2413507

Raspberry Pi: Computerschach per Pi-Cam spielen

22.08.2019 | 10:12 Uhr | Swen Hopfe

Würden Sie gern mal auf einem konventionellen Schachbrett gegen einen Computer spielen? Mit einem Raspberry Pi und Pi-Cam ist das machbar.

Ich wollte gern gegen einen elektronischen Schachgegner auf einem konventionellen Brett spielen. Die Züge sollten nicht per Hand in eine Software eingegeben, sondern automatisch übermittelt werden, ohne das Spiel mit Sensoren und Elektronik zu versehen und verkabeln zu müssen. Ein wertiges Schachbrett und Figuren waren vorhanden, da wollte ich nicht mit diversen Umbauten eingreifen.

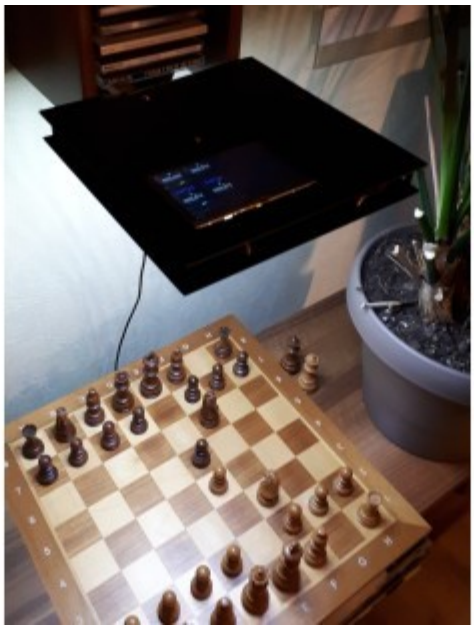
Vergrößern Mit Kamera abzutastendes Schachbrett

© Swen Hopfe

Deshalb sollte eine Kamera die Überwachung der Spielfläche übernehmen. Ein Raspberry Pi mit Pi-Cam erschien mir als richtig gut geeignet. Nutzt man Python, kann man auch eine Reihe von Chess-Engines als virtuellen Gegner einbinden.

So ist im Projekt ein Aufbau entstanden, bei dem der Pi mit seiner Kamera von oben auf die Spielfläche schaut, mittels Imagemagick die Bilder aufbereitet und die erkannten Züge der Schach-Engine Stockfish übermittelt.

Beim Entwurf kam ein Raspberry Pi 3 zum Einsatz. Imagemagick, unser Skript und die Chess-Engine funktionieren aber auch mit dem Pi Zero oder dem Raspberry Pi 4 mit Raspian *Stretch* oder *Buster* . Das gilt auch für die originale Cam. So hat man zwischen verschiedenen Modellen die Wahl. Unabhängig davon testet man bei der Anbindung von Displays verschiedener Hersteller am besten separat, um zuerst die passenden Einstellungen in der Pi-Konfiguration zu finden.



Vergrößern Die Chess-Cam über dem Spielbrett

Das für unsere Chess-Cam erdachte, recht einfache Konzept für die Erkennung der Züge hat nach einigen wenigen Anpassungen gut funktioniert.

