UNIVERSIDADE FEDERAL DO PIAUÍ
CAMPUS SENADOR HELVÍDIO NUNES DE BARROS
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

WILSON MARCELO DE SOUSA

CENTRAL BOOT: Um robô móvel com Sensor de Temperatura Umidade

PICOS – PI
2014

WILSON MARCELO DE SOUSA

CENTRAL BOOT: Um robô móvel com Sensor de Temperatura Umidade

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Sistemas de Informação, Campus Senador Helvídio Nunes de Barros da Universidade Federal do Piauí, como parte dos requisitos para obtenção do Grau de Bacharel em Sistemas de Informação.

.

ORIENTADOR: PROF. Esp. IVENILTON ALEXANDRE DE SOUZA MOURA

PICOS – PI
2014

Eu, **Wilson Marcelo de Sousa**, abaixo identificado(a) como autor(a), autorizo a biblioteca da Universidade Federal do Piauí a divulgar, gratuitamente, sem ressarcimento de direitos autorais, o texto integral da publicação abaixo discriminada, de minha autoria, em seu site, em formato PDF, para fins de leitura e/ou impressão, a partir da data de hoje.

Picos-PI 12 de março de 2014.

Wilson Marcelo de Sousa
Assinatura

FICHA CATALOGRÁFICA
Serviço de Processamento Técnico da Universidade Federal do Piauí
Biblioteca José Albano de Macêdo

S586c Sousa, Wilson Marcelo de.
Central Boot: um robô móvel com sensor de temperatura e umidade / Wilson Marcelo de Sousa. – 2013.
CD-ROM : il. ; 4 ¾ pol. (77 p.)

Monografia(Bacharelado em Sistemas de Informação) – Universidade Federal do Piauí. Picos-PI, 2013.
Orientador(A): Prof. Esp. Ivenilton Alexandre de S. Moura

1. Robótica. 2. Web. 3. Sensor. 4. Arduíno I. Título.

CDD 005.1

WILSON MARCELO DE SOUSA

CENTRAL BOOT: Um robô móvel com Sensor de Temperatura Umidade

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Sistemas de Informação, Campus Senador Helvídio Nunes de Barros da Universidade Federal do Piauí, como parte dos requisitos para obtenção do Grau de Bacharel em Sistemas de Informação.

Data de Aprovação: 10/03/2014

Prof. Ivenilton Alexandre de Souza Moura, Esp. (Orientador)

Prof. Patricia Medyna Lauritzen de Lucena Drumond, MSc. (Membro)

Prof. Flavio Henrique Duarte de Araújo (Membro)

Dedico esta monografia a toda minha família em especial aos meus pais Ana Maria de Sousa e João Faustino de Sousa e irmãos Francisco Avelar de Sousa, Francisco Solisvaldo de Sousa, Vicente de Paula Sousa, Maria Vera Lucia de Sousa, aos Meus avós maternos Maria Nazaré de Sousa (*In Memoriam*) e Domingos Cicero do Nascimento (*In Memoriam*) e aos paternos Francisco Faustino de Sousa (In Memoriam) e Domingas Maria de Sousa e minha Tia Pedrina também ao meu filho Álvaro e a sua mãe Josina Ramos; por estar sempre ao meu lado em todas as minhas conquistas e decisões.

AGRADECIMENTOS

Agradeço primeiramente a Deus, pelo dom da vida, pela minha capacidade de pensar e temer, proporcionando mais uma vitória e por me guiar nessa jornada de conquistas. Agradeço a toda minha família pela paciência, conselhos, ajuda de custo e por acreditar em mim, mesmo com as dificuldades encontradas.

Aos meus orientadores Msc. Algeir Sampaio por ter acreditado e confiado no meu potencial, pela oportunidade de trabalhar na área da robótica, e o Esp. Ivenilton Alexandre de Souza Moura por ter dado continuidade ao trabalho, pela paciência e obrigado por ajudar a concluir este trabalho.

A todos os professores que compõem o curso de Sistemas de Informação, que contribuíram através de seus conhecimentos repassados.

Aos meus colegas do curso, da UFPI, em especial a equipe do LIPPO, Allan, Marco Antonio, Varton, André, Bruno Rafael, Cliciano, Daniel, Mezzomo, Thiago Jose, Carine, Jonilton, Cavalcante, Abimael, Renan, Klisanderson, Auricélio, Rafael Gervasio, Paula Michele, obrigado por estarem presentes durante toda essa jornada.

Aos amigos que me ajudaram a superar todos os momentos difíceis e estiveram comigo apoiando e ajudando na escrita deste trabalho e no layout da página, em especial Rawena Rodrigues e Jose Hilton.

“O covarde nunca tenta, o fracassado nunca termina e o vencedor nunca desiste.”

(Norman Vicent Peale)

“É melhor atirar-se à luta em busca de dias melhores, mesmo correndo o risco de perder tudo, do que permanecer estático, como os pobres de espírito, que não lutam, mas também não vencem, que não conhecem a dor da derrota, nem a glória de ressurgir dos escombros. Esses pobres de espírito, ao final de sua jornada na Terra não agradecem a Deus por terem vivido, mas desculpam-se perante Ele, por terem apenas passado pela vida.”

(Bob Marley)

“Julgue seu sucesso pelas coisas que você teve que renunciar para conseguir.”

(Dalai Lama)

RESUMO

A robótica tem uma grande participação na sociedade, vem colaborando já a muito tempo com o crescimento da produtividade nas fábricas, na exploração de locais desconhecidos ou inacessíveis para os seres humanos, no auxílio de tarefas domésticas. Devido ao aumento das tecnologias foi possível a construção de um projeto de um robô móvel controlado via *web*, capaz de executar ações de acordo com os comandos enviados pela página *web* para o microcontrolador onde toda a informação é processada assim acionando os motores. O robô móvel usando um sensor de temperatura e umidade conectados ao um microcontrolador desempenhando o papel de capturar os dados de um ambiente e enviando para a mesma página. Sendo possível o protótipo robótico enviar e receber informações na aplicação. O presente trabalho Apresentando uma breve descrição sobre a robótica, redes de computadores, TCP/IP conceitos de inteligência artificial e de eletrônica plataforma Arduino utilizada para processar informações controlando os componentes deste projeto.

Palavras-chave: Robótica, *web*, sensor, Arduíno.

ABSTRACT

The robot has a large stake in the company, has been working since a long time with the growth of productivity in the factories, in the exploration of unknown or inaccessible places for humans, in aid of household chores. Due to the rise of technology was possible to construct a design of a mobile robot controlled via web, able to perform actions according to the commands sent by the web page to the microcontroller where all information will be processed as soon cranking the engine. Using a microcontroller connected to playing the role of capturing data from an environment and sending to the same page sensor temperature and humidity. Being able to send and receive information in the application. Giving a brief description of robotics, computer networking, TCP / IP concepts of artificial intelligence and electronic Arduino platform used to process information controlling the components of this project.

Keywords: Robotics, web, sensor, Arduino.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DHT11	Sensor de Temperatura e Humidade
HTML	*HyperText Markup Language*
IA	*Inteligência Artificial*
IDE	*Integrated Drive Eletronics*
PHP	*Personal Home Page*
CI	Circuito Integrado
CC	Corrente Continua
DC	Controle de diferentes tipos de motores

SUMÁRIO

1 INTRODUÇÃO

A robótica é uma área que vem evoluindo no decorrer dos últimos anos. Ela é dividida em várias subáreas e aplicada em locais variados como indústrias, empresas, no meio científico, operação de risco, entre outros. A cada dia que passa, a Inteligência Artificial (IA) permite que máquinas possam realizar trabalhos humanos, como em atividades perigosas, pesadas ou rotineiras; além de aumentar a sua eficiência na interação com equipamentos sofisticados para realização de tarefas de forma produtiva.

Diariamente nos deparamos com alguns tipos de problemas e muitas vezes não sabemos como resolver tais situações de imediato, uma solução encontrada é a atuação de agentes inteligentes na área da robótica. A robótica Sendo uma das maiores contribuições para o avanço da tecnologia atual e para projetos de automação e controles de unidades móveis. No entanto, o desenvolvimento de um projeto de robótica necessita de pesquisas de tecnologias de *hardware* e *software*, aperfeiçoamento de otimização e análise da eficiência de algoritmos, e utilização de componentes eletrônicos, tais como, sensores e mecanismos de comunicação. “O desenvolvimento do robô começa com a elaboração do projeto, construção do protótipo, aplicação de testes de validação e eficiência” (LIMA, 2013).

Muitos projetos de robótica envolvem a utilização da plataforma Arduino, que é uma alternativa bastante expansível, descomplicada e de baixo custo para o desenvolvimento de protótipos é necessário um kit com placas de Arduino, Ethernet shield, protoshield*,* sensor de temperatura e umidade, baterias e um protótipo de uma unidade robótica móvel controlada. O desenvolvimento de *software* na plataforma IDE Arduino e outro no PHP são capazes de incorporarem inúmeros sensores e módulos de comunicação para interagirem com outros aparelhos, inclusive com a *Internet*.

Para se atingir a excelência em um projeto deste porte, é necessário o desenvolvimento de alguns componentes, o que demanda muito tempo e estudo. Visando a criação de um protótipo de baixo custo, foram escolhidos componentes prontos como a placa de Arduino, que é um controlador lógico *open source*, que recebe informações, processa e retorna uma saída.

A programação desta placa do Arduino é feita através de uma IDE exclusiva, com a finalidade de tornar o processo de escrita de funcionalidades simples

e rápido. Para que a comunicação entre a placa do Arduino e a aplicação *web* seja feita, foi adotado o estudo dos modos de comunicação, o *Socket* e *WebSocket,* utilizados na linguagem de programação PHP.

Este trabalho apresenta uma aplicação envolvendo comunicação entre uma página *web* e a placa do Arduino para controle de um robô móvel. A aplicação poderá ser utilizada ainda para outras finalidades, bastando realizar a programação adequada na placa do Arduino para que os comandos do aplicativo da página *web* sejam atendidos adequadamente.

1.1 Motivação

No atual cenário tecnológico, torna-se evidente a necessidade de exploração dos mais diversos meios de controle para unidades robóticas móveis. A finalidade deste trabalho é proporcionar à máquina, controlada pelo homem, uma atuação em ambientes inacessíveis a humanos, como lugares com temperaturas extremas, desarmamento de bombas, exploração de usinas nucleares, montadoras automobilísticas e auxílio de cirurgias a distância.

Para a utilização desta tecnologia, muitas formas de comunicação podem ser aplicadas a placa do Arduino, dentre elas podem ser citadas a conexão cabeada, *Bluetooth, Wi-Fi*, Rádio Frequência, Infravermelho, dentre outras.

Esta grande variedade de formas de comunicação tornam o uso do Arduino cada vez mais crescente em projetos de robótica e automação, este foi um dos fatores que mais fortaleceram a ideia deste projeto. As possibilidades de criar um novo produto ou serviço utilizando esta tecnologia são inúmeras, e por isso, é cada vez maior o número de desenvolvedores que optam por utilizar a placa do Arduino na concepção de um novo trabalho na área.

Fatores como estes fortalecem ainda mais os trabalhos, consequentemente, tornam o processo de resolução de possíveis problemas ou imprevistos, que venham a ocorrer durante a implementação, mais simples e rápidos através de testes e troca de informações em fóruns de discussão, *sites* especializados e livros.

1.2 Problema

A manipulação de robôs através de computadores com *internet* já é realidade atualmente, mas não há um consenso ou uma forma padrão de utilização destes controladores.

Considera necessário, portanto, desenvolver uma página web que é capaz de receber e enviar sinais, proporcionando uma comunicação entre computadores ou dispositivos móveis com a placa do Arduino, para que este possa receber a temperatura do ambiente em um *site web* e controlar uma unidade robótica móvel terrestre que utilize a plataforma IDE Arduino. Para isso, basta que seja implementado um pequeno código de recepção dos dados na plataforma, que também é disponibilizado neste trabalho. Esta padronização poderia facilitar futuros projetos relacionados a este evitando o retrabalho com a implementação da comunicação.

1.3 Abordagem

Neste trabalho realizamos uma investigação sobre as técnicas de controles de unidades robóticas através da plataforma IDE Arduino e de páginas desenvolvidas em PHP, especificamente voltadas para a área de robótica móvel. Descrevendo suas arquiteturas e funcionalidades com a finalidade de obter uma forma de controle para unidade robóticas móveis que utilizem a plataforma Arduino, e qualquer meio de comunicação com a *internet*, sendo desenvolvido em linguagem dinâmicas para *web* HTML e PHP.

1.4 Objetivos

1.4.1 Objetivo Geral

O presente trabalho propõe o desenvolvimento de um protótipo de uma unidade robótica com funcionalidades automatizadas, controladas por um computador ligado à *internet* ou ainda via dispositivos móveis através de um *browser*. Este trabalho utilizará um protótipo pronto e se concentrará no desenvolvimento das funcionalidades do *software.*

1.4.2 Objetivos Específicos

- Desenvolver um sistema capaz de controlar um robô através de redes cabeadas e sem fio.
- O uso de sensores de temperatura e umidade, para medi-las e enviar esses dados em uma página web.
- Montar um protótipo robótico móvel terrestre controlado via web.
- Desenvolver site para controle do robô móvel à distância.

