UNIVERSIDADE
LUSÓFONA

Robot - Desenvolvimento da Ligação ao AI

TFC026/19

Diana Margarida Simões Soares da Silva de Jesus (21703012)

Filipe José Costa Silva (21705317)

Trabalho orientado por Sérgio Ferreira

Trabalho Final de Curso | LEI | 31/1/2020

www.ulusofona.pt

Direitos de cópia

Índice

Índice de Figuras

Índice de Gráficos

Resumo

No âmbito da cadeira de Trabalho Final de Curso, da licenciatura em Engenharia Informática da Universidade Lusófona de Humanidades e Tecnologias, foi-nos proposto o desenvolvimento de um sistema de streaming de vídeo e áudio, simulando um sistema de segurança ou um monitor de criança, tendo também a capacidade de efetuar gravações de um período pré-definido.

O projeto do sistema de streaming veio com o objetivo de substituir o projeto proposto inicialmente que consistia numa parceria com a empresa **Intelligent Algorithms Technologies**, projeto este que foi descartado devido a falta de comunicação por parte da empresa.

O projeto inicial consistia num sistema de envio e receção de dados via streaming para o servidor da empresa em parceria, mesmo este ter sido descontinuado, foi utilizado como base para o projeto atual, que será abordado neste relatório.

Para o desenvolvimento do projeto abordado, serão utilizadas ferramentas open source, Python e os componentes são constituídos por um Raspberry Pi, um microfone e uma câmara.

Palavras-Chave: Streaming, Open Source, Python, Raspberry Pi.

Abstract

Within the scope of the Final Course Work course, of the degree in Engenharia Informática da Universidade Lusófona de Humanidades e Tecnologias, we were proposed to develop a video and audio streaming system, simulating a security system or a child monitor, also having the ability to make recordings for a pre-defined period of time.

The streaming system project came with the objective of replacing the initially proposed project with a partnership with the company Intelligent Algorithms Technologies, a project that was discarded due to lack of communication on the part of the company.

The initial project consisted of a system for sending and receiving data via streaming to the server of the company in partnership, even though it was discontinued, it was used as the basis for the current project, which will be addressed in this report.

This tool is a program that is developed in Python which will be installed in a RaspBerry Pi, which has a built-in microphone, two speakers and a visual sensor.

Keywords: Streaming, Open Source, RaspBerry Pi.

1. Identificação do Problema

No âmbito da parceria com a empresa Intelligent Algorithms Technologies, tínhamos como objetivo final desenvolver um robô assistente. Devido a complexidade do projeto, foi necessária à sua divisão em dois projetos mais pequenos.

Sendo assim, o nosso grupo ficou responsável pela elaboração de um sistema que iria permitir a captura de vídeo e som e então iria enviar esses dados para o servidor da empresa e então o servidor iria devolver uma resposta em formato de som ao qual teríamos de conseguir imitir.

Já a parte responsável pelo deslocamento do robô assistente, iria ficar ao cargo do grupo constituído pelos nossos colegas Gonçalo Santos e Fábio Silveira.

Devido a problemas em relação a falta de comunicação com a empresa e falta na entrega de informação e componentes para a realização do projeto, este teve de ser adaptado de maneira a que fosse possível excluir as funções da empresa, retirando-a assim do projeto na sua totalidade.

Com a saída da empresa, o projeto foi convertido para um sistema de streaming que permite gravações de um período definido.

Este sistema irá utilizar um Raspberry Pi que terá uma câmara e um microfone e então este irá enviar as imagens e sons capturados por si para o servidor local. No servidor os dados serão processados e então disponibilizados para o utilizador ter acesso a eles, seja em forma de transmissão de imagem em tempo real, a qual chamamos de streaming, ou em modo de pequenas gravações.

2. Viabilidade e Pertinência

Este projeto é uma excelente oportunidade de perceber o funcionamento de um sistema de streaming de vídeo e áudio, sistemas semelhantes aos utilizados pelo Skype e Twitch, onde é possível fazer o envio e receção de dados. Com este projeto também conseguimos perceber como desenvolver um sistema que faça gravação de vídeo e de som através da deteção de movimento.

Mas para o projeto seja viável, o áudio e vídeo tem de ter uma resolução alta para ser fácil de detetar objetos por parte do sistema e ter um som com pouco ruído, bem como ser possível dar uma resposta em tempo real, por outras palavras, com baixa latência.

Exemplos de utilização:

- Sistemas de vigilância com som (aviso: isto é uma violação de direitos de privacidade)

Situação Exemplo:

Uma câmara a fazer envio de imagens e som constante para um computador na rede local através de acesso do endereço IP do equipamento.

Do ponto de vista tecnológico, pretende-se que o robot possua mecanismos de registo de vídeo/áudio e transfira essa informação em tempo real para um servidor remoto onde serão processados em tempo útil, estando disponível para o utilizador consultar.

