Manual Técnico

Microscópio Remoto

Experimentação Remota Móvel para o Ensino Básico e Superior

rexlab
LABORATÓRIO DE EXPERIMENTAÇÃO REMOTA

UNIVERSIDADE FEDERAL DE SANTA CATARINA

RNP

CAPES

Manual Técnico do Experimento Microscópio Remoto:
Experimentação Remota Móvel para a Educação Básica e Superior
Este guia, cada capítulo e suas imagens estão licenciados sob a licença Creative Commons
Rua Pedro João Pereira, 150, Mato Alto – CEP 88900-000
http://rexlab.ufsc.br/
rexlabufsc@gmail.com

Elaboração
Juarez Bento da Silva
João Paulo Cardoso de Lima
José Pedro Schardosim Simão
Josiel Pereira
Lucas Mellos Carlos
Editoria de arte, projeto gráfico e capa
Isabela Nardi da Silva
Ilustrações
Alex Moretti

Manual Técnico do Experimento Microscópio Remoto: Experimentação Remota Móvel para a Educação Básica e Superior / obra coletiva concebida, desenvolvida e produzida pelo Laboratório de Experimentação Remota (RExLab)

Araranguá - SC, Brasil, 2016

Experimento Microscópio Remoto

Apresentação

O experimento remoto Microscópio tem como objetivo auxiliar os estudantes do Ensino Fundamental e do Ensino Médio a efetuar práticas relacionadas às amostras que estão disponíveis no microscópio remoto. O estudo do reino das plantas é um exemplo de conteúdo, pois o microscópio proporciona a visualização das partes de uma planta (raiz, folha, caule, fruto, semente e flor).

Este experimento foi desenvolvido no âmbito do O GT-MRE (Grupo de Trabalho em Experimentação Remota Móvel), com o objetivo de ser disponibilizado o seu acesso por meio de dispositivos móveis e dispositivos convencionais.

Arquitetura

O dispositivo está implementado a partir da estrutura padrão de hardware e software básico. Na Figura 1 pode ser visualizado o diagrama de arquitetura do experimento Microscópio Remoto.

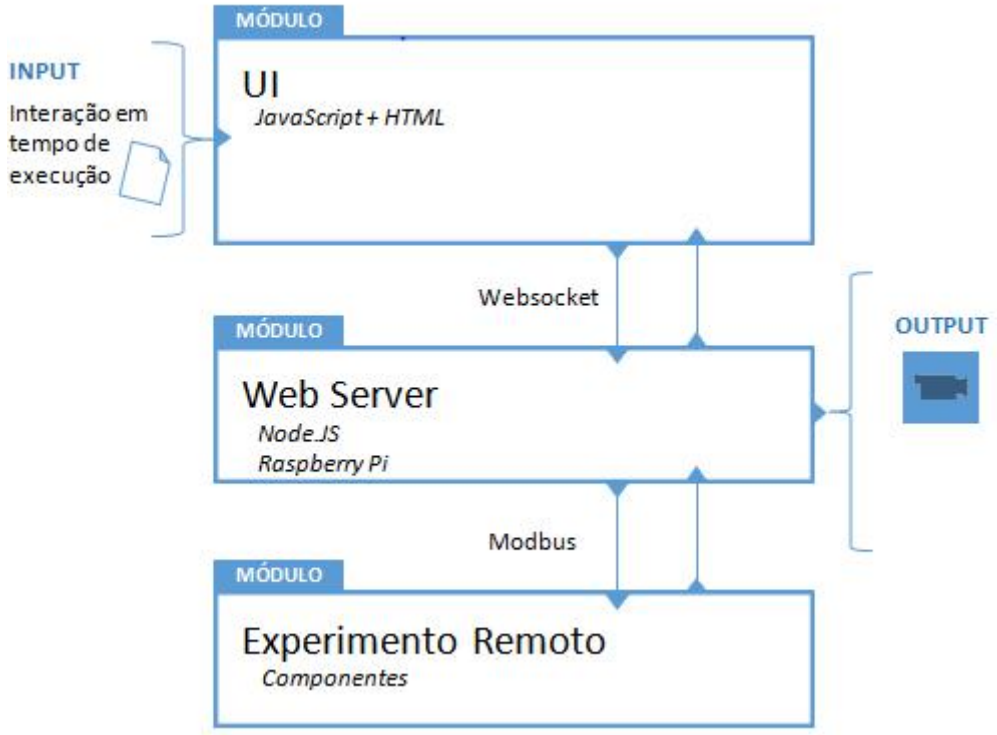

Figura 1 - Arquitetura Experimento: Microscópio.

Interface de usuário(UI)

O experimento está disponível no sistema de gerenciamento RELLE(Remote Labs Learning Enviromment), que provê uma série de funcionalidades necessárias para o gerenciamento de experimentos remotos.

A interface de acesso ao experimento foi desenvolvida utilizando HTML juntamente com o framework front-endBootstrap, o mesmo traz uma série de componentes prontos para o desenvolvimento além de prover tratamento para diferente tipos de resoluções de telas. Além de HTML e Bootstrap, é utilizada a biblioteca jQuery que traz uma série de funções JavaScript que simplificam o desenvolvimento.

A Figura 2 e a Figura 3 apresentam a tela de início e a interface com o usuário, respectivamente.