1.5 Organização Do Documento

O Capitulo 2 contém o embasamento científico utilizado para formação do referencial teórico, que subdivide-se nos seguintes tópicos: robótica, robótica móvel, história da robótica móvel e inteligência artificial.

O Capitulo 3 descreve os componentes essenciais para o concepção e desenvolvimento deste projeto, detalhando informações sobre o robô móvel, Arduino, *hardware e software*.

O Capítulo 4 demonstra a implementação dos códigos nas linguagens de programação Arduino e PHP, e a montagem do protótipo.

O Capítulo 5 apresenta os testes realizados entre a comunicação da aplicação e o protótipo.

O Capítulo 6 expõe a conclusão à qual se chegou após a realização dos testes.

2 REFERENCIAL TEÓRICO

2.1 Robótica

Comforme Russell e Norvirg (2004) a robótica é um ramo educacional e tecnológico que engloba computadores, robôs e computação, que trata de sistemas compostos por partes mecânicas automáticas e controlados por circuitos integrados, tornando sistemas mecânicos motorizados, seja manualmente ou automaticamente por circuitos elétricos. As máquinas, pode-se dizer que são vivas, mas ao mesmo tempo são uma imitação da vida, não passam de fios unidos e mecanismos, isso tudo junto concebe um robô. Cada vez mais as pessoas utilizam os robôs para suas tarefas. Em breve, tudo poderá ser controlado por robôs. Os robôs são apenas máquinas: não sonham nem sentem e muito menos ficam cansados. Esta tecnologia, hoje adaptada por muitas fábricas e indústrias, tem obtido de um modo geral, êxito em questões levantadas sobre a redução de custos aumento de produtividade e os vários problemas trabalhistas com funcionários.

2.2 Robótica Móvel

Conforme Heinen (2002), a robótica móvel é uma área de pesquisas que lida com controles de veículos autônomos ou controlados. O grande diferencial dessa área em relação as outras áreas de pesquisas da robótica, é que ela dá maior ênfase ao deslocamento e o ambiente que podem se modificar dinamicamente, compostos de obstáculos móveis ou estáticos. Nesse caso, o robô deve dispor de habilidades de reconhecer obstáculos e suas funcionalidades operar em conjunto para responder em tempo real o que possa acontecer com ele nesse cenário.

Robôs móveis podem atuar em diversas tarefas, no transporte de peças em uma indústria até à exploração de locais perigosos, como águas profundas, áreas radioativas, crateras de vulcões, ambientes espaciais e mesmo outros planetas. Diferentemente dos robôs manipuladores, um robô móvel é livre para se mover em todas as direções. Portanto, seus controles exigem um maior grau de interação com o ambiente, o que inclui o sensoriamento de variáveis de ambiente e eventualmente a detecção do resultado das ações do próprio robô.

2.3 História Da Robótica Móvel

Segundo Walter (1950) a robótica móvel vem sendo estudada e desenvolvida desde os anos 50, quando alguns pesquisadores já tinham interesse no desenvolvimento de robôs móveis. No ano de 1950 o pesquisador Willian Walter construiu vários robôs móveis capazes de executar tarefas tais como o desvio de obstáculos e seguir fontes luminosas, utilizando capacitores para controlar o robô.

Conforme Nilson (1969), nas cidades de Stanford no ano de 1969, ele desenvolveu um robô móvel SHAKEY (ver Figura 1). Esse robô utilizava dois motores de passo e uma configuração diferencial mais conhecida como (cinemática diferencial) para sua locomoção disponibilizava de sensores de distância, câmeras de vídeo e sensores táteis. Ele era conectado a dois computadores por *links* de rádio e de vídeo. O robô móvel SHAKEY utilizava programas para a percepção, modelagem, e atuação no ambiente. As tarefas executadas pelo robô incluíam desviar de obstáculos e a movimentação em blocos coloridos. Essa unidade robótica tinha grandes dificuldades de processamento e interpretação das informações sensórias obtidas do ambiente, e nunca foi capaz de completar uma sequência completa de ações em um ambiente real.

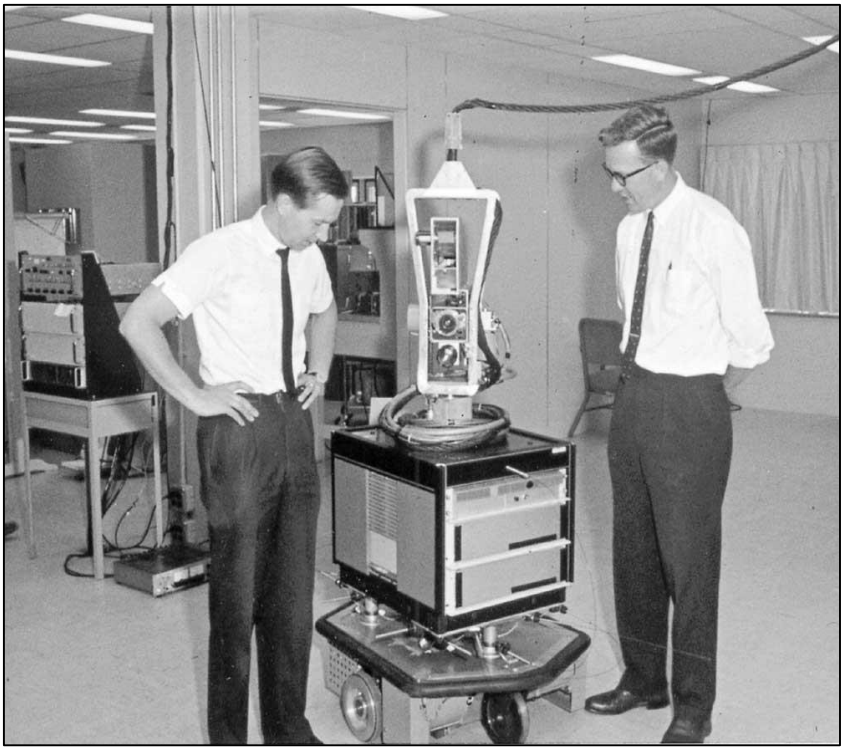

Figura 1 *- Primeiro robô móvel SHAKEY*

Fonte:<http://www.ai.sri.com/shakey/images/sven/shakey_2_bw_200x163.jpg>

Robôs móveis são hoje objetos de intensa atividade de pesquisas, principalmente devido à complexidade inerente em seu funcionamento, que transformam os robôs móveis em um grande laboratório para a inteligência artificial, considerando-se a facilidade com que os mesmos podem ser testados em simuladores.

2.4 A Inteligência Artificial

A inteligência Artificial (IA) é conceituada por vários autores da seguinte forma: *"*O novo e interessante esforço para fazer os computadores pensarem "máquinas com mentes", no sentido total e literal" (RUSSELL, 2004 apud HAUGELAND, 1985). "Automatização de atividades que associamos ao pensamento humano, atividades como tomada de decisões, a resolução de problemas, o aprendizado" (RUSSELL, 2004 apud BALLMAN, 1978). "A arte de criar maquinas que executam funções que exigem inteligência quando executadas por pessoas" (RUSSELL, 2004 apud KURZWEIL, 1990). "O estudo de faculdades mentais pelo uso de modelos computacionais" (RUSSELL, 2004 apud CHARNIAK; MCDERMOTT, 1985). "O estudo das computações que tornam possível perceber, raciocinar e agir" (RUSSELL, 2004 apud WINSTON, 1992). "A inteligência computacional é o estudo do projeto de agentes inteligentes" (RUSSELL, 2004 apud POOLE *et al*., 1998).

Enfatizando as opiniões dos autores a Inteligência Artificial (IA) é um ramo da ciência da computação que se propõe a elaborar dispositivos que simulem a capacidade humana de raciocinar, perceber, tomar decisões e resolver problemas, enfim, a capacidade de ser inteligente.

> A Inteligência Artificial (IA) surgiu como desafio de criar máquinas que pudessem pensar e (talvez) superar o pensamento humano: o santo graal da computação! Com o passar do tempo enquanto explorava os mecanismos do pensamento, a IA tornou-se mais pragmática. Usa estratégias diferentes para resolver problemas práticos e complexos que surgem na aplicação da tecnologia da computação. Percebeu-se também, que a inteligência é muito complexa para ser descrita por uma única teoria. Em vez disso, uma constelação de teorias caracterizam o tema a partir de diferentes níveis de abstração (LUGER, 2004).

Esta área da ciência é grandemente impulsionada com o rápido crescimento e desenvolvimento da informatização e da computação, permitindo que novos elementos sejam rapidamente agregados à IA.

Segundo Luger (2004), em níveis mais baixos, as redes neurais, os algoritmos genéticos e outras formas de computação ajudam o entendimento da adaptação, percepção, corporificação e interação com o mundo físico. Em nível mais abstrato, projetistas de sistemas especialistas, de agentes inteligentes, de modelos estocásticos e de programas de compreensão da linguagem natural refletem o papel do conhecimento. Os lógicos propõem ainda dedução, abdução, manutenção da verdade e outros modelos e modos para raciocínio.

Os robôs inteligentes podem ser de grande utilidade na medicina, diminuindo o número de erros médicos, na exploração de outros planetas, no resgate de pessoas soterradas por escombros, além de sistemas inteligentes para resolver cálculos e realizar pesquisas que poderão encontrar cura de doenças.

AZIMOV, renomado escritor e bioquímico, apresentou várias teorias sobre robótica, entre suas várias contribuições estão as Três Leis da Robótica para a convivência das máquinas com seres humanos:

1. Um robô não pode ferir um ser humano ou, por omissão, permitir que um ser humano sofra algum mal.
2. Um robô deve obedecer às ordens que lhe sejam dadas por seres humanos, exceto nos casos em que tais ordens contrariem a Primeira Lei.
3. Um robô deve proteger sua própria existência desde que tal proteção não entre em conflito com a Primeira ou a Segunda Lei (OLIVEIRA, 2001).

Esta tecnologia, hoje adaptada por muitas fábricas e indústrias, tem obtido de um modo geral, êxito em questões levantadas sobre a redução de custos aumento de produtividade e os vários problemas trabalhistas com funcionários.

O controle de robôs móveis é considerado uma importante área de pesquisa, para o desenvolvimento das trajetórias de ambientes a serem explorados, possibilitar a escolha das melhores técnicas de comunicação do robô e o ambiente, permitindo escolher as melhores ações a serem executadas durante o percurso.

A teoria da análise do comportamento demostra que os organismos vivos são capazes de aprender interagindo com o ambiente em que habita, recebendo estímulos e punições em respostas as suas ações. Baseado nos organismos vivos, nos robôs a forma de aprendizagem também pode modificar diretamente o seu comportamento móvel, reforçando ou inibindo determinadas rotas podendo influenciar

no seu controle. Sendo que algumas dessas aprendizagens estão ligadas diretamente as redes de computadores.

2.5 Redes De Computares

Conforme Tanenbaum (2011), as redes de computadores são vários computadores interligados trocando informações conectados por cabos ou sem fios sendo locais, metropolitanas e geograficamente distribuídas, usadas em aplicações domesticas, comerciais, por usuários móveis e questões sociais.

Segundo Tanenbaum (2011), Poucas outras áreas, em um período tão curto, apresentam tantas revoluções tecnológicas quanto a de redes de computadores. Em alguns poucos anos, vimos a popularização da *internet*, a predominância do protocolo IP sobre outros protocolos, o crescimento da preocupação com temas como “segurança” e “qualidade de serviço”, e mais recentemente a expansão de redes sem fio. Observamos também a convergência das tecnologias utilizadas em redes de telecomunicações e redes de computadores, nas quais, até mesmo os telefones celulares podem ser identificados por endereços IP.

De acordo com o Tanenbaum (2011) quando quisermos nos referir ao um conjunto de computadores autônomos interconectados por uma única tecnologia trocando informações. A conexão não precisa ser feita diretamente por um fio de cobre; também podem ser usadas fibras ópticas, microondas, ondas de infravermelho e satélites de comunicação. Existem redes em muitos tamanhos, modelos e formas. E uma forma bastante utilizada em comunicação é o TCP/IP.

O TCP/IP não é um único protocolo; ele é composto por um conjunto de protocolos por causa da sua diversidade, ele não utiliza diretamente as normas do modelo OSI que é composto por sete camadas, enquanto o TCP/IP dispõe de um modelo de quatro camadas para estabelecer comunicações em redes de computadores (ver Figura 2).

