

Tutorial KI-Objekterkennung

Mit Hilfe des Raspberry PI



Inhaltsverzeichnis

[1 Vorwort 1](#_Toc182970480)

[2 Aufsetzen des Systems 1](#_Toc182970481)

[2.1 Hardware 1](#_Toc182970482)

[2.2 Software 2](#_Toc182970483)

[3 Erster Start 3](#_Toc182970484)

[4 Anlernen 6](#_Toc182970485)

[4.1 Setup der Software 6](#_Toc182970486)

[4.2 labelImg 9](#_Toc182970487)

# Vorwort

Dieses Projekt mit dem Thema Objekterkennung mithilfe von KI wurde im Rahmen eines Gruppenprojekts der Klasse F2TV23 im Modul 4 durchgeführt. Ziel war es, ein Objekterkennungssystem auf einem Raspberry PI einzurichten und dieses in eine einfache Steuerung zu integrieren.

Anzumerken ist, dass wir zum Start des Projekts so gut wie keine Programmiererfahrungen hatten und dass diese für das Projekt auch kaum benötigt wurden.

Ein großer Dank geht an dieser Stelle an chatgpt, der uns bei allen Programmierfragen tatkräftig zur Seite stand.

Für die Einrichtung unserer Objekterkennung haben wir folgendes Tutorial durchgearbeitet: <https://youtu.be/3YqbO2AlepM?si=_XAimw8OLMK9Y_EE>

Dieses Dokument kann als schriftliches Tutorial verwendet werden, in dem wir unsere Vorgehensweisen und Problemstellungen festhalten.

# Aufsetzen des Systems

Wenn man dieses Projekt mit diesem Tutorial durchführen möchte, ist es von Vorteil **exakt** die gleiche Hardware sowie Software zu verwenden.

Wenn man eine andere Komponente verwendet/austauscht kann es zu Kommunikationsstörungen der einzelnen Komponenten kommen.

## Hardware

Wir haben folgende Hardware benötigt, um das System aufzusetzen:

* Raspberry PI 4 B
* Raspberry PI Camera Module 3, 12MP
* USB-C Netzteil
* HDMI zu HDMI Mikro Kabel
* USB-Tastatur und Maus
* microSD Karte, evtl. Kartenleser

## Software

Bevor wir auf dem Raspberry PI arbeiten können, müssen wir ein Betriebssystem auf unsere microSD Karte installieren. Die Karte sollte vorher formatiert werden, bzw. wichtige Daten sollten vorher separat abgespeichert werden, da diese sonst gelöscht werden.

Danach werden folgende Schritte durchgegangen:

1. Herunterladen und installieren von Raspberry PI Imager von <https://www.raspberrypi.com/software/>
2. Auswahl von Raspberry PI 4, Raspberry PI OS (64-BIT)**,** und der SD-Karte, stellt sicher, dass ihr das aktuelle Debian **Bookworm** (Stand 07.11.2024) auswähltEin Bild, das Text, Screenshot, Software, Webseite enthält.

   Automatisch generierte Beschreibung
3. (optional) Benutzerdaten festlegen, dies kann später noch am Raspberry PI geändert werden, Empfehlung ist zumindest das Tastaturlayout auf Deutsch zu stellen
4. Nach einiger Zeit ist das Betriebssystem installiert und die SD-Karte kann entfernt werden

# Erster Start

Wir haben für unser Projekt mit **Bookworm** (Stand 07.11.2024) und python 3.11 gearbeitet. Wenn hier andere Versionen genutzt werden kann es zu Kommunikations-Problemen zwischen den einzelnen Komponenten kommen.

