CENTRO UNIVERSITÁRIO DE BRASÍLIA - UNICEUB

FACULDADE DE TECNOLOGIA E CIÊNCIAS SOCIAIS APLICADAS – FATECS

CURSO DE ENGENHARIA DA COMPUTAÇÃO

MATHEUS PAZ COUTINHO

SISTEMA DE MONITORAMENTO RESIDENCIAL

Orientador: Prof. MsC Francisco Javier de Obaldia Diaz

BRASÍLIA, DF – BRASIL

JULHO DE 2016

MATHEUS PAZ COUTINHO

SISTEMA DE MONITORAMENTO RESIDENCIAL

Trabalho de Conclusão de Curso apresentado à Banca examinadora do curso de Engenharia da Computação da FATECS – Faculdade de Tecnologia e Ciências Sociais Aplicadas – Centro Universitário de Brasília como requisito para obtenção do título de Bacharel em Engenharia da Computação.

Orientador: Prof. MsC Francisco Javier de Obaldia Diaz

BRASÍLIA, DF – BRASIL

JULHO DE 2016

MATHEUS PAZ COUTINHO

SISTEMA DE MONITORAMENTO RESIDENCIAL

Trabalho de Conclusão de Curso apresentado à Banca examinadora do curs de Engenharia da Computação da FATECS – Faculdade de Tecnologia e Ciências Sociais Aplicadas – Centro Universitário de Brasília como requisito para obtenção do título de Bacharel em Engenharia da Computação.

Orientador: Prof. MsC Francisco Javier de Obaldia Diaz

BANCA EXAMINADORA

Prof.º MsC. Francisco Javier De Obaldía Díaz
Orientador

Prof.º MsC. Marco Antonio Araújo
UniCEUB

Prof.º Dr. Sidney Cerqueira Bispo dos Santos
UniCEUB

BRASÍLIA, DF – BRASIL

JULHO DE 2016

AGRADECIMENTOS

Agradeço primeiramente à minha família por tudo que ela me ensinou sobre a vida acadêmica e profissional e também aos meus amigos e colegas que conquistei nestes anos em que batalhei pelo título de engenheiro. Também desejo prestar a minha gratidão à instituição UniCEUB que proporcionou os meios e ferramentas para que eu conseguisse o conhecimento necessário para alcançar um lugar no mercado de trabalho.

SUMÁRIO

LISTA DE FIGURAS

RESUMO

Este trabalho propõe um sistema de monitoramento residencial que busca diminuir o tempo entre o acontecimento de uma invasão e o conhecimento do usuário. Para o desenvolvimento do projeto, foram utilizados um Raspberry PI, uma Raspicam e um sensor de presença PIR. Os componentes foram configurados e testados de modo que, ao ser percebida movimentação no ambiente monitorado, o sistema seja capaz de notificar o usuário com o envio de e-mail com imagem em anexo mostrando o local onde o movimento foi percebido pelo sensor PIR. Além disso, o sistema também disponibiliza uma página WEB com o *streaming* de vídeo, com o intuito de possibilitar o cliente a monitorar sua residência. Após o desenvolvimento do projeto, foi possível constatar, através da realização de testes, que os resultados foram satisfatórios apresentando uma redução significativa no tempo de ciência do cliente a respeito de uma invasão residencial e apenas 11.53% de erro nos testes de funcionalidade.

Palavras-chave: Monitoramento, Raspberry PI, Webpage, Streaming, sensor

ABSTRACT

This paper proposes a home monitoring system that seeks to reduce the time between the event of an invasion and the user's knowledge. For the development of the project, it was used a Raspberry PI, a Raspicam and a PIR sensor. The components are configured and tested so that if it is perceived movement by the monitored environment, the system is able to notify the user by sending e- mail with attached image showing where the movement took place. In addition, the system also offers a web page with the streaming video, in order to enable the customer to monitor his residence. After the development of the project, it was established through testing that the results were satisfactory, showing a significant reduction in customer science of time of a home invasion and only 11.53 % error in functionality testing.

Key words: Monitoring system, Raspberry PI, PIR sensor, Raspicam

CAPÍTULO 1 – INTRODUÇÃO

1.1 Apresentação do Problema

O número de crimes contra o patrimônio, que incluem roubos e furtos, em residências no Distrito Federal subiu em 17,5%, entre 2014 e 2015, e esse número tende a crescer para o ano de 2016. Esses crimes foram anunciados pela secretaria de Segurança Pública Márcia de Alencar, como prioridade para o ano de 2016 (LUIZ, 2016)

A falta de um sistema de segurança na residência pode acarretar um tempo de espera muito grande até que o crime seja percebido pela vítima, dificultando ainda mais a ação das autoridades competentes.

Com isso, a implementação de um sistema de segurança que ofereça os benefícios de uma resposta rápida para a vítima poderia ajudar na rápida identificação do criminoso ou mesmo de apoio à ação policial, oferecendo mecanismo adicional na proteção do patrimônio residencial.

1.2 Objetivos do Trabalho

Desenvolver um sistema de monitoramento, para ser implementado em residências, a fim de auxiliar o cliente a monitorar sua residência e diminuir o tempo de resposta a uma eventual invasão do ambiente interno.

1.2.1 Objetivos específicos:

- Diminuir o tempo de ciência do usuário em caso de uma violação no perímetro residencial monitorado.
- Desenvolver o código em linguagem Python para o módulo de monitoramento

- Promover a comunicação entre o protótipo e o usuário via e-mail
- Disponibilizar uma página web ao cliente com o streaming do sistema de monitoramento

1.3 Justificativa e Importância do Trabalho

O crescente número de vítimas de crimes contra o patrimônio e a dificuldade de recuperação dos bens perdidos devido à demora na efetivação de procedimento de ocorrência policial, proporciona um grande desconforto e uma crescente preocupação aos moradores de áreas afetadas por este tipo de crime.

Com isso em mente, a solução proposta de se desenvolver um sistema de monitoramento irá reduzir a insegurança proporcionada pelo aumento da criminalidade, uma vez que o tempo entre a invasão domiciliar e a ciência do morador sobre o problema será reduzido.

1.4 Escopo do Trabalho

Será desenvolvido um dispositivo por meio da plataforma Raspberry Pi com sensor de presença e uma mini câmera utilizando a tecnologia de infravermelho em conjunto com um Roteador Wireless para se conectar à rede e transmitir o *streaming* do vídeo e o e-mail de notificação de movimentação.

Os dados do *streaming* multimídia serão transmitidos pelo Raspberry Pi e estarão acessíveis ao cliente por meio de uma página web, que permitirá a visualização do local monitorado.

1.5 Resultados Esperados

O sistema será capaz de:

- Transmitir um streaming de vídeo de boa qualidade, ou seja, o usuário deverá ser capaz de identificar as imagens com clareza e precisão;
- Transmitir o streaming de vídeo via página Web;
- Notificar o usuário, por meio de um e-mail, caso haja alguma movimentação no ambiente monitorado;

1.6 Estrutura do Trabalho

O conteúdo deste trabalho foi estruturado em seis capítulos onde cada um deles terá um propósito, conforme a descrição a seguir:

No **capítulo 1** é apresentada uma breve introdução sobre o tema a ser tratado nesse projeto, contendo os objetivos, as justificativas, o escopo e os resultados esperados do projeto. O **capítulo 2** apresenta a proposta em concordância com o estudo realizado, ilustrando o modelo desenvolvido para a resolução do problema. **O capítulo 3** traz a aplicação prática do modelo proposto, os custos do projeto e os resultados obtidos com ele. **O capítulo 4** apresenta os testes e seus resultados, realizados para garantir o funcionamento do sistema. E por fim, o **capítulo 5** apresenta as conclusões finais e as possíveis sugestões para trabalhos futuros.

CAPÍTULO 2 - REFERENCIAL TEÓRICO E BASES METODOLÓGICAS

Este capítulo descreve os fundamentos técnicos e de metodologia, utilizados no desenvolvimento do projeto. São citados os assuntos mais relevantes e suas respectivas fontes, utilizadas como referência para este trabalho. O leitor pode consultar as fontes citadas, caso deseje explorar mais sobre assuntos aqui expostos.

2.1 Automação e controle

O controle de sistema é uma ideia que vem se desenvolvendo durante muitos anos, passando por diversas etapas da civilização. Existem diversos mecanismos capazes de mover ou atuar em um objeto a partir de um comando ou sinal de um sensor. O controle de retroalimentação é uma forma do próprio sistema se adaptar a uma situação variante, por exemplo, o controle de nível por boias que foi desenvolvido na Grécia (300 a.C) (FACCIN, 2004), esse é o primeiro indício do controle de retroalimentação já registrado.

Os resultados de sistemas que visam a automaticidade com controle de suas atividades, possuem resultados mais confiáveis e seguros, com isso garantindo uma diminuição na quantidade de erros e falhas oriundas da linha de processo do sistema, podendo também atingir um impacto socioambiental positivo, com menos entulho e desperdício de materiais. (OGATA, 2003). Uma vez que os métodos de controle se provaram eficientes em indústrias e linhas de produção, as residências incorporaram essa tecnologia, abrindo um grande mercado e pesquisas para a automação residencial.

A Figura 2.1, faz referência ao controle de retroalimentação, que é uma das formas de se garantir a automação do sistema. O elemento de medida é o sensor representado pelo H1(s), que capta as informações do meio, como por exemplo, a boia desenvolvida pelos gregos que foi citada acima. O resultado do sensor é então interpretado causando uma modificação no comportamento do Atuador, representado pelo H2(s). O sistema de retroalimentação irá garantir que as variações no meio não interfiram de forma brusca com o

comportamento desejado do sistema. A fórmula abaixo, extraída do livro Fundamentos de Sinais e Sistemas (ROBERTS, 2009), demonstra a Função de Transferência de Malha Fechada (FTM), representado pela razão entre (Y(s)) e (X(s)).

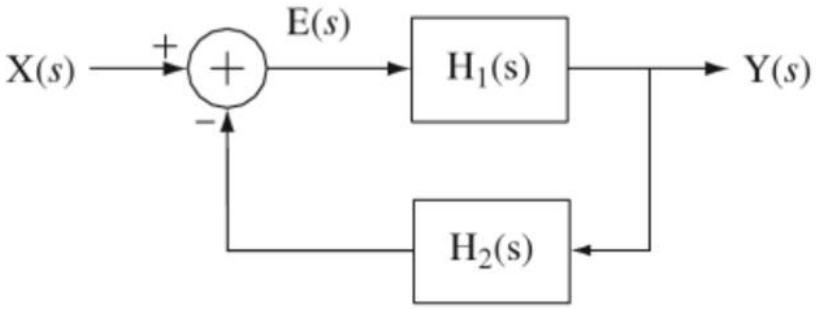

Figura 2.1 Conexão por retroalimentação de sistemas

Fonte: (ROBERTS, 2009)

$$H(S) = \frac{(Y(s))}{(X(s))} = \frac{(H1(s))}{(1 + H1(s)H2(s))}$$

Devido à natureza binária do projeto, o controle por retroalimentação não será utilizado, mas sim o controle automático de duas posições, controle *on-off*, explicado a seguir.

2.1.1 Controlador automático

O modelo de controle automático possui três entidades básicas, os controladores, atuadores e os sensores. Os sensores do sistema têm o objetivo de ler a situação do sistema, como, por exemplo, a temperatura de um forno ou a distância de um obstáculo a um veículo automotor. O controlador, por sua vez, recebe as informações geradas pelos sensores, executa um procedimento de comparação com os valores de controle pré-definidos e aciona os atuadores, que irão executar o sinal de ação do controlador. Quando o controlador aciona o

atuador, a saída do sistema é medida novamente pelos sensores, realimentando o sistema com as informações do sistema atualizadas. Esse ciclo será mantido enquanto o sistema não atingir valores de estabilidade aceitos pelo controlador (OGATA, 2003).

Todos os sistemas reais estão sujeitos a interferências externas e internas, um modelo de controle automático fará com que o sistema atinja um nível de estabilidade que não irá deteriorar o resultado esperado. Sistemas com níveis de instabilidade elevados podem causar problemas na linha de produção, elevando os gastos com matéria prima e com reparos dos equipamentos envolvidos, ou até mesmo causar acidentes graves para os operários. Logo, o sistema que possui um modelo de controle automático adequado, possuirá uma maior confiabilidade no resultado final.

