Bilder analysieren und beschreiben lassen mit künstlicher Intelligenz

Dass du mit Hilfe von ChatGPT bzw. DALL-E Bilder erzeugen kannst, weißt du sicherlich bereits schon. Aber es geht auch andersherum: Mit **Vision von OpenAI** kannst du Bilder analysieren – also herausfinden, was darauf zu sehen ist. Und: Das funktioniert auch mit Videos.

In diesem Projekt lernst du, wie du mit Python und der API von OpenAI

- Ein einzelnes Bild beschreiben lässt
- Ein Video ins passende Format bringst, um es beschreiben zu lassen und
- Ein Foto mit dem Raspberry Pi aufnimmst, es analysieren und die Beschreibung vertonen lässt.

Was ist OpenAI Vision?

Üblicherweise kommunizierst du mit ChatGPT oder anderen

Sprachmodellen über einen Text-Prompt. Du sendest eine Anforderung oder Frage – und die KI antwortet dir. Mit GPT-4 gibt es jedoch auch die Möglichkeit, **zusätzlich zum Text auch ein Bild mitzusenden.** Du kannst also zum Beispiel fragen, was auf dem Bild zu sehen ist. Damit machst du dir die multimodalen Eigenschaften zunutze, die dir ermöglichen, verschiedene Medientypen zu kombinieren.

Das Vision zugrunde liegende Modell **gpt-4-vision-preview** kannst du aktuell (März 2024) nur über die API von OpenAI nutzen. Hierfür benötigst du einen Account bei OpenAI und einen API-Key. Wie du beides erstellst, erfährst du in diesem Tutorial. Um mit der API zu interagieren, eignet sich Python. Hierfür stellt OpenAI eine Bibliothek zur Verfügung, mit der du die gewünschten Funktionen unkompliziert aufrufen kannst.

Ein einzelnes Bild beschreiben lassen

Als erstes Beispiel soll ein einzelnes Bild dienen, das du per API an ChatGPT sendest und von der KI beschreiben lässt. Dieses Bild ist lokal auf deinem Rechner gespeichert. Um es zu übertragen, konvertierst du es zunächst in das Format Base64.

Zusammen mit dem kodierten Bild sendest du deinen Prompt mit – die Frage, was auf dem Bild zu sehen ist. Nach wenigen Sekunden erhältst du die Antwort zurück, die du dann in der Konsole ausgeben kannst. Hier ein Beispiel eines Bilds eines Panthers auf dem Mond und darunter die Interpretation von ChatGPT:

Das sieht ChatGPT in dem Bild:

Das ist ein fiktives Bild, das eine Katze in einem Astronautenanzug darstellt. Die Katze steht auf einer unebenen, mondähnlichen Oberfläche, und im Hintergrund ist ein großer erdähnlicher Planet mit verschiedenen Monden und Sternen sowie Galaxien im Weltraum zu sehen. Es handelt sich um eine künstlerische Darstellung, die Elemente aus der Science-Fiction-Szene mit einem Hauch von Humor kombiniert, indem sie ein Haustier in den Kontext der Raumfahrt setzt.

Nicht schlecht, oder? Gut, die KI hat aus dem Panther eine Katze gemacht – aber das kann man ihr wohl verzeihen.

Das Python-Script

Um solch eine Bildbeschreibung zu erstellen, benötigst du nicht viel Code. Hier das Script, das ich dafür verwendet habe:

```
import base64
import requests
from openai import OpenAI
import os

#Fall nötig: Das Verzeichnis auf das setzen, in dem das Script liegt.
os.chdir(os.path.dirname(os.path.realpath(__file__)))

# OpenAI API-Key
api_key = "DEIN API-KEY VON OPENAI"

# Das Bild konvertieren
def encode_image(image_path):
  with open(image_path, "rb") as image_file:
    return base64.b64encode(image_file.read()).decode('utf-8')

# Pfad zum Bild
image_path = "panther.jpeg"

# Den Base64-String erstellen
base64_image = encode_image(image_path)

headers = {
  "Content-Type": "application/json",
  "Authorization": f"Bearer {api_key}"
}

payload = {
  "model": "gpt-4-turbo",
```

```
  "messages": [
    {
      "role": "user",
      "content": [
        {
          "type": "text",
          "text": "Was ist auf dem Bild zu sehen?"
        },
        {
          "type": "image_url",
          "image_url": {
            "url": f"{base64_image}"
          }
        }
      ]
    }
  ],
  "max_tokens": 300
}

response = requests.post("https://api.openai.com/v1/chat/completions", headers=headers, json=payload)
print(response.json()["choices"][0]["message"]["content"])
```

Das obige Script orientiert sich an einem Beispiel von OpenAI.

