INSTITUTO SUPERIOR TÉCNICO

REDES MÓVEIS E SEM FIOS

Relatório Final

Development of Raspberry Pi based video surveillance system with PTZ camera, microphone and WiFi connectivity

80872 – Magali Correia

81216 – Bernardo Amaral

Prof. António Cordeiro Grilo

Lisboa, 3 de junho de 2018

I. Introdução

O objetivo deste projeto é desenvolver um sistema de vigilância PTZ (*pan, tilt* & zoom) remotamente acessível por uma aplicação android para telemóvel, a partir da qual é possível ver o *stream* de vídeo em tempo real, e ainda controlar o movimento horizontal e vertical da câmara. O sistema de vigilância consiste num *Raspberry Pi* ao qual está ligada uma webcam USB, que transmite por IP o *stream* de vídeo proveniente da dita câmara, este terá também a correr um servidor que recebe os comandos da aplicação android.

II. Solução Encontrada

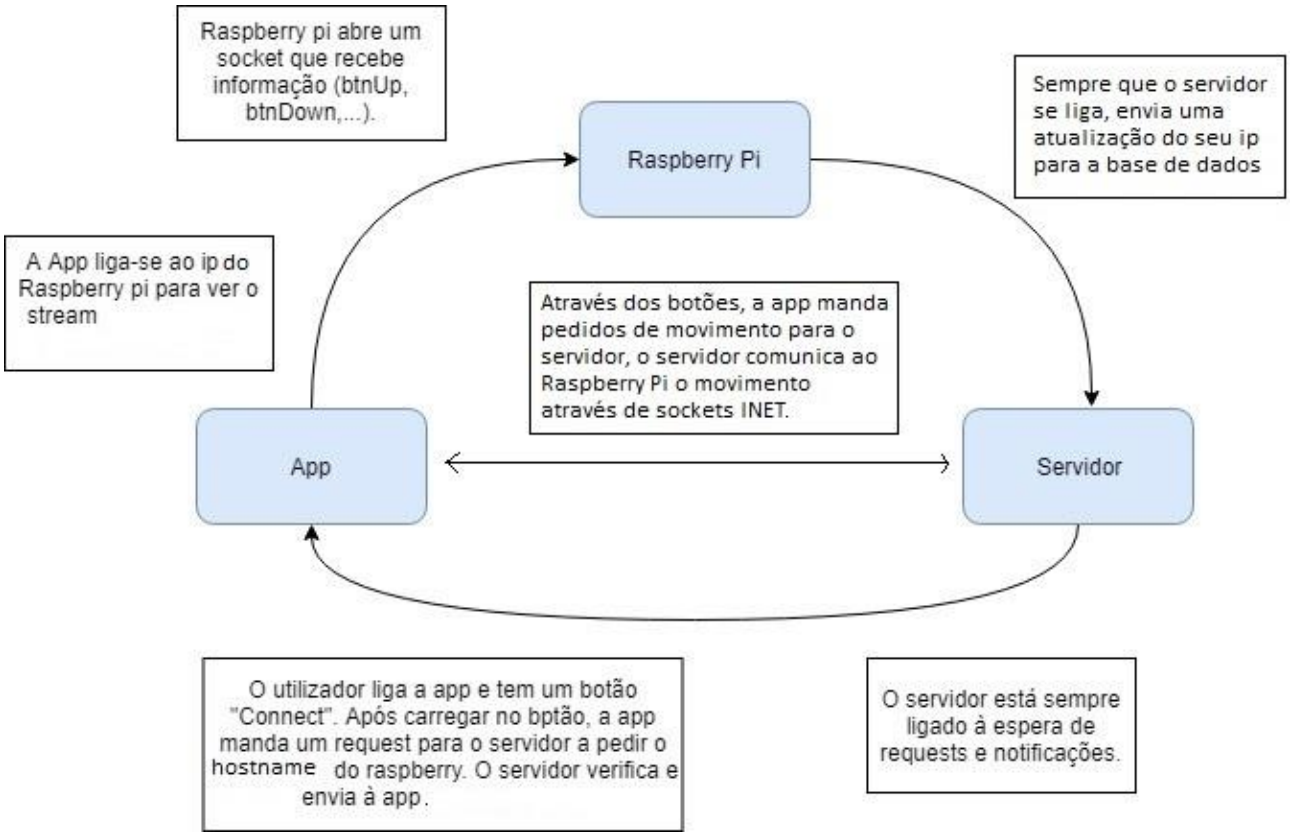


Figura 1 – Arquitetura Geral

De modo a desenvolver e a concretizar este sistema é necessário fazer *port forwarding* dos portos necessários do *Raspberry Pi*, no router local onde este se encontrar, para que o seu ip possa ser acessível diretamente pela aplicação android por *wi-fi*, independentemente da rede em que esta se encontre. Apesar de este ser o objetivo, o grupo não conseguiu com que o stream de vídeo fosse transmitido por *port forwarding*, ou seja, este apenas funciona na aplicação android num transmissão local, assunto que será abordado no relatório. O *Raspberry Pi* terá portanto um IP dinâmico (bem

como os routers podem de tempo em tempo mudar o seu endereço IP público) e como tal, para que todo o sistema possa continuar a funcionar independentemente da rede local em que se está, sempre que o Raspberry Pi se ligar à rede, envia um pedido para o servidor, de modo a atualizar o seu IP, através de um script em Python cuja execução foi agendado no *Crontab* para ser feita sempre que o dispositivo é ligado.

O *Raspberry P*i transmite o *stream* através do programa VLC, pelo protocolo RTSP.

