Kooperative, verteilte Überwachung eines Gebiets auf Eindringlinge

Dennis Lisiecki, Torsten Kühl

I. Einleitung

Eine Überwachungskamera dient im allgemeinem dem Zweck, einen bestimmten Bereich dauerhaft zu überwachen. Das Bild wird kann direkt auf einem Monitor wiedergegeben und parallel aufgezeichnet werden, in Supermärkten erfolgt die Ausstrahlung des Live-Bilds sogar zur Abschreckung oftmals direkt im Verkaufsraum. Zum Schutz vor Vandalismus oder ähnlichem steigt auch im privaten Bereich die Verbreitung von Überwachungskameras. Dabei tendiert der Markt zu immer kleineren Modellen, mit höheren Auflösungen. Geräte solcher Art gibt es bereits zu Genüge, weswegen unser Projekt sich auf einen anderen Ansatz konzentriert.
Unser Ziel war es, ein System für eine kooperative, verteilte Überwachung mit mehreren Raspberry Pis zu realiseren. Die Raspberry Pis verfügen jeweils über eine Kamera und einen Bewegungsmelder den PIR-Sensor (Passive Infrared Sensor). Die Geräte sollen dazu an unterschiedlichen Stellen platziert werden, die den zu überwachenden Raum aus unterschiedlichen Blickwinkeln beobachten. Die Kamera des jeweiligen Geräts soll nun auf jegliche wahrgenommene Bewegung reagieren. Um die Genauigkeit zu erhöhen, soll die Kamera außerdem mit dem PIR-Sensor zusammenarbeiten.
Der PIR-Sensor reagiert lediglich auf Objekte, die eine Wärmesignatur ausstrahlen. Wenn nun durch die Kamera und den PIR-Sensor eine Bewegung erkannt wird, sendet der Raspberry Pi per Broadcast eine Nachricht in das Netzwerk, welche von dem Anwender für jegliche weiteren Schritte verwendet werden kann. Die Wahrscheinlichkeit, dass es sich um einen Mensch oder ein Tier handelt wäre in diesem Fall sehr hoch.
Um eine noch höhere Genauigkeit zu erzielen, sollen die Geräte bei unserem Ansatz das gleiche Gebiet aus verschiedenen Blickwinkeln beobachten:
Stellen mehrere Raspberry Pis eine Bewegung fest, wird der Anwender darauf hingewiesen, dass ein Eindringling das Gebiet betreten hat. Wie das funktioniert und in welcher Art und Weise die Systeme miteinander kommunizieren, wird im Folgendem erklärt.

II. System

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 z.B. den meisten Handy-Ladegeräten kann die Stromversorgung auch über Batterie und Solarzelle erfolgen. So kann der Raspberry Pi auch mobil verwendet werden. Bis dato konnte sich der Raspberry Pi knapp 4 Millionen mal verkaufen und ist inzwischen in seiner vierten Version erschienen. [7] 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. Das aktuellste Modell ist der Raspberry Pi 2 B (stand 05. Februar 2015). Der Raspberry Pi 2 B verfügt über 1.024 MB RAM, vier Cortex-A7-Kerne mit 900 MHz und soll den gleichen Preis kosten, wie der Raspberry Pi B+. Alle Modelle verwenden eine Micro-SD-Karten als Speichermedium. Für unsere Prototypen verwenden wir den Raspberry Pi B+.

B. Raspberry Pi Kameramodul

Für unsere Ausarbeitung 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. *Das Camera Board und die PiNoIR sind kleine nackte Digitalkamera-Boards, die speziell für die Verwendung mit dem Raspberry Pi entwickelt wurden [...] . Das Camera Board kam als erstes auf den Markt und ist aufgrund des einfachen Anschlusses, der sehr leichten Bedienung und den vielfältigen Einsatzmöglichkeiten ein echtes Muss für Hobbybastler. Einige Zeit später wurde ein zweites Modell der Kamera entwickelt, das den Namen PiNoIR trägt. In der PiNoIR ist der gleiche Bildsensor wie in ihrem Vorgänger verbaut,und sie unterscheidet sich auch von der äußeren Form her nicht. Der PiNoIR fehlt lediglich der Infrarotfilter in der Kameralinse. Damit ist es möglich, infrarotes Licht zu erfassen. Eine normale Fernbedienung wird im Bild der PiNoIR plötzlich zur Taschenlampe, die einen dunklen Raum ausleuchtet.*[5, S. 511] 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. Unser Prototypen verfügen über keine Infrarotlichtquellen, können jedoch um diese Funktion erweitert werden.