Bevor du das Script laufen lässt, stelle zunächst sicher, dass du die benötigten Bibliotheken installierst hast. Trage außerdem deinen API-Key von OpenAI ein. Dann benötigst du noch den Pfad zum Bild, das du analysieren und beschreiben lassen möchtest. Trage diesen hinter **image_path =** ein. Falls dein Bild im gleichen Verzeichnis liegt, wie dein Script – Python dieses jedoch nicht findet, kann die Zeile ganz oben unterhalb der Moduleinbindungen hilfreich sein. Damit setzt du das Verzeichnis auf jenes, in dem dein Script gespeichert ist.

Weiter unten im Script findest du den Prompt, der deine Anweisung bzw. Frage zum Bild enthält. Hier **Was ist auf dem**

Bild zu sehen? Je nachdem, was du genau vorhast oder wissen möchtest, kannst du diesen Prompt natürlich anpassen. Wenn du also zum Beispiel wissen möchtest, welche Farben die Blumen auf einer Wiese haben, passe den Prompt entsprechend an.

Damit die Beschreibung von ChatGPT nicht zu ausschweifend und kostspielig wird, kannst du mit **max_tokens** noch eine maximale Anzahl an verwendeten Tokens festlegen.

Ein Video analysieren

Was mit einem einzelnen Bild geht, funktioniert auch mit einem Video – da ein solches nichts anderes ist als aneinander gereihte Fotos. Und so stellst du ein Video ChatGPT auch bereit, nämlich Frame für Frame.

Das folgende Script öffnet eine lokal gespeicherte Video-Datei und zerlegt es in einzelne Frames. Anschließend erfolgt die Abfrage bei der API, in der du diese Einzelbilder bereitstellst und ebenfalls deinen Prompt (also deine Frage) mitsendest:

```
import cv2 #Nicht installiert? -> pip install opencv-python
import base64
import time
from openai import OpenAI
import os

#Fall nötig: Das Verzeichnis auf das setzen, in dem das Script liegt.
os.chdir(os.path.dirname(os.path.realpath(__file__)))

client = OpenAI(api_key = "DEIN API-KEY VON OPENAI")

video = cv2.VideoCapture("DEIN VIDEO.mp4")

base64Frames = []
while video.isOpened():
    success, frame = video.read()
    if not success:
```

```
        break
    _, buffer = cv2.imencode(".jpg", frame)
base64Frames.append(base64.b64encode(buffer).decode("utf-8"))

video.release()
print(len(base64Frames), "frames read.")

PROMPT_MESSAGES = [
    {
        "role": "user",
        "content": [
            "Dies sind Frames eines Videos. Beschreibe, was
darin zu sehen ist.",
             *map(lambda x: {"image": x, "resize": 768},
base64Frames[0::50]),
        ],
    },
]
params = {
    "model": "gpt-4-turbo",
    "messages": PROMPT_MESSAGES,
    "max_tokens": 400,
}

result = client.chat.completions.create(**params)
print(result.choices[0].message.content)
```

Wie du siehst, ist das Script etwas anders aufgebaut als das erste. Hier sendest du deine Abfrage nicht per Request, sondern nutzt die Funktion **client**. Hinterlege wieder einen API-Key und in der Zeile **video = cv2.VideoCapture(„DEIN VIDEO.mp4“)** den Dateinamen des Videos, das du analysieren lasse möchtest. Falls es nicht im gleichen Ordner wie dein Script liegt, achte bitte auf den korrekten Pfad.