Figura 2– Página do experimento

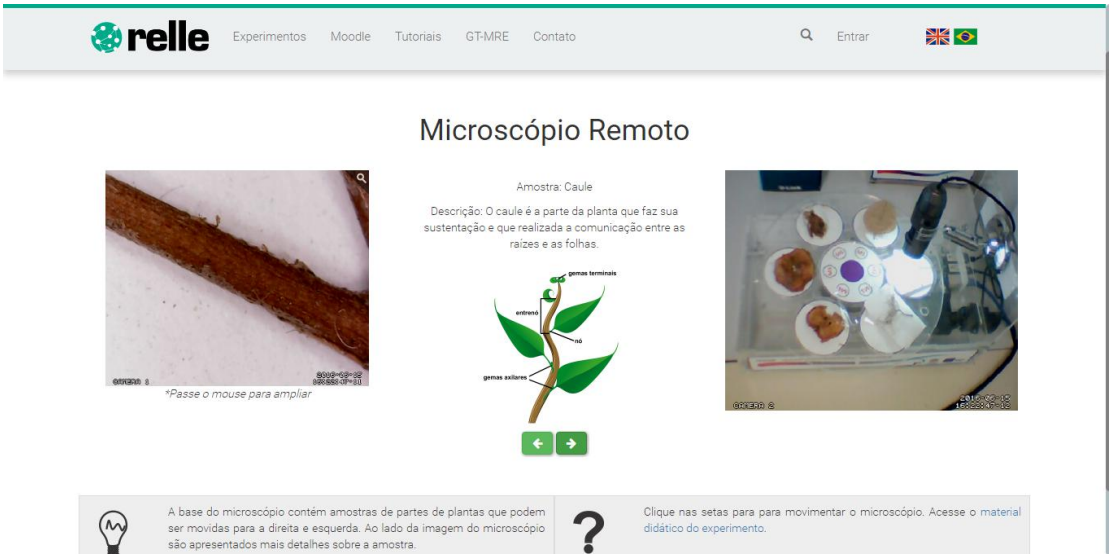

Figura 3 - Interface com usuário

Web Server

Atualmente, há uma ampla gama de bibliotecas e frameworks para construção de serviços web. Apesar de serviços baseados em HTTP predominarem a Internet, o uso do protocolo WebSocket é uma tendência em aplicações corporativas de grande porte. Uma das plataformas para desenvolvimento web para construção de serviços baseados em WebSocket é o framework NodeJS.

O NodeJS permite construir aplicações de servidor e de rede facilmente escaláveis. Ele é composto por um ambiente de execução multiplataforma e de código fonte aberto que interpreta códigos de aplicações escritas em Javascript. O NodeJS usa um modelo orientado a evento, com operações de entrada e saída não bloqueantes. Por este motivo, ele é ideal para aplicações em tempo real com troca intensa de dados entre dispositivos distribuídos.

A API para acesso às funcionalidades do SmartDevice contém funções vinculadas à *listeners*, comuns ao paradigma de orientação a eventos. Este módulo usa a biblioteca Socket.io e é o ponto de partida da aplicação, onde o servidor é iniciado e eventos são vinculados. O Socket.io é composto por dois componentes: servidor e cliente, ao qual usa principalmente o protocolo WebSocket, e polling HTTP como compatibilidade reversa.

A autorização de sessão no SmartDevice garante a integridade do acesso exclusivo, já que o dispositivo exposto como um serviço pode ser utilizado concorrentemente por outro cliente. Apesar de algumas funcionalidades poderem ser utilizadas no modo observador, como consultar o estado das chaves e metadados, as funcionalidades de controle necessitam de consulta ao sistema de fila.

O sistema de fila, ou mesmo agendamento, pode ser externo ou interno ao SmartDevice. O primeiro é baseado em um token de autenticação provido pelo usuário e validado pelo SmartDevice. As implementações dos experimentos de física exemplificam o uso do sistema de reserva externo (próprio do Relle). Já o controle de acesso no próprio SmartDevice é exemplificado pela implementação do Laboratório de desenvolvimento em Arduíno, pois neste encontra-se um modelo de acesso diferente dos anteriores.

O código fonte desenvolvido para comunicação serial e gerência dos sensores e atuadores são complementos para o NodeJS escritos em C++. Estes complementos são objetos compartilhados de vínculo dinâmico que pretendem dar suporte a códigos nativos, rapidez e portabilidade. Esses objetos compõem a abstração de cada experimento físico, que é representado por métodos e atributos intrínsecos a cada um. Por exemplo, são definidos os métodos de “get” e “set” para saídas digitais, “get” para valores de sensores, “get” e “set” para calibragem e configuração dos sensores.

O dispositivo central do experimento é o servidor de laboratório, que na plataforma desenvolvida pelo GT-MRE a escolha recaiu sobre o RaspberryPi[1], (Figura 4) modelo B+, que tem como principal função intermediar os acessos aos demais dispositivos de hardware dos experimentos via rede.

O servidor de laboratório (SL) tem função prover interfaceamento e gerenciamento para a conexão entre a rede (web) e a “placa de aquisição e controle” (PAC). O SL acessa a PAC para coletar os dados dos sensores ou para enviar comandos para os atuadores, essa comunicação é feita via porta

[1] O RaspberryPi é um computador é baseado em um system on a chip (SoC) Broadcom BCM2835, que inclui um processador ARM1176JZFS rodando a 700 MHz, GPU VideoCore IV, e 512 MB de memória RAM em sua última revisão. O Raspeberry PI foi desenvolvido no Reino Unido pela Fundação RaspberryPi.

UART(Universal asynchronous Receiver/Transmitter) que se comunica via protocolo MODBUS[2].