Num ponto de vista comercial, este projeto pode ser utilizado na criação de robot assistente para um centro comercial, aeroportos, isto é, um sistema que recebe inputs de vídeo e som, faz tratamento dessa mesma informação e transmite uma resposta relacionada à informação dada.

3. Solução Proposta

Para a realização do projeto do sistema de streaming, foi necessário recorrermos a componentes tecnológicos para podermos efetuar a captura de vídeo e de som e processar a informação e efetuar o seu envio para o servidor.

O sistema será constituído pelos componentes:

- Raspberry Pi e fonte de alimentação.

Figura 1 - Raspberry Pi

- Cartão de memória Micro SD para funcionar como disco Rígido do Raspberry Pi, sendo que o cartão terá de ter no mínimo 8GB de memória

Figura 2 - Cartão Micro SD

- Modulo da câmara do Raspberry Pi

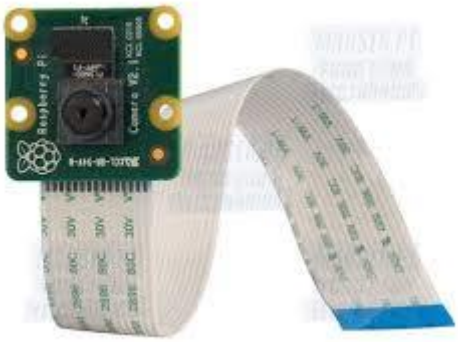

Figura 3 - Modulo da camara do Raspberry Pi V2

- Microfone com soundcard

Figura 5 - Soundcard

Figura 4 - Microfone

Por conta da facilidade de programação da linguagem Python, esta foi a escolhida para desenvolver o projeto de streaming, pois já existem bibliotecas que permitem de forma simples e rápida proceder ao envio de vídeo e de som capturado para o servidor em rede local.

Para a realização do ato de captura de vídeo e som, utilizamos o programa ffmpeg, que consiste num programa que ativa a câmara e microfone do Raspberry Pi, procede a guardar os dados capturados pela câmara e microfone e finaliza com a junção dos ficheiros de vídeo e som, criando assim um ficheiro do tipo RAW, que será enviado para o servidor. Chegando ao servidor, o mesmo programa irá converter o ficheiro RAW que foi recebido, para um ficheiro do tipo .mp4, onde este será exibido ao utilizador.

No gráfico 1 está representado em forma de diagrama como o envio e processamento dos dados obtidos pela câmara e microfone é efetuado até chegar ao nosso servidor local.

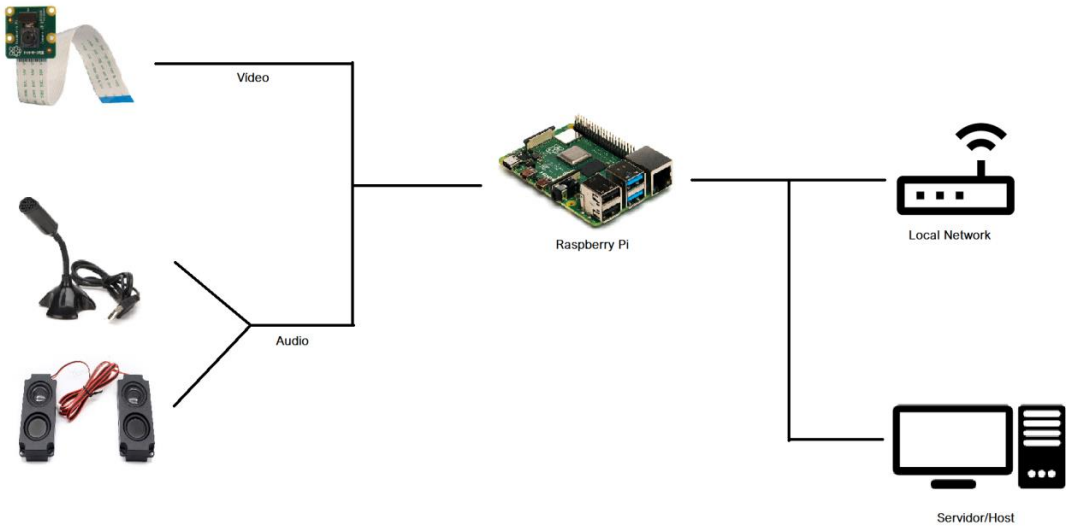

Gráfico 1 - Diagrama do Funcionamento

3.1. Modelos Conceptuais

O gráfico 2, representa um modelo conceptual correspondente ao funcionamento do servidor e todas as verificações que o mesmo efetua durante o seu funcionamento.

