FUNDAMENTALS OF FACIAL DETECTION

ISU - TASK FIVE

ENG4UN // Yuxi Qin // April 2021

Syllabus

- Demo
- Breaking down the code
- How it works
- Artificial Intelligence / Machine Learning
- Detection vs recognition
- Real life applications
- Relating to *Physics of the Future*

Demonstration

```python
import cv2
import sys

# this program applies basic object detection using OpenCV
# requires Python, the OpenCV library, and an internal video capturing device
# openCV docs: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html

cascPath = "cascade.xml" # path to the pretaught model for frontal-faces
faceCascade = cv2.CascadeClassifier(cascPath)
video_capture = cv2.VideoCapture(0)
# determines path to video capture device

# runs forever - as long as the program doesn't end
while True:
    # capture the source frame-by-frame
    ret, frame = video_capture.read()

    # casts a filter onto the frame so the computer can actually work with it
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # detects faces using whatever settings
    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.1,
        minNeighbors=5,
        minSize=(30, 30)
    )

    # draw a rectangle around the faces
    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

    # displays the final frame onto the video frame
    cv2.imshow('Video', frame)

    # exits the program if you press 'q'
    if cv2.waitKey(1) & 0xFF ==ord('q'):
        break

# release the final capture when everything is done
video_capture.release()
cv2.destroyAllWindows()
```

```
import cv2
import sys

# this program applies basic object detection using OpenCV
# requires Python, the OpenCV library, and an internal video capturing device
# openCV docs: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html

# path to the pretrained model for frontal-faces
cascPath = "haarcascade_frontalface_alt.xml"
faceCascade = cv2.CascadeClassifier(cascPath)
video_capture = cv2.VideoCapture(0)  # determines path to video capture device
```

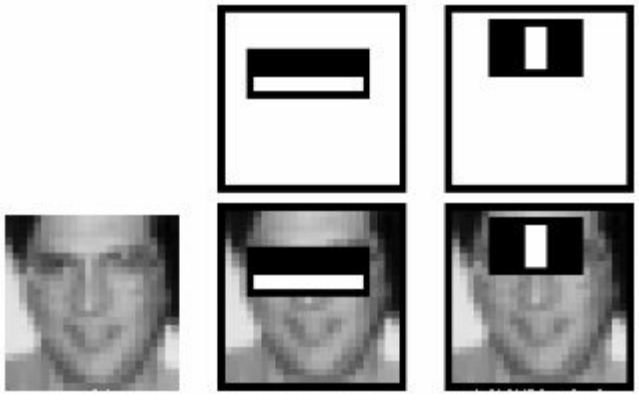
```
# runs forever - as long as the program doesn't end
while True:
    # capture the source frame-by-frame
    ret, frame = video_capture.read()

    # casts a filter onto the frame so the computer can actually work with it
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # detects faces using specified settings
    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.1,
        minNeighbors=5,
        minSize=(30, 30)
    )
```

```
        # draw a rectangle around the faces
        for (x, y, w, h) in faces:
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

        # displays the final frame onto the video frame
        cv2.imshow('Video', frame)

        # exits the program if you press 'q'
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

# release the final capture when everything is done
video_capture.release()
cv2.destroyAllWindows()
```

Demonstration

How it works

- Splits the image up into parts
- Calculates brightness transitions
- Identifies patterns within the image
  - ex. eye regions are darker than nose and cheek regions
- Uses trained models with pattern recognition algorithms



Artificial Intelligence and Machine Learning

Artificial Intelligence

- Creating machines that can mimic human thinking, decision making, and behaviour

Machine Learning

- Subset of AI
- Machines learn and make decisions based on large amounts of data rather than being explicitly programmed

Detection vs Recognition

Detection

- **Where is *this* object in this image?**
  - **Input: clear image of the desired object + another image (possibly) containing the object of interest**
  - **Output: position/box of input object**

Recognition

- ***Which* object is depicted in this image?**
  - **Input: image containing unknown objects**
  - **Output: position and labels/names of objects within the image**

Applications IRL

- Autonomous transportation
  - Detecting and identifying road signs, other vehicles, pedestrians, etc.
- Medical image processing
  - Help detect disease more accurately
- Surveillance and security
  - Facial recognition, object tracking, activity recognition

Relating to *Physics of the Future*

- "One can mass-produce hardware and increase its power by piling on more and more chips, but you cannot mass-produce the brain."
- Facial detection program acts as a watered-down version of what goes on behind the scenes of big tech solutions
- Expanding on ideas addressed in chapters that focused on the future of computers and AI

Link in description

To learn more about these new technologies, check out the VMCSC blog for student-written articles on topics such as:

- Autonomous Transportation
- Artificial Intelligence and Machine Learning
- VR/AR

Want to run the face detection program yourself? Check out my Github!