**Documentation: Seamlessly Integrating a Person into a Scene**

**Objective**

To create a photorealistic composite image where a person is seamlessly integrated into a chosen background. The integration must account for shadow generation, lighting direction, and color matching to create a realistic final output.

**Tech Stack and Tools Used**

**Libraries and Tools**

* **Python**: Main scripting language
* **PIL (Pillow)**: For image manipulation, drawing, and compositing
* **NumPy**: For pixel-wise operations and matrix transformations
* **OpenCV**: For image processing, edge detection, and Hough transforms
* **rembg**: For background removal using pre-trained deep learning models
* **Matplotlib**: For visualization

**Algorithms & Techniques**

* Gaussian Blur for soft shadows
* Affine transformation for skewing shadows
* Hough Transform for detecting shadows and estimating direction
* Colorization and transparency mapping
* Alpha compositing

**Step-by-Step Breakdown**

**Task 1: Capturing and Preparing the Person's Image**

**Step 1: Capture a High-Quality Image**

* Ensure proper lighting to minimize shadows on the person
* Front-facing image with a neutral or solid-colored background

**Step 2: Remove the Background**

* **Library**: rembg
* Converts the input image into a transparent PNG by removing the background
* Output is saved as person\_no\_bg.png

remove\_background("source.png", "person\_no\_bg.png")



**Task 2: Analyzing Shadows and Lighting of the Background Image**

**Step 1: Detect and Classify Shadows**

* Convert background to grayscale
* Threshold image to isolate dark regions (potential shadows)
* Apply edge detection (Canny)
* Use Hough Line Transform to find direction of shadow lines

\_, shadow\_mask = cv2.threshold(gray, 80, 255, cv2.THRESH\_BINARY\_INV)

lines = cv2.HoughLinesP(edges, 1, np.pi/180, threshold=80, ...)

**Classification**

* Hard Shadows: Well-defined, sharp edges in edge map
* Soft Shadows: Diffused, appear over larger regions

**Task 3: Determining Light Direction**

**Step 1: Compute Light Direction**

* Calculate average direction vector from shadow lines using HoughLinesP
* Normalize to get direction of incident light
* Arrow is drawn to visually represent this direction

**Step 2: Indoor Scenes**

* Use histogram and gradient detection to estimate general ambient light



**Task 4: Coloring and Blending**

**Step 1: Match Shadow Color**

* Use a light gray (#101010) instead of full black for more natural blending

shadow\_np[..., :3] = 68 # RGB value for light black

**Step 2: Align Shadow Horizontally**

* Resize shadow to be stretched horizontally
* Affine transformation to simulate light direction

shadow = shadow.resize((new\_width, shadow\_height))

shadow = shadow.transform(..., Image.AFFINE, ...)

**Step 3: Shadow Blurring**

* Gaussian blur applied to simulate natural falloff

shadow = shadow.filter(ImageFilter.GaussianBlur(5))

**Task 5: Generating the Final Output**

* Combine resized person and shadow into a single image
* Paste onto the background with shadow aligned properly
* Save the final composited image

background.paste(shadow, shadow\_offset, shadow)

background.paste(person, position, person)

**Final Output**

* Shows person and shadow aligned with background lighting



**Missing Steps Identified and Implemented**

1. **Resizing and Placement**: Resizing person image to blend with background perspective
2. **Color Matching**: Used color tint and transparency to simulate natural lighting
3. **Shadow Construction**: Custom horizontal shadow that matches estimated light direction
4. **Shadow Placement**: Adjusted offset so shadow appears to be cast on the ground realistically
5. **Smoothing**: Gaussian blur used to soften shadow edges for realism

**Deliverables**

* **Final Image**: final\_output.png (Photorealistic integration)
* **Algorithm Documentation**: (This document)
* **No code submission**: Only the final output image and methodology are required

**Summary**

By combining Python libraries such as PIL, OpenCV, and rembg with shadow generation and light direction estimation techniques, we’ve created a full workflow for inserting a person into a new scene photo realistically. This includes:

* Background removal
* Shadow creation
* Light direction estimation
* Color and spatial blending