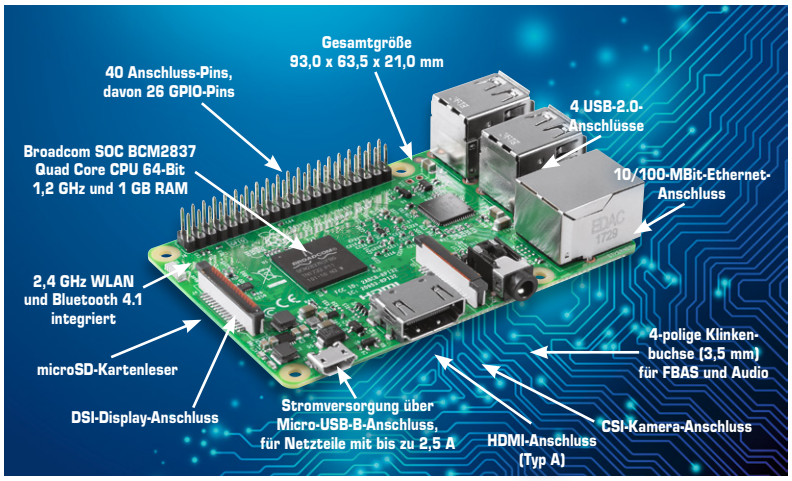
Raspberry Pi

Teil 8: Raspberry-Pi-Kameramodul als Kamerasensor

Artikelserie
im ELV Shop
#10036

Dass auf dem Raspberry Pi eine Muster- und Gesichtserkennung und sogar eine Gesichtswiedererkennung möglich sind, beweisen zahlreiche DIY-Projekte im Internet und die Projekte im Franzis-Buch „Foto und Video mit Raspberry Pi" (ISBN: 978-3645603140). Mit dem passenden Stück Software lässt sich eine angeschlossene Kamera – unabhängig davon, ob Sie das Raspberry-Pi-Kameramodul oder eine USB-Kamera verwenden – auch für Sensorik- und Robotikzwecke nutzen.



Erkennung von Gesichtern im Eigenbau

Grundsätzlich benötigen Sie neben dem passenden Treiber für die Kamera-Hardware, die in der Regel bereits installiert ist, eine passende Logik, mit der Sie die Informationen lesen und verarbeiten können. In diesem Beispiel wird ein einfaches Projekt vorgestellt, mit dem Sie das Raspberry-Pi-Kameramodul als Sensormodul für die einfache Erkennung von Gesichtern verwenden können. Damit ließe sich zum Beispiel in Kombination mit dem Bewegungsmelder ein Projekt für die Haustür realisieren: Wird eine Bewegung wahrgenommen, dann prüft die Kamera an der Haustür durch den Türspion o. Ä., ob ein Gesicht erkannt wird oder nicht. Falls ein Gesicht erkannt wird, dann schaltet sich das Eingangslicht ein; lief beispielsweise eine Katze im Eingangsbereich oder hat ein Windstoß den Bewegungsmelder ausgelöst, dann bleibt das Eingangslicht aus.

OpenCV für die Kamera

Für eine einfache Muster- oder Gesichtserkennung auf dem Raspberry Pi benötigt das Kameramodul mit der OpenCV-Bibliothek eine Erweiterung. Hier brauchen Sie also das Rad nicht komplett neu zu erfinden, sondern nur richtig anzuwenden. Um die OpenCV-Funktionen nachzurüsten, bringen Sie zunächst den Raspberry Pi auf den aktuellen Stand und installieren dazu noch die OpenCV-Bibliothek. Mit den OpenCV-Funktionen können Sie später auch per Python auf die Kamera zugreifen.

```
sudo -i
apt-get update
apt-get install libopencv-dev
apt-get install python-opencv
```

Die Paketverwaltung sorgt in diesem Fall dafür, dass 120 neue Pakete mit einer Größe von 138 MByte heruntergeladen und installiert werden – und das kann je nach vorhandener Internetgeschwindigkeit etwas dauern. Im nächsten Schritt erstellen Sie dazu ein Python-Skript, mit dem Sie die Fotoaufnahme nach bestimmten Mustern durchsuchen, um ein Gesicht in der Aufnahme erkennen zu können.