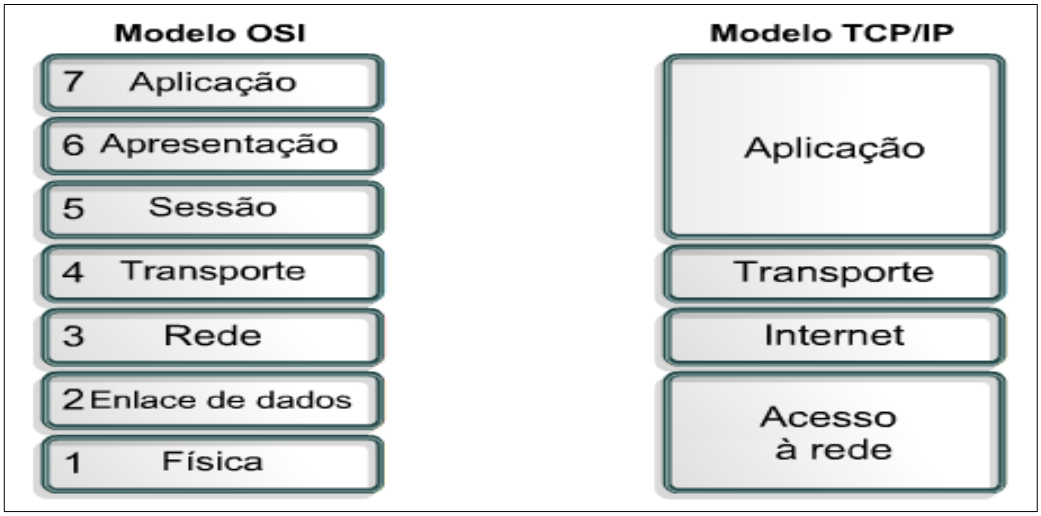

Figura 2 - Modelo OSI e Modelo TCP/IP

Fonte:< http://rdslocais.blogspot.com.br/2011/08/modelo-osi-x-modelo-tcpip.html>

Conforme Rob Scrimger (2002) a quarta camada ou camada de Aplicação do modelo de referência TCP/IP, seja tratado com maior ênfase, não que as outras três camadas não sejam importantes, mas é porque essa camada é responsável por tratar aplicativos baseados em soquetes e aplicativos de sistemas básicos de saída e entrada da rede (NetBios). Os aplicativos baseados em soquetes existem em todos os clientes que utilizam o TCP/IP. Três elementos são exigidos para aplicativos baseados em soquetes: um endereço IP, uma porta e um tipo de serviço. Cada cliente que utiliza o TCP/IP terá um endereço único de 32 bits. Cada endereço tem 65.536 pontos de entrada, chamados de portas esses aplicativos operam em portar particulares.

A terceira camada do modelo TCP/IP é a camada de Transporte. Essa camada tem um proposito bem simples que é conectar ou não conectar. Dois protocolos são utilizados nessa camada: *Transmission Control Protocol* (TCP) e User *Datagram Protocol* (UDP). O TCP estabelece uma comunicação confiável orientada para comunicações que são mais lentas na transmissão. Enquanto que o UDP estabelece uma comunicação não-garantida sem conexão que é mais rápida na transmissão (ROB SCRIMGER, 2002).

Quando um aplicativo utiliza o TCP para comunicação, um *handshake* de três vias é estabelecido, assegurando que os pacotes são entregues livres de erros, na sequência e sem perdas ou duplicação de dados. Enquanto que um aplicativo que utilize UDP não estabelece um *handshake* de três vias e não oferece uma garantia de

entrega do pacote. Essencialmente, o UDP envia os dados ao cliente receptor e espera que ele seja recebido. Não há nenhum acompanhamento da comunicação na transmissão dos dados. O UDP é muito mais rápido que o TCP, não garante a entrega dos dados.

A segunda camada do modelo TCP/IP é a camada de Inter-Rede ou *Internet* é responsável pelo endereçamento e roteamentos da rede. Além disso, essa camada também é responsável pela fragmentação dos pacotes. Os pacotes de dados são montados e remontados para a transmissão. Nessa camada operam vários protocolos onde os mais comuns são: *Intenert Protocol* (IP)-Um protocolo sem conexão que fornece seleção de endereçamento de rota. As informações do cabeçalho adicionadas ao pacote de dados contém os endereços de origem e de destino e a seleção da rota é feita com base nesses endereços. O IP também realiza a montagem e a desmontagem de pacotes chamados de fragmentações. O IP controla o tráfego por meio de roteadores ajustando o valor do *Time Live* (TTL) dos pacotes à medida que são transmitidos por meios de roteadores. O TTL especifica por quanto tempo um pacote pode permanecer em uma rede. A medida que um pacote passa por um roteador, o TTL é decrementado por 1 segundo e, quando um TTL 0 é alcançado, o pacote é descartado (ROB SCRIMGER, 2002).

Ainda segundo Rob Scrimger (2002), outro protocolo utilizado nessa camada é o *Internet Control Message Protocol* (ICMP), utilizado com frequência com o utilitário *Packet Internet Group* (PING). O *PING* é utilizado na maioria das vezes para solucionar problemas de conectividade. O ICMP também é utilizado para enviar ao roteador pacotes *source quench* que informa aos clientes que muito tráfego está vindo muito rápido e que os pacotes correm o risco de serem descartados.

De acordo com Rob Scrimger (2002) mais um protocolo utilizado pela camada de *Internet* é o *Addres Resolution Protocol* (ARP) – utilizados para mapear endereços IP em endereços MAC. Uma vez que o endereço de MAC é conhecido, o pacote pode ser enviado do cliente diretamente ao cliente receptor. Se os clientes tiverem no mesmo segmento. E se os clientes estiverem em segmentos diferentes, o pacote é enviado ao roteador.

Ainda conforme Rob Scrimger (2002) a primeira camada do modelo TCP/IP é a camada de Interface de Rede, correspondente a camada de enlace e a camada física do modelo OSI e é responsável pelo acesso à rede. Essa camada se comunica diretamente com a rede. Ela é a ligação entre a topologia de rede e a camada de inter-

rede. Praticamente o TCP/IP é uma importante comunicação para ser usado em sistemas distribuídos com o uso de redes de computadores.

Existe na literatura uma considerável confusão entre uma rede de computadores e um sistema distribuído. A principal diferença entre eles é que, em um sistema distribuído, um conjunto de computadores independentes parece ser, para seus usuários, um único sistema coerente. Ele tem um único modelo ou paradigma que apresenta aos usuários. Com frequência, uma camada de *software* sobre o sistema operacional, chamado *middleware*, é responsável pela implementação desse modelo. Um exemplo bem conhecido de sistema distribuído é a *World Wide Web*, na qual tem a aparência de um documento (uma página Web).

2.6 Aplicação Distribuída

Essa definição tem vários aspectos importantes. O primeiro é que um sistema distribuído consiste em componentes isto é, computadores autônomos. Um segundo aspecto é que os usuários, sejam pessoas ou programas, acham que estão tratando com um único sistema. Isso significa que de um modo ou de outro, os componentes autônomos precisam colaborar. Como estabelece essa colaboração com do desenvolvimento de sistemas distribuídos. Onde nenhuma premissa é adotada em relação ao tipo de computadores. Em princípio até mesmo dentro de um único sistema, eles poderiam variar desde computadores centrais (mainframes) de alto desempenho até pequenos nós em redes de sensores (TANENBAUM, 2009).

Uma outra característica importante é que usuários e aplicações podem interagir com sistema distribuído de maneira consistente e uniforme, independentemente de onde a interação ocorra. Em princípio, também deveria ser relativamente fácil expandir ou aumentar a escala de sistemas distribuídos. Essa característica é uma consequência direta de ter computadores independentes, porém, ao mesmo tempo, de ocultar como esses computadores realmente fazem parte do sistema como um todo. Em geral, um sistema distribuído estará continuamente disponível, embora algumas partes possam estar temporariamente avariadas. Os usuários e aplicações não devem perceber quais são as partes que estão sendo distribuídas ou consertadas ou quais são as novas adicionadas para atender mais usuários ou aplicações. Para suportar computadores e redes heterogêneos e, simultaneamente, oferecer uma visão de sistema único, os sistemas distribuídos

costumam ser organizados por meio de uma camada de *software* (ver Figura 3), que está situada logicamente entre uma camada de nível mais alto, composta de usuários e aplicações, e uma camada subjacente, que consiste em sistemas operacionais e facilidades básicas de comunicação, por isso tal sistema distribuído às vezes é denominado *middleware*, (TANENBAUM, 2009).

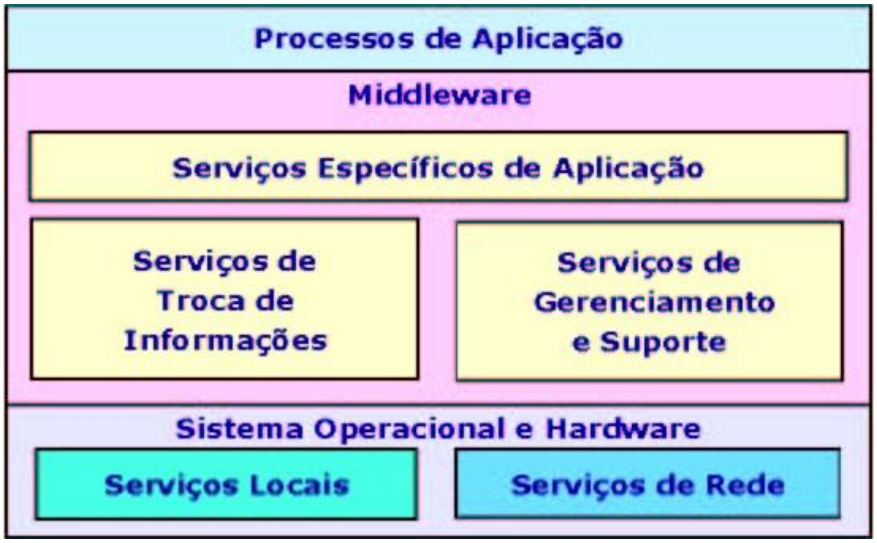

Figura 3 - Middleware em Sistemas Distribuídos

Fonte:< http://dc199.4shared.com/doc/cuAmdNLW/preview.html>

3 CONCEPÇÃO DO PROJETO

Este projeto apresenta o desenvolvimento de um sistema para controlar um robô móvel terrestre via web e capturar a temperatura e umidade do ambiente em que está atuando. Os dados coletados pelo robô através do sensor são enviados para um servidor por meio de uma placa de Arduino e um *ethernet Shield*, enquanto que outra placa de Arduino com outro *ethernet Shield* são responsáveis pelo controle do robô a partir de comandos recebidos de uma página web. Os dados são armazenados no servidor e estarão disponíveis para serem acessados a qualquer momento através de uma rede local ou externa. O protótipo construído proporciona uma simples interface de controle entre o usuário e o sistema controlador do robô móvel sendo capaz de transitar em superfícies sólidas e planas.

A primeira parte da construção do projeto está descrito no item 3.1, esta seção contempla a construção do protótipo robótico e os componentes utilizados na sua engenharia. A segunda parte é a construção da base do projeto descrito no item 3.2.

3.1 Robô Móvel

O robô móvel construído com uma base, duas placas de circuitos controladoras (Sendo uma placa de Arduino Duemilanove w/ com microcontrolador ATmega 328 e a outra placa de Arduino MEGA com microcontrolador ATmega 1280), mais detalhes na ver na seção 3.1.1, cada placa de Arduino contém um chip microcontrolador responsável pela execução do código gravado em sua memória dessa forma processando as informações pelas suas portas de entrada, tomando ações e controlando suas porta de saídas.

A comunicação do robô móvel entre computadores ou dispositivos móveis é feita através das placas de *ethernet shield Wiz5100*, permitindo a conectividade via *Internet* para projetos com Arduino. Esse chip pode suportar conexões TCP e UDP, ver na seção 3.1.2, nesse protótipo foram utilizadas duas placas de ethernet *shield Wiz5100*, pois esse chip acoplado na placa de Arduino é o responsáveis pelo deslocamento do robô móvel no ambiente através de controles enviados de um site para o Protótipo, e a outro chip acoplado em outra placa de Arduino, com sensor DHT11, desempenha a função de receber os dados e enviar para um servidor.

O DHT11 possui um complexo sensor de temperatura e umidade com uma saída de sinal digital calibrado, garantindo alta confiabilidade e estabilidade a longo prazo, ver na seção 3.1.3, responsável por capturar as temperaturas e umidades do ambiente.

O robô móvel possui três rodas: uma roda traseira, que indica a direção a ser tomada girando em todos os sentidos. As duas outras rodas são paralelas e independentes, conectadas cada uma a um eixo dianteiro, sendo estes eixos ligados cada um a um motor DC ver na seção 3.1.4, responsáveis pela tração do protótipo sendo controlado pelos microcontroladores. Todos esses componentes estão interligados por um *protoshield* descrito na seção 3.1.5.

3.1.1 Arduino

O Arduino (ver Figura 4), é uma plataforma de prototipagem eletrônica *open-source* que se baseia em *hardware* e *software* flexíveis e fáceis de usar. É destinado a artistas, designers e qualquer pessoa interessada em criar objetos ou ambientes interativos. (ARDUINO, 2013).