Para além de transmitir vídeo, o *Raspberry Pi* também tem um programa a ser executado que recebe os comandos de controlo da câmara, enviados pela aplicação android através de um *socket* INET (também agendado no Crontab).

O servidor, feito em PHP, tem uma base de dados SQL que armazenará o IP atualizado do *Raspberry Pi*, e receberá pedidos da aplicação Android. Consoante o tipo de pedido por parte da aplicação, o servidor enviará por *sockets* o movimento da câmara associado que por sua vez é recebido pelo programa no *Raspberry Pi*. O servidor também receberá notificação da atualização do endereço IP do *Raspberry Pi*.

O layout da aplicação *Android*, como será discutido nas secções seguintes será simples e conterá duas atividades, uma de ligação ao *Raspberry Pi*, e outra de observação do *stream* e de controlo da câmara.

III. Módulos

Nas secções seguintes serão analisadas com mais detalhe as diferentes secções desenvolvidas neste projeto. Pode-se dividir o trabalho desenvolvido em três módulos: *Raspberry Pi*, Servidor e Aplicação Android.

III.1. *Raspberry Pi*

O *Raspberry Pi* utilizado no projeto tem o sistema operativo *Raspbian* (versão *Stretch*), a última distribuição do OS *standard* usado em *Raspberry Pi's*, e é também o que tem mais suporte para os programas mais usados.

Os componentes a desenvolver neste módulo são: *Port Forwarding* e comunicação com o exterior, execução da transmissão do *stream* de vídeo em tempo real por *wi-fi*, e o controlo e configuração da webcam.

III.1.1. Port Forwarding

O port forwarding, ou "redireccionamento de portas" consiste no simples processo de permitir o tráfego proveniente da rede internet exterior à rede privada em que está o *Raspberry Pi*, nos portos indicados, de modo a que chegue a um programa que esteja a ser executado por este. Neste caso permitiu que a aplicação pudesse aceder ao programa de controlo da câmara, que se encontra a ser executado pelo *Raspberry Pi*.

Neste projeto fez-se *port forwarding* do router Technicolor TG784n v3, e para tal foi seguido um conjunto de passos simples que permitiram que o *port forwarding* fosse possível.

