import cv2

import mediapipe as mp

import numpy as np

import matplotlib.pyplot as plt

# Initialize mediapipe face mesh

mp\_face\_mesh = mp.solutions.face\_mesh

face\_mesh = mp\_face\_mesh.FaceMesh(static\_image\_mode=True)

# Upload an image to the session (you can do this in Colab using the file upload button)

from google.colab import files

uploaded = files.upload()

# Load the uploaded image

image\_path = list(uploaded.keys())[0]

image = cv2.imread(image\_path)

image\_rgb = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

# Process image to detect facial landmarks

results = face\_mesh.process(image\_rgb)

# Create copy for output

image\_output = image.copy()

# Nose landmark indexes (based on mediapipe face mesh index map)

nose\_indexes = [1, 2, 98, 327, 168]  # Tip, nostrils, sides

if results.multi\_face\_landmarks:

    for face\_landmarks in results.multi\_face\_landmarks:

        h, w, \_ = image.shape

        nose\_points = []

        for idx in nose\_indexes:

            x = int(face\_landmarks.landmark[idx].x \* w)

            y = int(face\_landmarks.landmark[idx].y \* h)

            nose\_points.append((x, y))

            cv2.circle(image\_output, (x, y), 2, (0, 255, 0), -1)

        # Simulate nose reshaping by drawing a polygon and shrinking it

        nose\_mask = np.zeros\_like(image)

        nose\_pts = np.array(nose\_points, np.int32)

        cv2.fillConvexPoly(nose\_mask, nose\_pts, (255, 255, 255))

        # Warp: Shrink the nose area inward a little

        shrink\_factor = 0.7

        center\_x = int(np.mean([pt[0] for pt in nose\_points]))

        center\_y = int(np.mean([pt[1] for pt in nose\_points]))

        for i in range(len(nose\_points)):

            dx = nose\_points[i][0] - center\_x

            dy = nose\_points[i][1] - center\_y

            nose\_points[i] = (int(center\_x + dx \* shrink\_factor), int(center\_y + dy \* shrink\_factor))

        # Warp nose region

        nose\_warp = cv2.seamlessClone(

            src=image,

            dst=image,

            mask=cv2.cvtColor(nose\_mask, cv2.COLOR\_BGR2GRAY),

            p=(center\_x, center\_y),

            flags=cv2.MIXED\_CLONE

        )

        # Show results

        plt.figure(figsize=(10, 5))

        plt.subplot(1, 2, 1)

        plt.imshow(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))

        plt.title("Before")

        plt.axis('off')

        plt.subplot(1, 2, 2)

        plt.imshow(cv2.cvtColor(nose\_warp, cv2.COLOR\_BGR2RGB))

        plt.title("After (Simulated Rhinoplasty)")

        plt.axis('off')

        plt.tight\_layout()

        plt.show()

else:

    print("No face detected.")