Ministério da Educação
Universidade Federal de Santa Maria
Centro de Tecnologia

RELATÓRIO FINAL DO PROJETO DE PESQUISA

Relatório final de atividades desenvolvidas no projeto sistema de controle veicular com plataformas Arduino e Android

Coordenadora:

Marcia Pasin

Executor:

Anthony Tailer Ribas de Almeida

Santa Maria, Dezembro de 2016.

SUMÁRIO

1. INTRODUÇÃO

Cada vez mais populares, sistemas automatizados servem como ferramentas que auxiliam tarefas mecânicas que anteriormente eram feitas manualmente, e muitas vezes representavam um trabalho cansativo e repetitivo e sujeito a erros. Entre os sistemas automatizados, merece destaque a plataforma robótica. A robótica é um ramo educacional e tecnológico que engloba computadores, robôs e computação, e proporciona a elaboração de diversos sistemas compostos por partes mecânicas automáticas controladas por circuitos integrados, onde seu uso atualmente ultrapassa os limites industriais.

Ainda, neste contexto, as práticas open-source e ferramentas atuais permitem acesso ao conhecimento de como produzir projetos sem necessariamente ter um profundo conhecimento sobre robótica ou computação.

A ideia base deste projeto é realizar o controle veicular de um carro desenvolvido com a plataforma Arduino. A primeira placa foi fabricada na Itália em 2005 e, desde então, é utilizada como plataforma de prototipagem eletrônica que torna a robótica mais acessível a todos, se baseando em hardware e software flexível de fácil uso sobre uma biblioteca que simplifica a escrita da programação em C/C++. Um Arduino pode usar sensores para capturar o estado do ambiente que o cerca por meio da recepção de sinais e pode interagir com os seus arredores, controlando luzes, motores e outros atuadores.

No contexto deste trabalho, o carro desenvolvido com esta plataforma será controlado por meio de um programa para a plataforma Android através de um módulo de comunicação Wi-Fi. A aplicação desenvolvida em linguagem de programação Android envia sinais de um dispositivo que execute o sistema Android, como um *smartphone*, através de botões direcionais específicos e retorna comandos lidos pela plataforma Arduino fazendo com que o carro se movimente para ambas as direções.

Para a realização deste projeto, além da programação em linguagem Android é necessária a construção de um algoritmo que ofereça tratamento dos componentes ligados ao Arduino e execute ações sobre as opções mantidas na aplicação Android.

As motivações da escolha do tema deste projeto se baseiam nas abrangências que o estudo destas duas áreas da computação proporcionam. A possibilidade de interação entre um hardware de fácil obtenção, juntamente com um software desenvolvido para a realização da comunicação de todos os componentes com suas respectivas funções.

Desta forma o presente relatório busca trazer os resultados obtidos através das pesquisas realizadas e atividades desenvolvidas no presente projeto de pesquisa. Os resultados demonstrados neste relatório visam exemplificar e ilustrar o funcionamento de componentes de hardware e software que foram utilizados neste projeto bem como analisar estes resultados, destacando problemas enfrentados, possíveis soluções e demais recomendações.

Ao final deste relatório estão os links que cedem acesso aos repositórios dos materiais desenvolvidos durante a pesquisa, em um repositório público de livre acesso.

2. REFERENCIAL TEÓRICO

SISTEMA DE AUTOMAÇÃO VEICULAR COM ARDUINO E ANDROID, Trabalho de Conclusão de Curso de **Wagner Rocha Barros**, Centro Universitário Adventista de São Paulo, curso de Tecnólogo em Sistemas para Internet, sob orientação do Prof. Me.Thales de Társis Cezare, Disponível em: www.unasp-ec.com/sistemas/admin/upload/1411505628tcc.pdf

SISTEMA BLUETOOTH PARA CONTROLE DE ACESSÓRIOS VEICULARES UTILIZANDO SMARTPHONE COM ANDROID, Trabalho de Conclusão de Curso de **Bruno Pereira Passos**, Centro Universitário de Brasília (UniCEUB), Curso de Engenharia de Computação. sob orientação de: M.C.Maria Marony Sousa Farias, Disponível em: http://www.repositorio.uniceub.br/bitstream/123456789/3132/2/20614551.pdf