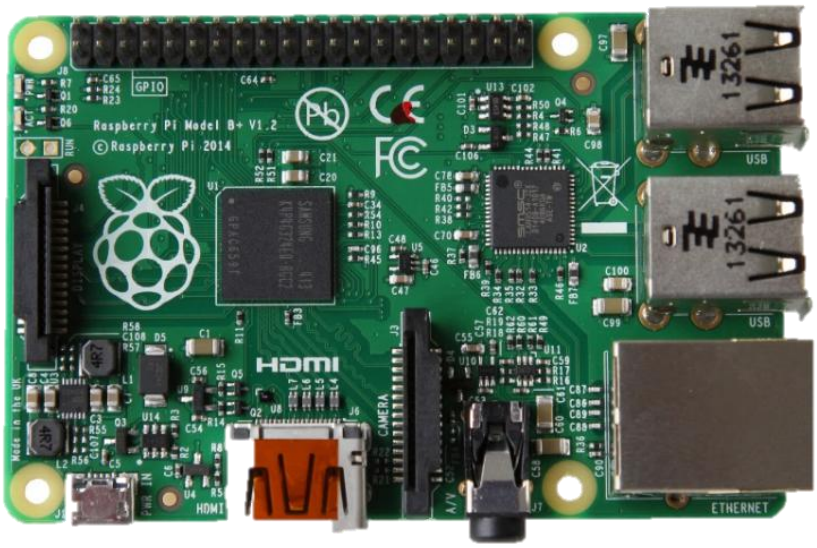

Figura 4 - Raspberry Pi, Model B+

API WebSocket

Os componentes da aplicação são suficientemente leves para serem executados por uma placa RaspberryPi ou outro computador Linux de baixo custo. Um dos componentes, a API WebSocket,oferece uma interface aos sensores e atuadores na estrutura de um serviço web. A aplicação não requer alto uso da memória e pode ser utilizada em qualquer sistema Linux.

O resultado é uma arquitetura fracamente acoplada, adotada pelo GT-MRE, que habilita o compartilhamento dos experimentos em outras plataformas, como as demais do nosso laboratório. Esse paradigma, chamado de SmartDevices[3] já é utilizado no projeto Go-Lab[4], no qual estão bem destacadas aplicações clientes e servidor, e fornecem interfaces bem definidas entre o usuário e o sistema.

---

[2]Modbus é um protocolo de comunicação de dados utilizado em sistemas de automação industrial. É um dos protocolos mais utilizados em redes de Controladores lógicos programáveis (PLC) para aquisição de sinais (0 ou 1) de instrumentos e comandar atuadores. É de utilização livre e sem taxas de licenciamento.
[3] DOI: 10.1109/REV.2015.7087292
[4]http://www.go-lab-project.eu/

Os tópicos seguintes apresentam com mais detalhes aspectos do serviço web utilizado no servidor de experimento, bem como as funcionalidades internas e as motivações para o uso de certos protocolos, padrões e ferramentas de desenvolvimento, conforme a Figura 5.

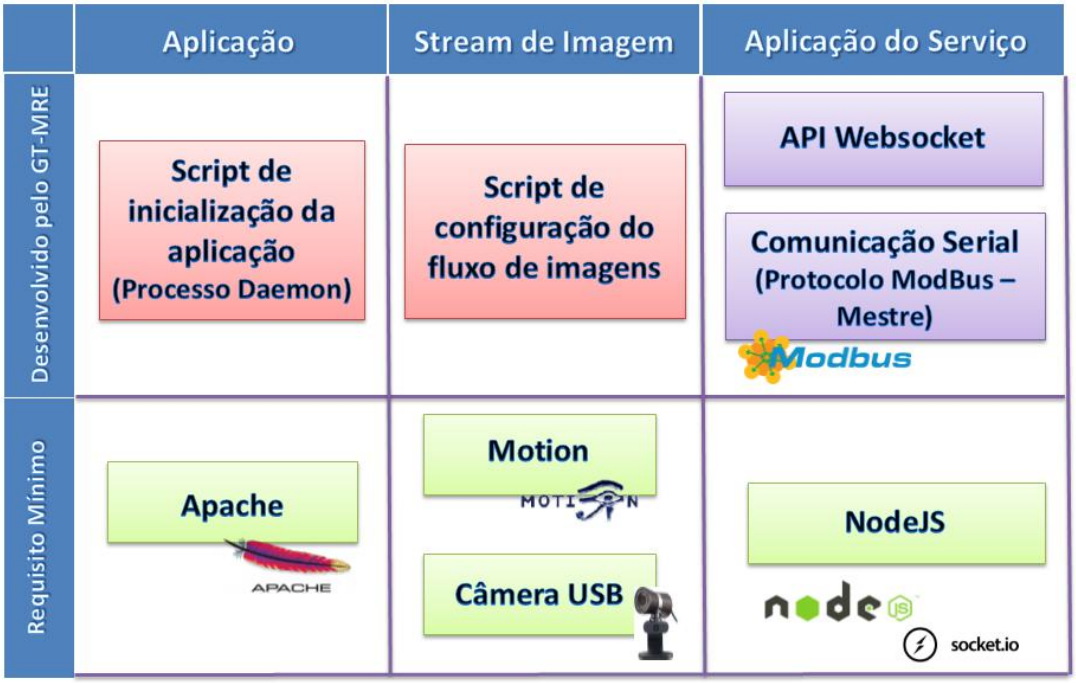

Figura 5 - Esquema de aplicação embarcada. Fonte: GT-MRE.