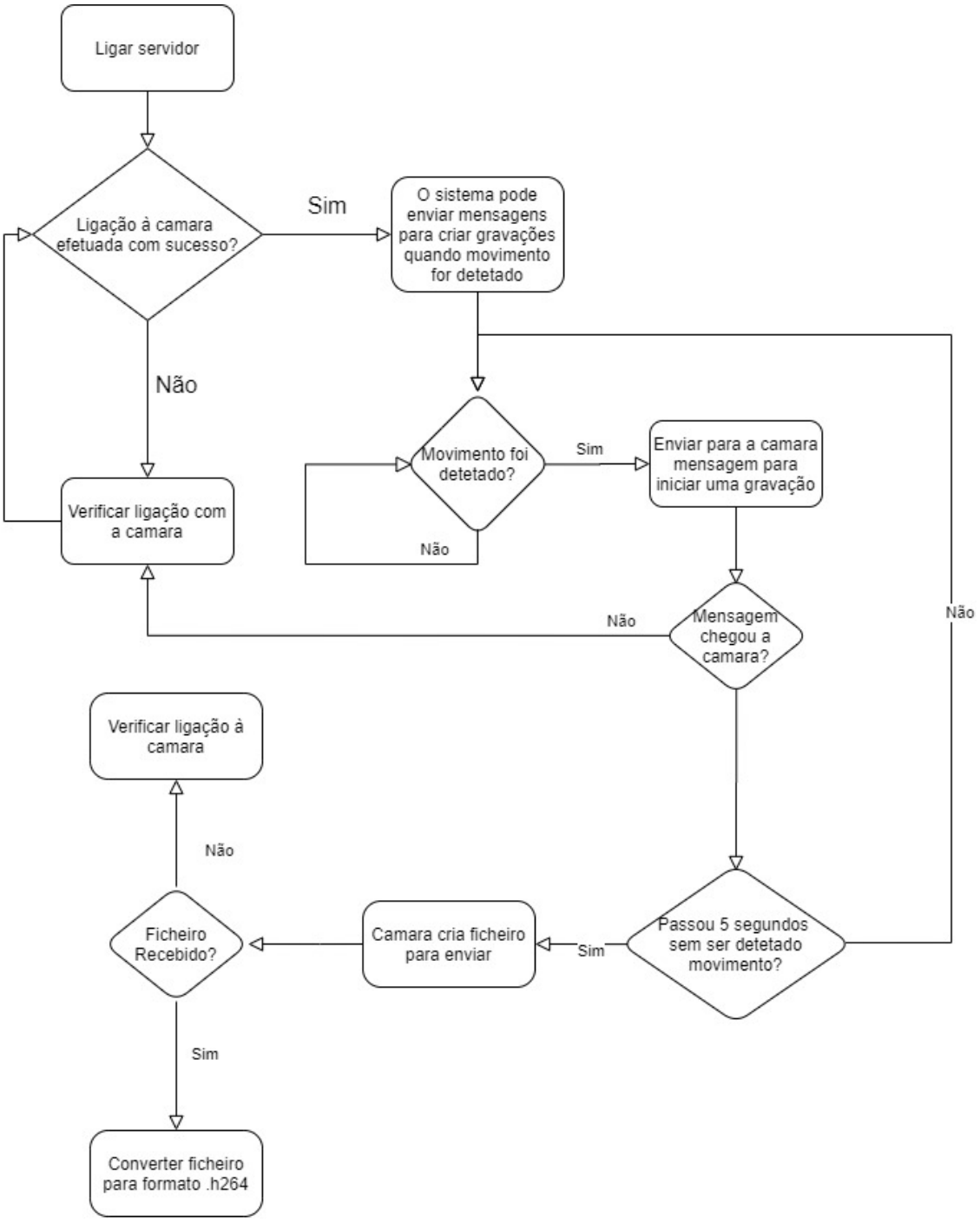

Gráfico 2 - Modelo Cliente

O grafico 3, representa também um modelo conceptual mas desta vez correspondente as ações e verificações que seram feitas pelo RaspBerry Pi do lado do cliente, isto é, do lado da camara e do microfone.

