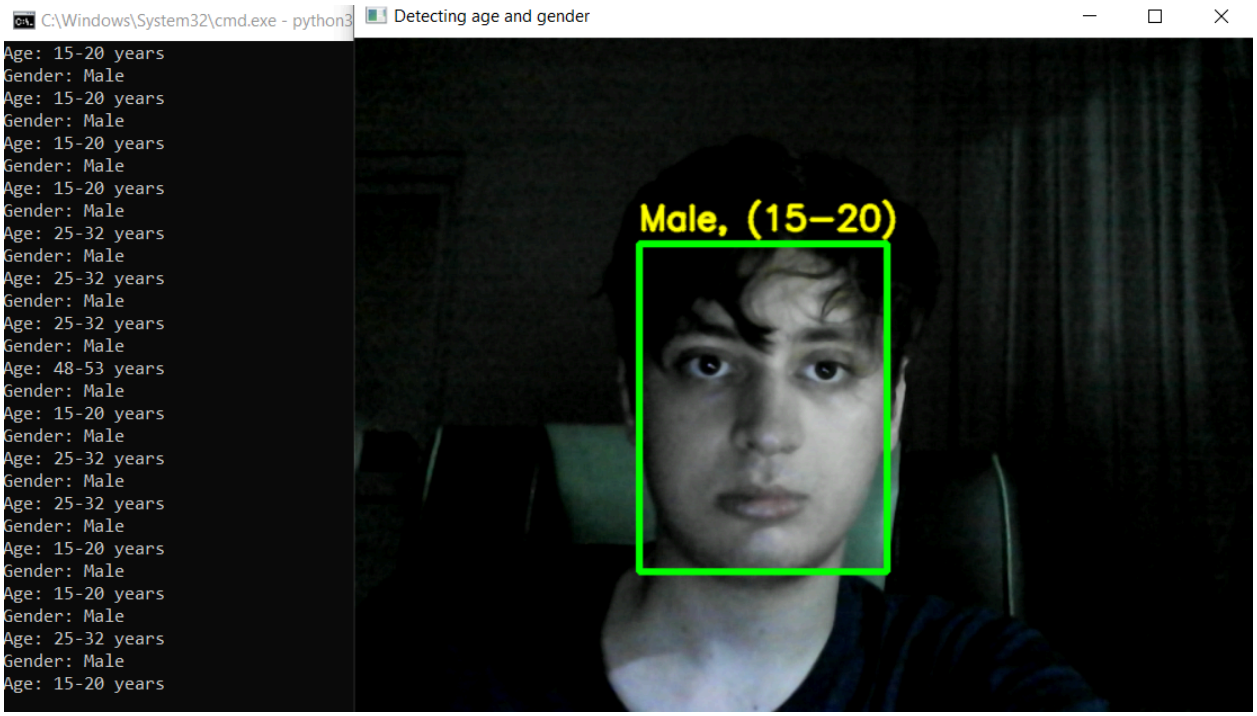
بردیا خلفی ۸۱۰۱۹۹۴۱۴

علی عطاءاللهی ۸۱۰۱۹۹۴۶۱

تعیین سن

کتابخانه‌های استفاده شده

cv2: این کتابخانه برای پردازش تصویر و ویدیو با استفاده از کتابخانه OpenCV استفاده می‌شود.
math: این کتابخانه برای انجام محاسبات ریاضی مورد استفاده قرار می‌گیرد، اگرچه در کد حاضر به طور مستقیم استفاده نشده است.
argparse: این کتابخانه برای تجزیه و تحلیل آرگومان‌های خط فرمان استفاده می‌شود.