Dentro do controle automático existem várias formas do controlador determinar a ação de controle a ser tomada, ou seja, como o controlador irá mandar os sinais para os atuadores. As ações de controle possuem desvantagens, vantagens e indicações para cenários específicos. Logo, a escolha da ação de controle precisa ser tomada com um estudo completo do comportamento do sistema, pois a escolha de uma ação errada poderá aumentar o nível de instabilidade do sistema.

As ações de controle mais comuns são: duas posições *on-off*, proporcional, integral proporcional-integral, proporcional-derivativo e proporcional-integral-derivativo (OGATA, 2003).

2.1.2 Controlador de duas posições *on-off*

A ação de controle *on-off*, possui apenas dois estados: ligado e desligado. É relativamente simples e possui um custo baixo. O controlador *on-off* é bastante usado em sistemas de controle industriais como mecanismo de segurança contra acidentes domésticos e em caso de acionamento de aparelhos. Existe um tempo de espera, *delay*, para o sinal de erro atuante se movimentar antes do acionamento do chaveamento do atuador. Esse tempo de espera é denominado de intervalo diferencial, ou histerese diferencial. A histerese diferencial pode ser causada por uma vagarosa ação mecânica, comparada com um impulso elétrico, ou introduzida no sistema de forma intencional para prevenir operações frequentes do mecanismo *on-off* (OGATA, 2003).

Considerando o sinal de saída do controlador u(t) e o sinal de erro e(t), o sinal de saída irá permanecer ou em um valor máximo (M1) ou em um valor mínimo (M2), dependendo se o sinal do erro for positivo ou negativo (BAPTISTA, 2013). O diagrama de blocos que representa o controle *on-off* é mostrado pela Figura 2.2 e pela sua fórmula, em seguida:

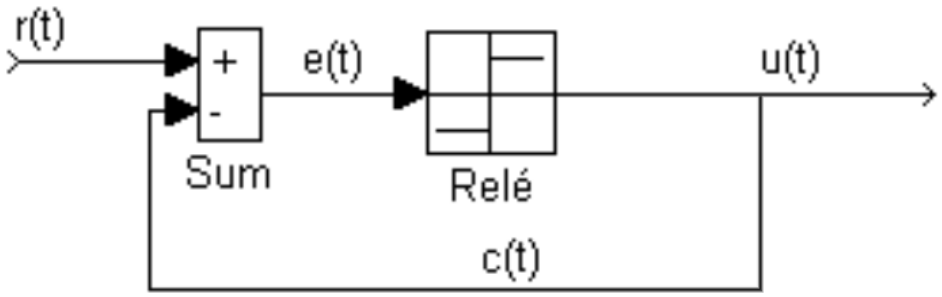

Figura 2.2 Diagrama de blocos de um controlador On-Off

Fonte: Adaptado de (BAPTISTA, 2013)

$$u(t) = \begin{cases} M1 & e > 0 \\ M2 & e < 0 \end{cases}$$

No projeto o controle *on-off* será utilizado em conjunto com o sensor de presença PIR, de forma que o valor máximo (M1) indica movimentação no ambiente e o valor mínimo (M2) indica que no ambiente monitorado não há movimento.

O controle *on-off* será utilizado em conjunto com o sensor de presença, monitorando seus dois estados: ativo, que indica movimentação no ambiente, e inativo, que indica que no ambiente monitorado não há movimento.

2.2 Linguagens de programação

A linguagem de baixo nível, utilizada pela máquina, é definida pelo design do componente físico do aparelho, e pode ser reduzida a zeros e uns, ou seja, a linguagem de baixo nível é como as instruções de um computador são executadas. Com o avanço das tecnologias e da complexidade dos computadores foi percebida a necessidade de se gerar

comandos que seriam interpretados pela máquina e, por fim, executados. A linguagem *Assembly*, conjunto de comandos de comunicação com o *hardware*, passou a ser utilizada por programas tradutores, efetuando a conversão dos comandos em linguagem de máquina (DEITEL; HARVEY, 2010).

O crescimento das demandas e do mercado de computadores, fez com que linguagens de alto nível fossem desenvolvidas para acelerar o processo de manipulação de informações oriundas dos computadores. Essas linguagens de alto nível são uma interpretação humana para a linguagem de máquina e visam fornecer ferramentas, para que uma simples instrução possa efetuar uma grande sequência de tarefas. Por serem linguagens para humanos, as linguagens de alto nível precisam de um tradutor, chamado compilador, que converte as instruções efetuadas para a linguagem de máquina. A compilação de um algoritmo robusto em uma linguagem de alto nível pode consumir muito processamento de uma máquina e, por isso, foram projetados programas interpretadores para executar essas instruções diretamente, diminuindo o consumo de *hardware* e o tempo gasto para ser executado (DEITEL; HARVEY, 2010).

2.2.1 Linguagem PHP 5

A linguagem de programação PHP: Hypertext Preprocessor, é bastante utilizada como ferramenta de desenvolvimento web, por ser uma linguagem com código aberto (open source). Esta linguagem foi criada em 1994 por Ramus Lerdof, em 1997 fortemente modificada pelos senhores Andi Gutmans e Zeev Suraski de Tel Aviv, e em 2014 chegou em sua mais nova versão, o PHP 5.0 (PHP, 2015).

É uma linguagem que facilita a programação tanto de códigos simples quanto de códigos complexos. Sua sintaxe é similar ao C, Java e ao Perl. O PHP é executado pelo servidor, ou seja, o usuário final só irá ter acesso ao código HTML gerado por ele (PHP, 2015).

O PHP 5.0 é utilizado nesse projeto para o desenvolvimento da página web onde o *streaming* de vídeo do módulo de câmera de segurança vai ser hospedado. O PHP 5.0 foi utilizado por sua fácil codificação, uma vez que é uma linguagem de programação orientada a objeto, e por ser uma linguagem para desenvolvimento de páginas web.

2.2.2 Linguagem Python 3

Criado por Guido Van Rossum a linguagem Python foi utilizada nesse projeto pela facilidade de integração com a plataforma Raspberry Pi, pela sua facilidade de codificação devido a sua estrutura orientada a objeto, e por ser uma linguagem com variáveis dinâmicas, ou seja, uma mesma variável pode receber diversos tipos de valores. Exemplo disso é que uma variável pode ser um texto e numa linha seguinte pode ser um inteiro (PYTHON, 2015).

Essas características da linguagem serão utilizadas amplamente no desenvolver do código de alerta a invasão, ou seja, o código em Python é responsável por ler o sinal do sensor e enviar o e-mail de alerta ao usuário cadastrado.

2.3 Materiais utilizados

Esta seção é reservada para a descrição dos materiais utilizados no projeto, sua função, integração com outros componentes e suas justificativas de uso.

Os materiais utilizados foram: sensor de presença, plataforma Raspberry e módulo de câmera com infravermelho, que são apresentados com mais detalhamento na sequência.

2.3.1 Sensor de presença

Com capacidade para identificar deslocamento de matéria a certa distância, os sensores de presença do tipo PIR (Passive Infrared) foram utilizados. Eles são capazes de analisar a luz infravermelha emitida por corpos que emitem calor, realizando, assim, sua tarefa de monitorar movimentação no ambiente.

Os dispositivos PIR (Figura 2.3) dispõem de dois apetrechos sensíveis à luz infravermelha que são orientados para o ambiente. No mercado, esses sensores possuem variável sensibilidade. O efeito ocelar das lentes utilizadas no dispositivo, estabelecem diferentes características de alcance, raio e padrão de detecção. As lentes utilizadas são de

plástico do tipo Fresnel, inventadas pelo físico Augustin Fresnel. Estas são compactas e condensam a luz, de modo a oferecer uma maior quantidade de raios infravermelhos para o sensor.

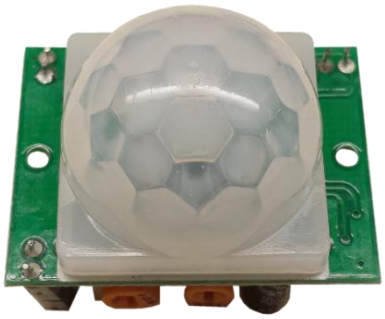

Figura 2.3 Sensor PIR

Fonte: Disponível em <http://eecs.oregonstate.edu/education/images/pir_large.png>.

Em sua inatividade, os dois sensores recebem a mesma quantidade de IR (Infravermelho), no entanto, quando um corpo que emite radiação IR entra no raio de detecção do dispositivo o primeiro sensor percebe a diferença de radiação. Essa diferença leva a uma mudança diferencial positiva entre os sensores (3,3V). Quando este mesmo corpo se afasta do alcance do sensor, outro sinal é gerado, provocando uma mudança diferencial negativa (-3,3).

O PIR HC-SR501 é utilizado no projeto como sensor de presença. Sua tensão de operação varia de 5 a 20 volts, o sinal de saída binário (0 volts - 3,3 volts), e possui *delay* ajustável de 0,3 a 5 minutos. Seu raio de alcance é de 120° até 7 metros (ajustável) e funciona de -15°C até 70°C. Esse dispositivo foi selecionado para o projeto por possuir baixo custo e distância e *delay* ajustáveis.

2.3.2 **Plataforma *Raspberry***

Desenvolvida pela fundação Raspberry PI, no Reino Unido, a plataforma Raspberry é um computador de baixo custo, pequeno, Open Source e com interfaces para vários periféricos. De acordo com a fundação, esse dispositivo foi desenvolvido para aperfeiçoar os

métodos de ensino da área computacional para pessoas das mais diversas idades. O modelo mais recente lançado é o Raspberry Pi 2 Modelo B (Figura 2.4), projetado para substituir a versão anterior, o Raspberry Modelo B+. Este possui um processador ARM Cortex-A7 com quatro núcleos de 900MHz, memória RAM de 1GB compartilhada com a GPU (Graphics Processing Unit), GPU Broadcom VideoCore IV 3D, alimentação de 5V, 4 portas com interface USB 2.0, 40 pinos E/S de uso geral (GPIO), porta HDMI (High-Definition Multimedia Interface), porta Ethernet 10/ 1000MB com interface RJ-45, interface de cartão SD, interface para câmera e interface para um display.

Figura 2.4 Raspberry PI

Fonte: Raspberry Pi Foundation (2015)

Existem várias versões diferentes de sistemas operacionais (SOs) compatíveis com a plataforma, sendo que o SO do Raspberry é instalado no cartão SD do dispositivo. Para realizar a instalação de um sistema operacional, é preciso formatar o cartão no sistema de arquivos FAT32 (File Allocation Table). No site do Raspberry, o sistema operacional Raspbian Jessie (versão do *Kernel 4.1)*, que é baseado em *Linux (Debian)*, é disponibilizado para os usuários, e foi o SO utilizado para realização deste projeto. Para que a plataforma seja colocada em uso, é preciso que seja feita previamente a instalação e configuração do Raspbian Jessie. Os periféricos necessários são: um monitor, um teclado e um mouse. Com um cabo *ethernet* é possível ter acesso a uma rede LAN *(Local Area Network*) local.

Devido à facilidade de programar na plataforma, seja utilizando o terminal do sistema operacional ou utilizando Ambiente Integral de Desenvolvimento (IDEs) de terceiros como o Eclipse, que possibilita a programação em várias linguagens, e sua alta capacidade de

processamento aliada ao baixo custo, essa plataforma foi escolhida como a principal para o desenvolvimento do projeto.

A Figura 2.5 apresenta o esquemático dos pinos GPIO (*General Purpose Input/Output*) que estão presentes no Raspberry Pi. Ele possui dois pinos de entrada de tensão de 3.3V, uma no pino número 1 e outro no pino de número 17. Ele também possui dois pinos de entrada de tensão de 5V, localizados nos pinos número 2 e 4. Existem oito pinos de aterramento espalhados pela esquemática apresentado, que são representados pelos números 6, 9, 14, 20, 25, 30, 34 e 39. O restante são pinos de GPIO que podem ser utilizados para o uso de periféricos em conjunto com o Raspberry Pi (RASPBERRY FOUNDATION, 2015).

