

```

image1.py ×
173 cumhist_manual = manual_cumulative_histogram(hist)
174 print("✓ Cumulative histogram computed using Manual implementation")
175
176 # =====
177 # PART 3: HISTOGRAM EQUALIZATION
178 # =====
179 print("\n" + "=" * 60)
180 print("STEP 4: PERFORMING HISTOGRAM EQUALIZATION")
181 print("=" * 60)
182
183 equalized_opencv = cv2.equalizeHist(image)
184 hist_eq_opencv = cv2.calcHist([image], channels=[0], histSize=[256], range=[0, 256])
185 print("Histogram equalization performed using OpenCV")
186
187 equalized_manual = manual_histogram_equalization(image)
188 hist_eq_manual = manual_histogram(equalized_manual)

in image1 ×
1. 4_comparision_all_methods.png
2. 4_comparision_all_methods.png
3. 4_comparision_all_methods.png
4. 4_comparision_all_methods.png

Thank you for using this program!
=====

Process finished with exit code 0

```

The screenshot shows a Jupyter Notebook interface. On the left, the code for histogram equalization is displayed, including imports and implementations for both OpenCV and manual methods. The right side shows a 'Plots' section with four subplots: 'Original Image' (the input mountain image), 'Histogram (OpenCV)' (a blue histogram showing the original frequency distribution), 'Histogram (Manual)' (a red histogram showing the manual implementation), 'Cumulative Histogram (OpenCV)' (a blue cumulative distribution plot), and 'Cumulative Histogram (Manual)' (a red cumulative distribution plot). Below the notebook, a larger figure displays the results of histogram equalization for both methods. It includes two versions of the 'Histogram Equalized' image (one for OpenCV and one for manual) and their corresponding histograms and cumulative distributions. The histograms show a more uniform distribution of intensity levels after equalization.



