Kooperative, verteilte Überwachung eines Gebiets auf Eindringlinge

Dennis Lisiecki, Torsten Kühl

I. Einleitung

Tottis Baustelle

Ziel ist es, mithilfe von zwei oder mehr Rasperry Pis, die jeweils mit Kamera und Infrarot-Sensoren bestückt sind, ein Gebiet oder Raum auf unerwünschte Eindringlinge zu überprüfen.
Die Geräte sollen dazu an unterschiedlichen Stellen platziert werden, die den zu überwachenden Raum aus unterschiedlichen Blickwinkeln zeigen. Die Kamera des jeweiligen Geräts soll nun auf jegliche wahrgenommene Bewegung reagieren. Um unerwünschte Alarme auszuschließen, soll die Kamera außerdem mit dem Infrarot-Sensor, ("PIR-Sensor") kooperieren.
Der PIR-Sensor kommt immer dann zum Einsatz, wenn die Kamera eine Bewegung wahrgenommen hat. Wenn nun durch die Kamera eine Bewegung erkannt und diese ebenfalls durch den PIR-Sensor wahrgenommen wurde, soll ein Alarm dafür sorgen, dass eine Mitteilung an das Handy des Benutzers geschickt wird und dieser entsprechend informiert wird. In diesem Fall ist die Wahrscheinlichkeit, dass es sich um eine Lebensform handelt, sehr groß.
Um eine noch höhere Genauigkeit zu erzielen, sollen die Geräte miteinander kommunizieren:
Eine einzelne wahrgenommene Bewegung durch die Kamera eines einzelnen Raspberry Pi stellt das geringste Risiko dar. Zwei unterschiedliche Raspberry Pis, welche mit Kamera und PIR-Sensor eine Bewegung feststellen, stellen das höchste Risiko dar.

II. System

Tottis Baustelle

A. Raspberry Pi

Der Raspberry Pi, welcher in unserem Projekt die Grundlage bildet, ist ein voll funktionsfähiger PC im Scheckkartenformat. In erster Linie wurde der Raspberry Pi mit dem Ziel entwickelt, interessierten Menschen das erlernen von hardwarenaher Programmierung zu erleichtern. Jedes Gerät besitzt ein frei programmierbares General Purpose Input Output-Board ("GPIO-Board"). Das GPIO-Board stellt je nach Modell 26 oder 40 Pins zur Verfügung, von denen 17 bzw. 26 Pins frei programmierbar sind und die weiteren der Spannungsversorgung oder als Masse dienen. Die einzelnen Pins dieses Boards lassen sich mit selbst programmierten Programmen ansteuern und können vielseitigen Zwecken dienen. So kann diverse externe Peripherie angesteuert werden, wie z.B. ein Temperatur-Sensor, ein Ultraschall-Sensor, ein Motor oder sogar ein kleiner externen Monitor mit Touch-Funktion. Als Betriebssystem können unter anderem an die Architektur angepasste Linux-Distributionen wie das auf Debian basierende *Raspbian* installiert werden. Auch Betriebssysteme, welche den Raspberry Pi zum Mediencenter umfunktionieren, um damit Filme und Musik abzuspielen, sind von der Community mittlerweile zur Verfügung gestellt worden. Wie solche Projekte zeigen, ist der Raspberry Pi nicht nur zum lernen gut geeignet. Die Video- und Audioausgabe erfolgt über eine HDMI-Schnittstelle, für die Audioausgabe steht alternativ auch ein 3,5mm Klinkenanschluss zur Verfügung. Für die Stromversorgung wird ein 5-V-Micro-USB-Anschluss genutzt. Hier stehen dem Anwender viele Türen offen:
Neben den meisten Handy-Ladegeräten kann die Stromversorgung auch über Batterie und Solarzelle erfolgen. So kann man den Raspberry Pi wirklich in alles Situationen zum Einsatz bringen. Bis dato konnte sich der Raspberry Pi knapp 4 Millionen mal verkaufen und ist inzwischen in seiner vierten Version erschienen. [5] Die erste Version dieses Rechner kam Anfang 2012 auf den Markt und erfreut sich seither größter Beliebtheit. Je nach Ausführung ist das Gerät zwischen 25 und 35 Euro teuer. Die unterschiedlichen Ausführungen unterscheiden sich in gewissen Punkten:
Modell A und A+ besitzen 256 MB Arbeitsspeicher und nur einen USB-Anschluss, Modell B und B+ besitzen 512 MB Arbeitsspeicher, eine Ethernet-Schnittstelle sowie zwei, respektive vier USB-Anschlüsse. Alle Modelle müssen ohne Festplattenschnittstelle auskommen und verwenden SD-Karten bzw. Micro-SD-Karten als Speichermedium. Die aktuellste Version, den Raspberry Pi B+, verwenden wir für dieses Projekt.