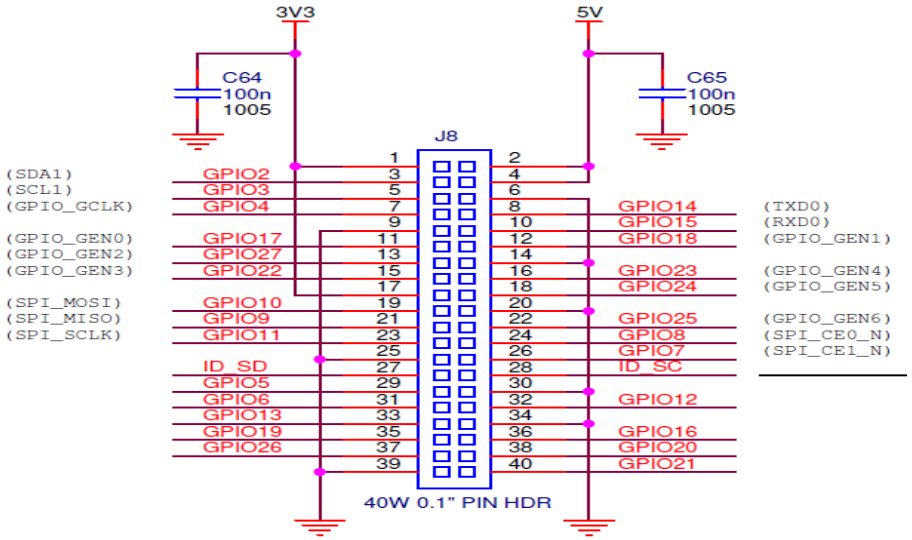

Figura 2.5 Pinagem do Raspberry PI

Fonte: Raspberry Pi Foundation (2015)

2.3.3 Módulo de câmera com infravermelho

O Módulo de câmera, produzido pela organização Raspberry Pi, pode ser utilizada para realizar a captura de fotos e vídeos em alta definição. Ela foi desenvolvida para ser compatível com qualquer modelo da placa Raspberry Pi e utiliza a porta CSI para realizar sua conexão. Possui uma resolução de cinco Megapixels, capaz de realizar a captura de fotos de 2592 x 1944 pixels e filmar a 1080p30(1920 x 1080 a 30 quadros por segundo), 720p60 (1280 x 720 pixels a 60 quadros por segundo) e VGA90 (640 x 480 pixels a 90 quadros por

segundo). É equipada com um sensor OmniVision OV 5647 e possui um *flash* de LED (Light *Emitting Diode)* além de possuir diversas bibliotecas voltadas à utilização dessa câmera, incluído a Picamera, uma biblioteca desenvolvida na língua Python (RASPBERRY FOUNDATION, 2015).

A Figura 2.6 mostra como é o módulo da câmera escolhido para o projeto devido à compatibilidade com a placa Raspberry PI; sendo de pequeno porte e baixo peso, além de possuir diversas bibliotecas compatíveis para a utilização no projeto. Graças aos focos de luz infravermelha é capaz de produzir vídeos mesmo em baixa luminosidade.

Figura 2.6 Módulo de câmera com infravermelho

Fonte: Disponível em <http://www.dx.com/p/5-0mp-lens-camera-ir-infrared-night-vison-led-board-for-raspberry-pi-b-b-black-390047#.VzOw1_krKUl>.

2.3.4 Roteador Wireless

O roteador Wireless IEEE 802.11g modelo ENHWI-G (Figura 2.7), produzido pela corporação Encore Eletronics, possui 4 (quatro) portas para cabo Ethernet, 1 (uma) porta WAN para ADSL, opera na infraestrutura do IEEE 802.11b/g, suporta segurança de criptográfica WPA, WPA-psl, filtro pelo endereço MAC e por protocolos, possui uma página WEB acessível pelo *browser* que permite diversas configurações de preferencias, utiliza o protocolo de comunicação TCP/IP, sua frequência sem fio varia entre 2412 e 2484 MHz e

podem ser divididas em 4 canais, utiliza como fonte de energia um adaptador de 5V e 2.5ª, pode operar entre as temperaturas de 0ºC e 40ºC.

O roteador ENHWI-G foi utilizado no projeto para fornecer a rede na qual o sistema irá disponibilizar o *stream* do vídeo disponibilizado pela Raspicam e acesso à internet para o envio do e-mail de notificação no caso de uma invasão domiciliar.

Figura 2.7 Wireless IEEE 802.11G, modelo ENHWI-G

Fonte: Disponível em <http://www.encore-usa.com/br/node/296#>.

Os aspectos técnicos e fundamentos aqui apresentados, assim como os componentes são fundamentais para o desenvolvimento do projeto conforme será apresentado no próximo capítulo.

CAPÍTULO 3 - DESENVOLVIMENTO DO PROTÓTIPO

Este capítulo apresenta o desenvolvimento do projeto, abrangendo as etapas de configuração dos seguintes itens: Raspberry PI, módulo de câmera, *stream* do vídeo de monitoramento, e finalmente da configuração do sensor de presença PIR e o envio de e-mail.

Essas etapas, em sua devida ordem apresentada, vão produzir o sistema de monitoramento residencial, utilizando os conceitos e materiais apresentados no capítulo anterior e que serão descritos com maior detalhamento a seguir.

3.1 Visão Geral do Projeto

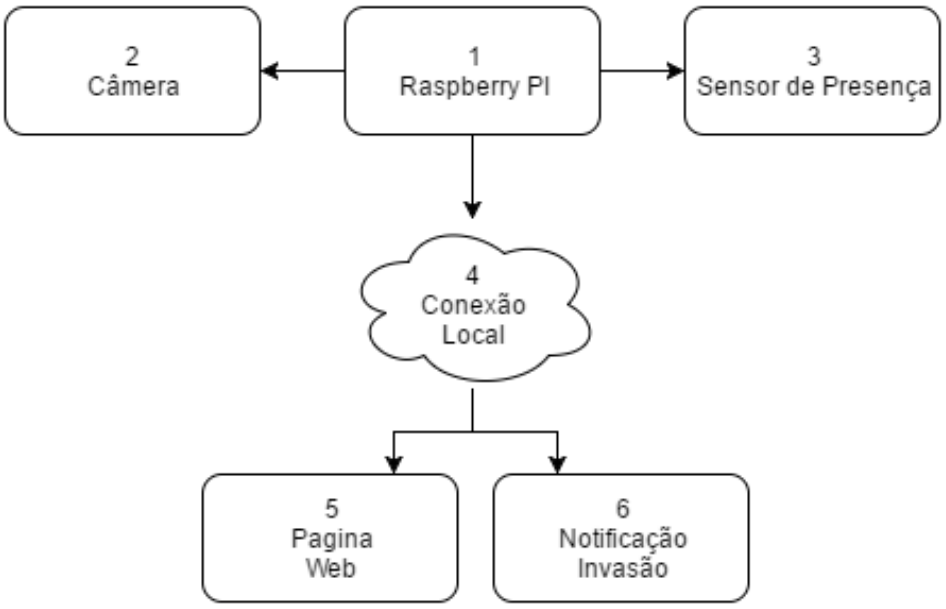

Figura 3.1 Visão Geral do Projeto

Fonte: Elaborado pelo autor.

Como mostrado na Figura 3.1, o protótipo é dividido em seis partes que se interligam. Cada uma dessas partes será descrita como uma etapa, sendo enumeradas de acordo com seu sequencial lógico para o funcionamento correto do sistema proposto.

Primeiramente foi realizada a etapa de configuração do Raspberry PI, em que entra a instalação do sistema operacional Raspbian Jessie, e do periférico do módulo da câmera. Tendo feito as devidas configurações, que serão abordadas com maiores detalhes nos tópicos a seguir, é realizada a configuração e codificação do sensor de presença PIR, em conjunto com a notificação de movimentação no ambiente monitorado, e finalmente a configuração da página web, que será o ponto de acesso ao sistema de monitoramento pelo cliente.

O sistema proposto visa reduzir o tempo de ciência do cliente na situação de invasão domiciliar. O Raspberry PI possui um servidor apache que disponibiliza um *stream* de imagens por uma aplicação web e também um serviço rodando automaticamente desenvolvido em Python para o monitoramento e envio dos e-mails ao usuário cadastrado de modo a realizar o objetivo deste projeto.

3.2 Etapas do desenvolvimento

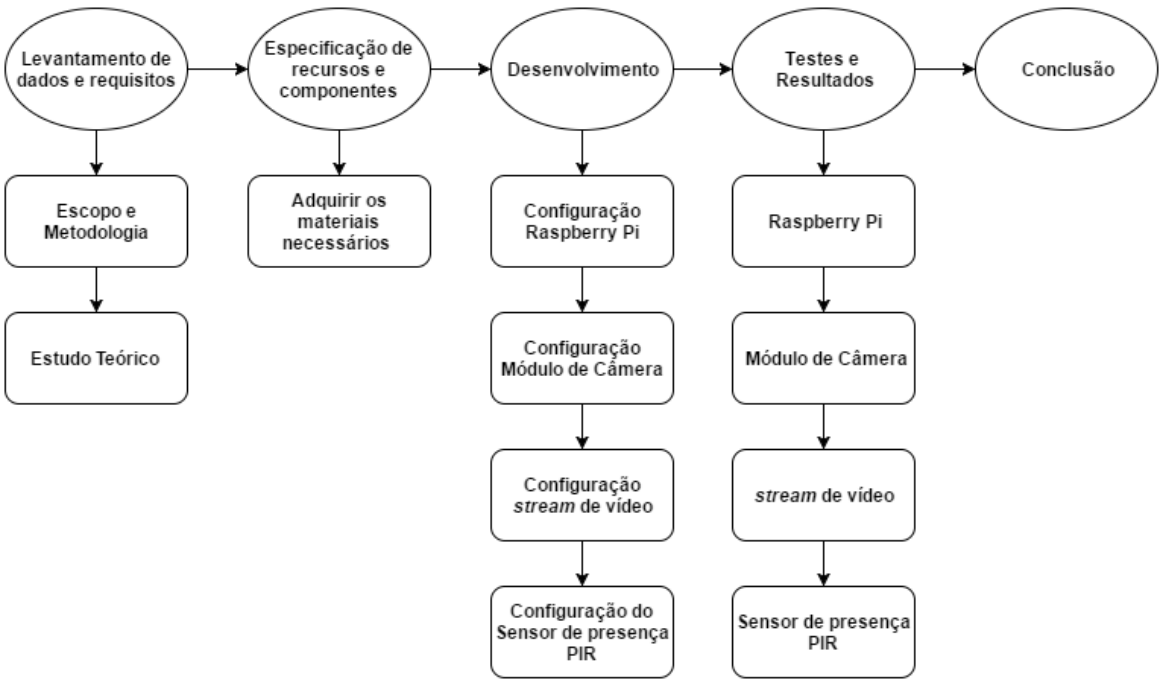

Figura 3.2 Etapas do Desenvolvimento

Fonte: Elaborada pelo autor

A Figura 3.2 explicita as etapas do desenvolvimento do projeto. A primeira etapa foi o levantamento de dados e requisitos, determinando o escopo e a metodologia, além da base teórica a serem utilizados ao longo do desenvolvimento do projeto de sistema de monitoramento residencial. A segunda etapa, especificação de recursos e componentes, trata sobre as ferramentas e materiais a serem adquiridos para a elaboração do sistema. Já a terceira etapa, desenvolvimento, é subdividia em quatro fases, sendo elas referentes às configurações dos materiais previamente levantados. A quarta etapa, testes, trata sobre a metodologia de verificação de funcionalidade do sistema como um todo e de cada um de seus módulos. Para o correto funcionamento do sistema proposto, é fundamental a utilização da ordem lógica de configuração e testes dos elementos expostos na etapas 3 e 4, uma vez que são dependentes um do outro em sua codificação. Por fim, a quinta etapa, conclusão, remete às considerações finais sobre o projeto de monitoramento residencial.