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 an 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.[5, 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 milchige Kuppel auf dem Sensor 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 Bereich von weniger als 120 Grad und einer Entfernung von höchstens sieben Metern feststellen. Um den PIR Sensor am Raspberry Pi anzuschließen, werden insgesamt 3 GPIO Anschlüsse benötigt. Einen 5V Anschluss für die Stromversorgung, einen Ground und ein frei programmierbarer GPIO Pin, um den PIR-Sensor zu steuern. Die Ausgangsspannung des PIR-Sensors beträgt 3,3V und kann somit vom Raspberry Pi ohne elektronischen Widerstand dazwischen verarbeitet werden.

III. Codis: Kooperative, verteilte Überwachung

Um eine kooperative, verteilte Überwachung zu realisieren, haben wir das Programm Codis entwickelt. Codis steht für cooperative, distributed surveillance[1]. Entdeckt Codis einen Eindringling, sendet es eine INTRUDER Nachricht an alle Geräte im Netzwerk. Wie diese Nachricht verarbeitet wird und welche Maßnahmen eingeleitet werden sollen, wenn ein Eindringling entdeckt wurde, ist den Anwendern von Codis überlassen. In diesem Abschnitt befassen wir uns damit, wie Codis als verteiltes System ein Gebiet auf Eindringlinge überwacht. Dazu klären wir, wie Codis einen Eindringling entdeckt und wie aus Codis ein verteiltes System entsteht. Anschließend stellen wir fest, wie Codis als verteiltes System seine Effizienz verbessert und seine Genauigkeit beim Entdecken von Eindringlingen erhöht. Wir verwenden im Nachfolgenden den Begriff Sensoren für die Kamera und den PIR-Sensor eines Raspberry Pis.

A. Bewegungserkennung mit der Kamera

Der Raspberry verwendet einen H264 Kodierer zur Videokompression. Um Videos zu kodieren, verwendet der Kodierer ein Verfahren, das als Motion Estimation[2][8] bezeichnet wird. Dabei wird das Bild in 16x16 Pixel große Quadrate eingeteilt. Diese Quadrate bezeichnet man als Makroblöcke. Beim Kodieren von Videos vergleicht der H264 Kodierer das aktuelle Bild mit einem Referenzbild. Dazu untersucht der Kodierer jeden einzelnen Makroblock im aktuellen Bild und sucht nach dem ähnlichsten Makroblock im Referenzbild. Der jeweilige Abstand zwischen den Makroblock im aktuellen Bild und im Referenzbild wird vom Kodierer gespeichert und gilt auch als Ausmaß einer Bewegung. Anhand dieses Abstands kann gemessen werden, wie sehr sich ein Makroblock im Bild gegenüber dem Referenzbild bewegt hat. Bis zu diesem Schritt wurde der komplette Vorgang im H264 Kodierer implementiert, um Bewegungen zu erkennen.[10] Als nächstes nehmen wir uns für jedes Bild die Bewegungsdaten als Array und berechnen damit das Ausmaß an Bewegung, die im Bild stattfindet. Dazu berechnen wir das Ausmaß der einzelnen Vektoren im Array mit dem Satz des Pythagoras'. Eine Bewegung wurde dann erkannt, wenn im Array mindestens zehn Vektoren vorhanden sind, deren Bewegungsausmaß mindestens 60 beträgt.[4]

B. Codis als verteiltes System

Codis' Hauptfunktion ist es, ein Gebiet aus verschiedenen Blickwinkeln zu überwachen. Dieser Ansatz soll die Fehlertoleranz beim Erkennen von möglichen Eindringlingen verbessern. Dazu verwendet Codis mehrere Raspberry Pis, die sich in einem Netzwerk befinden und untereinander Nachrichten