B. Raspberry Pi Kameramodul

Für unser Projekt verwenden wir das Raspberry Pi Infrarot Kamera Modul ("Pi NoIR Camera Board"), weil es vom Raspberry Pi selbst auf jeden Fall unterstützt wird und der Support der Raspberry Pi Community hervorragend ist. Die Kamera bietet eine Auflösung von bis zu 5 Megapixel und kann bei statischen Aufnahmen mit einer Auflösung von bis zu 2592 x 1944 Pixel aufwarten. Mit Abmessungen von 25 x 20 x 9 mm ist die Kamera äußerst klein, muss allerdings auch ohne eigenes Gehäuse auskommen. Für die Raspberry Pi Kamera gibt es am Raspberry Pi einen eigenen Slot, in dem das Flachbandkabel der Kamera passt, sodass die GPIO-Anschlüsse

am Raspberry Pi vollständig für andere Aufgaben verwendet werden können. Da die Kamera dazu in der Lage ist, Licht aus dem Infraroten Spektralbereich einzufangen, kann man auch bei schlechter Beleuchtung oder nachts noch Bewegungen erkennen lassen oder Fotos schießen. Dazu muss bei Nacht der entsprechende Bereich mit Infrarotlicht ausgeleuchtet werden. Bei vielen Überwachungskameras wird dieses Licht mittels LEDs erzeugt, welche um das Objektiv der Kamera platziert werden. Für unser Projekt sollen allerdings vorerst keine Infrarot-LEDs zum Einsatz kommen. Natürlich können zur Bewegungserkennung auch gewöhnliche USB-Webcams verwendet werden. Wir geben allerdings zu bedenken, dass das Python Script, das wir zur Bewegungserkennung verwenden, nur das Raspberry Pi Kamera Modul unterstützt. Aber auch für diesen Anwendungsfall gibt es Lösungen, die wir für die Verwendung in unserem Projekt in Betracht gezogen haben. Dazu später mehr.

C. PIR-Sensor

Beim zweiten Sensor handelt sich um ein passiven Infrarot Sensor:
Der PIR-Sensor (Passive Infrared Sensor) ist einer der gängigsten Bewegungsmelder und ist oftmals auch in bewegungssensitiven Außenleuchten oder Alarmanlagen verbaut. Erkennbar ist der PIR-Sensor an seiner meist runden, milchigen Kuppel, die mit vielen einzelnen Waben versehen ist. Der Sensor reagiert auf Temperaturveränderungen in seinem Sichtfeld. Somit können Menschen oder Tiere im Aktionsradius des Sensors erkannt werden. Jedoch kann der Sensor nur Veränderungen wahrnehmen. Bleibt ein Objekt ruhig im Bereich des Sensors stehen, so wird es nicht weiter erkannt. Sobald es sich weiterbewegt, schlägt der Sensor erneut an.[3, S. 493]

Im inneren eines solchen Moduls befinden sich zwei Folien, die an ihrer Oberfläche unterschiedliche elektrische Ladungen aufweisen. Trifft nun die Wärmestrahlung eines bestimmten Frequenzbereichs auf diese Folien, wird deren Polarisation verschoben und eine elektronische Spannung erzeugt, welche den Sensor zum auslösen bringt. Die auf den Sensor aufgesetzte ”milchige Kuppel” erweitert den Erfassungsbereich des Sensors, indem es wie eine Anordnung von Linsen fungiert und lenkt die Wärmestrahlung direkt auf eine der beiden Folien. Bewegt sich nun eine Wärmequelle durch den vom Sensor überwachten Raum, kann man eine Bewegung über einen großen Bereich und in relativ großer Entfernung registrieren.[2]Um den PIR Sensor am Raspberry Pi anzuschließen, werden insgesamt 3 GPIO Anschlüsse benötigt. Einen 3,3V Anschluss für die Stromversorgung, einen Ground und ein frei programmierbarer GPIO Pin, um den PIR-Sensor zu steuern.

freies zitieren einfügen

III. Unser Ansatz

In diesem Abschnitt, befassen wir uns damit, wie wir mit Codis eine kooperative, verteilte Überwachung eines Gebiets auf Eindringlinge realisieren. Wir beschäftigen uns mit den Fragen, wie wir überhaupt einen Eindringling erkennen und wie Codis als verteiltes System seine Genauigkeit beim Erkennen von Eindringlingen erhöht. Dabei befassen wir uns damit, wie aus Codis ein verteiltes System entsteht.

