## Step 1: Media Type Selection

**Chosen Media type:** Images

Examples might include AI-generated faces (e.g., from ThisPersonDoesNotExist) or art compared to real photos or human-drawn artwork.

## Step 2: Collected Examples

I asked 3 different AIs to generate an image of a cat:

1. **ChatGPT**:



1. **DeepAI**:



1. **Leonardo**



I have 3 pictures of my cats:

1. Deena:



1. Gordon:



1. Kiki:



## Step 3: Brainstorm Specific Traits with AI

1. Artifacts and Inconsistencies
   1. AI usually struggles with making faces symmetrical. While people and animals don't have perfectly symmetrical features they are usually somewhat symmetrical. AI can sometimes misalign facial features or make them inconsistent. AI also is not very good at generating backgrounds. They usually have weird distortions or random text.
2. Lack of Fine Detail
   1. AI pictures are usually overly smooth, especially in hair and skin. AI also struggles with unique patterns so it usually repeats elements.
3. Unusual Lighting and Reflections
   1. Shadows and highlights usually don't align properly with the light source. AI also does not create reflections well so mirrors will have mismatched or missing reflections.

## Step 4: Prompt an AI to Write a Detection Program

AI prompt:

Create a Python program to determine if an image is AI-generated or human-generated. The program should analyze the following traits: Artifacts and Inconsistencies, Lack of Fine Detail, and Unusual Lighting and Reflections. Use image processing libraries like OpenCV or PIL. Please include detailed comments explaining each step of the code.

Code:

import cv2

import numpy as np

def detect\_artifacts(image\_path):

"""Detects artifacts/blurriness using adaptive Laplacian variance."""

image = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)

laplacian = cv2.Laplacian(image, cv2.CV\_64F).var()

# Dynamic threshold based on median sharpness of natural images

threshold = max(100, laplacian \* 0.15) # Adjust threshold dynamically

return laplacian < threshold, laplacian

def detect\_fine\_details(image\_path):

"""Detects lack of fine details using edge density, normalized."""

image = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)

edges = cv2.Canny(image, 50, 150) # Adjusted Canny parameters

edge\_density = np.sum(edges) / (image.shape[0] \* image.shape[1])

# Normalize edge density for better AI detection

normalized\_edge\_density = edge\_density / 255

return normalized\_edge\_density < 0.02, normalized\_edge\_density

def detect\_lighting\_inconsistencies(image\_path):

"""Analyzes unnatural brightness distributions using histogram variance."""

image = cv2.imread(image\_path)

image\_hsv = cv2.cvtColor(image, cv2.COLOR\_BGR2HSV)

\_, \_, v = cv2.split(image\_hsv)

hist\_v = cv2.calcHist([v], [0], None, [256], [0, 256])

hist\_variance = np.var(hist\_v)

# AI images may have extremely low OR high variance, check for outliers

if hist\_variance < 50000 or hist\_variance > 2000000:

return True, hist\_variance

return False, hist\_variance

def analyze\_image(image\_path):

"""Runs all three detection functions and determines AI vs. real."""

artifacts, laplacian\_value = detect\_artifacts(image\_path)

fine\_details, edge\_density = detect\_fine\_details(image\_path)

lighting, hist\_variance = detect\_lighting\_inconsistencies(image\_path)

# Debugging output to analyze failures

print(f"Debug - Laplacian Variance: {laplacian\_value:.2f}, Edge Density: {edge\_density:.6f}, Histogram Variance: {hist\_variance:.2f}")

print("Analysis Results:")

print(f"Artifacts Detected: {artifacts} (Laplacian Variance: {laplacian\_value:.2f})")

print(f"Lack of Fine Details: {fine\_details} (Normalized Edge Density: {edge\_density:.6f})")

print(f"Lighting Inconsistencies: {lighting} (Histogram Variance: {hist\_variance:.2f})")

ai\_score = sum([artifacts, fine\_details, lighting])

if ai\_score >= 2:

print("Final Verdict: The image is likely Cat-generated (human-generated).")

else:

print("Final Verdict: The image is likely AI-generated.")

if \_\_name\_\_ == "\_\_main\_\_":

image\_path = "Test\_image.jpg" # Replace with your test image

analyze\_image(image\_path)

## Step 5: Run the Program and Test It

| Image | How the program labeled it | Artifacts Detected | Lack of Fine Details | Lighting Inconsistencies |
| --- | --- | --- | --- | --- |
| Deena | AI-generated | True | False | True |
| Kiki | AI-generated | True | True | True |
| Gordon | AI-generated | True | False | True |
| GPT | Cat-generated | False | False | True |
| DeepAI | Cat-generated | False | False | True |
| Leonardo | Cat-generated | False | False | True |

## Step 6: Reflect and Analyze

How well did the program differentiate between AI-generated and human-generated media? Were there any examples it misclassified?

It almost did great but did so badly instead. It classified everything as the opposite of what it should be. Everything that is true for AI images was false for the actual AI images but true for the real ones. It is a bit funny how wrong AI can be at detecting AI.

### Feature Analysis:

What aspects of the media (e.g., symmetry, coherence, artifacts) did the program analyze?

It allegedly analyzed the photos to see if they had artifacts, a lack of details, and inconsistencies in the lighting. I say allegedly because it analyzed everything incorrectly.

Why do you think these aspects were useful (or not useful) for detecting AI generation?

I would say not useful because of the fact it classified them wrong. However, if it had been analyzed correctly specifically the artifacts and details it would have been useful. To our eyes, we can tell that the AI images are AI because we can see the lack of detail in the fur and the background being distorted specifically the Leonardo and GPT ones. Those should have been analyzed correctly and if it had been the classification would have been different. The lighting inconsistencies analysis is not very useful, since real-life images usually have different lighting sources because of that it can look inconsistent.

### Limitations and Improvements:

What limitations did you notice in the program’s approach?

The program produces incorrect results 100% of the time (at least from my tests). The analysis of the images was incorrect and the opposite of the correct answers.

How could the detection program be improved?

Maybe using different or more traits to analyze. I think because some of the real images had backgrounds that were not simple might have messed up the artifacts analysis. That could be why the AI images with the more simple backgrounds were false and the real ones were true. I think adding something that specifically analyzes the cat's ears, eyes and whiskers would have been more accurate.