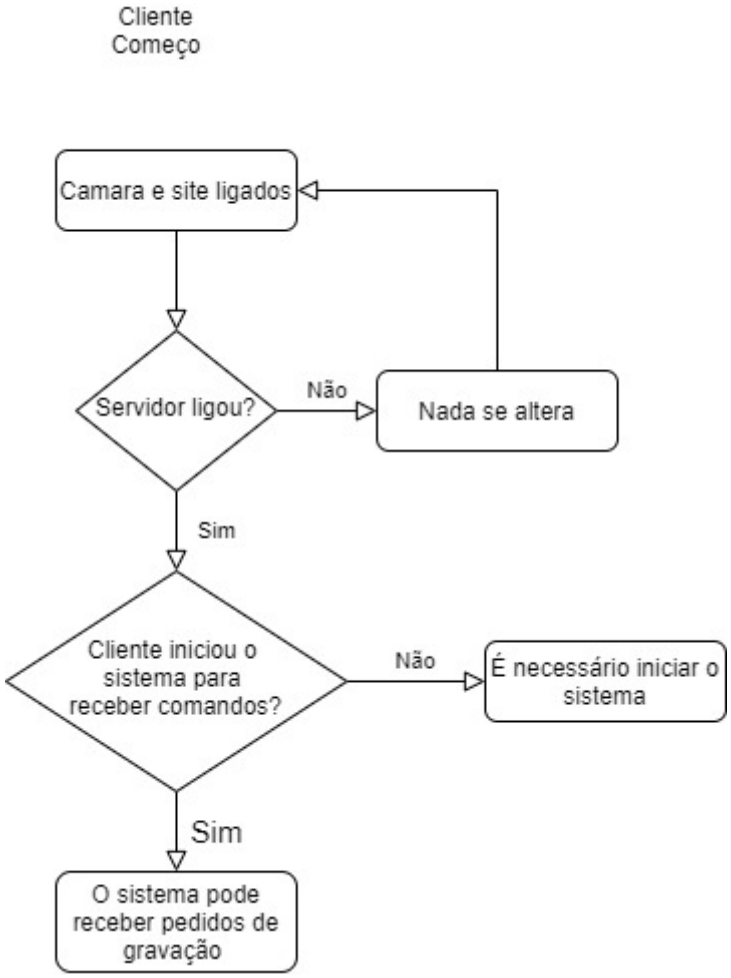

Gráfico 3 - Modelo Servidor

3.2. Alterações de Software

Como referido anteriormente, para o desenvolvimento deste projeto, foram utilizadas bibliotecas já existentes as quais foi necessário efetuar certas alterações para que os resultados correspondessem ao necessário para o nosso projeto.

Começando pelo lado do RaspBerry Pi, o primeiro ficheiro a ser alterado foi o ficheiro Data.json. Nesse ficheiro é possível fazer a alteração do utilizador e da palavra chave correspondente, nesse ponto mantivemos os originais da biblioteca, mas acrescentamos mais informação necessária, informação como a localização onde os dados seriam guardados, o IP do RaspBerry Pi e o IP do servidor local, o tempo de gravação após a câmara captar movimento, a resolução da câmara, o ID do dispositivo de gravação e a taxa de atualização da imagem.

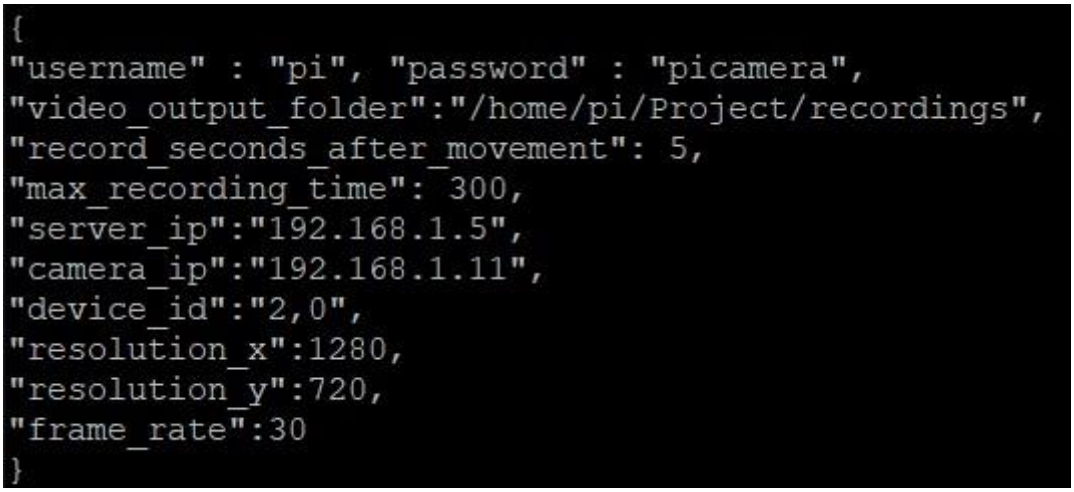

Figura 6 - Ficheiro Data.json Raspberry Pi

O ficheiro HTTPServer.py também teve de ser alterado. Neste ficheiro serão acedidos os dados do ficheiro Data.json, por essa razão foi necessário adaptar o ficheiro HTTPServer.py para receber os dados a mais inseridos.

```
data_file = open('/home/pi/Project/data.json')
stored_data = json.loads(data_file.read())
data_file.close()
AUTH_USERNAME = stored_data['username']
AUTH_PASSWORD = stored_data['password']
AUTH_BASE64 = base64.b64encode('{}:{}'.format(
    AUTH_USERNAME, AUTH_PASSWORD).encode('utf-8'))
BASIC_AUTH = 'Basic {}'.format(AUTH_BASE64.decode('utf-8'))
RESOLUTION_X = stored_data["resolution_x"]
RESOLUTION_Y = stored_data["resolution_y"]
FRAMERATE = stored_data["frame_rate"]
ROTATION = 0
HFLIP = True
VFLIP = True
```

