



Defense University Center Spanish Naval Academy

FINAL YEAR PROJECT

Video Surveillance System based on Raspberry Pi and Pi Camera

Mechanical Engineering Bachelor Degree

STUDENT: Carlos Hernández García del Valle

SUPERVISORS: Belén Barragáns Martínez

Norberto Fernández García

ACADEMIC YEAR: 2015-2016

Universida_{de}Vigo



**Defense University Center
Spanish Naval Academy**

FINAL YEAR PROJECT

Video Surveillance System based on Raspberry Pi and Pi Camera

Mechanical Engineering Bachelor Degree

Naval Technology Specialization

Naval Branch

Universidade de Vigo

Abstract

Given the high cost of current commercial video surveillance systems, this final year project intends to explore, design and implement an alternative video surveillance system using Raspberry Pi and Pi Camera. With the recent proliferation of cheap single board computers they have become easily accessible and have spawned lots of homebrew projects.

Among these projects there are a series of programs designed to be used in a video surveillance system. Having explored the different possibilities each one offers, a video surveillance system was designed using a central server type architecture. Each Raspberry Pi streams a video feed to the central server which then analyzes the video feeds. This analysis allows for motion to be detected and a series of actions to be executed.

Among these actions is the possibility of sending email notifications, recording movies, executing commands, etc. Each program used in the system was researched and tested to find the ideal configuration that offered the highest quality video feeds. This configuration was then implemented in a real test of the system which proved to be a cheap and viable alternative to current commercial systems.

Keywords

Raspberry Pi, Pi Camera, video, surveillance, motion detection

Resumen

El objetivo de este trabajo de fin de grado es el diseño e implementación de un sistema de vídeo-vigilancia basado en Raspberry Pi y Pi Camera.

Los sistemas de vídeo-vigilancia nacen en 1942 durante la segunda guerra mundial. Se crearon para poder monitorizar de manera remota el lanzamiento de los cohetes V-2. Con los avances tecnológicos adquirieron diferentes capacidades. Pasaron de ser un sistema simple, con una sola videocámara a tener multitud de videocámaras con capacidad de grabación, detección de movimiento, reconocimiento facial, aumento de resolución, visión nocturna, etc. Incluso son capaces de ejecutar ciertas acciones (alarmas, notificaciones, etc.) de manera independiente en base a lo que detectan.

Teniendo en cuenta estos avances, se puede aplicar en una gran variedad de entornos con diferentes ventajas y objetivos. Las fuerzas de seguridad del estado, por ejemplo, los utilizan para poder monitorizar y vigilar zonas públicas. También se utilizan para controlar el tráfico y disuadir a los conductores de cometer infracciones, entre otras aplicaciones.

En esencia, proporcionan una manera de vigilar zonas por un coste inferior a lo que supondría contratar un guardia de seguridad para vigilar una zona particular. Su uso, por ello, se extiende a la mayoría de negocios, donde los hurtos supusieron pérdidas de hasta \$32 millones en EEUU en el año 2014. Muchas empresas los utilizan también para monitorizar líneas de producción y maquinaria para así poder detectar posibles fallos.

En el entorno particular y personal ofrecen una herramienta más para la seguridad de los hogares y su uso es cada vez más extendido. Tanto es así que se estima que el mercado de vídeo-vigilancia tendrá un valor de hasta €38.8 billones en el año 2020. Existe por ello, una gran variedad de sistemas de vídeo-vigilancia con diferentes características en el mercado. Sin embargo, suelen ser relativamente caros, con un coste medio de €800 para un sistema de vídeo-vigilancia comercial medio (incluyendo su instalación). También suelen ser difíciles de configurar y requieren asistencia técnica externa para solventar cualquier problema.

Es evidente, que existe un mercado para un sistema de vídeo-vigilancia barato y fácilmente configurable. Un sistema basado en Raspberry Pi podría acaparar este mercado. Raspberry Pi es un ordenador de placa reducida de bajo coste. Sus creadores, la fundación Raspberry Pi, tienen como objetivo facilitar la enseñanza de programación de ordenadores. Existen diferentes modelos que cuestan entre \$5-\$35, que debido a su bajo coste y libertad de configuración, se usan en todo tipo de proyectos, desde un ordenador dedicado a dar de comer automáticamente hasta experimentos en la estación internacional espacial. Existen también, algunos proyectos dedicados a vídeo-vigilancia,

La motivación detrás de este proyecto es diseñar un sistema de vídeo-vigilancia basado en Raspberry Pi que sea barato y a la vez proporcionara capacidades avanzadas. Después del estudio de los sistemas de vídeo-vigilancia actuales, se establecieron como objetivos para este proyecto que el sistema pudiera tener más de una cámara, transmisión en tiempo real de los flujos de video de las cámaras, a la vez que control remoto del sistema desde otro dispositivo. Cada Raspberry Pi sería también portátil, utilizando tanto una batería como fuente de alimentación como Wi-Fi para conectarse a la red.

Como parte de las capacidades avanzadas, podrá también detectar movimiento y tomar acciones de manera automática, como empezar a grabar vídeo, mandar una notificación en forma de email. Finalmente el sistema será capaz de funcionar en la oscuridad utilizando luz infrarroja.

Durante el proyecto se llegó a la conclusión que la CPU de las Raspberry Pi era demasiado limitada para poder soportar la carga de analizar de manera adecuada más de un canal de vídeo, por lo que se adoptó un sistema con un servidor central con una CPU más adecuada para esta tarea.

Una vez estudiados los diferentes sistemas actuales de vídeo-vigilancia basados en Raspberry Pi, se utilizaron dos programas, *RPi-Cam Web Interface* y *MotionEye*. Cada uno tiene características y opciones diferentes que son estudiadas y explicadas en la memoria. El servidor central utiliza *MotionEye* mientras que cada Raspberry Pi utiliza *RPi-Cam Web Interface* para transmitir el vídeo. Esto se debe a que aunque *MotionEye* tiene más opciones avanzadas, se demostró que *RPi-Cam Web Interface* tenía hasta un 200% mayor calidad de imagen en la transmisión de vídeo que *MotionEye*. Por ello, *MotionEye* se utiliza exclusivamente para el análisis del vídeo e interfaz del sistema en el servidor central.

El sistema integra varias opciones avanzadas como detección de movimiento, notificación por email, visión nocturna, grabación de vídeo e imágenes, etc. Sin embargo, se llegó a la conclusión que faltaba la capacidad de apagar y reiniciar cada Raspberry Pi de manera remota desde la interfaz de *MotionEye*. Esto se implementó con una serie de modificaciones al programa *MotionEye* que son detalladas en la memoria.

Se realizaron varias pruebas para medir la calidad de la transmisión de vídeo a fin de determinar la mejor configuración y cómo afectaba la duración de las baterías. Dado que en cada Raspberry Pi se puede utilizar Wi-Fi o Ethernet se realizaron diferentes pruebas para determinar cómo afecta la calidad de la transmisión de vídeo y a la duración de la batería.

Finalmente se realizó una prueba real del sistema en el aula de grado del Centro Universitario de la Defensa en la Escuela Naval Militar. Esta prueba demostró que el sistema funcionaba correctamente y que podía servir como una alternativa a sistemas de vídeo-vigilancia comerciales con un coste mucho menor.

Palabras Clave

Raspberry Pi, Pi Camera, video, vigilancia, detección de movimiento

ACKNOWLEDGMENTS

In first place, I would like to thank both of my tutors, Belén Barragáns Martínez and Norberto Fernández García for their inspiration in doing this project, their constant support, their input and most of all their patience in correcting this project.

Having researched their goals and methods I feel that the Raspberry Pi Foundation must be mentioned and recognized for their non-profit aims and goals, as well as Calin Crisan and silvanmelchior for the development of MotionEye and RPi-Cam Web Interface.

It goes without saying that I'm very thankful to my girlfriend, Merce, for supporting me in everything I do as well as being my daily source of laughter and happiness.

I would also like to thank my father and mother for the effort they invested in my education, for the support they showed when I left University for the first time, and for the backing they gave me when I decided to join the Spanish Navy and pursue a career as a Naval Officer.

Finally I would like to mention both of my cats, Kitty and Blasa for being my “test subjects” in the majority of the motion detection testing phase...

TABLE OF CONTENTS

Table of Contents	1
List of Figures	4
List of Tables.....	8
1 Introduction and objectives	9
1.1 Introduction and motivation.....	9
1.1.1 Background of video surveillance systems.....	9
1.1.2 Government use of video surveillance systems.....	9
1.1.3 Private businesses use of video surveillance systems.....	11
1.1.4 Research purposes	11
1.1.5 Domestic use of video surveillance systems	11
1.1.6 Military use of video surveillance systems.....	11
1.1.7 Conclusion and motivation	12
1.2 Objectives.....	12
1.3 Resources used	12
1.3.1 Hardware.....	12
1.3.2 Software	15
1.4 Structure of the document	15
2 State of the art.....	17
2.1 History of video surveillance systems	17
2.1.1 Introduction.....	17
2.1.2 First CCTV	17
2.1.3 Video cassette recorders	18
2.1.4 CCD (Charge-Coupled Device).....	18
2.1.5 DVR's (Digital Video Recorder).....	18
2.1.6 Nanny Cams.....	18
2.1.7 IP Cameras	19
2.2 Current commercial video surveillance systems.....	19
2.2.1 Introduction.....	19
2.2.2 Capabilities and features.....	19
2.2.3 Examples and discussion of current commercial systems	22
2.2.4 Comparison/Conclusion/Table	23
2.3 Home automation video surveillance systems	23
2.3.1 Introduction.....	23
2.3.2 Withings Home	24

2.3.3 Nest Cam	25
2.3.4 Logi Circle	25
2.3.5 Conclusions of home automation video surveillance systems	26
2.4 Raspberry Pi	26
2.4.1 Introduction.....	26
2.4.2 History	28
2.4.3 Current status and future.....	30
2.5 Current Raspberry Pi video surveillance systems.....	32
2.5.1 Introduction.....	32
2.5.2 Tutorials Raspberrypi.org	32
2.5.3 Picamera python module	33
2.5.4 MotionEye and MotionEyeOS	34
2.5.5 RPi Cam Web Interface	36
2.6 Conclusion	37
3 Development and implementation.....	38
3.1 Description of the architecture of the video surveillance system	38
3.1.1 Introduction and features	38
3.1.2 Design of the video surveillance system	38
3.1.3 Description of the video surveillance system	40
3.2 Description of the options of RPi-Cam Web Interface and MotionEye	42
3.2.1 Introduction.....	42
3.2.2 RPi-Cam Web Interface options	42
3.2.3 MotionEye options.....	45
3.2.4 Modified MotionEye options.....	58
3.3 Description of the selected configuration of the system	59
3.3.1 Introduction.....	59
3.3.2 Improvements and modifications to MotionEye	59
3.3.3 Raspberry Pi: RPi-Cam Web Interface.....	59
3.3.4 Central server: MotionEye.....	62
3.3.5 Network	66
3.4 Description of how to install the necessary programs and the improvements implemented ..	66
3.4.1 Introduction.....	66
3.4.2 How to install Raspbian (Raspberry Pi OS)	66
3.4.3 How to install RPi-Cam Web Interface	69
3.4.4 How to install MotionEye on the central server with Ubuntu	71
3.4.5 How to implement remote reboot and shutdown functionality to MotionEye.	73

3.4.6 How to add a new camera in MotionEye.....	81
4 Experiments and results.....	84
4.1 Performance experiments.....	84
4.1.1 Introduction.....	84
4.1.2 Performance comparison of RPi-Cam Web Interface and MotionEye using different configurations	84
4.1.3 Performance comparison of the video feed using Wi-Fi and Ethernet.....	86
4.1.4 Performance comparison of the video feed when using multiple Raspberry Pi's	86
4.2 Battery life experiments	87
4.2.1 Introduction.....	87
4.2.2 Battery life using different configuration settings	87
4.2.3 Battery life using Wi-Fi and Ethernet.....	88
4.3 Advanced features tests.....	88
4.3.1 Introduction.....	88
4.3.2 Motion detection.....	88
4.3.3 Email notifications.....	89
4.3.4 Night vision	90
4.3.5 Remote reboot and shutdown of Raspberry Pi's from MotionEye.....	91
4.3.6 Security	92
4.4 Real world test	93
4.4.1 Introduction.....	93
4.4.2 Set-up and configuration.....	93
4.4.1 Results.....	97
4.4.2 Conclusions.....	100
5 Conclusions and future work.....	101
5.1 Overall conclusions.....	101
5.2 Future lines of study.....	102
5.3 Personal opinion.....	103
6 Bibliography	104

LIST OF FIGURES

Figure 1-1 Graph representing the decrease in crime in the UK since the use of CCTV cameras (taken from [3]).....	10
Figure 1-2 Schematic of London's street traffic control center (taken from [4])	10
Figure 1-3 Image of Raspberry Pi 2 Model B.....	13
Figure 1-4 Pi Camera with two infrared led light boards	14
Figure 1-5 Edimax wireless USB adapter	14
Figure 1-6 RAVPower battery	15
Figure 2-1 Example of an early CCTV system (taken from [10])	17
Figure 2-2 A Nanny Cam disguised as a teddy bear (taken from [11])	18
Figure 2-3 Comparison of SD resolution to HD resolutions (taken from [13]).....	19
Figure 2-4 Comparison of the same scenario with visible light and thermal imaging (taken from [14])	20
Figure 2-5 An example of a video surveillance system tracking movement (taken from [15])	21
Figure 2-6 Example of how a video surveillance system with remote viewing using the Internet is configured (taken from [16]).....	21
Figure 2-7 Graph showing the growing tendency in the number of gadgets per year (taken from [23])	23
Figure 2-8 Example of home appliances being controlled remotely with a smartphone (taken from [25]).....	23
Figure 2-9 Comparison by PC Magazine of the most popular home surveillance systems of 2016 (taken from [26]).....	24
Figure 2-10 Withings Home (taken from [27]).....	24
Figure 2-11 Nest Cam (taken from [28])	25
Figure 2-12 Logitech Circle (taken from [29])	25
Figure 2-13 Size comparison between a five dollar bill and the Raspberry Pi Zero (taken from [31])..	27
Figure 2-14 Image of a Raspberry Pi Model A (taken from [33]).....	27
Figure 2-15 Example of a prototype of the Raspberry Pi (taken from [34])	28
Figure 2-16 Eben Upton with Eric Schmidt (Google's president) in an event in which Google donated 15,000 Raspberry Pi's to UK schools (taken from [35])	29
Figure 2-17 Examples of the projects hosted by the Raspberry Pi Foundation (taken from [36])	29
Figure 2-18 A Raspberry Pi being used to emulate video game arcade machine (taken from [41]).....	30
Figure 2-19 Astronaut Tim Peake with an Astro Pi (taken from [40])	31
Figure 2-20 Automatic pet feeder using a Raspberry Pi (taken from [43])	31
Figure 2-21 Example of how to connect GPIO pins to a physical button (taken from [46]).....	33
Figure 2-22 iSpy camera security interface (taken from [48]).....	34
Figure 2-23 Example of motion configuration file (taken from [49])	35

Figure 2-24 Screenshot of MotionEye settings panel (taken from [51])	35
Figure 2-25 Screenshot of RPi Cam Web Interface (taken from [55])	37
Figure 3-1 Network topology of the system: Raspberry Pi's stream their video feeds to a central server where it is presented in a unified interface (MotionEye).....	39
Figure 3-2 Pi Camera connected to Raspberry Pi	40
Figure 3-3 Connections of the central server running MotionEye.....	41
Figure 3-4 Image of a smartphone accessing the video surveillance system externally using the Internet	42
Figure 3-5 Screenshot of RPi-Cam Web Interface	43
Figure 3-6 Screenshot of available options in Camera Settings in RPi-Cam Web Interface.....	44
Figure 3-7 Screenshot of System submenu in RPi-Cam Web Interface	45
Figure 3-8 Screenshot of login screen to MotionEye	45
Figure 3-9 Screenshot of main screen of MotionEye	46
Figure 3-10 Icons of main screen.....	46
Figure 3-11 Screenshot of options presented when a camera preview is clicked.....	46
Figure 3-12 Screenshot of picture browser	47
Figure 3-13 Screenshot of movie browser	47
Figure 3-14 Screenshot of MotionEye configuration/option screen.	48
Figure 3-15 Dropdown menu to select a different camera.....	48
Figure 3-16 Screenshot of MotionEye Preferences sub-menu.....	49
Figure 3-17 Screenshot of MotionEye General Settings sub-menu.....	49
Figure 3-18 Screenshot of MotionEye Video Device sub-menu	50
Figure 3-19 Screenshot of MotionEye File Storage sub-menu.....	51
Figure 3-20 Screenshot of MotionEye Text Overlay sub-menu	53
Figure 3-21 Screenshot of MotionEye Video Streaming sub-menu	53
Figure 3-22 Screenshot of MotionEye Still Images sub-menu	54
Figure 3-23 Screenshot of MotionEye Movies sub-menu	55
Figure 3-24 Screenshot of MotionEye Motion Detection sub-menu	56
Figure 3-25 Screenshot of MotionEye Motion Notifications sub-menu.....	57
Figure 3-26 Screenshot of MotionEye Working Schedule sub-menu	58
Figure 3-27 Screenshot of modified MotionEye interface.....	59
Figure 3-28 Screenshot of configuration used in RPi-Cam Web Interface	61
Figure 3-29 Screenshot of MotionEye Preferences	62
Figure 3-30 Screenshot of MotionEye General Settings	62
Figure 3-31 Screenshot of MotionEye Video Device	63
Figure 3-32 Screenshot of MotionEye File Storage.....	63
Figure 3-33 Screenshot of MotionEye Text Overlay.....	63

Figure 3-34 Screenshot of MotionEye Video Streaming.....	64
Figure 3-35 Screenshot of MotionEye Still Images.....	64
Figure 3-36 Screenshot of MotionEye Movies	64
Figure 3-37 Screenshot of MotionEye Motion Detection.....	65
Figure 3-38 Screenshot of MotionEye Motion Notifications	65
Figure 3-39 Screenshot of MotionEye Working Schedule	65
Figure 3-40 Screenshot of link to NOOBS software	66
Figure 3-41 Screenshot of unzipped NOOBS files.....	67
Figure 3-42 Screenshot of NOOBS menu.....	67
Figure 3-43 Screenshot of Network settings in Raspbian.....	68
Figure 3-44 Screenshot of Raspberry Pi localisation settings.....	68
Figure 3-45 Screenshot of Interfaces section	69
Figure 3-46 Screenshot of RPi-Cam Web Interface's code being downloaded	69
Figure 3-47 Screenshot of installation menu	70
Figure 3-48 Screenshot of configuration used in the installation of RPi-Cam Web Interface	71
Figure 3-49 Screenshot of installation of MotionEye using python-pip.....	72
Figure 3-50 Screenshot of reboot script.....	74
Figure 3-51 Screenshot of shutdown script.....	74
Figure 3-52 Screenshot of code to establish SSH connection	75
Figure 3-53 Script to remotely execute reboot.....	76
Figure 3-54 Script to remotely execute shutdown	77
Figure 3-55 Screenshot of scripts moved.....	77
Figure 3-56 Screenshot making the scripts executable	78
Figure 3-57 Icons available for the predefined actions.....	78
Figure 3-58 Default values of the file main.css	79
Figure 3-59 Screenshot of default main.js	80
Figure 3-60 Screenshot of modified main.js.....	81
Figure 3-61 Screenshot of preview windows with reboot and shutdown icons.....	81
Figure 3-62 Screenshot of MotionEye login screen.....	82
Figure 3-63 Screenshot of the MotionEye menu to a configure a new camera	82
Figure 3-64 Screenshot of MotionEye menu to add camera.....	82
Figure 3-65 Screenshot of MotionEye to add a network camera.....	83
Figure 4-1 Screenshot of how MotionEye shows it detects movement.....	89
Figure 4-2 Screenshot of emails received notifying that movement has been detected	90
Figure 4-3 Comparison of a photo taken in low light with Pi Camera and an iPhone 6	90
Figure 4-4 Comparison of a photo taken with no light with Pi Camera and an iPhone 6.	91

Figure 4-5 Screenshot of MotionEye interface with reboot/shutdown functionality.....	91
Figure 4-6 Screenshot of login screen for RPi-Cam Web Interface	92
Figure 4-7 Screenshot of login attempt in MotionEye.....	92
Figure 4-8 Photo of the conference room of the Defense University Center, circles indicate the location of the Raspberry Pi's.....	93
Figure 4-9 Image of the three Raspberry Pi's, each connected to a battery and using Wi-Fi	94
Figure 4-10 Location of Raspberry Pi 1.....	94
Figure 4-11 Location of Raspberry Pi 2.....	95
Figure 4-12 Location of Raspberry Pi 3.....	95
Figure 4-13 Asus router used for the test.....	96
Figure 4-14 Toshiba notebook being used as central server	96
Figure 4-15 Screenshot of MotionEye running on the central server	97
Figure 4-16 Pictures browser of MotionEye.....	97
Figure 4-17 Example of an image taken by the system	98
Figure 4-18 Sequence of pictures taken by MotionEye when detecting movement.....	99
Figure 5-1 Raspberry Pi inside a CCTV casing (taken from [70])	102
Figure 5-2 Pi Camera mounted on an electric motor with pan and tilt functionality (taken from [71])	103

LIST OF TABLES

Table 2-1 Comparison of the most popular user-installable video surveillance systems	22
Table 2-2 Comparison of the most popular home automation video surveillance systems.....	26
Table 2-3 Hardware differences between the different Raspberry Pi versions (taken from [39]).....	30
Table 3-1 Different options available to each type of user	48
Table 3-2 Description of MotionEye's Preferences options	49
Table 3-3 Description of MotionEye's General Settings options	50
Table 3-4 Description of MotionEye's Video Device options	51
Table 3-5 Description of MotionEye's File Storage options	52
Table 3-6 Description of MotionEye's Text Overlay options	53
Table 3-7 Description of MotionEye's Video Streaming options	53
Table 3-8 Description of MotionEye's Still Images options	54
Table 3-9 Description of MotionEye's Movies options.....	55
Table 3-10 Description of MotionEye's Motion Detection options	56
Table 3-11 Description of MotionEye's Motion Notifications options.....	57
Table 3-12 Description of MotionEye's Working Schedule options.....	58
Table 3-13 Recommended settings for RPi-Cam Web Interface.....	60
Table 3-14 MotionEye commands for Ubuntu 14.10 or previous	73
Table 3-15 MotionEye commands for Ubuntu 15.04 or later.....	73
Table 3-16 Naming scheme of reboot/shutdown scripts.....	76
Table 3-17 Configuration options to add a network camera configured with RPi-Cam Web Interface.....	83
Table 4-1 Fps and CPU load using 100% image quality and different configurations	84
Table 4-2 Fps and CPU load using 75% image quality and different configurations	85
Table 4-3 Fps and CPU load using 50% image quality and different configurations	85
Table 4-4 Fps and CPU load using 25% image quality and different configurations	85
Table 4-5 Fps and CPU load using 10% image quality and different configurations	85
Table 4-6 Fps comparison between Wi-Fi and Ethernet.....	86
Table 4-7 Fps when using various Raspberry Pi's using Wi-Fi and Ethernet.....	86
Table 4-8 Configuration settings used in battery life tests.....	87
Table 4-9 Results of battery life using different settings	87
Table 4-10 Results of battery life using Wi-Fi and Ethernet	88

1 INTRODUCTION AND OBJECTIVES

1.1 Introduction and motivation

1.1.1 Background of video surveillance systems

Although we now consider them commonplace, we must ask ourselves why the need for a video surveillance system in the first place. The first video surveillance system was designed to monitor the launch of the V-2 rockets in 1942 during World War II. It was meant to give a live video feed of the rocket launch, which was too dangerous for humans to remain. The spirit of modern video surveillance systems remains very much the same: monitor a remote area. However, with the progress of technology we have redefined what we use these systems for.

1.1.2 Government use of video surveillance systems

Governments now use them for a variety of purposes and have installed them throughout big cities. In the case of the UK, there are a total of 5.9 million CCTV (Closed Circuit Television) cameras [1]. The main reason for which governments justify their existence is crime prevention. In 2009 an analysis was conducted by the Northeastern University and the University of Cambridge, "Public Area CCTV and Crime Prevention: An Updated Systematic Review and Meta-Analysis," [2] which reviewed 44 different studies from the UK and the US. It was determined that:

1. Surveillance systems were most effective in parking lots, where their use resulted in a 51% decrease in crime;
2. Public transportation areas saw a 23% decrease in crimes;
3. Systems in public settings were the least effective, with just a 7% decrease in crimes overall. When sorted by country, however, systems in the United Kingdom accounted for the majority of the decrease; the drop in other areas was insignificant.

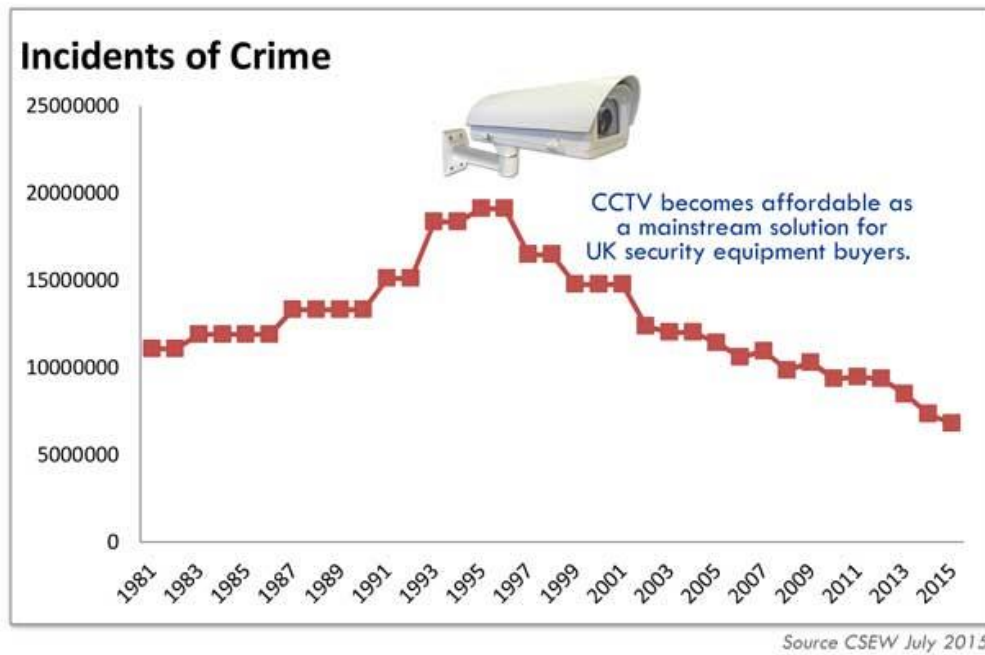


Figure 1-1 Graph representing the decrease in crime in the UK since the use of CCTV cameras (taken from [3])

Cities also use video surveillance systems to monitor traffic. This allows them to detect traffic congestion and accidents and act accordingly.

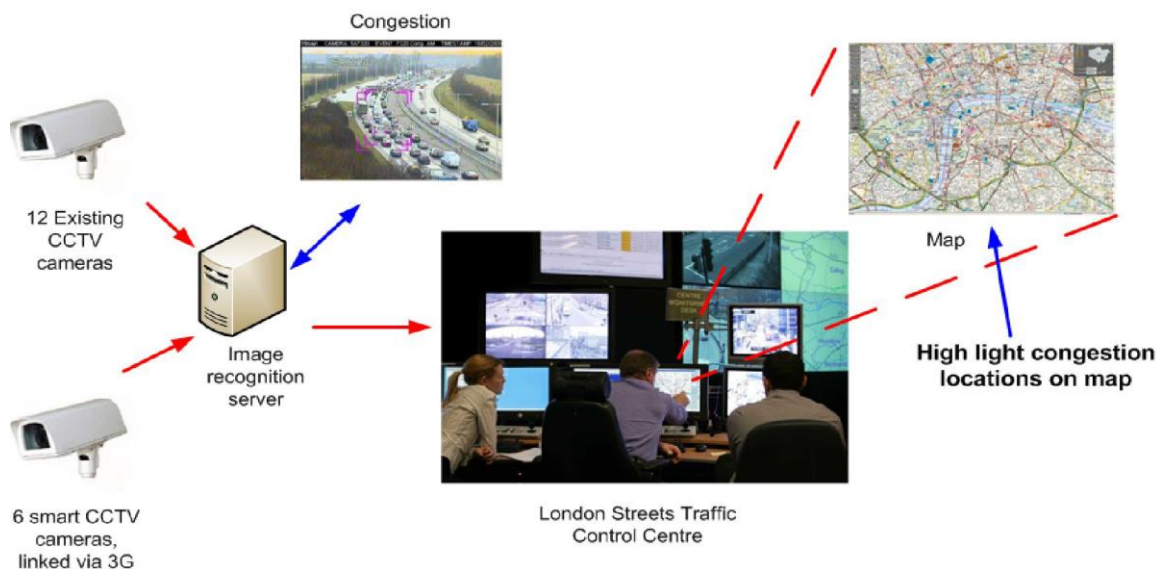


Figure 1-2 Schematic of London's street traffic control center (taken from [4])

Video surveillance technology also allowed for the development of speed cameras, which monitor the speed of a car from one point to another. A study found that areas where there are speed cameras, mortal accidents fell by 42% [4].

Although for obvious reasons governments will never admit to it, they also use video surveillance systems for intelligence gathering. After the terrorist attack in New York on the 11th of September

2001, their use to collect intelligence became more predominant under the Homeland Security Grant Program [5].

1.1.3 Private businesses use of video surveillance systems

In order to be a successful business it is crucial to find and exploit any advantage over your competitors. For this reason, businesses are quick to adopt and drive any technology that will reduce costs and make them more efficient. Alphabet Inc. (previously Google) for example, has surpassed the technology giant AT&T in total revenue while it employs a tenth of the number of workers, by implementing and taking advantage of technology. One such technology is video surveillance systems.

The first use of video surveillance systems in businesses was to increase security. Shoplifting and worker theft cost \$32 million in the U.S. in 2014 [6], the ability to monitor and reduce this behavior is crucial for businesses. Evidently, video surveillance systems also reduce security costs significantly as you can monitor your entire business with video cameras instead of security guards.

Video surveillance equipment is also used to monitor operations in manufacturing. Whereas before it was necessary to have several employees to supervise an entire manufacturing process, a combination of sensors and video cameras can replace them at a fraction of the cost.

Recently video surveillance systems are also being used in businesses not only for security purposes but for evaluating the performance of their employees. However, it raises privacy and ethical concerns for the employees.

1.1.4 Research purposes

The original use for which it was conceived is not only in use today but has evolved for other scenarios. Time-lapse videos, for example, allow the study of how something changes over a long period of time, such as the increase in the size of the ozone hole in Antarctica, or weather reports.

But one of the most advantageous uses of video surveillance systems in research has been in the use of space missions. It has allowed us to receive images and video of places that we still do not have the technological capability to send humans to.

1.1.5 Domestic use of video surveillance systems

In most cases during office hours, homes are generally left unattended and are an easy target for criminals. Therefore the need for a system that provides security to homeowners has a huge market. Video surveillance systems offer just that. A way of monitoring your home remotely while you are not present is very important. So much so, that the market of home video surveillance systems is estimated to be worth €38.8 billion [7].

Parents also use these systems to monitor their children when they leave the house. Nanny cams became popular in the 1990s as it became increasingly common for both parents to have a job. The footage of Nanny Cams has often been in the subject of controversy and outrage as there have been cases of child abuse being proven with this method [8].

1.1.6 Military use of video surveillance systems

It is well known that the military often drives technological advance. A technological superiority is decisive; a clear case was that of the radar or the atomic bomb in World War II [9]. The recent economic crisis in Europe has meant drastic spending cuts throughout the public sector. Attempting to maintain the necessary military capability as well as cut costs requires innovative solutions. One way

to achieve this could be by reducing the cost in personnel by automating certain tasks using technology instead of people.

In the case of the Spanish Navy, this is one of the solutions they have adopted in order to cut costs. Automating as many tasks as possible, in order to have less crew on board results in a significant reduction in expenses. Modern technological ships have all range of systems which are inevitably prone to error and malfunction. Therefore constant supervision is necessary. However, current commercial video surveillance systems tend to be expensive and prone to errors which require assistance from a technician.

Video surveillance systems are also widely used in the field of physical security. The Spanish Navy uses them throughout their naval bases, ships and headquarters. Given the tendency of reducing costs by automating tasks and the inherent need for security, the Spanish Navy has a particular interest and need for an easily configurable as well as cheap video surveillance system.

1.1.7 Conclusion and motivation

Current commercial video surveillance systems tend to be expensive and prone to have errors, which require assistance from a technician. There is a market for an easily configurable as well as cheap video surveillance system. Home automation video systems are designed to fill that niche, however they remain expensive.

One such solution could be the implementation of a video surveillance system based on the SoC (System on a Chip) Raspberry Pi. Highly configurable and extremely cheap, if implemented correctly it could solve this dilemma. This is the idea which motivates this project.

1.2 Objectives

The main objective of this project will be the design and implementation of a video surveillance system using a Raspberry Pi computer board and Pi Camera. The system will be capable of allowing more than one video feed, remote real time streaming of the video feeds and remote control of the system from another network. Each Raspberry Pi will also have the option of being portable by using both a battery as a power source and Wi-Fi to stream the video feed.

It will also have advanced features such as motion detection and act upon detecting motion by sending an email notification and/or recording video. The system will be able to adopt a particular configuration depending on a configurable schedule. The cameras used will also include night vision to be able to work correctly in low-light conditions. The system should also be cheap when compared to commercial video surveillance systems so as to be considered a viable alternative. Finally the system should meet a minimum security standard by being password protected and have different user groups with different user permissions.