Der erste Start dauert etwas länger, da noch Updates installiert werden. Ist der Raspberry gestartet, öffnen wir zunächst das Terminal und geben folgende Befehle ein:

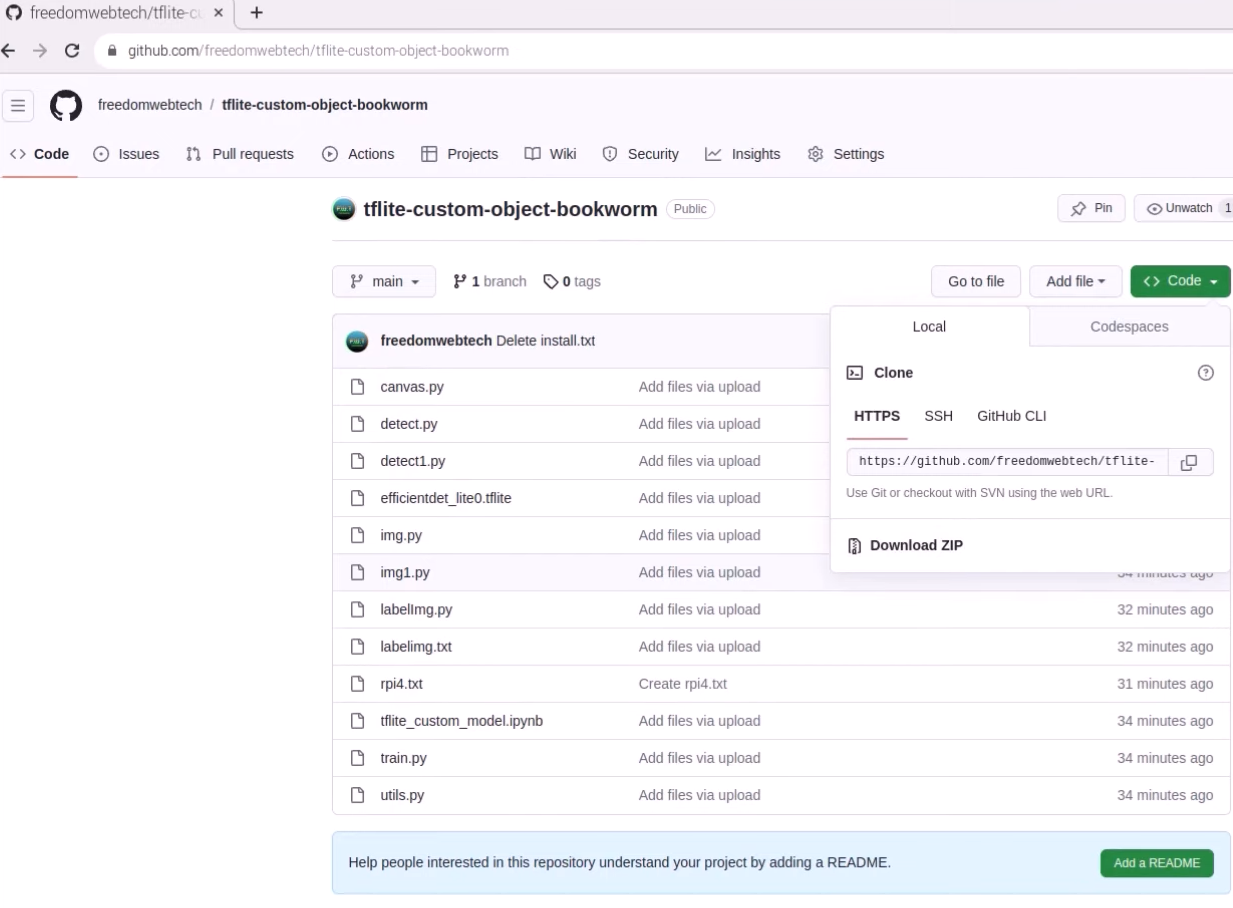
sudo apt update

sudo apt upgrade

Hiermit setzen wir unser Betriebssystem auf den neuesten Stand.

Dann besuchen wir folgende Seite, klicken auf <> Code und downloaden die .zip Datei:

<https://github.com/freedomwebtech/tflite-custom-object-bookworm>



Die .zip Datei extrahieren wir in den Documents Ordner. Der Dateipfad für den Ordner sollte jetzt lauten:

/home/kibbz/Documents/tflite-custom-object-detection-bookworm-main

Dann gehen wir wieder in unser Terminal und geben die Befehle

sudo rm /usr/lib/python3.11/EXTERNALLY-MANAGED

sudo pip3 install opencv-python

und

sudo pip3 install mediapipe

ein.