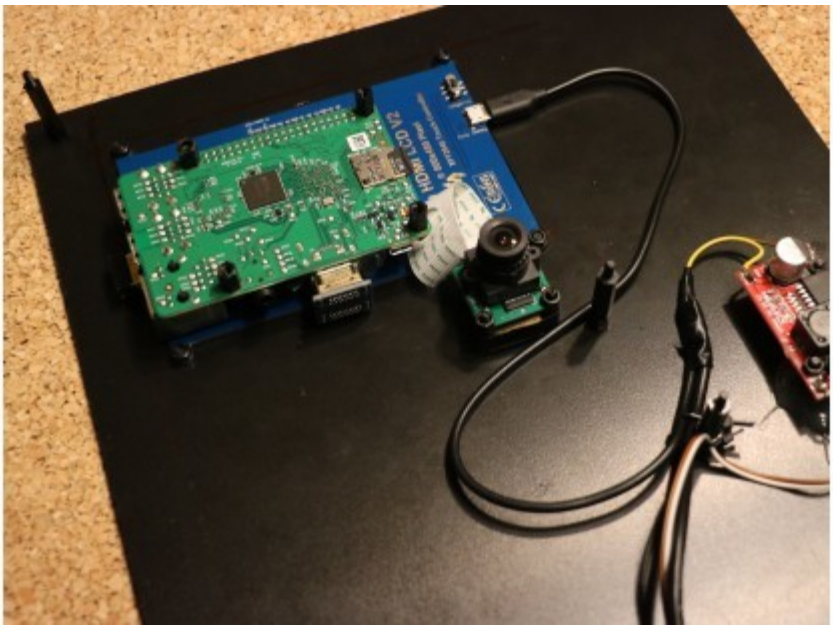
Zuerst wird eine Aufnahme der Grundstellung gemacht. Ist unsere Software bereit, fordert sie zum Zug auf. Hat man gezogen, quittiert man das mit einer Taste, analog dem Tippen auf die Schachuhr. Damit weiß die Anwendung, dass sie ein nächstes Foto machen kann.

Dann werden erste und zweite Aufnahme verglichen, indem ein Differenzbild erstellt wird. Das wird dann Ausschnitt für Ausschnitt abgetastet, so dass man hinterher weiß, auf welchen Feldern sich etwas verändert hat. Diese Felder sind also Anfang und Ende des Zuges einer Spielfigur. Wo sich der Anfang befindet, kann die Software leicht feststellen, denn die Positionen jeder Spielfigur sind in einer Matrix vermerkt. Die wird jetzt aktualisiert und der ermittelte Zug an die Chess-Engine übergeben.

Jetzt macht der Computergegner seinen Zug. Ist er fertig, bekommt man den Zug über das Display mitgeteilt. Die gegnerischen Figuren muss man jetzt nachrücken. Ist das gemacht, quittiert man wieder.

Und jetzt darf alles von vorn losgehen, bis der Computergegner das Ende der Partie festgestellt hat.

Nun muss man dafür sorgen, dass die Konstruktion in ausreichender Höhe angebracht ist, so dass die Kamera das Bild komplett erfassen kann. Man ist gut beraten, Beleuchtung und Bilderstellung separat zu testen, um beispielsweise festzustellen, ob genügend Kontrast zwischen Figuren und Feldern vorhanden ist, damit die Auswertung auch funktioniert. Für den Aufbau des Ganzen braucht es wie immer etwas handwerkliches Geschick, um die benötigten Teile zusammenzufügen.



Vergrößern Verkabelung der Komponenten

Bei uns sitzt der Pi rückwärtig am TFT-Display, für das ich in die obere Gehäuseplatte eine Aussparung geschnitten habe. Für die Nutzereingaben per Funktastatur steckt ein Sender an einem USB-Port vom Pi.

Das Gehäuse enthält noch einen DC/DC-Wandler, der die 12 V für die LED-Beleuchtung nach unten bereitstellt. Somit braucht es nur eine Verbindung nach draußen zu einem 5-V-Steckernetzteil und alles ist sehr übersichtlich. Im unteren Gehäusedeckel gibt es auch eine kreisrunde Aussparung für die Kamera.