Controle e monitoramento do experimento

O SmartDevice é capaz de comunicar-se com sensores através do barramento serial (Porta UART). Ao invés de usar o protocolo serial em sua forma bruta, optamos por incluir o protocolo Modbus na camada de aplicação para identificação de erros, endereçamento e controle de colisão. Conectados ao mesmo barramento (rede), cada sistema embarcado, responsável por um ou mais sensores ou atuadores, é um dispositivo escravo que responde às requisições da aplicação que é executada no Raspberry Pi.

Um dos módulos desenvolvidos para aplicação é responsável pelo serviço de fila externo ou interno, sendo possível acoplar o serviço de fila provido pelo RELLE ou habilitar serviços internos. No primeiro caso, a aplicação usa a lógica necessária para validação de token de sessão enviado

pelo cliente. Na segunda, todo processo realizado pela web API de fila é realizado pelo SmartDevice.

Acesso à web API pelo cliente

A Figura 6 apresenta o esquema de comunicação no uso da API desenvolvido para o serviço/protótipo.

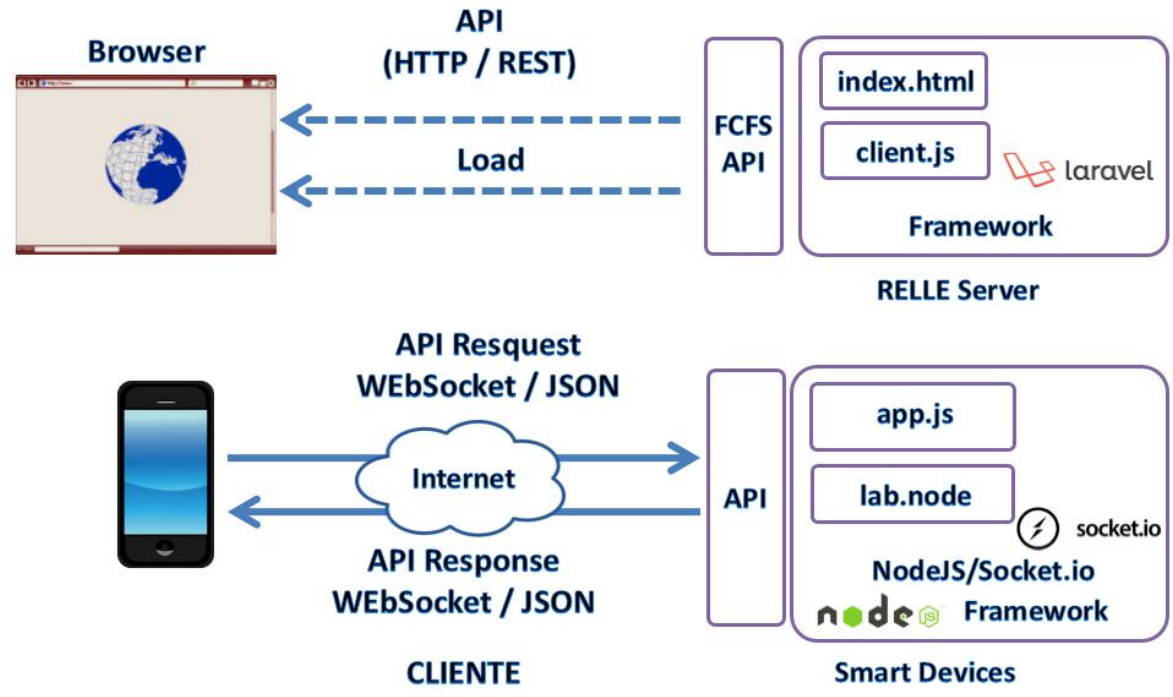

Figura 6 - Esquema de comunicação crossdomain no uso da API desenvolvida pelo GT-MRE.

O cliente web disponibilizado pelo sistema RELLE é composto por um arquivo html, css e javascript diferentes para cada experimento. O RELLE provê uma página comum para cada experimento onde carrega os dados que foram inseridos no momento da publicação do experimento (armazenados numa base de dados). Por exemplo, o experimento de ID 1 é acessível pela URL “relle.ufsc.br/labs/1” pelo método GET e contém suas informações dentro do layout padrão do sistema. A partir do botão “Acessar” é possível disparar um evento para comunicação com a Web API FCFS (first-come firstserved).

Ao obter a permissão no navegador, o cliente navegador poderá carregar os arquivos (html, css e js), pois a API já tem o seu token de sessão como usuário sendo servido. Após carregar o cliente para o SmartDevice (client.js), uma conexão WebSocket com este dispositivo é estabelecida.

Streaming de imagens

No GT-MRE foi optado pelo uso de câmeras web com conexão USB devido ao baixo custo e a facilidade de aquisição. O mesmo computador embarcado utilizado para controle do experimento também é o responsável pelo gerenciamento e disponibilização do streaming no formato MJPEG (Motion JPEG). O MJPEG é um formato de compressão de vídeo na qual cada frame de vídeo é comprimido separadamente como uma imagem JPEG.

Visto que existem muitos servidores de streaming de código aberto, optou-se pelo Motion para explorar aspectos de leveza (utilização de poucos recursos) e configuração flexível. O Motion[5] é um software escrito em C para sistemas Linux que usa a API de vídeo Linux, e é capaz de detectar se uma parte significante da imagem tem mudado. Algumas variáveis são ajustadas através de seu arquivo de configuração principal para adequar-se aos requisitos de nossa aplicação.