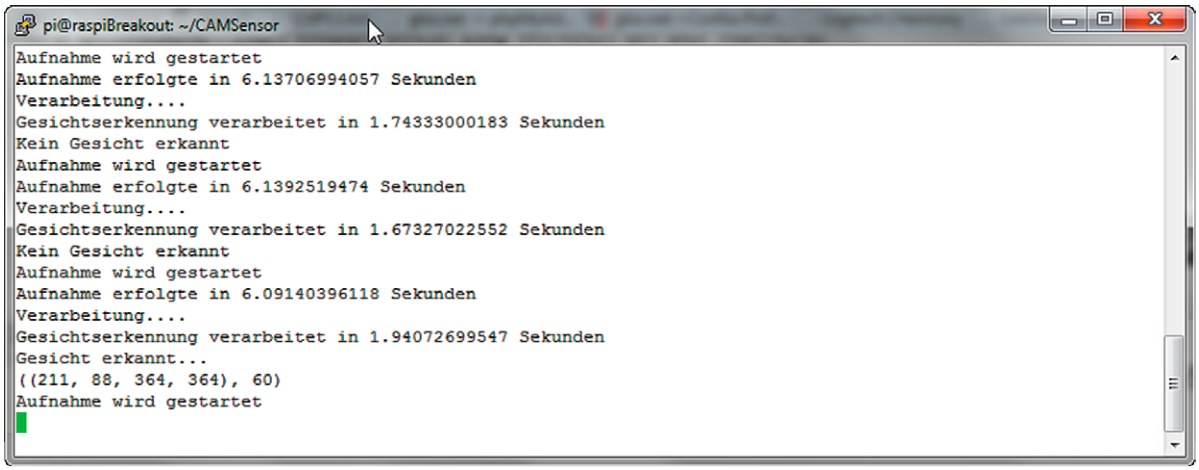
Kamerasensor für Gesichtserkennung

Grundsätzlich verwendet OpenCV für die Gesichtserkennung eine Technik, die auf sogenannten Beschreibungsdateien beruht. Diese Dateien werden aus einer Vielzahl von Bildern errechnet und OpenCV über eine „xml"-Beschreibungsdatei hinzugefügt. OpenCV liefert bereits ein Paket an solchen vortrainierten Beschreibungsdateien, sodass Sie umgehend starten können. Für die Gesichtserkennung hat sich in der Praxis die sogenannte *haarcascade_frontalface_default.xml*-Datei bewährt, wenngleich diese Methode lediglich mit Frontalaufnahmen umgehen kann und fehleranfällig gegenüber verschiedenen Größen von Gesichtern ist.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# ------------------------------------------------------------------
# sensor-face-detect.py
# ACHTUNG: sensor-face-detect.py funktioniert nur dann, falls RPi-Kameramodul
```

```
# UND cv-Bibliothek auf dem RPi installiert sind.
# Pfad: /CAMSensor
import os, time, cv
storage = cv.CreateMemStorage()
haarcascade = cv.Load('/usr/share/opencv/haarcascades/haarcascade_frontalface_default.xml')
imagefile = "/home/pi/aufnahme.jpg"
# ------------------------------------------------------------------------
def main():
  os.system('clear')
  while (True):
    print("Aufnahme wird gestartet")
    start = time.time()
    os.system("if [ -e " + imagefile +" ]; then sudo rm " + imagefile +"; fi; ")
    os.system("/opt/vc/bin/raspistill -w 800 -h 600 -o " + imagefile)
    end = time.time()
    print("Aufnahme erfolgte in " + str(end-start) + " Sekunden")
    print("Verarbeitung....")
    start = time.time()
    image = cv.LoadImage(imagefile)
    facefound = cv.HaarDetectObjects(image, haarcascade, storage, 1.2, 2, cv.CV_HAAR_DO_CANNY_PRUNING, (100,100))
    end = time.time()
    print("Gesichtserkennung verarbeitet in " + str(end-start) + " Sekunden")
    if facefound:
      print("Gesicht erkannt...")
      for face in facefound:
        print(face)
    else:
      print("Kein Gesicht erkannt")
    time.sleep(5)
if __name__ == '__main__':
  main()
sys.exit(0)
# ------------------------------------------------------------------------
```

Die Beschreibungsdatei binden Sie über das Load-Kommando ein. Im nächsten Schritt starten Sie eine While-Schleife und fertigen mit der Kamera eine Aufnahme an, die Sie über Load Image für OpenCV bereitstellen und anschließend mit der „Haar Detect Objects"-Funktion auf gemeinsame Treffer prüfen. Je nachdem, ob in der Aufnahme ein Gesicht gefunden wurde oder nicht, erfolgt die entsprechende Bildschirmausgabe über das „if"-Konstrukt. Hier können Sie noch weitere Dinge antriggern, beispielsweise bei erfolgreicher Gesichtserkennung einen Schaltvorgang/Lichtschalter etc. automatisiert starten und vieles mehr.



Gesicht erkannt, Gefahr gebannt: Schlägt die Gesichtserkennung des Kamerasensors an, dann lassen sich dank Python weitere Dinge automatisch über den Raspberry Pi erledigen – beispielsweise eine Relaissteuerung für die Beleuchtung im Treppenhaus.

Im Vergleich zu anderen Sensoren ist der Kamerasensor zwar unheimlich praktisch, erfordert jedoch einen passgenauen Anwendungszweck, um die Fehlerrate zu minimieren. Aus diesem Grunde ist der Kamerasensor gerne als Ergänzung zu bestehenden Ultraschall- und Infrarotsensoren in der Robotik im Einsatz. ELV