A placa Arduino é uma placa com Circuito Integrado (CI) de baixo custo que vem sendo muito utilizada no desenvolvimento de protótipos eletrônicos, composta por entradas e saída analógica e digital, sendo que nessas entradas podem ser conectados componentes eletrônicos para receber informações e para executar ações a partir das informações obtidas. Com a placa é possível desenvolver projetos simples (fazer um led piscar) até projetos mais complexo (automação de uma residência).

O Arduino pode perceber o estado do ambiente que o cerca por meio da recepção de sinais de sensores e pode interagir com os seus arredores, controlando luzes, motores e outros atuadores. O microcontrolador na placa é programado com a linguagem de programação Arduino, baseada na linguagem *Wiring* e o ambiente de desenvolvimento Arduino, (baseada em Processamento). Os projetos desenvolvidos com o Arduino podem ser *autônomos* ou podem comunicar-se com um computador para a realização da tarefa, com uso de *software* específico (ARDUINO, 2013).

As placas podem ser construídas à mão ou compradas e o *software* pode ser baixado gratuitamente. Os projetos de *hardware* de referência estão disponíveis

sob uma licença de código aberto, você é livre para adaptá-los às suas necessidades (ARDUINO, 2013).

De acordo com Banzi (2012), o Arduino tem dois principais componentes que são essenciais para o desenvolvimento de projetos são eles: a placa Arduino, elemento de *hardware* utilizado para a criação de projetos e a IDE que vai auxiliar na implementação dos códigos formando uma *sketch* (pequeno programa de computador), o qual será feito um upload para a placa Arduino. O programa enviado para a placa dirá que função deve ser executada.

3.1.1.1 Hardware

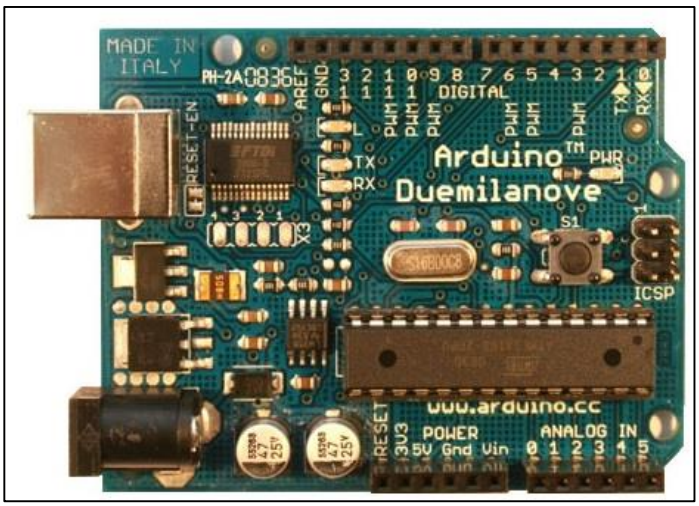

Figura 4 - Placa Aduino Duemilanove

Fonte: <http://arduino.cc/en/Main/arduinoBoardDuemilanove>

As placas de Arduino são compostas por CIs, além de um microcontrolador responsável pelo processamento das informações de acordo com o código implementado e gravados no *hardware*. Os controles e os movimentos do robô móvel são executados através dos dados gerenciados pelas portas de entrada e saída.

Para trabalhar com projetos em Arduino existem diversos modelos de placas, nesse trabalho foram utilizados dois Arduinos na construção do protótipo robótico (sendo um Arduino Duemilanove e um Arduino Mega).

O Arduino Duemilanove (2009) é uma placa de microcontrolador baseado no ATmega168 (folha de dados) ou ATmega328 (datasheet). Tem 14 pinos digitais de

entrada / saída (dos quais 6 podem ser usados como saídas PWM), 6 entradas analógicas, um cristal oscilador de 16 MHz, uma conexão USB, um conector de alimentação, um cabeçalho ICSP, e um botão de reset. (ARDUINO, 2013).

Mais detalhes sobres as especificações do Arduino Duemilanove está representado na Tabela 1.

Tabela 1 - Especificações da placa Arduino Duemilanove

Microcontrolador	ATmega168
Tensão de funcionamento	5V
Tensão de entrada (recomendado)	7-12V
Tensão de entrada (limites)	6-20V
Digital I / O Pins	14 (dos quais 6 oferecem saída PWM)
Pinos de entrada analógica	6
Corrente DC por I / O Pin	40 mA
De corrente DC para 3.3V Pin	50 mA
Memória Flash	16 KB (ATmega168) ou 32 KB (ATmega328), dos quais 2 KB usados por bootloader
SRAM	1 KB (ATmega168) ou 2 KB (ATmega328)
EEPROM	512 bytes (ATmega168) ou 1 KB (ATmega328)
Velocidade do relógio	16 MHz

O Arduino Mega (Figura 5) é uma placa de microcontrolador baseado no ATmega1280 (folha de dados). Ele tem 54 pinos de entrada / saída digital (dos quais 14 podem ser usados como saídas PWM), 16 entradas analógicas, 4 UARTs (portas seriais de *hardware*), um cristal oscilador de 16 MHz, uma conexão USB, um conector de alimentação, um cabeçalho ICSP, e um botão de reset. Ele contém tudo o necessário para apoiar o microcontrolador, basta conectá-lo a um computador com um cabo USB ou ligá-lo com um adaptador AC ou bateria-to-DC para começar. O mega é compatível com a maioria dos shields projetados para o Arduino Duemilanove ou Diecimila (ARDUINO, 2013).

As especificações do Arduino Mega encontra-se mais detalhadas na Tabela 2.

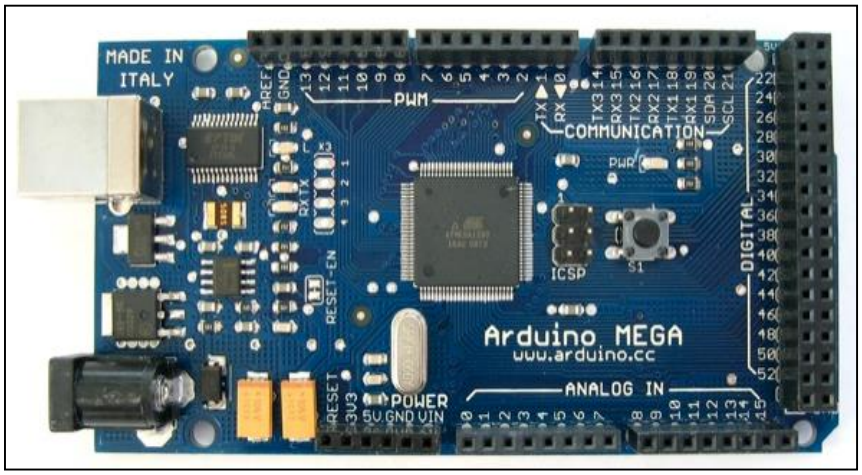

Figura 5 - Placa Aduino Mega 1280

Fonte: <http://arduino.cc/en/Main/arduinoBoardMega>

Tabela 2 - Especificações da placa Arduino Mega 1280

Microcontrolador	ATmega1280
Tensão de funcionamento	5V
Tensão de entrada (recomendado)	7-12V
Tensão de entrada (limites)	6-20V
Digital I / O Pins	54 (dos quais 15 oferecem saída PWM)
Pinos de entrada analógica	16
Corrente DC por I / O Pin	40 mA
De corrente DC para 3.3V Pin	50 mA
Memória Flash	128 KB, dos quais 4 KB usados por bootloader
SRAM	8 KB
EEPROM	4 KB
Velocidade do relógio	16 MHz

3.1.1.2 Software

Segundo o (Arduino, 2013), a IDE é um ambiente de código fonte aberto facilitando escrita de código e enviando para a placa i/o. Além encontrar o *software* em várias versões e compatível com as seguintes plataformas: Windows, Mac OS X e Linux. O ambiente é escrito em Java e baseado em Processing, avr-gcc. A IDE utilizada nesse projeto foi 1.0 do Arduino para ambiente Windows de 32 bits, (ver Figura 6).

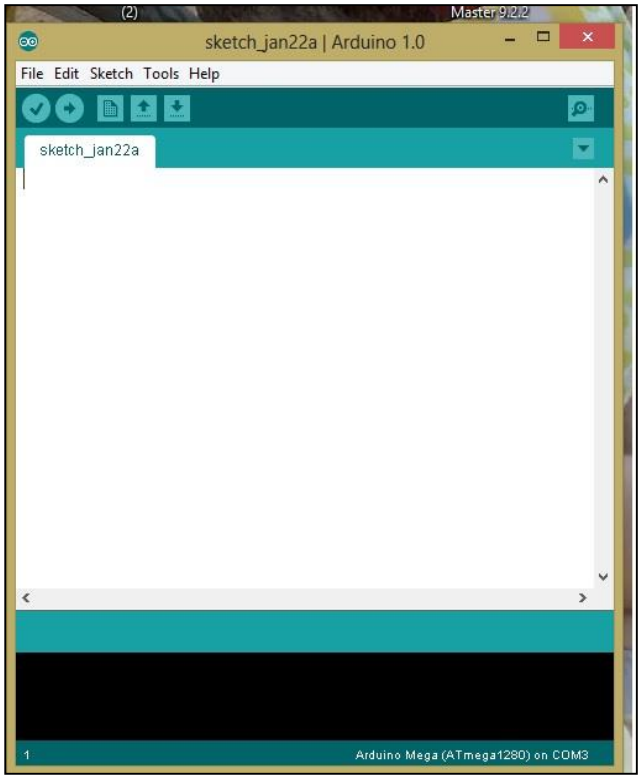

Figura 6 - IDE do Arduino versão 1.0

Fonte: o Autor

O ambiente de desenvolvimento Arduino contém um editor de texto para o código escrito, uma área de mensagem, um console de texto, uma barra de ferramentas com botões para funções comuns, e uma série de menus. Ele se conecta ao *hardware* Arduino para carregar programas e comunicar com eles (ARDUINO, 2013).

Os Programas escrito usando a IDE Arduino são chamados de sketches. Estes esboços são escritos no editor de texto. Os sketches são salvos com a extensão de arquivo. Ino. Ele tem uma barra de ferramentas com as opções de corte, colar,

recorta, procurar e substituir o texto e também exibe erros. O console exibe saída de texto pelo ambiente Arduino, incluindo mensagens de erro completa e outras informações. O canto direito inferior da janela exibe a *board* que está sendo usada no momento e a porta serial. Os botões da barra de ferramentas permitem que você verifique e faça upload de programas, possibilita a criação de novos *sketches*, abrir *sketches* já salvos e o serial monitor (ARDUINO, 2013).

A IDE do Arduino é similar as linguagens de programação C/C++, conservando toda uma sintaxe em aspectos como: declaração de variáveis, utilização de operadores, manipulação de vetores, na conservação da estrutura, mas ao contrário das linguagens citadas ela não utiliza uma função *main(),* o Aduino é composto por duas funções elementares, sendo elas: *setup()* e *loop()* (ARDUINO, 2013).

3.1.2 Ethernet Shield W5100

O Ethernet Shield com uma placa Arduino permite que seja possível a conexão com à *internet* (ver Figura 7.). Ele baseia-se na Wiznet W5100 chip de ethernet (folha). O Wiznet W5100 fornece uma rede (IP) pilha composta pelos protocolos TCP e UDP. Ele suporta até quatro conexões de soquete simultâneos. Usando a biblioteca Ethernet ela permite escrever esboços que se conectam à *internet* usando o escudo. O escudo ethernet conecta a uma placa Arduino usando cabeçalhos fio de finalização longos que se estendem através do Ethernet Shield. Isso mantém a pinagem intacta e permite que outro escudo seja empilhados em cima (ARDUINO, 2013).

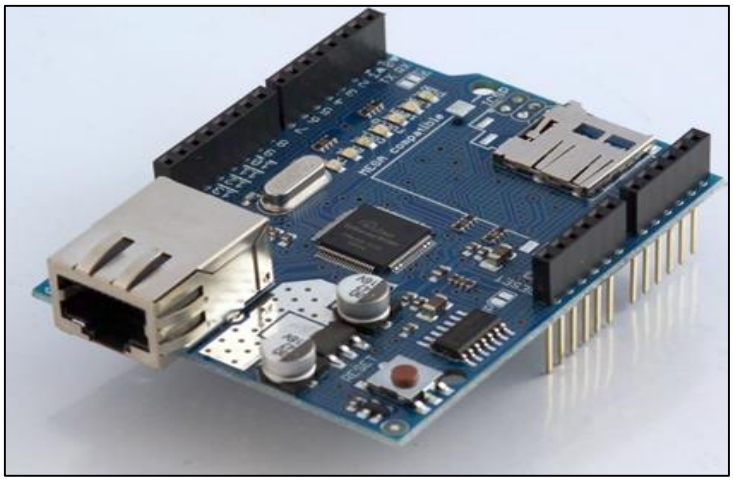

Figura 7 - Escudo *Ethernet Shield* W5100

Fonte: http://arduino.cc/en/Main/arduinoEthernetShield

O *Ethernet Shield* possui um slot para cartão micro-SD acoplado, que pode ser usado para armazenar arquivos para servir através da rede. É compatível com a Arduino Uno e mega (usando a biblioteca de Ethernet). O leitor de cartão microSD onboard é acessível através da Biblioteca SD (ARDUINO, 2013).