AUTOMAÇÃO VEICULAR ATRAVÉS DE UMA INTERAÇÃO ENTRE ARDUINO E SISTEMA ANDROID, Artigo acadêmico, Universidade Paranaense (Unipar), **Ademir Conessa Arroyo**, **Ricardo Ribeiro Rufino**, disponível em: http://web.unipar.br/~seinpar/2015/_include/artigos/Ademir_Conessa_Arroyo.pdf

3. METODOLOGIA

- Levantar funcionalidades gerais que o sistema deverá atender;
- Estudar e compreender a linguagem Android bem como bibliotecas específicas para interação com o *hardware* do dispositivo;
- Desenvolver a aplicação Android com base nas predefinições;
- Estudar e compreender a linguagem para a plataforma Arduino bem como as bibliotecas específicas de interação entre a placa e o dispositivo Android;
- Desenvolver o algoritmo que será responsável por ouvir os comandos mandados do Android para o Arduino bem como os comandos que serão executados sobre o *hardware* do mesmo;
- Realizar a integração dos algoritmos desenvolvidos com o carro;
- Realizar todos os testes necessários para o aprimoramento de possíveis funções;

4. APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

A proposta inicial fazia-se em que uma aplicação Android enviasse uma requisição, para o módulo Wi-Fi (ESP8266) utilizado como intermediário entre a comunicação Arduino Android, desta forma a requisição seria enviada do Android até o módulo ESP8266, que a encaminharia via conexão Wi-Fi, conectando o módulo e o dispositivo Android a uma rede sem fio existente, para o Arduino, onde seria processada pela Serial do Arduino e utilizada para movimentar os motores interligados ao carro. Os componentes eletrônicos básicos envolvidos neste processo são:

- 1x Arduino Mega;
- 4x Motores DC 5V
- 2x L293D
- 1x Módulo Wi-Fi ESP8266-01;

Cada componente envolvido neste processo possui uma especificação própria, a qual precisou ser estudada para que pudesse ser compreendida e utilizada.

4.1 COMPONENTES ELETRÔNICOS ENVOLVIDOS

4.1.1 Arduino Mega

O Arduino Mega é uma plataforma de prototipagem eletrônica open-source, e é baseada na flexibilidade do hardware e na facilidade de uso por meio de software. Seu hardware contém um Microcontrolador baseado no processador ATmega2560, com 54 portas digitais de entrada/saída de dados, 16 portas analógicas usadas para manipular sensores e atuadores diversos. Seu funcionamento se baseia em manipular os sinais eletrônicos por estas portas de comunicação via linguagem de programação C/C++ utilizada com uma IDE específica para este fim. Mais sobre Arduino em: https://www.arduino.cc/en/Main/ArduinoBoardMega2560. No contexto deste projeto o Arduino seria utilizada para receber os dados vindos do módulo ESP8266.

4.1.2 L293D

O componente L293D é responsável por controlar os motores DC que estariam ligados as rodas do carro. Um componente destes pode controlar dois motores distintos, desta forma dois deles são necessários para controlar todas as rodas dos carro. Seu funcionamento se baseia em sinais elétricos, que em uma combinação específica movimentam os motores em sentido horário e anti-horário. Para cada motor controlado dois sinais elétricos são necessários. Sendo assim seriam necessárias 8 portas digitais do Arduino para um controle total de todas os motores do carro. Mais sobre o L293D pode ser encontrado em: http://www.arduinoecia.com.br/2014/04/controle-de-motor-cc-com-o-l293d-ponte-h.html

4.1.3 MÓDULO WI-FI ESP8266-01

Para que a comunicação fosse feita entre Arduino e o Android, seria necessária alguma forma de comunicação sem fio. Para isto foi utilizado o módulo ESP8266, um

módulo Wi-Fi de baixo custo, capaz de gerar redes sem fio, possibilitando assim um canal de comunicação entre a aplicação que foi desenvolvida e os componentes controlados pelo Arduino. Mais sobre esse módulo em: http://www.esp8266.com/wiki/doku.php?id=getting-started-with-the-esp8266.