A. Erkennung eines möglichen Eindringlings

1) Erkennen von Bewegungen:
2) Entdecken eines Eindringlings:

B. Kooperative, verteilte Überwachung

Codis' Hauptmerkmal ist die Überwachung eines Gebiets mit mehreren RasPis, die in einem Netzwerk miteinander kommunizieren und dadurch zuverlässiger arbeiten sollen. Wir verwenden im Folgenden den Begriff Codis-Netzwerk für die Menge aller RasPis, auf denen Codis läuft und die miteinander kommunizieren. Als Koordinator bezeichnen wir ein RasPi im Codis-Netzwerk, der ausgewählt wurde, spezielle Aufgaben zu übernehmen, die wir im Laufe dieses Unterkapitels klären. Der Koordinator ist allen RasPis im Codis-Netzwerk bekannt und wird innerhalb des Codis-Netzwerks ausgewählt. Wodurch das Codis-Netzwerk erzeugt wird, wie der Koordinator ausgewählt wird und wie wir eine kooperative, verteilte Überwachung realisieren, klären wir in den nächsten Unterabschnitten.

1) Verteilte Liste: Codis verwendet eine verteilte Liste der IPv4-Adressen aller RasPis im Codis-Netzwerk. Als verteilte Liste bezeichnen wir eine Liste, die auf allen Geräten eines verteilten Systems lokal abgespeichert ist und stets redundant zu den lokal abgespeicherten Listen der jeweils anderen Geräten ist. Die verteilte Liste, die Codis verwendet, bezeichnen wir im Folgenden als Codis-Liste. Die Codis-Liste wird zu Koordinationszwecken zwischen den RasPis im Codis-Netzwerk benötigt. Die Codis-Liste wird zum ersten Mal dann gebildet, wenn das Codis-Netzwerk erzeugt wird. Möchte ein RasPi dem Codis-Netzwerk beitreten, sendet dieser eine JOINREQUEST Nachricht an das Codis-Netzwerk. Daraufhin wartet der RasPi N Sekunden lang auf eine JOINRESPONSE Nachricht vom Koordinator. Die JOINRESPONSE Nachricht enthält die Codis-Liste und die Position des Koordinators in der Codis-Liste. Erhält der RasPi nach N Sekunden keine JOINRESPONSE Nachricht, trägt er sich als Erster in die Codis-Liste ein und ist somit auch der Koordinator im Codis-Netzwerk. Bekommt der RasPi eine JOINRESPONSE Nachricht, aktualisiert er seine Codis-Liste anhand der Nachricht, trägt sich ans Ende der Codis-Liste ein und sendet eine JOIN Nachricht an alle Geräte im Codis-Netzwerk. Erhält ein RasPi eine JOIN Nachricht, trägt er den Absender der JOIN Nachricht ans Ende seiner Codis-Liste ein.

2) Wahl eines Koordinators:

3) Abwechselnde Überwachung: Bei der verteilten wechselseitigen Überwachung, wird ein einzigartiges Token an einem RasPi zugeteilt. Der RasPi dem das Token zugeteilt ist, gilt als aktiv und überwacht als einziger das Gebiet für ein Zeitinterval X. Währenddessen gelten alle anderen RasPis als inaktiv. Inaktive RasPis haben ihre Sensoren abgeschaltet und warten auf eine eingehende Nachricht, die ihnen das Token übergibt. Entdeckt der aktive RasPi einen möglichen Eindringling, dann sendet dieser eine Nachricht an alle anderen RasPis. Empfängt ein inaktiver RasPi diese Nachricht, geht

dieser in einen Alarmzustand über. Im Alarmzustand sind die Sensoren des RasPis aktiviert. Entdeckt ein RasPi im Alarmzustand für N Minuten keine Bewegung, dann geht dieser wieder in den inaktiven Zustand. Erhält ein RasPi im Alarmzustand das Token, geht dieser in den aktiven Zustand über. Während sich ein RasPi im aktiven Zustand befindet, kann sich dieser auch im Alarmzustand befinden. Dieser Fall trifft ein, wenn der RasPi selbst einen möglichen Eindringling entdeckt. Das verhindert, dass der RasPi in den inaktiven Zustand geht, während er einen möglichen Eindringling entdeckt hat und das Token weitergibt. Wir verwenden für das Verteilen des Tokens den wechselseitigen Ausschluss. Wir realisieren den wechselseitigen Ausschluss durch einen Ringalgorithmus. Für den Ringalgorithmus nutzt jeder RasPi die Liste, die alle