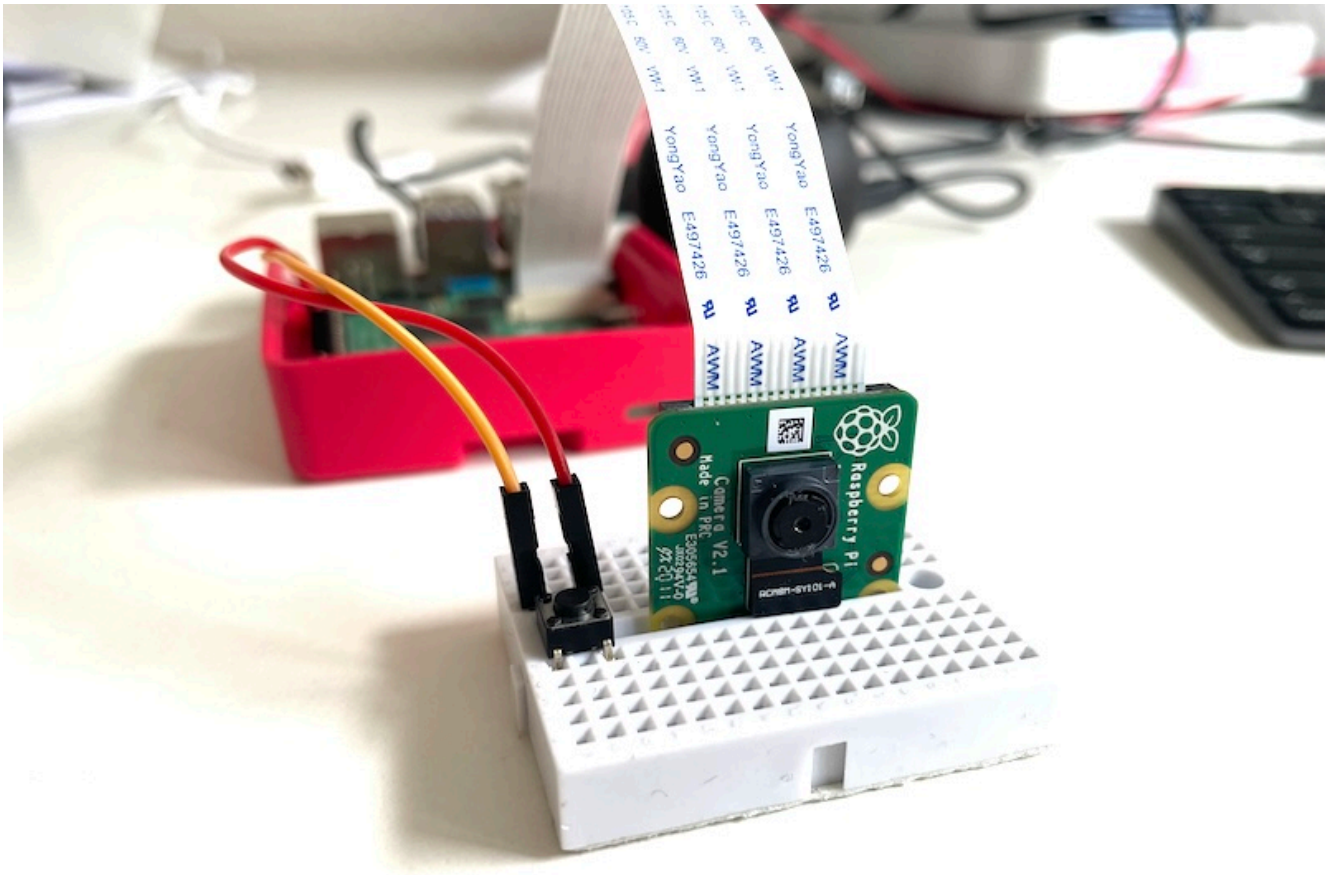
Im Bereich **PROMPT_MESSAGE = [** findest du den Prompt, den du mitsendest. Diesen kannst du wieder an deine Wünsche anpassen. Ebenso gibt es wieder die Einstellung **max_tokens** – hier kann allerdings eine größere Zahl nötig sein als bei dem einzelnen

Bild von vorhin. Wenn die erlaubte Anzahl an Tokens zu gering ist, erhältst du möglicherweise keine vollständige Beschreibung des Videos.