Vergrößern LED-Beleuchtung

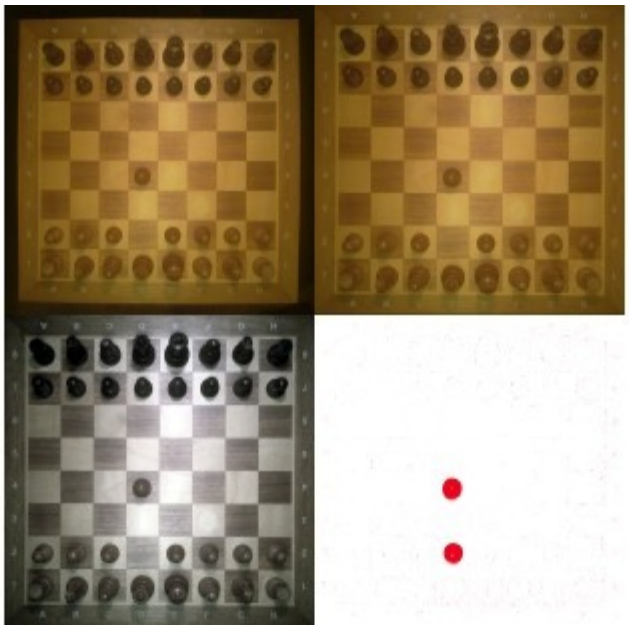
© Swen Hopfe

Unsere Software ist ein Python-Skript, das nach dem Einschalten und Hochfahren des Pi gestartet wird. Dafür haben wir es in */etc/rc.local* untergebracht.

Die Aufbereitung der Bilder mit Imagemagick umfasst bei uns das Zurechtschneiden, Entzerren und einen Weißabgleich. Zum Beispiel schneiden wir uns am Anfang aus der ersten Aufnahme der Pi-Cam das eigentliche Schachbrett quadratisch heraus:

```
$ envstr="convert "+basedir+"start.jpg"+" -crop 1400x1400+517+20
"+basedir+"start_crop.jpg"
$ os.popen(envstr)
```

Erst am Ende geschieht die Differenzbildung mit dem Vorgängerbild. Am Ende erhält man ein Differenzbild mit nur zwei Farben, im Bild hier Weiß und Rot für die veränderlichen Bestandteile. Das kann man nun über alle 8x8 Felder abtasten.



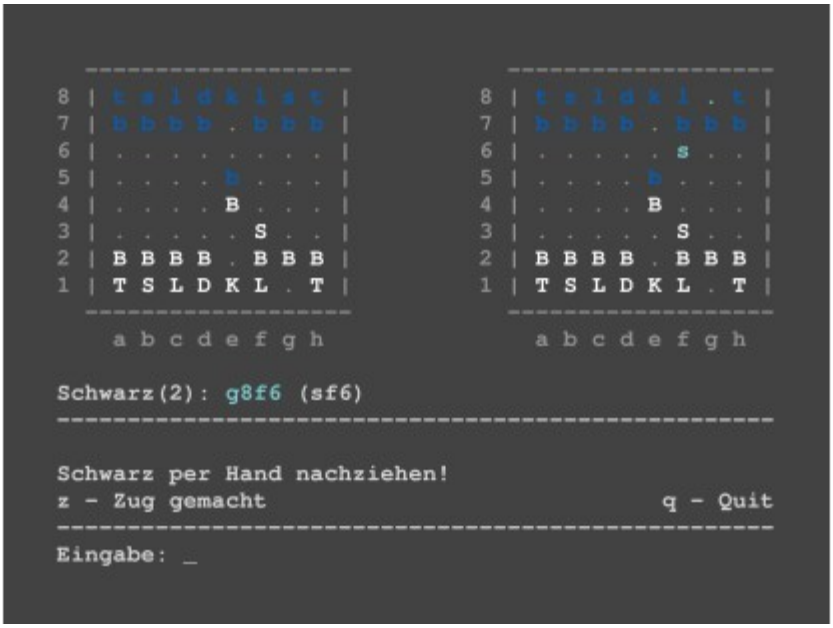
Vergrößern Bildberechnung

© Swen Hopfe

Ausgewertet wird bei uns, ob sich ein bestimmter Prozentsatz an Pixeln im Zentrum eines der 64 Felder vor und nach einem Zug geändert hat. Die Ergebnisse gehen in ein Koordinatensystem über. So werden auch Rochaden bemerkt.

Wie alles genau geschieht, ist aus dem Quelltext ersichtlich. Das Skript für den Raspi findet man auf Git-Hub .

Die Voreinstellungen sind so, dass man selber mit den weißen Figuren spielt. Im folgenden Bild hat gerade Schwarz einen Springer von g8 nach f6 gezogen. Unser Display zeigt nach einem Zug von Weiß oder Schwarz jeweils ein neues Brett in ASCII-Grafik an.



