

img_hrv_lan – interface gráfica para aquisição de vídeo e frequência cardíaca em uma LAN: captura um dataset de dois voluntários interagindo em dois computadores conectados a uma LAN enquanto visualizam imagens mostradas na tela.

1. Introdução/Justificativa

O objetivo deste software é capturar vídeo, áudio e sinal de ECG de dois voluntários usando dois computadores conectados a uma LAN. Durante a captura, ilustrações de um dataset serão exibidas na tela. O objetivo é reproduzir o experimento de Dra. Suzanne Dikker[1].

As principais características do software são:

- 1 - Possui janela do operador, onde se controla o início da captura.
- 2 - Permite estabelecer a rotina de captura. tempo de exibição das imagens, pausa etc.
- 3 - Possui janelas para os subjects.
- 4 - Deve ser capaz de exibir as imagens contidas em uma pasta aleatoriamente.
- 5 - Deve ser capaz de salvar o sinal de ECG.
- 6 - Deve ser capaz consultar as cameras disponiveis no localhost e nos demais hosts da rede. Deve ser possível localizar automaticamente os servidores de câmeras e de microfones na rede com uma determinada porta aberta.

2. Classes

2.1. Classe LanDevice

A classe LanDevice é uma camada que expõe as funcionalidades dos dispositivos de captura disponíveis na LAN. Os dispositivos em que estamos interessados são as câmeras, microfones e o sensor cardíaco Polar H10. Possui os seguintes métodos:

`start_server`

Inicia o servidor que disponibiliza os dispositivos do localhost para a rede.

`stop_server`

Interrompe o servidor.

`list_servers`

Envia um sinal de broadcast para a rede para localizar todos os servidores ativos.

`list_devs_local(dev_type)`

Lista os dispositivos do localhost, com valor de dev_type em {CAM, POLAR, MIC}.

`list_devs_host(dev_type, ip_addr)`

Lista os dispositivos do host de IP especificado, com valor de dev_type em {CAM, POLAR, MIC}.

`list_devs_lan(dev_type)`

Lista os dispositivos da LAN, com valor de dev_type em {CAM, POLAR, MIC}.

`handle = stream_start(dev_type, dev_id, ip_addr)`

Inicia streaming de vídeo ou de ECG do dispositivo especificada. Retorna um handle.

`stream_show(dev_type, handle, window)`

Exibe streaming de vídeo ou de ECG, com handle especificado, em window.

`stream_save(dev_type, handle, filename)`

Salva streaming de vídeo ou de ECG com handle especificado para arquivo.

2.2. Classe Routine

A classe Routine lê os arquivos que controlam a rotina de captura e exibição de um experimento. Esses arquivos estão em formato CSV, com valores separados por ponto e vírgula. A primeira coluna contém o tempo em segundos que uma ação é executada, a segunda coluna contém a ação e as demais contém os parâmetros ou operandos. Linhas iniciadas com o caractere “#” são ignoradas.

Abaixo está um exemplo de arquivo de rotina de captura.

```
#Show random slide, 7.5s
0.0;    image; all; "images/slides"
0.0;    label; block0; slide
#Show blank screen, 7.5s
7.5;    image; all; "images/Slide46.png"
7.5;    label; block1; blank0
#Flashing fixation cross, 3.0s
7.5;    label; block2; flash0
15.000; image; all; "images/Slide8.png"
15.375; image; all; "images/Slide46.png"
15.750; image; all; "images/Slide8.png"
16.125; image; all; "images/Slide46.png"
16.500; image; all; "images/Slide8.png"
16.875; image; all; "images/Slide46.png"
17.250; image; all; "images/Slide8.png"
17.625; image; all; "images/Slide46.png"
#Describe figure, 7.5s
18.000; image; all; "images/CUE_PRED.png"
18.000; label; block3; utter
18.000; message; s1; "Please describe the figure."
#Show blank screen, 7.5s
25.500; image; all; "images/Slide46.png"
25.500; label; block4; blank1
18.000; message; s1; ""
#Flashing fixation cross, 3.0s
25.500; label; block5; flash1
33.000; image; all; "images/Slide8.png"
33.375; image; all; "images/Slide46.png"
```

```

33.750; image; all; "images/Slide8.png"
34.125; image; all; "images/Slide46.png"
34.500; image; all; "images/Slide8.png"
34.875; image; all; "images/Slide46.png"
35.250; image; all; "images/Slide8.png"
35.625; image; all; "images/Slide46.png"
36.000; stop
...