O *Ethernet Shield* também inclui um controlador de reset, para garantir que o módulo W5100 *Ethernet* seja devidamente reposto o código. Revisões anteriores do escudo não eram compatíveis com o mega e precisa ser reposto manualmente após a inicialização (ARDUINO, 2013).

O escudo tem um poder módulo Ethernet (PoE) projetado para extrair energia a partir de um cabo Ethernet convencional par trançado categoria 5 mais:

- IEEE802 0,3 af compatível
- Ondulação de baixa potência e ruído (100mVpp)
- Faixa de tensão de entrada de 36V a 57V
- Sobrecarga e proteção contra curto-circuito
- 9V saída
- Alta eficiência DC / DC conversor: tip 75% a carga de 50%
- Isolamento 1500V (entrada à saída)

Arduino se comunica tanto com o W5100 e cartão SD usando o barramento SPI (através do cabeçalho ICSP). Este é nos pinos digitais 10, 11, 12 e 13 sobre o Uno e os pinos 50, 51, e 52 no mega. Em ambas as placas, pino 10 é usado para selecionar o W5100 e o pino 4 para o cartão SD. Estes pinos não pode ser usado para

geral I / O. Sobre os Mega, o pino SS *hardware*, de 53 anos, não é usado para selecionar o W5100 ou o cartão SD, mas deve ser mantido como uma saída ou a interface SPI não vai funcionar (ARDUINO, 2013).

O escudo fornece um conector Ethernet padrão RJ45.

O botão reset nas redefinições escudo tanto o W5100 e a placa Arduino.

O escudo contém uma série de LEDs informativos:

- PWR: indica que a placa e o escudo são alimentado
- LINK: pisca quando a comunicação é estabelecida
- FULLD: indica que a conexão de rede é full duplex
- 100M: indica a presença de uma conexão de 100 Mbs de rede (em oposição a 10 Mbs)
- RX: pisca quando o escudo recebe dados
- TX: pisca quando o escudo envia dados
- COLL: pisca quando são detectadas colisões de rede

O jumper de solda marcado "INT" pode ser conectado para permitir que a placa Arduino receber a notificação de interrupção de eventos do W5100, mas isso não é suportado pela biblioteca Ethernet. O jumper conecta o pino INT do W5100 ao pino digital 2 do Arduino (ARDUINO, 2013).

3.1.3 O Sensor DHT11

O DHT 11 é um sensor de temperatura, umidade e ponto de orvalho DHT11, (ver Figura 8), complexo com uma saída de sinal digital calibrado. Ao utilizar o exclusivo digital de aquisição de sinal técnica e da temperatura e tecnologia de detecção de umidade, que garante alta confiabilidade e excelente estabilidade a longo prazo. Este sensor inclui uma medição de umidade do tipo resistiva em um componente de medição de temperatura NTC, e conecta-se a um maior microcontrolador 8-bit do desempenho, oferecendo excelente qualidade, resposta rápida, anti-interferência capacidade e custo-efetividade (D-ROBOTICS). Este trabalho foi desenvolvido com o sensor.

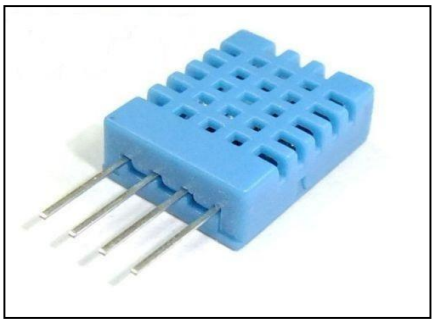

Figura 8 - Sensor DHT11

Fonte:http://www.arduinoecia.com.br/2013/05/sensor-de-umidade-e-temperatura dht11.html

Os coeficientes de calibração são armazenados como programas na memória OTP, os quais são utilizados pelo processo de detecção do sinal interno do sensor. A interface serial-fio único faz a integração do sistema rápido e fácil. Seu pequeno tamanho, baixo consumo de energia e up-to-20 transmissão de sinal tornando-se a melhor escolha para várias aplicações, incluindo aquela mais exigentes (D-ROBOTICS).

O DHT11 é um sensor de temperatura e umidade, que permite medir temperaturas de 0 a 50 celsius, e umidade na faixa de 20 a 90 %, e o ponto de orvalho temperatura à qual o vapor de água presente no ar ambiente passa ao estado líquido na forma de pequenas gotas (chuva). Não é um sensor extremamente rápido e preciso, por isso não é recomendada a utilização em ambientes de alto risco. Sua faixa de precisão para temperatura é de 2 graus, e de umidade, 5%. O sensor em si tem 4 pinos, mas o pino 3 não é utilizado (ARDUINO E CIA). Como mostra a Tabela 3.

Tabela 3 - Especificação do sensor DHT11

Especificação do Pinos	
Pino 1	VCC (5V)
Pino 2	Sinal
Pino 3	NC Não conectado
Pino 4	GND
Especificações do sensor	
Dimensões	23mm x 12mm x 5mm (incluindo terminais)
Alimentação	3,0 a 5,0 VDC (5,5 Vdc máximo)
Corrente	200uA a 500mA, em stand by de 100uA a 150 uA
Faixa de medição de umidade	20 a 90% UR
Faixa de medição de temperatura	0º a 50ºC
Precisão de umidade de medição	± 5,0% UR
Precisão de medição de temperatura	± 2.0 ºC
Tempo de resposta	< 5s

3.1.4 Motores DC

Um motor de corrente contínua é uma comutado mecanicamente elétrico alimentado por corrente contínua (CC). O estator é estacionária no espaço por definição, e, portanto, a corrente do rotor é comutado pelo comutador para ser também estacionária no espaço (ver Figura 9). Esta é a forma como o ângulo relativo entre o estator e rotor de fluxo magnético é mantida perto de 90 graus, o que gera o binário máximo (CENTRO DE REFERENCIA PARA ESTUDO DA FISICA, 2013).

O funcionamento de um motor DC está fundamentado nos princípios de atração e repulsão dos polos magnéticos, fluxo magnético e indução de tensão elétrica. O presente projeto dispõem de dois motores DC responsáveis pela tração do robô móvel girando nos dois sentido para frente e para traz.

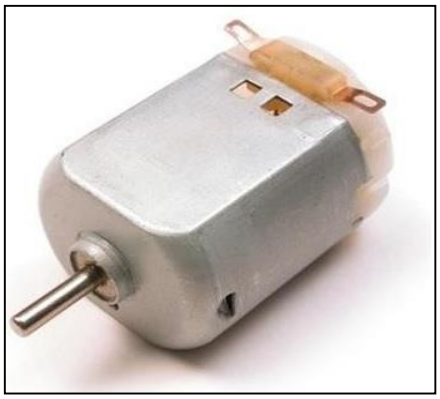

Figura 9 - Motor DC

Fonte: o Autor

Para se utilizar um motor DC em dispositivos eletromecânicos não é preciso muito. Podemos obter motores de diversos tipos ou de aparelhos comuns fora de uso como em casas especializadas, o que simplifica bastante o trabalho do projetista de Robótica, Mecatrônica ou Automatismos diversos. Entretanto, é preciso saber trabalhar com estes pequenos motores respeitando suas características elétricas e mecânicas. Utilizando-se caixas de redução apropriadas e controles de velocidade e sentido, é possível obter qualquer tipo de movimento com estes motores, desde os mais suaves até o movimento rápido de propulsão de um veículo controlado à distância (CENTRO DE REFERENCIA PARA ESTUDO DA FISICA, 2013).

O motor em questão opera com ímã permanente e, portanto, a intensidade da indução magnética aplicada sobre o rotor com o enrolamento (espiras condutoras) não dependa da fonte de alimentação (CENTRO DE REFERENCIA PARA ESTUDO DA FISICA, 2013). As especificações do motor estão contida na Tabela 4.

Tabela 4 - Especificação do Motor DC

Voltagem:	3-6VDC
Corrente:	80-150mA
Rotação sem carga:	3V-125RPM, 5V-200RPM, 6V-230RPM
Rotação com carga:	3V-95RPM, 5V-160RPM, 6V-175RPM
Torque de saída:	3V-0.8kg.cm, 5V-1.0kg.cm, 6V-1.1kg.cm
Relação de Redução:	1/120
Diâmetro da roda incluindo o pneu:	65mm
Largura da roda:	25mm
Dimensão do motor/caixa de redução:	20mm x 22mm x 65mm
Peso:	50g
Ruído:	<65dB

3.1.5 Arduino Protoshield

O Arduino Protoshield faz com que seja fácil projetar circuitos personalizados (ver Figura 10). Você pode soldar peças para a área de prototipagem para criar seu protótipo, ou usá-lo com uma pequena placa de ensaio sem solda, para testar rapidamente ideias com circuito sem ter que usar soldar. Ele tem conexões extras para todos os pinos do Arduino I/O integrados. É uma maneira conveniente de fazer o seu circuito personalizado e Arduino em um único módulo (ARDUINO, 2013).

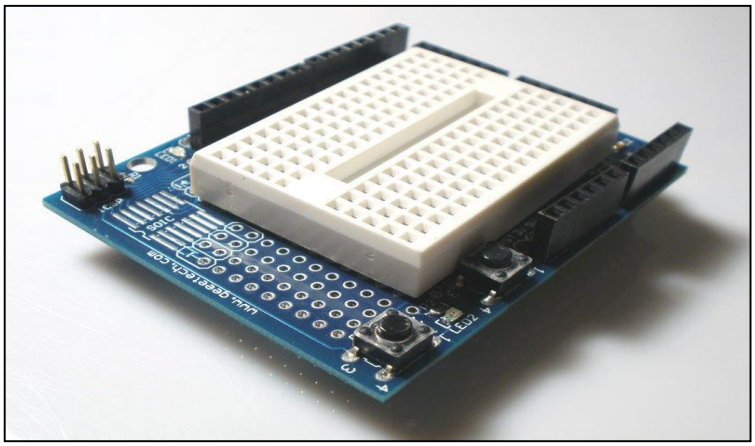

Figura 10 - Protoshield

Fonte: http://www.lib.sfu.ca/node/11945

O comprimento máximo e a largura do PCB ProtoShield são de 2,7 e 2,1 centímetros, respectivamente. Três orifícios de parafuso permitem o escudo a ser ligado a uma superfície. Note-se que a distância entre os pinos digitais 7 e 8 é de 160 mil (0,16 "), e não um múltiplo do espaçamento 100 mil dos outros pinos.

O conector ICSP disponível no shield pode ter suas conexões feitas diretamente aos pinos SPI (ARDUINO, 2013), mais detalhes ver na Tabela 5.

Tabela 5 - Especificações do Protoshield

1:	MISO conectado a D12
2:	+5 V 3:
3:	SCK ligado a D13
4:	MOSI ligado a D11
5:	SS ligado D10
6:	GND

3.2 Base do Robô Móvel

Nela são acoplados os componentes citados neste capítulo. O Robô Móvel do tipo carrinho (ver Figura 11), ele utiliza um chassis TG007, ele é alimentado por três baterias sendo duas de 7,4 volts, uma para o microconntrolador Arduino ATmega 1280 e outra para o microcontrolador Arduino Duemilanove e outra de 9,0 volts para

o roteador que vai estar em cima do carrinho pronto para estabelecer a comunicação local ou web.

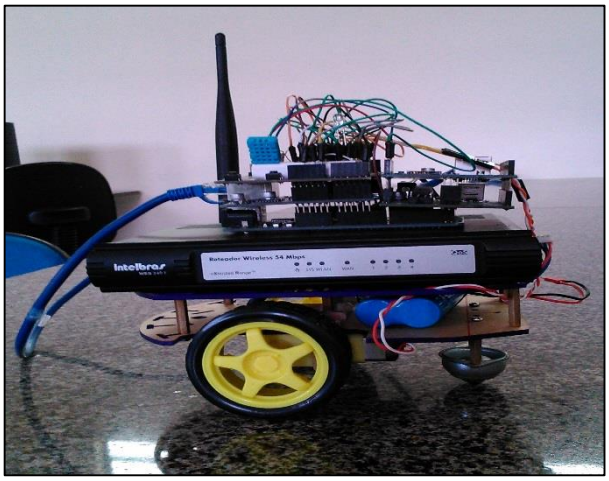

Figura 11 - Robô móvel
Fonte: o Autor

3.2.1 O Notebook

Com o notebook foi desenvolvido toda a programação do Arduino e envido os códigos para as placas. O computador usado foi com processador Dual-Core P6200 4 GB de RAM 500 GB de HD, o projeto foi iniciado com o Windows 7 Ultimate de 32 bits, depois foi instalado o Windows 8 Interpriser de 64 bits. A escolha desse sistema não vai implicar no funcionamento do projeto ela é feita por escolha própria.