توضیح و ترتیب اجرا

کد با استفاده از کتابخانه‌های cv2، math و argparse نوشته شده است. ابتدا با استفاده از argparse آرگومان‌های خط فرمان تجزیه و تحلیل می‌شود تا در صورت ارائه یک تصویر، آدرس آن را دریافت کند؛ در غیر این صورت، از دوربین پیش‌فرض استفاده می‌کند. سپس مدل‌های شبکه عصبی که شامل مدل تشخیص چهره، سن و جنسیت هستند، بارگذاری می‌شوند. حلقه اصلی کد به طور مداوم فریم‌های ورودی از دوربین یا تصویر را دریافت کرده و با استفاده از تابع highlightFace چهره‌ها را شناسایی و آنها را در فریم‌ها با استفاده از جعبه‌های قرمز هایلایت می‌کند. اگر چهره‌ای شناسایی شود، ناحیه چهره استخراج و به مدل‌های شبکه عصبی جنسیت و

سن داده می‌شود تا جنسیت و سن تخمین زده شوند. نتایج تشخیص به همراه فریم و جعبه‌های هایلایت شده نمایش داده می‌شوند. فرآیند تا زمانی که کلید خاصی فشرده نشود یا فریم دیگری موجود نباشد، ادامه می‌یابد.

کارتونی کردن چهره

کتابخانه‌های استفاده شده

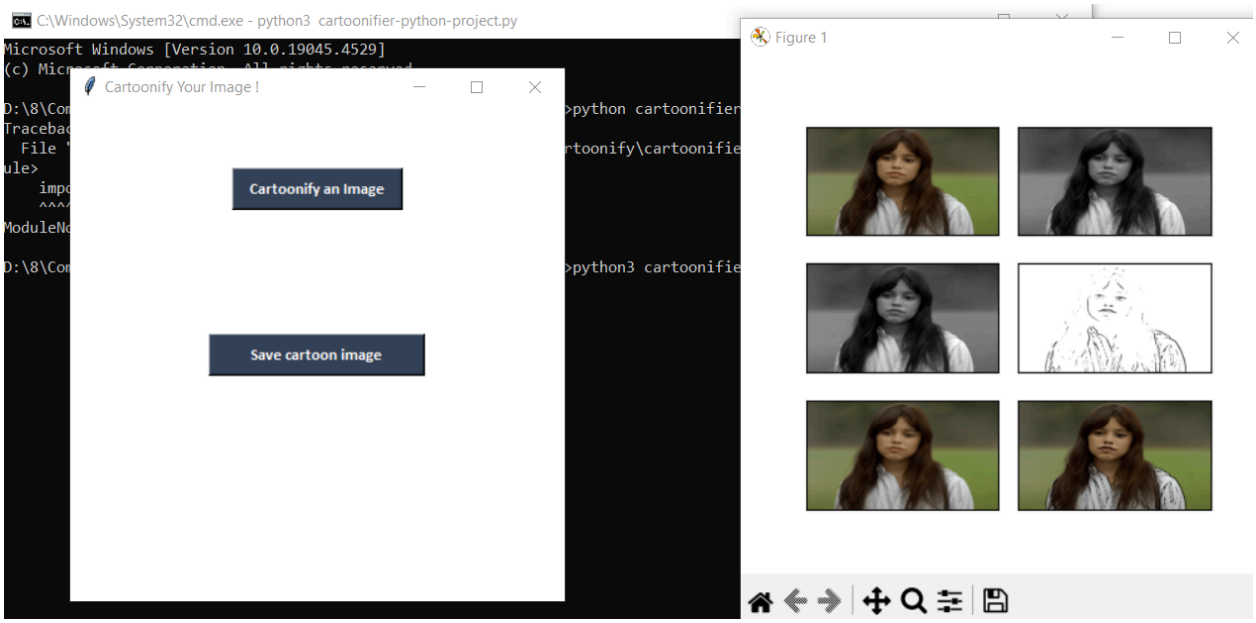
cv2: این کتابخانه برای پردازش تصویر استفاده می‌شود، شامل تبدیل تصویر به مقیاس خاکستری، اعمال فیلترها و ماسک کردن تصاویر.

easygui: برای باز کردن فایل با استفاده از یک رابط کاربری ساده استفاده می‌شود.

imageio: برای خواندن و نوشتن فایل‌های تصویری استفاده می‌شود، اگرچه در کد فعلی استفاده نشده است.

tkinter: برای ساخت رابط کاربری گرافیکی (GUI) که شامل دکمه‌ها و پنجره‌ها می‌شود.

Pillow یا PIL: برای کار با تصاویر، باز کردن و نمایش آنها.