1.3 Resources used

The resources used in this project are described in the following subsections:

1.3.1 Hardware

1. Three **Raspberry Pi 2 Model B computers**, whose main characteristics are listed below:

- SoC Broadcom BCM2836

- CPU 900 MHz quad-core ARM Cortex-A7
- GPU Broadcom VideoCore IV at 250MHz
- Memory SDRAM 1GB shared with GPU



Figure 1-3 Image of Raspberry Pi 2 Model B

- Input/Output
 - 4 x USB 2.0
 - Video input 15 pin MIPI camera interface
 - Video/audio output HDMI 1.3
 - Audio output 3.5mm Jack
- Connectivity
 - 10/100 Ethernet port
 - Wi-Fi using USB WIFI adapter
- Storage: MicroSDHC
- Low-Level Peripherals: 17 x GPIO
- Power Source: 5 V via MicroUSB
- Weight: 45g
- Size: 85,6 mm x 56,5 mm

2. Three **Pi Camera Modules** including an infrared led light with the following characteristics:
 - 5 Megapixels
 - Supports 1080p 30fps, 720p 60fps
 - 15cm ribbon cable
 - Infrared board has photo resistor to automatically turn on infrared light in low light conditions

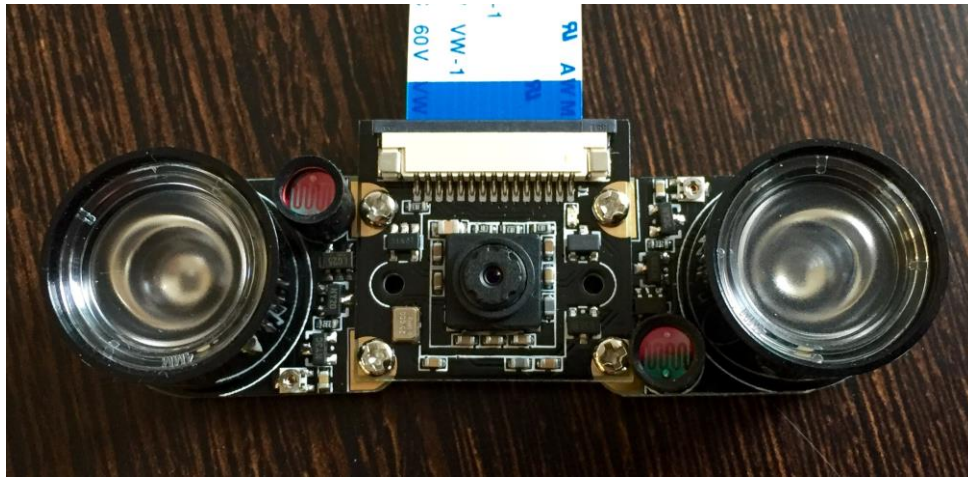


Figure 1-4 Pi Camera with two infrared led light boards

3. Three **Samsung MicroSD's 8GB**
4. Three **Edimax Wireless 802.11b/g/n nano USB adapters** with up to 150Mbps



Figure 1-5 Edimax wireless USB adapter

5. **Toshiba Satellite Pro C850 -1HD Notebook**

6. Three **Batteries**: “RAVPower 13000mAh Deluxe Portable Charger”



Figure 1-6 RAVPower battery

1.3.2 Software

Next, we have a list with the main programs that were used for the development of this project:

1. Raspbian Jessie OS
2. MotionEye 0.3
3. Ubuntu OS 14.04
4. Motion
5. RPi Cam Web Interface 6.0.30
6. Python
7. SSH
8. Bash

1.4 Structure of the document

The document has been divided into five sections.

1. The first chapter of the document gives a background into the motivation behind this project. In particular it discusses the many benefits offered by a video surveillance system in different scenarios. Having introduced the motivation behind installing a video surveillance system it goes on to define the objectives established in this project as well as the hardware and software used to develop it.
2. The second chapter includes the state of the art of video surveillance systems. It goes into detail into the history and current features of video surveillance systems. It then goes on to

explore the history of the Raspberry Pi and its current capabilities. Finally it lists the features and capabilities of current video surveillance systems designed for the Raspberry Pi.

3. In the third chapter, a video surveillance system using Raspberry Pi is developed. This section goes into detail with regard to the architecture to the system, including every detail and its features. An ideal configuration is recommended based on testing. Finally the changes made to the code of the programs to increase their functionality is introduced and explained.
4. The fourth chapter explains the different tests that were carried out. This includes performance and battery life tests in different scenarios. A test on the advanced features of the system was also done. Finally the system is tested in a real world scenario to ensure it works properly.
5. Finally in the fifth chapter, conclusions are drawn from the entire project and an evaluation as to whether or not the initial objectives have been met is done. Possible future lines of study are also explored given the many possibilities the system offers.

2 STATE OF THE ART

In this chapter we will review what current commercial video surveillance systems entail, specifically what they offer, what technology they use, cost and ease of use. From there we will discuss the different Raspberry Pi video surveillance solutions that exist, compare them to the commercial counterparts and determine which one would be the best option.

2.1 History of video surveillance systems

2.1.1 Introduction

Video surveillance systems have evolved significantly in the past 70 years since the first one was implemented. At first they were purely analogue systems that could only display a live image from a remote location. Nowadays they are purely digital and have incorporated a multitude of advancements including: recording, motion detection, facial recognition, infra-red detection, night vision, etc.

2.1.2 First CCTV

The first instance of a live video being displayed elsewhere took place in Germany in 1942 as part of the tests of the launch of the V-2 Rockets. This very same technology was later used to monitor tests of the first atomic bombs. However, these systems were unable to record video and limited only to displaying a live feed such as the system shown in Figure 2-1.



Figure 2-1 Example of an early CCTV system (taken from [10])

2.1.3 Video cassette recorders

With the introduction of Video Cassette Recorders (VCR's) in 1951, it was possible to record from an analogue camera. This was a crucial step in video surveillance systems. The ability to record a video feed meant that it was no longer necessary to watch the monitor constantly, as the video could be reviewed at another time.

In order to extend how much could be recorded to a single tape, the video feed was recorded with a time-lapse at a lower rate of frames per second, allowing more video to be recorded albeit at a lesser quality. This paved the way for multiplexers, which allowed various feeds from different cameras to be integrated in one single video feed and displayed on a monitor, which meant you could use one single monitor to view various video feeds.

2.1.4 CCD (Charge-Coupled Device)

During the 1970s CCD technology was developed, allowing to record in low-light situations. This broadened the use of video surveillance technology, increasing its popularity in the private sector (banks, private companies) as they could now monitor during the night which is from a security standpoint when they are most vulnerable.

2.1.5 DVR's (Digital Video Recorder)

As DVR's are developed in the 1990s they replace VCR's. They allow the video feed to be recorded to a hard drive which saves huge amounts of video tapes. It also increases the quality of the video allowing to zoom into an image.

2.1.6 Nanny Cams

Nanny cams were conceived as a way for parents to monitor what was occurring at their homes while at work. Nanny Cams had considerable success; disguised as inconspicuous objects their aim was to monitor caregivers as shown in Figure 2-2. This required smaller, higher resolution cameras to be viable. It was the first real step towards a home surveillance system.



Figure 2-2 A Nanny Cam disguised as a teddy bear (taken from [11])

2.1.7 IP Cameras

Unlike CCTV, IP Cameras are designed to send the video feed using the IP protocol across computer networks. This means that they can be used to transmit video using the Internet. Advances in resolution and image quality have resulted in advanced surveillance systems used in an ever-growing number of scenarios.

2.2 Current commercial video surveillance systems

2.2.1 Introduction

Video surveillance systems offer a lot of advantages to businesses, governments and homes. They are mainly used to increase security and offer the capability of constant remote surveillance. To be able to increase the security and deter criminal activity by as much as 51% [2] is extremely important. They also reduce costs considerably, allowing monitoring various areas with less manpower. In the UK alone there are over 5,9 million CCTV's [1]. The increase in both technology and capability of these systems has made them commonplace with a wide range of systems from which to choose from.

2.2.2 Capabilities and features

As discussed in section 2.1.1. video surveillance systems have evolved considerably in the past 70 years since their first implementation in 1942 [12]. Since their conversion to purely digital technology they have gained many capabilities such as:

1. **High-resolution images and video in HD resolution (720p/1080p).** Evidently this enables a much better opportunity to recognize and see the elements of the video feed. As we can see in Figure 2-3, the increase in resolution is quite significant.



Figure 2-3 Comparison of SD resolution to HD resolutions (taken from [13])

2. **Infrared imaging, improved night-time/low-light scenarios.** The capability of displaying and recording images using thermal imaging and low-light scenarios presents significant advantages over previous video surveillance systems. With this new technology, video surveillance systems can not only be effective during daytime, but all the time, even allowing for surveillance with no light.



Figure 2-4 Comparison of the same scenario with visible light and thermal imaging (taken from [14])

3. **Video content analysis (VCA). Motion detection, facial detection, facial recognition, video tracking, etc.** Having converted video surveillance systems to purely digital, VCA was introduced with different capabilities. By processing the video feed and determining the difference in pixels from one frame to another we can apply different algorithms to detect motion, faces, objects etc.

Motion detection is a very important advantage as it allows the system to take action upon detecting movement. Therefore, the system does not require the constant analysis and supervision of a person. Different actions based on motion being detected could involve sending an alert, initiating an alarm, alerting the police, etc. This introduces a whole new vision for video surveillance systems, giving them a certain autonomy to act and carry out predefined actions in certain situations.



Figure 2-5 An example of a video surveillance system tracking movement (taken from [15])

4. **Network-based connectivity allowing remote viewing** (e.g. from a smartphone). CCTV stands for “closed-circuit television”, what this meant was that the video surveillance system was closed, you could only view the video feed from within the system. With the introduction and rise of Internet, video surveillance systems introduced the capability of watching video feeds remotely. This opens up lots of new possibilities, such as watching a live video feed of your home from your workplace, etc.



Figure 2-6 Example of how a video surveillance system with remote viewing using the Internet is configured (taken from [16])

5. **Capability to orient a camera (pan and tilt).** The ability to orient a camera allows the coverage of a wider area as well as focusing on a particular element. This is achieved with a small electric motor. The user sends a command (usually with a joystick) and the camera will move in real-time.

2.2.3 Examples and discussion of current commercial systems

Given the range of capabilities that now exist for video surveillance systems and the range of customers from home owners to big businesses, there are lots of different types of video surveillance systems tailored to suit different needs and budgets. There are systems that include a dedicated DVR/NAS recording and monitoring system with analogue cameras connected with a physical cable to home automation surveillance cameras. In 2010, according to the Security Sales and Integration website, there are 10 cameras installed on average in commercial settings, with an average cost of €12.278 [17].

The most expensive systems such as those offered by ADT (ADT Corporation is the largest security Company of the United States) include 24-hour monitoring for a monthly fee. Systems that are not based on a monthly fee are known as IP based video surveillance. Current cameras based on IP technology include Wi-Fi technology making them much easier to install without the need of a technician given that they do not require a physical connection to the DVR/NAS/computer that controls the system. Given the ease of installation they are the most popular. They generally include:

- DVR device or a NAS device to record video.
- Up to four individual Wireless IP security cameras
- Software on a dedicated home computer and an LCD monitor
- Motion detection

In Table 2-1, the most popular video surveillance systems that include the aforementioned specifications and are user-installable without the need of a technician from different sites are listed.

WEBSITE	NAME	EXTRA FEATURES	TOTAL PRICE
www.supercircuits.com (taken from [18])	Amcrest WLD895US 720P 4CH 7-Inch Wireless Video Surveillance System, Bullet IP66 Camera (Black)	iOS/Android app Night Vision Outdoor weather resistant	€913,65
www.amazon.com (taken from [19])	ZOSI 4PCS Megapixel 720P Wireless Outdoor IP Camera System 30M Night vision with 4CH Security 720P HD Network NVR Wi-Fi Kit Support Smartphone Remote view with 1TB Hard disk	iOS/Android app Email alert upon motion detection Night vision	€283,43
www.amazon.com (taken from [20])	ANRAN 4CH Network NVR Wi-Fi Kit Megapixel 720P HD Wireless Outdoor IP Network Home Surveillance Camera System with 1TB Hard Drive Free APP for iPhone Plug&Play	Outdoor weather resistant Night vision	€320,47

Table 2-1 Comparison of the most popular user-installable video surveillance systems

2.2.4 Comparison/Conclusion/Table

From the examples in Table 2-1, we can determine that modern surveillance systems at the very least will cost approximately €300 if they are user-installed. A medium range video surveillance system installed by a licensed electrician will cost from €740 - €926 [21].

When compared to other “homebrew” alternatives such as a Raspberry Pi with a camera that has an approximate cost of €72 [22] it is clear that commercial systems are expensive. The question that remains is whether a Raspberry Pi with a camera setup is a viable solution to a standard commercial video surveillance system.

2.3 Home automation video surveillance systems

2.3.1 Introduction

In recent years, the unstoppable rise and proliferation in advanced and cheap technology and information systems (as we can see in Figure 2-7) has paved the way for entirely new fields that were often times unheard of. This is very true in particular for the idea of “home automation”.

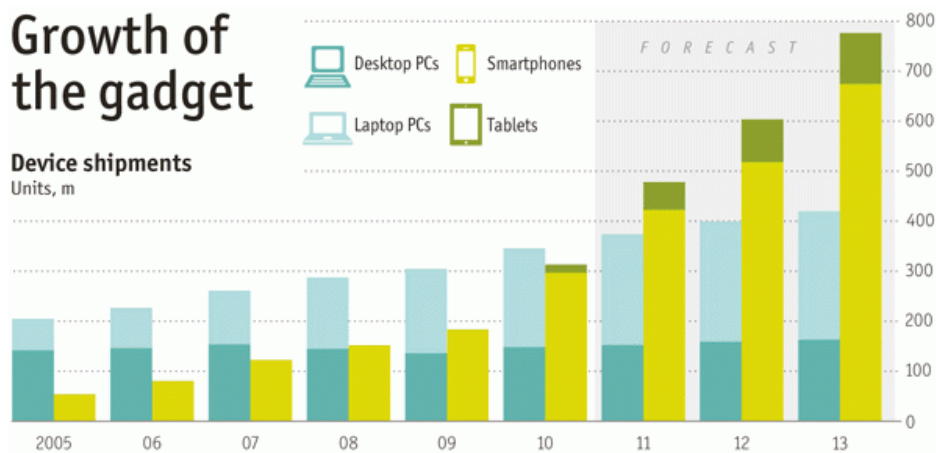


Figure 2-7 Graph showing the growing tendency in the number of gadgets per year (taken from [23])

Many have seen and dreamed of the image portrayed in many futuristic films like the science-fiction cartoon “The Jetsons” [24] in which the entire house was automated. The idea of controlling lights, temperature, home appliances, entertainment systems, etc. with your voice or your mobile phone is very attractive. Recently this idea has come into fruition. Pioneering technology companies such as Apple have begun defining and launching their own home automation systems.



Figure 2-8 Example of home appliances being controlled remotely with a smartphone (taken from [25])

Among these integrated systems, video camera surveillance systems have also become more accessible and advanced. Unlike the commercial video surveillance systems discussed in section 2.2.3, they are specifically tailored for the home automation market. They are meant to be easily user-installable, appealing, with Wi-Fi, include advanced features and without the need for a dedicated DVR/NAS/computer.

However, they remain difficult to customize to one's needs. Without mentioning that they remain expensive as we can see in Figure 2-9 where the most popular home surveillance video systems are shown.

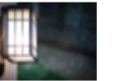
D-Link HD Wireless N Day/Night Outdoor Cloud Camera (DCS-2330L)	Icontrol Networks Piper nv	Withings Smart Baby Monitor	Canary All-In-One Home Security Device	Nest Cam	Netgear Arlo Security System (VMS3230)	Flir FX WiFi Home Monitoring Camera FXV101-H	Kuna	Logi Circle	Withings Home
									
€170	€213	€163	€176	€185	€259	€139	€185	€185	€199
Pixmania	Amazon	Amazon	Amazon	Amazon	Amazon	PC World	MSRP	Amazon	Amazon

Figure 2-9 Comparison by PC Magazine of the most popular home surveillance systems of 2016 (taken from [26])

2.3.2 Withings Home

Withings is a company that specializes in health sensors and equipment. The “Withings Home” (shown in Figure 2-10) is their offering for a home surveillance video system. It includes a speaker, microphone, Wi-Fi, Bluetooth, 1080p video, night vision and air quality sensors. Motion and sound sensors are also included and it can alert through push notifications. It also features push-to talk communications as well as recording video for two days, and offers 30 days video recording in its Premium plan for €7,95 a month.



Figure 2-10 Withings Home (taken from [27])

2.3.3 Nest Cam

Nest Labs was acquired by Google in 2014. Although at first Nest specialized in thermostats they began building other products related to home automation. One such product was the Nest Cam as we can see in Figure 2-11, which as of writing is the most popular home automation surveillance video camera. It includes 1080p video, speaker, microphone, digital zoom, night vision, motion and sound sensors as well as Wi-Fi and Bluetooth. It also includes push-to talk and push notifications. However, to view video recordings one must subscribe to their Premium service for €10 a month.



Figure 2-11 Nest Cam (taken from [28])

2.3.4 Logi Circle

Logitech has been a long time PC accessory maker, especially in the field of keyboards and mice. The Logi Circle (shown in Figure 2-12) features 720p video, speaker, microphone, motion detection, push alerts, push-to talk, night-vision, digital zoom, 24-hour recordings, Wi-Fi and Bluetooth. Unlike the majority of its competitors it also features a battery that lasts 12 hours on a single charge. However, it cannot store video for longer than 24 hours.



Figure 2-12 Logitech Circle (taken from [29])

2.3.5 Conclusions of home automation video surveillance systems

We can conclude from Table 2-2 that products in the home automation video surveillance market are very similar regarding their price and the features they offer. There are several notable differences they offer when compared to commercial video surveillance systems such as audio recording as well as the push-to talk feature which allows two-way communication. Initially they are also less expensive, however, if we were to buy four home automation video surveillance cameras to rival the commercial video surveillance systems we find that they stand at similar price point of €800 [21].

NAME	FEATURES	PRICE	NOTABLE DIFFERENCES
Withings Home	Speaker & Microphone Wi-Fi & Bluetooth 1080p video: Night vision, digital zoom Motion & sound sensors Push Notifications Alerts Push-to talk	€199 €7,95 for 30 days video storage.	Air quality sensors
Nest Cam	Speaker & Microphone Wi-Fi & Bluetooth 1080p video: Night vision, digital zoom Motion & sound sensors Push Notifications Alerts Push-to talk	€185 €10 to view recordings	
Logi Circle	Speaker & Microphone Wi-Fi 720p video: Night vision, digital zoom Motion & sound sensors Push Notifications Alerts Push-to talk	€185	Maximum video storage: 24 hours 12 hour battery

Table 2-2 Comparison of the most popular home automation video surveillance systems

2.4 Raspberry Pi

2.4.1 Introduction

In February 2015, the Raspberry Pi Foundation announced they had sold 5 million units since they first began selling Raspberry Pi's in 2012, and thus became the fastest-selling British Computer [30]. That in and of itself is a huge achievement but it was eclipsed a few months later by the announcement of the fastest-selling computer in history achieved by the \$5 Raspberry Pi Zero [31].

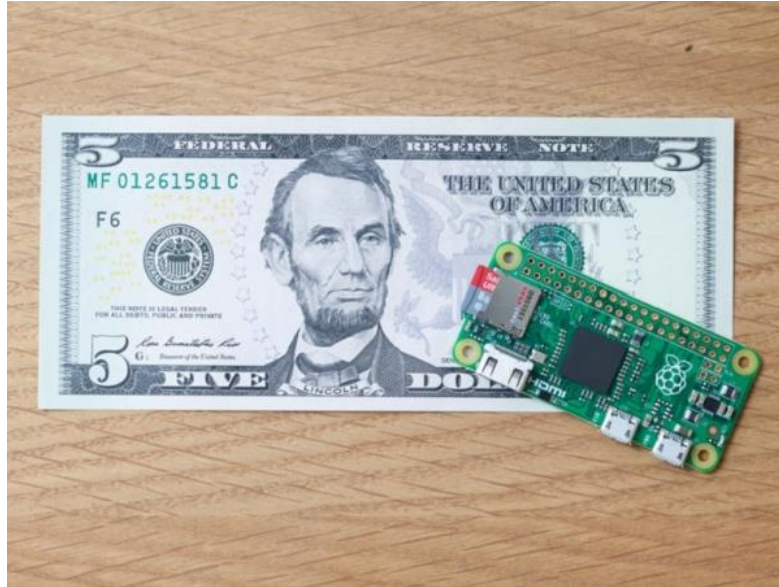


Figure 2-13 Size comparison between a five dollar bill and the Raspberry Pi Zero (taken from [31])

But what exactly is a Raspberry Pi? Its creators describe it as *“The Raspberry Pi is a credit-card sized computer that plugs into your TV and a keyboard. It is a capable little computer which can be used in electronics projects, and for many of the things that your desktop PC does, like spreadsheets, word-processing, browsing the internet and games. It also plays high-definition video. We want to see it being used by kids all over the world to learn programming”* [32].

The Raspberry Pi Foundation is a non-profit charity which aims at teaching and enabling children on how to program code. Their idea was to create the cheapest possible hardware platform as well as the software necessary so that it would become an affordable, educational tool for computer science. They offer different Raspberry Pi models ranging from \$5-\$35 which is a key factor of their success.

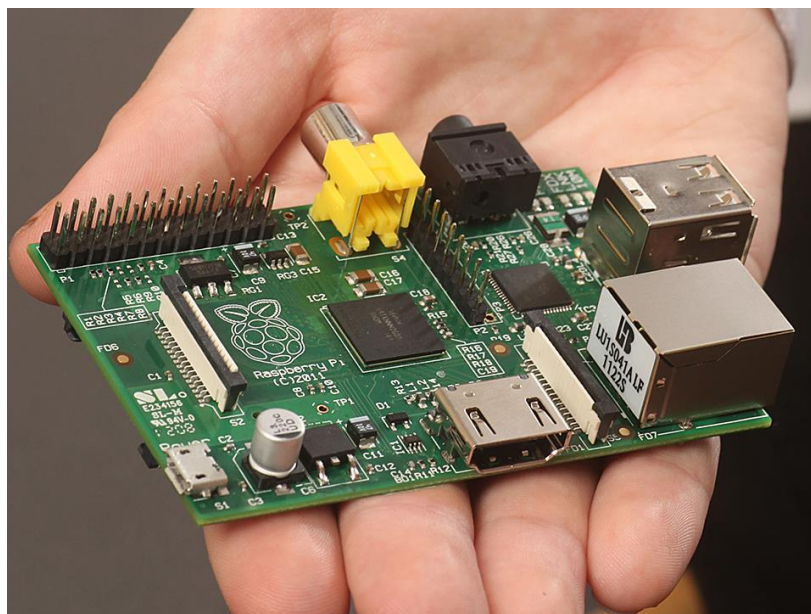


Figure 2-14 Image of a Raspberry Pi Model A (taken from [33])

2.4.2 History

The idea behind the Raspberry Pi Foundation had its origin in the Computer Science Department of the University of Cambridge in the early 2000s. Working as the Director of Studies in Computer Science in the University of Cambridge, Eben Upton was responsible for the undergraduate admissions in Computer Science. He realized that the students applying had far less knowledge and experience of different programming languages than when he had first applied 10 years ago. He shared this sentiment with his colleagues who happened to share this frustration.

Upton concluded that this was due to the tendency of computer manufactures making their products more user-friendly and more difficult to program and experiment with. Along with several of his colleagues they founded the Raspberry Pi Foundation whose goal was not that of building a small cheap computer, but of educating children on how to program.

However they realized that in order to achieve their goal they would need a common hardware platform that was cheap and had the proper programming environment. From 2006 to 2008, they designed various prototypes that fit their vision. As processors designed for mobile devices, specifically, ARM, evolved and become powerful enough, they designed and announced the launch of their first products.

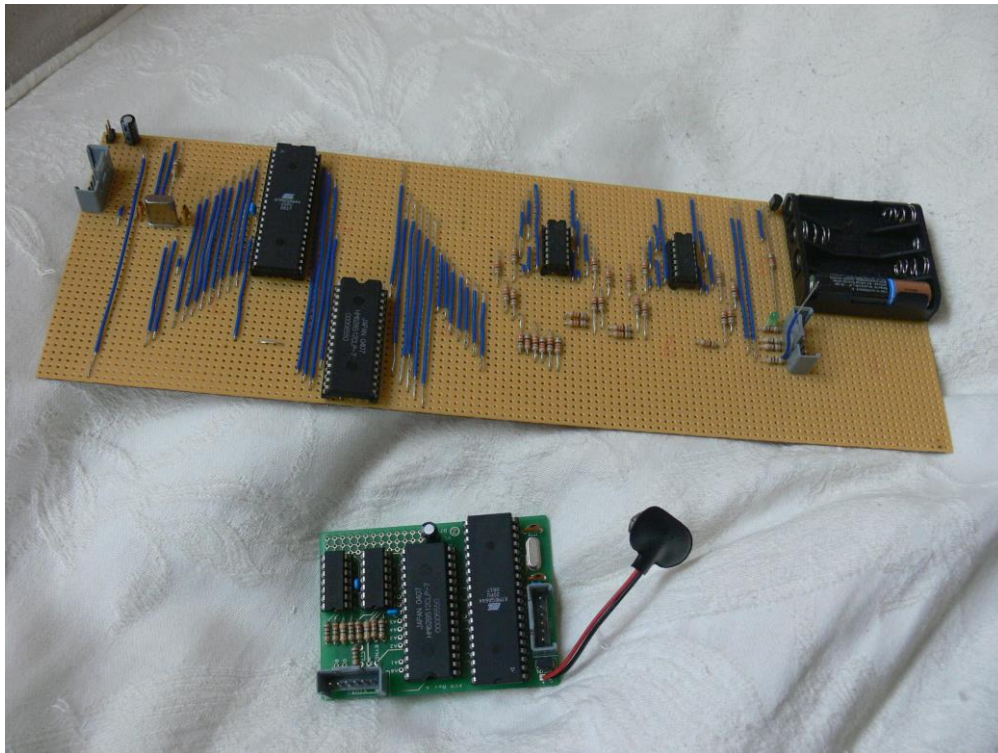


Figure 2-15 Example of a prototype of the Raspberry Pi (taken from [34])

Although the Raspberry Pi Foundation had originally planned for the launch of two models (Model A and Model B) in February 2012, the sale launch was plagued with delays due to licensing and manufacturing issues, so at first only Model B was released. The difference between Model A and Model B lies in the hardware and price. Model A costs \$25 while Model B costs \$35 and includes an Ethernet port and an extra USB port.

Initial supply immediately sold out and although the Raspberry Pi was a huge success the Raspberry Pi Foundation barely sold above cost price given its charitable nature and original goals. However, Upton explains that they invest that money in a variety of educational projects.

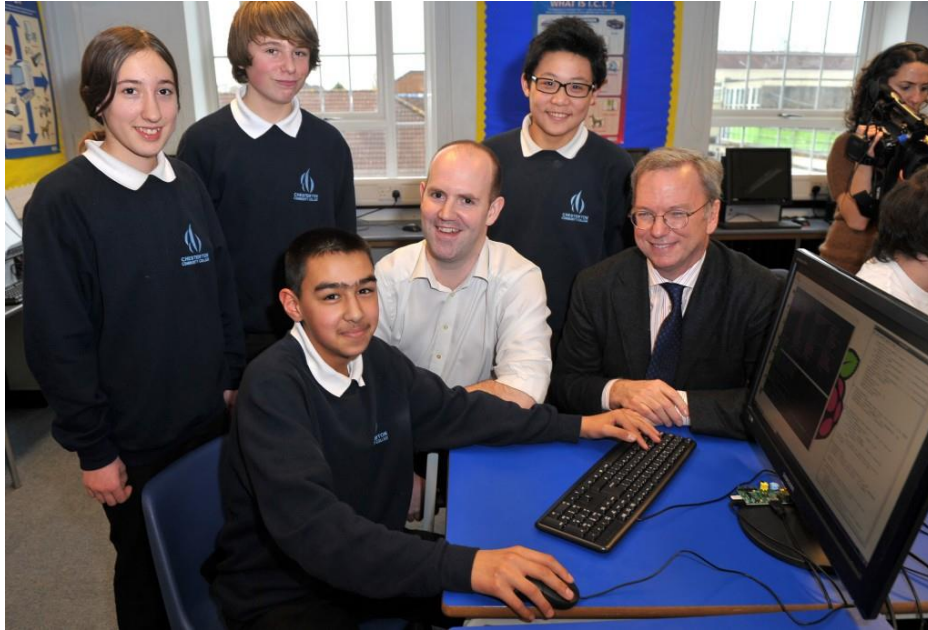


Figure 2-16 Eben Upton with Eric Schmidt (Google's president) in an event in which Google donated 15,000 Raspberry Pi's to UK schools (taken from [35])

They host a multitude of free teaching resources, step by step guides and invest heavily in free teacher training courses.

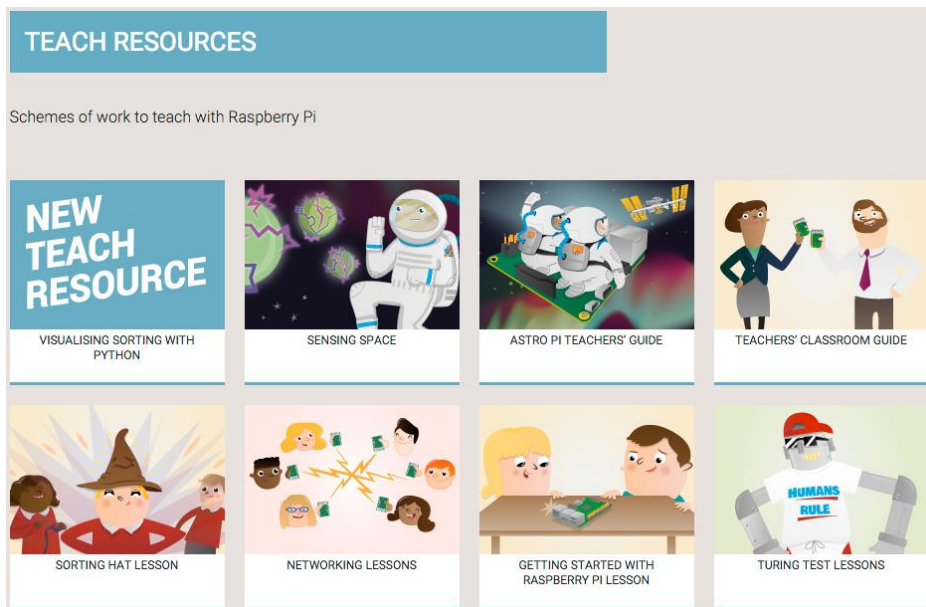


Figure 2-17 Examples of the projects hosted by the Raspberry Pi Foundation (taken from [36])

The Foundation continued to innovate and release new products and upgrades. They also began to focus on software optimization. As the community around the Raspberry Pi grew, two volunteers, Mike Thompson and Peter Green set out to compile and optimize a version of Debian Linux specifically for the Raspberry Pi hardware which they named Raspbian [37]. Raspbian is as a fully-fledged OS specifically designed and optimized for Raspberry Pi which includes educational programs specifically designed to learn to code such as Scratch [38].

As detailed in Table 2-3, since the first release in February 2012 the Raspberry Pi has kept a steady release of new models with hardware upgrades, the latest being the Raspberry Pi Zero released in November 2015 with a retail price of \$5.

	Raspberry Pi 1 Model A	Raspberry Pi 1 Model A+	Raspberry Pi 1 Model B	Raspberry Pi 1 Model B+	Raspberry Pi 2 Model B	Compute Module	Raspberry Pi Zero
Release date	February 2012	November 2014	April–June 2012	July 2014	February 2015	April 2014	November 2015
Target price	US\$25	US\$20	US\$35	US\$25	US\$35	US\$30 (in batches of 100)	US\$5
SoC	Broadcom BCM2835 (CPU, GPU, DSP, SDRAM, one USB port)				Broadcom BCM2836 (CPU, GPU, DSP, SDRAM, one USB port)	Broadcom BCM2835 (CPU, GPU, DSP, SDRAM, one USB port)	
CPU	700 MHz single-core ARM1176JZF-S				900 MHz quad-core ARM Cortex-A7	700 MHz single-core ARM1176JZF-S	1 GHz ARM1176JZF-S single-core
GPU	Broadcom VideoCore IV @ 250 MHz OpenGL ES 2.0 (24 GFLOPS) MPEG-2 and VC-1 (with license), 1080p30 H.264/MPEG-4 AVC high-profile decoder and encoder						
Memory (SDRAM)	256 MB (shared with GPU)		512 MB (shared with GPU) as of 15 October 2012. Older boards had 256 MB (shared with GPU)		1 GB (shared with GPU)	512 MB (shared with GPU)	

Table 2-3 Hardware differences between the different Raspberry Pi versions (taken from [39])

2.4.3 Current status and future

Given the success and sales of the Raspberry Pi it has been used in all sorts of projects and scenarios. From emulating a video game console (Figure 2-18) to being launched to space and used for research in the International Space Station in the Astro Pi project [40].



Figure 2-18 A Raspberry Pi being used to emulate video game arcade machine (taken from [41])



Figure 2-19 Astronaut Tim Peake with an Astro Pi (taken from [40])

Other hobbyist projects include from self-piloting ships to home automation projects, such as an automatic pet feeder as shown in Figure 2-20, the Raspberry Pi community is using them to build just about anything imaginable. Rob Bishop, one of the developers of the Raspberry Pi, described it as the “*lego of computers*” [42] intended to be used as cheap building blocks for just about anything, even a video surveillance system.



Figure 2-20 Automatic pet feeder using a Raspberry Pi (taken from [43])

2.5 Current Raspberry Pi video surveillance systems

2.5.1 Introduction

Given the price and lack of configurability of current commercial video surveillance systems, the Raspberry Pi community has created a series of different homebrew alternatives. Due to its price, at just \$35 it is very affordable and therefore very accessible. Its characteristics make it ideal for hobbyist projects, in particular those related to home automation. From feeding pets according to a schedule, acting as an entertainment center, turning on lights, etc.

In the field of video surveillance with the Raspberry Pi, there are a series of different projects, using both standard USB Cameras and the official Pi Camera module.

2.5.2 Tutorials Raspberrypi.org

The first resource to configure and control the Pi Camera is the official Raspberry Pi webpage [44], which offers two different options on how to interact with the Pi Camera, either using Shell (Linux command line) or Python.

1. Shell

In the Linux shell we are presented with three applications used to interact with the Pi Camera from the command line:

1. *raspistill*: Used to capture images
2. *raspivid*: Used to capture video
3. *raspiyuv*: Used to capture images without an encoder.

There are a whole range of options on how to control the camera, from rotating the image, inverting colors, recording at different frame rates, etc. However, other than the basic usage of the Pi Camera we do not find any advanced tutorials or guides on designing a video surveillance system using these applications in the Shell.

2. Python

The Raspberry Pi Foundation promotes and implements the use of python as a programming language for several reasons. They argue that python is a powerful programming language which is easy to use [45], cross-platform and free.

In the python resource section [46] the basics of the *python-picamera* interface are explained, in a similar fashion to the Shell use of the Pi Camera, there are tutorials on the basic options of the python-picamera interface.