3.2.2 O PHP

PHP é uma linguagem poderosa e um interpretador, seja incluído em um servidor web como um módulo ou executado separadamente como binário CGI, é possível acessar arquivos, executar comandos e abrir conexões de rede no servidor.

PHP é desenhado especificamente para ser uma linguagem mais segura para escrever programas CGI que Perl ou C, e com a escolha correta de opções de configuração em tempo compilação ou de execução, e práticas corretas de programação, ela pode dar a combinação exata de liberdade e segurança que você precisa, (MANUAL DO PHP,2014).

A flexibilidade de configuração do PHP é comparável com a flexibilidade de código. PHP pode ser usado para montar um servidor de aplicações completo, com todo o poder de um usuário shell, ou pode ser usado para inclusão simples de arquivos no lado do servidor com pouco risco em um ambiente bem controlado. Como montar o ambiente, e o quão seguro ele é, depende muito do desenvolvedor, (MANUAL DO PHP,2014).

4 IMPLEMENTAÇÃO

As informações abordadas neste capitulo são referentes a construção do protótipo robótico até o desenvolvimento da página web, desde de os primeiros teste até a sua versão final.

4.1 Implementando o Protótipo Robótico

Logo de início foi adquirid**o** o chassi com todos os outros acessórios como: rodas, motores, sensores, Microcontroladores, *protoshield, Shield Ethernet* para que sejam utilizados de acordo com as funções requisitadas.

A primeira fase da implementação do protótipo utilizava uma placa *Shield Enthernet enj2860* e um microcontrolador Arduino Duemilanove /W ATmega328, depois de algumas pesquisas e testes realizados ficou claro que o *Ethernet Shield enj2860,* continha um número de instruções reduzidas, portanto ela *n*ão seria capaz de suportar comunicações com soquetes, também existiam problemas de compatibilidades com as bibliotecas utilizadas e o código do HTML era programado dentro do código do Arduino. Para resolver o problema com o *Ethernet Shield enj2860* foi necessário substitui-lo por uma nova placa *Ethernet ShieldW5100 da wiznet.*

Tabela 6 - Configurando o shield ethernet com o Arduino e declaração de variáveis

1	#include <SPI.h>
2	#include <Ethernet.h>
23	byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
24	byte ip[] = { 10,180,16,120};
30	char Luz[7] = "0000L#";
31	EthernetServer server(8080);
35	char msg[7] = "0000L#";

Na segunda fase da implementação do protótipo foi utilizado uma placa *Ethernet ShieldW5100* e um microcontrolador Arduino Duemilanove /W ATmega328,

responsáveis pelos controles do robô móvel através da página em PHP, esse *Ethernet ShieldW5100* disponibiliza até quatro comunicações simultâneas via soquetes, utiliza a biblioteca *SPI.h,* ela é um protocolo para fazer a comunicação entre o microcontrolador Arduino e o *Ethernet ShieldW5100.* Já a biblioteca *Ethernet.h*. foi desenvolvida para funções de comunicação com a *internet*, para que essa comunicação seja estabelecida é necessária de uma variável pra receber o endereço MAC na linha 23 da Tabela 6 e outra para receber o endereço IP. Na linha 30 foi declarado uma variável *char luz[7]* = “0000L#” um vetor de caracteres que representa em que estado o dispositivo vai estar mandando ou não sinal, a linha 31 vai criar um servidor na *porta 8080* e na linha 35 foi declarada outra variável *msg* [7] que é outro vetor de cárteres responsável por armazenar as mensagens recebidas.

A Tabela 7 mostra parte do código onde o Arduino é responsável, por saber qual direção o motor vais girar. Na linha 60 foi estabelecida uma condição que quando receber um caractere manda de volta o status da variável se foi acionado ou não e repassar dentro da funções referente frente ou ré, na linha 61 o *switch* vai auxiliar na criação do menu nas linhas 63 e 68, na linha 97 a função *frente()* aciona os pinos do motor esquerdo como a velocidade, quando o motor receber 0 ele para, recebendo 1 girar. O mesmo acontece dentro da função *re()* apenas invertendo os pinos mostrados nas linha 99,100,102,103 e nas linas108,109, 111 e 112. As informações processadas representadas nessa tabela vão ser enviados na página PHP obtendo comunicação via soquetes, ver na Tabela 7.

Tabela 7 - Trecho responsável pelo Controles do motores

60	if (msg[6]=='#') {
61	switch(msg[5]) {
63	case 'R': {
64	client.write(var);
65	break;
66	}
68	case 'F': {
69	frente();
70	break;
71	}
97	void frente(){
98	analogWrite(pinMotorEsq, velE);
99	digitalWrite(pinELE, 1);
100	digitalWrite(pinELD, 0);
101	analogWrite(pinMotorDir, velD);
102	digitalWrite(pinDLE, 1);
103	digitalWrite(pinDLD, 0);
104	}
106	void re(){
107	analogWrite(pinMotorEsq, velE);
108	digitalWrite(pinELE, 0);
109	digitalWrite(pinELD, 1);
110	analogWrite(pinMotorDir, velD);
111	digitalWrite(pinDLE, 0);
112	digitalWrite(pinDLD, 1);
	}

Para controlar os motores foi necessário a declaração de variáveis que vão ser utilizadas para especificar quais pinos vão estar conectados aos motores mostrados nas linha 9,10,11,13,14,15, pois os mesmos irão trabalhar de forma independente um do outro. Dispondo da declaração de mais duas variáveis para controlar a velocidade da rotação dos motores representado nas linhas 17 e 18. Ver na Tabela 8.

Tabela 8 - Declaração de pinos dos motores e pinos responsáveis pela rotação.

9	int pinMotorEsq = 5;
10	int pinELE = 4;
11	int pinELD = 3;
13	int pinMotorDir = 9;
14	int pinDLE = 8;
15	int pinDLD = 6;
17	int velE = 220;
18	int velD = 220;

Depois de mais alguns testes, novamente um outro problema foi encontrado quando era requisitado da página web mais de quatro comandos a conexão via soquetes era perdida, logo surgiu a necessidade de outra substituição, mas dessa vez foi o microcontrolador Arduino Duemilanove /W ATmega328, pois para esse projeto tinha que usar uma placa com um número de memória maior e com mais instruções, então foi substituído por um o microcontrolador Arduino Mega ATmega1280, capaz de atender todos os comandos requisitados pela página web.

Na terceira fase da implementação do protótipo foi utilizado uma outra placa *Ethernet ShieldW5100* e um microcontrolador Arduino Duemilanove /W ATmega328 com um sensor de temperatura, umidade e ponto de orvalho, o DHT11 que usa a biblioteca *dht11.h* como mostrada na linha 3 da Tabela abaixo essa biblioteca pré-escrita desenvolvida especifica para esse *Shield.* A biblioteca *dht11.h* cuida de todas as funções necessárias para calcular a temperatura e a umidade e ainda o ponto de orvalho, é necessário a utilização das bibliotecas da Tabela 9, bastando apenas mudar o endereço IP sem alterar o endereço MAC para que a comunicação seja estabelecida e capaz de enviar dados na página web. Nessa

terceira fase o endereço IP que vai ser atribuído ao Arduino vai ser o IP do servidor ou seja endereço que o computador receber quando conectado em uma rede, e na linha 5 é o pino onde o sensor vai ser conectado no Arduino.

Tabela 9 - Configurando o shield ethernet com o Arduino e DHT11

1	#include <Ethernet.h>
2	#include <SPI.h>
3	#include <dht11.h>
5	#define DHT11PIN 2
6	byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
7	IPAddress server(10,180,14,138);

Na linha 15 da Tabela 10 existe uma condição para estabelecer uma comunicação com a *internet* da função setup (). Nesse caso quando o endereço MAC for igual a 0, não vai ser estabelecida nenhuma conexão. Quando o endereço MAC for diferente de zero a comunicação é estabelecida. Dentro da função loop () na linha 22 foi criada uma condição que a comunicação foi estabelecida no servidor com a porta 8080.

Tabela 10 - Função setup () e função loop () e condições para conexão com a *internet*

11	void setup()
13	{
15	if (Ethernet.begin(mac) == 0) {
	}
17	delay(1000);
	}
20	void loop(){
22	if (client.connect(server, 8080)){

Na linha 26 da Tabela 11 foi criado uma variável para ler o pino onde o sensor está conectada e nas linhas28,29 e 30 são declaradas variáveis para leitura da temperatura, da umidade e ponto de orvalho. Já na linha 31 é estabelecida uma condição para quando a variável *client* estiver conectado manda dados das linhas 33,34,35,36,37,38,39 na página em PHP através da linha 41 esses dados são enviados para a arquivo *gravar.php*

Tabela 11 - Variáveis do sensor e conexão com PHP

26	int chk = DHT11.read(DHT11PIN);
28	int hum=DHT11.humidity;
29	int temp=DHT11.temperature;
30	int dew=DHT11.dewPoint();
31	if (client.connected()){
33	client.print(hum,DEC);
34	client.print("&temp=");
35	client.print(temp,DEC);
36	client.print("&dew=");
37	client.print(dew,DEC);
38	client.println(" HTTP/1.1");
39	client.println("Host: arduino1");
40	client.println();
41	client.print("GET /dht11_php_funcionando/gravar.php?hum=");

A Tabela 12 representa a comunicação do Arduino com PHP. Onde na linha 6 são criado os soquetes para estabelecer a conexão, a linha 8 é a parte em que o soquete vai se conectar ao IP e a porta, os endereços IP e a porta são os mesmo do Arduino caso aconteça de colocar os endereços diferentes o soquete não estabelece a comunicação ocasionando um erro na página PHP. Da linha 10 até a linha 15 são as condições dos comandos recebidos do Arduino referente aos controles do robô móvel na página *web* executando as ações correspondentes aos botões pressionados.

Tabela 12 *- Trecho do código em PHP que recebe comunicação do Arduino*

6	$sock = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
8	socket_connect($sock,'10.180.15.108', 8081);
10	if(isset($_POST['bits'])) {
11	$msg = $_POST['bits'];
12	if(isset($_POST['Frente'])){ $msg = 'F#'; }
13	if(isset($_POST['Re'])){ $msg = 'T#'; }
14	if(isset($_POST['Esquerda'])){ $msg = 'E#'; }
15	if(isset($_POST['Parar'])){ $msg = 'P#'; }
	}
18	socket_write($sock,'R#',2); //Requisita o status do sistema.
19	$status = socket_read($sock,6);
	}

No presente projeto foi desenvolvido um banco de dados no phpmyadmin utilizado para armazenar os dados capturados pelo sensor, transmitindo na página onde fica os controles do robô móvel. O trecho do código responsável por estabelecer a conexão com o banco de dados ver na Tabela 13.

Tabela 13 - Trecho do código responsável pela conexão com o banco de dados

3	$local_serve = "localhost";
4	$usuario_serve = "root";
5	$senha_serve = "";
6	$banco_de_dados = "bancoth";
8	$conn = @mysql_connect($local_serve,$usuario_serve,$senha_serve) or die ("O servidor não responde!");
11	$db = @mysql_select_db($banco_de_dados,$conn) or die ("conectado ao banco de dados!");

A linha 14 da Tabela 14 vai ser responsável por inserir os dados no banco enviados pelos sensor, enquanto que a linha 16 vai ficar atualizando esses dados na página *web,* esse trcho de código e do arquivo *gravar.php* que recebe as informações mandadas pelo Arduino.

Tabela 14 - Trecho do código responsável por armazenar os dados do sensor no banco de dados

14	mysql_query("INSERT INTO dados (Temperatura, Umidade, Ponto_Orvalho, Data, Hora) VALUES (".$values['temp'].",".$values['hum'].",".$values['dew'].",'".$data."','".$hora."')");
16	mysql_query("UPDATE atualiza SET Temperatura=".$values['temp'].",Umidade=".$values['hum'].",Ponto_Orvalho=".$values['dew'].""); fwrite($file,"Data: ".$data . ' \| '. "Hora: ".$hora . ' | Temp: ' . $values['temp'] . ' | Humi: ' . $values['hum'] . ' \| P. Orva: '.$values['dew'])."\n";

Esquema todos os pinos dos *Shiedl Ethernet W5100* (ver Figura 12) ligados ao Arduino sendo acoplado uma placa na outra e suas referentes conexões.