Atualmente, os principais navegadores do mercado como Firefox, Google Chrome e Safari já possuem o suporte nativo para o streaming MJPEG. Para clientes Android existem bibliotecas de código fonte aberto para incluir um visualizador MJPEG em aplicações de código nativo.

[5]http://www.lavrsen.dk/foswiki/bin/view/Motion/WebHome

Experimento Remoto

O experimento microscópio remoto é composto por um microscópio e uma base em acrílico. A base acrílica é composta por seis “slots” de amostras para serem analisadas e sua base é movimentada por um servo motor. Na Figura 7 é ilustrado o diagrama dos componentes do experimento:

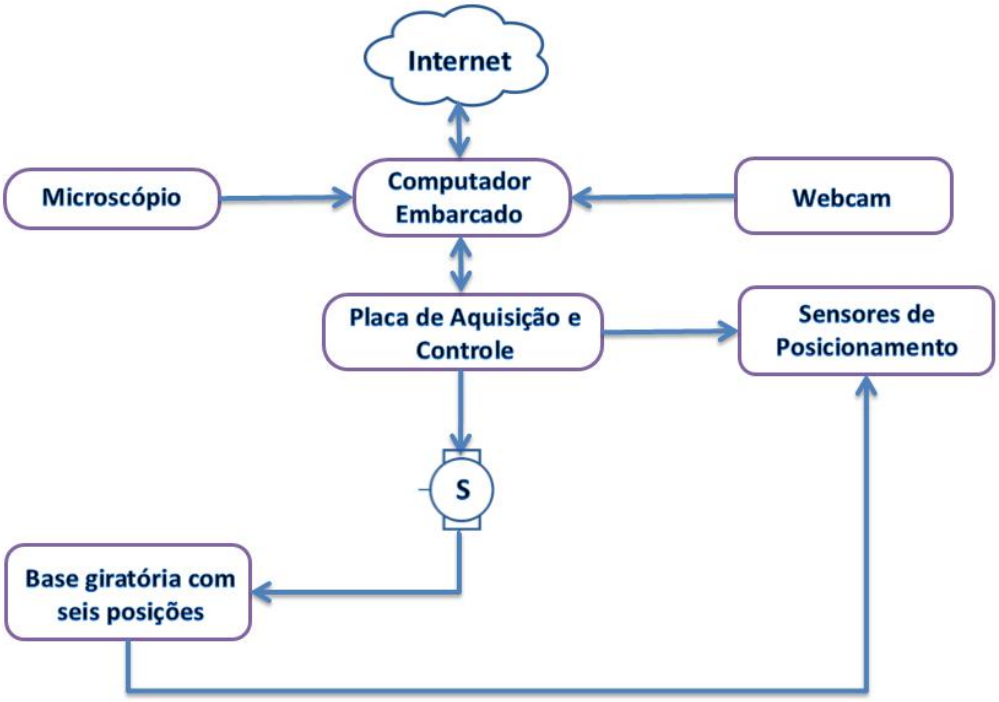

Figura 7 - Diagrama de blocos do experimento

A Figura 8 apresenta as principais partes que compõe o experimento remoto “Microscópio Remoto”. São elas:

1. Computador embarcado;
2. Placa de Aquisição e Controle;
3. Microscópio Digital;
4. Slot;
5. Reed Switch;
6. Base giratória;
7. Motor servo DC.

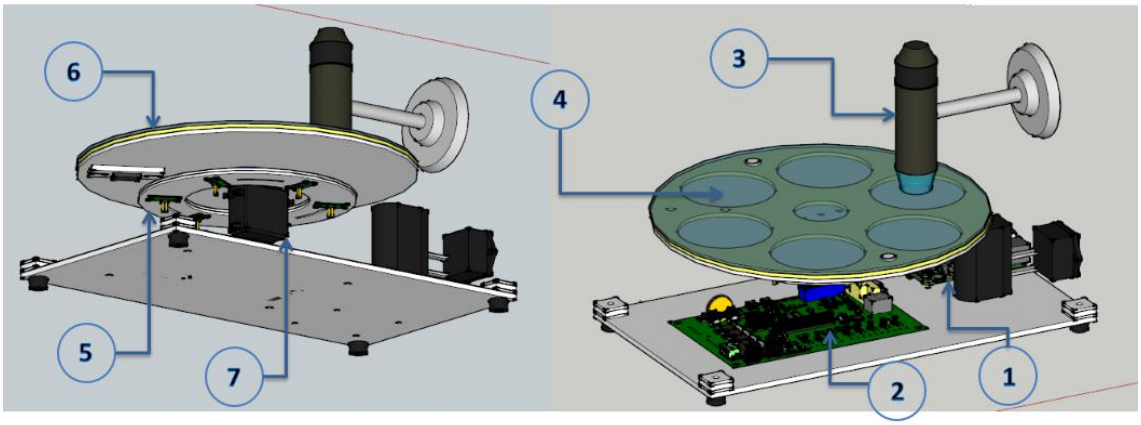

Figura 8 - Componentes Microscópio Remoto

Microscópio

Microscópio digital USB, de baixo custo, com funcionalidade de zoom para permitir visualização mais apurada das matérias em análise.

Base Giratória com seis Slots

Consiste de uma bandeja com seis espaços para armazenamento de amostras para análise, espaços os quais são denominados Slots.