Within the educational resources there is a workshop [46] aimed at teaching students the use of the picamera python module and how to connect it to a physical button using the GPIO pins. The workshop lists the code necessary and explains what the code will do, but it does not explain how the code works so a minimum understanding of Python is required.

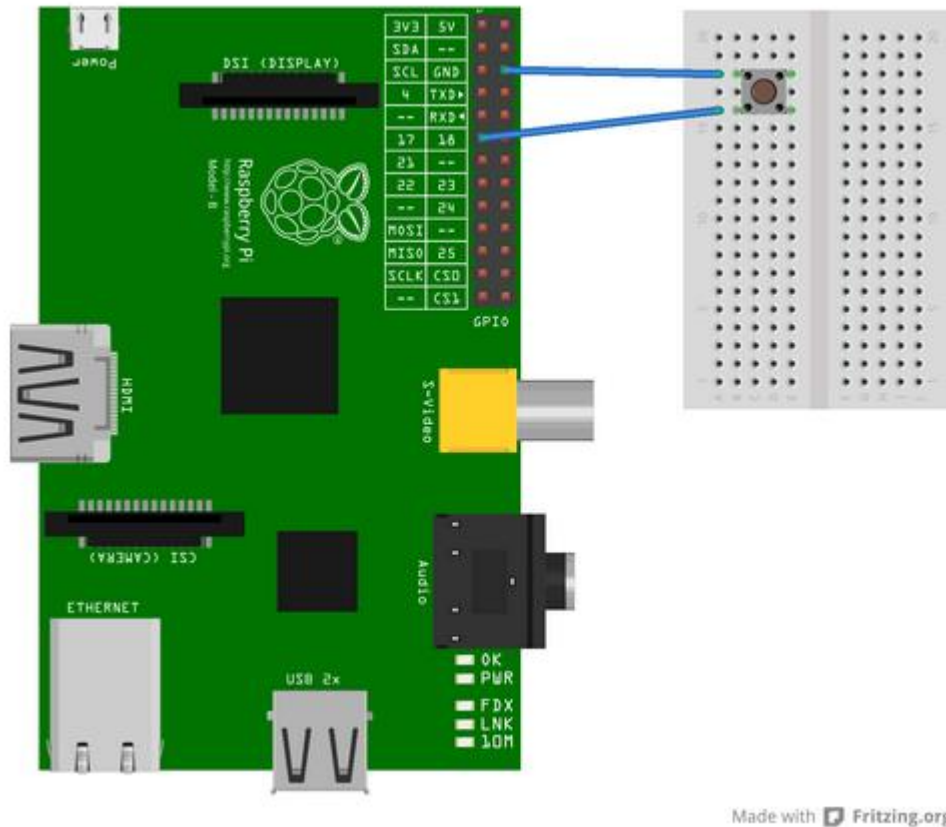


Figure 2-21 Example of how to connect GPIO pins to a physical button (taken from [46])

The workshop goes on to explain other options and variations that can be done to the code to achieve different results, such as:

- Camera programming: countdown capture (selfies)
- Camera programming: camcorders
- Camera programming: circular buffers
- Time-lapse photography
- Stop-motion animation

However, there are no resources into how to build a video surveillance system with the characteristics detailed in section 2.2.3. It does mention that it is possible to build a security camera system capable of detecting movement either using a sensor or using software to detect motion by using other resources found online for more advanced users such as the picamera python module documentation, discussed in section 2.5.3.

2.5.3 Picamera python module

The picamera python module can be used to achieve many of the functions of modern video surveillance systems. In the documentation [47] there are details on the hardware of the camera and how the encoders and technology of the module work.

There are scripts for both basic and advanced functions as well as explanations on how these scripts work and the different options that can be implemented to the scripts. Some of the scripts within the documentation that would be of use for a video surveillance system are:

- Capturing to a file
- Capturing to a stream

- Capturing in low light
- Capturing to a network stream
- Overlaying images on the preview
- Controlling the LED
- Recording at multiple resolutions
- Capturing images whilst recording
- Custom outputs
- Motion detection

Although these functions are certainly part of the features expected from a video surveillance system, they are presented as individual scripts. It is necessary to have a unified and intuitive interface that contains these functions, similar to the interface of a commercial video surveillance system such as the one detailed in Figure 2-22.

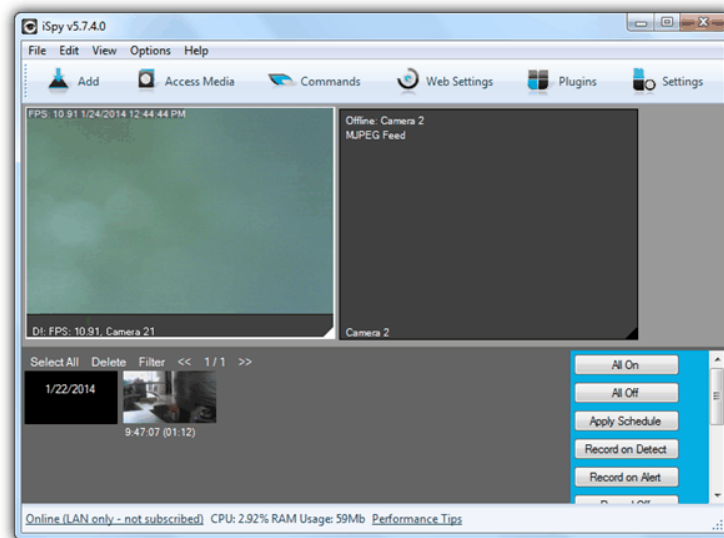


Figure 2-22 iSpy camera security interface (taken from [48])

2.5.4 MotionEye and MotionEyeOS

The most popular software solution for motion detection for Linux is the program *motion*. It is able to analyze the individual frames of a video stream and determine if there are changes. Motion incorporates many features and options such as:

- Movement detection
- Take automated snapshots/videos based on movement
- Execute external commands based on movement (e.g. send an email)
- Real time encoding of video stream into MPEG movies

However, the program is command-line driven and does not have a graphical user interface. To configure its parameters and options, one must edit a configuration file, as shown in Figure 2-23, which is not user-friendly.

```

GNU nano 2.2.6      File: /etc/motion/motion.conf

# Bitrate to be used by the ffmpeg encoder (default: 400000)
# This option is ignored if ffmpeg_variable_bitrate is not 0 (disabled)
ffmpeg_bps 500000

# Enables and defines variable bitrate for the ffmpeg encoder.
# ffmpeg_bps is ignored if variable bitrate is enabled.
# Valid values: 0 (default) = fixed bitrate defined by ffmpeg_bps,
# or the range 2 - 31 where 2 means best quality and 31 is worst.
ffmpeg_variable_bitrate 0

# Codec to used by ffmpeg for the video compression.
# Timelapse mpegs are always made in mpeg1 format independent from this option.
# Supported formats are: mpeg1 (ffmpeg-0.4.8 only), mpeg4 (default), and msmpeg$
# mpeg1 - gives you files with extension .mpg
# mpeg4 or msmpeg4 - gives you files with extension .avi
# msmpeg4 is recommended for use with Windows Media Player because
# it requires no installation of codec on the Windows client.
# swf - gives you a flash film with extension .swf
# flv - gives you a flash video with extension .flv
# ffv1 - FF video codec 1 for Lossless Encoding ( experimental )
# mov - QuickTime ( testing )
ffmpeg_video_codec msmpeg4

# Use ffmpeg to deinterlace video. Necessary if you use an analog camera
# and see horizontal combing on moving objects in video or pictures.
# (default: off)
ffmpeg_deinterlace off

```

Figure 2-23 Example of motion configuration file (taken from [49])

MotionEye [50] serves as a web frontend for *motion*. Providing a user-friendly interface as shown in Figure 2-24, in which it is simple to configure the options for the underlying program (*motion*).

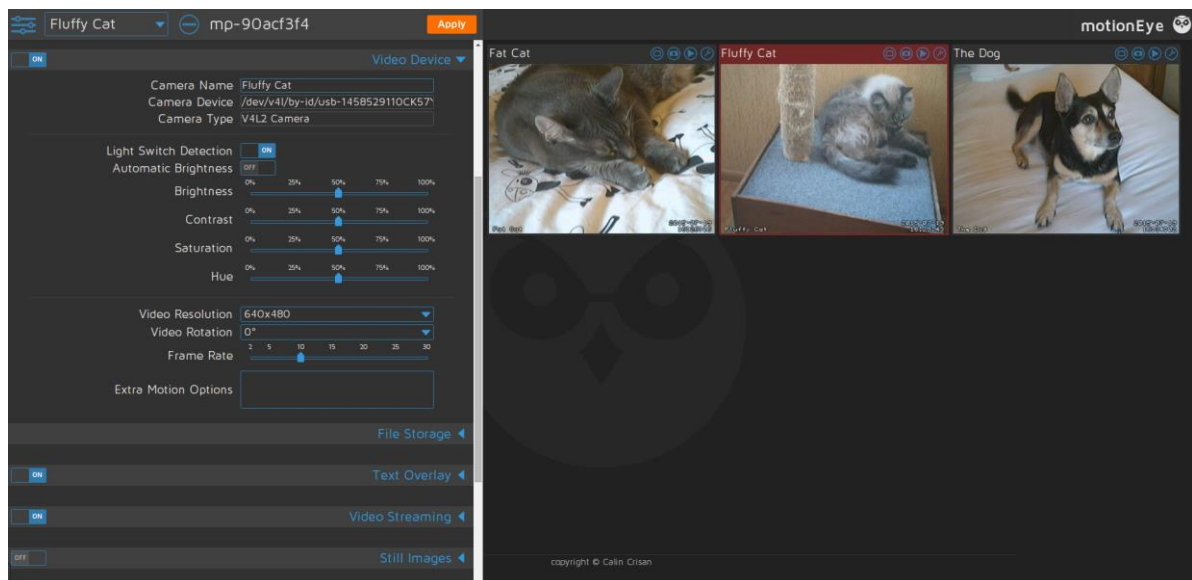


Figure 2-24 Screenshot of MotionEye settings panel (taken from [51])

All the features it incorporates according to [52] are:

- “Web-based, mobile/tablet-friendly user interface
- compatible with most USB cameras as well as with the Raspberry PI camera module
- support for IP (network) cameras
- motion detection with email notifications and working schedule

- *JPEG files for still images, AVI files for videos*
- *time-lapse movies*
- *uploading media files to Google Drive and Dropbox”*

MotionEye provides a suitable video surveillance environment which is installable in a variety of Linux distributions and systems.

MotionEyeOS [53] is a Linux distribution specifically designed for single-board computers (such as Raspberry Pi). It incorporates all the functions of the MotionEye program using motion as the underlying technology. It is ideal for a scenario in which a single-board computer is going to be used exclusively for video surveillance.

2.5.5 RPi Cam Web Interface

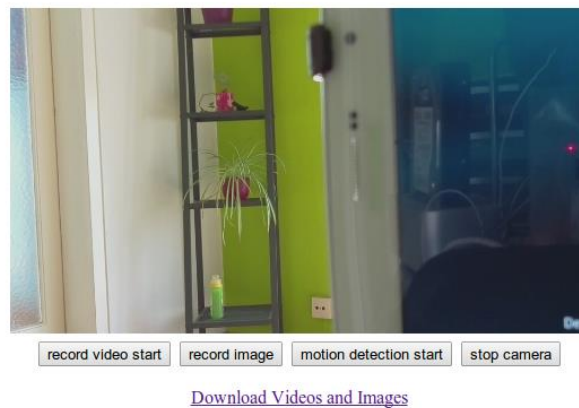
Unlike MotionEye, RPi Cam Web Interface [54] is specifically designed for the Raspberry Pi and Pi Camera. It also features a web interface to control the program *motion* and *RaspiMJPEG* which provides the underlying technology for most of its features.

According to [54] it includes:

- *“View, stop and restart a live-preview with low latency and high frame rate. Full sensor area available.*
- *Control camera settings like brightness, contrast, ... live*
- *Record full-HD videos and save them on the SD-card packed into mp4 container while the live-preview continues*
- *Take single or multiple (time-lapse) full-res pictures and save them on the SD-card (live-preview holds on for a short moment)*
- *Preview, download and delete the saved videos and pictures, zip-download for multiple files*
- *Trigger captures by motion detection*
- *Trigger captures by many scheduling-possibilities*
- *Circular buffer to capture the last actions afterwards*
- *Control Pan-Tilt or Pi-Light*
- *Shutdown/Reboot your Pi from the web interface*
- *Show annotations (e.g. timestamp) on live-preview and taken images/videos”*

RPi Cam Web Interface has been in development since 2013. As shown in Figure 2-25, it also offers a web interface to control the Pi Camera.

RPi Cam Control



Settings

Sharpness (-100...100), default 0:	OK
Contrast (-100...100), default 0:	OK
Brightness (0...100), default 50:	OK
Saturation (-100...100), default 0:	OK
ISO (100...800), default 0:	OK
Video Stabilisation	ON OFF
Exposure Compensation (-10...10), default 0:	OK
Exposure Mode, default 'auto':	Select option... ▼
White Balance, default 'auto':	Select option... ▼
Rotation, default 0:	Select option... ▼
Image quality (0...100), default 85:	OK

Figure 2-25 Screenshot of RPi Cam Web Interface (taken from [55])

Unlike MotionEye, it is focused on the parameters of the camera and the program *motion*, offering more control over the Pi Camera although this makes it more complicated for the end-user.

2.6 Conclusion

As explained in section 2.5 there are currently a variety of technologies used to control the Pi Camera and expand its functionality. The two most popular programs that implement the different underlying programs and technologies (*picamera-python* module, *motion*, *RaspiMJPEG*, *Apache*, etc.) to offer a user-friendly web interface oriented towards video surveillance are RPi Cam Web Interface and MotionEye.

While MotionEye is oriented towards offering a simple video surveillance system to a variety of Linux systems and configurations, RPi Cam Web Interface is oriented exclusively to the Raspberry Pi and in theory offers better performance.

3 DEVELOPMENT AND IMPLEMENTATION

In this chapter, both the process and steps in order to implement a video surveillance system are explained. As well as the necessary configuration and description of the parameters used from both a hardware and software perspective.

3.1 Description of the architecture of the video surveillance system

3.1.1 Introduction and features

Based on the characteristics of a standard commercial video surveillance system as detailed in section 2.2.3 and the objectives established in section 1.2, the video surveillance system that has been developed has the following features:

- More than one video feed from independent video cameras
- Motion detection
- Remote control of the video surveillance system
- Remote monitoring of the different video feeds
- Video/Image recording triggered by motion detection
- Email notification with images attached triggered by motion detection
- Different control configurations of the system based on administrator/user rights
- Uploading of video/image recordings to remote servers (FTP/Google Drive/Dropbox)
- Password protection of the system
- User-friendly interface

3.1.2 Design of the video surveillance system

There are a number of possible architectures that could be implemented depending on the features required and resources available. In this case, the system as shown in Figure 3-1, is composed of multiple Raspberry Pi's streaming video feeds to a **central server** which allows access to the system from another device connected to the network such as a computer or smartphone.

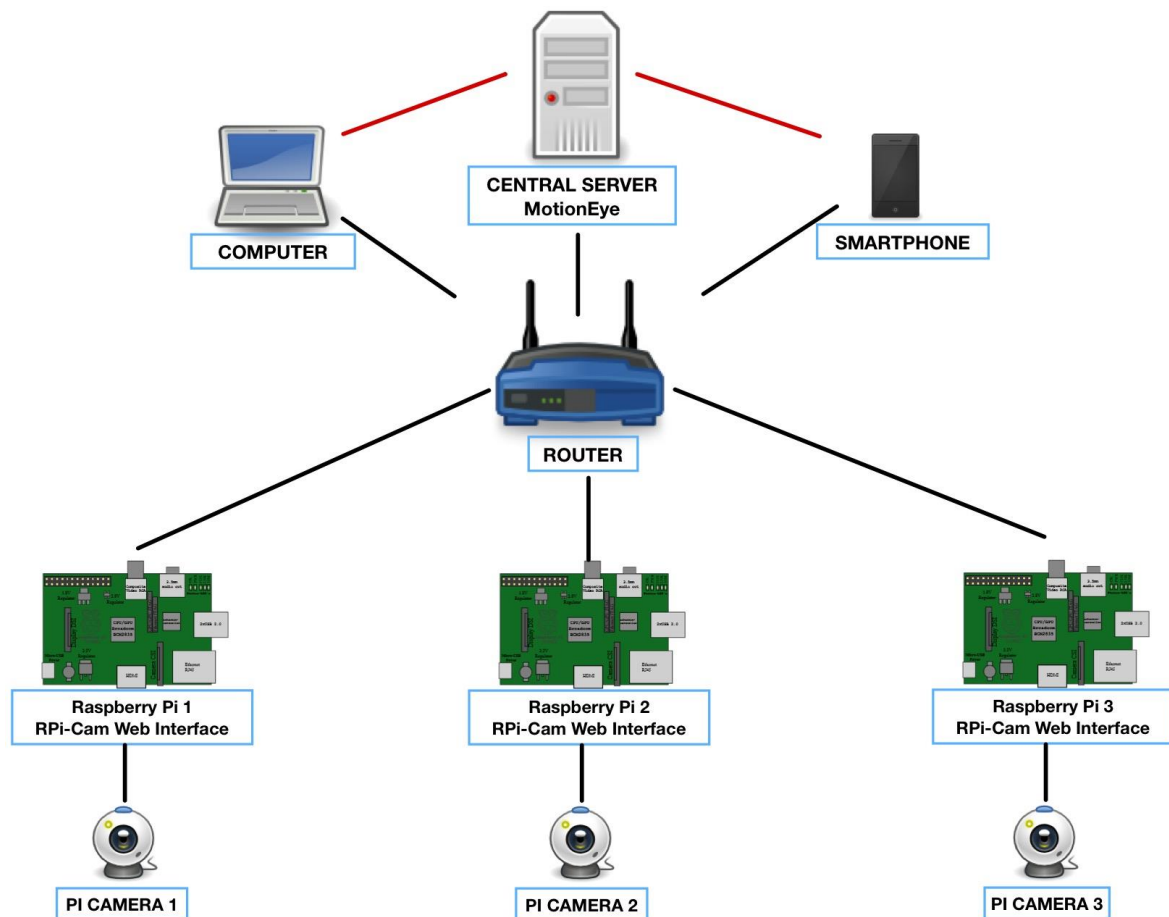


Figure 3-1 Network topology of the system: Raspberry Pi's stream their video feeds to a central server where it is presented in a unified interface (MotionEye)

There are a number of reasons for which this architecture has been adopted:

1. The program *motion* used to analyze and detect movement is very demanding on the CPU of the Raspberry Pi and is not adequate given its hardware specifications. To avoid this issue and maintain a fluid frame rate of the video feeds the program *motion* is off-loaded to the central server which has a CPU that is more suited for this task.
2. In order to have an adequate web interface, it is necessary to stream all the video feeds to a single device.
3. All the video and image files are recorded by the central server which is less demanding on the CPU's of each Raspberry Pi, which is less suited for this task.
4. Having all the media files on a hard drive of a server increases the available space and recording capability.
5. This simplifies the possibility of accessing the system from the Internet as only one port (from the central server) needs to be forwarded.

3.1.3 Description of the video surveillance system

As shown in Figure 3-1 there are different elements with different software in the video surveillance system. A description of each element is provided:

1. Raspberry Pi and Pi Camera

The first element of the system is the Raspberry Pi connected to the Pi Camera. It serves as a video camera that streams a video feed to the central server. The Raspberry Pi uses Raspbian as its operating system. Raspbian is a port of Debian which is known for being a very stable and scalable Linux operating system.

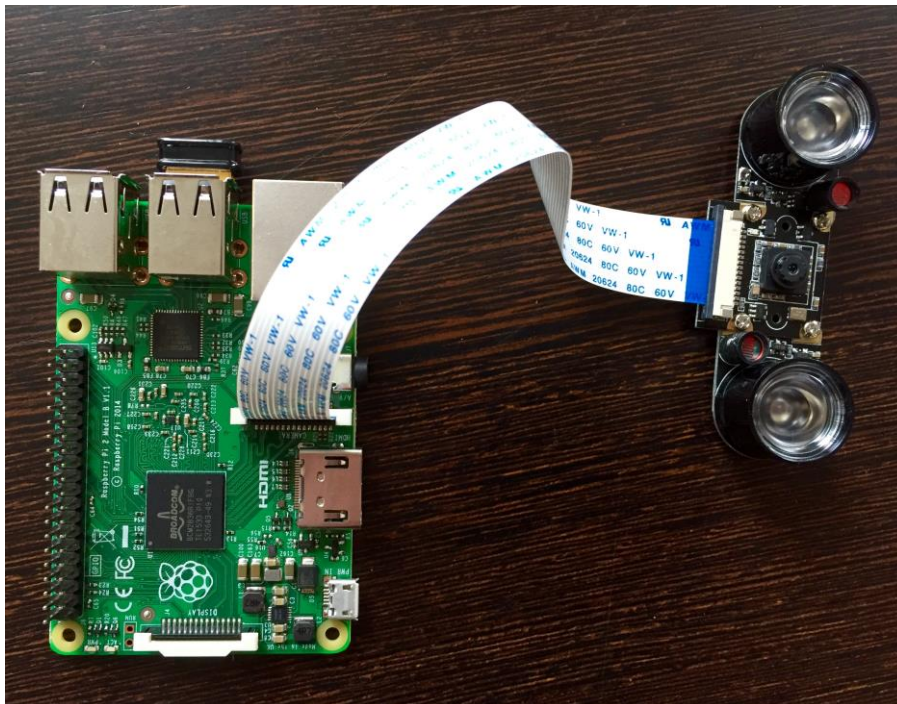


Figure 3-2 Pi Camera connected to Raspberry Pi

Although Raspbian serves as the operating system it is the program RPi-Cam Web Interface which provides a web interface as well as a video stream. The video stream is composed of continuous JPEG captures using the Pi Camera. The interface provides numerous configuration settings discussed later in section 3.2.2. It also integrates the program *motion* allowing for motion detection and video recording on the Raspberry Pi itself, but because of the reasons given in section 3.1.2, this function is done by the central server.

Details on how to install both Raspbian and RPi-Cam Web Interface are provided later in section 3.4.

2. Router

All of the elements of the system are interconnected through the router using static IP address the video feeds are streamed using MJPEG format. As we can see in Figure 3-1, the router serves as the networking device that directs the traffic and video streams of each device. It also allows forwarding the port of the central server where the interface of the video surveillance

system resides in. This would mean that the entire system can be accessed from a remote location using the Internet.

3. Central Server

The central server uses the Linux distribution Ubuntu and the program MotionEye to control the video streams and serves as a web interface for the program *motion* as discussed in section 2.5.4. The basic requirement is to have a recent Linux distribution installed (e.g. Ubuntu/Debian/Fedora etc.) evidently the better the hardware of the computer the better the program will run. For the purposes of this project a standard notebook was used (as described in section 1.3.1). For further instructions on how to install MotionEye refer to section 3.4.4.

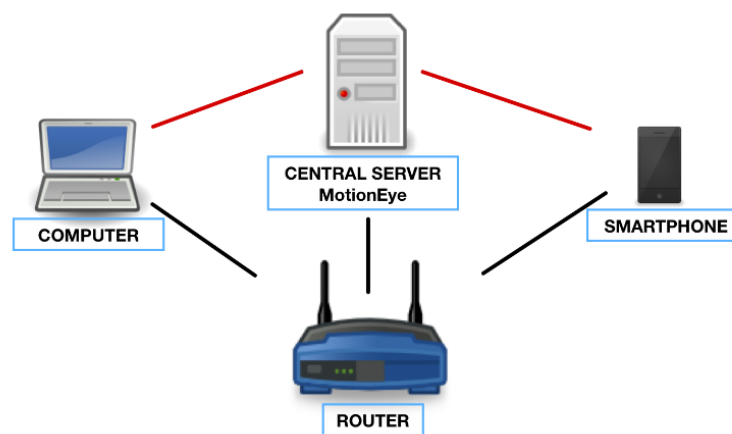


Figure 3-3 Connections of the central server running MotionEye

MotionEye allows for multiple video feeds to be added to the interface. Furthermore, it provides individual settings and options for each video feed. The most notable options that MotionEye features are motion detection, video/image recording triggered by motion, email notifications, password protected user/admin access levels and a unified web interface. A complete description of the options MotionEye offers can be found in section 3.2.3.

The video streams are provided from the Raspberry Pi's through the router to the central server. It then enables other devices (e.g. PC/smartphone) to connect to the MotionEye interface to view the video streams and configure the settings of the program.

4. External access to the video surveillance system

There are two ways of connecting to the video surveillance system, either from within the local network itself or externally from the Internet. MotionEye provides a specific port which can be forwarded from the central server to the Internet. Provided the device we are using has a modern Internet browser and knowing the IP or URL and the specific port.

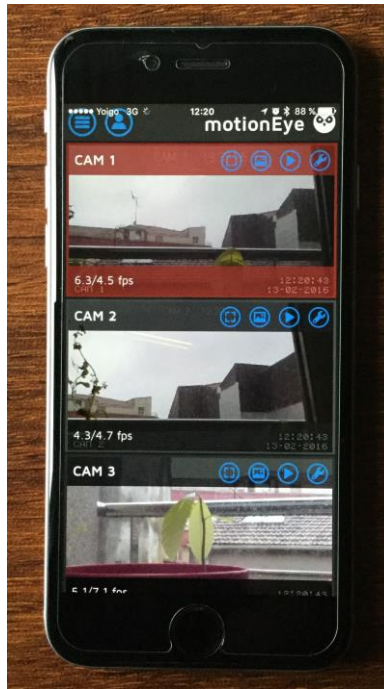


Figure 3-4 Image of a smartphone accessing the video surveillance system externally using the Internet

This feature allows the system to be accessed remotely, which is extremely useful, especially if we combine this with email notifications when motion is detected. It also allows for multiple users to access the system at the same time.

3.2 Description of the options of RPi-Cam Web Interface and MotionEye

3.2.1 Introduction

Both MotionEye and RPi-Cam Web Interface are designed to be user-friendly and intuitive. However they contain some advanced features and options. RPi-Cam Web Interface is used exclusively to stream video from the Raspberry Pi to the central server which uses MotionEye to analyze all the video streams and present them in a single web interface. In order to fully comprehend the use of both programs, each option is explained in section 3.2.2. (RPi-Cam Web Interface options) and section 3.2.3. (MotionEye options)

3.2.2 RPi-Cam Web Interface options

Having followed the installation as described in section 3.4.3, we are presented with a new installation of RPi-Cam Web interface. If we open a browser in the Raspberry Pi and navigate to <http://localhost> from the Raspberry Pi or to <http://raspberry-pi-IP:8765> from another device connected to the network, we are presented with the default set-up of RPi-Cam Web Interface as shown in Figure 3-5.

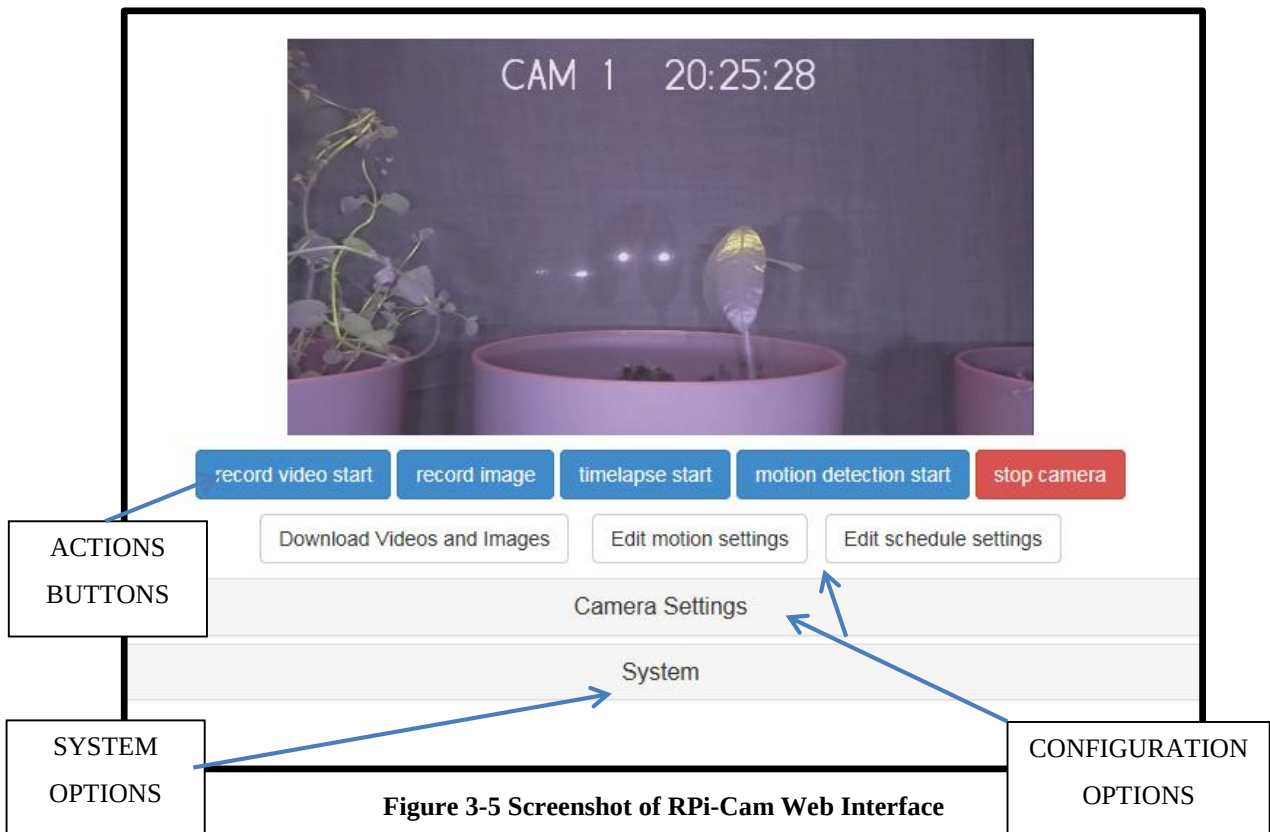


Figure 3-5 Screenshot of RPi-Cam Web Interface

The interface has a series of buttons that allow us to take certain actions, configure the camera/motion settings and the Raspberry Pi system. For the purposes of this project, only the camera settings will be modified/used as the motion analysis is carried out by MotionEye on the central server. It is important therefore not to click the button “*motion detection start*” as this will initiate motion detection on the Raspberry Pi slowing down the frame rate of the video feed.

Camera Settings

When we click on “*Camera Settings*” we are presented with the options shown in Figure 3-6 which are for the most part intuitive and self-explanatory. These options include various settings related to the quality of the image, saturation, resolution, contrast, ISO level, quality of the video stream, etc.

Resolutions:	Load Preset: Select option... Custom Values: Video res: 1280 x 0720 px Video fps: 25 recording, 25 boxing Image res: 2592 x 1944 px OK
Timelapse-Interval (0.1...3200):	3 s OK
Annotation (max 127 characters):	Text: OK Default Background: > Off ▼
Buffer (1000... ms), default 0:	0 OK
Sharpness (-100...100), default 0:	0 OK
Contrast (-100...100), default 0:	0 OK
Brightness (0...100), default 50:	50 OK
Saturation (-100...100), default 0:	0 OK
ISO (100...800), default 0:	0 OK
Metering Mode, default 'average':	Average ▼
Video Stabilisation, default: 'off'	Off ▼
Exposure Compensation (-10...10), default 0:	0 OK
Exposure Mode, default 'auto':	Auto ▼
White Balance, default 'auto':	Auto ▼
Image Effect, default 'none':	None ▼
Colour Effect, default 'disabled':	Disabled ▼ u:v = 128 : 128 OK
Image Statistics, default 'Off':	Off ▼
Rotation, default 0:	No rotate ▼
Flip, default 'none':	Both ▼
Sensor Region, default 0/0/65536/65536:	x 0 y 0 w 65536 h 65536 OK
Shutter speed (0...330000), default 0:	0 OK
Image quality (0...100), default 10:	50 OK
Preview quality (1...100) Default 10: Width (128...1024) Default 512: Divider (1-16) Default 1:	Qu: 50 Wi: 768 Di: 1 OK
Raw Layer, default: 'off'	Off ▼
Video bitrate (0...25000000), default 17000000:	25000000 OK
MP4 Boxing mode :	Off ▼
Annotation size(0-99):	50 OK
Custom text color:	Disabled ▼ y:u:v = 255 : 128 : 128 OK
Custom background color:	Disabled ▼ y:u:v = 0 : 128 : 128 OK
Watchdog, default interval 3s, errors 3	Interval 3 s Errors 3 OK
Motion detect mode :	External ▼

System

Figure 3-6 Screenshot of available options in Camera Settings in RPi-Cam Web Interface

System

In the system submenu we have the option to select the MJPEG-Stream instead of the default stream although MJPEG streaming is not compatible with all browsers unlike the default stream. The system submenu also has controls to reboot and shutdown the Raspberry Pi. It also has the option of resetting all settings to default and change the style of the interface from a white to dark background.

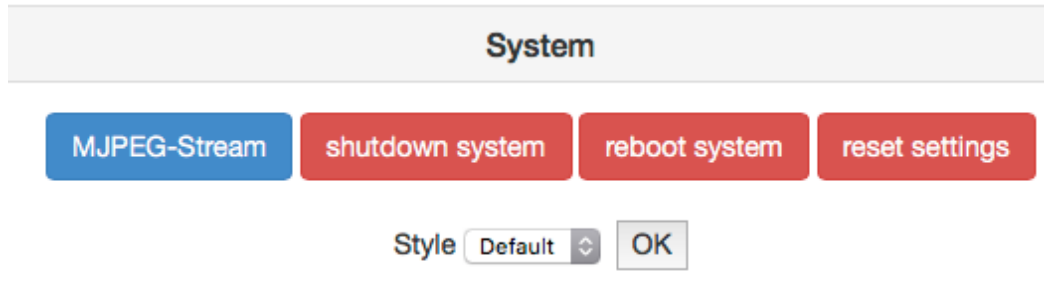


Figure 3-7 Screenshot of System submenu in RPi-Cam Web Interface

3.2.3 MotionEye options

Once MotionEye is installed following the steps in section 3.4.4, to access the program it is necessary to open a browser and navigate from the central server to <http://localhost:8765> or to <http://rapsberry-pi-IP:8765> from another device connected to the network. We are presented with a login screen as shown in Figure 3-8. To access the program the default username is “admin” and we leave the password blank.