```

Abaixo está um exemplo de arquivo de pausa.

```

#Interval specification
0.0;    show; all; #8F8F8F
0.0;    message; all; "Please, wait a moment."
0.0;    label; blockp; pause
120.0;  stop
...

```

Os arquivos de captura e de pausa são usados para gerar o arquivo de captura definitivo. O arquivo de captura é repetido tantas vezes quanto especificado na interface pelos valores de *Number of blocks* e *Number of repetitions per block*. Entre cada bloco há uma execução do arquivo de pausa. No arquivo de rotina definitivo, os *timestamps* precisam ser recalculados. No arquivo de captura definitivo, no lugar da pasta com as imagens, deve ser colocado o nome do arquivo de imagem sorteado a ser exibido.

Abaixo está a lista de comandos suportados. Cada comando recebe dois operandos exceto `stop`.

Instrução	Op. 1	Op. 2	Comentário
message	User	String	Show message on screen
label	Block	Condition	Label for annotation purposes
show	User	Cam	Show camera content on video canvas
clear	User	Color	Clear video canvas and fill it with color
play	User	String	Play video and show on video canvas
image	User	String	Show image on canvas
stop			End the routine

Os Valores de que User, Cam, Block e Condition podem assumir são

```

User = {s1, s2, all}
Cam = {c1, c2}
Block = {block0, ..., block5}

```

```
Condition = {slide, blank0, flash0, utter, blank1, flash1}
```

onde *s1* corresponde à janela do subject 1, *s2* corresponde à janela do subject 2, *all* corresponde a todas as janelas dos subjects, *c1* corresponde à câmera do subject 1 e *c2* corresponde à câmera do subject 2.

A string do comando *image* pode ser um nome de arquivo de imagem ou uma pasta. Se for uma pasta, seu conteúdo deve ser lido e uma de suas imagens deve ser sorteada sem reposição.

2.3. Classe WinOp

A classe WinOp cria a janela usada pelo operador do sistema. Expõe as funcionalidades dos dispositivos---câmeras, microfones e sensores cardíacos---da LAN ao usuário, permite definir a rotina de captura, selecionar os dispositivos a serem usados e iniciar a rotina de captura. Para evitar atrasos de comunicação, é definido um intervalo de tempo *t* em segundos. O comando de iniciar a rotina agenda o início da rotina nos clientes na hora atual, arredondada para um segundo inteiro, mais *t* segundos. Idealmente, todas as máquinas envolvidas devem ter seus relógios sincronizados.