Ein Bild, das Text, Elektronik, Screenshot, Software enthält.

Automatisch generierte Beschreibung

Wenn eine Webcam verwendet wird, öffnet man nun **detect.py** mit dem vorinstallieren python-Programm thonny. Falls wie bei uns eine PiCamera verwendet wird, muss stattdessen **detect1.py** geöffnet werden.

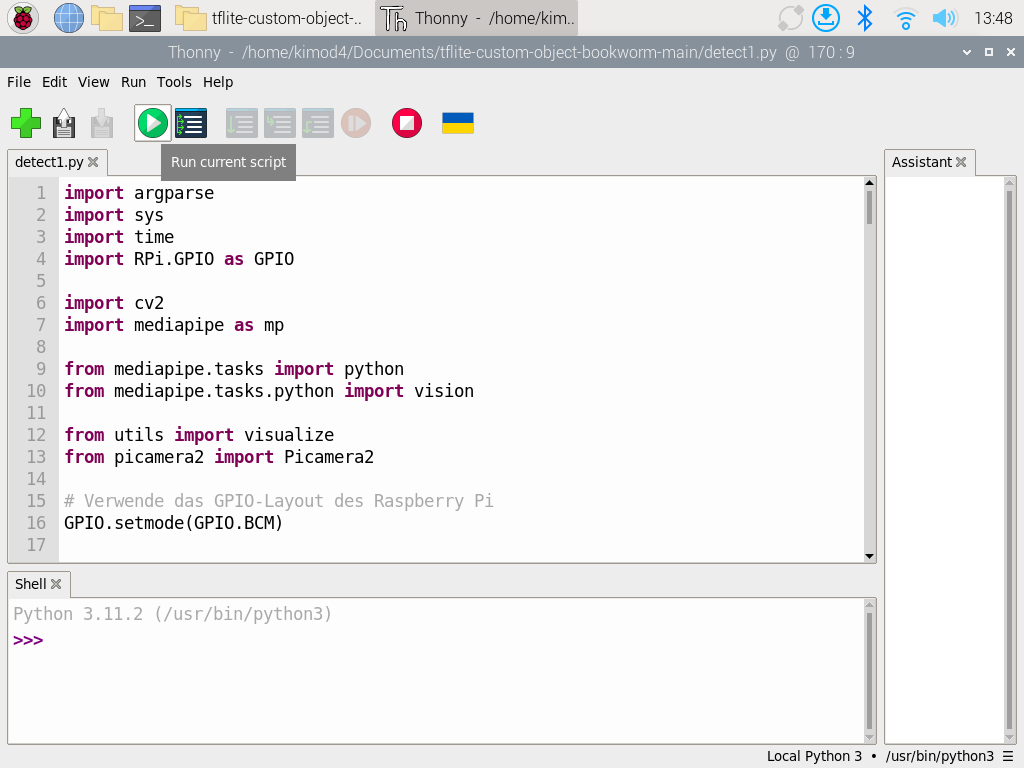
Mit einem # werden in python Kommentare gekennzeichnet, diese haben keine Auswirkungen auf den Code, sondern dienen als Orientierung.

In Zeile 132 und 133 des Scripts wird auf die tflite-Dateien verwiesen. Hier muss eine der Zeilen aktiviert sein und die andere als Kommentar stehen. Wenn Zeile 132 (efficientdet\_lite0.tflite) aktiv ist, wird ein Standard/Vorgegebenes Erkennungsprogramm gestartet. Wenn Zeile 133 (best.tflite) aktiv ist wird ein selbst angelerntes Programm gestartet.

Ein Bild, das Text, Schrift, Screenshot enthält.

Automatisch generierte Beschreibung

Um unser Skript zu testen, aktivieren wir nun Zeile 132, kommentieren Zeile 133 und lassen thonny das Skript ausführen.



Nun sollte sich ein extra Fenster öffnen, in dem Objekte erkannt werden.

# Anlernen

Im nächsten Schritt wird erklärt, wie man mit tflite eigene Objekte anlernt und diese erkennt. Zunächst müssen Bilder der Objekte gemacht werden, die erkannt werden sollen. Diese können mit dem Raspberry selbst oder mit einer externen Kamera (z.B. Handy) erstellt werden.