[1] zu Deutsch: Kooperative, verteilte Überwachung
[2] wortwörtlich: Bewegungsvorhersage

austauschen. Wir verwenden im Folgenden den Begriff Codis-System für die Menge aller Raspberry Pis, auf denen Codis läuft und die miteinander in einem Netzwerk kommunizieren. Als Koordinator bezeichnen wir einen Raspberry Pi im Codis-System, der spezielle Aufgaben übernimmt, die wir im Laufe dieses Abschnittes klären. Der Koordinator ist allen Raspberry Pis im Codis-System bekannt und wird innerhalb des Codis-Systems ausgewählt. Das Codis-System wird dann erzeugt, wenn ein Raspberry Pi Codis ausführt, während kein anderer Raspberry Pi im Netzwerk Codis ausführt und wird dann zerstört, wenn Codis vom letzten Raspberry Pi im Codis-System beendet wird. Um die Implementierung des Prototypen einfach zu halten, verwendet Codis das UDP-Protokoll.

1) Codis-Liste: Codis verwendet eine verteilte Liste der Netzwerkadressen aller Raspberry Pis im Codis-System. 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 Raspberry Pis im Codis-System benötigt. Die Codis-Liste wird dann gebildet, wenn das Codis-System erzeugt wird. Möchte ein Raspberry Pi dem Codis-System beitreten, sendet dieser eine JOINREQUEST Nachricht an das Codis-System. Daraufhin horcht der Raspberry Pi fünf Sekunden lang auf eine JOINRESPONSE Nachricht, die von allen Raspberry Pis im Codis-System versendet wird. Die JOINRESPONSE Nachricht enthält die Position des Absenders in der Codis-Liste. Nachdem der Koordinator seine JOINRESPONSE Nachricht versendet hat, wartet er zwei Sekunden und sendet daraufhin eine COORDINATOR Nachricht an den Raspberry Pi, der dem Codis-System beitreten möchte. Die COORDINATOR Nachricht, enthält die Position des Koordinators in der Codis-Liste. Erhält der Raspberry Pi die COORDINATOR Nachricht, dann trägt er ein, welche Position der Koordinator in der Codis-Liste hat. Erhält der Raspberry Pi nach fünf Sekunden keine JOINRESPONSE Nachricht, trägt er sich als Erster in die Codis-Liste ein und ist somit auch der Koordinator im Codis-System. Hat der Raspberry Pi von allen anderen Raspberry Pis im Codis-System eine JOINRESPONSE Nachricht erhalten, trägt er sich ans Ende der Codis-Liste ein und sendet eine JOIN Nachricht an alle Geräte im Codis-System. Erhält ein Raspberry Pi eine JOIN Nachricht, trägt er den Absender der JOIN Nachricht ans Ende seiner Codis-Liste ein.

