


Identifying Friend or Foe

Oğuz Kaynak



Project Overview



Build a Python program that detects faces and determines if they're a friend or foe using OpenCV and Haar cascade classifiers.

Tools and Libraries

- **Python 3.x**
- **OpenCV (cv2):** A powerful computer vision library that provides tools for image processing, face detection, and video analysis.
- **Haar Cascade XML Classifiers:** Pre-trained classifiers provided by OpenCV for detecting faces and other objects using the Viola-Jones algorithm.
- **NumPy:** Often used in background for efficient matrix operations and pixel manipulation.
- **Image Processing Techniques:**
 - Grayscale conversion
 - Histogram analysis for face comparison
 - Blurring or labeling based on recognition outcome

How It Works

1. Load Haar classifier

2. Capture frames

3. Convert to grayscale

4. Detect faces

5. Mark and classify

6. Action based on classification

Code: Face Detection

- `face_cascade = cv2.CascadeClassifier(...)`

Loads the pretrained face detection model

- `gray = cv2.cvtColor(frame, ...)`

Converts the image to grayscale to simplify detection

- `faces = face_cascade.detectMultiScale(...)`

Detects faces in the image

Code: Classification

For each detected face:

- Crop the face region from the frame
- Resize to match stored reference images
- Convert both images to grayscale for consistency
- Calculate similarity (e.g., using histogram comparison or mean squared error)

Example:

```
similarity = cv2.compareHist(hist_face, hist_known, cv2.HISTCMP_CORREL)
```

```
if similarity > threshold:
```

```
    label = 'Friend'
```

```
else:
```

```
    label = 'Foe'
```

References

- Vaughan, Lee. *Real-World Python: A Hacker's Guide to Solving Problems with Code*. No Starch Press, 2020.
- OpenCV Documentation:
<https://docs.opencv.org/>
- Python Official Documentation:
<https://docs.python.org/>
- face_recognition GitHub Repository:
https://github.com/ageitgey/face_recognition