Wir empfehlen möglichst viele Bilder zu machen, um eine hohe Präzision beim Erkennen zu erreichen.

Außerdem empfehlen wir erstmal nur wenige verschiedene Objekte anzulernen.

Mit den Skripten **img.py** und **img1.py** können die Fotos mit dem Raspberry PI erstellt werden.

## Setup der Software

Für das weitere Anlernen muss das Programm labelImg installiert werden. labelImg dient dazu unseren Bildern eine Beschriftung zuzufügen.

Dafür geben wir folgende Befehle in das Terminal ein:

sudo apt-get install pyqt5-dev-tools

sudo pip3 install labelimg

Ein Bild, das Text, Elektronik, Screenshot, Software enthält.

Automatisch generierte Beschreibung

Als nächstes müssen wir die canvas.py-Datei im Pfad

/usr/local/lib/python3.11/dist-packages/libs

sowie die labelImg.py-Datei im Pfad

/usr/local/lib/python3.11/dist-packages/labelImg

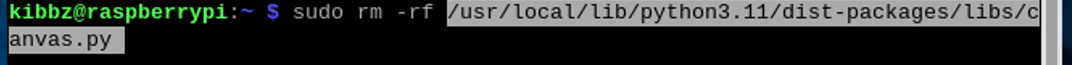
ersetzen. Dazu öffnen wir diese Pfad im Datei-Browser und kopieren den Dateipfad der canvas.py Datei.

Ein Bild, das Text, Screenshot, Software, Display enthält.

Automatisch generierte Beschreibung

Um die Datei zunächst zu löschen, geben wir folgenden Befehl ins Terminal ein:

sudo rm -rf „kopierter Dateipfad“



Nun müssen wir die canvas.py-Datei, welche wir am Anfang heruntergeladen haben in diesen Ordner bewegen. Dafür müssen wir zuerst im Terminal in das richtige directory wechseln und verwenden dann den move-Befehl:

cd /home/kibbz/Dokumente/tflite-custom-object-bookworm-main/

sudo mv canvas.py usr/local/lib/python3.11/dist-packages/libs

Ein Bild, das Text, Schrift, Software, Reihe enthält.

Automatisch generierte Beschreibung

Das gleiche führen wir mit der labelImg.py-Datei durch.

**Wichtig**: Darauf achten, dass immer im korrekten Dateipfad gearbeitet wird!

Nun wird es Zeit sich etwas um die Ordnerstruktur zu kümmern. Wir erstellen in unserem tflite Ordner zwei weitere Ordner und benennen den ersten Images und den anderen freedomtech.

Ein Bild, das Text, Screenshot, Software, Computersymbol enthält.

Automatisch generierte Beschreibung

Im freedomtech Ordner erstellen wir dann zwei weitere Unterordner names train und validate.

Ein Bild, das Text, Schrift, Screenshot, weiß enthält.

Automatisch generierte Beschreibung

In unseren Images Ordner fügen wir nun die zuvor erstellten Bilder ein, die wir zum Anlernen benutzen wollen.

## labelImg

Jetzt können wir anfangen mit labelImg unsere Bilder zu beschriften. Dazu geben wir im Terminal einfach

labelImg

ein und es sollte sich die Software öffnen. Dann ändern wir den Ort zum Öffnen (1) und Speichern (2) der Bilder auf unseren Images Ordner ab.

Nun sollten sich die Bilder öffnen lassen und wir können mit der Auswahl Create RectBox (3) auf der linken Seite unsere Bilder beschriften.

Ein Bild, das Text, Screenshot, Software, Display enthält.

Automatisch generierte Beschreibung

3

2

1

Die Box muss möglichst genau um das Objekt gezogen werden. Nach Erstellung der Beschriftung speichern wir diese und klicken auf Next Image. Diesen Vorgang führen wir für alle Bilder einzeln durch.

Ein Bild, das Text, Visitenkarte, Screenshot, Materialeigenschaft enthält.