2) Wahl eines Koordinators: Der Raspberry Pi von dem aus das Codis-System erzeugt wurde, wird als erster Koordinator ausgewählt. Betreten weitere Raspberry Pis das Codis-System, wird ein modifizierter Ringalgorithmus[9][1] ausgeführt, der in einem Intervall von 15 Minuten einen neuen Koordinator auswählt. Der Ringalgorithmus, den wir für Codis modifiziert haben, wird in der Literatur folgendermaßen beschrieben: *Ein weiterer Wahl-Algorithmus basiert auf der Verwendung eines Rings. Anders als einige andere Ring-Algorithmen verwendet dieser kein Token. Wir setzen voraus, dass die Prozesse physisch oder logisch geordnet sind, sodass jeder Prozess weiß, wer sein Nachfolger ist. Wenn ein Prozess erkennt, dass der Koordinator nicht funktioniert, erzeugt er eine ELECTION-Nachricht mit seiner eigenen Prozessnummer und sendet die Nachricht an seinen Nachfolger. Ist der Nachfolger nicht aktiv, überspringt das Senden diese und geht zum nächsten Mitglied entlang des Rings, oder auf den wiederum nachfolgenden, bis ein aktiver Prozess gefunden ist. Bei jedem Schritt trägt der Sender seine eigene Prozessnummer in die Liste der Nachrichten ein, wodurch er sich selbst zu einem Kandidaten macht, der als Koordinator gewählt werden kann. Irgendwann gelangt die Nachricht zurück zu dem Prozess, der das Ganze gestartet hat. Dieser Prozess erkennt dieses Ereignis, wenn er eine einkommende Nachricht empfängt, die seine eigene Prozessnummer enthält. Jetzt wird der Nachrichtentyp in COORDINATOR geändert und noch einmal gesendet, um jeden darüber zu informieren, wer der Koordinator ist (das Listenelement mit der höchsten Nummer) und wer die Mitglieder des neuen Rings sind. Nachdem diese Nachricht einmal den Umlauf gemacht hat, wird sie entfernt, und jeder geht wieder an die Arbeit*[9, S. 300] Um einen logischen Ring darzustellen, verwendet Codis die Codis-Liste. Die Raspberry Pis im Codis-System sind anhand ihrer Position in der Codis-Liste aufsteigend, im Ring angeordnet. Wird ein neuer Koordinator ausgewählt, verschickt der derzeitige Koordinator eine ELECTION Nachricht, an seinen Nachfolger im Ring. Erhält ein Raspberry Pi eine ELECTION Nachricht, dann trägt er sich als neuer Koordinator ein und sendet eine COORDINATOR Nachricht an alle Raspberry Pis im Codis-System. Empfängt der Raspberry Pi, der die ELECTION Nachricht gesendet hat, nach fünf Sekunden keine COORDINATOR Nachricht, sendet er eine HEARTBEATREQUEST an seinen Nachfolger. Erhält er daraufhin eine HEARTBEATRESPONSE Nachricht zurück, sendet er die ELECTION Nachricht erneut an seinen Nachfolger. Empfängt er dagegen keine HEARTBEATRESPONSE von seinem Nachfolger, dann entfernt er diesen aus der Codis-Liste und sendet eine LISTUPDATE Nachricht, an das Codis-System, damit der Nachfolger aus der Codis-Liste entfernt wird. Danach versucht der Raspberry Pi die ELECTION Nachricht, an seinen nächsten Nachfolger zu senden.

3) Abwechselnde Überwachung: Unter abwechselnder Überwachung verstehen wir, dass nur der Koordinator das Gebiet überwacht, bis der nächste Koordinator gewählt wurde. Währenddessen gelten alle anderen Raspberry Pis als inaktiv. Inaktive Raspberry Pis haben ihre Sensoren abgeschaltet und warten auf eine ELECTION Nachricht. Entdeckt der Koordinator einen möglichen Eindringling, dann sendet dieser eine INTRUDER Nachricht an alle anderen Raspberry Pis und geht in einen Alarmzustand über. Empfängt ein inaktiver Raspberry Pi diese Nachricht, geht dieser auch in einen Alarmzustand über. Im Alarmzustand sind die Sensoren des Raspberry Pis aktiviert. Entdeckt ein Raspberry Pi im Alarmzustand für fünf Minuten keinen Eindringling, dann geht er wieder in den inaktiven Zustand, außer der Koordinator der den Alarmzustand lediglich verlässt. Wird ein Raspberry Pi im Alarmzustand zum Koordinator, verbleibt er im Alarmzustand. Wird ein neuer Koordinator gewählt, verbleibt der vorherige Koordinator im Alarmzustand, falls er sich in diesem bereits befindet. Das verhindert, dass der vorherige Koordinator inaktiv wird, wenn sein Nachfolger zum Koordinator wird, während ein Eindringling entdeckt wird.