IV. Andere Ansätze (bzw. Verwandte Arbeiten)

Zu Beginn des Projekts haben wir als Grundlage für die Bearbeitung das Programm Motion[4] als passend angesehen. Mit Motion ist es möglich, seinen (Linux-)Rechner in eine Überwachungskamera mit vielen zusätzlichen Funktionen zu verwandeln und war aus diesem Grund für uns von Interesse. Die Konfigurationsmöglichkeiten sind sehr umfangreich und die Bewegungserkennung in der Software hat auf Wunsch Videos und/oder Fotos aufgezeichnet, sobald der Algorithmus eine Bewegung wahrgenommen hat. Bei der Aufzeichnung von Videos können die Auflösung und die Framerate selber bestimmt werden, bei Fotos kann ebenfalls die Auflösung an eigene Bedürfnisse angepasst werden. Auf Wunsch besteht auch die Möglichkeit, sich einen http-Stream zur Verfügung stellen zu lassen, mit welchem das aktuelle Geschehen vor der Kamera auch aus der Entfernung zu beobachten möglich ist. Auf dem Rechner, auf welchem Motion ausgeführt wird, verzichtet es auf eine grafische Oberfläche und kann als Daemon im Hintergrund laufen. Also muss man auch als lokaler Anwender auf den http-Stream zugreifen, um das aktuelle Bild abzugreifen. Da Motion mit handelsüblichen USB-Webcams in Betrieb genommen werden kann, konnten wir das Programm sofort erproben:
Wir haben die Softwarepakete zu Motion also auf den Raspberry Pi installiert, eine USB-Webcam angeschlossen und konnten die Bewegungserkennung testen. Um eine ausreichende Qualität der bei Bewegung automatisch erstellten Fotos und Videos zu erreichen, musste die Auflösung hoch genug sein, was zu ersten Problemen führte: Der Raspberry Pi war mit der Speicherung der Daten sehr ausgelastet und auch der Algorithmus zur der Bewegungserkennung forderte bei entsprechend hoher Auflösung, und damit einhergehender Genauigkeit, seinen Tribut. Das System wäre dauerhaft bei einer CPU-Auslastung zwischen 80 - 100%. Zu diesem Zeitpunkt betrachteten wir dies als notwendiges übel und fuhren trotzdem fort. Um den Anforderungen unseres Projekts gerecht zu werden, auch bei Nacht Bewegungen zu erkennen, musste allerdings ein anderes Kamerasystem genutzt werden. Das bereits vorgestellte Raspberry Pi Kameramodul wurde diesen Anforderungen gerecht, allerdings unterstützt Motion, wie erwähnt, nur USB-Kameras. In der Raspberry Pi Community war man sich dessen bereits bewusst, weshalb eine modifizierte Version von Motion existiert, mit welcher auch das Raspberry Pi Kamera Modul genutzt werden kann. Auch hier gab es leider Probleme mit der Performance, ein anderes Problem führte allerdings erst dazu, dass wir die Arbeiten mit Motion letztlich eingestellt haben: Sowohl vor als auch nach Modifizierung des Quellcodes von Motion gab es bei der Kompilierung viele Systemseitige Fehler, welche auch nach viel investierter Zeit nicht vollständig auszumerzen waren. Die Konfigurationsdatei zu verändern, mit welcher z.B. die Auflösung und der http-Stream eingestellt werden konnten, half uns bei dem Ziel nicht weiter, ein verteiltes Überwachungssystem zu erstellen. Das Ziel, ein verteiltes Überwachungssystem zu erstellen, war nicht erreichbar ohne den Quellcode auch dahingehend anzupassen.

V. Evaluation

Dennis Baustelle

VI. Zusammenfassung

Tottis Baustelle

References

[1] Oct. 2014. URL: https : / / github . com / waveform80 / picamera/blob/master/docs/recipes2.rst.

[2] *Elektronikwissen zu: PIR-Sensor - Aufbau und Funktion*. URL: http://www.elv.de/controller.aspx?cid=758&detail=10&detail2=129.

[3] Michael Kofler. *Raspberry Pi : das umfassende Handbuch ; [Grundlagen verstehen, spannende Projekte realisieren ; Schnittstellen des Pi, Schaltungsaufbau, Steuerung mit Python ; Erweiterungen für den Pi: Gertboard, PiFace, Quick2Wire u.a. in Hardware-Projekte einsetzen ; aktuell zu allen Versionen, inkl. Modell B+]*. Bonn : Galileo Press, 2014.

[4] Lavrsen, Kenneth. *Motion - Web Home*. URL: http://www.lavrsen.dk/foswiki/bin/view/Motion/WebHome.

[5] Mirko Lindner. *Über 3,8 Millionen Raspberry Pi verkauft*. Oct. 2014. URL: http://www.pro-linux.de/news/1/21613/ueber-38-millionen-raspberry-pi-verkauft.html.