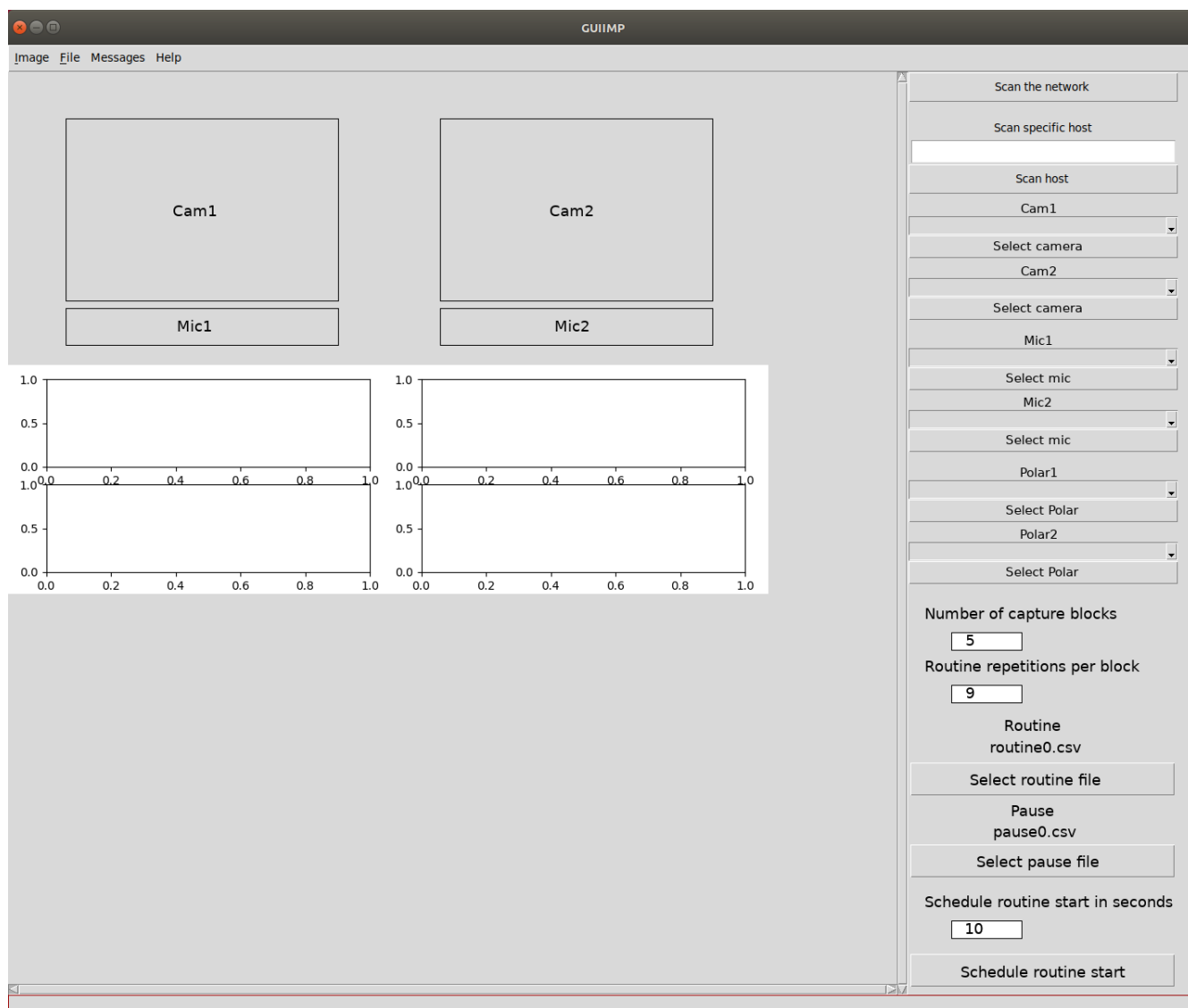


Figura 1 – Janela WinOp

A janela possui duas áreas principais. À direita está o canvas onde o conteúdo das câmeras selecionadas é exibido. À esquerda está o menu com as funcionalidades disponíveis.

O botão *Scan the network* varre a rede em busca de instâncias de servidores rodando. Os servidores são ativados ao se chamar a janela WinSub.

A caixa de texto *Scan specific host* permite definir um IP previamente conhecido. Caso haja servidores rodando nesse IP, seus *devices* são listados ao se clicar no botão *Scan host*.

Uma vez que o *host* é escaneado, seus *devices* aparecem nos menus *dropdown*. Os *devices* iniciam a transmissão ao se clicar nos respectivos botões *Select camera*, *Select mic* e *Select Polar*.

Ambos os botões *Select routine file* e *Select pause file* abrem uma caixa de diálogo

para que o operador selecione o arquivo contendo a rotina de captura ou de pausa. A rotina de captura é repetida em cada bloco tantas vezes quanto o valor especificado em *Routine repetitions per block*. O número de blocos é especificado em *Number of capture blocks*. Entre cada bloco é executado o conteúdo especificado no arquivo de pausa. O total de repetições da rotina é dado pelo produto desses dois números.