C. Entdecken eines Eindringlings

Entdeckt Codis in mehreren Bildern für einen kurzen Abstand eine Bewegung, setzt es ein MOTION Flag auf WAHR. Das MOTION Flag wird nach 50 Bildern, in denen keine Bewegung erkannt wurde, auf FALSCH gesetzt. Ein möglicher Eindringling wird dann erkannt, wenn der PIR-Sensor Bewegungen dann erkennt, während das MOTION Flag auf WAHR gesetzt ist. Entdeckt ein Raspberry Pi im Codis-System einen möglichen Eindringling, dann sendet dieser eine INTRUDER Nachricht an den Koordinator. Empfängt der Koordinator eine INTRUDER Nachricht, merkt er sich, von welchem Raspberry Pi diese Nachricht versendet wurde und wann die Nachricht versendet wurde. Entdeckt der Koordinator einen möglichen Eindringling, dann prüft er, ob ein anderer Raspberry Pi in den letzten drei Sekunden einen möglichen Eindringling entdeckt hat. Ist das der Fall, sendet der Koordinator eine INTRUDER Nachricht ans Netzwerk.

IV. Andere Ansätze (bzw. Verwandte Arbeiten)

Eine Realisierung einer kooperativen, verteilten Überwachung mit dem Raspberry Pi konnte bei unserer Recherche nicht gefunden werden. Es existiert jedoch eine Lösung, ein Linux-System als pure Überwachungskamera nutzbar zu machen. Für diese Lösung wird das C-Programm Motion verwendet[6], das wir uns für unsere Ausarbeitung angesehen haben, da es im Kern am ehesten mit unserer Ausarbeitung vergleichbar ist und die am weitesten verbreitete Open Source Lösung dieser Art darstellt.
Motion verfolgt das Ziel, aus dem Computer mit angeschlossener USB-Webcam eine Überwachungskamera zu erstellen. Der Bewegungserkennungsalgorithmus von Motion ist der gleiche, der im H264 Kodierer des Raspberry Pis bereits implementiert wurde. Das heißt, nutzt man Motion auf dem Raspberry Pi um Bewegungen zu erkennen, führt Motion den Bewegungserkennungsalgorithmus aus, obwohl dies bereits vom H264 Kodierer des Raspberry Pis erledigt wurde. Eine erkannte Bewegung führt bei Motion dazu, dass wahlweise ein Foto oder Video aufgezeichnet. Ebenso kann der aktuelle Video-Stream der Kamera abgegriffen werden, um das aktuelle Geschehen selbst zu beobachten. Motion kann von einem Anwender auch so eingestellt werden, dass ein Quadrat um eine Bewegung im Kamerabild gezeichnet wird.
Motion lässt sich im Originalzustand nicht mit dem Raspberry Pi Kameramodul betreiben, es gibt allerdings Projekte, die mit einer modifizierten Version von Motion das Raspberry Pi Kameramodul nutzen. Motion lässt sich durch den Anschluss mehrerer Kameras erweitern, wobei diese sich untereinander in ihrer Funktionsweise nicht beeinflussen. Der Ansatz mit Motion war für uns dahingehend uninteressant, weil durch Motion keine kooperative, verteilte Überwachung realisiert wird. Außerdem bietet der Ansatz mit Motion nicht die Möglichkeit, auf einen Eindringling zu reagieren, wenn dieser entdeckt wurde. In der von uns verwendeten Literatur, findet sich ein Beispielprojekt einer Überwachung, die mit Motion realisiert wurde. Bei diesem Beispielprojekt, wird der Ansatz mit Motion dazu verwendet, das Innere eines Vogelhaus zu observieren.[5]

V. Evaluation

Dieser Abschnitt fasst zusammen, welche Vorteile Codis gegenüber nicht-verteilten Überwachungssystemen hat. Außerdem setzen wir uns damit auseinander, ob Codis bestimmte Eigenschaften eines verteilten Systems erfüllt. Aber zuallererst stellen wir uns die Frage, ist Codis überhaupt ein verteiltes System? Eine Definition von verteilten Systemen lautet: *Bei einem verteilten System arbeiten Komponenten zusammen, die sich auf vernetzten Computern befinden und die ihre Aktionen durch den Austausch von Nachrichten koordinieren. Aus dieser Definition leiten sich die folgenden Eigenschaften verteilter Systeme ab: Nebenläufigkeit der Komponenten, es gibt keine globale Uhr und die Komponenten können unabhängig voneinander ausfallen.*[1, S. 17] Die einzelnen Raspberry Pis im Codis-System arbeiten nebeneinander. Codis verfügt auch über keine globale Uhr, sondern nutzt jeweils die lokalen Uhren der einzelnen Raspberry Pis. Ferner können einzelne Raspberry Pis ausfallen, während Codis ein Gebiet überwacht.