4.2 DESENVOLVIMENTO DO PROJETO

Com o estudo destes componentes pode-se perceber que alternativas mais simples poderiam ser mais viáveis do que manter o ESP8266 como intermediário de comunicação entre o Arduino e o Android. A comunicação Serial entre o ESP8266 tornaria-se mais complicada e suscetível a erros, o que para este projeto demandou um tempo considerável de pesquisa, pois a comunicação não poderia ter delays de resposta para atuar com os motores.

O módulo ESP8266 além de se conectar a uma rede Wi-Fi, ele pode gerar uma rede própria e acionar suas próprias portas digitais(GPIO's). Estas portas funcionam da mesma forma que as portas digitais do Arduino , sendo assim, estas poderiam ser utilizadas para o controle dos componentes ligados aos motores.

Para que estas GPIO's pudessem ser manipuladas foi necessário fazer upload de código para dentro do módulo ESP8266. Para que este processo pudesse ser realizado alguns procedimentos deveriam ser feitos.

Os procedimentos necessários foram encontrados neste site: https://www.hackster.io/ROBINTHOMAS/programming-esp8266-esp-01-with-arduino-011389.

4.2.1 LIMITAÇÕES DE HARDWARE

Apesar da alternativa encontrada, o ESP8266 possui apenas 2 portas GPIO(GPIO0 e GPIO2) o que impossibilitava o acionamento de 4 motores, pois como relatado 8 portas seriam necessárias para o controle total dos motores.

Para amenizar essa limitação sem a compra de novos componentes eletronicos, uma alternativa encontrada foi utilizar as portas RX e TX do módulo como GPIO's. Um artigo no site: http://pedrominatel.com.br/pt/esp8266/utilizando-rx-e-tx-como-gpio-no-esp01/ descreve como esse processo pode ser realizado. Desta forma teríamos 4 porta digitais em vez de 2 como anteriormente, o que ainda não era suficiente para controlar os 4 motores, mas abria possibilidade para novas ideias que serão descritas posteriormente neste relatório.

4.2.2 PROTOTIPAGEM E INTEGRAÇÃO DOS COMPONENTES

A prototipagem para o controle do veículo consiste em 4 botões principais na tela, sendo alusivos às 4 direções possíveis que o carro pode tomar, esquerda, direita, para frente e para trás.

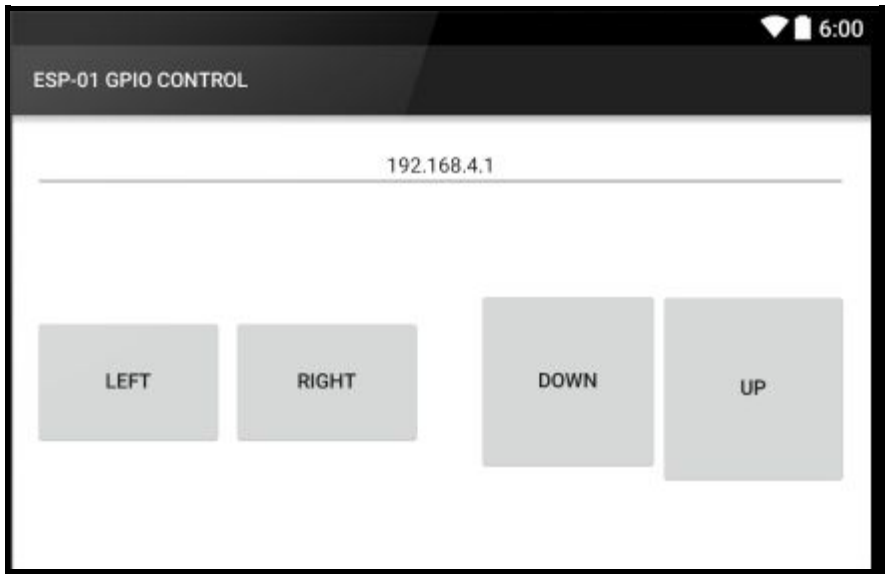

Figura 1 - Protótipo Inicial

Esta prototipagem inicial foi desenvolvida graficamente utilizando o software que serviu como principal ferramenta para o desenvolvimento para a aplicação Android, o software é uma IDE chamada **Android Studio** disponibilizada gratuitamente pela Google para o desenvolvimento de aplicações mobile. O download desta ferramenta pode ser efetuado em: https://developer.android.com/studio/index.html