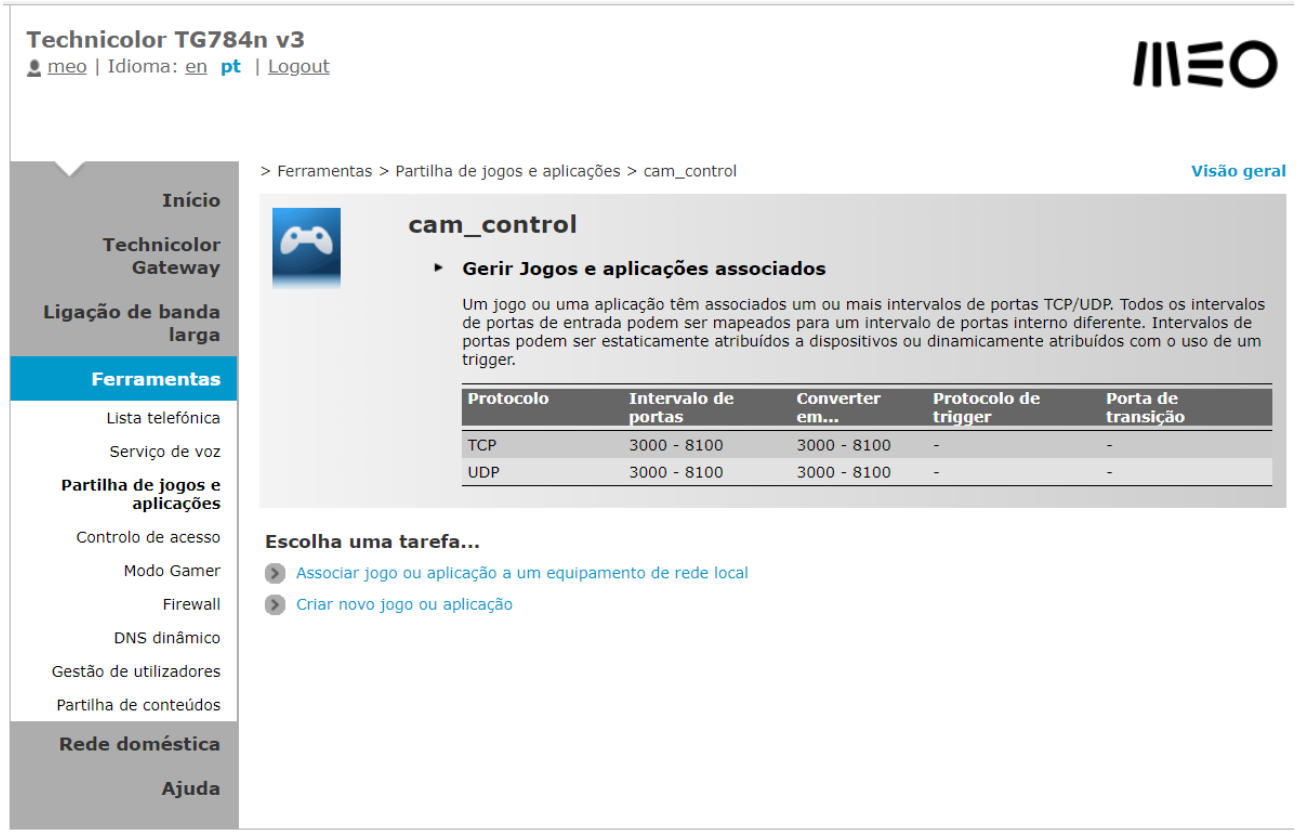


Figura 2 – Aplicação criada na central de controlo do router

Através do endereço IP do router acedeu-se à central de controlo do router e em "Ferramentas > Partilha de jogos e aplicações" criou-se uma aplicação "*cam_control"* que quando associada ao *Raspberry Pi* redireciona todo o tráfico que é passa pelo intervalo de portas indicado.

III.1.2. Transmissão do Stream de Vídeo

Após uma extensa pesquisa, foram encontrados vários programas com diferentes tipos de transmissão de vídeo sendo alguns deles VLC, Motion, GStreamer, MJPEG, FFMPEG e UV4L. O VLC permite transmitir vídeo por ambos http (TCP) e rtp (UDP) e foi o software escolhido para fazer a transmissão de vídeo para a aplicação, visto ser o que apresentou menos problemas a nível de transmissão local.

Foram encontrados inúmeros problemas com os restantes programas. O Uv4l é de momento o programa ideal para fazer stream a partir de um Raspberry Pi ligado a uma webcam. No entanto, após inúmeras tentativas que resultaram em erro, verificou-se que ao ser executado, não permite qualquer controlo da câmara a partir de outros meios, o que neste caso é absolutamente necessário, de modo a poder controlar a câmara a partir do programa construído. O programa FFMPEG, deu erros de compilação e também de execução. No entanto, a sua utilização seria ideal, pois permite fazer port forwarding do stream e simultaneamente vê-lo na aplicação, mas não foi possível que funcionasse. Quanto ao programa Motion, foi possível fazer transmissão por port forwarding, no entanto, não foi possível coloca-lo na aplicação da maneira desejada. Os programas GStreamer e MJPEG não foram muito explorados visto que já estão relativamente descontinuados, visto que já são antigos e já foram ultrapassados por softwares mais recentes como os previamente mencionados.