A. CPU-Auslastung als verteiltes System

Codis bietet als verteiltes System eine bessere CPU-Auslastung gegenüber nicht-verteilten Überwachungssysteme. Das ermöglicht die abwechselnden Überwachung von Codis. Während bei einem nicht-verteilten Überwachungssystem ein Raspberry Pi das Gebiet kontinuierlich überwacht, ist bei einer verteilten Überwachung nur der Koordinator des Codis-Systems aktiv. Überwacht ein Raspberry Pi mit Codis das Gebiet, beansprucht er ungefähr 30 Prozent an CPU Leistung, ohne einen Eindringling zu entdecken. Das heißt, ohne der abwechselnden Überwachung wäre die CPU jedes Raspberry Pis im Codis-System dauerhaft für 30 Prozent ausgelastet. Durch die abwechselnde Überwachung, teilen sich die Raspberry Pis die CPU-Last auf, indem nur einer für ein Intervall von 15 Minuten das Gebiet überwacht, bis ein Eindringling entdeckt wurde und daraufhin alle Raspberry Pis im Gebiet aktiv werden. Ein Vorteil ist der geringere Stromverbrauch durch die bessere CPU-Auslast. Die Raspberry Pis, die mit Codis ein Gebiet im Freien überwachen, müssten mit externen Akkus betrieben werden. Durch die abwechselnde Überwachung wird der Akku eines Raspberry Pis weniger beansprucht.

B. Evaluation der Ausfalltoleranz von Codis

Codis hat als verteiltes System den Vorteil, dass das Codis weiterhin funktioniert, wenn darin einzelne Raspberry Pis ausfallen. Codis fällt erst dann aus, wenn alle Raspberry Pis im Codis-System ausgefallen sind. Codis wurde so entwickelt, dass es sich selbst reorganisieren kann, wenn Raspberry Pis ausfallen. Das bedeutet, wenn z. B. ein Raspberry Pi unerwartet abstürzt, wird das vom Codis-System erkannt und der entsprechende Raspberry Pi wird aus der Codis-Liste entfernt. Dazu nutzt Codis ein Heartbeat-System, das vom Web Framework Vaadin eingesetzt wir, um zu erkennen, ob eine Browsersitzung erloschen ist. *UI instances are cleaned up if no communication is received from them after some time. If no other server requests are made, the client-side sends keep-alive heartbeat requests. A UI is kept alive for as long as*

requests or heartbeats are received from it. It expires if three consecutive heartbeats are missed.[3, S. 95] In Codis erhält jeder Raspberry Pi von seinem Nachfolger im Ring in einem Interval von zwei Minuten eine HEARTBEAT Nachricht. Bleiben drei HEARTBEAT Nachrichten nacheinander aus, entfernt der Raspberry Pi seinen Nachfolger vom Codis-System und informiert darüber alle anderen Raspberry Pis im Codis-System, damit diese das Gleiche tun. Die HEARTBEAT Nachricht empfängt nicht speziell der Koordinator, damit Codis auch mit dem des Koordinators leicht umgehen kann. Der schlimmste Fall bei einem Ausfall des Koordinators wäre, wenn der Koordinator dann ausfällt, kurz nachdem dieser eine HEARTBEAT Nachricht gesendet hat. In diesem Fall könnte Codis einen Anwender für sechs Minuten lang nicht darüber informieren, ob ein Eindringling das Gebiet betreten hat. Sollten bis auf ein Raspberry Pi alle anderen im Codis-System ausfallen, informiert der noch vorhandene Raspberry Pi einen Anwender über Eindringlinge, obwohl mindestens zwei Raspberry Pis im Codis-System einen Eindringling entdecken müssten. Das passiert aber nur dann, wenn der letzte Raspberry Pi im Codis-System der Koordinator ist. Das ermöglicht auch, Codis als nicht-verteiltes System zu nutzen. Sämtliche Mechanismen für eine verteilte Überwachung, werden dann nicht mehr von Codis ausgeführt.