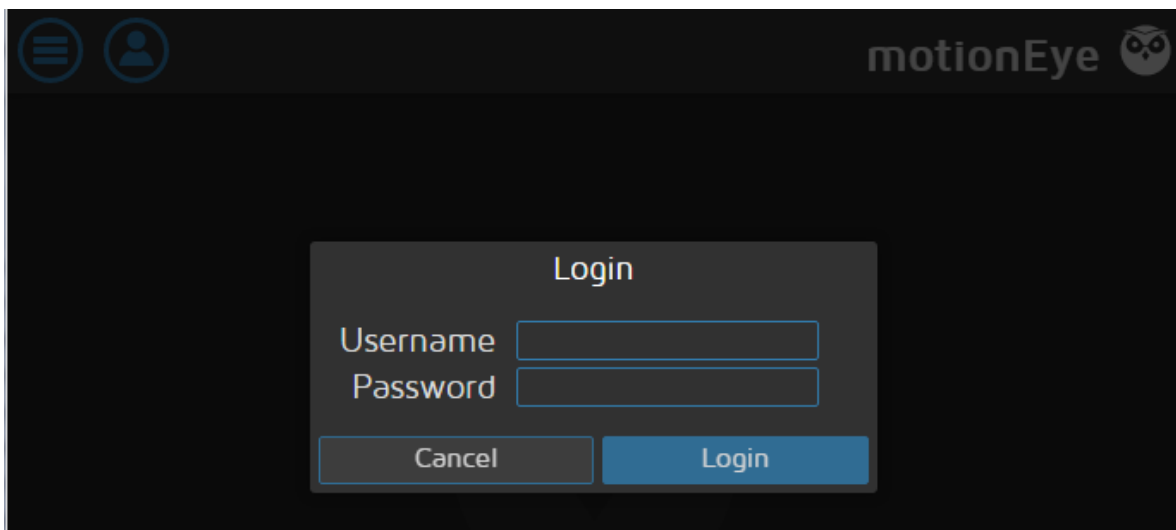


Figure 3-8 Screenshot of login screen to MotionEye

1. Preview Screen

Once we login, we access the main screen with a live feed of the cameras that are currently configured (Figure 3-9). On the upper left corner we have two icons which we see amplified in Figure 3-10: the first is to access the different configuration options and the second to switch between users.

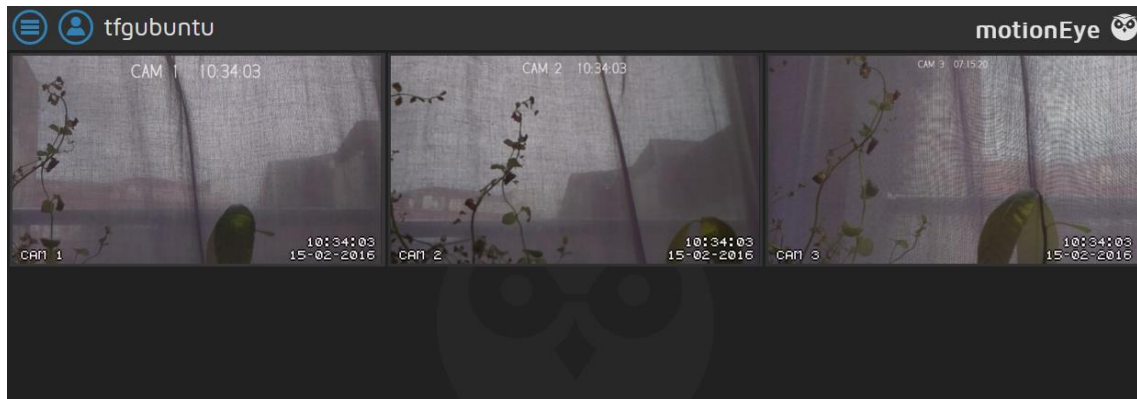


Figure 3-9 Screenshot of main screen of MotionEye



Figure 3-10 Icons of main screen

If we click the preview of one of the cameras, we can access different options and information:

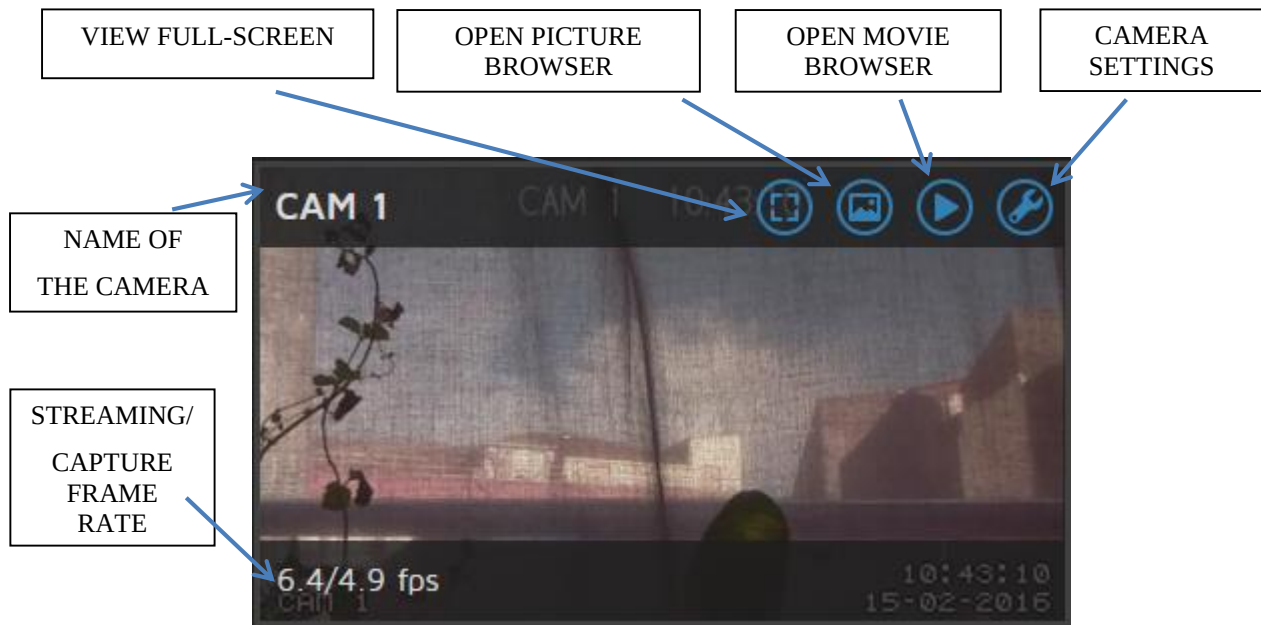


Figure 3-11 Screenshot of options presented when a camera preview is clicked

VIEW FULL-SCREEN: When clicked, the preview of the live feed of the selected camera goes to full-screen, to exit click it again.

OPEN PICTURE BROWSER: As shown in Figure 3-12, we can browse the pictures taken by each camera and choose whether to download them individually or download them all in a zipped file. The pictures are organized by day and the time they were taken. We can also configure and commence a time-lapse movie. Finally we can delete the pictures individually or all at once.

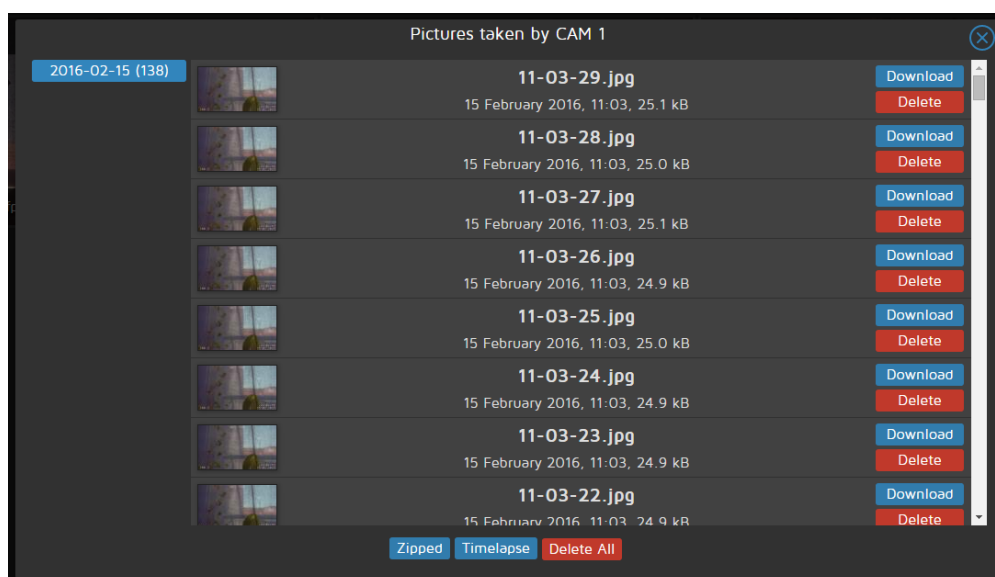


Figure 3-12 Screenshot of picture browser

OPEN MOVIE BROWSER: Movies are organized by name and time as we can see in Figure 3-13. Movies can be downloaded or deleted individually. There is also an option to delete them all at once.

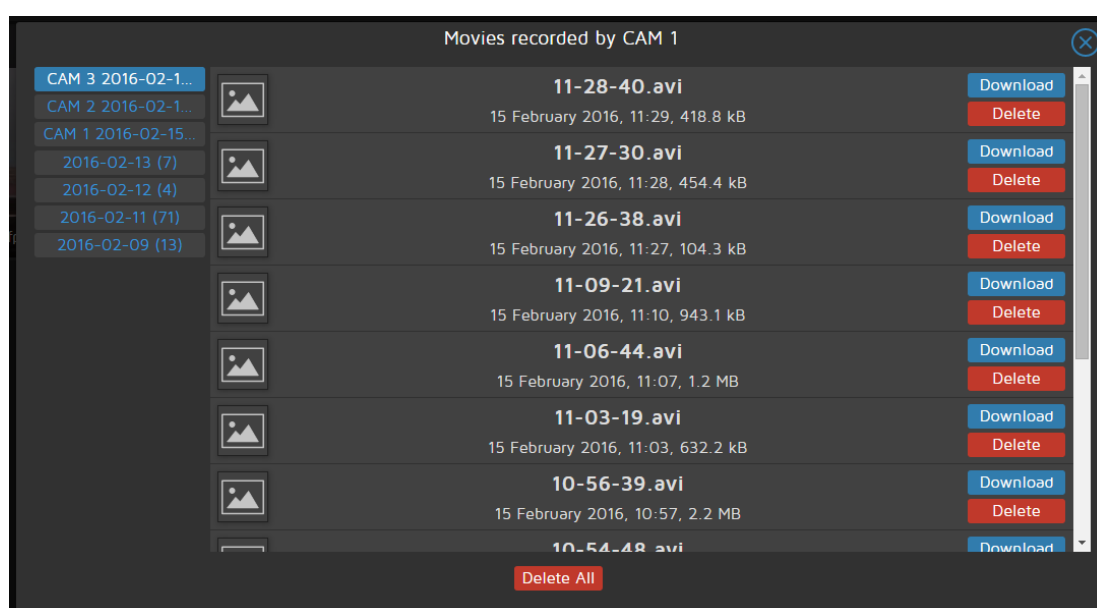


Figure 3-13 Screenshot of movie browser

CAMERA SETTINGS: Clicking on the camera settings directs us to different configuration settings depending on whether we have accessed using admin or user privileges.

2. Configuration/Options Screen

Depending on whether we have logged in as an administrator or user we are presented with different options and allowed to configure different features.

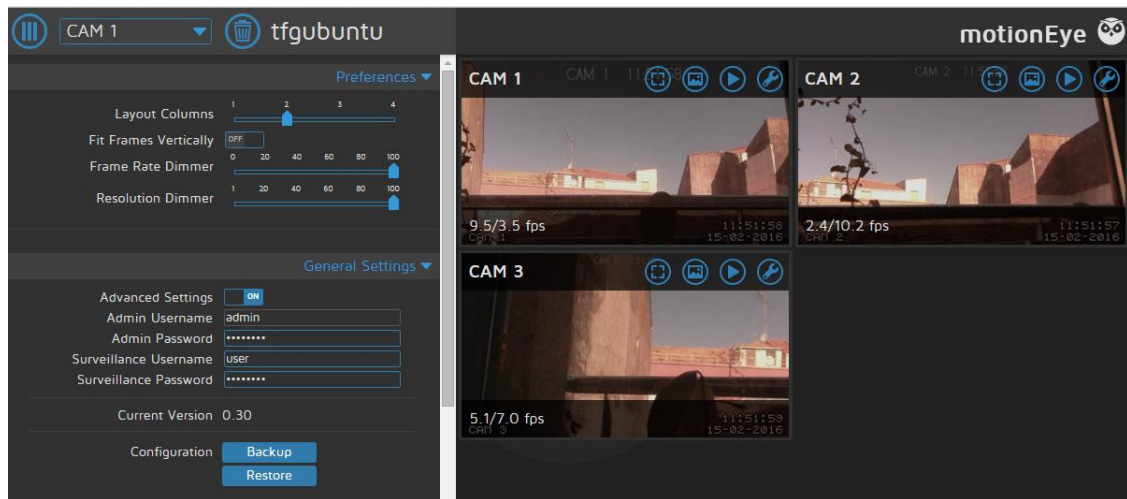


Figure 3-14 Screenshot of MotionEye configuration/option screen.

At the top left there is a dropdown menu in which we can select the different cameras and add a new camera or delete an existing one (refer to section 3.4.6 on how to add a new camera and configure it).

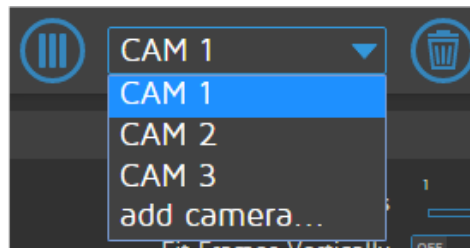


Figure 3-15 Dropdown menu to select a different camera

Depending on whether we have logged in as an administrator or a surveillance user we are presented with different options and allowed to configure different features as show in Table 3-1. Surveillance users can only configure the menu “*Preferences*” whereas administrators can configure all the available menus.

<u>Administrator</u>	<u>Surveillance User</u>
<i>Preferences</i>	<i>Preferences</i>
<i>General Settings</i>	
<i>Video Device</i>	
<i>File Storage</i>	
<i>Text Overlay</i>	
<i>Video Streaming</i>	
<i>Still Images</i>	
<i>Movies</i>	
<i>Motion Detection</i>	
<i>Motion Notifications</i>	
<i>Working Schedule</i>	

Table 3-1 Different options available to each type of user

a) **Preferences sub-menu**

From the preferences sub-menu we can configure the layout and quality of the previews of each camera.

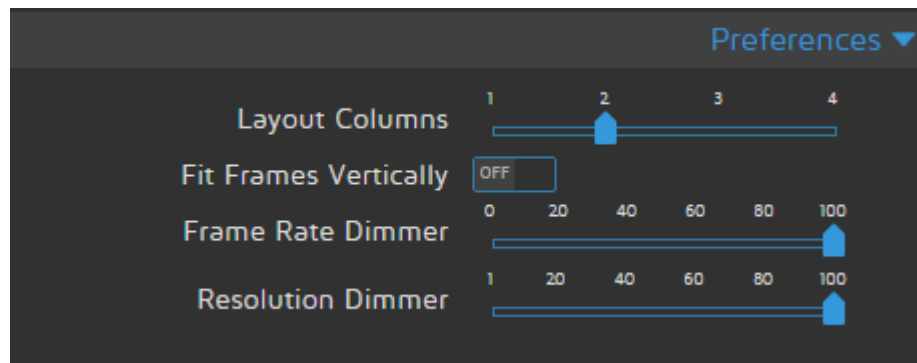


Figure 3-16 Screenshot of MotionEye Preferences sub-menu

Layout Columns	Number of video feeds available in horizontal in the preview screen.
Fit Frames Vertically	Controls whether or not the frame size is reduced to fit the window of the browser
Frame Rate Dimmer	Sets a percentage of the original frame rate of the video feed in order to save network and traffic bandwidth
Resolution Dimmer	Sets a percentage of the original resolution of the video feed in order to save network and traffic bandwidth

Table 3-2 Description of MotionEye's Preferences options

b) **General Settings sub-menu**

The general settings contains different user settings and allows to backup and restore MotionEye configuration.

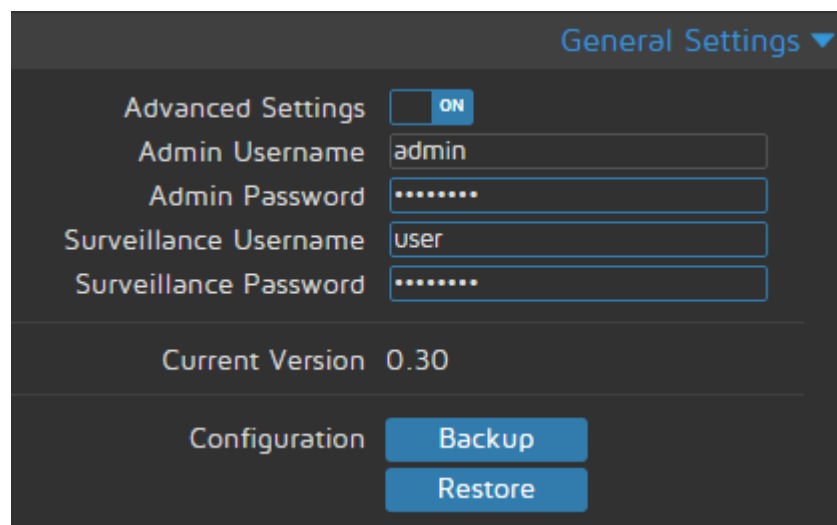


Figure 3-17 Screenshot of MotionEye General Settings sub-menu

Advanced Settings	Enables the access to advanced settings (motion settings)
Admin Username	To modify the username for the administrator
Admin Password	To modify the password for the administrator
Surveillance Username	To modify the username for the Surveillance User
Surveillance Password	To modify the password for the Surveillance User
Current Version	Specifies the current version of the MotionEye program
Backup	Creates a local file with the current configuration settings of MotionEye
Restore	Restores configuration settings from a previous backup file.

Table 3-3 Description of MotionEye's General Settings options

c) **Video Device sub-menu**

Once a camera is added, the video device sub-menu allows to change the name of the camera, rotate the image, set automatic brightness and specify the frame rate which will have a direct impact on the performance of the system.

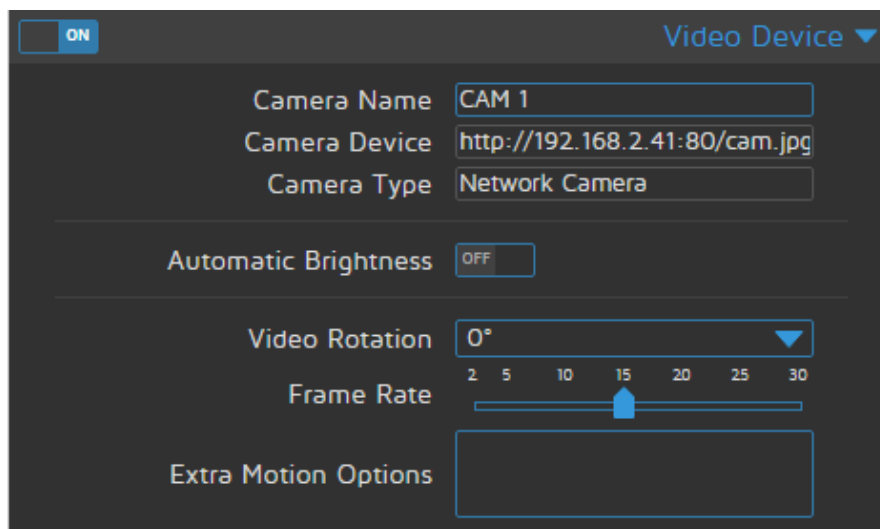


Figure 3-18 Screenshot of MotionEye Video Device sub-menu

Camera Name	Allows to specify a name for each camera
Camera Device	The address/location of the camera
Camera Type	Specifies the type of camera
Automatic Brightness	If the camera used does not support automatic brightness, there is an option to enable software automatic brightness
Video Rotation	Allows for the image to be rotated 0°, 90°, 180° or 270°
Frame Rate	Specify the desired frame rate per second of the video device, ranges from 2-30
Extra Motion Options	Allows for extra options for the motion daemon to be added

Table 3-4 Description of MotionEye's Video Device options

d) **File Storage sub-menu**

There are multiple ways of storing the media files MotionEye records, both in a local drive or using Google Drive/Dropbox. From this menu we can also configure web hooks and specific commands to be executed when motion is detected.

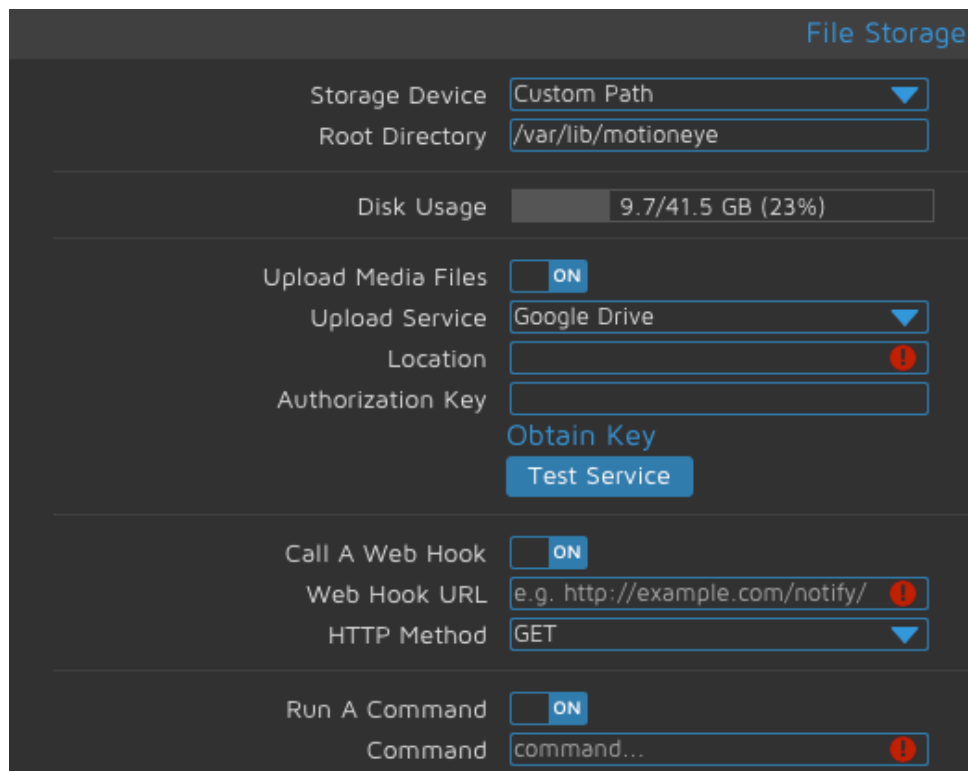


Figure 3-19 Screenshot of MotionEye File Storage sub-menu

Storage Device	Specifies which path is used to store the media files.
Root Directory	Allows to define which is the path to the folder where the media files will be stored locally.
Disk Usage	Shows how much disk space is used and how much is available in percentage
Upload Media Files	When enabled, it allows to upload the media files to either Google Drive or Dropbox
Upload Service	Select either Dropbox or Google Drive
Location	Allows to define which is the path to the folder where the media files will be stored on Google Drive/Dropbox
Authorization Key	Authorization key necessary to connect MotionEye to Google Drive/Dropbox
Test Service	Used to test whether the configuration for Google Drive/Dropbox is working correctly
Call A Web Hook	When enabled, every time a media file is created a specified URL is requested (Useful for example to send PUSH notifications)
Web Hook Url	Specifies the URL for the web hook requests
HTTP Method	Determines the URL HTTP method
Run A Command	When enabled, it allows commands to be executed every time a media file is created
Command	Specify the command to be executed. More than one command should be separated by a semi colon.

Table 3-5 Description of MotionEye's File Storage options

e) **Text Overlay sub-menu**

This feature is used to overlay specific text on each video previews. Two texts can be overlaid on the bottom, on either side of the video preview. There are three options as to what can be displayed:

- Camera Name
- Custom Text: certain special tokens are defined (%Y = year, %m = month, %d = date, %H = hour, %M = minute, %S = second, %T = HH:MM:SS, %q = frame number, \n = new line)
- Timestamp

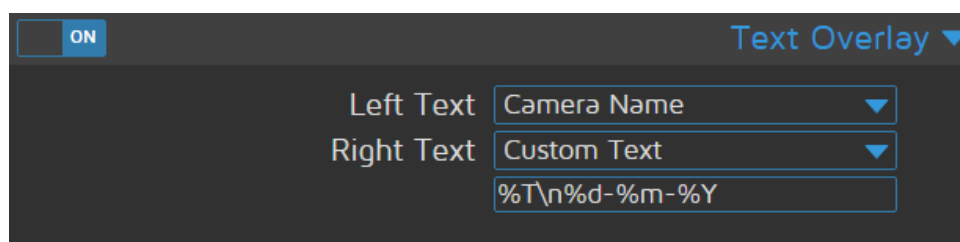


Figure 3-20 Screenshot of MotionEye Text Overlay sub-menu

Left Text	Overlays the selected text on the bottom left of the video preview
Right Text	Overlays the selected text on the bottom right of the video preview

Table 3-6 Description of MotionEye's Text Overlay options

f) **Video Streaming sub-menu**

This enables video streaming of a camera. This feature is designed for use with a camera that is locally connected to the device in which MotionEye is installed and the feed of the camera is streamed to another device.

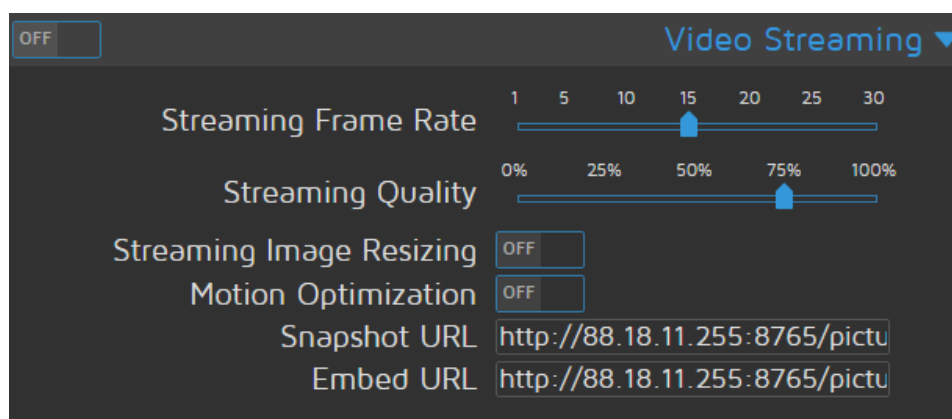


Figure 3-21 Screenshot of MotionEye Video Streaming sub-menu

Streaming Frame Rate	Specifies the number of frames per second to be streamed
Streaming Quality	Used to set the percentage quality of the video image to be streamed
Streaming Image Resizing	When enabled, the images of the video stream are resized before sent to the browser in which it is to be viewed
Motion Optimization	To reduce bandwidth, when no motion is detected the video stream is of lower quality and automatically increases when motion is detected
Snapshot URL	Used to specify a URL in which the most recent JPEG image taken is displayed
Embed URL	Provides a HTML document in which the camera frame is embedded

Table 3-7 Description of MotionEye's Video Streaming options

g) **Still Images sub-menu**

This option allows MotionEye to record images with certain options. The capture of images can be triggered either by motion detection, all frames or every given seconds.

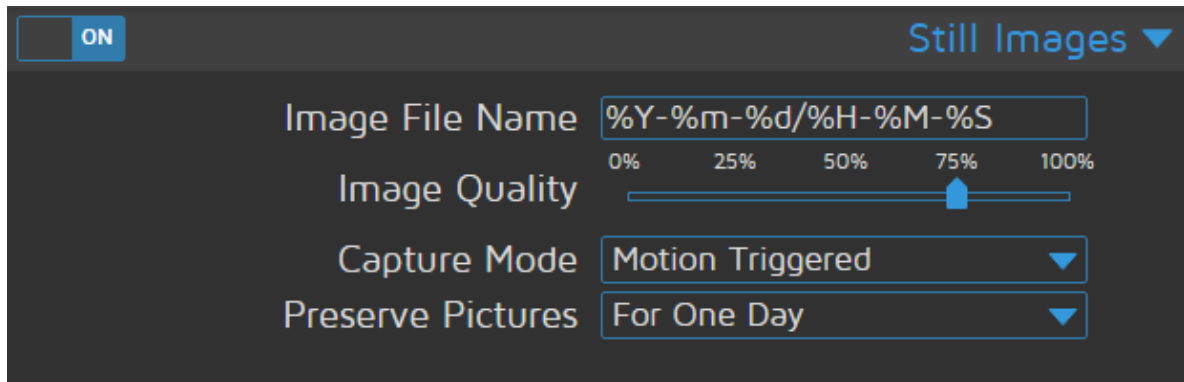


Figure 3-22 Screenshot of MotionEye Still Images sub-menu

Image File Name	Used to set a pattern for the name of the image files to be saved. Certain special tokens can be used (%Y = year, %m = month, %d = date, %H = hour, %M = minute, %S = second, %T = HH:MM:SS, %q = frame number, %v = event number)
Image Quality	Allows to specify a percentage for the quality of the images that will be saved.
Capture Mode	Dropdown menu which allows for one of three options: • Motion Triggered: An image is taken when motion is detected • Interval Snapshots: An image is taken every “x” seconds (from 1 to 86400 seconds can be selected) • All Frames: Records every frame in individual files
Preserve Pictures	Pictures can be preserved indefinitely or deleted automatically after a certain amount of days which is specified in this parameter.

Table 3-8 Description of MotionEye’s Still Images options

h) **Movies sub-menu**

The movies sub-menu when enabled allows to record movies in MPEG format and is identical to the images sub-menu except that there is an option to automatically delete movies that surpass a given length.

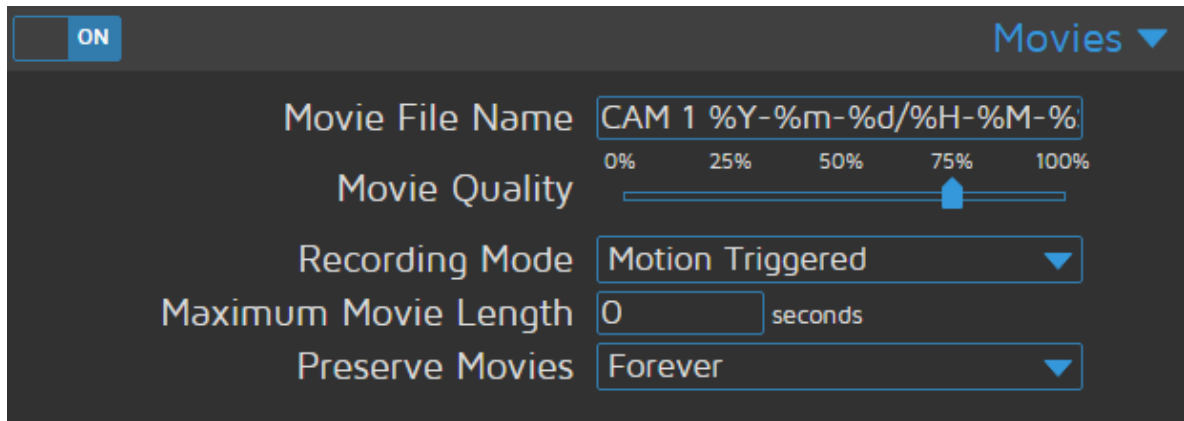


Figure 3-23 Screenshot of MotionEye Movies sub-menu

Movie File Name	Used to set a pattern for the name of the movie files to be saved. Certain special tokens can be used (%Y = year, %m=month, %d=date, %H = hour, %M = minute, %S = second, %T = HH:MM:SS, %q = frame number, %v = event number)
Movie Quality	Allows to specify a percentage for the quality of the movies that will be saved
Recording Mode	Dropdown menu which allows for one of two options: • Motion Triggered: A movie is taken when motion is detected • Continuous Recording: Records the video feed into a continuous movie
Maximum Movie Length	Used to specify the maximum length of a movie. If there is motion being detected and is configured to trigger video recording, a new movie file is created
Preserve Movies	Movies can be preserved indefinitely or deleted automatically after a certain amount of days which is specified in this parameter

Table 3-9 Description of MotionEye's Movies options

i) **Motion Detection sub-menu**

Motion detection is detected by using the program *motion*. By enabling the Motion Detection feature we can configure a number of parameters that when met will be considered as movement and trigger the detection and other actions such as sending notifications, recording a video, etc.