O stream por VLC, o programa escolhido, pode ser feito através de dois protocolos diferentes, ou HTTP, ou RTSP. O protocolo HTTP funcionou por port forwarding, mas não se adaptou à aplicação Android. O protocolo RTSP funcionou perfeitamente na aplicação, mas quando se tentou fazer port forwarding com o mesmo, não foi possível. Isto deve-se ao tipo de protocolo que é RTSP, e à sua ocupação de portas no router, e até mesmo às definições do router como *gateway*, que dependendo do dispositivo, procede de maneira diferente face ao redireccionamento do tráfego. Deste modo o não foi possível resolver o problema da sua transmissão por *port forwarding*. No entanto, apesar disso, este foi o programa selecionado para a continuação do projeto, visto que foi o único que funcionou corretamente na aplicação, de todos os testados.

III.1.3. Controlo e Configuração da Webcam

Utilizou-se uma biblioteca escrita em linguagem C fornecida pelo INESC com o objetivo de controlar dispositivos exteriores (neste caso a webcam), através do software *video4linux2*. A partir da biblioteca escreveu-se um programa através do qual é controlado o movimento da câmara utilizando *sockets INET* que recebem pedidos do servidor diretamente pela rede. Foi-lhe atribuída a designação *cam_control.c*. O movimento da câmara dá-se na vertical e na horizontal.

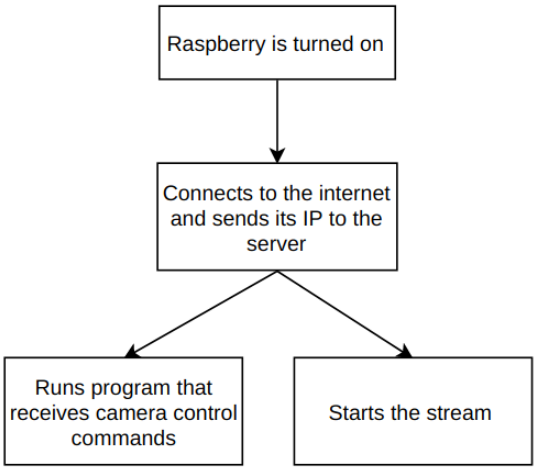


Figura 3 - Atividade do Raspberry Pi desde o momento em que é ligado ao momento em que faz o stream e recebe controlos da aplicação Android

III.2. Servidor

O servidor consiste numa base de dados *MySQL* e em duas páginas em PHP. A primeira página em PHP, index.php, recebe um *hostname* da aplicação android, faz um query à base de dados e retorna à aplicação por um JSON se este *hostname* se encontra ou não na base de dados. A seugnda página PHP, movement.php, recebe um *hostname* e uma indicação de movimento da câmara, e faz um query à base de dados de modo a obter o endereço IP correspondente ao *hostname* recebido, e de seguida manda a indicação de movimento, por sockets INET, para o Raspberry Pi, através do seu endereço. Na base de dados é guardado o endereço IP do *Raspberry Pi*, sendo este atualizado periodicamente pelo Raspberry Pi, cada vez que este é ligado.
Será usado o cluster sigma do técnico para alojar este servidor.

```
MySQL [ist181216]> select * from PiAddress;
+-------------+------------+---------------------+
| name        | ipAddress  | lastUpdate          |
+-------------+------------+---------------------+
| raspberrypi | 1442201284 | 2018-06-03 17:11:39 |
+-------------+------------+---------------------+
1 row in set (0.00 sec)
```

```
MySQL [ist181216]> show tables;
+---------------------+
| Tables_in_ist181216 |
+---------------------+
| PiAddress           |
+---------------------+
1 row in set (0.00 sec)
```

Figura 4 – Query's feitos à base de dados, como exemplo.

