Function Documentation: `apply_curve_transformation`

1 Description

The `apply_curve_transformation` function applies a specified transformation function to an image. It ensures that the image is in a float format to avoid clipping during the transformation, applies the transformation, clips the values to the 0-1 range, and converts the image back to an 8-bit format.

2 Function Definition

```
def apply_curve_transformation(image, transformation_function, lower_threshold,
↪  upper_threshold):
    # Ensure image is in a float format to avoid clipping during the transformation
    float_image = image.astype(np.float32) / 255.0

    # Apply the transformation function
    transformed_image = transformation_function(float_image, lower_threshold,
    ↪  upper_threshold)

    # Clip values to the 0-1 range and convert back to an 8-bit format
    transformed_image = np.clip(transformed_image, 0, 1) * 255
    transformed_image = transformed_image.astype(np.uint8)

    return transformed_image
```

3 Function Explanation

3.1 Step-by-Step Breakdown

Function 1: Ensure Image is in Float Format

Ensure the image is in a float format to avoid clipping during the transformation.

```
# Ensure image is in a float format to avoid clipping during the transformation
float_image = image.astype(np.float32) / 255.0
```

Explanation: The image is first converted to a floating-point format by dividing the pixel values by 255. This normalization step ensures that the pixel values range between 0 and 1. This is important to avoid any clipping issues that might occur during the transformation process.

Function 2: Apply the Transformation Function

Apply the specified transformation function.

```
# Apply the transformation function
transformed_image = transformation_function(float_image, lower_threshold, upper_threshold)
```

Explanation: The transformation function is applied to the normalized image. The `transformation_function` is passed along with the `float_image`, `lower_threshold`, and `upper_threshold` parameters. This function modifies the pixel values according to the specified transformation logic.

Function 3: Clip Values and Convert Back to 8-Bit Format

Clip the transformed image values to the 0-1 range and convert back to an 8-bit format.

```
# Clip values to the 0-1 range and convert back to an 8-bit format
transformed_image = np.clip(transformed_image, 0, 1) * 255
transformed_image = transformed_image.astype(np.uint8)
```

Explanation: After applying the transformation, the pixel values are clipped to the range [0, 1] to ensure they stay within valid bounds. The image is then multiplied by 255 to convert it back to the 8-bit format and cast back to an unsigned 8-bit integer type (`np.uint8`).

4 Conclusion

The `apply_curve_transformation` function is designed to facilitate image preprocessing by applying a specified transformation function to an image. It ensures the image is in a float format to avoid clipping during the transformation, applies the transformation, and then clips and converts the image back to an 8-bit format.