Com base na ideia de utilizar 4 botões direcionais, pode-se fazer uma alusão às 4 GPIO's disponíveis do ESP8266, sendo assim quando um botão for segurado o mesmo deve ativar o sinal de uma GPIO em específico.

Partindo deste princípio o problema ainda se consistia em conseguir acionar os 4 motores com menos do que 8 portas. Porém além de acionar suas portas digitais o Arduino suporta "ler" os valores destas portas digitais, sendo assim a solução do problema fez-se em conectar as 4 GPIO's do ESP em 4 portas digitais do Arduino e via código fazer com que o Arduino leia os valores destas GPIO's e para cada porta acionada acione as 8 portas necessárias para controlar o carro. Como exemplo de possível configuração esquemática o protótipo da Figura 2 ilustra as conexões necessárias.

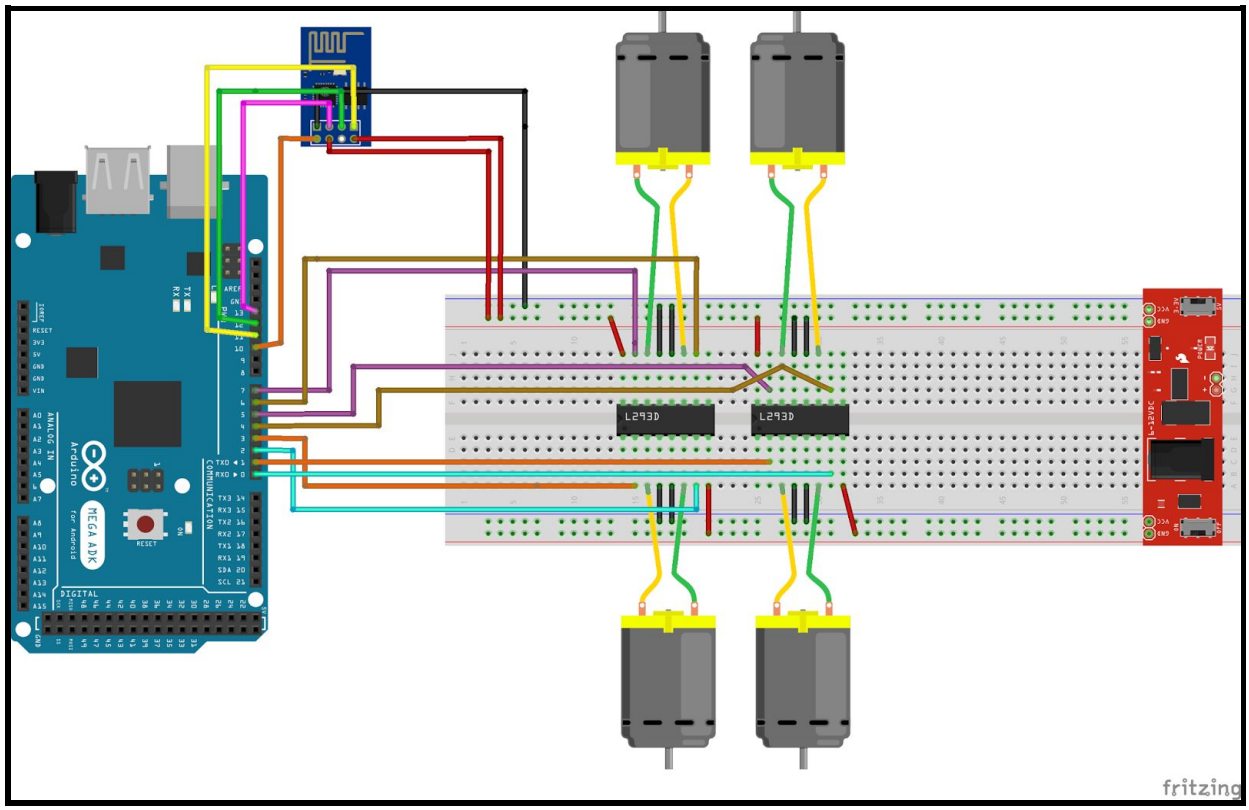

Figura 2 - Protótipo esquemático dos componentes eletrônicos

Neste protótipo as conexões entre o ESP8266 e Arduino estão organizadas conforme a Tabela 1, da seguinte forma:

ESP 8266	Arduino INPUT
GPIO 2	13
GPIO 0	12
GPIO 3 (RX)	11
GPIO 1 (TX)	10

Tabela 1 - ESP GPIO's Outputs para Arduino Inputs

As conexões entre o Arduino e o componentes L293D estão organizadas conforme a Tabela 2 abaixo:

Arduino porta digital	L293D - 1 porta	L293D - 2 porta
7	Input 4	-
6	Input 3	-
5	-	Input 4

4	-	Input 3
3	Input 1	-
2	Input 2	-
1	-	Input 1
0	-	Input 2

Tabela 2 - Conexões entre o Arduino e L293D

Estas configurações são suficientes para que o projeto funcione adequadamente.

4.2.3 PROGRAMANDO O ARDUINO

O Sketch utilizado no Arduino é fundamentado em ler os estados das portas em que os pinos GPIO's do ESP8266 foram conectados. Quando um pino estiver em "HIGH" significa que está ativo.

```
void loop(){
  gpio2 = digitalRead(GPIO2);
  gpio0 = digitalRead(GPIO0);
  gpio3 = digitalRead(GPIO3);
  gpio1 = digitalRead(GPIO1);

  if( gpio2 == HIGH ) //carro move para frente
    up();
  else if( gpio0 == HIGH ) //carro move para tras
    down();
  else if( gpio3 == HIGH ) //carro move para direita
    right();