Automatisch generierte Beschreibung

Ein Bild, das Elektronisches Gerät, Text, Gerät, Computermaus enthält.

Automatisch generierte BeschreibungRechteckige Objekte bieten sich zum Anlernen am besten an.

Wenn wir mit diesen Vorgang abgeschlossen haben, sollte in unserem Images Ordner zu jedem Bild eine .xml Datei existieren.

Ein Bild, das Text, Screenshot, Software, Computersymbol enthält.

Automatisch generierte Beschreibung

Wir kopieren als nächstes den ganzen Inhalt des Ordners nun ebenfalls in unsere vorher erstellten train und validate Ordner.

Im nächsten Schritt erstellen wir aus dem freedomtech Ordner eine .zip Datei. Dazu müssen wir uns im Terminal wieder im richtigen Pfad befinden:

cd /home/kibbz/Dokumente/tflite-custom-object-bookworm-main/

Dann können wir mit dem Befehl

sudo zip -r freedomtech.zip freedomtech/\*

eine .zip Datei erstellen, welche im gleichen Ordner liegt.

Ein Bild, das Text, Screenshot, Software, Computersymbol enthält.

Automatisch generierte Beschreibung

## Google Colab

Im letzten Schritt wird Google Colab verwendet, um aus der .zip eine tflite Datei zu erstellen. Voraussetzung hierfür ist ein Google-Konto.

Wir laden die freedomtech.zip auf Google Drive hoch.

Nun öffnen wir Google Colab und klicken auf Hochladen.

<https://colab.research.google.com/>

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

Hier wählen wir die Datei tflite\_custom\_model.ipynb aus, welche sich in unserem tflite Ordner befindet.

Als erstes klicken wir oben in der Leiste auf Laufzeit und stellen den Laufzeittyp auf Python3 und T4 GPU, dann klicken wir oben rechts auf verbinden

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte BeschreibungEin Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Jetzt klicken wir einfach von oben beginnend die einzelnen Skripte durch, wobei manchmal eine Eingabeaufforderung bestätigt werden muss

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Wenn die Zeile



durchgelaufen ist, muss die train.py Datei hochgeladen werden.

Ein Bild, das Text, Screenshot, Software, Computersymbol enthält.

Automatisch generierte Beschreibung

3

2

1

1. Auf das Dateisymbol klicken
2. Auf Hochladen klicken
3. train.py auswählen

Als nächstes klicken wir links im Google Colab unsere train.py an woraufhin sich auf der rechten Seite ein python Skript öffnet.

Hier müssen wir in Zeile 22 und Zeile 28 die Label eintragen, welche wir mit der LabelImg erstellt haben (z.B. wie bei uns USB und Presenter).

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

Hier die eigenen Label eintragen

Außerdem ändern wir in Zeile 40 den epochs Wert von 20 auf 100

Ein Bild, das Text, Schrift, Screenshot, Schwarz enthält.

Automatisch generierte Beschreibung

Wir speichern unsere train.py Datei schließen diese und führen den letzten Satz aus.

Dieser Schritt dauert sehr lange, abhängig davon wie viele Bilder/Objekte erstellt wurden. Am besten über Nacht laufen lassen.

Google Colab erstellt uns nun daraus eine best.tflite Datei, welche wir auf der linken Seite herunterladen können.

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

Diese Datei kopieren wir in unseren tflite Ordner.

Nun müssen wir nur noch kontrollieren, dass die korrekte Datei aktiv ist und unsere eigene angelernte Objekterkennung sollte funktionieren.



# Nachwort

Das Projekt hat uns durch die vielen Kommunikationsproblemen zwischen den einzelnen Komponenten teilweise sehr verzweifeln lassen. Dafür war der Augenblick in dem von dem Raspberry der erste Schraubenzieher als Zahnbürste erkannt wurde umso schöner.

Auf die Integration in ein LOGO Netzwerk wird an dieser Stelle nicht eingegangen, die daran angepasste tflite Datei kann jedoch unter

Heruntergeladen werden.

Wir hoffen, dass das Dokument als hilfreicher Lernfaden dient und wünschen allen zukünftigen Nutzern viel Spaß und Erfolg bei ihren Projekten.