Abaixo há uma caixa de texto para que o operador selecione um intervalo de tempo t em segundos. Ao clicar no botão *Schedule routine start*, é enviada a rotina de captura e o horário atual mais t segundos que a rotina deve ser iniciada. A janela deve possuir ao menos o menu abaixo.

2.4. Classe WinSubj

A classe WinSubj cria a janela usada pelo voluntário participante do experimento. É controlada por comandos enviados por WinOp via socket, mesmo que ambos os processos estejam no mesmo host. Exibe o conteúdo de alguma câmera da rede, mostra vídeos, imagens e mensagens com instruções.



Figura 2 – Janela WinSubj

3. Arquitetura

A Figura 3 mostra as camadas do sistema. A Figura 4 mostra como as janelas são distribuídas entre os hosts e como se comunicam pela rede.

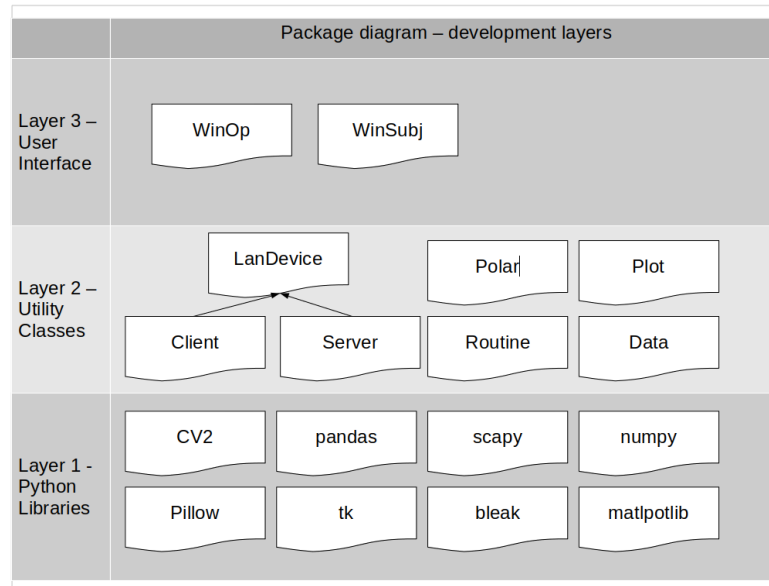


Figura 3 - Package diagram: deployment layers

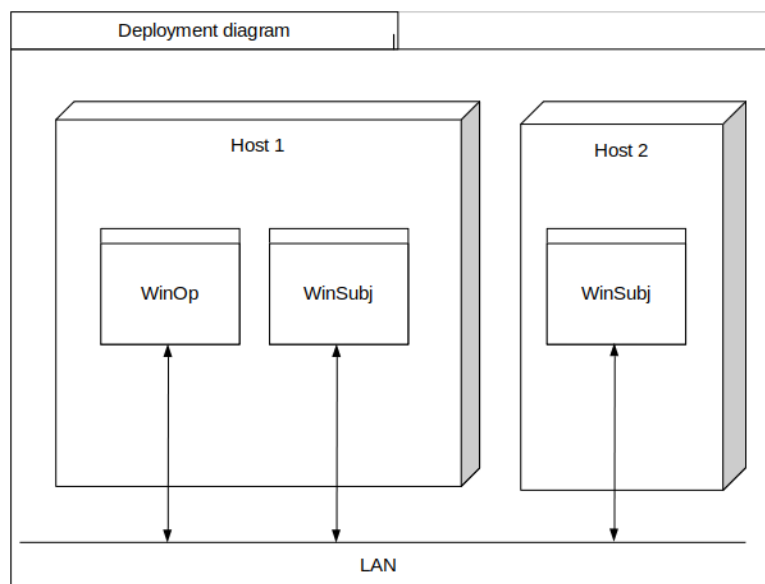


Figura 4 – Deployment diagram

4. Protocolo de comunicação

A janela WinOp envia comandos e consultas às janelas WinSubj rodando na LAN. Os comandos são

`query_servers`

Sinal de broadcast para encontrar os servidores ativos.