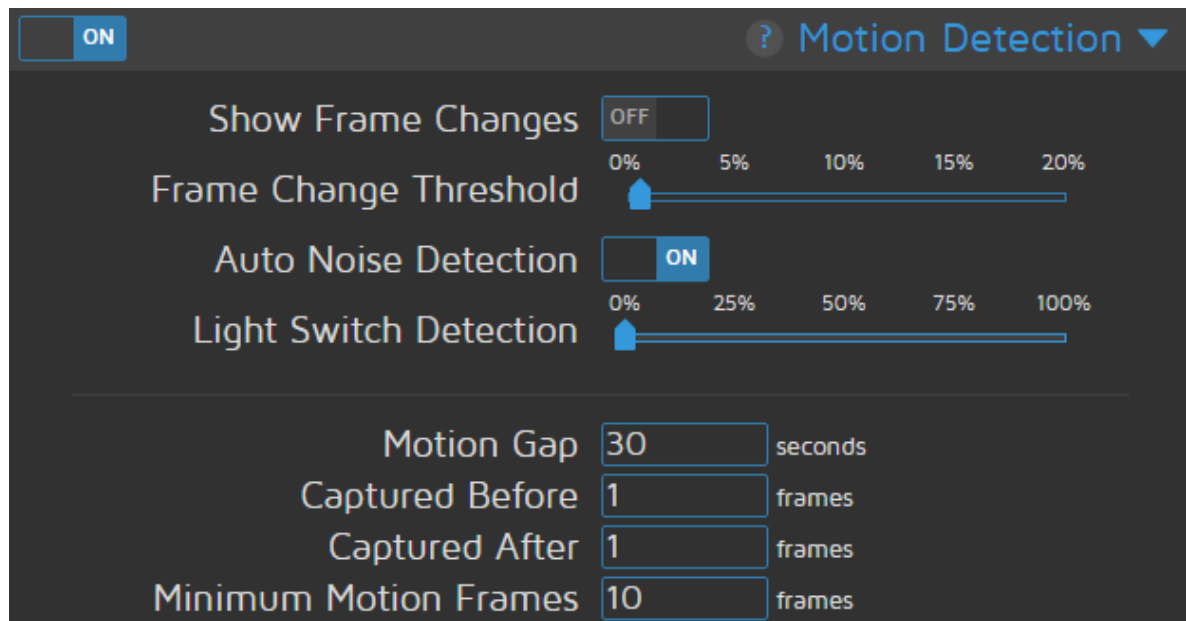


Figure 3-24 Screenshot of MotionEye Motion Detection sub-menu

Show Frame Changes	When enabled, the number of pixels that changes is displayed and the area where the change is detected is highlighted
Frame Change Threshold	Determines what percentage of the total pixels has to change to be considered as actual “movement”
Auto Noise Detection	Noise level is automatically adjusted when this option is enabled. If not enabled it must be adjusted manually
Light Switch Detection	Used to set the percentage of sudden change that should be considered as a light change and not movement. 0% disables this option
Motion Gap	Establishes the number of seconds in which no motion is detected that is needed to stop the motion even
Captured Before	Used to determine the number of frames to be included in the motion event before motion was detected. This frames are also included in the movie recording
Captured After	Used to determine the number of frames to be included in the motion event after motion was detected. This frames are also included in the movie recording
Minimum Motion Frames	Established the minimum amount of successive frames in which motion was detected to actually be considered as a movement

Table 3-10 Description of MotionEye’s Motion Detection options

j) **Motion Notifications sub-menu**

This feature will enable certain notifications to be sent when motion is detected. There are multiple types of possible notifications that can be configured, including an email with attached images in which motion was detected to web hooks or specific commands.

Figure 3-25 Screenshot of MotionEye Motion Notifications sub-menu

Send An Email	When enabled an email will be sent to the specified addresses when motion is detected
Email Addresses	The email addresses (separated by a comma) specified in this section will receive an email when motion is detected
SMTP Server	This section is to specify the IP address of the SMTP Server
SMTP Port	This section is to specify the port used by the SMTP Server
SMTP Account	This section is to specify the SMTP account
SMTP Password	This section is to specify the password of the SMTP account
Use TLS	Enable if required
Attached Pictures Time Span	Specifies the time interval in which to search for images where motion was detected and attach them
Call A Web Hook URL	Specifies the URL for web hook requests
HTTP Method	Determines the URL HTTP method
Run A Command	When enabled it allows commands to be executed every time a media file is created
Command	Specify the command to be executed. More than one command should be separated by a semi colon

Table 3-11 Description of MotionEye's Motion Notifications options

k) Working Schedule sub-menu

This feature is used to determine a particular daily schedule per week in which motion detection is enabled. This allows to configure motion detection only in a particular timeframe.

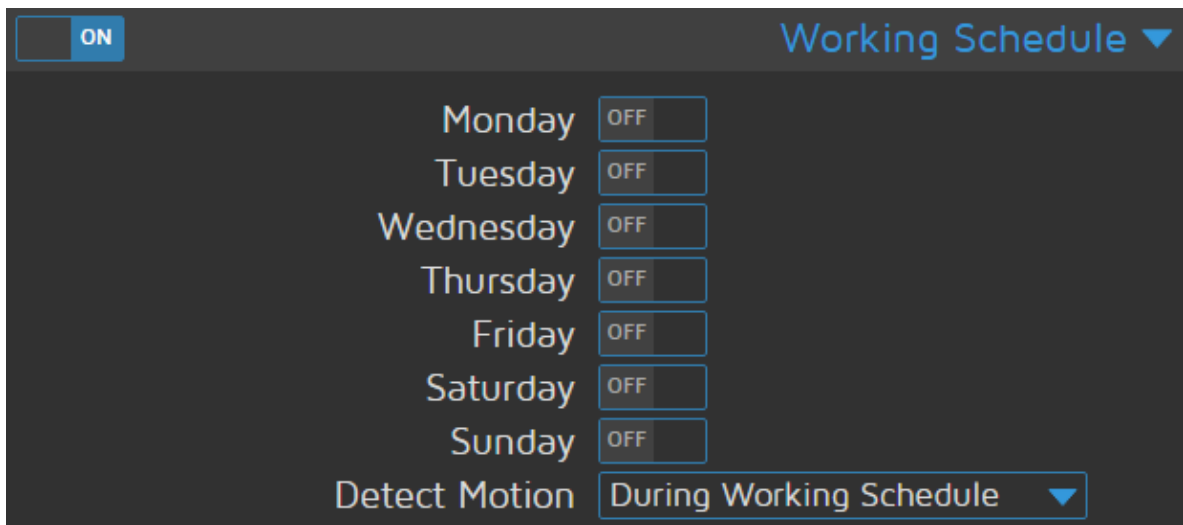


Figure 3-26 Screenshot of MotionEye Working Schedule sub-menu

Monday Tuesday Wednesday Thursday Friday Saturday Sunday	When enabled, allows to determine a start and end time in which motion detection is enabled. Specified times are in 30 minute intervals
Detect Motion	Allows to specify whether motion detection should work during the specified timeframes or outside those timeframes

Table 3-12 Description of MotionEye's Working Schedule options

3.2.4 Modified MotionEye options

If the how-to guide described later, in section 3.4.5 is implemented, MotionEye's interface is modified to display two new icons (as shown in Figure 3-27) that enables the ability to reboot and shutdown each Raspberry Pi from within the MotionEye interface.

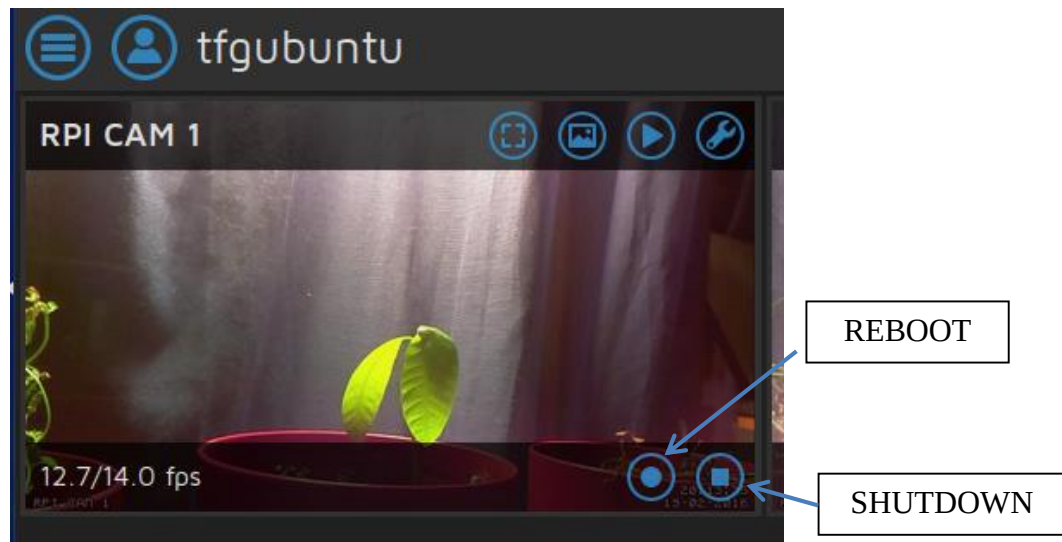


Figure 3-27 Screenshot of modified MotionEye interface

3.3 Description of the selected configuration of the system

3.3.1 Introduction

MotionEye is designed to work on a wide range of systems; therefore, there are multiple possible configurations. To use it with the hardware of this project and meet the intended objectives, a determined configuration will have to be designed, tested and adopted.

The main limit to the Raspberry Pi hardware is the CPU, given its price and characteristics it can certainly be considered as more than adequate. However, when attempting to stream a video feed at the highest possible resolution and best quality, the frame rate we obtain is very low. Therefore, certain concessions will have to be made.

Given the ease of use and features offered by MotionEye, it will be used on the central server as to control and display the video surveillance system. However, for the actual streaming of the video from the Raspberry Pi's to the central server, there are a variety of ways in which this can be done.

3.3.2 Improvements and modifications to MotionEye

In order to fulfill the objectives of the video surveillance system it was determined that there was a concern with regard to the current available version of MotionEye (version 0.3): the lack of being able to control the Raspberry Pi from the MotionEye interface.

Using `ssh` [56] to connect to each Raspberry and a series of modifications to the MotionEye code we are able to issue commands from the MotionEye interface to each Raspberry. The details and how this is achieved is described in section 3.4.5.

3.3.3 Raspberry Pi: RPi-Cam Web Interface

During the testing phase detailed in section 4.1.2, it was determined that RPi-Cam Web Interface provided a higher quality video stream than MotionEye. Therefore, RPi-Cam Web Interface was selected and installed on the three Raspberry Pi's.

Once installed as described in section 3.4.3, the following settings are modified from the default values in the "Camera Section", based on the tests detailed in section 4.1.2. The following

configuration shown in Table 3-13, offers a high quality video stream at approximately 18 frames per second and a resolution of 1280 x 720. The final configuration can be seen in Figure 3-28.

RESOLUTION PRESET		1280 x 720 “HD-Ready”
ANNOTATION		The default text is erased (the text overlay will be done by MotionEye on the Central Server)
FLIP		Both (given the way the Pi Camera was attached, it was necessary to rotate the image both vertically and horizontally)
IMAGE QUALITY		50
PREVIEW	QUALITY	50
	WIDTH	768
	DIVIDER	1
VIDEO BITRATE		25000000

Table 3-13 Recommended settings for RPi-Cam Web Interface

Resolutions:	Load Preset: Select option... Custom Values: Video res: 1280 x 0720 px Video fps: 25 recording, 25 boxing Image res: 2592 x 1944 px OK
Timelapse-Interval (0.1...3200):	3 s OK
Annotation (max 127 characters):	Text: OK Default Background: > Off
Buffer (1000... ms), default 0:	0 OK
Sharpness (-100...100), default 0:	0 OK
Contrast (-100...100), default 0:	0 OK
Brightness (0...100), default 50:	50 OK
Saturation (-100...100), default 0:	0 OK
ISO (100...800), default 0:	0 OK
Metering Mode, default 'average':	Average
Video Stabilisation, default: 'off'	Off
Exposure Compensation (-10...10), default 0:	0 OK
Exposure Mode, default 'auto':	Auto
White Balance, default 'auto':	Auto
Image Effect, default 'none':	None
Colour Effect, default 'disabled':	Disabled u:v = 128 : 128 OK
Image Statistics, default 'Off':	Off
Rotation, default 0:	No rotate
Flip, default 'none':	Both
Sensor Region, default 0/0/65536/65536:	x: 0 y: 0 w: 65536 h: 65536 OK
Shutter speed (0...330000), default 0:	0 OK
Image quality (0...100), default 10:	50 OK
Preview quality (1...100) Default 10: Width (128...1024) Default 512: Divider (1-16) Default 1:	Qu: 50 Wi: 768 Di: 1 OK
Raw Layer, default: 'off'	Off
Video bitrate (0...25000000), default 17000000:	25000000 OK
MP4 Boxing mode :	Off
Annotation size(0-99):	50 OK
Custom text color:	Disabled y:u:v = 255 : 128 : 128 OK
Custom background color:	Disabled y:u:v = 0 : 128 : 128 OK
Watchdog, default interval 3s, errors 3	Interval 3 s Errors 3 OK
Motion detect mode :	External

System

Figure 3-28 Screenshot of configuration used in RPi-Cam Web Interface

3.3.4 Central server: MotionEye

As was discussed in section 2.5.4, MotionEye can be installed in a variety of Linux distributions. Currently, the most popular and widespread Linux distribution is Ubuntu [57]. For this project Ubuntu 14.04 LTS, MotionEye version 0.3 and three Raspberry Pi's with a Pi Camera were used. Both a screenshot of each parameter and explanation of why those parameters were selected is detailed in this section.

PREFERENCES

Given that three Raspberry Pi's will be used, three layout columns are selected to able to see the three preview windows in a single row. Both Frame Rate Dimmer and Resolution Dimmer are maintained at 100% so as not to downgrade the quality of the video feed.

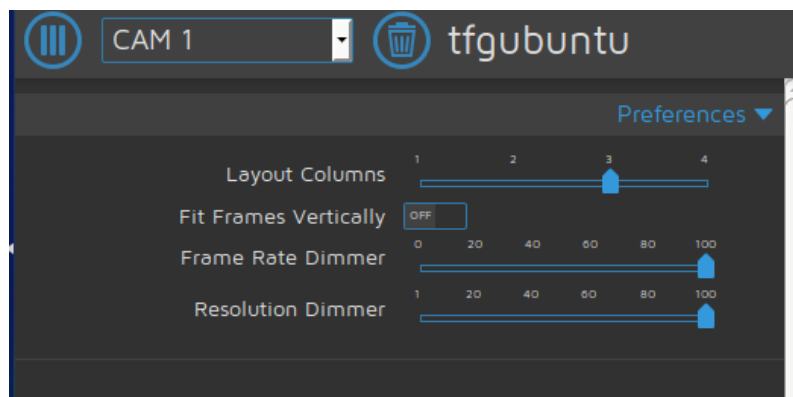


Figure 3-29 Screenshot of MotionEye Preferences

GENERAL SETTINGS

The first setting to be enabled is “Advanced Settings” so we can configure the rest of the system. Both admin and user passwords are established to increase the overall security of the system.

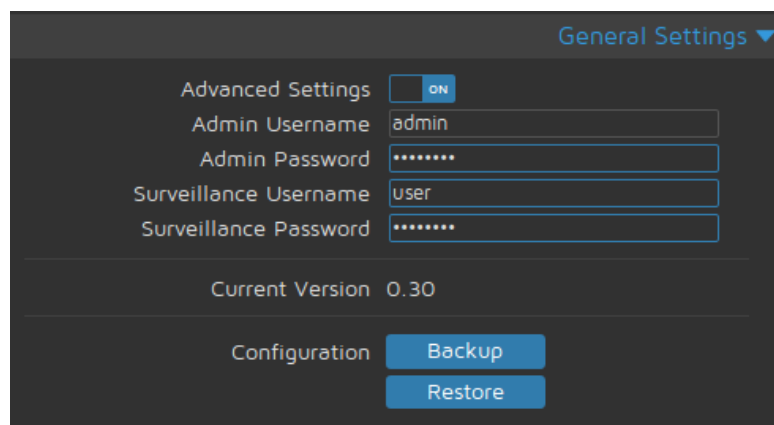


Figure 3-30 Screenshot of MotionEye General Settings

VIDEO DEVICE

Given that we will use three Raspberry Pi's, three separate Video Devices will be added following the process described in section 3.4.6. In Camera Name we put a different name for each camera to differentiate each one. In this case they were named: RPI CAM 1, RPI CAM 2 and RPI CAM 3. The tests described in section 4 showed that 15 fps (frames per second) was an adequate frame rate when the system had up to three Raspberry Pi's.

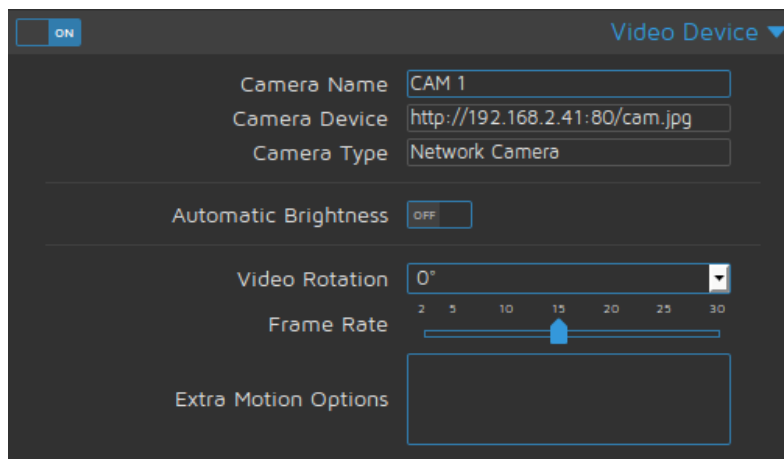


Figure 3-31 Screenshot of MotionEye Video Device

FILE STORAGE

Although there are a variety of options for where and how to store the files recorded by MotionEye, in this system the files will be stored locally and the default directory was not changed.

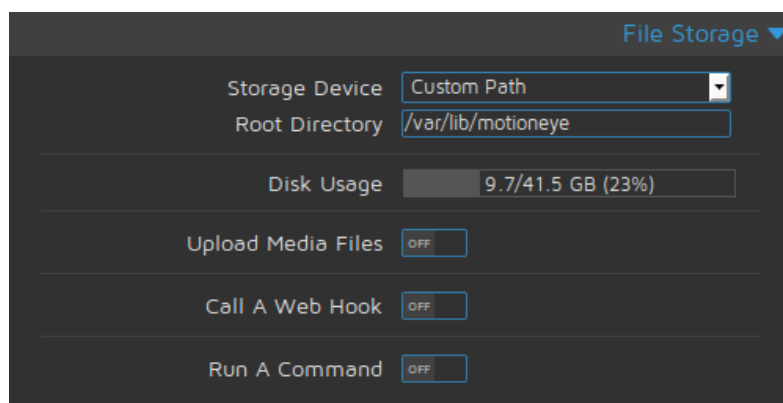


Figure 3-32 Screenshot of MotionEye File Storage

TEXT OVERLAY

In order to ensure each video feed is properly distinguished, the text overlay option is enabled with the left text set to display the Camera Name. The right text was set to display the time using a custom format in two lines. The first line displays the time while the second line the date.

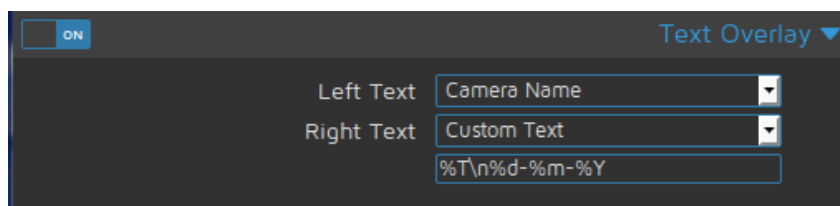


Figure 3-33 Screenshot of MotionEye Text Overlay

VIDEO STREAMING

Given that the MotionEye interface can be accessed by any device that has access to the server, it is simple to view all three streams and control them. Enabling this option would be redundant and consume unnecessary resources. Therefore, it is not enabled.

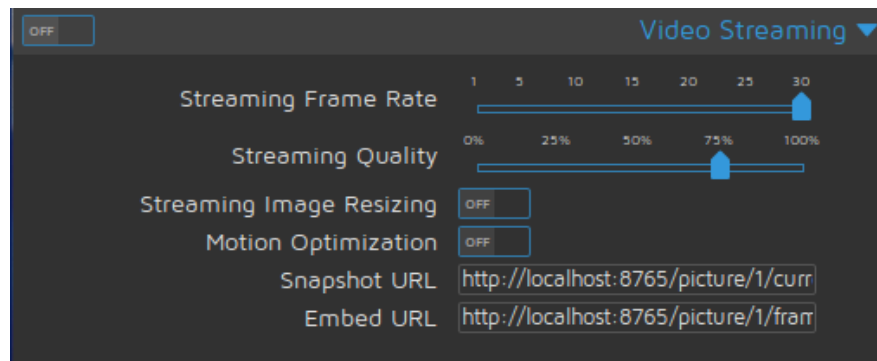


Figure 3-34 Screenshot of MotionEye Video Streaming

STILL IMAGES

Although the text overlay is already sufficient to distinguish both the Camera that took the image and at what time, the Image File Name also offers this option. In the Image File Name, the name of each camera is put followed by a date and time naming format. The quality of the image is set to 50% which is sufficient. It is set to capture an image when it detects motion and preserve the pictures until manually deleted.

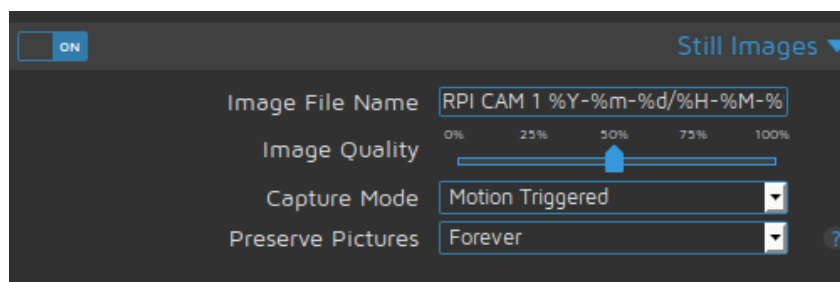


Figure 3-35 Screenshot of MotionEye Still Images

MOVIES

The same naming scheme used in Images is adopted in Movies. However, the movie quality is lowered to 37% as it consumes a lot of CPU cycles. It is also set to record when it detects motion. No maximum movie length is set (0 = disabled) and movies are set to be preserved until manually deleted.

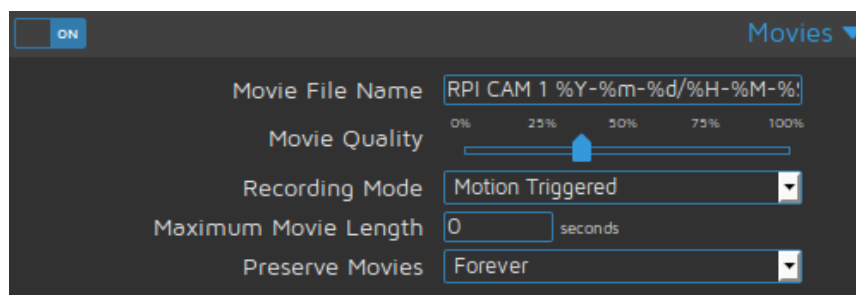


Figure 3-36 Screenshot of MotionEye Movies

MOTION DETECTION

Proper configuration of Motion Detection is crucial. In order to properly calibrate the threshold, it is recommendable to enable “Show Frame Changes”. Once properly configured it is disabled as it has an impact on performance. The Frame Change Threshold was set to 1%, therefore it is very sensitive to any changes. The Light Switch Detection is disabled whereas Auto Noise Detection is enabled.

Motion Gap was set at 30 seconds, frames captured before and after the motion event was set at 1 frame. The minimum motion frames to be considered as movement was set at 10 frames. This configuration proved to work very well during testing, detecting even the slightest movement.

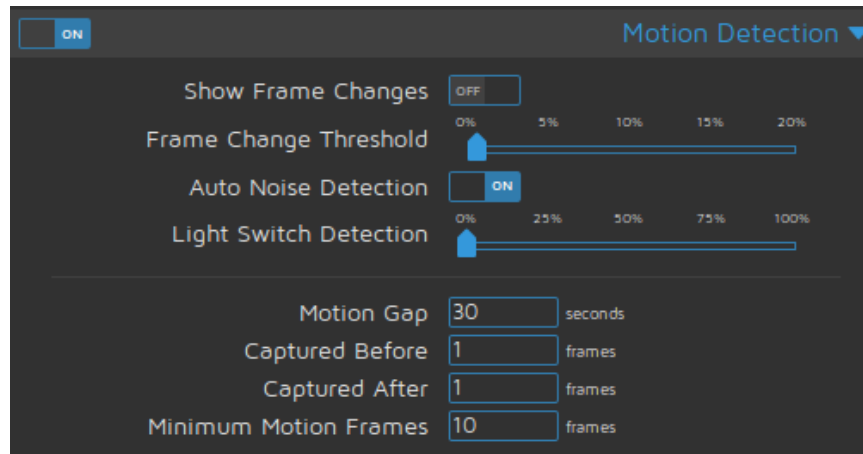


Figure 3-37 Screenshot of MotionEye Motion Detection

MOTION NOTIFICATIONS

The system was configured to send an email using Gmail’s SMTP email settings [58] to send an email when it detects movement. It was configured to send one image of the movement detected as an attachment in the email it sends.

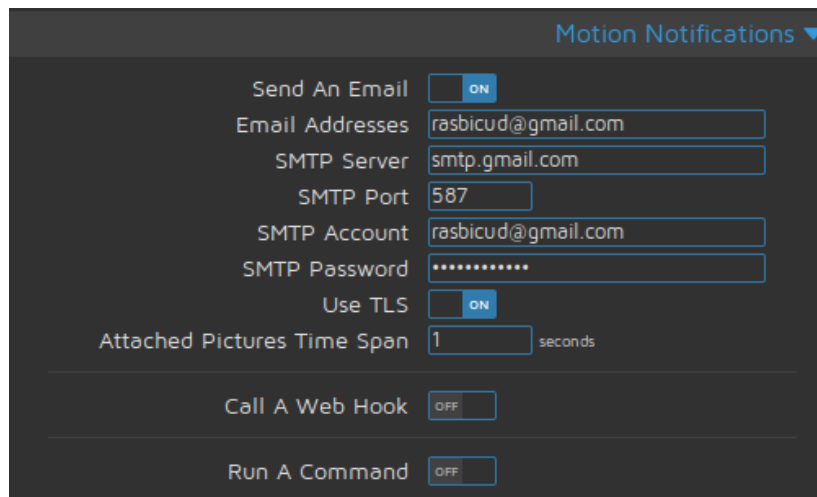


Figure 3-38 Screenshot of MotionEye Motion Notifications

WORKING SCHEDULE

For this system, the Working Schedule was not enabled. It is configured to function in the same manner all the time.



Figure 3-39 Screenshot of MotionEye Working Schedule

3.3.5 Network

Although the tests carried out in section 4.1.3 clearly show that there is a downgrade in the quality of the video stream when using a Wi-Fi connection, compared to an Ethernet connection, the advantages Wi-Fi offers in terms of mobility are more desirable. Evidently, if the system were purely performance focused, the Raspberry Pi's would be connected to the network using Ethernet. However, given that each Raspberry Pi has a battery and a Wi-Fi connection they are very portable and can be placed anywhere within the range of the router's Wi-Fi signal.

For this system, each Raspberry Pi was configured with a static IP. Given that MotionEye and its configuration requires knowing the IP of each Raspberry Pi, DHCP was not used as this would mean the IP is susceptible to change.

There is also the possibility of forwarding the port MotionEye uses to the Internet so it can be controlled from another network. In this system, the port being used was forwarded and tested as shown in Figure 3-4.

3.4 Description of how to install the necessary programs and the improvements implemented

3.4.1 Introduction

In order to replicate the video surveillance system implemented, a how-to guide is provided. This includes the installation of the OS on each Raspberry Pi and the RPi-Cam Web Interface to stream the video feed. An installation guide on how to install the MotionEye software on the central server is also provided in section 3.4.4. Also, a guide on how to modify and improve MotionEye is detailed in section 3.4.5.

3.4.2 How to install Raspbian (Raspberry Pi OS)

This guide is based on [59] and adapted to implementing the designed video surveillance system.

1. Download the latest version of the NOOBS Offline and Network Install (as shown in Figure 3-40) software that includes Raspbian from [60].

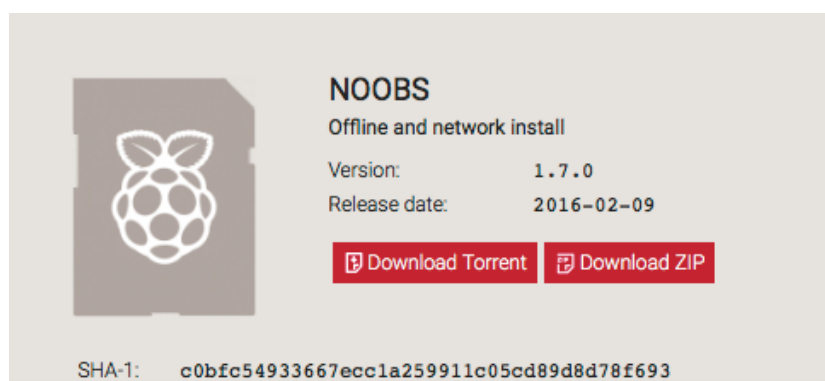


Figure 3-40 Screenshot of link to NOOBS software

2. Format a MicroSD card of at least 8GB to FAT32 format.
3. Once the download is completed unzip the NOOBS file and drag the unzipped files to the MicroSD card as shown in Figure 3-41.

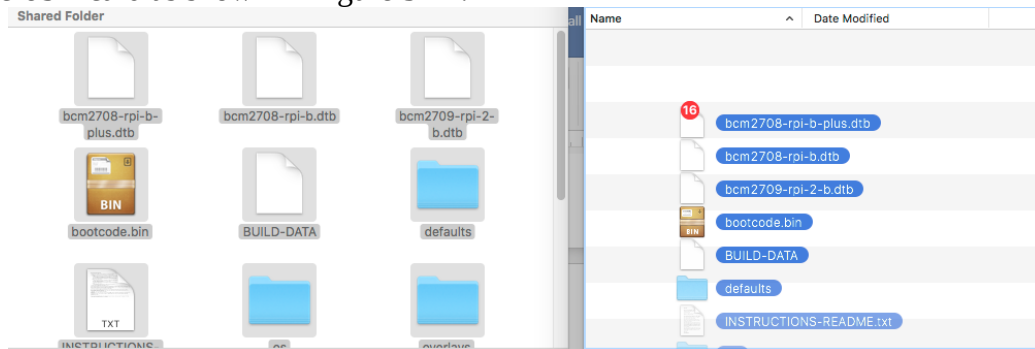


Figure 3-41 Screenshot of unzipped NOOBS files

4. Eject the MicroSDcard and insert it into the Raspberry Pi, connect a computer monitor, keyboard, mouse and power source.
5. Wait until it boots to the NOOBS installation screen, select to install Raspbian (as shown in Figure 3-42) and wait until the installation is completed (approximately: 17 minutes).

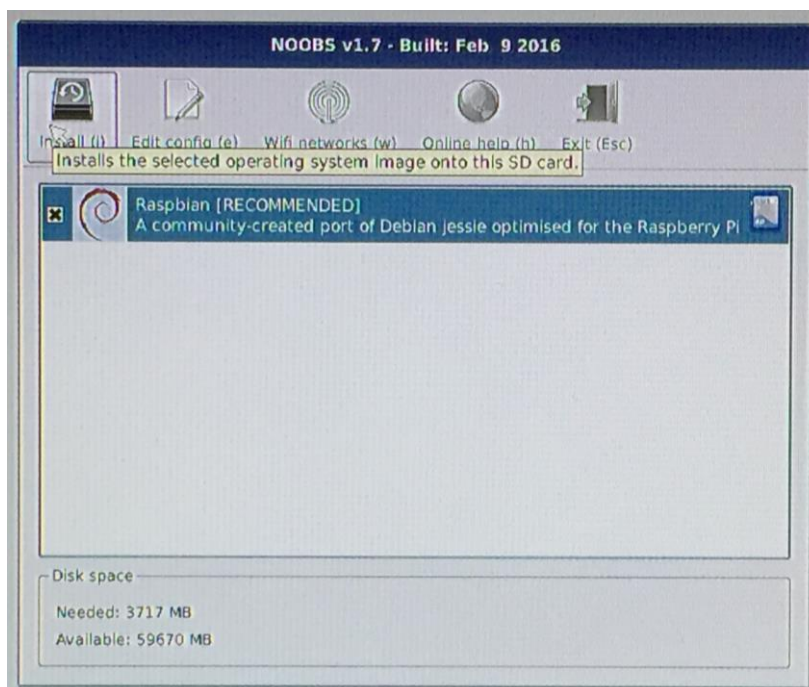


Figure 3-42 Screenshot of NOOBS menu

6. Once it completes the installation, it will reboot and we will have a fully-fledged operating system installed on the Raspberry Pi. Before installing RPi-Cam Web Interface, it is necessary to configure the following:
 - a. Wi-Fi/Ethernet: the menu to configure the network connection is on the top right as we can see in Figure 3-43.

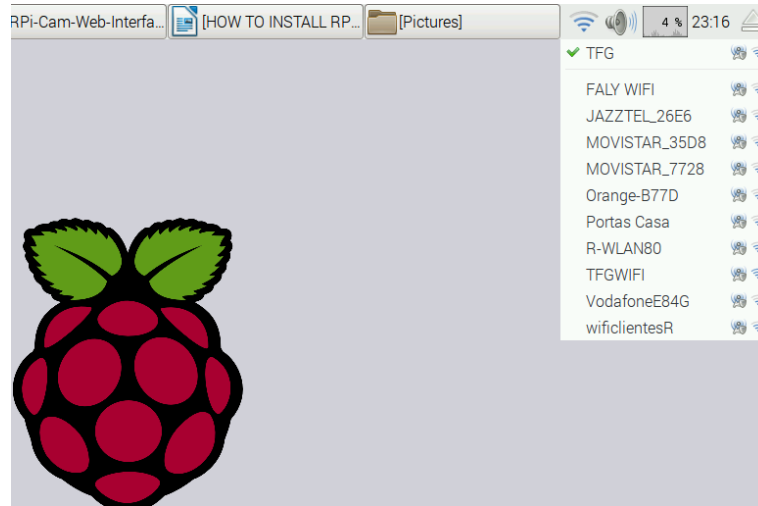


Figure 3-43 Screenshot of Network settings in Raspbian

- b. Locale, Keyboard and Time zone: these settings (as shown in Figure 3-44) can be found under Menu → Preferences → Raspberry Pi Configuration:

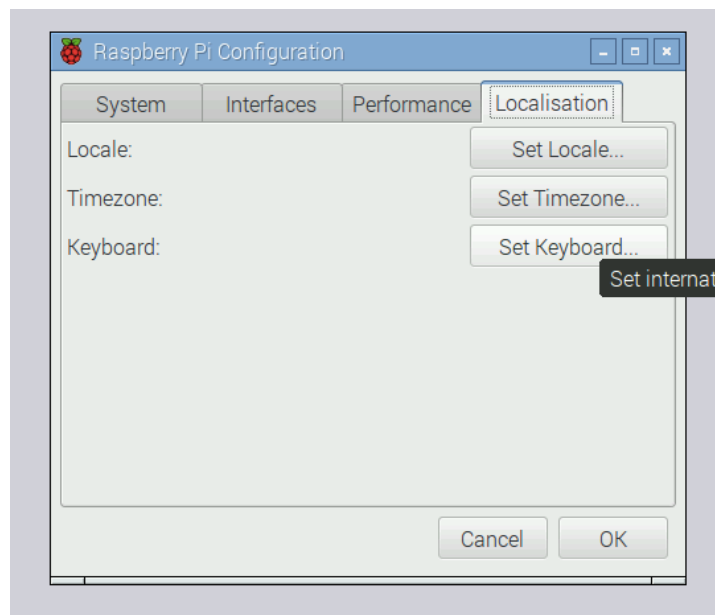


Figure 3-44 Screenshot of Raspberry Pi localisation settings

- c. Pi Camera: Finally, it will be necessary in the Raspberry Pi configuration menu. Go to the “Interfaces” section to enable it (as shown in Figure 3-45) and reboot the Raspberry Pi.