  else if( gpio1 == HIGH ) //carro move para esquerda
    left();
  else
    stop_all();
}
```

Figura 3 - Arduino Sketch função principal

Cada pino GPIO acionado chama uma função diferente, que aciona determinados pinos para movimentar os motores. A programação deste Sketch foi realizada utilizando a IDE Arduino. O código completo pode ser visto no final deste relatório no índice de anexos.

4.2.4 PROGRAMANDO O ESP8266

Para a programação do ESP8266 com a IDE Arduino é necessário que a mesma esteja configurada conforme instruções do link disponibilizado no início da seção 4.2.

O Sketch desenvolvido para o ESP8266 se baseia em criar um site via concatenação de strings e enviá-lo para o ESP, servindo como um servidor que manterá as funcionalidades principais das 4 ações possíveis.

```
void geraSite() {
  site = "<html>\n";
  site += "<head><title> ESP8266 CarControl</title></head>\n";
  site += "<body>\n";
  site += "<h5>UP</h5><br>\n";
  site += "<center><a href=\"/up?state=on\"><button style=\"font-size: 34px;\">UP ON</button></a></center>\n";
  site += "<br><center><a href=\"/up?state=off\"><button style=\"font-size: 34px;\">UP OFF</button></a></center>";
  site += "<hr>\n";
  site += "<h5>DOWN</h5><br>\n";
  site += "<center><a href=\"/down?state=on\"><button style=\"font-size: 34px;\">DOWN ON</button></a></center>\n";
  site += "<br><center><a href=\"/down?state=off\"><button style=\"font-size: 34px;\">DOWN OFF</button></a></center>";
  site += "<hr>\n";
  site += "<h5>LEFT</h5><br>\n";
  site += "<center><a href=\"/left?state=on\"><button style=\"font-size: 34px;\">LEFT ON</button></a></center>\n";
  site += "<br><center><a href=\"/left?state=off\"><button style=\"font-size: 34px;\">LEFT OFF</button></a></center>";
  site += "<hr>\n";
  site += "<h5>RIGHT</h5><br>\n";
  site += "<center><a href=\"/right?state=on\"><button style=\"font-size: 34px;\">RIGHT ON</button></a></center>\n";
  site += "<br><center><a href=\"/right?state=off\"><button style=\"font-size: 34px;\">RIGHT OFF</button></a></center>";
  site += "<h5>STOP ALL</h5><br>\n";
```