Figura 7 - Ficheiro HTTPServer.py

O ficheiro Camera.py foi alterado pela mesma razão que o ficheiro HTTPServer.py, pois como a informação sobre a câmara no ficheiro Data.json foi alterada, neste ficheiro, a informação também teve de ser alterada de maneira a manter a consistência.

```
import time
import threading
import datetime
import os
import subprocess
import socket
import json

file = open('data.json')
stored_data = json.loads(file.read())
video_output_folder_json = stored_data["video_output_folder"]
record_seconds_after_movement_json = stored_data["record_seconds_after_movement"]
max_recording_time_json = stored_data["max_recording_time"]
device_id = stored_data["device_id"]
server_ip_json = stored_data["server_ip"]
camera_ip_json = stored_data["camera_ip"]
file.close()

class Camera:
    video_output_folder = video_output_folder_json
    record_seconds_after_movement = record_seconds_after_movement_json
    max_recording_time = max_recording_time_json
    server_ip = server_ip_json
    transfer_port = 5005
    message_port = 5006
    # ------------------------------------------------------------

    timer = 0
    detected_motion = False
    is_connected = False

    def __init__(self,url,id,up):
        self.ip = camera_ip_json
        self.url = url
        self.id = id
        self.is_connected = True
        self.username = up[0]
        self.password = up[1]
        record_thread = threading.Thread(target=self.recv_msg).start()
```

Figura 8 - Ficheiro Camera.py Informação

O ficheiro Camera.py foi alterado também de maneira a que permitisse que a gravação do video tivesse audio, sendo este capturado pelo microfone.

```
proc = subprocess.Popen(['ffmpeg', '-i', f'http://{self.username}:{self.password}@localhost:8000/delayed_stream.mjpg', '-f','alsa', '-i',
f"plughw:"+device_id, '-vcodec', 'copy', f"{output_filepath}"], stdin=subprocess.PIPE)
```

Figura 9 - Ficheiro Camera.py Audio

Para finalizar as alterações do lado do RaspBerry Pi, o ficheiro Use.py foi alterado para poder suportar as alterações a adições de informação feitas no ficheiro Data.json.

```
from CameraReceiver import CameraReceiver
import json

file = open('data.json')
stored_data = json.loads(file.read())
camera_ip = stored_data["camera_ip"]
file.close()

receiver = CameraReceiver([camera_ip])
```

Figura 10 - Ficheiro Use.py Raspberry Pi

Passando agora para o lado do Servidor, o primeiro ficheiro a ser alterado foi o ficheiro Data.json. Como o ficheiro no Raspberry Pi, este ficheiro Data.json vai funcionar com um ficheiro de configuração onde podemos altera o nome de utilizador e a sua palavra chave, neste caso essa informação foi mantida tendo sido só alterada os dados em relação ao IP da câmara, o IP do servidor local, o tamanho máximo que a pasta com os dados poderá ocupadar, a sua localizar e a sensabilidade do sensor de movimento existente na câmara.

Tem de se ter em consideração os seguintes factores quando se estiver a fazer a alteração dos dados:

- O utilizador e a palavra chave deveram ser os mesmo que foram indicados no ficheiro Data.json que se encontra no RaspBerry Pi.
- Um ponto a ter em consideração nesse ultimo dado é que quanto menor for o valor do campo da sensibilidade, maior será a sensibilidade.

```
"username" : "pi","password" : "picamera",
"cameraIP": "192.168.0.251", "localIP": "192.168.0.230",
"max_storage": 25, "video_output_folder":"D:/ANO3/TFC/TFC_Server/Recordings/Camera2",
"motion_sensitivity": 10
```

Figura 11 - Ficheiro Data.json Servidor

Os restantes ficheiros alterados no lado do servidor foram modificados de maneira a manter a consistencia dos dados entre o ficheiro de configuração, o ficheiro Data.json, e os restantes ficheiros necessários para o funcionar do nosso projeto. Esses ficheiros alterados são o ficheiro FileReceiver.py, o ficheiro Detector.py e o ficheiro Use.py.