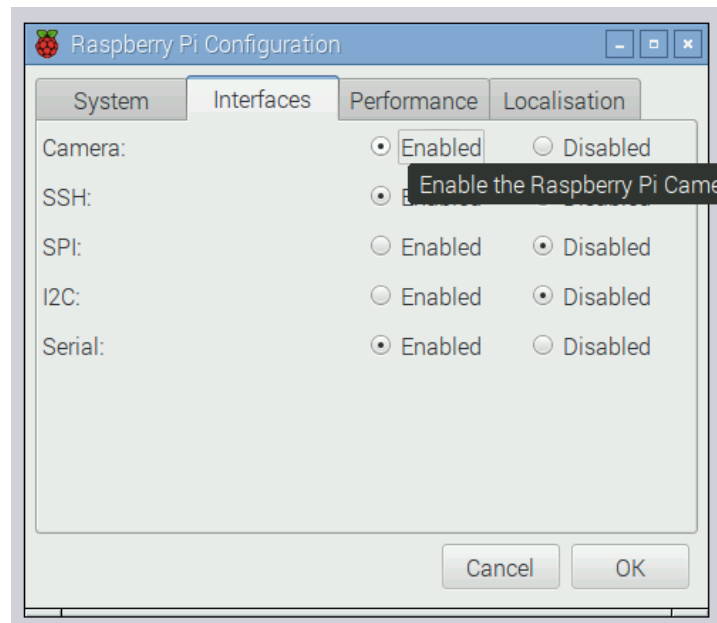


Figure 3-45 Screenshot of Interfaces section

3.4.3 How to install RPi-Cam Web Interface

Once Raspbian is installed and configured, RPi-Cam Web Interface will be installed based on instructions of [54]. The necessary steps are listed with the commands that will be executed from the Terminal.

1. Before installing, it must be ensured that all programs, dependencies, firmware are up to date:

```
sudo apt-get update
```

```
sudo apt-get dist-upgrade
```

```
sudo rpi-update
```

2. Once updated, the RPi-Cam Web Interfaces' code will be cloned (downloaded) onto the Raspberry Pi:

```
pi@raspberrypi:~/r $ git clone https://github.com/silvanmelchior/RPi_Cam_Web_Interface.git
Cloning into 'RPi_Cam_Web_Interface'...
remote: Counting objects: 1836, done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 1836 (delta 0), reused 0 (delta 0), pack-reused 1830
Receiving objects: 100% (1836/1836), 3.45 MiB | 810.00 KiB/s, done.
Resolving deltas: 100% (1025/1025), done.
Checking connectivity... done.
```

Figure 3-46 Screenshot of RPi-Cam Web Interface's code being downloaded

3. When the download finishes, navigate to the directory and make all the scripts executable:

```
cd Rpi_Cam_Web_Interface
sudo chmod u+x *.sh
```

4. There are 5 scripts included in RPi-Cam Web Interface:

- `install.sh` : to install RPi-Cam Web Interface and the necessary dependencies
- `update.sh`: to update RPi-Cam Web Interface
- `start.sh`: used to start RPi-Cam Web Interface (unnecessary if auto start is enabled)
- `stop.sh`: stops RPi-Cam Web Interface
- `remove.sh`: to uninstall RPi-Cam Web Interface

To begin the installation, execute the install script to initiate the configuration options as can be seen in Figure 3-47.

```
sudo ./install.sh
```

The Cam subfolder defines in which directory of the webserver we wish RPi-Cam Web Interface to be hosted, auto start indicates whether or not we wish the program to start automatically when we initiate the Raspberry Pi. There are two options as to which webserver program to install, either Apache [61] or Nginx [62]. The web port indicates on which port the interface will be accessible from. Lastly there is an option to define a user and password so as to restrict access to the program.

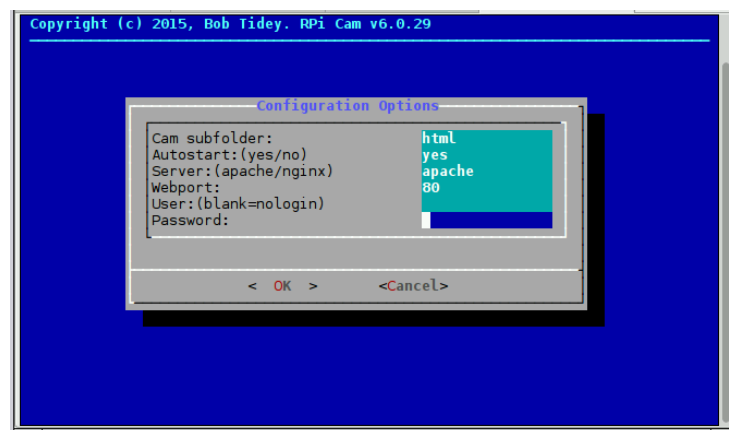


Figure 3-47 Screenshot of installation menu

For the installation, the following parameters will be used as shown in Figure 3-48. Obviously both the user and password should be different to the example in this guide.

Cam subfolder:

Auto start: yes

Server: apache

Webport: 80

User: admin

Password: opentome

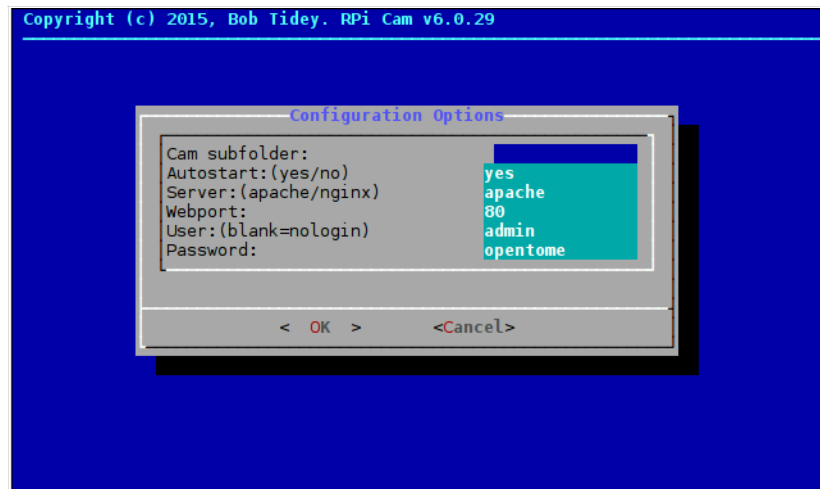


Figure 3-48 Screenshot of configuration used in the installation of RPi-Cam Web Interface

The script will install all the necessary components of RPi-Cam Web Interface including *motion*, PHP, Apache webserver, etc. When the installation is complete a reboot is necessary.

5. Having installed RPi-Cam Web Interface to access the program, open a browser on the Raspberry Pi and go to <http://localhost> or from another device connected to the network go to <http://rapsberry-pi-IP:8765> to configure the program as indicated in section 3.3.3.

3.4.4 How to install MotionEye on the central server with Ubuntu

This guide is based on [63]. It must be noted that depending on the version of Ubuntu that is installed different steps must be taken. MotionEye is not limited to Ubuntu and can be installed on any Linux OS, instructions on how to install in different distributions can be consulted in [64].

1. **Skip this step if your version of Ubuntu is 15.04 or newer.**

It is necessary to add the following software source to be able to install *ffmpeg* (program used by MotionEye)

```
sudo add-apt-repository -y ppa:kirillshkrogalev/ffmpeg-next
sudo apt-get update
```

2. Next, it is necessary to install the programs: *ffmpeg*, *motion* and *v4l-utils*:

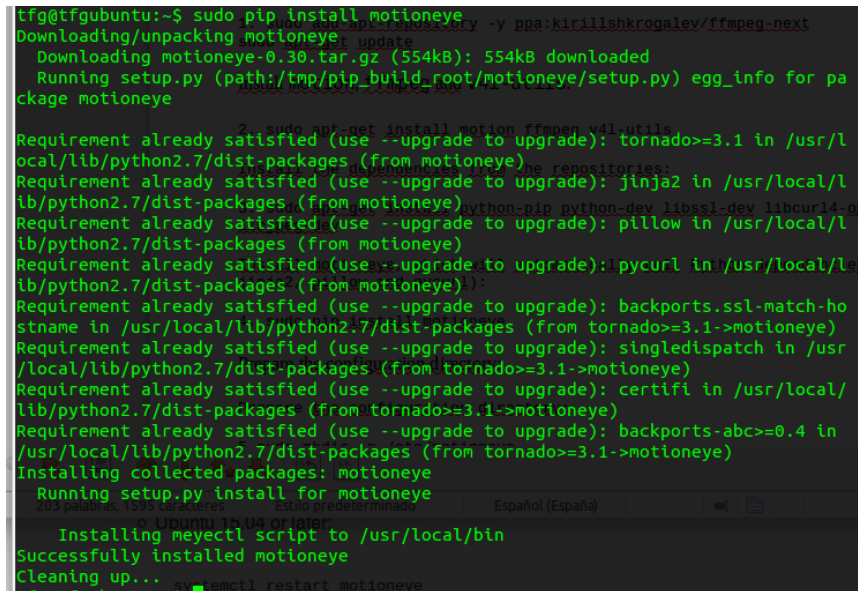
```
sudo apt-get install motion ffmpeg v4l-utils
```

3. Install the necessary dependencies:

```
sudo apt-get install python-pip python-dev libssl-dev libcurl4-openssl-dev libjpeg-dev
```

- Having installed python-pip (a tool to install python packages [65]) we will install MotionEye by executing the following command (as can be seen in Figure 3-49).

```
sudo pip install motioneye
```



```
tfg@tfgubuntu:~$ sudo pip install motioneye
Downloading/unpacking motioneye
  Downloading motioneye-0.30.tar.gz (554kB): 554kB downloaded
  Running setup.py (path:/tmp/pip_build_root/motioneye/setup.py) egg_info for package motioneye
Requirement already satisfied (use --upgrade to upgrade): tornado>=3.1 in /usr/local/lib/python2.7/dist-packages (from motioneye)
Requirement already satisfied (use --upgrade to upgrade): Jinja2 in /usr/local/lib/python2.7/dist-packages (from motioneye)
Requirement already satisfied (use --upgrade to upgrade): pillow in /usr/local/lib/python2.7/dist-packages (from motioneye)
Requirement already satisfied (use --upgrade to upgrade): pycurl in /usr/local/lib/python2.7/dist-packages (from motioneye)
Requirement already satisfied (use --upgrade to upgrade): backports.ssl-match-hostname in /usr/local/lib/python2.7/dist-packages (from tornado>=3.1->motioneye)
Requirement already satisfied (use --upgrade to upgrade): singledispatch in /usr/local/lib/python2.7/dist-packages (from tornado>=3.1->motioneye)
Requirement already satisfied (use --upgrade to upgrade): certifi in /usr/local/lib/python2.7/dist-packages (from tornado>=3.1->motioneye)
Requirement already satisfied (use --upgrade to upgrade): backports-abc>=0.4 in /usr/local/lib/python2.7/dist-packages (from tornado>=3.1->motioneye)
Installing collected packages: motioneye
  Running setup.py install for motioneye
    Installing meyeectl script to /usr/local/bin
Successfully installed motioneye
Cleaning up...
```

Figure 3-49 Screenshot of installation of MotionEye using python-pip

- Prepare the configuration directory and copy the sample configuration of MotionEye to it.

```
sudo mkdir -p /etc/motioneye
```

```
sudo cp /usr/local/share/motioneye/extra/motioneye.conf.sample /etc/motioneye/motioneye.conf
```

- Create the directory where MotionEye will store the images and videos it records.

```
sudo mkdir -p /var/lib/motioneye
```

- In order to start MotionEye automatically every time we initiate the computer execute the following commands:

- **Ubuntu 14.10 or earlier:**

```
sudo cp /usr/local/share/motioneye/extra/motioneye.init-debian /etc/init.d/motioneye
```

```
sudo chmod +x /etc/init.d/motioneye
```

```
sudo update-rc.d -f motioneye defaults
```

```
sudo /etc/init.d/motioneye start
```

- **Ubuntu 15.04 or later:**

```
sudo cp /usr/local/share/motioneye/extra/motioneye.systemd-unit-local
/etc/systemd/system/motioneye.service
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl enable motioneye
```

```
sudo systemctl start motioneye
```

8. Once MotionEye is installed, the commands to update, start, stop, restart, etc. are listed in Table 3-14 and Table 3-15.

Ubuntu 14.10 or earlier	
Restart MotionEye	<i>sudo service motioneye restart</i>
Stop MotionEye	<i>sudo service motioneye stop</i>
Start MotionEye	<i>sudo service motioneye start</i>
Status of MotionEye	<i>sudo service motioneye status</i>
Upgrade MotionEye	<i>sudo pip install motioneye --upgrade</i>

Table 3-14 MotionEye commands for Ubuntu 14.10 or previous

Ubuntu 15.04 or later	
Restart MotionEye	<i>sudo systemctl restart motioneye</i>
Stop MotionEye	<i>sudo systemctl start motioneye</i>
Start MotionEye	<i>sudo systemctl stop motioneye</i>
Status of MotionEye	<i>sudo systemctl status motioneye</i>
Upgrade MotionEye	<i>sudo pip install motioneye --upgrade</i>

Table 3-15 MotionEye commands for Ubuntu 15.04 or later

3.4.5 How to implement remote reboot and shutdown functionality to MotionEye.

As part of version 0.3, MotionEye incorporated an option to perform six actions [66]:

- lock
- unlock
- light_on
- light_off

- alarm_on
- alarm_off

When enabled, within the camera preview windows a unique icon would be activated and when clicked will execute a script. However, these six actions are not defined and it is left to the user to write the script and decide the commands it should execute.

Using this functionality and modifying part of MotionEye's code another icon can be activated on each video preview and can be linked to a specific script to carry out certain commands.

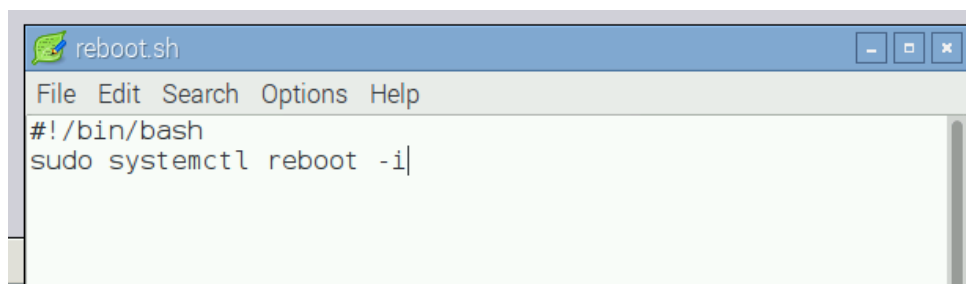
This will be used to enable rebooting or performing a shutdown on each Raspberry Pi individually from the MotionEye interface. To achieve this, the script that MotionEye executes will need to perform the following actions:

- Open a SSH secure connection to the specific Raspberry Pi
- Execute a reboot/shutdown script located on the Raspberry Pi.

The first step is to write an executable script to reboot/shutdown the Raspberry Pi and save it on each Raspberry Pi.

FIRST STEP: PREPARE REBOOT/SHUTDOWN SCRIPT ON EACH RASPBERRY PI

1. To create the reboot script, open the text editor “*LeafPad*” in Raspberry Pi (Menu→Office→LeafPad) and copy the following code as we can see in Figure 3-50.



2. Save the script as *reboot.sh* in the home folder.
3. For the shutdown script, open a new document and copy the following code as we can see in Figure 3-51.

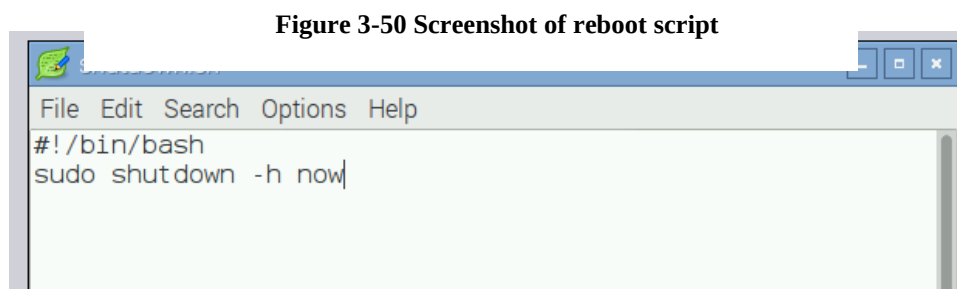


Figure 3-51 Screenshot of shutdown script

4. Save the script as *shutdown.sh* in the home folder.

5. Make both scripts executable using:

```
sudo chmod +x reboot.sh
sudo chmod +x shutdown.sh
```

Repeat these steps for each Raspberry Pi

SECOND STEP: ESTABLISH A SSH CONNCETION FOR EACH RASPBERRY PI

1. In order to execute the reboot/shutdown script remotely, an SSH connection will be established. First it must be ensured that an SSH connection is working and that each Raspberry Pi is added to the list in the file: *known_hosts*. It will be necessary to install the program *sshpass* to simplify connecting to each Raspberry Pi.

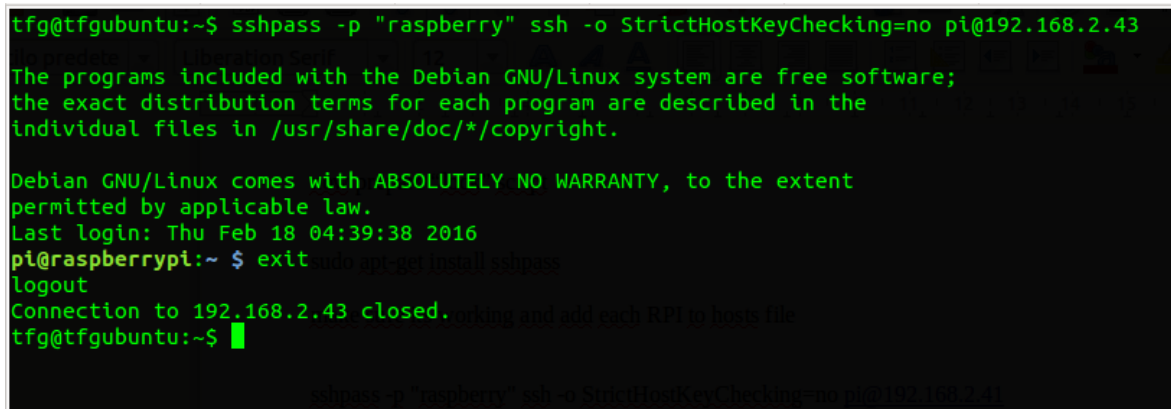
```
sudo apt-get install sshpass
```

2. The following code must be executed for each Raspberry Pi's IP address which is configured with a static address to ensure the IP does not change. An example of how this code is executed can be found in Figure 3-52.

```
sshpass -p "raspberrry" ssh -o StrictHostKeyChecking=no pi@192.168.2.41
```

```
exit
```

Substitute with each
Raspberry Pi's IP address



```
tfg@tfgubuntu:~$ sshpass -p "raspberrry" ssh -o StrictHostKeyChecking=no pi@192.168.2.43
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Feb 18 04:39:38 2016
pi@raspberrypi:~$ exit
logout
Connection to 192.168.2.43 closed.
tfg@tfgubuntu:~$
```

Figure 3-52 Screenshot of code to establish SSH connection

THIRD STEP: WRITE THE SCRIPTS FOR MOTIONEYE TO EXECUTE

1. MotionEye is configured to execute the scripts named *unlock*, *lock*, *light_on*, *light_off*, *alarm_on*, *alarm_off*, that are in the folder */etc/motioneye*. The scripts must be named with one of the actions followed by a “_” and the number of the camera. An example of this naming scheme would be: *action_1*.

The action named *lock* will be used to reboot and *unlock* to shutdown. Therefore naming scheme shown in Table 3-16 will be implemented.

NAME OF SCRIPT	ACTION
lock_1	Reboots Raspberry Pi 1
lock_2	Reboots Raspberry Pi 2
lock_3	Reboots Raspberry Pi 3
unlock_1	Shutdown Raspberry Pi 1
unlock_2	Shutdown Raspberry Pi 2
unlock_3	Shutdown Raspberry Pi 3

Table 3-16 Naming scheme of reboot/shutdown scripts

2. Prepare the reboot script for each Raspberry Pi using the following code as can be seen in Figure 3-53. For each Raspberry Pi the IP must be changed in the code:

```
#!/bin/bash
```

```
sshpass -p "raspberrry" ssh -o StrictHostKeyChecking=no pi@192.168.2.41 'sudo ./reboot.sh'
```

Substitute with each
Raspberry Pi's IP address

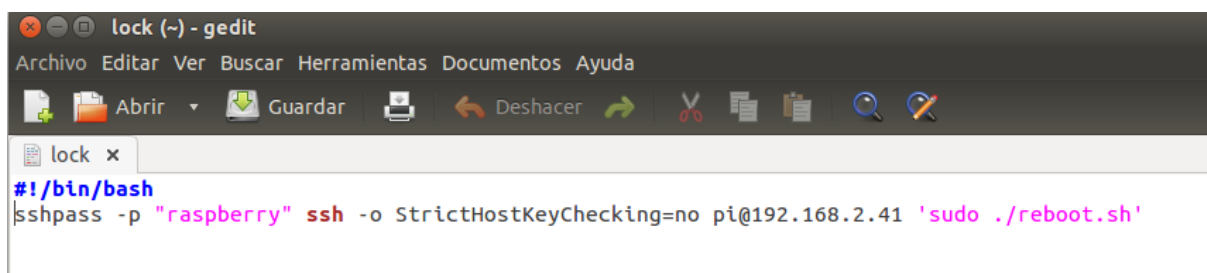


Figure 3-53 Script to remotely execute reboot

3. Save the script using the naming scheme in Table 3-16.

- Repeat the same process for the shutdown script using the following code as we can see in Figure 3-54. For each Raspberry Pi the IP must be changed in the code:

```
#!/bin/bash
```

```
sshpas -p "raspberr" ssh -o StrictHostKeyChecking=no pi@192.168.2.41 'sudo ./shutdown.sh'
```

Substitute with each
Raspberry Pi's IP address

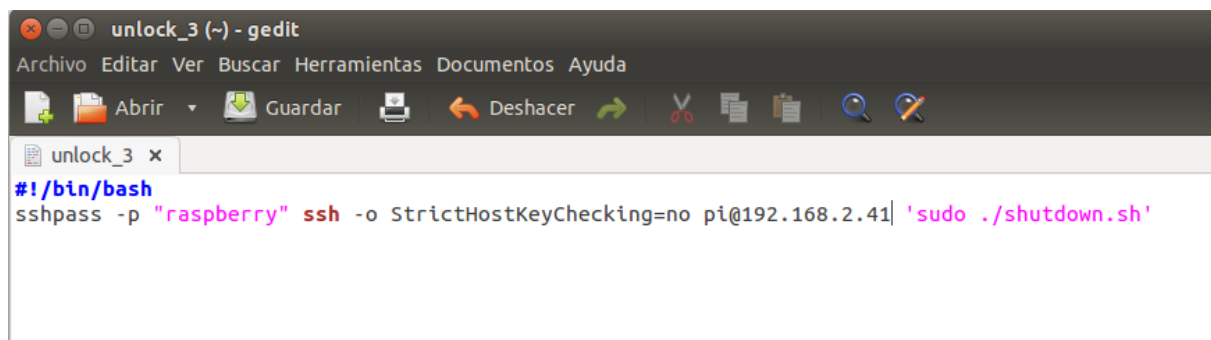


Figure 3-54 Script to remotely execute shutdown

- When the scripts are completed they must be moved to the directory `/etc/motioneye` as we can see in Figure 3-55.

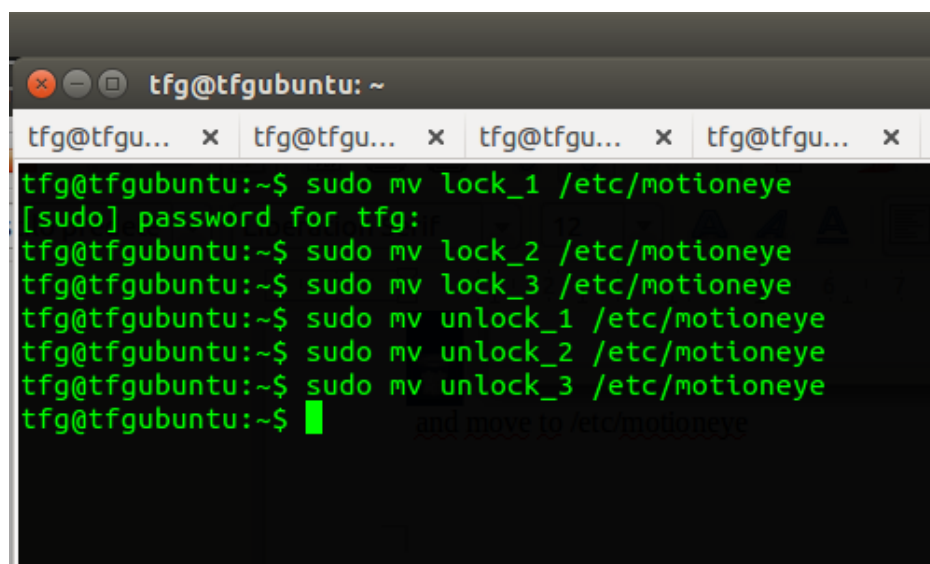
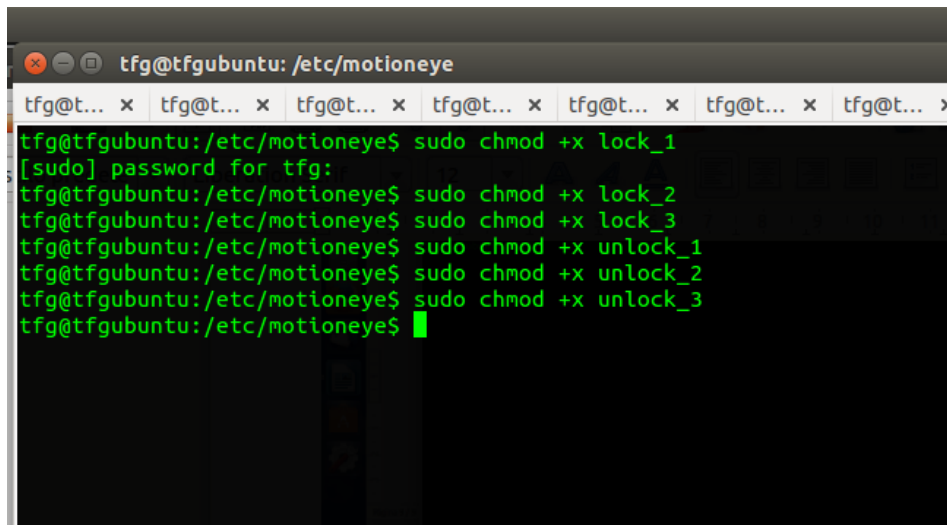


Figure 3-55 Screenshot of scripts moved

- Next, it is necessary to make the scripts executable by issuing the following commands as shown in Figure 3-56.



```
tfg@tfgubuntu: /etc/motioneye
tfg@tfgubuntu: /etc/motioneye$ sudo chmod +x lock_1
[sudo] password for tfg:
tfg@tfgubuntu: /etc/motioneye$ sudo chmod +x lock_2
tfg@tfgubuntu: /etc/motioneye$ sudo chmod +x lock_3
tfg@tfgubuntu: /etc/motioneye$ sudo chmod +x unlock_1
tfg@tfgubuntu: /etc/motioneye$ sudo chmod +x unlock_2
tfg@tfgubuntu: /etc/motioneye$ sudo chmod +x unlock_3
tfg@tfgubuntu: /etc/motioneye$
```

Figure 3-56 Screenshot making the scripts executable

FOURTH STEP: CUSTOMIZE THE MOTIONEYE INTERFACE

At this point, the scripts would be working and if the MotionEye program were restarted there would be two new icons within each video preview that would restart/shutdown the corresponding Raspberry Pi.

However, the icon of each action is not intuitive and the description within the interface is incorrect. In order to ensure that these actions have an appropriate icon and description, further steps were taken as described:

- First it is necessary to stop MotionEye by executing the following command depending on the version of Ubuntu:

Ubuntu 15.04 and later: *sudo systemctl stop motioneye*

Ubuntu 14.10 and previous: *sudo service motioneye stop*

- Within the MotionEye interface we can find the following icons which are intended for use with the predefined actions:

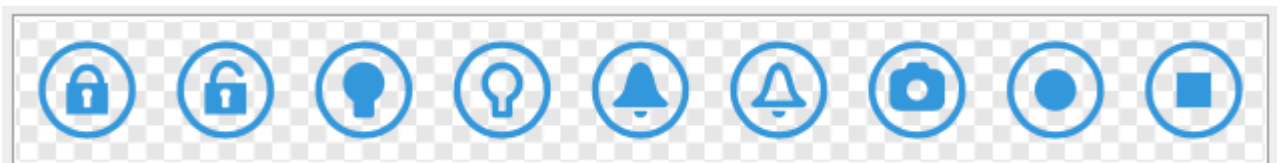


Figure 3-57 Icons available for the predefined actions

From an aesthetic point of view the following icons were selected to represent reboot and shutdown (although by replacing the icon image file in the MotionEye folders any icon can be established).



3. To bind the action *lock* and *unlock* to the selected icon, we will need to modify part of the code that defines the interface of MotionEye:

```
sudo gedit /usr/local/lib/python2.7/dist-packages/motioneye/static/css/main.css
```

4. Search within the file for the following values (see Figure 3-58) and modify them as follows:

change “0px” to “-700%”

change “-100%” to “-800%”

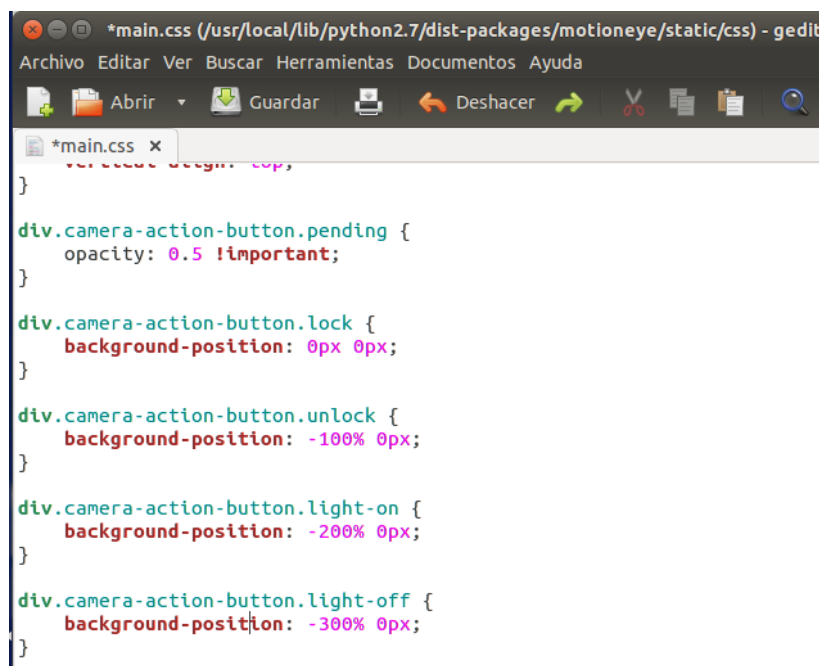
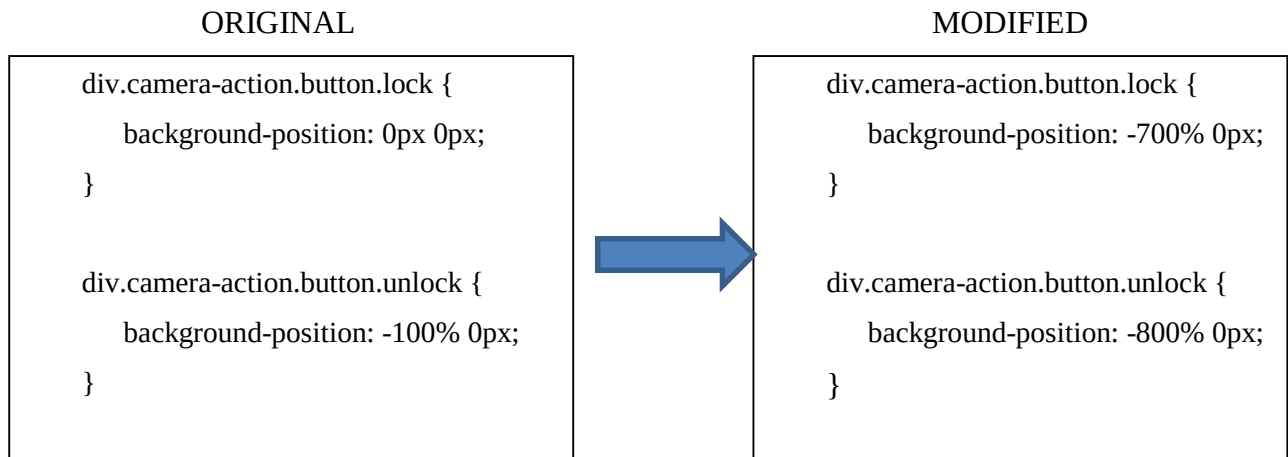


Figure 3-58 Default values of the file main.css



This will change the position within the icons file defined for each action, effectively changing which icon is defined for the *lock* and *unlock* action to the two icons selected.

5. The final step consists in changing the description of the icon when the mouse cursor hovers on top of the icon in the MotionEye interface. The goal is to modify the description from *lock* to *reboot* and *unlock* to *shutdown*.