3.3 Configurações do Raspberry PI

A configuração do Raspberry PI se inicia com a configuração do cartão SD de pelo menos 16 gigas de armazenamento, que é o espaço minimo necessário para a instalação do sistema operacional Raspbian JESSIE, disponível no site do Raspberry PI e dos outros aplicativos necessários no sistema de monitoramento. Após o download da imagem do sistema operacional, é preciso instalar a imagem do sistema operacional no cartão SD e colocar o cartão na entrada do Raspberry PI, marcado com o número 1 (um) na Figura 3.3.

A plataforma Raspberry PI é conectada a um monitor através de um adaptador VGA/HDMI, marcado com o número 2 (dois) na Figura 3.3, e a um mouse e teclado através dos conectores USB, marcado com o número 3 (três) na Figura 3.3. Um cabo UTP (cabo de par trançado), marcado com o número 4 (quatro) na Figura 3.3, é utilizado para acessar a rede local e realizar atualizações e instalações de aplicativos pelo terminal. Uma fonte de alimentação de 110-240V/ 5V e 2,5A, marcado com o número 5 (cinco) na Figura 3.3 conecta o *Raspberry* a rede elétrica.

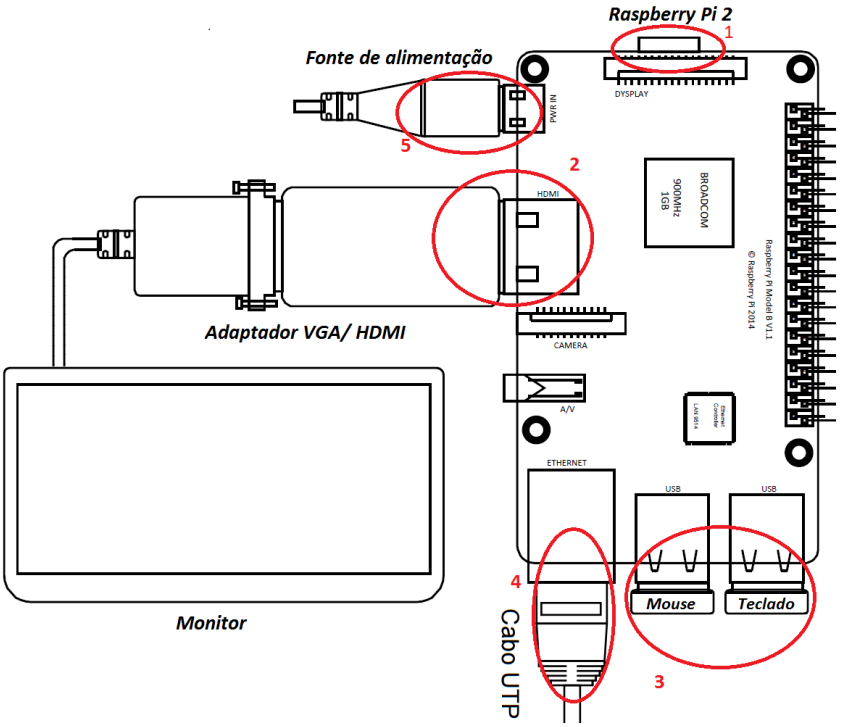

Figura 3.3 Conexões do Raspberry e periféricos

Fonte: Elaborado pelo autor

O sistema operacional precisa ser atualizado quando conectado pela primeira vez à rede Internet, utilizando os comandos no terminal, que se encontra no canto superior esquerdo da área de trabalho do Linux, como mostra a Figura 3.4:

- sudo apt-get install rpi-update
- sudo rpi-update
- sudo apt-get update
- sudo apt-get upgrade

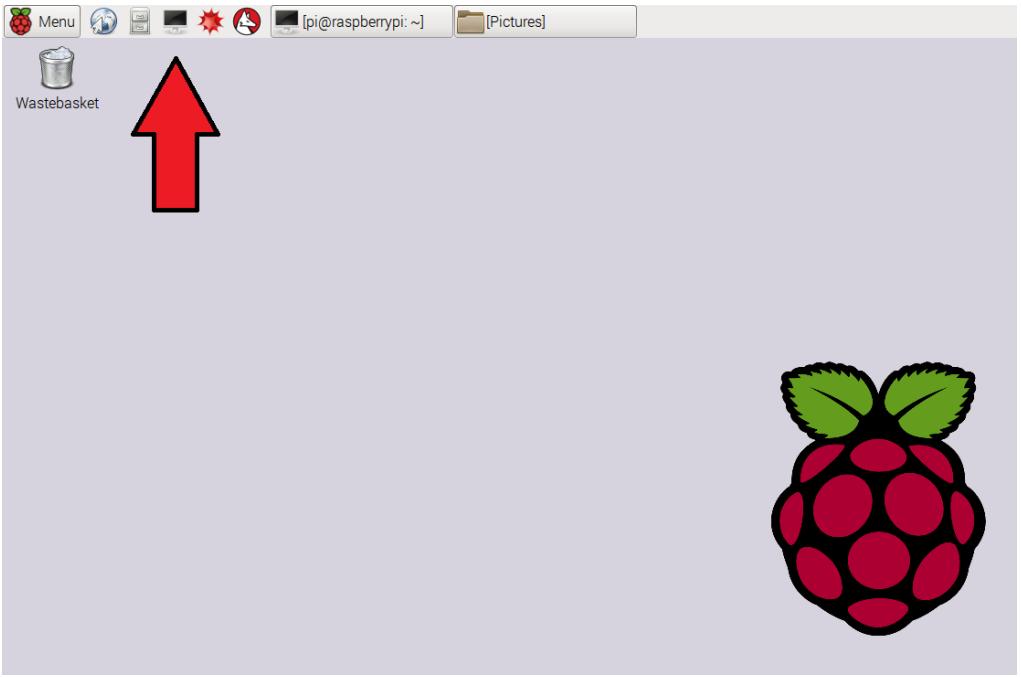

Figura 3.4 Área de trabalho do Linux com o ícone do terminal destacado

Fonte: Elaborada pelo autor

O processo de atualização pode levar alguns minutos. Quando esse processo de atualização terminar é necessário reiniciar o sistema com o comando:

- sudo reboot

A instalação do Raspberry PI disponibiliza do cartão SD somente o espaço necessário para a instalação do sistema operacional. É possível verificar o espaço do cartão SD utilizando o comando:

- df -h

Para ‘liberar’ o restante é necessário acessar o painel de configurações do Raspberry PI (Figura 3.5), para fazer a liberação do espaço restante, utilizando o comando:

- sudo raspi-config

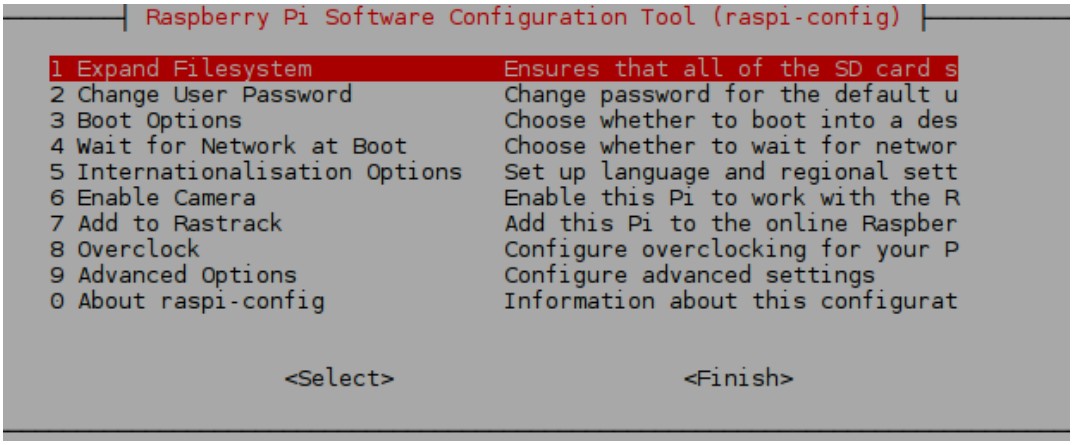

Figura 3.5 Painel de configuração - '1.Expand Filesystem'

Fonte: Elaborada pelo autor

No painel de configurações, utilizando as setas de direção, navega-se até a opção '*1.Expand Filesystem*', representado pela Figura 3.5, e pressionando a tecla ENTER. A Figura 3.6 indica que o a partição do Raspberry PI foi redimensionada para o tamanho total do cartão. O sistema irá ser reiniciado após o redimensionamento do cartão.

Figura 3.6 Mensagem de confirmação de redimensionamento

Fonte: Elaborada pelo autor

Com o cartão SD com o tamanho correto, é preciso agora executar o comando *sudo raspi-config* novamente e navegar para a opção '*9.Advanced Options'*, representado pela Figura 3.7, e pressionar a tecla ENTER. Nesse painel estão as configurações avançadas do Raspberry PI, Figura 3.8, navega-se até a opção '*A4 SSH*' e, pressionando ENTER, seleciona-se a opção *Enable* para habilitar o acesso por SSH, Figura 3.9. A habilitação do SSH é importante, pois propicia o acesso remoto às configurações do sistema, simplificando o acesso caso haja a necessidade de alguma manutenção do sistema de monitoramento.

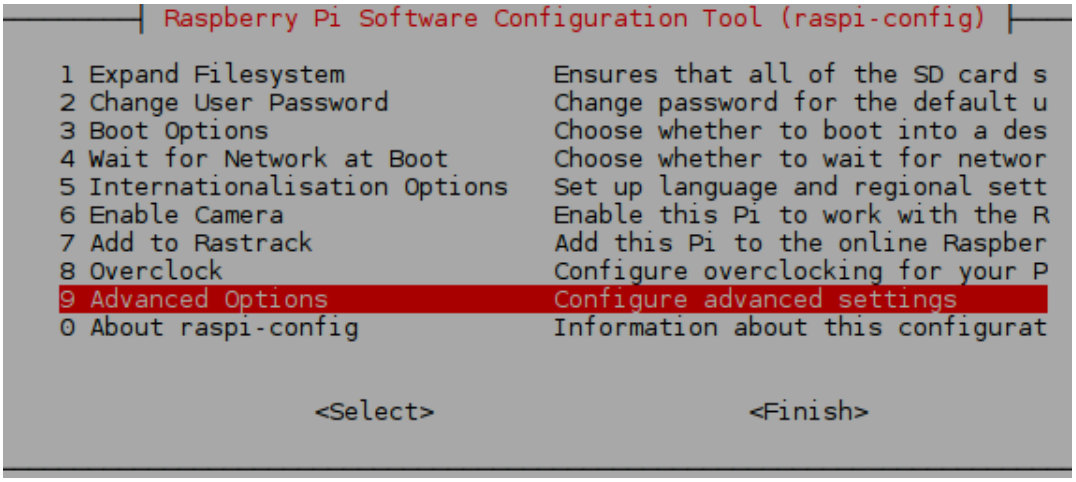

Figura 3.7 Painel de configuração - '9.Advanced Options'

Fonte: Elaborada pelo autor

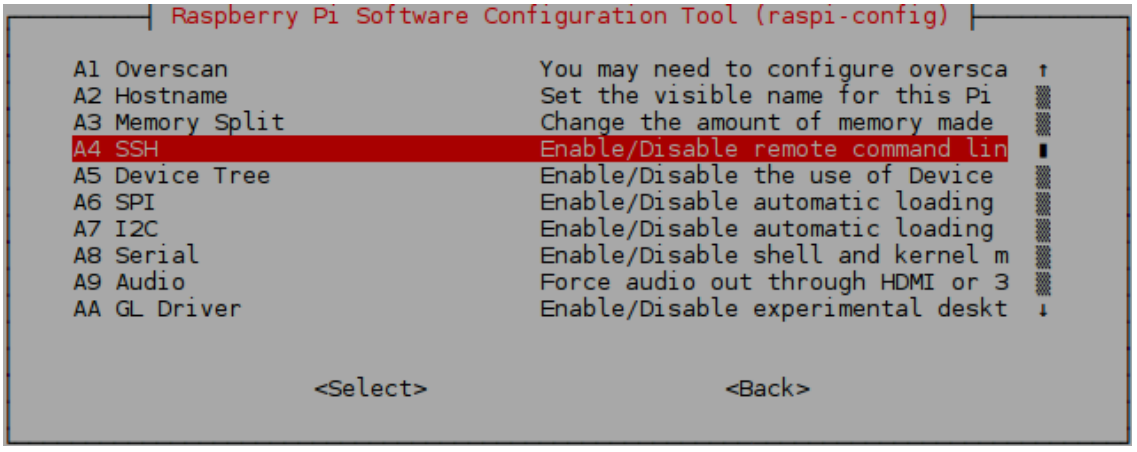

Figura 3.8 Painel de configuração - 'A4 SSH'