توضیح و ترتیب اجرا

کد با استفاده از tkinter یک رابط کاربری گرافیکی (GUI) ایجاد می‌کند که کاربر می‌تواند از طریق آن یک تصویر را انتخاب کند و سپس آن را به یک تصویر کارتونی تبدیل کند. پس از انتخاب تصویر با استفاده از easygui، تصویر با کمک OpenCV (cv2) پردازش می‌شود. ابتدا تصویر به مقیاس خاکستری تبدیل و سپس با استفاده از فیلترهای مختلف، لبه‌های تصویر استخراج و صاف می‌شوند. با اعمال یک ماسک بر روی تصویر اصلی، تصویر کارتونی ایجاد می‌شود. سپس تصاویر مراحل مختلف پردازش به ترتیب در یک پنجره نمایش داده می‌شوند. در نهایت، تصویر کارتونی شده را می‌توان با استفاده از دکمه‌های موجود در رابط کاربری ذخیره کرد.

تشخیص رنگ

کتابخانه‌های استفاده شده

numpy: برای انجام محاسبات عددی و مدیریت آرایه‌ها استفاده می‌شود.
pandas: برای خواندن و پردازش داده‌های فایل CSV که شامل اطلاعات رنگ‌ها است.
OpenCV یا cv2: برای پردازش تصویر و تعامل با رویدادهای ماوس استفاده می‌شود.

توضیح و ترتیب اجرا

کد با استفاده از OpenCV یک برنامه شناسایی رنگ ایجاد می‌کند که به کاربر امکان می‌دهد رنگ هر نقطه‌ای از یک تصویر را با دوبار کلیک روی آن نقطه شناسایی کند. ابتدا تصویر با نام colorsimage.jpg بارگذاری شده و داده‌های رنگ از فایل colors.csv خوانده می‌شود. این فایل شامل نام رنگ‌ها و مقادیر RGB آنها است. با استفاده از یک تابع، رنگ نزدیک‌ترین مقدار RGB به مقادیر کلیک شده شناسایی می‌شود. با دوبار کلیک بر روی یک نقطه در تصویر، مقادیر RGB آن نقطه به دست آمده و نزدیک‌ترین رنگ از لیست رنگ‌ها پیدا شده و در بالای تصویر به همراه مقادیر RGB نمایش داده می‌شود. اگر مجموع مقادیر RGB نقطه کلیک شده بیشتر از یک مقدار خاص باشد (یعنی رنگ بسیار روشن باشد)، متن رنگ به رنگ مشکی نمایش داده می‌شود. این فرآیند تا زمانی که کاربر کلید Esc را فشار دهد، ادامه دارد و در نهایت پنجره‌ها بسته می‌شوند.

تار کردن چهره

کتابخانه‌های استفاده شده

cv2: این کتابخانه برای پردازش تصویر و استفاده از مدل‌های یادگیری عمیق (Deep Learning) در شناسایی و پردازش تصویر استفاده می‌شود.
numpy: برای انجام محاسبات عددی و مدیریت آرایه‌ها، که برای پردازش نتایج مدل و انجام عملیات هندسی روی تصاویر استفاده می‌شود.
os.path: برای کار با مسیر فایل‌ها و ترکیب مسیرها استفاده می‌شود تا فایل‌های مدل را به درستی بارگذاری کند.

توضیح و ترتیب اجرا

کد ابتدا با استفاده از کتابخانه OpenCV و فایل‌های مدل deploy.prototxt و res10_300x300_ssd_iter_140000_fp16.caffemodel یک مدل تشخیص چهره Caffe را بارگذاری می‌کند. تصویر مورد نظر با نام musk.jpg بارگذاری می‌شود و اندازه‌های آن به دست می‌آید. تصویر برای پردازش در شبکه عصبی پیش‌پردازش شده و به شبکه عصبی داده می‌شود تا چهره‌ها شناسایی شوند. سپس، اگر اطمینان شناسایی هر چهره بیشتر از ۴۰ درصد باشد، مختصات جعبه اطراف چهره به دست آمده و چهره برش داده می‌شود. روی این ناحیه از تصویر، فیلتر گوسی اعمال می‌شود تا چهره تار شود. در نهایت، تصویر اصلی با ناحیه‌های چهره‌ی تار شده جایگزین شده و به عنوان یک فایل جدید با نام blurred-face1.jpg ذخیره می‌شود. این فرآیند به کاربر اجازه می‌دهد تا چهره‌ها را در تصویر شناسایی کرده و آنها را به صورت خودکار تار کند.