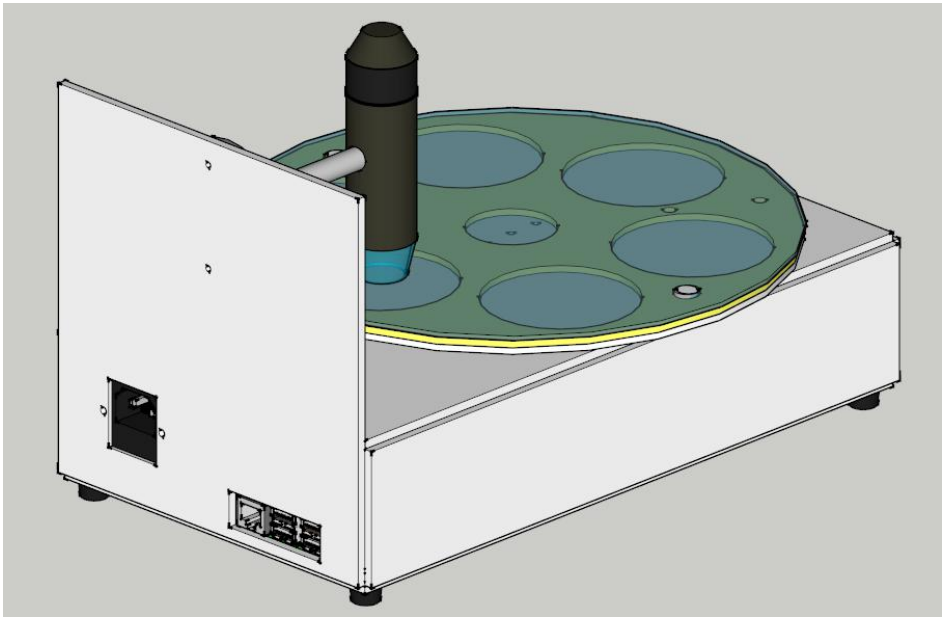

Figura 9 - Vista lateral do experimento

Servo Motor

É um motor DC, do tipo, servo que possui como funcionalidade girar a base para que o slot desejado fique sob a lente do microscópio.

Figura 10 - Servo motor DC

Sensores de Posicionamento

São sensores para indicar à “Placa de Aquisição e Controle” o posicionamento dos slots, em relação ao microscópio. Os sensores utilizados são do tipo “reed switches”. Estes são dispositivos que funcionam como interruptores, acionados por campos magnéticos produzidos por ímãs ou eletroímãs dele aproximados.

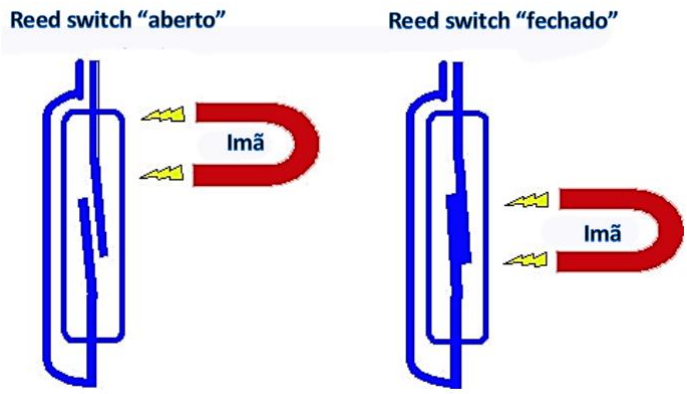

Figura 11 - Funcionamento do reed switch

Reed Switch é uma chave que é acionada por aproximação a um campo magnético. O reed switch é composto de duas aletas internas dentro de um

bulbo de vidro com gás inerte para evitar a oxidação. As aletas são formadas por material ferroso e quando expostas a um campo magnético as mesmas se comportam como imas e se atraem até se encostarem e com isso fechando o circuito. Há muito tipos de configurações de reed switch, mas esta é a mais comum e o funcionamento é semelhante para as outras.

Placa de Aquisição de Controle

É um sistema embarcado dedicado ao controle sistemático e de temporização dos elementos que compõe o experimento, tais como leitura de sensores e os drivers dos atuadores. O hardware desenvolvido é responsável pela leitura dos comandos recebidos e enviados do “Computador Embarcado”, via MODBUS, posicionando o slot desejado sob o microscópio para ser analisado pelo usuário na WEB.

A “placa de aquisição e controle”, do projeto (Figura 12 e Figura 13) tem como objetivo controlar os sensores e atuadores dos experimentos. Esta placa é baseada no processador ARM PLC1752 Cortex-M3 rodando a uma frequência de clock de 80 Mhz. A opção por este modelo de ARM deve-se ao fato de sua disponibilidade e relação custo/benefício.

O LPC1752 é um microcontrolador baseado na arquitetura ARM Cortex-M3 que opera com frequências de até 100 MHz. Este tipo de microcontrolador é indicado para aplicações embarcadas, uma vez que, são caracterizados por um alto nível de integração e baixo consumo de energia. O LPC1752 incorpora uma “pipeline” de três estágios e conta com barramentos de dados e instruções separados, inspirados na arquitetura Harvard. Também inclui 64KB de memória flash, 16KB de memória de dados (SRAM), 53 pinos de I/O, interfaces USB, UART (4), I²C (3), SPI (2), além de ADC (6), Timers (6) e PWM(6). O LPC1752 opera com tensão de 3,3V.