O ficheiro FileReceiver.py é responsável pela receção e conversão dos ficheiros enviados para o servidor por parte do RaspBerry Pi, ficheiros esses que correspondem as gravações e streaming.

```
import socket
import threading
import datetime
import subprocess
import time
import os
import json

file = open('data.json')
stored_data = json.loads(file.read())
video_output_folder = stored_data["video_output_folder"]
host = stored_data["localIP"]
max_storage = stored_data["max_storage"]# In Gb
file.close()
```

Figura 12 - Ficheiro FileReceiver.py

O ficheiro Detector.py é o ficheiro responsável por gerir o sensor de movimentos existente na câmara.

```
import cv2
import threading
import numpy as np
import urllib.request
import imutils
import json
import socket

stream_url = "http://{}:8000/stream.mjpg"


# Getting private data from 'data.json'
file = open('data.json')
stored_data = json.loads(file.read())
username = stored_data["username"]
password = stored_data["password"]
motion_sensitivity = stored_data["motion_sensitivity"]  # Higher number = less detection
file.close()
```

Figura 13 - Ficheiro Detector.py

O ficheiro Use.py tem como função juntar e correr todos os ficheiros utilizados para que o projeto possa correr.

```
from CameraReceiver import CameraReceiver
import json

file = open('data.json')
stored_data = json.loads(file.read())
cameraIP = stored_data["cameraIP"]
file.close()

receiver = CameraReceiver([cameraIP])
```

Figura 14 - Ficheiro Use.py Servidor

3.3. Funcionamento Exemplo

Neste capítulo iremos demostrar um exemplo para o funcionamento do nosso projeto, para poder mostrar de uma maneira um pouco mais clara como o programa funciona na sua totalidade.

Em primeiro lugar, será necessário verificar se os dados no ficheiro Data.json estão de acordo com os sistemas onde serão utilizados, tanto do lado do RaspBerry Pi como do seu computador que será utilizado com servidor local. Os dados a confirmar se estão bem colocados serão:

- IP do servidor
- IP do Raspberry Pi
- Localização das pastas onde os dados serão colocados

Opcionalmente poderá alterar os seguintes dados:

- Alterar o tempo de duração do vídeo após movimento ser detetado e o tempo total do vídeo no RaspBerry Pi.
- Alterar o nome do utilizador e a palavra chave correspondente para aceder à câmara. Por predefinição, o nome do utilizador é pi e a palavra chave picamera. Estes dados estão predefinidos desta maneira tanto no RaspBerry Pi quanto no servidor local.
- Alterar o tamanho do espaço máximo de armazenamento para os vídeos.
- Definir a sensibilidade do detetor de movimentos da câmara.

A nível do lado do RaspBerry Pi, deverá abrir dois terminais Linux e dirigir-se a pasta onde se encontra o projeto e de seguida será necessário que corra os seguintes comandos em simultâneo:

1. python3 HTTPServer.py
2. python3 use.py

Deverá ter atenção que a ordem pela qual os comandos são corridos têm importância, por essa razão é que os comandos se encontram numerados com os números 1 e 2, pois devem ser corridos nesta ordem.

Para realizar os passos acima indicados poderá abrir os terminais diretamente na pasta onde se encontra o projeto, recorrendo a ferramenta Linux File Manager.

Agora iremos correr os seguintes passos no seu computador pessoal, pois este será utilizado como servidor local. Existem duas maneiras destintas de utilizar o programa, sendo estas:

- Correr o website desenvolvido
- Estabelecer ligação ao Raspberry Pi através do seu IP

Iremos começar pela maneira menos convencional, a ligação ao Raspberry PI através do seu IP. Deverá ter atenção que caso queira utilizar a seguinte abordagem, só irá visualizar o streaming.

Para começar, terá de abrir duas linhas de comando e seguir para a pasta onde se encontra o ficheiro Use.py. De seguida deverá correr as seguintes linhas de código, tendo em atenção que deverão correr em linhas de comando distintas uma da outra:

- python use.py
- python FileReceiver.py

Após as linhas de código terem sido corridas, deverá abrir o seu navegador web e colocar o seguinte link, só tenha em atenção que onde está indicado raspberrypi_IP deverá ser colocado o IP correspondente ao seu Raspberry Pi:

- raspberrypi_IP:8000/stream.mjpg

Caso deseje utilizar o website desenvolvido e ter acesso as gravações assim com ao streaming, deverá executar mais alguns passos após os que fez até agora.

Terá de abrir mais uma linha de comando e direcionar-se a pasta onde encontra-se o ficheiro Manage.py. Quando estiver dentro dessa pasta na sua linha de comandos deverá correr a seguinte linha de código:

- python manage.py runserver

Após a linha ter sido corrida, deverá abrir o seu navegador web e colocar o seguinte link:

- http://127.0.0.1:8000/

Quando a página for carregada, será lhe pedido que faça o login caso não tenha feito, caso já tenha feito, a página será aberta automaticamente. Na página terá disponível um menu com duas opções disponíveis:

- Replay
- Cameras

Na primeira opção, o Replay, terá a listagem de todos os ficheiros referentes às gravações efetuadas recentemente.