تشخیص چهره

کتابخانه‌های استفاده شده

cv2: این کتابخانه برای پردازش تصویر و انجام عملیات تشخیص چهره استفاده می‌شود، همچنین برای ایجاد و آموزش یک تشخیص‌دهنده چهره به کمک الگوریتم‌های موجود در OpenCV به کار می‌رود.
numpy: برای انجام محاسبات عددی و مدیریت آرایه‌ها، که برای تبدیل تصاویر به آرایه‌های عددی و پردازش داده‌های تصویری استفاده می‌شود.
PIL: برای کار با تصاویر و تبدیل آنها به مقیاس خاکستری استفاده می‌شود.
os: برای کار با مسیر فایل‌ها و مدیریت فایل‌های سیستم استفاده می‌شود.

توضیح و ترتیب اجرا

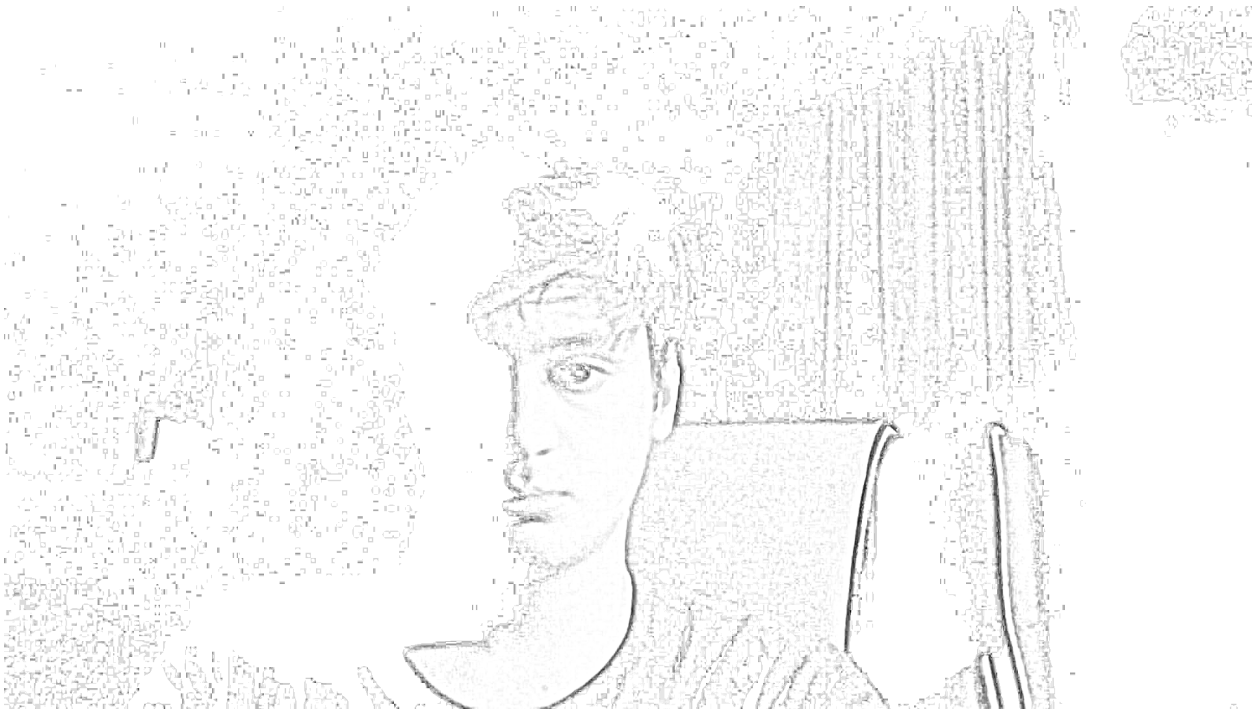
کد برای آموزش یک مدل تشخیص چهره با استفاده از الگوریتم LBPH (Local Binary Patterns Histograms) از کتابخانه OpenCV نوشته شده است. ابتدا، فایل‌های تصویری چهره‌ها از مسیر ./images/ بارگذاری شده و با استفاده از فایل Haar cascade به نام haarcascade_frontalface_default.xml، چهره‌ها شناسایی می‌شوند. هر تصویر به مقیاس خاکستری تبدیل شده و شناسه کاربر از نام فایل استخراج می‌شود. سپس چهره‌ها و شناسه‌های مربوط به آنها در دو لیست مجزا ذخیره می‌شوند. با استفاده از این داده‌ها، مدل تشخیص چهره با استفاده از cv2.face.LBPHFaceRecognizer_create() آموزش داده شده و در نهایت مدل آموزش