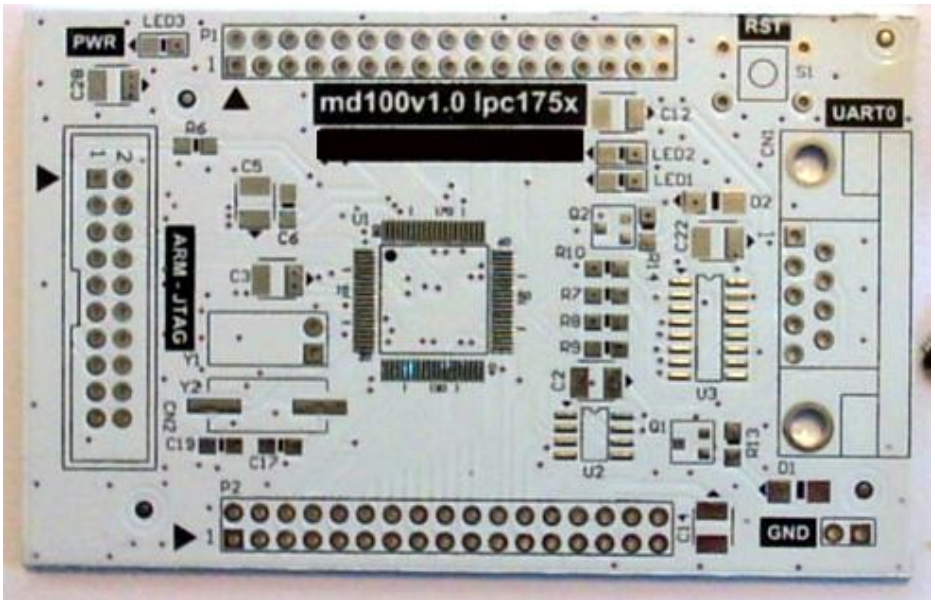

Figura 12 - Placa de aquisição e controle - Construção

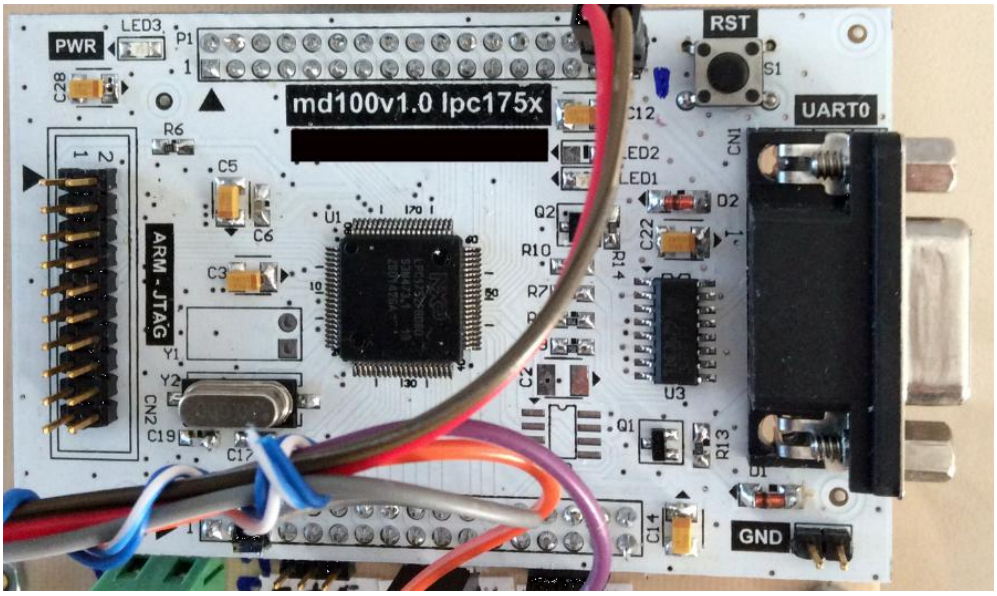

Figura 13 - Placa de aquisição e controle - Montagem

Apêndices

Mapa dos Registradores ModBus do Recurso de Hardware

Todos os registradores são de 16 bits e os endereçamentos mostrados neste documento são em hexadecimais. Não é permito ler ou escrever mais que 120 registradores em uma só transação, isto devido à taxa do baudrate do aparelho.

Registradores de Identificação		
Endereço	**Tipo de Acesso**	**Descrição**
0x0	Leitura	Identificador do modelo do aparelho. Valor em ASCII
0x1	Leitura	Versão do modelo do aparelho. Valor em ASCII
0x2	Leitura	Versão do firmware. Valor em ASCII. O formato da versão é x.y, porém o será enviados em o ponto decimal. Exemplo: versão 1.0 será transmitido 10.

Registradores de Trabalho		
Endereço	**Tipo de Acesso**	**Descrição**
0x10	Leitura	Sinaliza se está ou não um slot sendo posicionado. Valor 0 a bandeja está parada, valor 1 sinaliza que a bandeja está em movimento.
0x11	Leitura	Contém o valor do slot que a bandeja está posicionada ou está sendo posicionada. Valor de 1 a 6 representando os slots de 1 a 6.
0x12	Leitura	Retorna com o valor das chaves de posições. Valores: – Chave 1 valor 1; – Chave 2 valor 2; – Chave 3 valor 4; – Chave 4 valor 8; – Chave 5 valor 16; – Chave 6 valor 32.

0x13	Escrita	Instrui ao recurso de hardware qual slot da bandeja deve ser posicionado sob o microscópio. Este comando controla o servo motor e os sensores de posições para que o slot seja posicionado no local correto.
0x14	Escrita	Controle do servo motor. O valor emitido deve ser em microssegundos (ms). Este comando é útil para controlar qualquer servo motor que não seja necessariamente o servo do experimento. Ver tabela abaixo a relação de tempo em ms com posicionamento do servo.