III.3. Aplicação Android

A aplicação Android foi desenvolvida em *Android Studio* em Java. Cada ficheiro *.java* representa uma *"atividade"*, e neste projeto utilizara-se duas atividades principais: *MainActivity.java* (que representa o ecrã inicial); e *DisplayActivity.java* (que controla o *stream*). Na Figura 4 é apresentado o layout para a aplicação Android utilizada neste projeto:

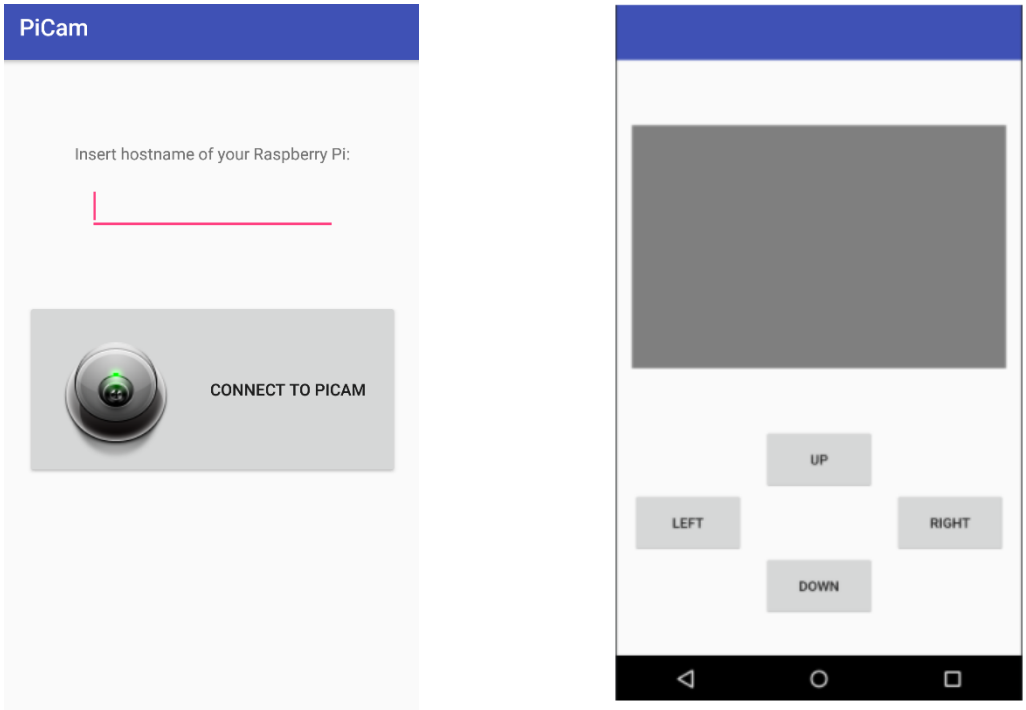


Figura 5 - Layout inicial da aplicação android: 5.1. Activity inicial; 5.2. Segunda Activity na qual estão os botões de controlo do Raspberry e o Video View onde será apresentado o stream

Ao ser pressionado o botão "Connect to PiCam" da atividade inicial, a aplicação envia um pedido ao servidor para verificar o hostname do *Raspberry Pi*. Assim que o servidor recebe o pedido

e confirma que o Raspberry Pi está na base de dados, passa-se para a *"atividade"* seguinte na qual será apresentado o *stream* do vídeo bem como os botões de controlo da câmara. Tanto o vídeo como a transmissão dos pedidos de controlo serão feitos através do IP do *Raspberry Pi*.

A aplicação será também constituída por ficheiros *.xml* nos quais está definida a estrutura das atividades. Estes ficheiros associam cada botão, *TextView* e *VídeoView* da aplicação a um id. Este id é futuramente utilizado como referência nos ficheiros .java.