دیده به نام trainer.yml ذخیره می‌شود. برنامه با نمایش تعداد چهره‌های آموزش دیده به پایان می‌رسد و فایل مدل آموزش دیده برای استفاده‌های بعدی ذخیره می‌شود.

تصویر کردن به خط‌های سازنده

کتابخانه‌های استفاده شده

cv2: این کتابخانه برای پردازش تصویر استفاده می‌شود، از جمله خواندن و نوشتن تصاویر، تبدیل تصاویر به مقیاس خاکستری، معکوس کردن تصویر، اعمال فیلتر گوسی و ترکیب تصاویر.



توضیح و ترتیب اجرا

کد با استفاده از کتابخانه OpenCV تصویری را به یک اسکچ (نقاشی خطی) تبدیل می‌کند. ابتدا تصویر با نام input2-300x295.jpg خوانده شده و به مقیاس خاکستری تبدیل می‌شود. سپس تصویر خاکستری معکوس می‌شود تا تضاد رنگ‌ها ایجاد شود. با اعمال فیلتر گوسی، تصویر به طور نرم و محو تبدیل شده و مجدداً معکوس می‌شود. در مرحله بعد، تصویر خاکستری اصلی با تصویر معکوس و محو شده ترکیب می‌شود تا نتیجه نهایی که

یک اسکچ است، به دست آید. این اسکچ در نهایت به نام sketch.png در پوشه فعلی ذخیره می‌شود. این فرآیند به کاربر اجازه می‌دهد تا به سادگی تصاویر را به یک طرح خطی زیبا تبدیل کند.

شناسایی بارکد

کتابخانه‌های استفاده شده

cv2: این کتابخانه برای پردازش تصویر و ویدئو، تشخیص و رمزگشایی کدهای QR، رسم شکل‌های هندسی روی تصاویر و نمایش تصویر و ویدئو استفاده می‌شود.
numpy: برای انجام محاسبات عددی و مدیریت آرایه‌ها که به عنوان ورودی و خروجی برای عملیات‌های پردازشی تصویر در کتابخانه OpenCV به کار می‌رود.