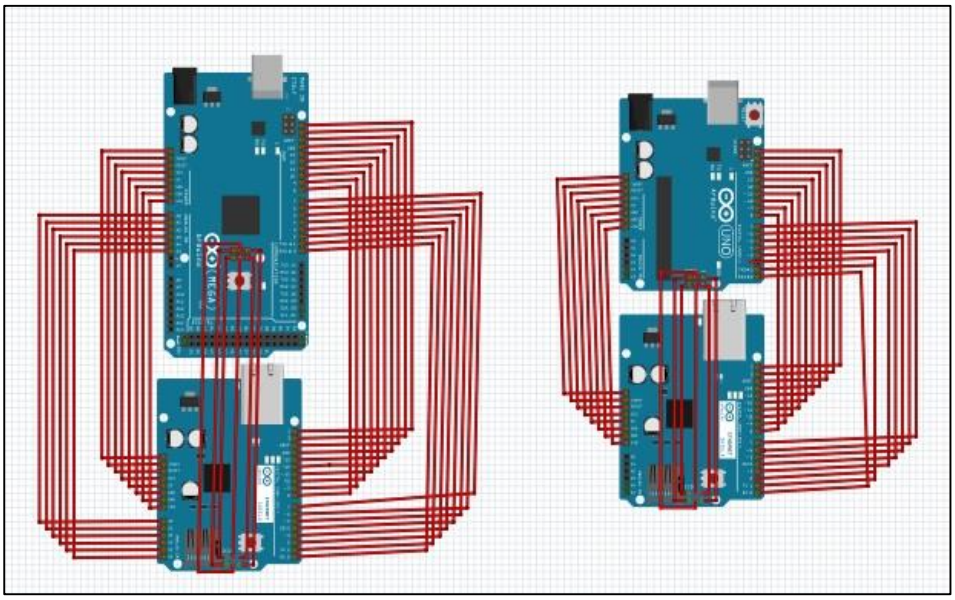

Figura 12 - Esquema de Ligação entre os Arduinos com os Shiedl Ethernet

Fonte: o Autor

Esquema completo do *ProtoShield* conectados ao *Shield Ethernet* (ver Figura 13), também as ligações do motores e do sensor de DHT11 no outro *Shield Ethernet.*

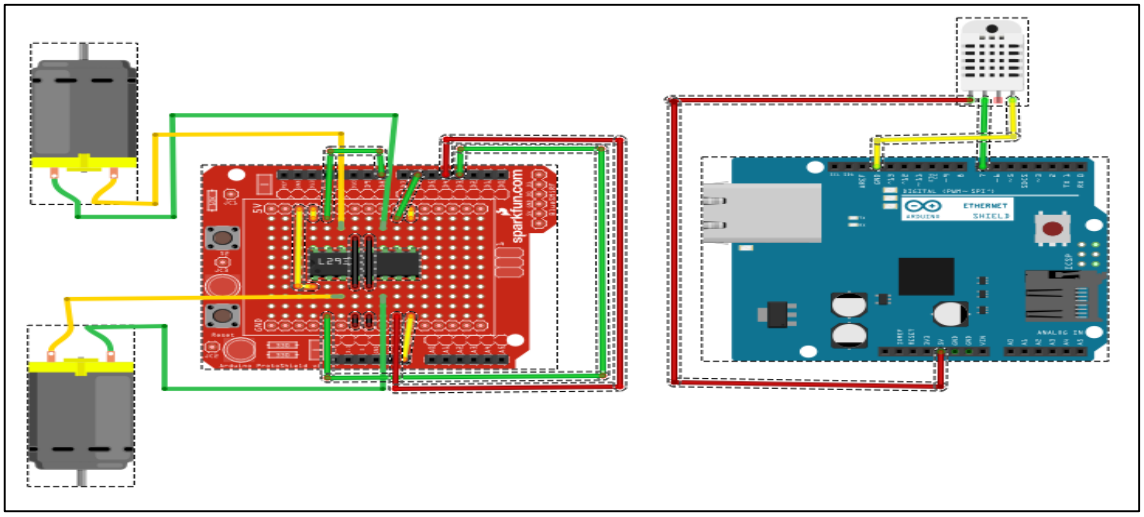

Figura 13 - Esquema completo de Ligação dos componentes

Fonte: o Autor

Aplicação *web* na sua página inicial contendo um título (ver Figura 14), um link para *login* do usuário e um menu com três opções: *início, sobre o projeto e Lippo.* Clincando sobre a opção *login* irá abrir uma tela para o usuário inserir login e senha, depois de logar serão disponibilizados os controles do robô, a temperatura e umidade capturadas pelo robô móvel. Ao clicar na opção *sobre o projeto* vai conter uma breve

descrição do projeto, clicando na opção *Lippo* será mostrado Informações sobre o Laboratório de Investigação e Pesquisa em Poéticas Digitais.

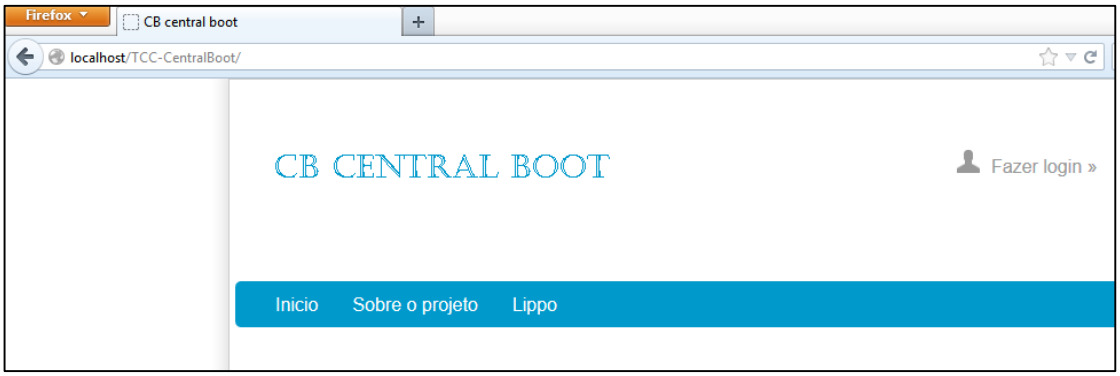

Figura 14 - Aplicação *web*

Fonte: o Autor

Página *web* em modo funcional com os botões em forma de setas representando os controles do robô móvel indicando o sentido em que vai se movimentar e no meio tem o botão parar responsável por parar qualquer ação dos outros botões (ver Figura 15).

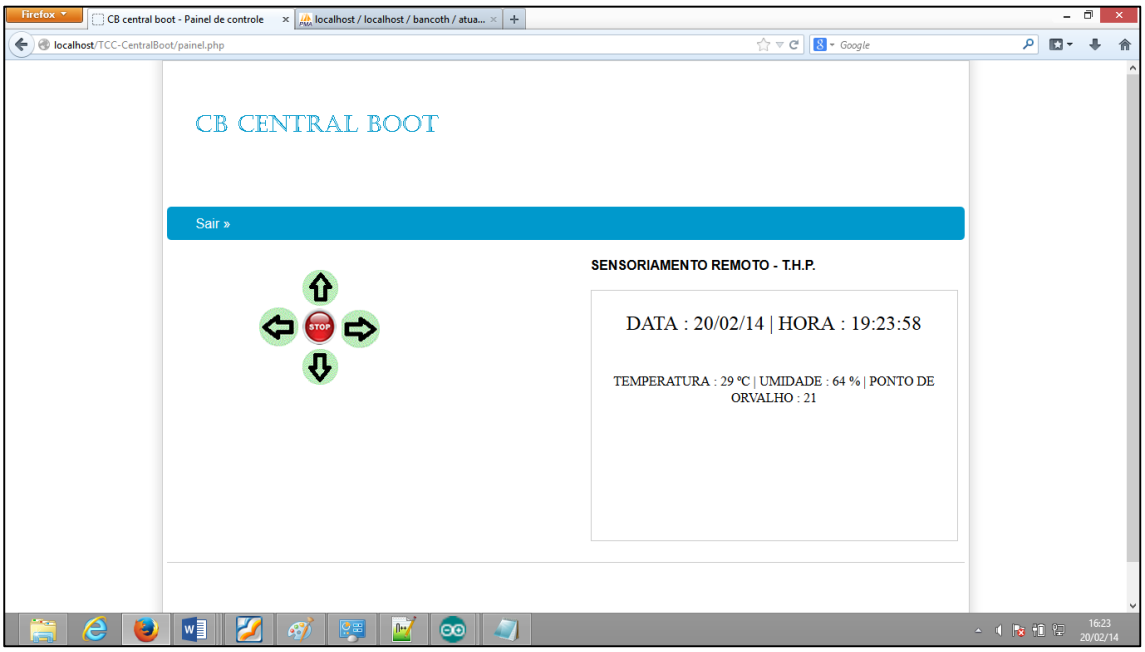

Figura 15 - Aplicação *web* Central *Boot*

Fonte: o Autor

O protótipo já concluído (ver Figura 16) pois o chassi com as rodas foram comprados protos apenas foram montados, acrescentado o roteador e as Placas de Arduino com os *Shiedls Ethernet.*

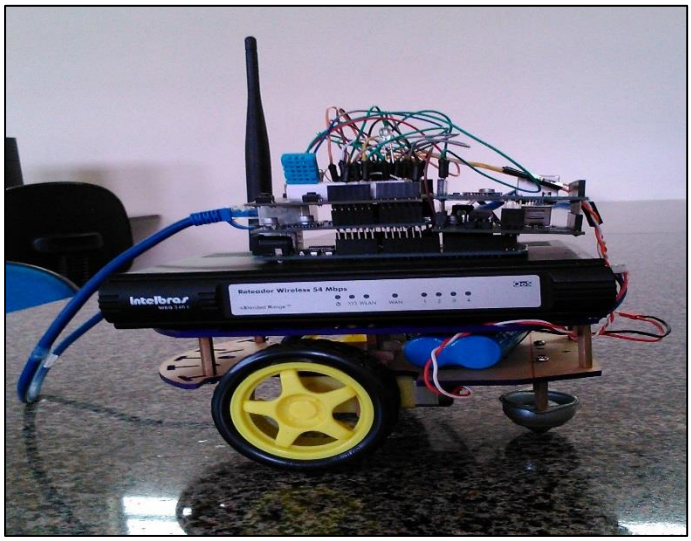

Figura 16 - Protótipo Concluído

Fonte: o Autor

5 RESULTADOS

Os testes foram realizados para verificar a eficiência da aplicação *web* determinado a ação de cada botão de acordo com a direção em que o robô móvel vai se deslocar ainda com uma interface precária em HTML com PHP em uma rede local aplicando qualquer IP da rede para o *Shield Ethernet*.

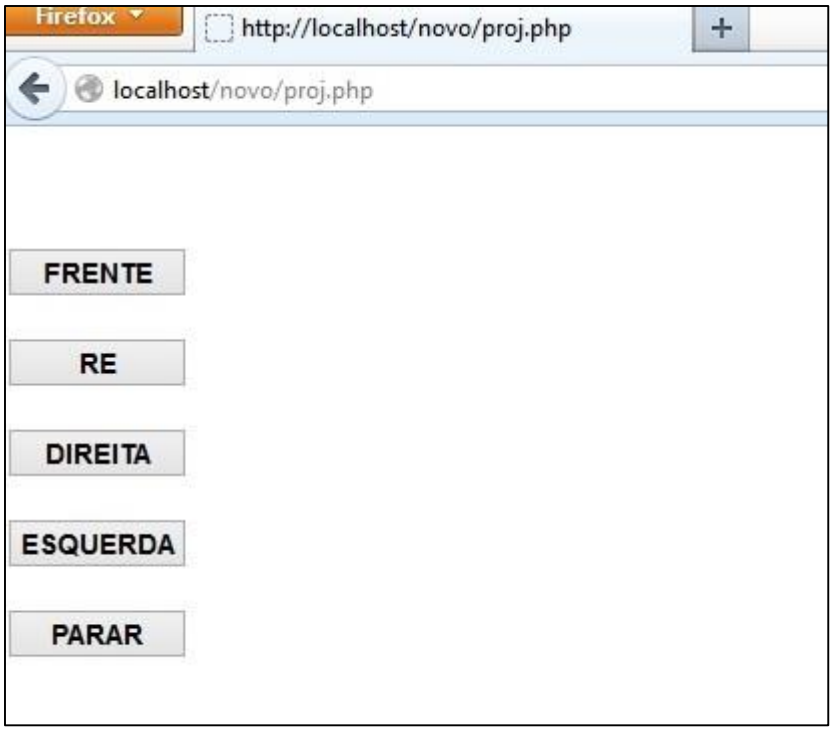

Figura 17 Controle do robô móvel

Fonte: o Autor

Nessa primeira fase de implementação ver (Figura 17), foram observados os seguintes componentes as placas utilizadas e comunicação e o *WampServer* para testes com aplicações *web*, sempre ocorria um erro quando comunicação da extensão *php_socket* no *Wampserver* não eram habilitada, logo depois veio a observação com a placa de Arduino Duemilanove e Ethenet *Shield* , sempre que eram acionados mais de 4 botões a comunicação era perdida entre o Arduino e aplicação *web*, Surgindo assim a necessidade de substituir a placa Arduino Duemilanove por uma Placa Aduino Mega 1280, onde os controles começaram a funcionar corretamente obedecendo o tempo de resposta de acordo com a solicitação envidada entre *software e hardware.*

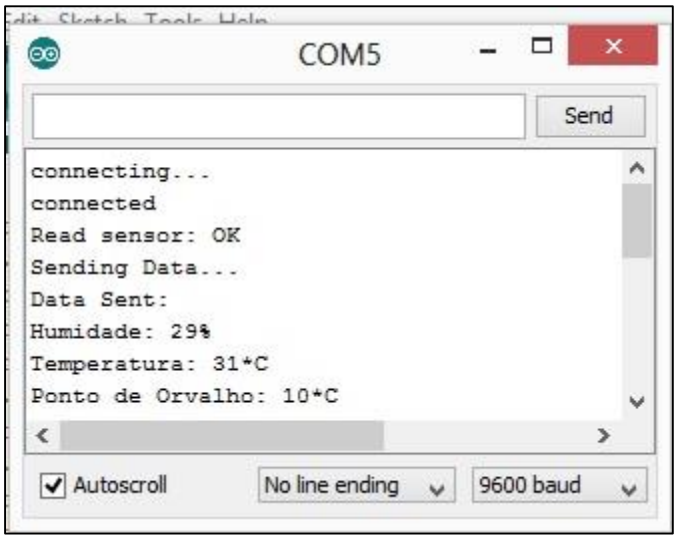

Figura 18 - Teste no serial mostrando os dados capturado pelo sensor

Fonte: o Autor

A segunda fase dos testes (ver Figura 18), foi utilizando o sensor DHT11 (usado para medir temperatura, umidade e ponto de orvalho) apresentando as informações no seu serial, componente da IDE do Arduino, com o intuito de mostrar os dados recebido pelo sensor como: temperatura, umidade e ponto de orvalho. O sensor tem quer ser conectado nos pinos analógicos de 1 a 9 para esta aplicação, pois conectados a outros pinos que não seja esses impossibilita o sensor de ler os dados e manda informações.