Vergrößern Ausgabe mit ASCII-Brettern

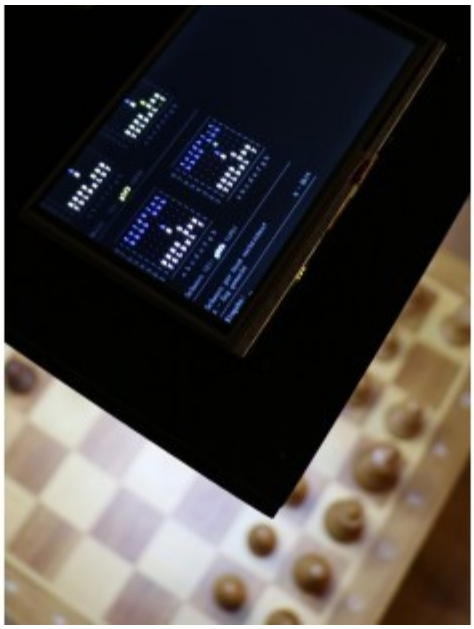
© Swen Hopfe

Dadurch kann man auch in einem Terminalfenster auf einem Rechner im Netzwerk spielen, wo man sich vorher per *ssh* auf dem Raspi eingeloggt und das Chess-Cam-Skript gestart hat. Das Interface sieht im Terminal gleich aus.

Die Stockfish-Bibliothek ist fest eingebunden. Man kann den Import aber auch modifizieren, um gegen andere Engines spielen zu können, die nach dem UCI-Standard funktionieren. Ein neues Spiel und Brett lässt sich unter der Chess-Bibliothek in Python so kreieren:

```
$ engine = chess.uci.popen_engine("stockfish")
$ command = engine.uci(async_callback=True)
$ command.result()
$ command = engine.ucinewgame(async_callback=True)
$ command.result()
$ board = chess.Board()
```

Als Hobby-Schachspieler wird man kaum gegen so eine Schach-Engine gewinnen können. Da hilft es, die Zeiten für das „Nachdenken“ vom elektronischen Gegner herabzusetzen. Und man kann auch noch an weiteren Parametern drehen. Für mich war wichtig, dass die Kamera-Abtastung vom Schachbrett funktioniert hat.



Vergrößern Fertige Chess-Cam während des Spiels

© Swen Hopfe

Bei uns ist nun ein Aufbau entstanden, wo Gehäuse und Kamera etwa 60 Zentimeter über dem Brett angebracht sind. Darauf ist auch die Kalibrierung bei der Bildverarbeitung abgestimmt. Ändert man seinen Aufbau bezüglich der Abstände oder der Brettgröße, muss man in der Programmierung wieder nachbessern.

Die zusätzliche Beleuchtung nach unten hilft, immer gleiche Lichtverhältnisse für die Aufnahmen zu haben. Steht das Ganze in einer schattigen Ecke im Raum, umso besser. Dass bei den Aufnahmen keine Hand störend im Bild ist, wird dadurch sichergestellt, dass immer

knapp nach dem Quittieren fotografiert wird. Der Taster auf der Schachuhr ist bei uns eine Taste auf der angebundenen Funktastatur.

Nach dem Einschalten kann man gleich loslegen. Und es macht richtig Spaß, zu sehen, wie der gerade bestätigte Zug erkannt und an die Chess-Engine übermittelt wird.

Dass man deren Antwort auf dem originalen Brett nachsetzen muss, ist wohl nur ein kleiner Nachteil, wo man doch auf sämtliche Mechanik verzichten kann...

Einige weitere Raspberry-Pi-Projekte:

NFC-Reader mit Raspberry Pi selbst bauen

Raspberry Pi Zero W als smarter USB-Stick

Zeitraffer-Aufnahmen mit dem Raspberry Pi Zero machen

Webcam mit Raspberry Zero Pi W einrichten

Mit Raspberry Pi Zero und Webcam unterwegs

Nächtliche Tierfotografie mit NoIR-Cam und Raspberry Pi