Figura 4 - Criação do servidor criado no ESP8266

Para cada ação existe uma função que aciona a GPIO solicitada via uma REQUEST ao servidor html criado na Figura 4.

```
server.on("/up", [](){ //Quando o direcional pra cima for clicado
  String state = server.arg("state");
  if (state == "on"){
     digitalWrite(GPIO2, HIGH);
  }else if (state == "off"){
     stop_all();
  }
  geraSite();
});
```

Figura 5 - Manipulação do servidor do ESP8266

Esta função “server.on()” demonstrada na Figura 5 verifica o estado do servidor criado e avalia qual GPIO foi requisitada e então a ativa ou desativa conforme o valor da variável “state” definida em seu escopo. Cada direção do carro possui uma função semelhante que verifica seu estado.

Desta forma o servidor fica aguardando a solicitação do cliente para acionar as portas do ESP. O código completo pode ser visto no final deste relatório no índice de anexos.

Caso se queira executar testes sem a necessidade da aplicação Android, pode-se acessar o servidor por qualquer Browser desejado. Para isso basta conectar o computador/celular na rede Wi-Fi gerada pelo ESP, que neste exemplo está como “ESP_TAILER”, e acessar o endereço “192.168.4.1” padrão de redes geradas pelo ESP, este valor pode alterar para ESP’s diferentes. Para que o carro se movimente para frente por exemplo, basta digitar “192.168.4.1/up?state=on” ou clicar no botão “UP ON” que aparece na tela.

4.2.4 PROGRAMANDO A APLICAÇÃO ANDROID

A aplicação Android constitui-se de criar um “HttpRequest” que age com em modo “cliente” conectando ao “servidor” criado no ESP. Para gerenciar requisições HTTP no Android o uso da biblioteca “Apache HttpClient 4.1” fez-se necessário. Esta biblioteca utiliza “HttpUriRequest” para enviar e receber dados, assim com a subclasse “HttpGet” podemos requisitar uma resposta do servidor do ESP. Para que as requisições tenham efeito, o dispositivo deve estar conectado na rede Wi-Fi gerada pelo ESP8266, caso contrário o aplicativo não terá conhecimento do servidor que deve requisitar. O nome da classe responsável por esta comunicação chama-se “TaskEsp”, o código completo pode ser visto no final deste relatório no índice de anexos.

Para cada botão presente na interface existe um evento atrelado, o mesmo capita quando um botão é clicado e segurado. As requisições ao servidor ESP são executadas neste evento, atuando individualmente para cada botão.

```
View.OnTouchListener btnUpClickListener = new View.OnTouchListener(){
    @Override
    public boolean onTouch(View v, MotionEvent event) {
        v.setSelected(true);
        String state;
        String serverIP = editIp.getText().toString()+":80";
        TaskEsp taskEsp = new TaskEsp(serverIP);

        if(event.getAction() == MotionEvent.ACTION_DOWN){
            state = "/up?state=on";
            taskEsp.execute(state);
        }else if(event.getAction() == MotionEvent.ACTION_UP){
            v.setSelected(false);
            state = "/up?state=off";
            taskEsp.execute(state);
        }
        return true;
    }
};
```

Figura 6 - Tratamento dos eventos dos botões

As áreas destacadas da Figura 6 ilustram como que a requisição é realizada, passando uma URL quando o botão se mantém pressionado e outra quando o mesmo não foi pressionado.

A interface final desenvolvida é ilustrada na Figura 7 abaixo.