C. Evaluation der Fehlertoleranz Codis

Das Problem mit Ansätzen, die Programme wie Motion verwenden, ist, dass diese Überwachungssysteme nicht fehlertolerant ist. Möchte man ein Gebiet überwachen, in dem es windig ist und Dinge wie z. B. Blätter umhergewirbelt werden, dann kann es passieren, dass oft Bewegungen erfasst werden, die aber von keinem Eindringling verursacht werden. Ein Raspberry Pi in Codis hat dagegen den Vorteil, dass Eindringlinge nur entdeckt werden, wenn Kamera und PIR-Sensor ungefähr gleichzeitig Bewegungen erkennen. Der PIR-Sensor erkennt dabei nur Bewegungen, die von einer Wärmequelle verursacht werden.[5] Dadurch dass Codis als verteiltes System Gebiete aus verschiedenen Blickwinkeln überwacht, ist Codis präziser darin, einen Eindringling zu entdecken, gegenüber nicht-verteilten Überwachungssystemen. Das Ergebnis ist, dass die Wahrscheinlichkeit für Fehlalarm bei Codis geringer ist.

VI. Zusammenfassung

Durch seine kompakte Bauform und der frei programmierbaren GPIO-Schnittstelle ist der Raspberry Pi für viele Anwendungsbereiche wie geschaffen. Er kann völlig kabellos betrieben werden und findet Platz in den kleinsten Ecken. So ist es nichts ungewöhnliches, dass das System auch als Überwachungskamera eingesetzt wird. Dabei existiert jedoch kein Ansatz, der in diesem Zusammenhang auch eine kooperative, verteilte Überwachung mit dem Raspberry Pi unterstützt. In unserer Ausarbeitung haben wir uns diesem Umstand angenommen. Unsere Motivation bestand darin, das Vorhandensein mehrerer Raspberry Pis möglichst effektiv auszunutzen. Mit Codis besteht die Möglichkeit, ein leicht zu erweiterndes Netzwerk zur Überwachung zu errichten, bei welchem sich die Geräte gegenseitig in den Alarmzustand versetzen. Wie der Alarmzustand im Endeffekt genutzt wird, kann vielfältiger Natur sein und steht dem Anwender offen. Die leichte Erweiterbarkeit ist einer der größten Vorteile, der dadurch gegeben ist, dass Codis auf Veränderungen im Codis-System völlig autonom reagiert. Zusätzlich wird sehr sparsam mit den Ressourcen umgegangen, da in einem Codis-System nur ein Koordinator existiert, der alle Sensoren aktiv schaltet. Gerade bei Verwendung mit einem Akku ist der Unterschied enorm. Wir möchten auf eine Android-App hinweisen, die wir zusammen mit dem Codis Prototypen entwickelt haben. Diese Android-App kann die Netzwerknachrichten, die Codis sendet, empfangen und weist somit daraufhin, wenn Codis einen Eindringling entdeckt hat. Diese Android-App lässt sich im Google Play Store unter den Namen Surveillance Receiver kostenlos runterladen.

Literatur

[1] George Coulouris, Jean Dollimore und Tim Kindberg. *Verteilte Systeme : Konzepte und Design*. München : Pearson Studium, 2003.

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

[3] Marko Grönroos. *Book of Vaadin: Vaadin 7 Edition - 2nd Revision*. Vaadin Ltd, 2014.

[4] Hughes, Dave. *Recording motion vector data*. URL: http://picamera.readthedocs.org/en/release-1.8/recipes2.html#recording-motion-vector-data.

[5] 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.

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

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

[8] Shilpa Metkar und Sanjay Talbar. *Motion Estimation Techniques for Digital Video Coding*. Springer India, 2013.

[9] Andrew S. Tanenbaum und Martinus R. van Steen. *Verteilte Systeme : Grundlagen und Paradigmen. Distributed Systems ¡dt.¿* München : Pearson Studium, 2003.

[10] Upton, Liz. *Vectors from coarse motion estimation*. URL: http://www.raspberrypi.org/vectors-from-coarse-motion-estimation.