Mit dem Raspberry Pi ein Foto aufnehmen und es beschreiben lassen

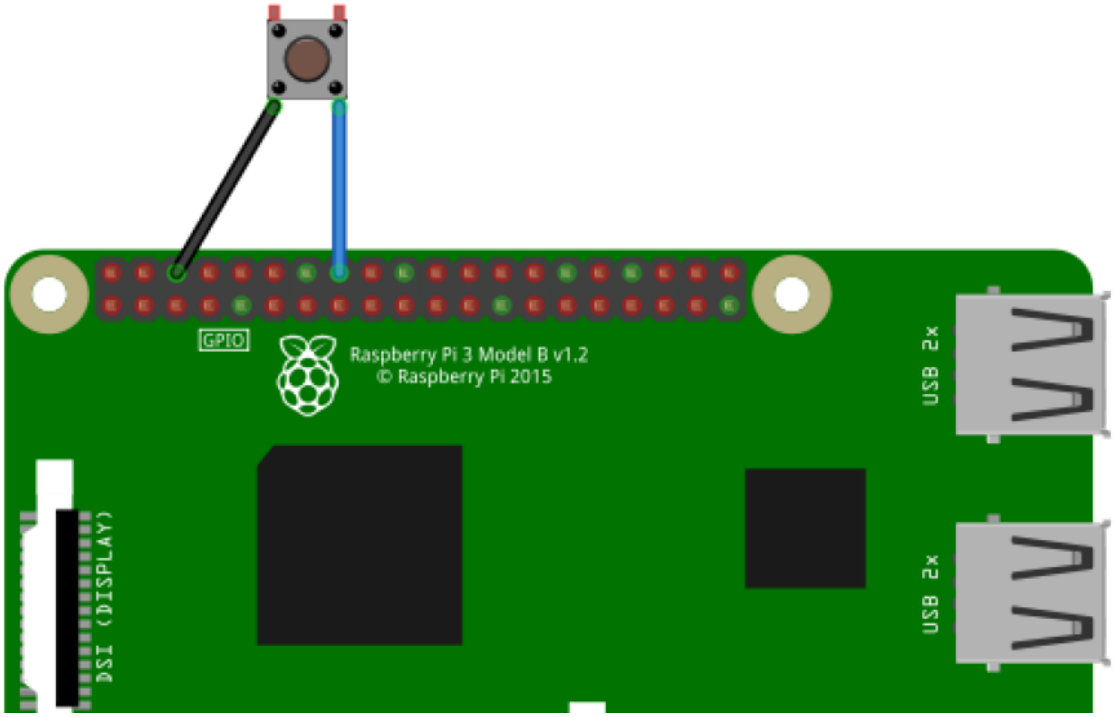
Bisher hast du bereits vorhandene Bilder oder Videos verwendet. Du kannst aber natürlich auch ein Foto mit der Kamera am Raspberry Pi aufnehmen und dieses von ChatGPT beschreiben lassen. Diesmal gibst du die Analyse jedoch nicht einfach nur in der Konsole aus, sondern lässt sie dir vorlesen.

Damit baust du dir ein Gerät, das du verwenden könntest, um zum Beispiel dir unbekannte Gegenstände, Gemälde oder Gebäude zu identifizieren. Auch für Menschen mit eingeschränktem oder überhaupt keinem Sehvermögen könnte das eine interessante und hilfreiche Anwendung sein.



Das Projekt aufbauen

Ausgelöst wird die Kamera und die anschließende Analyse durch einen gedrückten Button am Raspberry Pi. Schließe diesen wie folgt an. Einen Pulldown- oder Pullup-Widerstand benötigst du hierbei nicht.



Wie du die Kamera anschließt, habe ich in diesem Projekt zur Objekterkennung am Raspberry Pi beschrieben. Allerdings verwende ich darin noch die Option **Legacy Camera**, die du nun eigentlich nicht mehr benötigst. Leider war das Thema „Kamera am Raspberry Pi“ eine Zeitlang etwas komplex, weswegen es schwierig ist, eine Patentlösung anzubieten, die für die meisten Versionen von Pi und zugehörigem Betriebssystem passt.