Caso selecione a segunda opção, Cameras, terá uma listagem de todas as câmaras registadas no sistema. Para poder aceder a transição de streaming, terá de clicar na câmara que deseja visualizar e então será pedido que efetue o login novamente por motivos de segurança e então terá acesso as imagens em tempo real da câmara.

4. Planeamento

1ª fase – Criar um desenho conceptual (22/11/2019)

2ª fase – Estudar, entender e fazer tutoriais acerca de API's para streaming (presente)

3ªfase – Adquirir o material (a cargo do parceiro empresarial) (31/3/2020)

4ª fase – Criar a API de streaming (fazer ações básicas) (24/4/2020)

5ª fase – Fazer a comunicação com a inteligência artificial, verificar se a comunicação ocorre sem problemas e tratar de problemas de concorrência (24/5/2020)

6ª fase – Verificar se os inputs da API são enviados e se as mensagens de som enviadas pela AI são recebidas e efetuadas (15/6/2020)

7ª fase – Fazer a comunicação entre o nosso RaspBerry e a IA e os componentes do projeto de navegação (30/6/2020)

8ª fase – Colocar todos os sistemas do robot – num único RaspBerry (opcional)

5. Calendário

8/11/2019 – 1ª Reunião com o Professor Sérgio Ferreira acerca do plano conceptual do Trabalho Final de Curso

12/11/2019 – 1ª Reunião com o Representante da empresa "Intelligent Algorithms Technologies", Breno Lehmann e o grupo responsável pela construção do corpo do robot (constituintes: Gonçalo Santos e Fábio Silveira) para planear o desenvolvimento do Projeto

22/11/2019 – 2ª Reunião com o Professor Sérgio Ferreira para revisão do relatório para 1ª entrega intercalar

13/12/2019 – 2ª Reunião com o Representante da empresa "Intelligent Algorithms Technologies", Breno Lehmann, o grupo responsável pela construção do corpo do robot (constituintes: Gonçalo Santos e Fábio Silveira) e o Professor Sérgio Ferreira para consultar a listagem de produtos necessários para o projeto e visita ao laboratório de robótica para pedir autorização de uso dos utensílios lá disponíveis.

24/1/2020 – 3ª Reunião com o Representante da empresa "Intelligent Algorithms Technologies", Breno Lehmann, o grupo responsável pela construção do corpo do robot (constituintes: Gonçalo Santos e Fábio Silveira) para verificar a listagem dos componentes necessários para a criação do projeto, bem como pequena pesquisa de alternativas a componentes propostos

31/1/2020 – Entrega intermédia do Relatório

6 a 10/2/2020 – Apresentação do Relatório perante os Júris responsáveis da Unidade Curricular TFC.

6. Resultados

Neste capítulo iremos apresentar os resultados obtidos em teste realizados para determinar o tempo de captura, processamento e envio dos vídeos.

Resolução	FPS(taxa de atualização da imagem)	Tempo médio de gravação (s)	Tempo médio necessário para a gravação estar disponível no servidor (s) RaspBerry Pi 3	Tempo máximo necessário para a gravação estar disponível no servidor (s)	Tempo mínimo necessário para a gravação estar disponível no servidor (s)	Tempo médio necessário para a gravação estar disponível no servidor (s) RaspBerry Pi 4
1920x1080	30	~6.23	~38.37	42.1	34.96	~35.36
		~12.6	~70.12	72.97	65.5	~54.44
		~19.01	~99.36	101.03	97.92	~79.93
	24	~5.68	~40.18	43.46	35.69	~32.06
		~9.52	~72.58	76.29	71.94	~50.86
		~14.96	~112.30	116.35	107.53	~90.33
1280x720	30	~5.5	~35.76	31.27	45.39	~27.76
		~11.72	~51.36	51.36	60.55	~41.32
		~16.74	~73.35	77.69	69.31	~59.01
	24	~5.07	~35.94	40.5	32.34	~28.91
		~9.85	~65.05	66.0	58.35	~52.33
		~14.88	~94.65	99.31	90.38	~76.14
960x 540	30	~7.37	~40.67	45.43	32.88	~32.72
		~12.63	~58.4	62.94	55.12	~49.98
		~17.81	~78.01	81.55	75.44	~62.75
	24	~4.75	~34.65	37.18	31.4	~27.87
		~9.53	~70.07	71.14	68.08	~56.37
		~14.33	~88.55	95.87	82.47	~71.07

Os dados obtidos e representados na tabela acima têm em consideração os requisitos dos componentes que são utilizados para o projeto.

A nível do Raspberry Pi, as especificações correspondem as seguintes:

- Raspberry Pi 3B 1Gb
- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
- 1GB RA

A nível do servidor, foi utilizado um computador Asus ROG GL552J com as seguintes especificações:

- Intel(R) (TM) i7-4720HQ CPU @ 2.60GHz
- RAM PDR 3L
- DDR3L 8GB RAM

7. Conclusão

Com o desenvolvimento deste projeto conseguimos aprofundar os nossos conhecimentos a nível de envio de dados via rede assim como conseguimos entrar um pouco neste mundo da transição de vídeo e som vai streaming, algo bastante popular atualmente graças ao desenvolvimento dos jogos online e dos respetivos torneios a nível mundial.