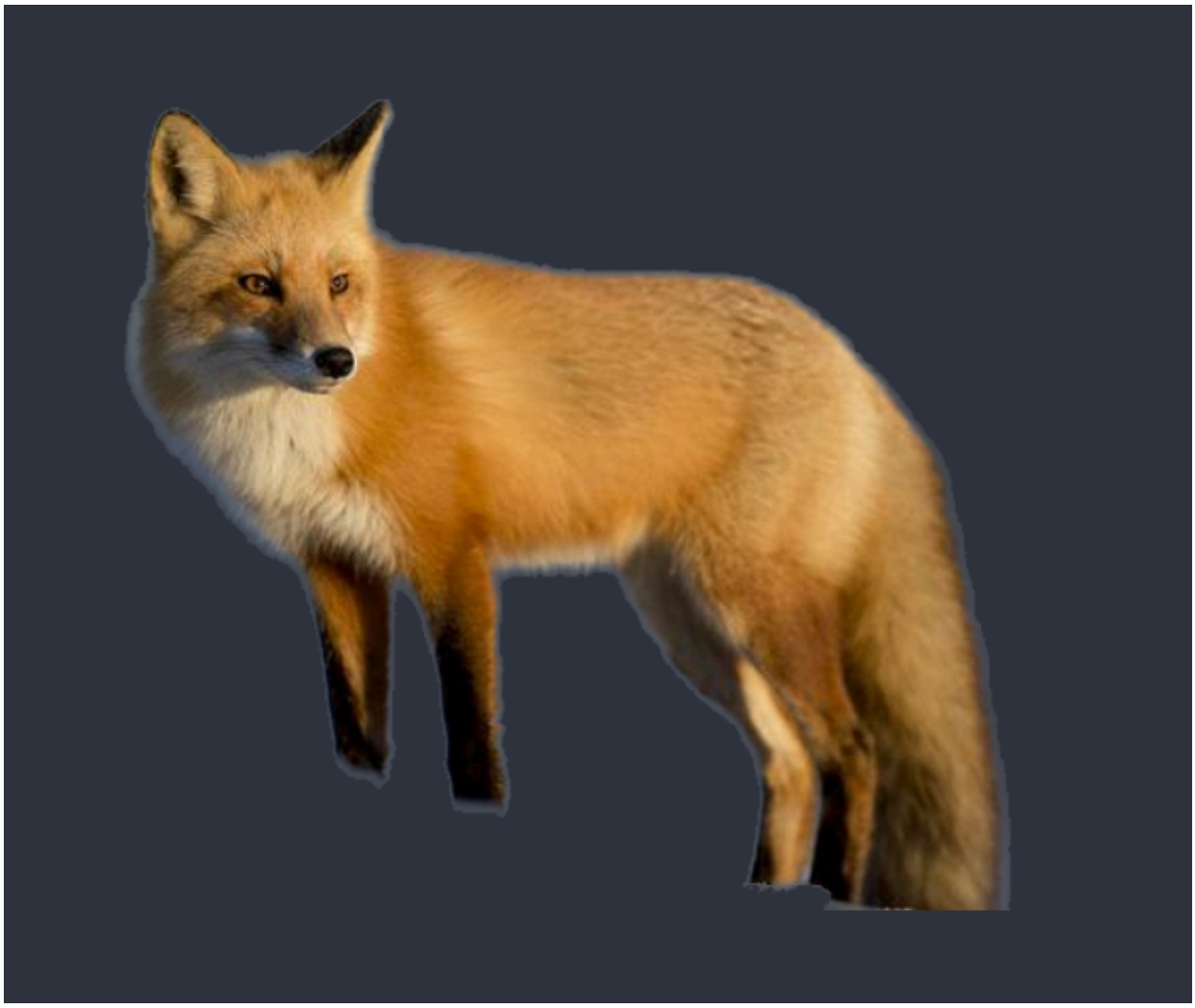
توضیح و ترتیب اجرا

با استفاده از کتابخانه OpenCV و از طریق ماژول QRCodeDetector، توانایی تشخیص و رمزگشایی کدهای QR را در یک تصویر و همچنین در جریان ویدئویی از دوربین وب‌کم پیاده‌سازی می‌کند. در ابتدا، کد نسخه‌ی OpenCV را چاپ می‌کند و سپس تصویر qrcode.png را خوانده و با استفاده از تابع detectAndDecodeMulti، اطلاعات رمزگشایی شده و نقاط مختصات کدهای QR موجود در تصویر را استخراج می‌کند. این اطلاعات به صورت متن در کنسول چاپ می‌شوند. سپس، خطوط چند ضلعی اطراف کدهای QR رسم شده و تصویر نتیجه در مسیر data/dst/qrcode_opencv.jpg ذخیره می‌شود. در بخش دوم کد، با استفاده از دوربین وب‌کم، به صورت پیوسته فریم‌های ویدئویی دریافت می‌شوند و در هر فریم، کدهای QR شناسایی و اطلاعات رمزگشایی شده آنها در کنسول چاپ می‌شود. همچنین، برای هر کد QR شناسایی شده، خطوط چند ضلعی به رنگ سبز یا قرمز بر اساس موفقیت یا عدم موفقیت در رمزگشایی، رسم می‌شود. نمایش ویدئو در پنجره‌ای با نام OpenCV QR Code صورت می‌گیرد و با فشردن کلید q، برنامه متوقف و پنجره بسته می‌شود.