Fonte: Elaborada pelo autor

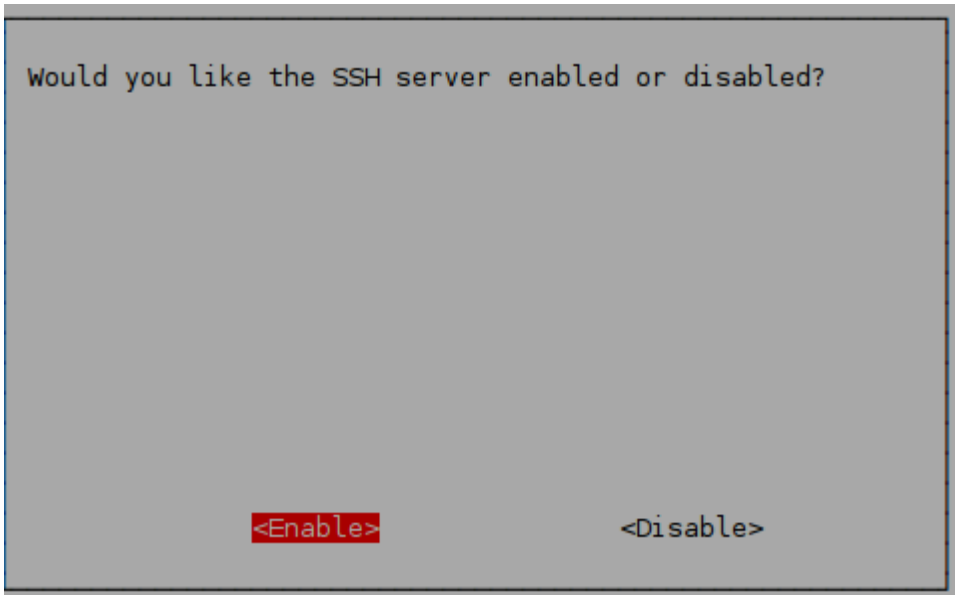

Figura 3.9 Painel de configuração SSH - confirmação

Fonte: Elaborada pelo autor

Isso finaliza os passos necessários para a configuração do Raspberry PI.

3.4 Configurações do Módulo de Câmera

Para a configuração do Módulo de Câmera é necessário inicializar o sistema e acessar o terminal de configuração. No terminal entrar com o comando *sudo raspi-config* e navegar até a opção '*6.Enable Camera*' (Figura 3.10). Nesse menu deve-se selecionar a opção *ENABLE* (Figura 3.11), e pressionar ENTER. A seguir, é preciso reiniciar o Rapsberry PI.

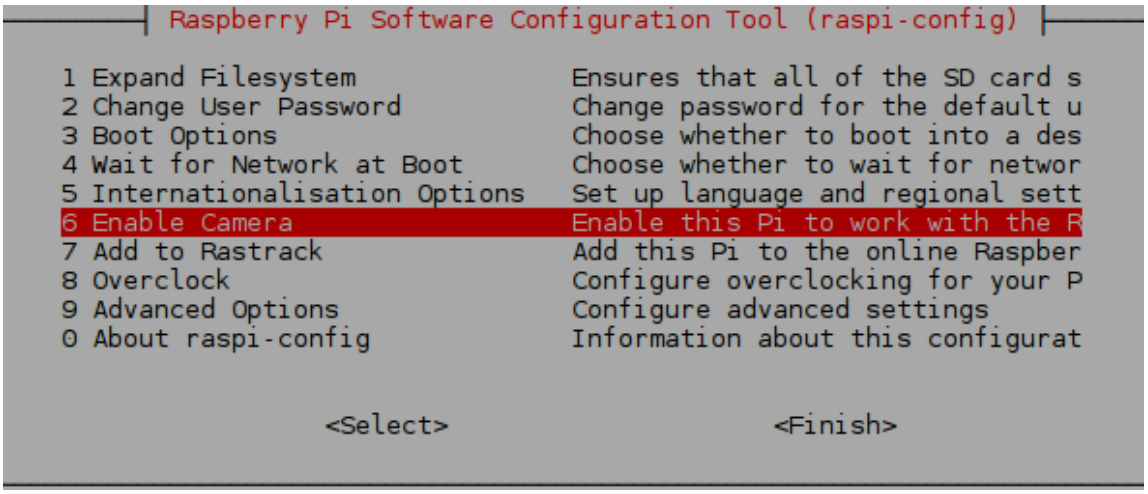

Figura 3.10 Painel de configuração - '6.Enable Câmera'

Fonte: Elaborada pelo autor

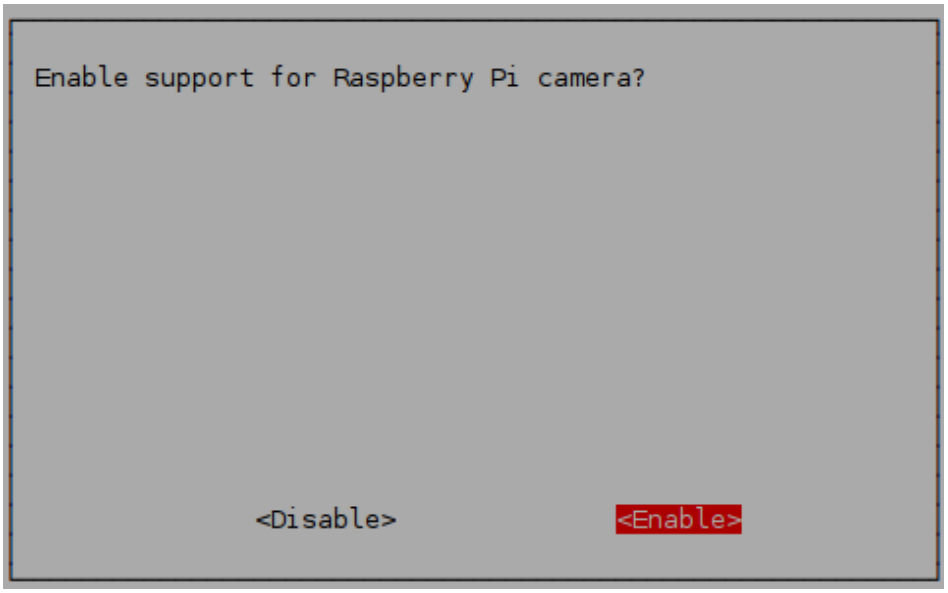

Figura 3.11 Painel de configuração da câmera

Fonte: Elaborada pelo autor

Nesse momento é necessário configurar o Raspberry para o funcionamento correto do *streaming* do vídeo. Para tanto, deve-se acessar novamente o terminal de comandos e entrar com o comando:

- sudo nano /etc/network/interfaces

Esse comando irá abrir um arquivo direto pelo console do Linux, onde é preciso adicionar as linhas de código descritas no quadro abaixo. Caso essas novas linhas não sejam inseridas, o IP do Raspberry PI fica automático, ou seja, sempre que ele for iniciado, um novo IP poderá ser atribuído ao Raspberry PI, dificultando a configuração do *streaming*. A Figura 3.12 ilustra como deverá ser o arquivo após a modificação.

- auto lo
- iface lo inet loopback
- iface etho0 inet static
- address 192.168.1.101
- netmask 255.255.255.0
- network 192.168.1.0
- broadcast 192.168.1.255
- gateway 192.168.1.1

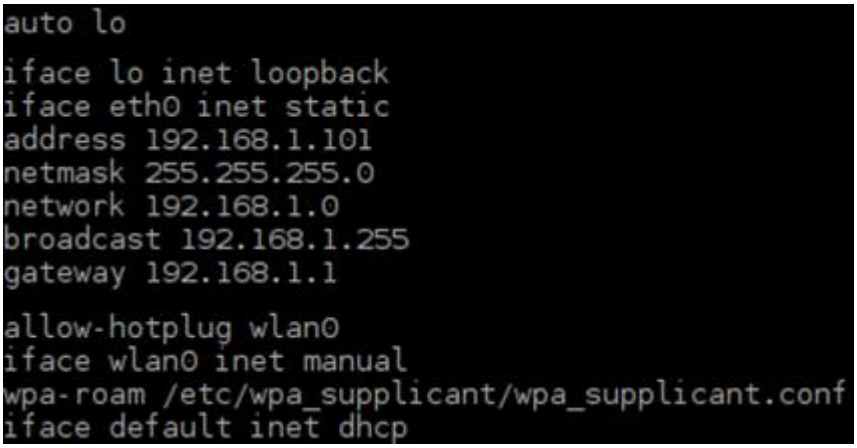

Figura 3.12 Arquivo Interfaces no console do Linux

Fonte: Elaborada pelo autor

Onde está escrito "Nome da Rede Wifi" trocar pelo nome da rede wifi a que o Raspberry PI irá se conectar. No caso deste projeto, a wifi a ser conectada é a ‘Andoid AP’, e onde está "Senha da Wifi", entrar com a senha da rede Wifi. Fechar, então, o arquivo apertando as teclas CTRL+X, e salvar o arquivo apertando a tecla Y e finalmente a tecla ENTER para confirmar as modificações. Agora é necessário reiniciar o aparelho, para que as alterações tenham efeito, para reiniciar o Raspberry via console entrar com o comando:

- sudo reboot

3.5 Configuração do *stream* vídeo

O stream do vídeo é feito utilizando uma biblioteca open source chamada RPi_cam_web_interface, essa biblioteca disponibiliza uma interface web para o módulo da câmera, que pode ser aberto em qualquer browser. Para instalar a biblioteca, primeiro é necessário instalar o sistema de controle de versionamento Git, com o comando:

- sudo apt-get install git

A instalação do Git disponibiliza acesso a uma variedade de comandos de controle de versionamento, como por exemplo: ‘add’, adiciona um conteúdo à pasta do Git; ‘checkout’ finaliza uma arvore de trabalho; ‘commit’, grava as modificações em um arquivo na árvore de trabalho; ‘clone’ duplica um repositório em um novo diretório. Este comando será utilizado para copiar os arquivos da biblioteca do Git para o Raspberry PI.

- git clone https://github.com/silvanmelchior/RPi_Cam_Web_Interface.git

Tendo a biblioteca duplicada para o diretório do Raspberry navegar até a pasta do repositório, com o comando:

- cd RPi_Cam_Web_Interface

Dentro do repositório existem 5 (cinco) arquivos *shell scripts*: *install.sh*, *update.sh*, *start.sh* e *stop.sh*. Os arquivos são providenciados para a instalação e manutenção da biblioteca, esses arquivos precisam ter a permissão para ser executados via console, essa permissão é feita com a execução do comando:

- chmod u+x *.sh

O arquivo *install.sh* agora pode ser executado, possibilitando o uso da biblioteca para o sistema de monitoramento. Para instalar a biblioteca utilizar o comando:

- ./install.sh

Por fim, reiniciar o Raspberry para que as modificações tomem efeito, utilizando o comando:

- sudo reboot

3.6 Configuração do sensor de presença PIR

O sensor PIR possui 3 (três) pinos conectores, o VCC, OUT e GND, indicados na Figura 3.13, respectivamente pelos cabos vermelho, laranja e preto. Estes precisam ser conectados ao Raspberry PI nos pinos: 2 (dois), de 5V, 4 (quatro), o GP4, e 6 (seis), GND, na ordem apresentada anteriormente.