To achieve this, the file *static.js* will be modified as follows:

```
sudo gedit /usr/local/lib/python2.7/dist-packages/motioneye/static/js/main.js
```

6. Search for the code shown in Figure 3-59.

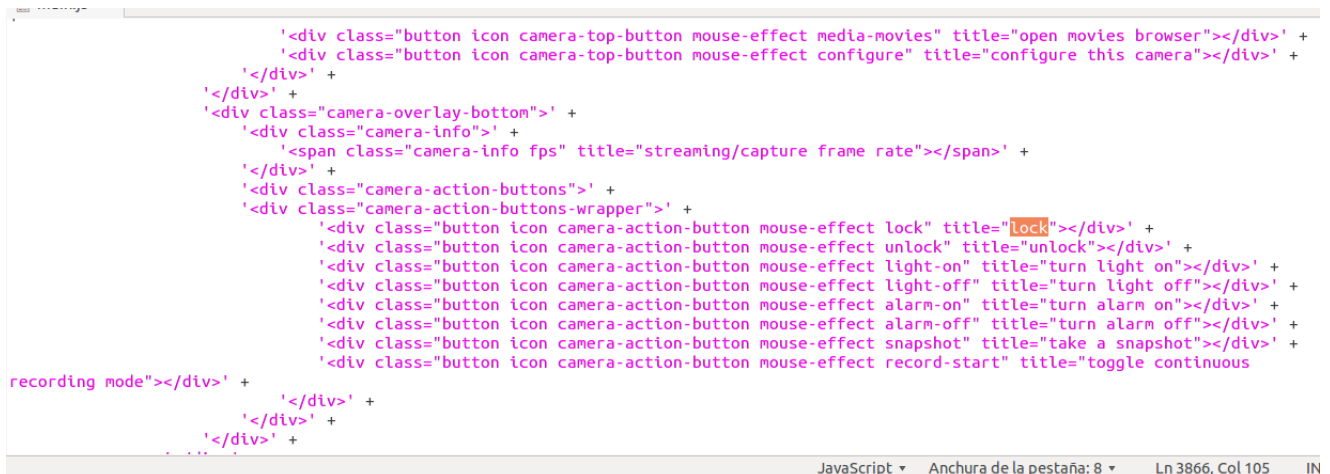


Figure 3-59 Screenshot of default main.js

Modify the following code

`<div class="button icon camera-action-button mouse-effect lock" title="lock">`

to

```
'<div class="button icon camera-action-button mouse-effect lock" title="Reboot RPI">
```

so it resembles Figure 3-60.

```

    '<div class="button icon camera-top-button mouse-effect media-movies" title="open movies browser"></div>' +
    '<div class="button icon camera-top-button mouse-effect configure" title="configure this camera"></div>' +
    '</div>' +
    '<div class="camera-overlay-bottom">' +
    '<div class="camera-info">' +
    '<span class="camera-info fps" title="streaming/capture frame rate"></span>' +
    '</div>' +
    '<div class="camera-action-buttons">' +
    '<div class="camera-action-buttons-wrapper">' +
    '<div class="button icon camera-action-button mouse-effect lock" title="reboot RPI"></div>' +
    '<div class="button icon camera-action-button mouse-effect unlock" title="shutdown RPI"></div>' +
    '<div class="button icon camera-action-button mouse-effect light-on" title="turn light on"></div>' +
    '<div class="button icon camera-action-button mouse-effect light-off" title="turn light off"></div>' +
    '<div class="button icon camera-action-button mouse-effect alarm-on" title="turn alarm on"></div>' +
    '<div class="button icon camera-action-button mouse-effect alarm-off" title="turn alarm off"></div>' +
    '<div class="button icon camera-action-button mouse-effect snapshot" title="take a snapshot"></div>' +
    '<div class="button icon camera-action-button mouse-effect record-start" title="toggle continuous
recording mode"></div>' +
    '</div>' +
    '</div>' +
    '</div>' +
  
```

Figure 3-60 Screenshot of modified main.js

FIFTH STEP: START MOTIONEYE

Once all the steps have been completed start MotionEye:

Ubuntu 15.04 and later: *sudo systemctl start motioneye*

Ubuntu 14.10 and previous: *sudo service motioneye start*

Now, there will be two new icons in each preview window (see Figure 3-61), that when clicked will either reboot or shutdown the corresponding Raspberry Pi.



Figure 3-61 Screenshot of preview windows with reboot and shutdown icons

3.4.6 How to add a new camera in MotionEye

In order to add one of the Raspberry Pi's video feeds (which has been configured with RPi-Cam Web Interface) to MotionEye it is necessary to follow these steps:

1. Open a browser on the central server and go to <http://localhost:8765> (or from another device connected to the network: <http://central-server-IP:8765>), insert the login and password (default configuration is user: *admin* and no password as shown in Figure 3-62).

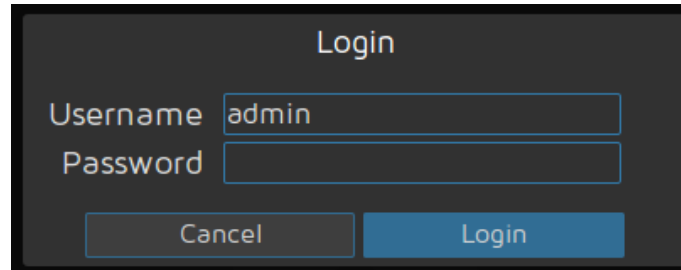


Figure 3-62 Screenshot of MotionEye login screen

2. Select “add camera” (see Figure 3-63)

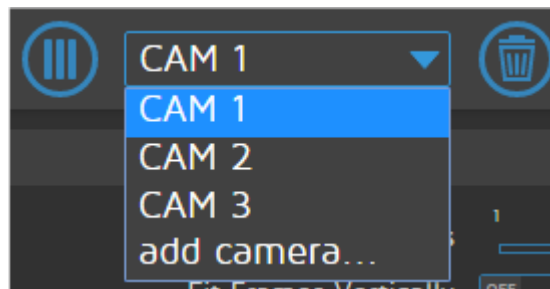


Figure 3-63 Screenshot of the MotionEye menu to a configure a new camera

3. There are four different camera types that can be selected as shown in Figure 3-64, select “Network Camera”.
 - a. Local Camera: A camera that is physically connected to the server
 - b. Network Camera: A remote camera accessible from the local network
 - c. Remote MotionEye Camera: A remote camera that uses MotionEye for the video stream
 - d. Simple MJPEG Camera: A remote camera that uses MJPEG to stream the video feed.

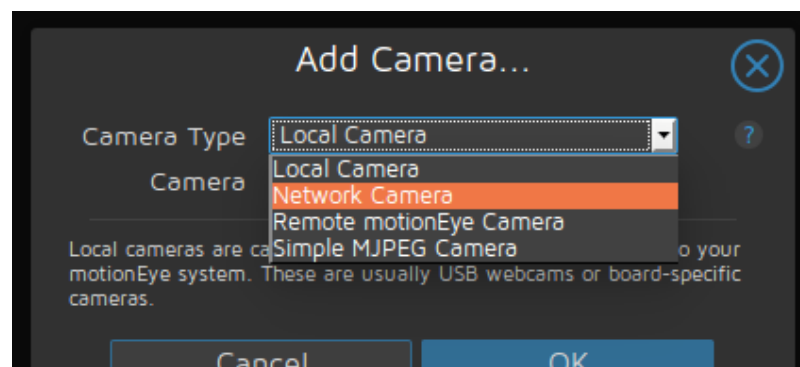


Figure 3-64 Screenshot of MotionEye menu to add camera

4. In the following menu (as shown in Figure 3-65), input the options from Table 3-17.

URL	http://raspberry-pi-IP/cam.jpg
Username	Put the username configured in the RPi-Cam Web Interface installation.
Password	Put the password configured in the RPi-Cam Web Interface installation.
Camera	Select JPEG Network Camera

Table 3-17 Configuration options to add a network camera configured with RPi-Cam Web Interface

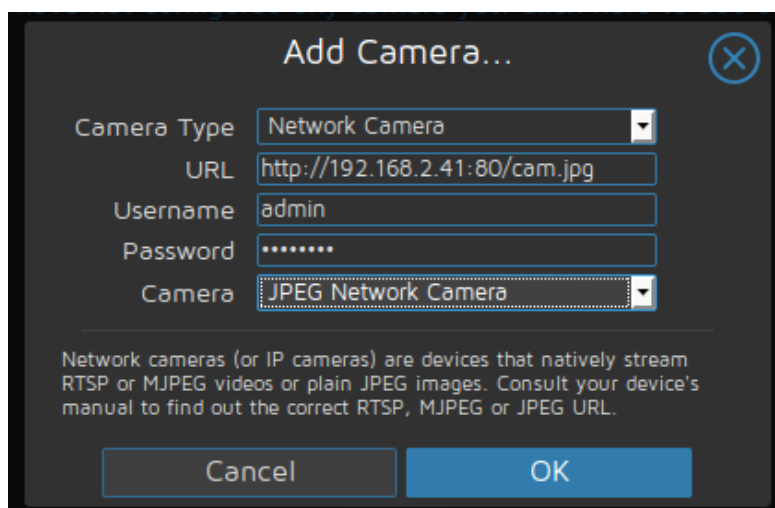


Figure 3-65 Screenshot of MotionEye to add a network camera

4 EXPERIMENTS AND RESULTS

To ensure the functionality of the designed surveillance system, a series of tests were designed. The tests that were carried out measured different parameters and evaluated the performance of the system. In particular, the tests were focused on the performance of the system, battery autonomy of each Raspberry Pi, individual tests of each advanced feature and finally a real-world experiment.

4.1 Performance experiments

4.1.1 Introduction

In order to evaluate the quality of the video feeds and the overall performance of the system under different conditions, two experiments were carried out. The first experiment is a comparison of the frame rate of the video feed and CPU load on each Raspberry Pi with both RPi-Cam Web Interface and MotionEye. The second experiment measures the difference in frame rate using Wi-Fi and Ethernet.

4.1.2 Performance comparison of RPi-Cam Web Interface and MotionEye using different configurations

The objective of this experiment is to determine which program offers the highest quality video feed of at least 10 fps. The parameters that were modified were both the resolution and image quality. Both of these parameters have a direct effect on CPU load and network bandwidth.

To determine the performance of each configuration and each program, both the frame rate of the video feed and the CPU load on each Raspberry Pi was measured.

RESULTS

100% Image Quality			
Resolution	1280 x 1024	1024 x 768	682 x 512
MotionEye	2.2 fps, 55% CPU	4.2 fps, 52% CPU	5.8 fps, 54% CPU
RPi-Cam Web Interface	10.2 fps, 31% CPU	14.2 fps, 28% CPU	20 fps, 17% CPU

Table 4-1 Fps and CPU load using 100% image quality and different configurations

75% Image Quality			
Resolution	1280 x 1024	1024 x 768	682 x 512
MotionEye	4.3fps, 55% CPU	5.4 fps, 43% CPU	9.2 fps, 52% CPU
RPi-Cam Web Interface	11.2 fps, 30% CPU	16.4 fps, 25% CPU	20.8 fps, 20% CPU

Table 4-2 Fps and CPU load using 75% image quality and different configurations

50% Image Quality			
Resolution	1280 x 1024	1024 x 768	682 x 512
MotionEye	4.3 fps, 54% CPU	5.4 fps, 43% CPU	11.8 fps, 53% CPU
RPi-Cam Web Interface	12.1 fps, 31% CPU	18.8 fps, 26% CPU	22.2 fps, 20% CPU

Table 4-3 Fps and CPU load using 50% image quality and different configurations

25% Image Quality			
Resolution	1280 x 1024	1024 x 768	682 x 512
MotionEye	4.9 fps, 56% CPU	5.6 fps, 43% CPU	12.8 fps, 52% CPU
RPi-Cam Web Interface	18.2 fps, 30% CPU	22.3 fps, 24% CPU	24.7 fps, 17% CPU

Table 4-4 Fps and CPU load using 25% image quality and different configurations

10% Image Quality			
Resolution	1280 x 1024	1024 x 768	682 x 512
MotionEye	5.7 fps, 55% CPU	5.9 fps, 42% CPU	14.5 fps, 53% CPU
RPi-Cam Web Interface	22.1 fps, 27% CPU	23.0 fps, 24% CPU	26 fps, 17% CPU

Table 4-5 Fps and CPU load using 10% image quality and different configurations

CONCLUSION

Based on these results it was determined that RPi-Cam Web Interface uses around 50% less CPU resources while offering a frame rate that is nearly 200% higher than MotionEye. Therefore, RPi-Cam Web Interface will be used when streaming the video feed from the Raspberry Pi's to the central server as previously explained in the design of the video surveillance system in section 3.1.2.

These results also allow to determine an adequate configuration in order to ensure a smooth video feed. These settings are described in section 3.3.3, Table 3-13.

4.1.3 Performance comparison of the video feed using Wi-Fi and Ethernet

Ethernet offers higher network speed rates than Wi-Fi. This experiment measures the difference in frame rate and determines the impact on the quality of the video feed when using either Wi-Fi or Ethernet. The configuration for the Raspberry Pi is described in section 3.3.3, Table 3-13.

RESULTS

Wi-Fi	Ethernet
18,8 fps	27 fps

Table 4-6 Fps comparison between Wi-Fi and Ethernet

CONCLUSION

As expected, the frame rate of the video feed when using Ethernet is higher than when using Wi-Fi. However, Wi-Fi has the advantage that it allows the Raspberry Pi's to be anywhere within the Wi-Fi range of the router allowing them to be very portable.

4.1.4 Performance comparison of the video feed when using multiple Raspberry Pi's

The previous experiment (section 4.1.3) clearly shows that when using Ethernet the video feed has a higher frame rate. The question remains how would additional Raspberry Pi's in the network affect the video feed? This experiment, measures the frame rate when using both Ethernet and Wi-Fi to measure how additional Raspberry Pi's affects performance.

RESULTS

When using more than one Raspberry Pi, the frame rate shown is the average.

	Wi-Fi	Ethernet
1 Raspberry Pi	18,8 fps	27 fps
2 Raspberry Pi's	12 fps	26,7 fps
3 Raspberry Pi's	10,1 fps	26,8 fps

Table 4-7 Fps when using various Raspberry Pi's using Wi-Fi and Ethernet

CONCLUSION

Based on these results, it is clear that when using Ethernet, adding more Raspberry Pi's to the network (in this scenario) has very little effect in the frame rate of the video feeds. However, with Wi-Fi, the frame rate drops considerably. If the video surveillance system is performance-oriented, and the objective is to have the best possible video feeds, Ethernet should be used instead of Wi-Fi. This would mean a physical connection to the network is required. Evidently both Wi-Fi and Ethernet could be combined; the decision would depend on the characteristics of the physical setting in which the video surveillance system is to be used.

4.2 Battery life experiments

4.2.1 Introduction

As part of the hardware used in this project, each Raspberry Pi had a battery. Combined with Wi-Fi this results in a truly mobile configuration, allowing the Raspberry Pi's to be placed anywhere within the Wi-Fi range of the router for the duration of the battery life. Evidently, this simplifies the set-up of the system and allows for multiple possible locations for each Raspberry Pi.

In order to determine the battery life of the Raspberry Pi's, two tests were carried out. The first varied the configuration settings of RPi-Cam Web Interface to know how it affected battery life. The second, compared how using Wi-Fi against Ethernet influenced the battery.

4.2.2 Battery life using different configuration settings

The objective of this test is to measure how different configuration settings affect the battery life. Three different configurations were adopted as specified in Table 4-8, and established on each of the Raspberry Pi's. In order to ensure proper results, each battery was fully charged, was using Wi-Fi and were put at the same distance from the router.

	Resolution	Video bitrate	Frames per second	Image quality
High Settings	1920 x 1080	250000000	15	100
Medium Settings	1280 x 720	200000000	10	75
Low Settings	854 x 480	150000000	5	50

Table 4-8 Configuration settings used in battery life tests

RESULTS

The test was carried out three times in order to obtain an average and reduce possible anomalies. The results in Table 4-9 show how long the battery lasted on each Raspberry Pi.

	First Test	Second Test	Third Test	Average
Raspberry Pi 1 with high settings	13 hours and 36 minutes	12 hours and 10 minutes	11 hours and 22 minutes	12 hours and 22 minutes
Raspberry Pi 2 with medium settings	18 hours and 02 minutes	12 hours and 55 minutes	14 hours and 32 minutes	15 hours and 10 minutes
Raspberry Pi 3 with low settings	19 hours and 58 minutes	16 hours and 24 minutes	17 hours and 56 minutes	18 hours and 6 minutes

Table 4-9 Results of battery life using different settings

CONCLUSION

As would be expected, the results show that the higher the configuration of the video feed, the lower the battery life of the Raspberry Pi. There is a difference of up to 30% between the highest and lowest settings. This should be taken into account when using batteries for an extended period of time in the video surveillance system.

4.2.3 Battery life using Wi-Fi and Ethernet

Having established the difference in performance between Wi-Fi and Ethernet in section 4.1.3, this test will determine the impact on battery life. Two Raspberry Pi's with the same configuration were used, one with Wi-Fi and the other with Ethernet.

RESULTS

The test was carried out three times in order to obtain an average and reduce possible anomalies. The results in Table 4-10 show how long the battery lasted on each Raspberry Pi.

	First Test	Second Test	Third Test	Average
Wi-Fi	18 hours and 12 minutes	16 hours and 34 minutes	17 hours and 03 minutes	17 hours and 16 minutes
Ethernet	24 hours and 12 minutes	22 hours and 15 minutes	23 hours and 43 minutes	23 hours and 22 minutes

Table 4-10 Results of battery life using Wi-Fi and Ethernet

CONCLUSION

The results clearly show that Wi-Fi consumes more battery than Ethernet. In fact, when using Ethernet, the battery life is extended up to 48% more than when using Wi-Fi. When the performance results are taken into account, Ethernet is shown to have clear advantages over Wi-Fi in both performance and battery life. If battery life is a concern, whenever possible, it is recommendable to use an Ethernet connection.

4.3 Advanced features tests

4.3.1 Introduction

The video surveillance system that has been designed incorporates a series of advanced features. Each one has been tested independently to ensure they are working correctly. These features include: motion detection, email notification, night vision and remote reboot/shutdown of each Raspberry Pi.

4.3.2 Motion detection

Motion detection is done by the program *motion*, which serves as the underlying technology of MotionEye. It compares pixel changes in consecutive frames, and when it surpasses a certain amount defined by the user it considers there is motion. Depending on the minimum frames established to be

considered as a motion event, it will consider an immediate change between frames a motion event or wait for a more consistent change in more consecutive frames.

By using different settings we can set the sensitivity of the motion detection. Different scenarios were used and the results showed that when configured correctly, any movement in the video feed would be detected. When the system detects movement, a red frame is overlaid on the top and bottom bar of the video preview window as shown in Figure 4-1. Depending on how it has been configured it can perform certain actions (record a video, execute a command, send a notification etc.).

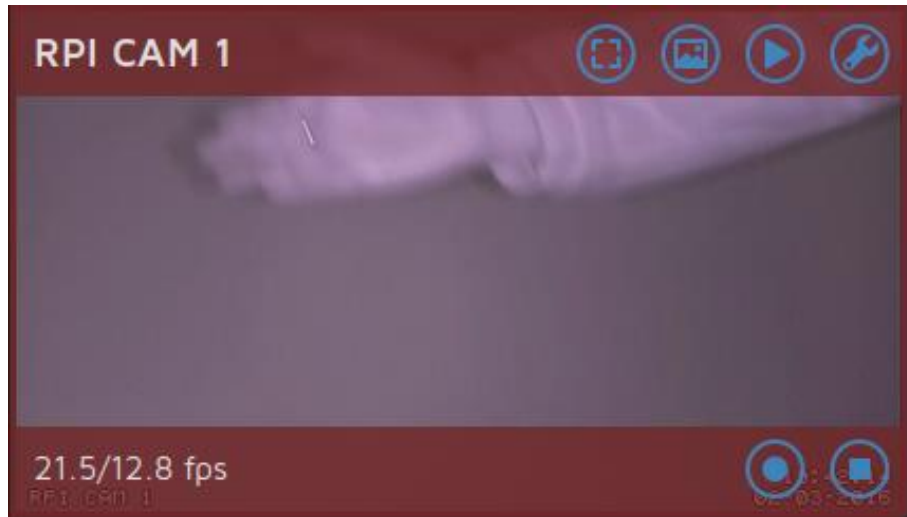


Figure 4-1 Screenshot of how MotionEye shows it detects movement

The system was able to detect anything from clouds to birds moving. An example of motion being detected is shown in Figure 4-18. Throughout the development of the project, motion detection was tested and proved to be very effective. The settings used during testing are detailed in section 3.3.4, under “Motion Detection”.

4.3.3 Email notifications

As part of the advanced features of the system, MotionEye can send an email notification when movement is detected. In the email it details the time and which camera detected movement. It also has an option to attach images of the motion that was detected.

To test this feature, the configuration defined in section 3.3.4 was used. Tests showed that when movement was detected by the system, an email was immediately sent as can be seen in Figure 4-2, in which the camera and time are specified as well as an attached image. This shows that the email notification system is working correctly.

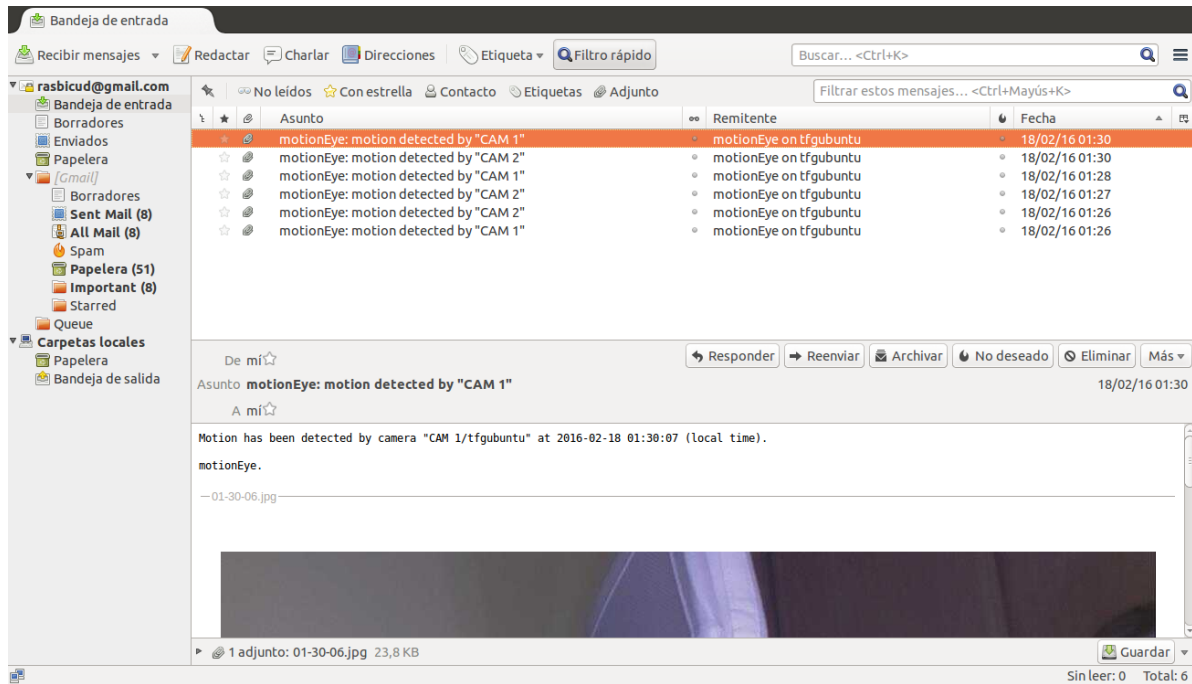


Figure 4-2 Screenshot of emails received notifying that movement has been detected

4.3.4 Night vision

As part of the hardware of the Pi Camera there is an infrared board which automatically turns on an infrared light in low light conditions. This enables the Pi Camera to take images in conditions with no light. To test this capability, a comparison was made with a picture taken with an iPhone 6 and the Raspberry Pi in both low-light conditions and complete darkness.

LOW LIGHT CONDITIONS

Figure 4-3 shows two pictures taken in low light conditions using an iPhone 6 and a Raspberry Pi. It is evident that the iPhone 6 takes better quality pictures in low-light conditions.

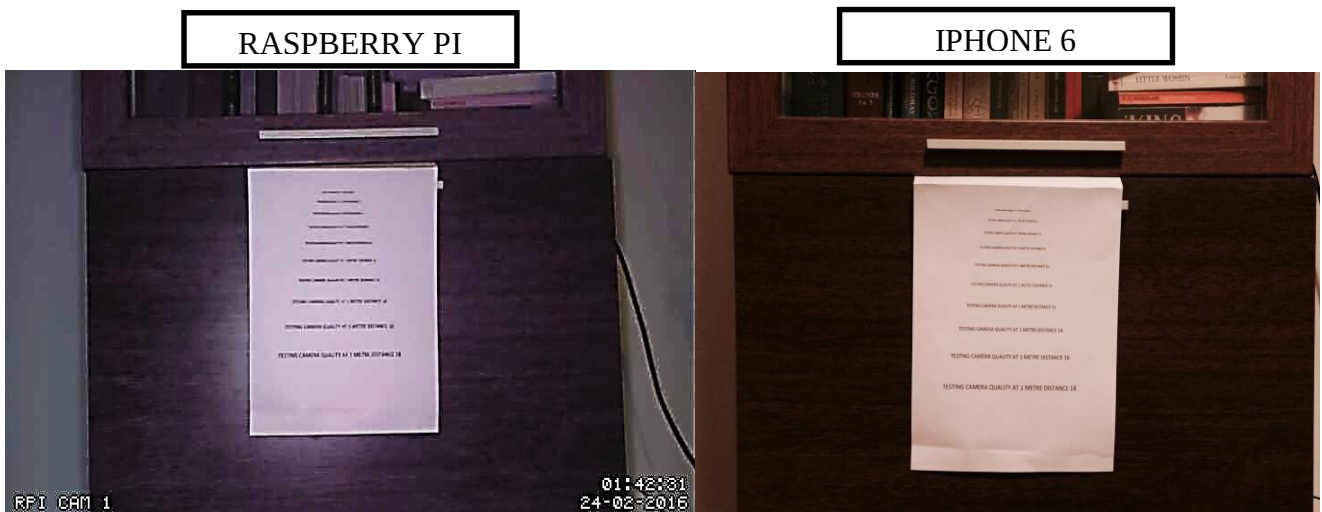


Figure 4-3 Comparison of a photo taken in low light with Pi Camera and an iPhone 6

However, in dark conditions, as can be seen in Figure 4-4, the image taken by the iPhone 6 is completely dark whereas the Raspberry Pi was able to take a black and white image of considerable quality.

Evidently, the workaround used by standard cameras when taking an image in the dark is to use flash. This solution however, would not be applicable in a video surveillance system, as it would need continuous exterior lighting to monitor activity and detect motion. This scenario is not realistic, which is why infrared lighting is used in video surveillance systems to be able to record when in dark conditions.

This test clearly shows that the video surveillance system also functions in dark conditions.



Figure 4-4 Comparison of a photo taken with no light with Pi Camera and an iPhone 6.

4.3.5 Remote reboot and shutdown of Raspberry Pi's from MotionEye

This test consists simply in evaluating whether or not the modifications made to the MotionEye interface described in 3.4.5 do in fact function properly and actually shutdown each Raspberry Pi when the reboot/shutdown icon is clicked.

Each reboot icon in each video preview was individually clicked, as a result the connection to the video feed was lost and the corresponding Raspberry Pi rebooted. The same was done with the shutdown icon as can be seen in Figure 4-5. The test showed that the reboot/shutdown functionality was working correctly and that the Raspberry Pi's could be administered remotely.

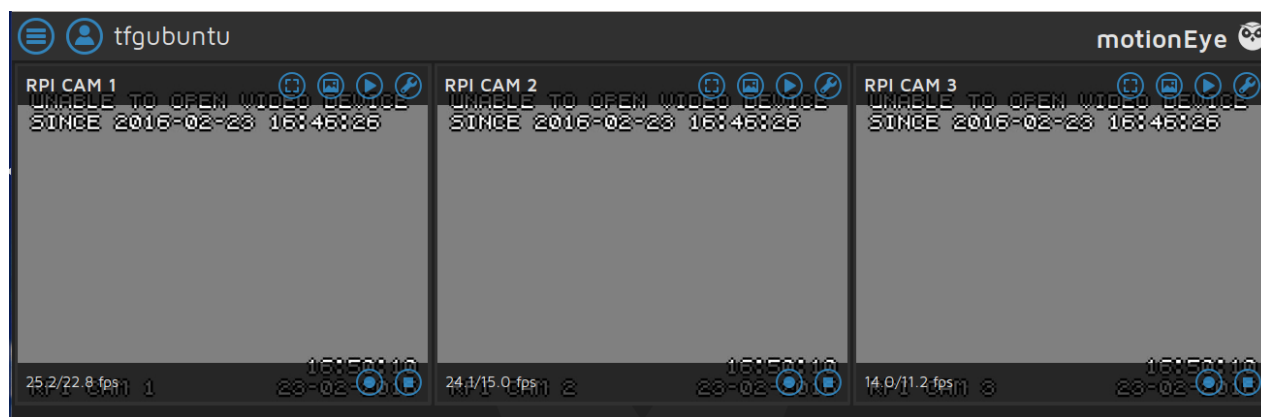


Figure 4-5 Screenshot of MotionEye interface with reboot/shutdown functionality

4.3.6 Security

As part of the features included in both RPi-Cam Web Interface and MotionEye there is an option to require a user and password to login. To test this feature, it was enabled in both programs and tested.

RPi-Cam Web Interface Security

When security is not enabled, to access the program all that is needed is to open a Internet browser and go to the address <http://localhost> on the Raspberry Pi or to <http://rapsberry-pi-IP>. However, when a user with a password is enabled and the interface is accessed, the program asks for the user and password as can be seen in Figure 4-6. The interface does not allow access to the program unless the correct user and password are input.

This option also restricts access to the video feed. When configuring the video device in MotionEye, the user and password are required.



Figure 4-6 Screenshot of login screen for RPi-Cam Web Interface

MotionEye Security

Like RPi-Cam Web Interface, MotionEye's security option also asks for a user and password when accessing the system from a browser. When an incorrect login is attempted it warns of "Invalid credentials" as can be seen in Figure 4-7.

Unlike RPi-Cam Web Interface, MotionEye has the option of defining two users, one being the administrator and the other a surveillance user. Surveillance users cannot access any of the configuration options while administrators have access to all the configuration options as explained in section 3.2.3, Table 3-1.

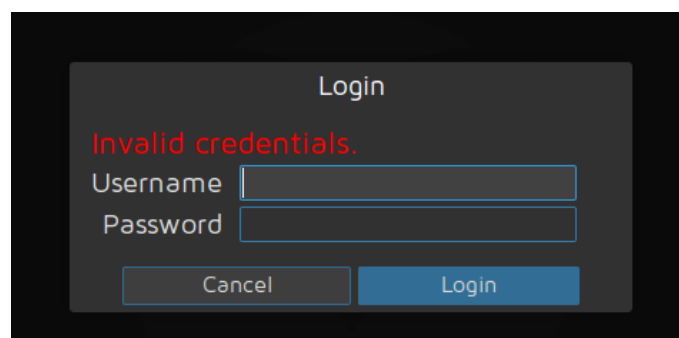


Figure 4-7 Screenshot of login attempt in MotionEye

Both RPi-Cam Web Interface and MotionEye restrict access to their interfaces by implementing a user and password system, which was adequate in basic testing. However, more advanced techniques to try to break the systems security were not attempted.

4.4 Real world test

4.4.1 Introduction

In order to properly evaluate the effectiveness of the video surveillance system it is necessary to carry out a realistic test for an extended period of time. Therefore, using the recommended configuration detailed in section 3.3, the system was set up in a real-world scenario and properly tested.

4.4.2 Set-up and configuration

The setting used for the test was the conference room of the Defense University Center in the Spanish Naval Academy. The room is big enough to record various angles and positions as can be seen in Figure 4-8. It has two doors which are ideal to place a Raspberry Pi and test the motion detection feature of the system. It also has a series of windows which face an area which tends to have lots of passers-by to test continuous activation and recording of the system.

The Raspberry Pi's were placed as shown in Figure 4-8.



Figure 4-8 Photo of the conference room of the Defense University Center, circles indicate the location of the Raspberry Pi's

Each Raspberry Pi had a fully-charged battery and was connected to the router shown in Figure 4-13. In order to keep the Pi Camera and the battery fixed, as a temporal solution rubber bands were used to give a certain degree of stability to the cameras as can be seen in Figure 4-9.

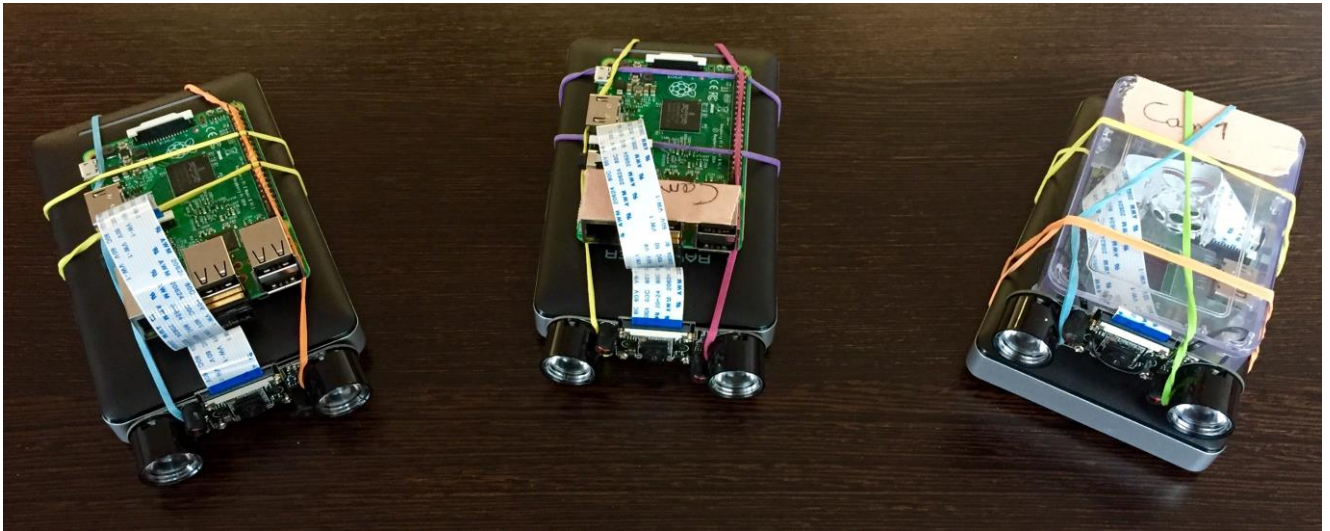


Figure 4-9 Image of the three Raspberry Pi's, each connected to a battery and using Wi-Fi

The first Raspberry Pi was placed at the end of the room, next to one of the windows facing the outside, to detect and survey passers-by as shown in Figure 4-10.