Tabela de posicionamento de servos:

Tempos (ms)	**Servo normal**	**Servo de modo continuo**
1000	0º	Máxima velocidade sentido anti-horário
1250	45º	Média velocidade sentido anti-horário
1500	90º	Parado
1750	135º	Média velocidade sentido horário
2000	180º	Máxima velocidade sentido horário

Terminal

O Recurso de Hardware oferece um terminal para emitir comandos semelhantes ao prompt do DOS. Este terminal é acessado via porta UART a 115200 bps. Digite “help” no prompt para ver os comandos disponíveis.

Tutorial de reinicialização do experimento

Para reiniciar o experimento usa-se um terminal para conexão ssh, por exemplo o software PuTTY, o qual pode ser baixado pelo seguinte endereço:

http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html. Utilizando o PuTTY, basta inserir o endereço IP do experimento que se deseja reinicializar.

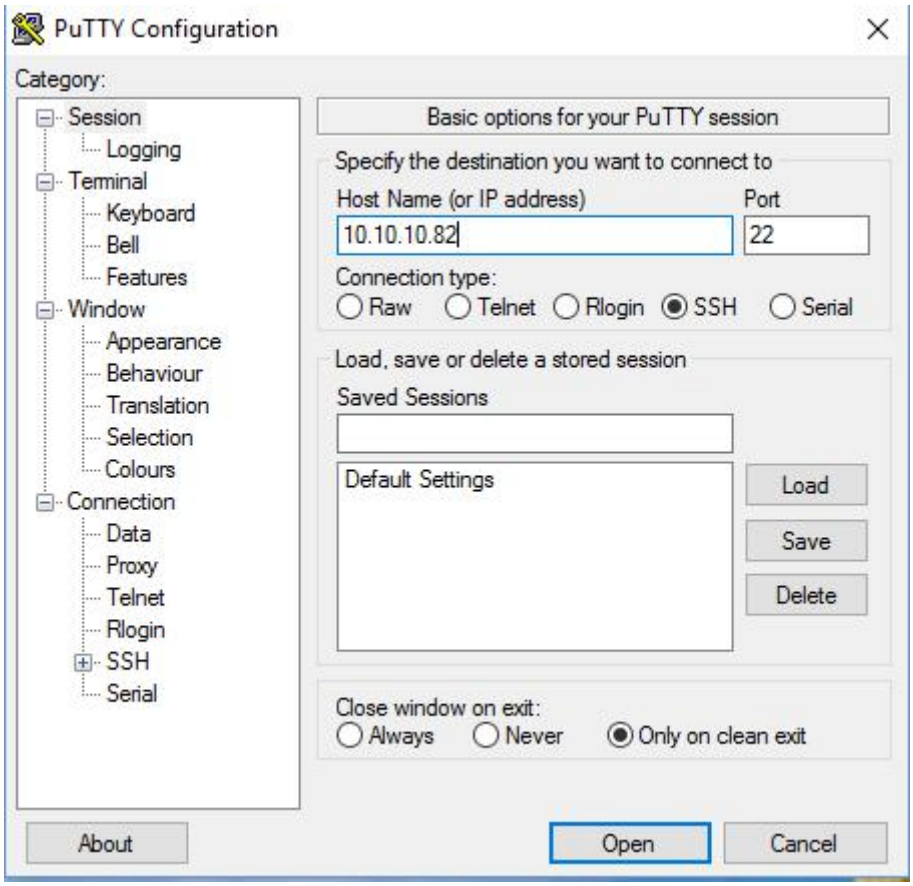

Figura 14 – PuTTY

Ao abrir a conexão será inicializado um terminal (Figura 15), onde será solicitado um usuário (user) para autenticação, recomenda-se autenticar com o usuário root, logo em seguida será requisitada a senha do computador embarcado. E por fim, para reiniciar o computador embarcado, digite o comando *reboot* no terminal.

Figura 15 - Terminal SSH com experimento

Verificação e reinicio do serviço

Para verificar os se os serviços do laboratório remoto estão rodando, basta usar o comando "*ps –aux | grep node"* que ira verificar os processos rodando referente ao servidor web Node.JS responsável por executar o serviço da aplicação. Caso o serviço esteja rodando, o resultado será algo similar a Figura 16 que exibe o usuário e número do processo em execução. Neste caso o processo PID 2434.

```
root@raspberrypi:~# ps -aux | grep node
warning: bad ps syntax, perhaps a bogus '-'?
See http://gitorious.org/procps/procps/blobs/master/Documentation/FAQ
root      2434  0.1  9.7 1182324 43412 ?       Sl   May12 102:36 /usr/local/bin/node /home/conducao_app/apps.js
root     21479  0.0  0.3   3520  1740 pts/0    S+   14:17   0:00 grep node
```

Figura 16 - Verificação do serviço

Ações de iniciar, pausar ou verificar status do serviço podem ser executadas usando os comandos *"service dc app start|stop|status".*

Manutenção do streaming de vídeo

O vídeo é transmitido pelo software Motion. Para instalação do software pode-se fazer seu download via repositório através do comando *"apt-getinstall motion"* e acessar os arquivos de configurações *motion* e *motion.conf* através

de algum editor de código no diretório */etc/default/motion* definindo o parâmetro *start_motion_deamon* para o valor *yes.*

As configurações relacionadas à qualidade da imagem e a transmissão ficam disponíveis no arquivo motion.conf no diretório /etc/motion/. Ainda para inicio da transmissão os parâmetros *deamon* e *webcam_localhost* devem ser mudados para *on*e *off*, respectivamente.