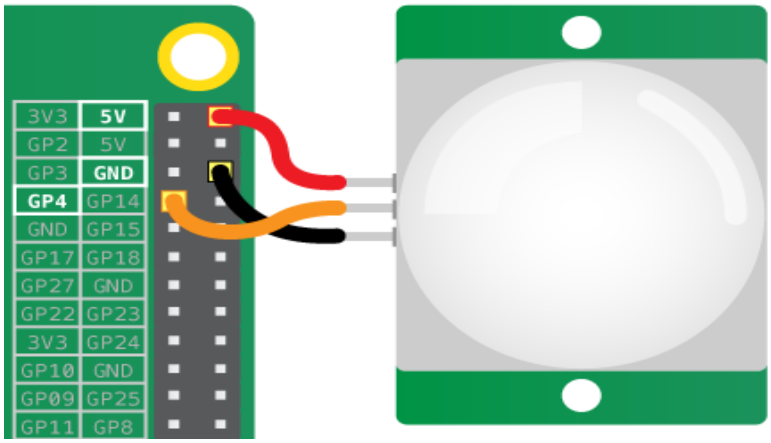

Figura 3.13 Sensor PIR conectado no Raspberry PI

Fonte: Adaptado de <https://www.raspberrypi.org/learning/parent-detector/worksheet/>.

Tendo feito as devidas conexões é preciso desenvolver um código em Python, apresentado no APÊNDICE A, que irá interagir com as portas do Raspberry PI e interpretar os sinais que o sensor estará emitindo. É necessário salvar o script com o nome 'motion_pir' uma vez que esse será o comando chamado no *boot* do Raspberry Pi. Neste projeto o código utilizado cria variáveis que irão receber o e-mail do emissor (e-mail do sistema: *gmail_user*, *gmail_pwd*) e do receptor (e-mail do cliente: *to*) e estrutura o assunto (*subject*) e o corpo do e-mail (*text*), contendo o endereço de IP do servidor do *stream* do vídeo (*ip.getsockname()[0]*). Além disso, o código seta os estados: atual (*current_state*) e anterior (*previous_state*) para falso e monta um laço (*while)* sempre verdadeiro, que recebe o estado atual do sensor (*GPIO.input(sensor)*) e verifica se o estado anterior sofreu modificação.

Quando o estado atual (current_state) for diferente do estado anterior (*previous_state*) o sistema irá indicar no terminal do Linux para qual estado ele foi (*LOW* ou *HIGH*) e será

apresentada a mensagem 1 (um). Caso o novo estado seja *HIGH* então o sistema apresentará a mensagem 2 (dois), indicando que foi reconhecida alguma movimentação no ambiente monitorado, neste momento, o sistema passará por mais três etapas: a cópia da imagem estática do *stream* (mensagem 3), o ato do envio do e-mail para o receptor (cliente) (mensagem 4) e, finalmente, a confirmação que o e-mail foi enviado (mensagem 5). As mensagens anteriormente citadas estão indicadas na Figura 3.14.

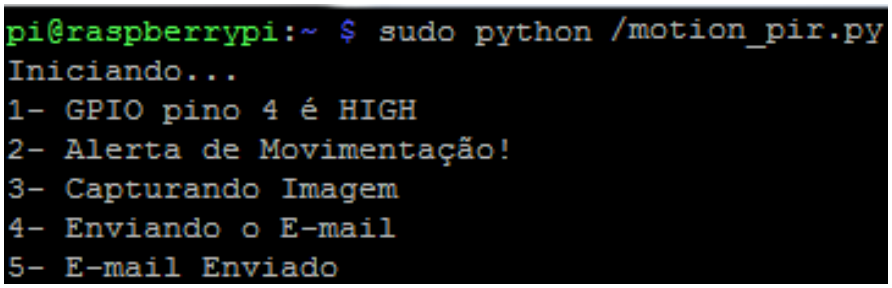

Figura 3.14 Mensagens do código do sensor de presença PIR

Fonte: Elaborado pelo Autor

O arquivo com o código do sensor PIR precisa ser adicionado nos programas iniciados pelo *boot* do Raspberry PI, portanto é preciso abrir o arquivo ".bashrc" localizado na raiz do sistema operacional do Linux. Nesse arquivo existem diversos comandos que são executados no momento em que o usuário inicia o sistema operacional, então é preciso adicionar também a chamada do script com o código do sensor PIR, para efetuar a chamada do código basta adicionar './motion_pir' ao final do script. A seguir estão as linhas de código necessárias para acessar o arquivo ".bashrc":

- cd ~
- sudo nano .bashrc

Isso finaliza os passos necessários para a configuração do sensor de presença PIR.

Este capítulo mostrou as etapas necessárias para o desenvolvimento do projeto, como a configuração necessária para o funcionamento do Raspberry PI, do módulo de câmera, do

stream de vídeo e finalmente do sensor de presença PIR. Ao fim do desenvolvimento, foi obtido o sistema de monitoramento, esquematizado na Figura 3.15.

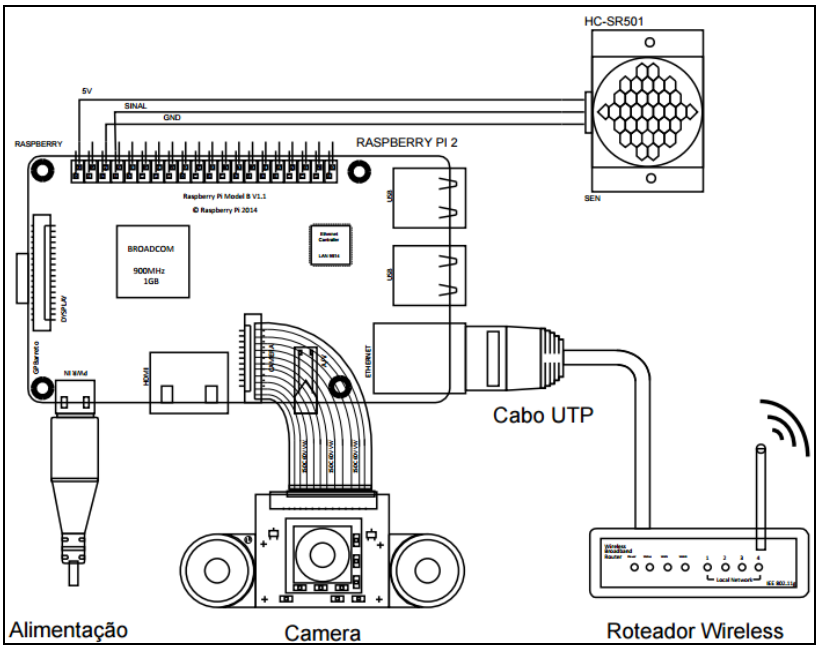

Figura 3.15 Esquemático do protótipo

Fonte: Elaborado pelo Autor

A Figura 3.15 mostra como os dispositivos e periféricos são conectados ao Raspberry PI, a tabela 3.1 mostra os preços de cada um dos periféricos utilizados nesse projeto e também seu total, em reais.

Tabela 3.1 Custo por periférico.

Item	Preço
RASPBERRY PI	R$ 122,50
Sensor de Presença PIR	R$ 14,46
RaspiCam Infravermelho	R$ 65,80
Roteador Wireless	R$ 57,00
TOTAL:	R$ 259,76

Fonte: Elaborado pelo Autor

O próximo capítulo abordará os testes e cenários a que o sistema foi submetido, buscando mostrar o seu funcionamento e os resultados.

CAPÍTULO 4 - TESTES E RESULTADOS

Este capítulo apresenta os testes realizados para garantir o funcionamento do sistema, e os resultados obtidos, sendo que, para tanto, foram estruturados cinco tópicos. O primeiro trata sobre os testes realizados para conferência das configurações do dispositivo Raspberry PI. O segundo, sobre os testes realizados para garantir que o módulo de câmera tenha sido ativado com sucesso e que está gerando as imagens corretamente. O terceiro tópico é relativo ao *stream* de vídeo, de modo a verificar se a biblioteca foi instalada com sucesso. Já o quarto é referente aos testes com o sensor de presença e o envio da notificação ao cliente. Finalmente, o quinto tópico trata dos testes e resultados finais do protótipo do sistema de monitoramento residencial.

4.1 Testes da configuração do Raspberry PI

Os testes relativos à configuração do Raspberry PI foram iniciados com a instalação do sistema operacional adotado no projeto na plataforma Raspberry PI. A Figura 4.1 mostra a área de trabalho do Raspberry PI após sua devida instalação. É possível verificar com a imagem que até o momento não houve erros na inicialização do sistema operacional Raspbian-Jessie, pois quando há erros durante a instalação do SO ou em sua inicialização, a área de trabalho do Linux não é inicializada.

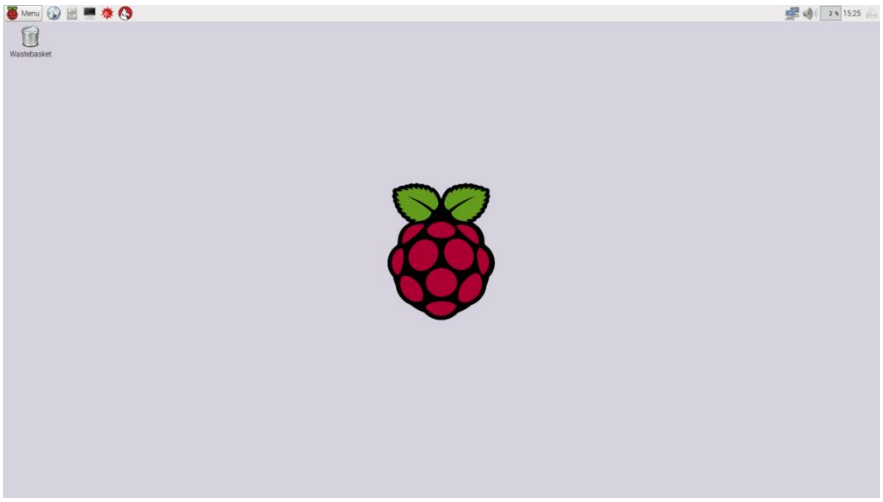

Figura 4.1 Área de trabalho do Raspberry PI com o SO Raspbian-Jessie

Fonte: Elaborado pelo Autor

Uma vez que o espaço liberado pela instalação do SO é uma pequena fração do espaço total do cartão de memória, a biblioteca do módulo de câmera não consegue ser instalada, muito menos gerar todas as imagens para o manuseio do sistema de monitoramento visado neste projeto. Portanto é preciso que seja liberado o restante do espaço total do cartão SD. Essa liberação ocorreu conforme descrito no capítulo 3. Com a Figura 4.2 é possível averiguar que o cartão de memória está com seu espaço máximo disponível para uso e não somente o espaço mínimo liberado para a instalação do sistema operacional Raspbian-Jessie.

```
pi@raspberrypi:~ $ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root        15G  6.6G  7.3G  48% /
devtmpfs        427M     0  427M   0% /dev
tmpfs           432M   16K  432M   1% /dev/shm
tmpfs           432M  6.0M  426M   2% /run
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           432M     0  432M   0% /sys/fs/cgroup
/dev/mmcblk0p1   60M   20M   41M  34% /boot
tmpfs            87M     0   87M   0% /run/user/33
tmpfs            87M     0   87M   0% /run/user/1000
```

Figura 4.2 Mensagem do cartão de memória redimensionado

Fonte: Elaborado pelo Autor

4.2 Testes da configuração do módulo de câmera

Na realização dos testes de configuração do módulo de câmera foram utilizados os comandos, mostrados no quadro abaixo, diretamente no console do Linux. Eles serviram para a geração de uma imagem e um vídeo, respectivamente. A Figura 4.3 e a Figura 4.4 mostram que o Raspberry PI está reconhecendo os comandos da Raspicam, pois gerou a imagem e o vídeo com êxito.