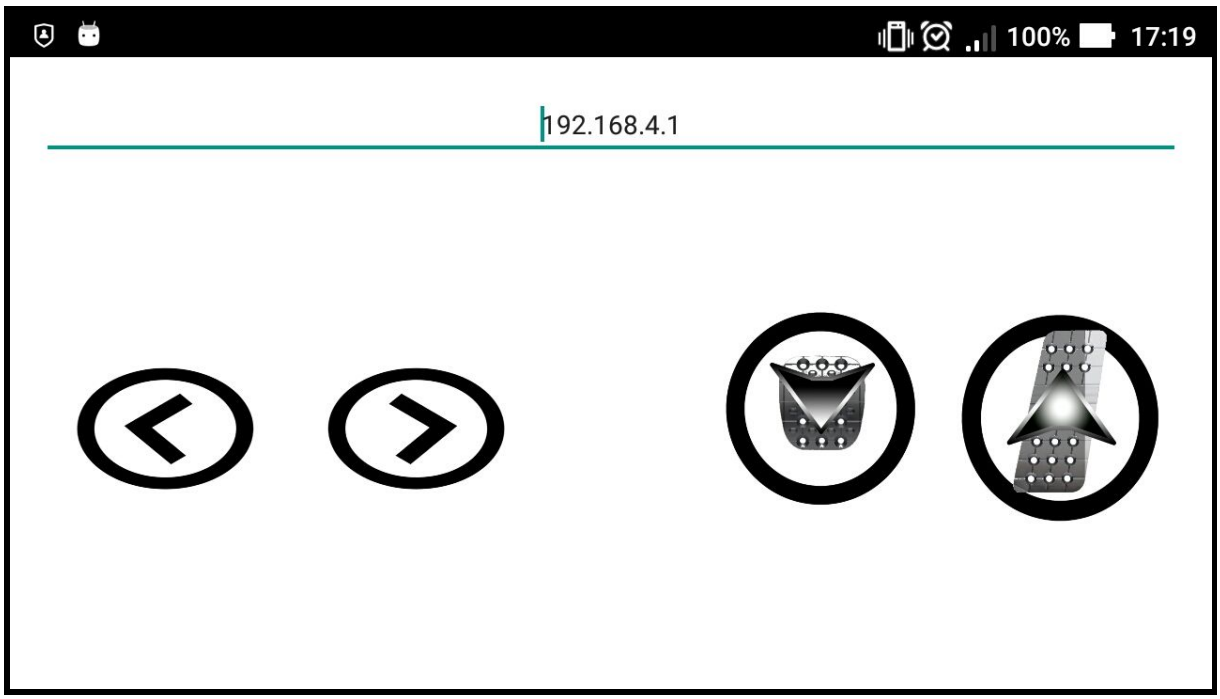

Figura 7 - Interface final

4.3 SUGESTÕES E RECOMENDAÇÕES

O trabalho desenvolvido até aqui foi totalmente funcional, porém ainda um pouco imaturo, a aplicação Android poderia utilizar recursos de captura de movimentos para controlar as direções do carro, por exemplo.

O hardware responsável pela comunicação Wi-Fi(ESP8266) possui versões bem mais robustas e com mais funcionalidades e facilidades, o que poderia melhorar o processo de comunicação, sendo capaz até de não necessitar a utilização da plataforma Arduino.

5. CONCLUSÕES

O presente projeto acabou por agregar um bom conhecimento sobre como funciona a internet das coisas, assunto que está cada vez mais em alta. O controle remoto a dispositivos traz infinitas possibilidades de utilização em qualquer ramo social. Os resultados obtidos através das pesquisas são com certeza relevantes para um conhecimento pessoal, sendo assim espero poder compartilhar um pouco do trabalho realizado aqui com a comunidade a que interessar, de forma a difundir ainda mais a acessibilidade disponibilizada por estes meios informáticos de baixo custo.

A interação hardware/software utilizada neste projeto traz uma experiência totalmente diferente do que a academia poderia me disponibilizar em um curso como Sistemas de Informação, principalmente no que se diz respeito ao estudo das várias áreas da informática que tive de conhecer e aprender um pouco a respeito.

6. ANEXOS

Todos arquivos citados neste relatório estão armazenados em: https://github.com/AnthonyTailer/Arduino-Car-Control-Project.