`query_server_cams`

Consulta as câmeras disponíveis de um certo servidor

`query_server_mics`

Consulta os microfones disponíveis de um certo servidor

`query_server_polars`

Consulta os sensores cardíacos disponíveis de um certo servidor

`select_cam`

Envia para servidor o id da câmera selecionada e atribui um rótulo em {c1, c2}. Inicia o streaming imediatamente para o canvas Cam1 ou Cam2.

`select_mic`

Envia para servidor o id do microfone selecionado e atribui um rótulo em {m1, m2}. Inicia o streaming imediatamente para o canvas Mic1 ou Mic2.

`select_polar`

Envia para servidor o id do sensor cardíaco selecionada e atribui um rótulo em {p1, p2}. Inicia o streaming imediatamente para o canvas Polar1 ou Polar2.

`schedule_routine`

Envia para servidor o arquivo com a rotina completa a ser executada e agenda o início da captura para algum horário nos próximos segundos. O arquivo de rotina é gerado a partir dos arquivos de rotina básicos e arquivo de pausa. O arquivo de rotina é repetido tantas vezes quanto necessário e deve conter o nome da imagem a ser exibida no lugar do nome da pasta que aparece como parâmetro no comando `image`.

5. Análise dos dados

Os dados de cada experimento ficam salvos na pasta `data`. Cada captura recebe um número inteiro, zero-padded, a partir de 001. Abaixo está a lista de arquivos gerados em cada captura:

<code>log.txt</code>	Log de eventos
<code>routine.txt</code>	Rotina usada no experimento
<code>subj1_ecg.tsv</code>	ECG do subject 1 em formato TSV (tab-separated values)
<code>subj1_rr.tsv</code>	HR (heart rate) e intervalo RR do subject 1 em formato TSV
<code>subj1.mp4</code>	Vídeo do subject 1
<code>subj2_ecg.tsv</code>	ECG do subject 2 em formato TSV (tab-separated values)
<code>subj2_rr.tsv</code>	HR (heart rate) e intervalo RR do subject 2 em formato TSV
<code>subj2.mp4</code>	Vídeo do subject 2

O par de vídeos é anotado usando o software ELAN, que salva a anotação em formato XML. Na anotação serão considerados os rótulos abaixo:

<code>IsSync</code>	Indica se há ou não sincronia. {F/NSync, T/Sync}
<code>IsImit</code>	Indica se há ou não imitação. {F/NIm, T/Im}
<code>Imitator</code>	Indica que sujeito imita nos períodos onde <code>IsImit = 1</code> .
<code>Model</code>	Indica que sujeito é imitado nos períodos onde <code>IsImit = 1</code> .
	Ambos assumem valores no conjunto {1, 2}

O primeiro passo da análise é um pré-processamento dos arquivos de ECG, RR, anotação em XML e rotina para gerar um arquivo em formato TSV contendo uma linha para cada segundo de captura:

Cada linha contém

- Índice inteiro do tempo em segundos
- Rótulo obtido dos comandos `label` do arquivo `routine.txt`
- HR interpolado a partir dos arquivos `subj*_rr.tsv`
- HR estimado a partir dos intervalos RR dos arquivos `subj*_ecg.tsv`
- Rótulos obtidos da anotação feita no ELAN obtidos do arquivo XML: `IsSync`, `IsImit`, `Imitator`, `Model`.

A exibição em gráfico dos valores de HR deve ser analisada em busca de erros de sincronia.

References

[1] Dikker S, Silbert LJ, Hasson U, Zevin JD. On the same wavelength: predictable language enhances speaker-listener brain-to-brain synchrony in posterior superior temporal gyrus. *J Neurosci*. 2014 Apr 30;34(18):6267-72. doi: 10.1523/JNEUROSCI.3796-13.2014. PMID: 24790197; PMCID: PMC4004812.