Os conhecimentos base que recebemos durante a nossa licenciatura foram bastante uteis durante a realização deste projeto, expecialmente as bases da linguagem de programação Python que nos foram fornecidas pelo professor Manuel Pita durante as aulas da cadeira de Sistemas de Informação Multimédia e a cadeira de Inteligência Artificial.

Os conhecimentos obtidos durante a cadeira de Redes de Computadores, lecionada pelo professor Miguel Tavares, foi outro tema bastante relevante para o desenvolvimento deste projeto, pois o desenvolvimento da comunicação entres os dois sistemas foi efetuada sobre o formato de sockets.

7.1. Dificuldades Encontradas

A falta de comunicação por parte da empresa Intelligent Algorithms Technologies, quer tenha sido em formato de falta de documentos com descrições básicas dos requisitos requisitados para o desenvolvimento do projeto inicial e pelo incumprimento ao fornecer os componentes necessários para a realização do mesmo foi o principal obstáculo que tivemos durante a realização do projeto.

Por consequência desses problemas que tivemos com a empresa em pareceria foi a razão pela qual o nosso cronograma acabou por ficar tão reduzido, o que criou bastantes conflitos na parte da implementação da informação já obtida durante o período de realização do primeiro projeto.

Pelas razões indicadas acima, pensamos que a coordenação deveria criar um procedimento que possa melhorar as parcerias com empresas durante a realização de TFCs, de maneira a que tanto a empresa com os alunos e a instituição de ensino, neste caso a faculdade, ficassem protegidas de situações constrangedoras como a que ocorreu durante o desenvolvimento do projeto inicial.

Seria bastante interessante desenvolver cadeiras opcionais ou workshops durante o ano letivo que complementassem a TecWeb e os seminários que decorrem durante essa altura.

7.2. Agradecimento

Gostaríamos de dar um agradecimento ao professor Sérgio Ferreira e ao professor José Faísca pelo o enorme apoio e encaminhamento que nos foi dado durante a realização deste problema, seja a resolver erros existente, ideias de como melhorar o projeto no seu todo, disponibilização de material, seja eletrónico para pesquisa seja físico para a realização do projeto.

Damos um especial agradecimento ao engenheiro informático Rui Jesus, ao membro Sonic do servidor da plataforma Discord “Raspberry Pie” e ao utilizador da plataforma GitHub Ruud Brouwers, pelo seu enorme apoio em dúvidas e problemas que nos foi surgindo durante o projeto.

Gostaríamos de gradecer ao professor Pedro Alves e a coordenação do curso pela sua disponibilidade total em resolver qualquer problema que tivéssemos durante o recorrer da licenciatura nestes 3 anos.

Gostaríamos de finalizar agradecendo a todos os professores da licenciatura em Engenharia Informática pelo seu apoio e dedicação durante os 3 anos que frequentamos a licenciatura.

Bibliografia

[1] Python (1991) – Linguagem de Programação. Disponível em https://www.python.org/. [consultado em 21-11-2019]

[2] ROS (2007) – Middleware de Robótica. Disponível em https://www.ros.org/. [consultado em 26-1-2020]

Links de Pesquisa:

- https://kamranicus.com/guides/raspberry-pi-3-baby-monitor#installing-picam
- https://github.com/Ruud14/SecurityCamera
- https://github.com/Ruud14/Django-Camera-View-And-Playback

Anexos

Guião de instalação:

- https://drive.google.com/file/d/10iGaheCN-oQEQm1CZvpnJv5N-dGsuUPX/view?usp=sharing

Link do Projeto:

- https://github.com/Jol273/TFC2020-ULHT-Diana-Filipe

Vídeo:

- https://youtu.be/TVE0z4MBEhc

Glossário

Raspberry Pi – Computador de placa única do tamanho reduzido, desenvolvido no Reino Unido, pela Fundação com o mesmo nome do produto.

Open Source - Modelo de desenvolvimento criado em 1998, que promove o licenciamento livre para o design ou esquematização de um produto, e a redistribuição universal desses, com a possibilidade de livre consulta, examinação ou modificação do produto, sem a necessidade de pagar uma licença comercial, promovendo um modelo colaborativo de produção intelectual.

RAW - Tipo de ficheiro que ainda não sofreu nenhum processo de compressão (p.e. mp4, JPEG).

Port Forwarding - Configuração de um router para que um determinado serviço que esteja a ser executado num desktop na rede interna possa ser acedido por outros dispositivos através da Internet.