- sudo raspistill -vf -hf -o teste_imagem.jpg
- sudo raspivid –vf –hf -o teste_video.h264 -t 10000

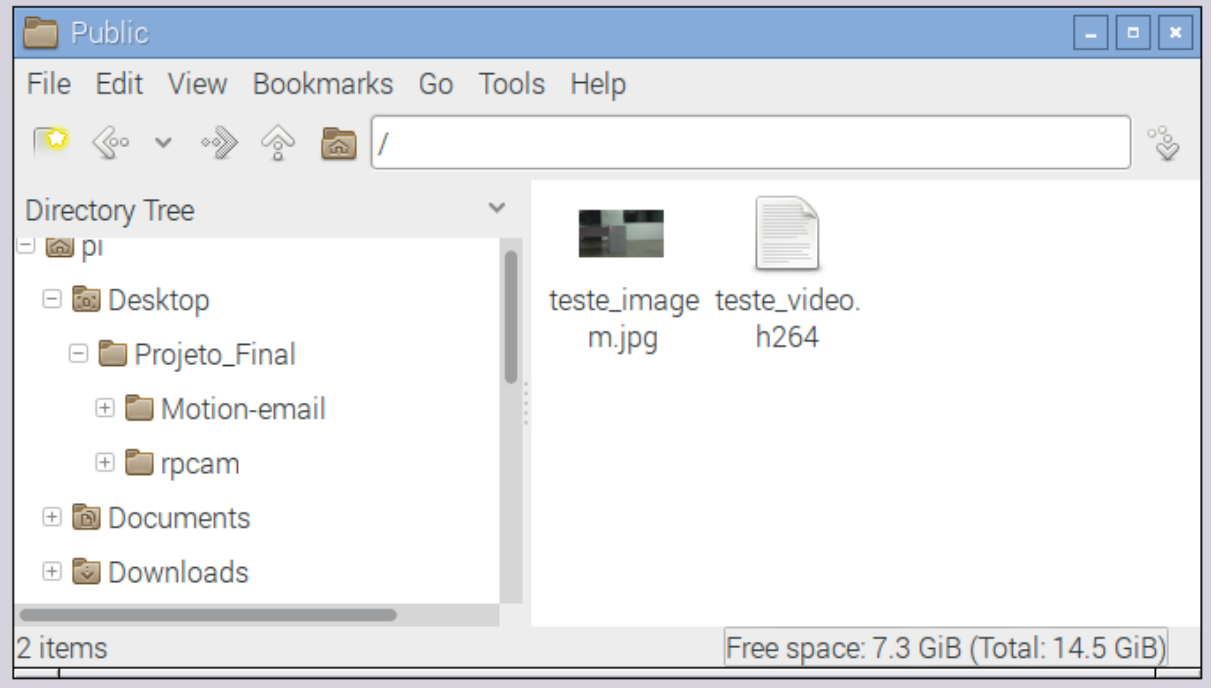

Figura 4.3 Arquivos de vídeo e imagem

Fonte: Elaborado pelo Autor

Figura 4.4 Imagem tirada em teste com módulo de câmera

Fonte: Elaborado pelo Autor

Entre os testes desta etapa foi encontrada uma dificuldade para posicionar a fita serial conectora. Apesar de a fita serial poder ser encaixada de duas formas, somente uma leva ao funcionamento do módulo de câmera. O posicionamento correto é aquele em que os conectores prateados ficam virados para frente do Raspberry PI, como indicado na documentação do site do Raspberry PI (RASPBERRY FOUNDATION, 2015).

4.3 Testes do *stream* de vídeo

A fase de testes de *stream* de vídeo consistiu, primeiramente, na tentativa de acesso ao IP do Raspberry PI na porta 8080, criada pela biblioteca do Raspberry. Como este acesso foi realizado com sucesso (Figura 4.5), o passo seguinte foi a visualização do streaming na pasta RPi_cam localizada na porta 8080 (Figura 4.6).

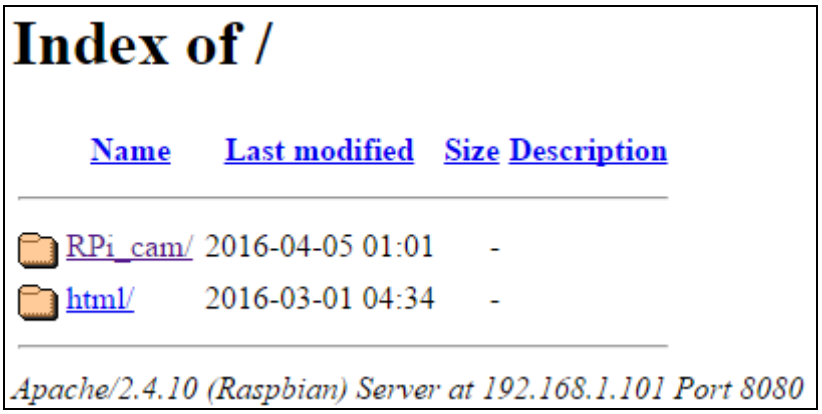

Figura 4.5 Acesso ao IP do Raspeberry PI

Fonte: Elaborado pelo Autor

Figura 4.6 Acesso à página Web

Fonte: Elaborado pelo Autor

4.4 Testes do sensor de presença PIR

Para a realização de testes do sensor de presença PIR, inicialmente é preciso verificar se todos os fios foram conectados corretamente às portas definidas no capítulo 3 (três) e, então, executar o script do sensor PIR conforme o código a seguir:

- sudo python Desktop/Projeto_Final/motion_pir.py

Em seguida, deve ser verificado no console se as mensagens de log, previamente mencionadas (Figura 3.14), foram geradas. Caso contrário, será necessário realizar a verificação tanto das conexões feitas entre o Raspberry PI e o sensor de presença PIR, quanto no script com o algoritmo do sensor de presença PIR.

Para concluir esta etapa de testes, é preciso confirmar que o sistema de envio de e-mail de notificações está funcionando adequadamente, ou seja, é preciso entrar no endereço de e-mail do cliente e verificar se existe uma mensagem enviada pelo sistema de monitoramento residencial (Figura 4.7).

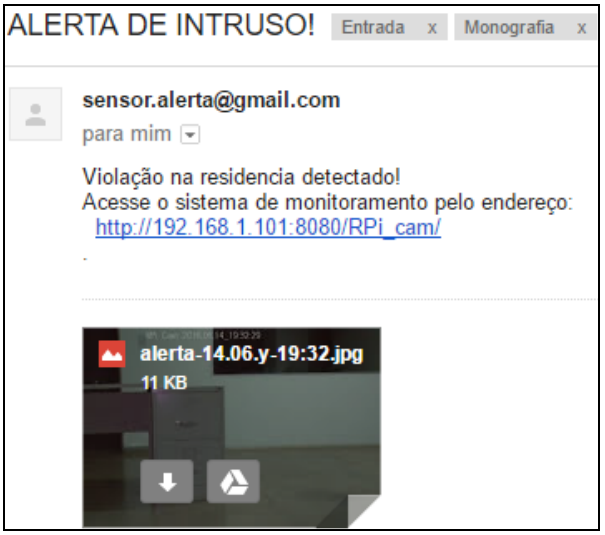

Figura 4.7 E-mail do sistema no cliente

Fonte: Elaborado pelo Autor

Durante a fase de testes do script do sensor de presença PIR, foi detectado que o envio da notificação por e-mail estava com a frequência de duas mensagens por segundo, ou seja, uma frequência desnecessariamente alta. Para solucionar este problema foi adicionada uma linha no código que força o sistema a esperar pelo menos 10 (dez) segundos para executar outra validação do estado do sensor de presença PIR, como mostrado no APÊNDICE A. Assim o envio de e-mails passou a ter uma frequência mais coerente com as necessidades do projeto.

4.5 Testes e resultados finais do funcionamento do protótipo

Depois de realizados os testes específicos citados nos tópicos anteriores, foram feitos os testes do sistema como um conjunto. O primeiro teste realizado para garantir o funcionamento do sistema foi um teste de estresse da disponibilidade de streaming. Para tanto, o sistema foi colocado em trabalho constante durante 48 horas, tempo considerado suficiente pelo autor desta pesquisa para os fins do projeto. Ao fim deste intervalo, o sistema manteve sua capacidade de monitoramento aparentemente estável, sendo assim considerado funcional.

No decorrer das 48 horas, também foram realizados testes de detecção de presença pelo sensor PIR. No total, foram executadas 104 amostras da capacidade de detecção (total de amostras), seguidas de envio do e-mail de notificação com imagem da área monitorada em anexo.

Nas amostras foram encontrados 12 erros. Dentre eles, 4 (quatro) foram falhas no envio do e-mail (falha e-mail), 3 (três) foram envios do e-mail de notificação de movimentação sem a imagem em anexo (falha anexo) e 5 (cinco) foram testes em que não foi detectado movimentação alguma (falha movimentação). Portanto o sistema mostra um erro de 11.53% (%Erro), calculado com a seguinte fórmula:

$$\%Erro = \frac{(falha\ email + falha\ anexo + falha\ movimentação)}{total\ de\ amostras} * 100$$

Após o teste de estresse, foi efetuada uma bateria de testes com 32 amostras para garantir que há de fato uma diminuição do tempo de resposta em caso de violação do perímetro residencial monitorado. Conforme o teorema central do limite são necessárias, no mínimo, 31+1 amostras para considerar um espaço amostral válido para o levantamento estatístico de um experimento dessa natureza. Essas amostras foram coletadas da seguinte forma:

a) Uma pessoa entra em um cômodo monitorado, simulando a invasão domiciliar, e anota o horário em que entrou.
b) Uma segunda pessoa irá receber a notificação via e-mail, e deverá anotar o horário em que leu a notificação.

Observa-se que ambos os relógios devem estar marcando o mesmo horário, para que a análise possua uma precisão mínima.

Após todas as amostras coletadas foi feita a média de tempo decorrido entre o horário da simulação de invasão e os horários da ciência da notificação, obtendo uma média de 17 minutos.

Em seguida foram coletadas mais 32 amostras, dessa vez sem a utilização do sistema de monitoramento residencial. O tempo médio obtido foi de 08 horas e 29 minutos.

O tempo de tomada de conhecimento é ligado ao tempo de possibilidade de resposta do cliente, uma vez que é a partir dele que uma ação, como avisar o serviço de segurança, pode ser tomada. Assim, com o experimento feito, é possível perceber que houve uma diminuição no tempo de ciência da violação residencial. Também, no decorrer do experimento não foi detectado nenhum falso positivo, ou seja, todos os alertas de movimentação que foram enviados ao e-mail do cliente foram resultados de uma movimentação no local monitorado.

A Figura 4.8 mostra uma vista superior do protótipo, ou seja, o Raspberry PI e o cabo de força (1), o jumper e o sensor de presença (2), o cabo de UTP (3) e o módulo da câmera com sua fita serial conectora (4).

Figura 4.8 Vista superior do protótipo

Fonte: Elaborado pelo Autor

Na Figura 4.9, numa visão frontal do protótipo, é possível ver os focos de infravermelho e a câmera (1) e o sensor de presença PIR (2).

Figura 4.9 Vista frontal do protótipo

Fonte: Elaborado pelo Autor

A Figura 4.10 mostra o sistema de monitoramento em sua fase final. Os equipamentos foram acomodados em uma caixa com a finalidade de posicionar o sensor e a câmera adequadamente, além de manter o conjunto em ordem.

Figura 4.10 Protótipo montado

Fonte: Elaborado pelo Autor

Em decorrência dessa acomodação, foi percebido um aumento na temperatura do módulo de monitoramento, a temperatura máxima encontrada foi de 62ºC. Apesar disso, tendo em consideração as especificações de fábrica dos materiais utilizados, Raspberry Pi (70ºC), sensor de presença PIR (70ºC) e do módulo de câmera (70ºC), o sistema se manteve em uma faixa de temperatura aceitável, não prejudicando o funcionamento dos equipamentos.

Com os resultados dos testes foram gerados dois gráficos que serão explicados a seguir.

O primeiro gráfico, Figura 4.11, ilustra a diferença de tempo, em horas, entre a percepção de movimentação indevida pelo sistema de monitoramento e a notificação do usuário sobre tal situação. A linha indicada como “Sem sistema de segurança” mostra valores de tempo substancialmente elevados, variando entre 14 horas e 29 minutos e 1 hora e 9 minutos, em relação aos valores encontrados na linha “Com sistema de segurança”, que varia entre 3 minutos e 1 hora e 45 minutos.