Möglicherweise wirst du hier etwas herumprobieren müssen. Im Folgenden verwende ich das Betriebssystem **Debian Bookworm (64 Bit)** und die Bibliothek **Picamera2**. Ob deine Kamera korrekt angeschlossen ist und funktioniert, kannst du übrigens im Terminal schnell testen. Führe dafür einfach den folgenden Befehl aus:

```
libcamera-still -o test.jpg
```

Wenn alles funktioniert, erscheint ein Vorschaufenster mit dem aktuellen Kamerabild. In diesem Fall steht dem folgenden Python-Script dann nichts mehr im Wege.

Das Python-Script

Das folgende Script erweitert die hier vorangegangen Programme ein wenig. Zunächst wird die Bildbeschreibung erst ausgeführt, sobald der angeschlossene Button gedrückt wurde. Dann wird die Kamera gestartet, ein Foto aufgenommen und dieses dann an ChatGPT übertragen. Nachdem die Antwort vorliegt, wird diese in Sprache umgewandelt und vorgelesen.

Mehr zum Thema „Text in Sprache umwandeln“ findest du übrigens in diesem Tutorial bei uns. **Und noch ein Hinweis:** Möglicherweise fragst du dich, was mit deinen aufgenommen Fotos passiert, nachdem du sie an ChatGPT übertragen hast. Laut Angaben von OpenAI werden diese nicht für das Training der KI verwendet. Falls du hier jedoch sichergehen möchtest, solltest du auf die Verwendung dieser Fotos vielleicht lieber verzichten.

Hier das Script:

```
import cv2 #Nicht installiert? -> pip install opencv-python
import base64
from openai import OpenAI #pip install openai
import os
from picamera2 import Picamera2 #pip install picamera2
import RPi.GPIO as GPIO
import pygame #pip install pygame
import time
from pathlib import Path

GPIO.setmode(GPIO.BOARD)
buttonPin = 16
GPIO.setup(buttonPin, GPIO.IN, pull_up_down=GPIO.PUD_UP)

# Dateiname des aufgenommenen Fotos
image_file = 'image.jpg'

def main():

    # Ein Foto machen
```

```
    print("Ich nehme ein Foto auf.")
    picam2 = Picamera2()
                                    camera_config        =
picam2.create_still_configuration(main={"size": (1920, 1080)},
lores={"size": (640, 480)}, display="lores")
    picam2.configure(camera_config)
    picam2.start()
    time.sleep(2)
    picam2.capture_file(image_file)

     client = OpenAI(api_key=os.environ.get("OPENAI_API_KEY",
"DEIN OPENAI API-KEY"))

    image = cv2.imread(image_file)
    _, buffer = cv2.imencode(".jpg", image)
    base64Image = base64.b64encode(buffer).decode("utf-8")

    print("Ich beschreibe das Foto.")

    PROMPT_MESSAGES = [
        {
            "role": "user",
            "content": [
                "Dies ist ein Foto. Beschreibe, was darauf zu
sehen ist.",
               {"image": base64Image, "resize": 768},
            ],
        },
    ]
    params = {
        "model": "gpt-4-turbo",
        "messages": PROMPT_MESSAGES,
        "max_tokens": 200,
    }

    result = client.chat.completions.create(**params)
    description = result.choices[0].message.content
    print(description)

    #Vertonung
    speech_file_path = Path(__file__).parent / "vision.mp3"
```

```
    response = client.audio.speech.create(
    model="tts-1",
    voice="alloy",
    input=description
    )
    response.stream_to_file(speech_file_path)
    pygame.init()
    pygame.mixer.init()
    pygame.mixer.music.load(speech_file_path)
    pygame.mixer.music.play()
    while pygame.mixer.music.get_busy():
        pass
    pygame.quit()

if __name__ == "__main__":
    while True:
        buttonState = GPIO.input(buttonPin)
        if buttonState == GPIO.LOW:
            main()
        else:
            print(".")
```

Wie du siehst, kommen hier noch ein paar weitere Bibliotheken ins Spiel. Wie du sie mit Pip im Terminal installiert, steht als Kommentar jeweils dahinter.

Sobald du alles vorbereitet hast, starte das Script (Kopfhörer bzw. Lautsprecher nicht vergessen). Nachdem du den Button gedrückt hast, sollte das Foto übertragen werden und dir die Bildbeschreibung vorgelesen werden. Hab ein bisschen Geduld, so richtig schnell wird das leider nicht funktionieren – aber es sollte meistens deutlich unter einer Minute dauern.