حذف پس‌زمینه

کتابخانه‌های استفاده شده

torch: برای انجام محاسبات علمی و یادگیری عمیق استفاده می‌شود و پشتیبانی از پردازش موازی روی GPU را فراهم می‌کند.
IPython.display: برای نمایش تصاویر و پاک کردن خروجی‌ها در نوت‌بوک‌های Jupyter استفاده می‌شود.
google.colab.files: برای بارگذاری فایل‌ها از سیستم کاربر به محیط Colab استفاده می‌شود.
carvekit.web.schemas.config.MLConfig: برای تنظیمات مربوط به شبکه‌های تفکیک‌کننده و فرآیندهای پیش‌پردازش و پس‌پردازش استفاده می‌شود.
carvekit.web.utils.init_utils.init_interface: برای راه‌اندازی و پیکربندی رابط کاربری جهت پردازش تصاویر استفاده می‌شود.



توضیح و ترتیب اجرا

در یک محیط نوت‌بوکی مانند Google Colab اجرا می‌شود و هدف آن بارگذاری تصاویر از سیستم کاربر و اعمال تکنیک‌های مختلف تفکیک (segmentation) و پردازش تصویر بر روی آنهاست. ابتدا کتابخانه‌های مورد نیاز وارد می‌شوند که شامل torch برای انجام محاسبات یادگیری عمیق، IPython.display برای نمایش تصاویر، google.colab.files برای بارگذاری فایل‌ها، و کتابخانه carvekit برای پیکربندی و استفاده از مدل‌های تفکیک است. سپس، پارامترهای مختلف مانند نوع شبکه تفکیک، روش‌های پیش‌پردازش و پس‌پردازش، و اندازه ماسک تفکیک مشخص می‌شوند. با استفاده از این پارامترها، یک شیء پیکربندی MLConfig ایجاد شده و رابط کاربری با استفاده از تابع init_interface راه‌اندازی می‌شود. تصاویر بارگذاری شده توسط کاربر به این رابط ارسال شده و

نتایج پردازش شده به صورت تصویری در محیط نوت‌بوک نمایش داده می‌شوند. همچنین، این امکان وجود دارد که تصاویر با اندازه کامل یا در اندازه کوچک‌تری برای نمایش سریع‌تر نمایش داده شوند.

حذف سایه

کتابخانه‌های استفاده شده

cv2: برای پردازش تصاویر و استفاده از توابع مربوط به OpenCV استفاده می‌شود.
numpy (np): برای انجام عملیات عددی و کار با آرایه‌ها در پایتون استفاده می‌شود.
skimage.measure: برای محاسبه ویژگی‌های مختلف از تصاویر، مانند محاسبه ماسک‌های تفکیک (segmentation masks) استفاده می‌شود.
matplotlib.pyplot as plt: برای نمایش تصاویر و نمودارها استفاده می‌شود.
typing: برای اعلام و استفاده از نوع داده‌ها و ورودی‌های توابع استفاده می‌شود.

توضیح و ترتیب اجرا

برای حذف سایه‌های موجود در تصاویر با استفاده از تکنیک‌های پیچیده‌ای از جمله تفکیک، فیلترینگ میانه، و تصحیح رنگ طراحی شده است. ابتدا، کتابخانه‌های مورد نیاز وارد می‌شوند. سپس توابعی برای فیلترینگ میانه،

تفکیک و تصحیح رنگ برای مناطق مختلف سایه‌ها تعریف می‌شوند. فرآیند شروع با تعریف تابع median_filter برای اعمال فیلترینگ میانه بر روی نقاط مختلف تصویر است. سپس تابع edge_median_filter برای اعمال فیلترینگ میانه بر روی نقاط لبه‌ها استفاده می‌شود. تابع display_region برای نمایش تصاویر مختلف از جمله منطقه سایه‌دهی شده و نتایج نهایی استفاده می‌شود. توابع دیگری مانند correct_region_lab و correct_region_bgr برای تصحیح رنگ مناطق سایه‌دهی شده بر اساس میانگین رنگ‌های لب و BGR تعریف شده‌اند. نهایتاً، توابع calculate_mask و process_regions برای تولید ماسک تفکیک و پردازش هر منطقه سایه‌دهی شده به ترتیب استفاده می‌شوند. تابع process_image_file نیز تصویر را می‌خواند، سایه‌ها را حذف می‌کند و نتایج را نمایش می‌دهد. در نهایت، با استفاده از تصویر نمونه "shadow1.jpg"، تمامی این فرآیندها انجام شده و نتایج نهایی نمایش داده و در صورت نیاز ذخیره می‌شود.