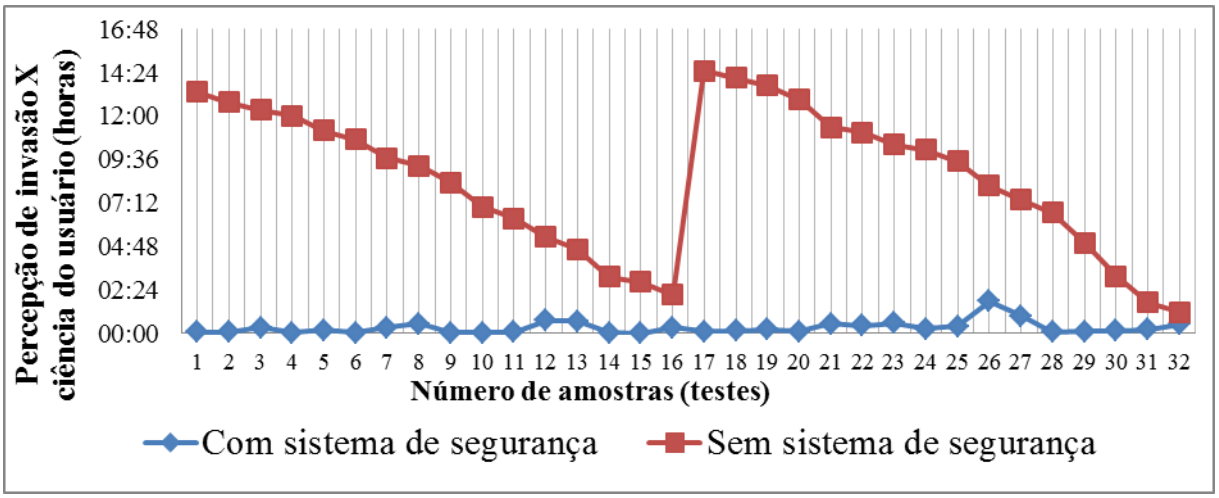

Figura 4.11 Gráfico de tempo de ciência

Fonte: Elaborada pelo autor

O segundo gráfico, Figura 4.12, ilustra os resultados dos testes de estresse, o sistema obteve uma faixa de sucesso de 88% e uma faixa de erros de 12%. No entanto, com os testes feitos para a diminuição do tempo de ciência, foi possível verificar que essa margem de erro não levou a falha no funcionamento do sistema, pois, como descrito no algoritmo do apêndice A, o sistema realiza uma verificação a cada dez segundos, minimizando a chance de uma invasão não ser notificada ao cliente.

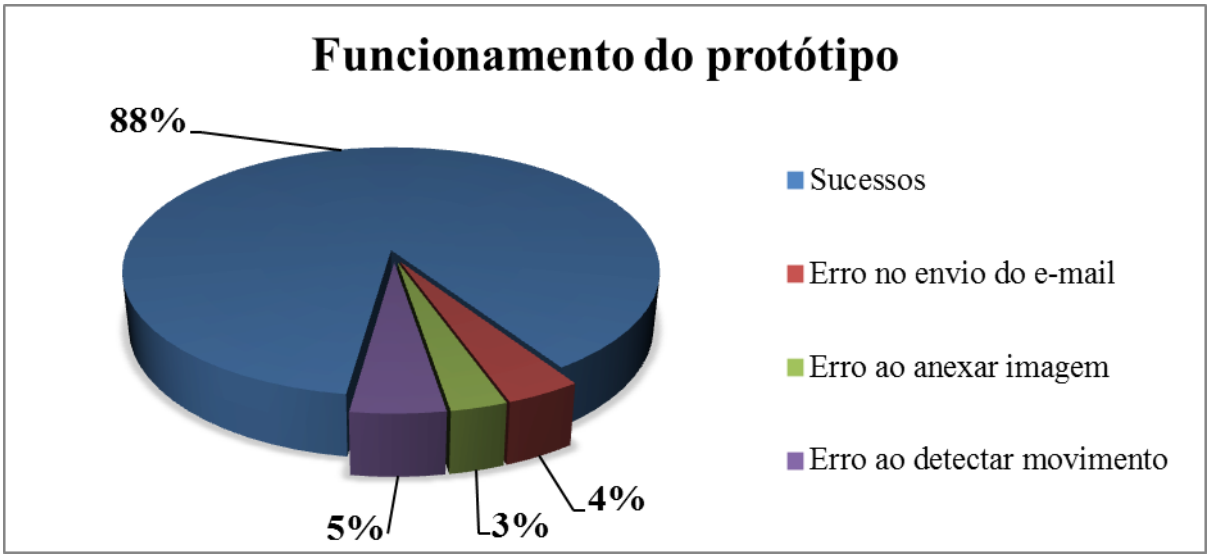

Figura 4.12 Gráfico funcionamento do protótipo

Fonte: Elaborada pelo autor

Após a finalização do projeto com a realização dos testes de funcionamento do protótipo, foi possível chegar às seguintes respostas aos objetivos propostos:

a) O sistema desenvolvido se mostrou capaz de reduzir o tempo de ciência do usuário a uma invasão domiciliar, através da utilização de notificação via e-mail.
b) O desenvolvimento de código em linguagem Python foi realizado, de modo que o módulo de monitoramento se mostrou eficaz para realizar as atividades esperadas.
c) O envio de notificação via e-mail ao usuário foi implementado e possui baixa taxa de erro.
d) Foi aferida a possibilidade de disponibilização de uma página web com streaming da imagem da câmera, utilizada para o sistema de monitoramento.

Assim, o desenvolvimento do sistema de monitoramento residencial planejado para este projeto é possível e foi realizado com sucesso, aferido através da realização dos testes apresentados.

CAPÍTULO 5 – CONCLUSÃO

Após a realização deste projeto, foi aferido que o sistema de monitoramento desenvolvido com a utilização do Raspberry PI, a Raspicam e o sensor PIR se mostrou muito eficaz, pois leva rapidamente ao conhecimento do usuário a ocorrência de invasão ao local monitorado.

Tendo em vista este fato, recomenda-se que novos estudos sejam realizados para a melhora e evolução do sistema de monitoramento desenvolvido nesse projeto, dentre eles:

- Procurar diminuir a taxa de falha do sistema, para aumentar sua confiabilidade.
- Disponibilizar a página WEB para acesso via internet e não só uma rede local
- Diminuir a temperatura do protótipo, com pesquisa dos melhores materiais para manter o sistema em funcionamento e em segurança.
- Ligar o sistema a uma rede de segurança local, para que a força policial seja acionada com ainda mais rapidez.

REFERÊNCIAS

BAPTISTA, L. F. Escola Superior Náutica. **enautica**, 2013. Disponivel em: <http://www.enautica.pt/publico/professores/baptista/instrum/Ap_IC_cap8.pdf>. Acesso em: 14 Junho 2016.

DEITEL, P.; HARVEY, D. **Java como programar**. 8ª. ed. Sao Paulo: Pearson, 2010. 5,6 p.

FACCIN, F. **Abordagem Inovadora no Projeto de Controladores PID - Dissertação de Mestrado**. Universidade Federal do Rio Grande do Sul. Porto Alegre, p. 2. 2004.

LUIZ, G. G1 Globo. **Site de noticias do Globo**, 01 jan. 2016. ISSN http://g1.globo.com/distrito-federal/noticia/2016/01/df-tem-aumento-de-roubos-em-casas-e-coletivos-em-2015-homicidios-caem.html. Acesso em: 7 Março 2016.

OGATA, K. **Engenharia de controle moderno**. 4ª. ed. Rio de Janeiro: Pearson Prentice Hall, 2003. 1,4-6, 170-173 p.

PHP. Site oficial do PHP. **Site do PHP**, 2015. Disponivel em: <http://php.net/manual/pt_BR/history.php.php>. Acesso em: 10 Abril 2016.

PYTHON. Getting started with python. **Python**, 2015. Disponivel em: <http://thepythonguru.com/getting-started-with-python/>. Acesso em: 2016 Maio 2016.

RASPBERRY FOUNDATION. RASPBERRY PI 2 MODEL B. **Raspberry PI**, 2015. Disponivel em: <https://www.raspberrypi.org/>. Acesso em: 11 Maio 2016.

ROBERTS, M. J. **Fundamentos de Sinais e Sistemas**. [S.l.]: [s.n.], 2009.

APÊNDICE A – CÓDIGO DO SENSOR DE PRESENÇA.

O código a seguir é responsável pelo controle do sensor de presença. Ele é iniciado com a importação das bibliotecas que serão utilizadas no decorrer do algoritmo. Em seguida, são carregadas as variáveis com as informações dos e-mails do sistema de segurança e do cliente e, por fim, um laço de repetição é utilizado para verificação do estado atual do sensor e comparação com o estado anterior, de forma a disparar a notificação via e-mail caso o estado atual seja *HIGH*.

```
# -*- coding: utf-8 -*-
import RPi.GPIO as GPIO
import time
import numpy as np
from datetime import datetime
import os
import smtplib
from email.MIMEMultipart import MIMEMultipart
from email.MIMEBase import MIMEBase
from email.MIMEText import MIMEText
from email import Encoders
import socket
ip = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
ip.connect(("gmail.com",80))

gmail_user = "sensor.alerta@gmail.com"#E-mail do sistema de seguraça
```

```
gmail_pwd = "********"#Senha do E-mail do sistema de segurança
to = "paz.matheus@gmail.com"#E-mail do cliente
subject = "ALERTA DE INTRUSO!"#Assunto do E-mail
text = "Violação na residencia detectado!\n"
        +"Acesse o sistema de monitoramento pelo."
        +"endereço: \n  http://"+ip.getsockname()[0]+":"
        +"8080/RPi_cam/ \n."# Corpo do E-mail a ser enviado


sensor = 4
GPIO.setmode(GPIO.BCM)
GPIO.setup(sensor, GPIO.IN, GPIO.PUD_DOWN)


previous_state = False # Estado inicial
current_state = False # Estado inicial
print "Iniciando..." # Log de inicio
print("1- GPIO pino %s é %s" % (sensor, current_state)) # Log da variavel inicial
while True: # Laço de verificação
   time.sleep(0.1)
   previous_state = current_state
   current_state = GPIO.input(sensor)


   if current_state != previous_state: # Comparação com o estado anterior
      new_state = "HIGH" if current_state
                                    else "LOW" # Compara os dois estados, para escrever
HIGH ou LOW
```

```
            print("1- GPIO pino %s é %s" % (sensor, new_state)) # Escreve a mensagem com o estado
                if current_state: # Compara se o estado é HIGH
                        print "2- Alerta de Movimentação!"
                        print "3- Capturando Imagem"
                        time.sleep(1)

                        picname = datetime.now().strftime("%d.%m.%y-%H:%M") # Nome do anexo
                        picname = '/tmp/alerta-'+picname+'.jpg' # Endereço de copia para o anexo
                        os.system("sudo cp /dev/shm/mjpeg/cam.jpg "+picname+"") # Copia a imagem para ser anexada
                        attach = picname # Adiciona o caminho da figura para ser enviado
                        msg = MIMEMultipart() # Cria array para os endereços de email
                        msg['From'] = gmail_user # E-mail do sistema
                        msg['To'] = to # E-mail do cliente
                        msg['Subject'] = subject # Assunto do E-mail

                        print "4- Enviando o E-mail" # Inicia o processo de Email

                        msg.attach(MIMEText(text))

                        part = MIMEBase('application', 'octet-stream')
        part.set_payload(open(attach, 'rb').read())
                        Encoders.encode_base64(part)
                        part.add_header('Content-Disposition',
```

```
'attachment; filename="%s"' % os.path.basename(attach))
msg.attach(part)

mailServer = smtplib.SMTP('smtp.gmail.com:587') # inicia o processo de SMTP
mailServer.ehlo() # Identifica ao servidor o Protocolo de SMTP extendido
mailServer.starttls() # Inicia uma camada de transporte segura
mailServer.login(gmail_user, gmail_pwd) # Loga no serviço de e-mail
mailServer.sendmail(gmail_user, to, msg.as_string()) # envia o e-mail
mailServer.close() # Finaliza a Conexão com o servidor de e-mail
print "5- E-mail Enviado"

os.remove(picname)
time.sleep(10)
```