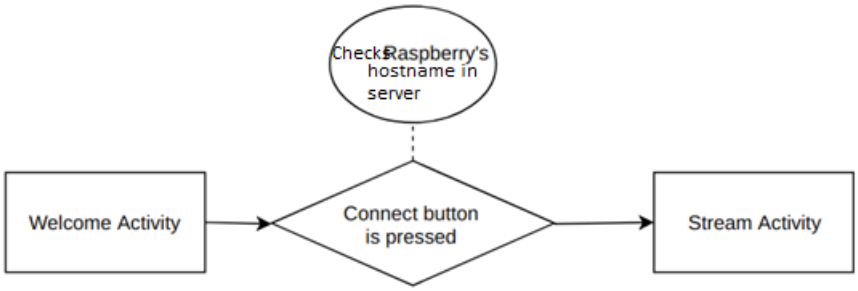


Figura 5 - Fluxograma que representa o funcionamento da aplicação Android

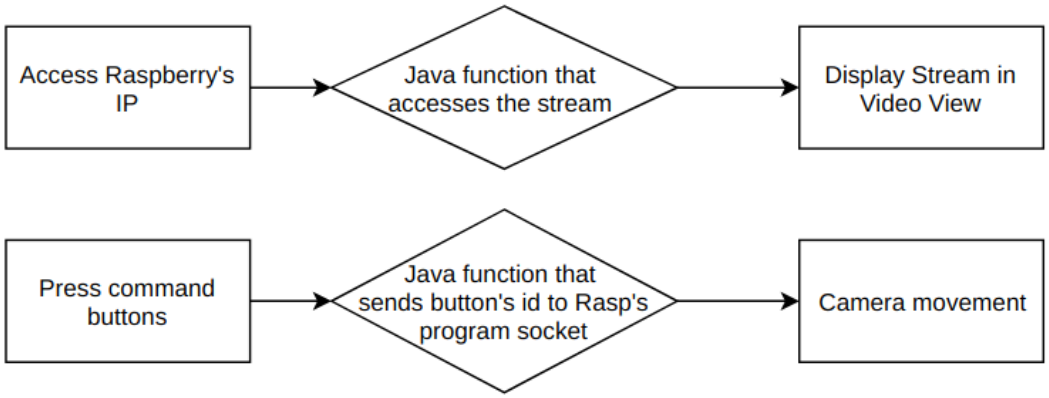
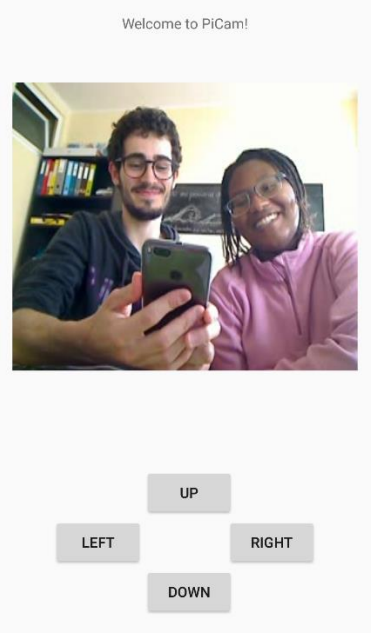


Figura 6 - Fluxogramas que representam o funcionamento da Stream Activity, actividade da app Android em que é feito o stream de vídeo e o controlo da câmara

V. Considerações Finais

Este trabalho serviu de continuação à entrega intermédia, e nele foram feitos consideráveis avanços, pois o previsto foi executado na sua maioria, tendo a ficar a faltar de relevante, a transmissão do stream por port forwarding, factor que resulta numa aplicação menos versátil e extensível do projeto desenvolvido. No entanto, a aplicação continua a ter bastantes utilidades, mesmo sendo o seu uso restrito a uma ligação local.

Como solução alternativa, o grupo deparou-se com o módulo de câmara próprio e original do Raspberry Pi, que várias pesquisas levaram a concluir que teria sido uma maneira bastante mais facilitada e útil para a utilização de um Raspberry Pi para a transmissão de vídeo. Apesar desta não ter possibilidade de movimentação, poderiam ser acrescentados servos mecânicos ao material do projeto, de modo a torna-lo também mais interativo no sentido de hardware. O grupo chegou a esta conclusão, visto que a maior parte do trabalho e do tempo disponibilizado para fazer este projeto foi feito a fazer pesquisas intensivas relacionadas com a câmara usb e as suas possibilidades de transmissão.



VII. Bibliografia / Webgrafia

- Raspberry Pi:

-https://www.raspberrypi.org/forums/

-https://pt.wikipedia.org/wiki/Crontab

-https://forum.meo.pt/t5/Tutoriais/Portforwarding-TECHNICOLOR-TG784n-V3/td-p/24343

-https://hmbd.wordpress.com/2016/08/01/raspberry-pi-video-and-audio-recording-and-streaming-guide/

-https://itsfoss.com/solve-open-shared-object-file-quick-tip/~

- Aplicação Android:

-https://developer.android.com/guide/index.html

-https://stackoverflow.com/questions/16220069/rtsp-gstreamer-pipepline-to-android-videoview?rq=1