Figure 4-10 Location of Raspberry Pi 1

The second Raspberry Pi was placed on-top of a table, in the middle of the room, facing the main door as shown in Figure 4-11.

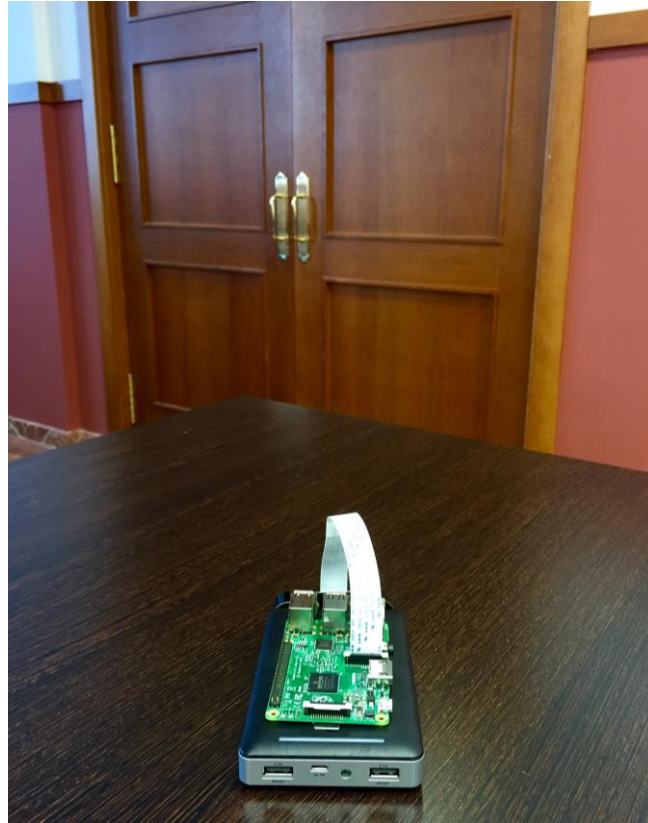


Figure 4-11 Location of Raspberry Pi 2

The third Raspberry Pi was placed at the end of the room, facing the emergency exit on the other side of the room as shown in Figure 4-12.

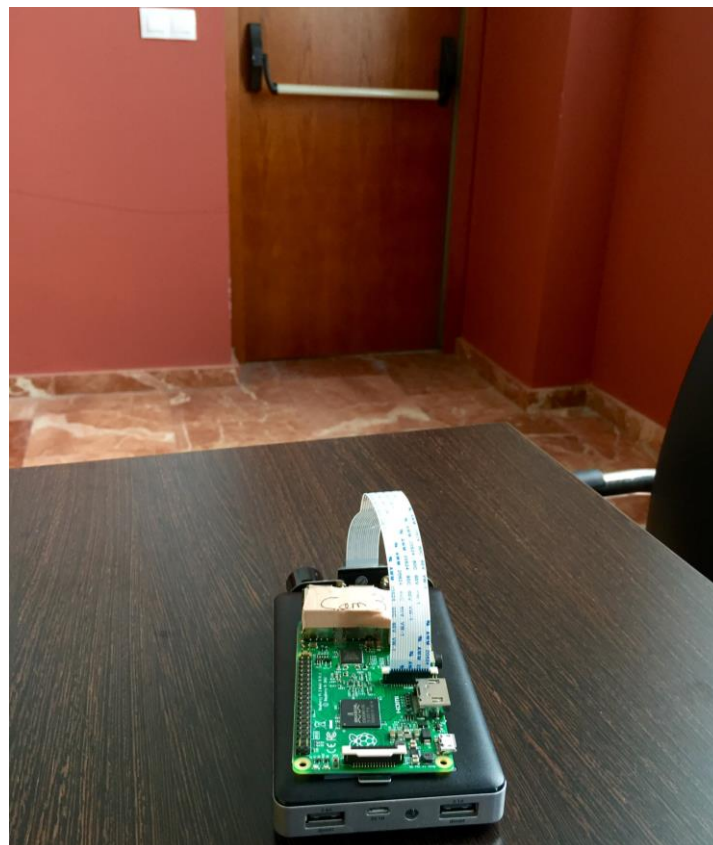


Figure 4-12 Location of Raspberry Pi 3

The router (Figure 4-13), along with the notebook (Figure 4-14) acting as the central server was placed at the front of the room.



Figure 4-13 Asus router used for the test

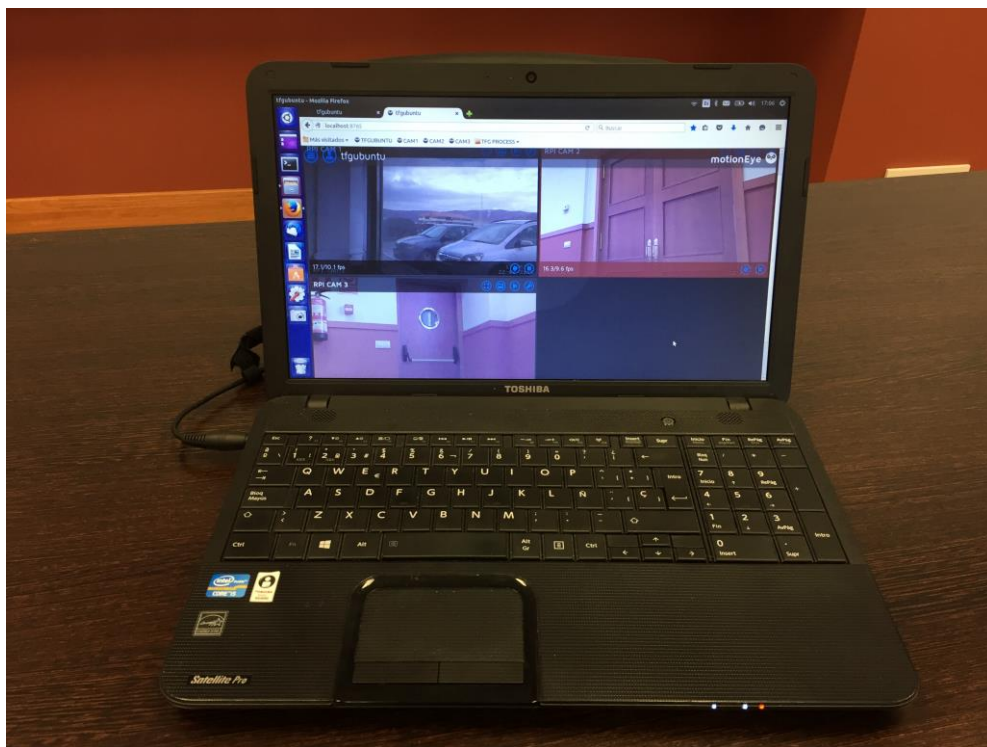


Figure 4-14 Toshiba notebook being used as central server

Once the system was set-up, each component of the system was turned on. Having ensured the system was working correctly as we can see in Figure 4-15 in which MotionEye is running and displaying the three video feeds of each Raspberry Pi.

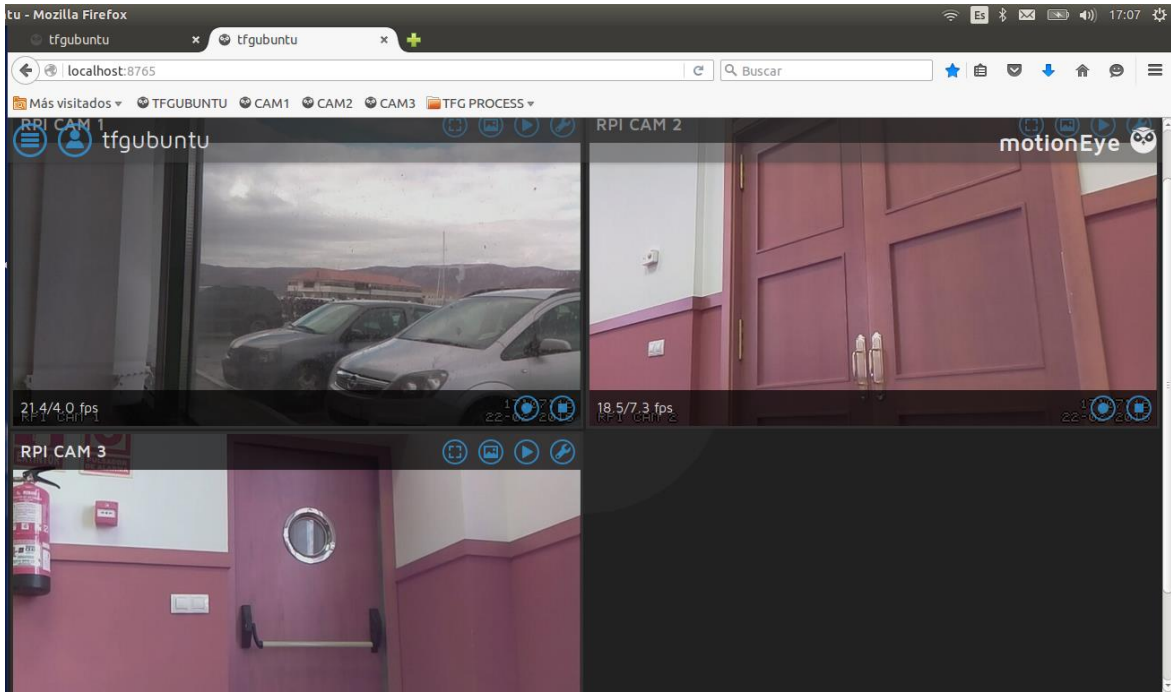


Figure 4-15 Screenshot of MotionEye running on the central server

4.4.1 Results

The test was carried out the 22nd February 2016, it began at 09:34 and ran independently with no supervision until 17:08, for a total time of 7 hours and 34 minutes. During that time, the results showed that the system had functioned correctly during that time. As we can see in Figure 4-16, in the pictures browser, each Raspberry Pi detected movement and recorded both images and video of what it detected.

Each Raspberry Pi, recorded both images and movies in different folders

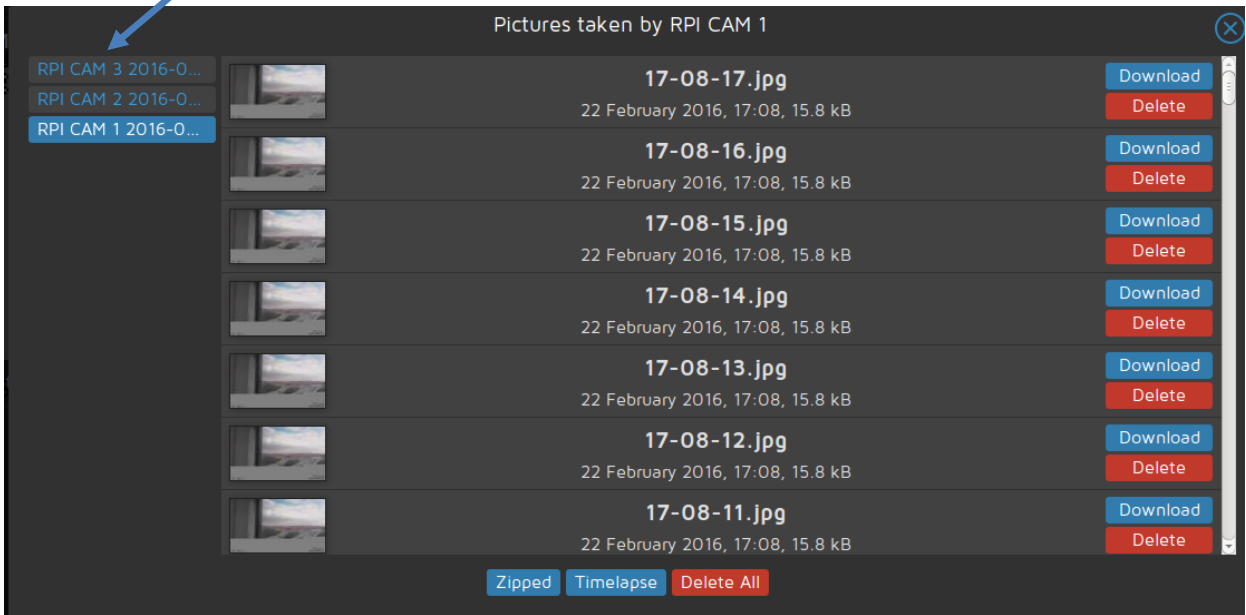


Figure 4-16 Pictures browser of MotionEye

The two Raspberry Pi's that were facing the doors detected movement, in the photos and videos that were recorded it can clearly be seen who it is that enters and at what time as shown in Figure 4-17.



Figure 4-17 Example of an image taken by the system

Because the Raspberry Pi that was facing the window was recording an area that has a lot of movement it had a lot of recordings. In total, the Raspberry Pi facing the windows took 1325 photos (24,5 MB) and 156 videos (94,1 MB). An example of movement detected by the system can be seen in the sequence of photos shown in Figure 4-18. As soon as the system detected the car, it took a picture every second and recorded a movie.



Figure 4-18 Sequence of pictures taken by MotionEye when detecting movement

4.4.2 Conclusions

Based on this test, it is clear that the system functions as expected. Certain features were not tested during this scenario such as night vision, however, they have been tested independently various times as explained in section 4.3. The system proved to be effective and dependable enough to be used as a video surveillance system and a viable alternative to a commercial video surveillance system.

5 CONCLUSIONS AND FUTURE WORK

5.1 Overall conclusions

Having completed the project and revisiting the original objectives it must be determined whether or not these have been met.

The tests carried out in chapter 4, especially the real world test (section 4.4) clearly show that the design and implementation of a video surveillance system using a Raspberry Pi computer and Pi Camera is not only possible but was completed with success.

The system that was designed not only integrates a video feed, but was designed to have multiple video feeds. In this project up to three video feeds were integrated in the system. Upon review of the current technology in the field of video surveillance systems, we find that modern video surveillance systems integrate advanced features. The system that has been developed, integrates a variety of advanced features as well.

Among these features and most importantly we find motion detection. The system is able to analyze the video feed and determine the difference between one frame and another. From this it can determine whether or not there has been any movement. Having detected movement, the system can be configured to take certain actions such as record video, take pictures, send an email, execute a command, etc. These actions are user-configurable allowing for the user to define the quality of the movie, motion detection threshold, etc.

As part of the system, the Pi Camera integrates an infrared board which automatically turns on an infrared light in low light conditions. This allows the system to continue working in total darkness. Each Raspberry Pi achieved the objective of being very portable by integrating both a battery and a Wi-Fi connection. Given the low power consumption of each Raspberry Pi, tests showed that when using high settings the battery life lasted for more than 12 hours.

Another of the features the system has is the possibility of defining different configurations depending on a weekly schedule. This would allow for a scenario in which the system is configured to only detect movement and send notifications in a certain time frame. The system can also be controlled and configured remotely, all that is necessary is to have access to the network in which the central server is connected. In order to ensure proper use of the system, the system is password-protected and even defines two types of access, one being the administrator and the other a surveillance user.

One of the issues that developed during the project was the ability to control the behavior of each Raspberry Pi's remotely. By modifying part of the code of the program the system uses, we are able to

execute user-defined scripts on each Raspberry Pi. Therefore, as part of the project the ability to reboot and shutdown each Raspberry Pi was developed and integrated into the system.

As part of the limitations of the video surveillance system there is an evident downgrade in the quality of the video feeds when adding more cameras to the system. This happens exclusively when using Wi-Fi, the solution therefore is as simple as using Ethernet as shown in section 4.1.3 or using a different Wi-Fi technology with more network bandwidth and speed.

The final and main drawback of the system is the limitation in hardware of the Raspberry Pi's when compared to the top-tier commercial video surveillance systems. However, when compared to more modest commercial video surveillance systems the system is on par and in some cases has more features. The limitation in hardware of the Raspberry Pi is a direct consequence of its cost, making it also its main strength when compared to current commercial systems.

Overall, the designed system proved to have various advanced features and offered a multitude of personalization options while having a very low cost. There are numerous possible upgrades and future lines of study explored in section 5.2.

5.2 Future lines of study

Given the open source nature of both the Raspberry Pi and programs used in this project, there are lots of possible improvements and future lines of study.

To begin with, it would be necessary to make the Raspberry Pi's more robust by putting them in a case such as in Figure 5-1. The ideal case would allow them to be weather-proof so they could be used in the outside.



Figure 5-1 Raspberry Pi inside a CCTV casing (taken from [70])

Another key element to improve would be to include microphone/audio support when recording video. This would greatly improve the capabilities of the system, allowing it even, to be used for videoconference and have sound in the movies recorded.

Within RPi-Cam Web Interface, there is the possibility to integrate a motor with the GPIO pins of the Raspberry Pi and use it to pan and tilt the Pi-Camera as shown in Figure 5-2.



Figure 5-2 Pi Camera mounted on an electric motor with pan and tilt functionality (taken from [71])

Using the action buttons of MotionEye, this functionality could be integrated into the MotionEye interface with a script in a similar way that was applied to the remote reboot/shutdown functionality.

The GPIO pins and the action buttons could be used in many more scenarios. Within the MotionEye interface there is an option to execute commands every time motion is detected. If GPIO pins were connected to a light or a speaker, it could be used as an alarm, turning on every time movement is detected.

Another possible line of improvement would be to integrate a battery to the router and central server so that in the case of blackout, the system would continue working, given that each Raspberry Pi already has a very long battery life and works in the dark.

Finally, taking advantage of the time the Raspberry Pi functions with a battery. If instead of a Wi-Fi connection a 3G/4G modem were connected, it would expand the portability of the system to anywhere that has 3G/4G coverage. This way it could potentially be used on a drone and facilitate its control.

5.3 Personal opinion

The system proposed in this project represents a viable alternative with advanced features. The possibility of detecting movement and performing certain actions enables lots of possibilities. This system would be ideal to monitor closed spaces. It is especially useful if the interface's port is forwarded to the Internet allowing it to be accessed from anywhere.

Having explored and developed this system I strongly believe that with the appropriate implementation, it could be used as an alternative to current commercial video surveillance systems.

6 BIBLIOGRAPHY

- [1] "5.9 million CCTV cameras in UK, BBC" BBC, 11 July 2013. [Online]. Available: <http://www.bbc.co.uk/newsround/23279409>. [Accessed 28 January 2016].
- [2] Welsh, Brandon & Farrington, David, "Public Area CCTV and Crime Prevention: An Updated Systematic Review and Meta-Analysis" Justice Quarterly, 12 October 2009. [Online]. Available: <http://www.tandfonline.com/doi/full/10.1080/07418820802506206>. [Accessed 25 January 2016].
- [3] S. McDowell, "Does the latest Crime Survey Report show that CCTV really does work?" CCTV BLOG, 01 August 2015. [Online]. Available: <https://cctvsupplies.wordpress.com/2015/08/01/does-the-latest-crime-survey-report-show-that-cctv-really-does-work/>. [Accessed 28 January 2016].
- [4] "CCTV - Video Analytics for Traffic Management" [Online]. Available: <http://www.largevents.eu/wp/wp-content/uploads/2012/10/cctv.pdf>. [Accessed 25 January 2016].
- [5] K. Proctor, "The great surveillance boom" Fortune, 26 April 2013. [Online]. Available: <http://fortune.com/2013/04/26/the-great-surveillance-boom/>. [Accessed 25 January 2016].
- [6] P. Wahba, "Shoplifting, worker theft cost retailers \$32 billion last year" Fortune, 24 June 2015. [Online]. Available: <http://fortune.com/2015/06/24/shoplifting-worker-theft-cost-retailers-32-billion-in-2014/>. [Accessed 25 January 2016].
- [7] "Video Surveillance Market worth \$42.06 Billion by 2020" marketsandmarkets.com, November 2014. [Online]. Available: <http://www.marketsandmarkets.com/PressReleases/global-video-surveillance-market.asp>. [Accessed 26 January 2016].
- [8] "Nanny cam records babysitter intentionally burning child" Fox43, 16 April 2015. [Online]. Available: <http://fox43.com/2015/04/16/nanny-cam-records-babysitter-intentionally-burning-child/>. [Accessed 26 January 2016].
- [9] D. D. Mindell, "The science and technology of World War II" The National Museum of World War II., 2009. [Online]. Available: <http://www.learnnc.org/lp/editions/nchist-worldwar/6002>. [Accessed 01 February 2016].
- [10] R, Yoichi, "The central police control station, manned 24 hours a day controls all traffic lights, receives remote tv inputs" Wikipedia, 20 September 2011. [Online]. Available:

https://commons.wikimedia.org/wiki/File:THE Central_Police_Control_Station,_Manned_24_Hours_A_Day_controls_all_traffic_lights,_receives_remote_tv_inputs_from..._-_NARA_-_551905.tif. [Accessed 02 February 2016].

- [11] "*Teddy Bear 450TVL Hidden Nanny Camera*" CCTVCAMERADVRS, [Online]. Available: <http://www.cctvcameradvrs.com/teddy-bear-450tvl-hidden-nanny-camera>. [Accessed 02 February 2016].
- [12] W. Dornberger, V-2, Ballantine Books, 1954.
- [13] Soltech Security, "*600TVL & 720p & 1080p. Is there much of a difference?*" Soltech Security, 29 January 2014. [Online]. Available: <https://soltechsecurity.wordpress.com/2014/01/29/600tvl-720p-1080p-is-there-much-of-a-difference/>. [Accessed 02 February 2016].
- [14] J. D. Lewin, "*Night Vision vs. Thermal Imaging Cameras*" Surveillance One, 10 September 2014. [Online]. Available: <https://survone.com/night-vision-vs-thermal-imaging-cameras/>. [Accessed 02 February 2016].
- [15] IPS, "*IPS Motion Detection (Axis)*" IPS, 2015. [Online]. Available: <http://www.ips-analytics.com/en/products/ips-videoanalytics-new/camera-based/ips-motion-detection-axis.html>. [Accessed 02 February 2016].
- [16] "*Surveillance Station Professional Network Video Recorder (NVR)*" QNAP, 2016. [Online]. Available: <https://www.qnap.com/event/station/en/?page=surveillance-station-w>. [Accessed 02 February 2016].
- [17] III, George N. Root, "*Importance of Video Camera Surveillance in a Workplace*" Demand Media, 2016. [Online]. Available: <http://smallbusiness.chron.com/importance-video-camera-surveillance-workplace-11657.html>. [Accessed 02 February 2016].
- [18] "*Alibi 4-Camera 2.1 Megapixel WDR 65' IR Outdoor IP Video Security System*" Super Circuits, [Online]. Available: <http://www.supercircuits.com/alibi-4-camera-ip-system-sys4421ipd>. [Accessed 02 February 2016].
- [19] "*ZOSI 4PCS Megapixel 720P Wireless Outdoor IP Camera System 30M Night vision with 4CH Security 720P HD Network NVR Wifi Kit Support Smartphone Remote view with 1TB Hard disk*" Amazon, [Online]. Available: <http://www.amazon.com/ZOSI-Megapixel-Wireless-Security-Smartphone/dp/B017ISY85O>. [Accessed 02 February 2016].
- [20] "*ANRAN 4CH Network NVR Wifi Kit Megapixel 720P HD Wireless Outdoor IP Network Home Surveillance Camera System with 1TB Hard Drive Free APP for iPhone Plug&Play*" Amazon, [Online]. Available: <http://www.amazon.com/ANRAN-Network-Megapixel-Wireless-Surveillance/dp/B015PWTFNA>. [Accessed 02 February 2016].
- [21] "*Install Video Surveillance Cameras Cost*" FIXR, 2016. [Online]. Available: <http://www.fixr.com/costs/install-video-surveillance-cameras>. [Accessed 02 February 2016].
- [22] "*Raspberry Pi 2 Complete Camera Kit -- Includes Raspberry Pi 2 900 MHz Quad-Core CPU (1GB) --5MP Camera Module--Edimax EW-7811Un Wi-Fi Adapter--20 Page User Guide--Clear Case--Power Supply--Kingston 8GB Micro SD Card and Micro SD to SD Adapter--HDMI Cable*" AMAZON, [Online]. Available: <http://www.amazon.com/Module--Edimax-Adapter--20-Guide--Clear-Supply--Kingston-Adapter--HDMI/dp/B010MMPZCS>. [Accessed 22 February 2016].
- [23] M. Giles, "*Beyond the PC*" The Economist, 8 October 2011. [Online]. Available: <http://www.economist.com/node/21531109>. [Accessed 02 February 2016].

- [24] "*The Jetsons*" Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/The_Jetsons. [Accessed 22 February 2016].
- [25] "*Home Automation*" HYDE AV, [Online]. Available: <http://www.hyde-av.co.uk/business-av/home-automation/>. [Accessed 02 February 2016].
- [26] A. Colon, "*The Best Home Security Cameras of 2016*" PCMag, 22 January 2016. [Online]. Available: <http://www.pcmag.com/article2/0,2817,2475954,00.asp>. [Accessed 02 February 2016].
- [27] "*Withings Home*" [Online]. Available: <http://www.withings.com/eu/en/products/home>. [Accessed 02 February 2016].
- [28] "*Nest Cam*" [Online]. Available: <https://nest.com/camera/meet-nest-cam/>. [Accessed 02 February 2016].
- [29] "*Logitech Circle Home Monitoring*," Logitech, [Online]. Available: <http://www.logitech.com/en-us/product/circle>. [Accessed 02 February 2016].
- [30] L. Upton, "*Five Million Sold!* " Raspberry Pi, 18 February 2015. [Online]. Available: <https://www.raspberrypi.org/blog/five-million-sold/>. [Accessed 02 February 2016].
- [31] Q. Plummer, "*\$5 Raspberry Pi Zero Has Just Become Fastest-Selling Computer In History*" 30 November 2015. [Online]. Available: <http://www.techtimes.com/articles/111757/20151130/5-raspberry-pi-zero-has-just-become-fastest-selling-computer-in-history.htm>. [Accessed 02 February 2016].
- [32] "*What is a Raspberry Pi?*" Raspberry Pi Foundation, [Online]. Available: <https://www.raspberrypi.org/help/what-is-a-raspberry-pi/>. [Accessed 02 February 2016].
- [33] "*Demand for Raspberry Pi goes through the roof*" 09 March 2012. [Online]. Available: <http://tech4tea.com/blog/2012/05/09/demand-for-raspberry-pi-goes-through-the-roof/>. [Accessed 02 February 2016].
- [34] E. Upton, "*Raspberry Pi – 2006 edition*" 23 October 2011. [Online]. Available: <https://www.raspberrypi.org/blog/raspberry-pi-2006-edition/>. [Accessed 22 February 2016].
- [35] L. Upton, "*15,000 Raspberry Pis for UK schools – thanks Google!*" Raspberry Pi Foundation, 29 January 2013. [Online]. Available: <https://www.raspberrypi.org/blog/15000-raspberry-pis-for-uk-schools-thanks-google/>. [Accessed 02 February 2016].
- [36] "*Resources Raspberry Pi*" Raspberry Pi Foundation, [Online]. Available: <https://www.raspberrypi.org/resources/>. [Accessed 25 January 2016].
- [37] "*Raspbian*" [Online]. Available: <https://www.raspbian.org/>. [Accessed 22 February 2016].
- [38] "*Getting started with SCRATCH*" Raspberry Pi Foundation, [Online]. Available: <https://www.raspberrypi.org/learning/getting-started-with-scratch/>. [Accessed 02 February 2016].
- [39] "*Raspberry Pi Wikipedia*" Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Raspberry_Pi. [Accessed 02 February 2016].
- [40] "*Astro Pi Mission*" Astro Pi Foundation, [Online]. Available: <https://astro-pi.org/about/mission/>. [Accessed 02 February 2016].
- [41] N. Heath, "*'We thought we'd sell 1,000': The inside story of the Raspberry Pi*" ZDNet, 17 January 2013. [Online]. Available: <http://www.zdnet.com/article/we-thought-wed-sell-1000-the-inside-story-of-the-raspberry-pi/>. [Accessed 02 February 2016].

- [42] R. Bishop, "*Raspberry Pi: Past, Present and Future*" in *Bacon*, London, 2014.
- [43] D. Bryan, "*Raspberry Pi Power Cat Feeder – Updates*" 20 December 2013. [Online]. Available: <http://drstrangelove.net/2013/12/raspberry-pi-power-cat-feeder-updates/#more-16>. [Accessed 02 February 2016].
- [44] "*Raspberry Pi Foundation*" [Online]. Available: <https://www.raspberrypi.org/>. [Accessed 22 February 2016].
- [45] "*Python, Raspberry Pi Foundation*" Raspberry Pi Foundation, [Online]. Available: <https://www.raspberrypi.org/documentation/usage/python/>. [Accessed 02 February 2016].
- [46] "*Raspberry Pi Learning Resource Pi Camera*" Raspberry Pi Foundation, [Online]. Available: <https://www.raspberrypi.org/learning/python-picamera-setup/worksheet.md>. [Accessed 03 February 2016].
- [47] D. Jones, "*picamera-python read the docs*" 2014. [Online]. Available: <http://picamera.readthedocs.org/en/release-1.10/index.html>. [Accessed 02 February 2016].
- [48] Raymond CC, "*12 Software to Turn your Computer into Video Surveillance System*" 2014. [Online]. Available: <https://www.raymond.cc/blog/set-up-logitech-webcam-as-motion-detector-and-send-alert-via-sms-text-message/>. [Accessed 02 February 2016].
- [49] P. Dickson, "*How to Setup Home Surveillance with Zoneminder on Linux OS*" 30 April 2015. [Online]. Available: <http://www.pddesign.com.au/blog/articles/2014/home-surveillance-with-zoneminder-using-linux-pg03.php#.VrNvw7WEqM8>. [Accessed 4 February 2016].
- [50] C. Crisan, "*MotionEye*" 12 December 2015. [Online]. Available: <https://github.com/ccrisan/motioneye/wiki>. [Accessed 22 February 2016].
- [51] C. Crisan, "*Screenshots of MotionEye*" 4 September 2015. [Online]. Available: <https://github.com/ccrisan/motioneye/wiki/Screenshots>. [Accessed 4 February 2016].
- [52] C. Crisan, "*MotionEye Features*" 27 November 2015. [Online]. Available: <https://github.com/ccrisan/motioneye/wiki/Features>. [Accessed 4 February 2016].
- [53] C. Crisan, "*MotionEyeOS*" [Online]. Available: <https://github.com/ccrisan/motioneyeos>. [Accessed 22 February 2016].
- [54] "*RPi-Cam-Web-Interface*" 29 January 2016. [Online]. Available: <http://elinux.org/RPi-Cam-Web-Interface>. [Accessed 08 February 2016].
- [55] "*sweet pi-sweet home*" 6 March 2014. [Online]. Available: <https://www.sweetpi.de/blog/783/raspberry-pi-kamera-als-live-webcam-mit-aufnahmefunktion>. [Accessed 08 February 2016].
- [56] "*SSH Tutorial for Linux*" SUSO, [Online]. Available: http://support.suso.com/supki/SSH_Tutorial_for_Linux. [Accessed 23 February 2016].
- [57] "*Ubuntu*" Canonical, [Online]. Available: <http://www.ubuntu.com/>. [Accessed 17 February 2016].
- [58] "*Set up Gmail with Outlook, Apple Mail, or other mail clients*" Google, [Online]. Available: <https://support.google.com/mail/troubleshooter/1668960?hl=en#ts=1665018%2C1665144>. [Accessed 23 February 2016].
- [59] "*NOOBS Guide*" Raspberry Pi Foundation, [Online]. Available: <https://www.raspberrypi.org/documentation/installation/noobs.md>. [Accessed 23 February 2016].

- [60] "NOOBS" Raspberry Pi Foundation, q. [Online]. Available: <https://www.raspberrypi.org/downloads/noobs/>. [Accessed 23 February 2016].
- [61] "Apache HTTP Server Project" The Apache Software Foundation, 2016. [Online]. Available: <https://httpd.apache.org/>. [Accessed 01 March 2016].
- [62] "NGINX" NGINX INC., 2016. [Online]. Available: <https://www.nginx.com/>. [Accessed 01 March 2016].
- [63] C. Crisan, "Install on Ubuntu" 06 December 2015. [Online]. Available: <https://github.com/ccrisan/motioneye/wiki/Install-On-Ubuntu>. [Accessed 23 February 2016].
- [64] C. Crisan, "Installation" 12 September 2015. [Online]. Available: <https://github.com/ccrisan/motioneye/wiki/Installation>. [Accessed 23 February 2016].
- [65] "pip 8.0.2" Python Software Foundation, 2015. [Online]. Available: <https://pypi.python.org/pypi/pip/>. [Accessed 23 February 2016].
- [66] C. Crisan, "Action Buttons" 16 December 2015. [Online]. Available: <https://github.com/ccrisan/motioneye/wiki/Action-Buttons>. [Accessed 23 February 2016].
- [67] Amazon, "Raspberry PI 5MP Camera Board Module" Amazon, [Online]. Available: <http://www.amazon.com/Raspberry-5MP-Camera-Board-Module/dp/B00E1GGE40>. [Accessed 02 February 2016].
- [68] E. Upton, "Raspberry Pi 2 on sale now at \$35" Raspberry Pi Foundation, 02 February 2015. [Online]. Available: <https://www.raspberrypi.org/blog/raspberry-pi-2-on-sale>. [Accessed 02 February 2016].
- [69] "Surveillance Station Professional Network Video Recorder (NVR)" QNAP, 2016. [Online]. Available: <https://www.qnap.com/event/station/en/?page=surveillance-station-w>. [Accessed 02 February 2016].
- [70] C. Crisan, "Github Motioneye" 1 December 2015. [Online]. Available: <https://github.com/ccrisan/motioneye/blob/master/motioneye/static/img/camera-action-buttons.svg>. [Accessed 23 February 2016].
- [71] C. Buenger, "Raspberry Pi as low cost HD surveillance camera" 12 May 2014. [Online]. Available: <http://www.codeproject.com/Articles/665518/Raspberry-Pi-as-low-cost-HD-surveillance-camera>. [Accessed 24 February 2016].
- [72] "Pan and tilt mechanism" OpenElectrons, 20 September 2013. [Online]. Available: <https://blog.adafruit.com/2013/09/20/pipan-pan-and-tilt-mechanism-for-raspberry-pi-camera-raspberrypi-piday-raspberrypi/>. [Accessed 23 February 2016].
- [73] "RaspbianFAQ" [Online]. Available: https://www.raspbian.org/RaspbianFAQ#I.27m_new_to_Raspberry_Pi_and_Raspbian.__What_should_I_know.3F. [Accessed 12 02 2016].