Temperatura é uma medida estatística do nível de agitação entre moléculas, relacionado com o deslocamento da energia cinética de um átomo ou molécula. Em Física, a temperatura está relacionada com a energia interna de um sistema termodinâmico.

A temperatura costuma ser medida por um termômetro e indica o grau de intensidade do calor em um determinado território. A temperatura atmosférica da Terra é resultado das ondas eletromagnéticas que vêm do Sol. A variação de temperatura depende de vários fatores, como o vento, a umidade do ar, a latitude, o ângulo de incidência do raio solar na superfície terrestre

A quantidade de vapor de água existente na atmosfera é denominada umidade do ar. Os valores podem ser expressos em números absolutos (g/m³) ou em

forma relativa (%) ao seu ponto de saturação. Esse é um dos elementos analisados para a caracterização climática de um determinado local.

Entre os métodos utilizados para medir a umidade do ar estão o psicrômetro (calcula a velocidade de evaporação da água) e o higrômetro (mede a quantidade de água presente nos gases). Esses dados podem ser obtidos através de porcentagens, por exemplo: a umidade relativa do ar é de 75%. Nesse caso, significa que restam 25% para o ar reter todo o vapor de água e transformá-lo no estado líquido.

Ponto de orvalho é a temperatura à qual o vapor de água presente no ar ambiente passa ao estado líquido na forma de pequenas gotas (chuva). O ponto de orvalho no exterior dará a indicação da possibilidade de ocorrência de orvalho ou geada se a temperatura descer durante a noite abaixo do ponto de orvalho. Esta possibilidade aumenta ainda no caso de céu limpo, devido ao efeito de diminuição de temperatura motivado pela maior radiação em direção ao céu. Quando o valor do ponto de orvalho se aproxima da temperatura externa, existe uma forte indicação de que haverá chuva.

Dada a Temperatura T (°C) e a Umidade Relativa UR (%), é a seguinte a fórmula de cálculo do ponto de orvalho To (°C): $To = T - (14,55 + 0,114 \times T) \times [1 - (0,01 \times UR)] - \{(2,5 + 0,007 \times T) \times [1 - (0,01 \times UR)]\}^3 - (15,9 + 0,117 \times T) \times [1 - (0,01 \times UR)]^{14}$

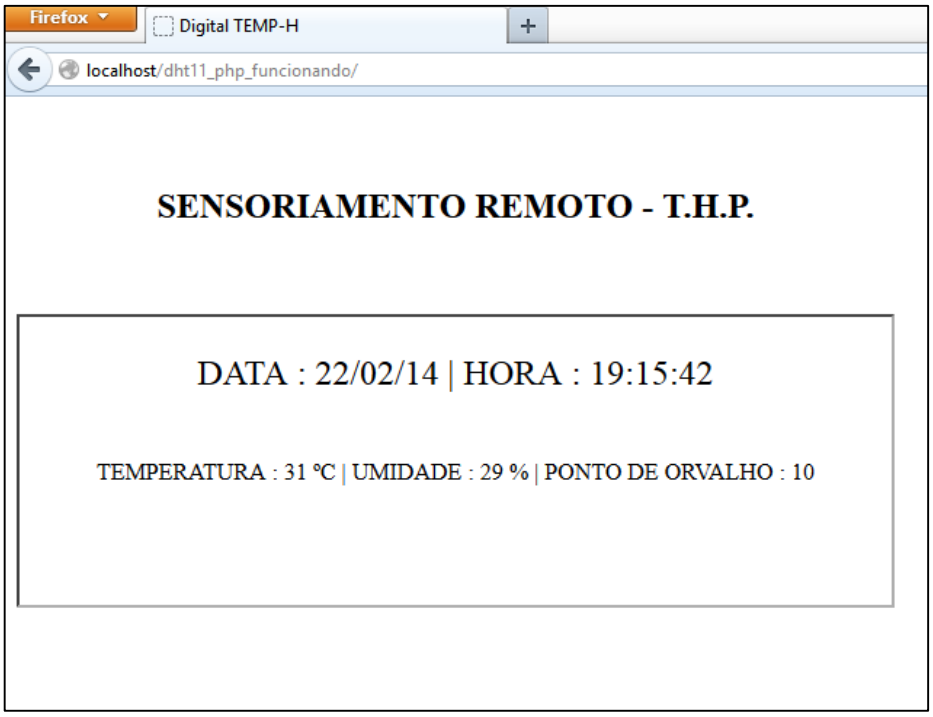

Figura 19 - Teste mostrando os dados capturado pelo sensor em uma aplicação PHP

Na terceira fase de testes (ver Figura 19), o sensor DHT11 comunica-se com a aplicação *web* em PHP e mostrando a data, hora, temperatura, umidade e ponto de orvalho. Para que os dados fossem exibidos foi necessário a criação de um banco de dados *bancoth* no *Phpmyadmin* com duas tabelas “dados” e “atualiza”, o arquivo *conectar.php* vai estabelecer a conexão do banco de dados com a aplicação *web*, um outro arquivo chamado *conteudo.php* tem como função mostrar os dados na página *web*, (ver Figura 19). E um arquivo *gravar.php* que vai inserir os dados na tabela “dados” e atualizar na tabela “atualiza”, e ainda cria um arquivo data.txt para salvar as informações capturadas pelo sensor. Não sendo possível comunicar com nenhum desses arquivos citados acima, não será possível visualizar qualquer informação obtida pelo sensor.

Na quarta fase dos testes com todas as partes funcionando corretamente foi montado todo o protótipo, a base já pronta com rodas e chassi e acoplado a um roteador para receber e transmitir sinal simulando a conexão com a *internet*.

Onde computadores conectados a esse roteador possibilitando acessar a aplicação *web*, desde de que seja usada a configuração válida do endereço IP do computador usado como servidor, pois a aplicação não foi hospedada em servidores online.

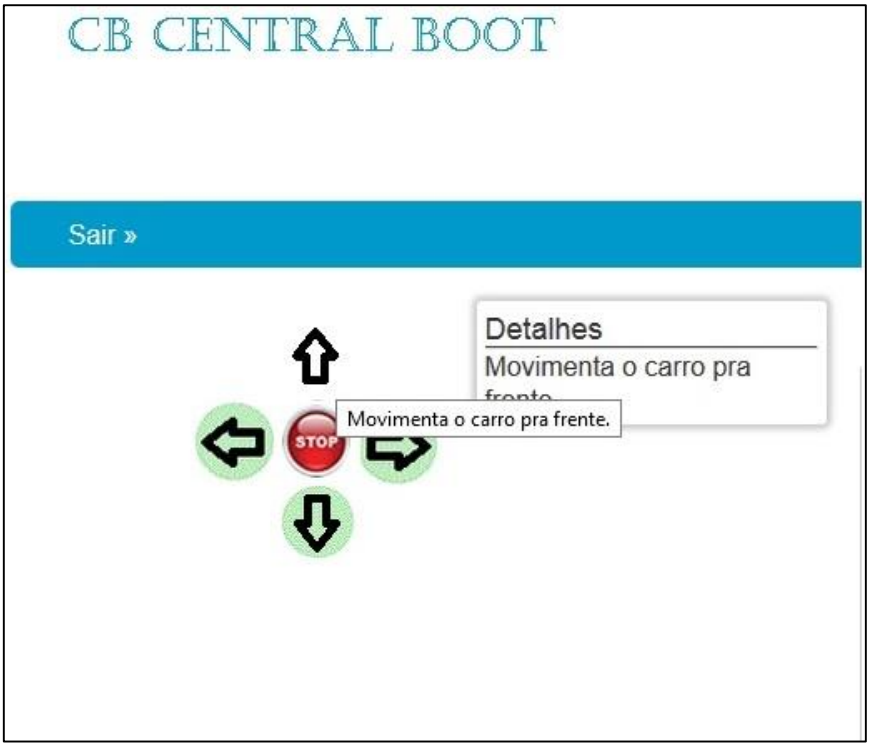

Figura 20 - Controles do robô móvel

Fonte: o Autor

São mostrados os controles do robô (ver Figura 20) através de setas indicando em que sentido o robô móvel deve se deslocar, colocando o cursor em uma das setas será visualizada uma mensagem indicar a ação do botão da aplicação, correspondendo a cada click, mas quando a rede está sobrecarrega, muito lenta o tempo de envio dos comandos pode ultrapassar 5 segundos para que o robô possa receber tal informação e acionar o motor correspondente.

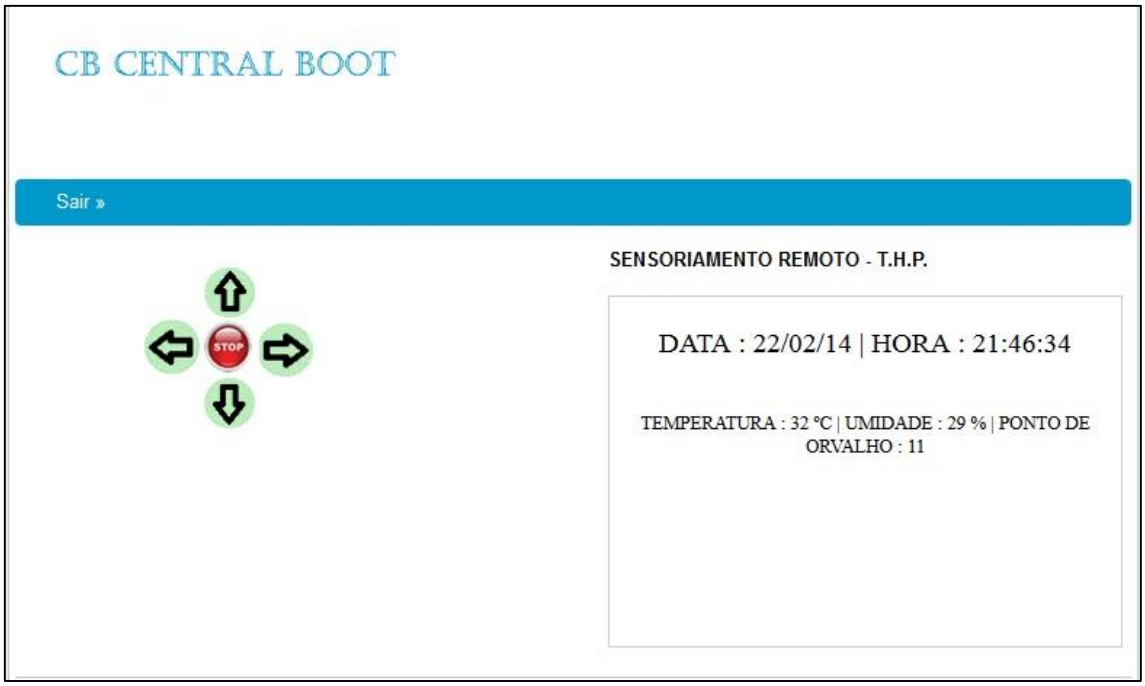

Figura 21 - Aplicação completa

Fonte: o Autor

Mostra a aplicação em funcionamento (ver Figura 21,) ocorrendo alguns conflitos na rede impossibilitando o sensor de mandar os dados na página deixando esses dados estáticos perdendo comunicação com o sensor. Os botões um atraso ao enviar a requisição para o robô móvel. Um outro problema é quando as baterias estão com pouca carga ou a carga menor que 5volts os *Shield Ethenert* não serão alimentados com a carga correta impossibilitando a comunicação com a aplicação *web.*

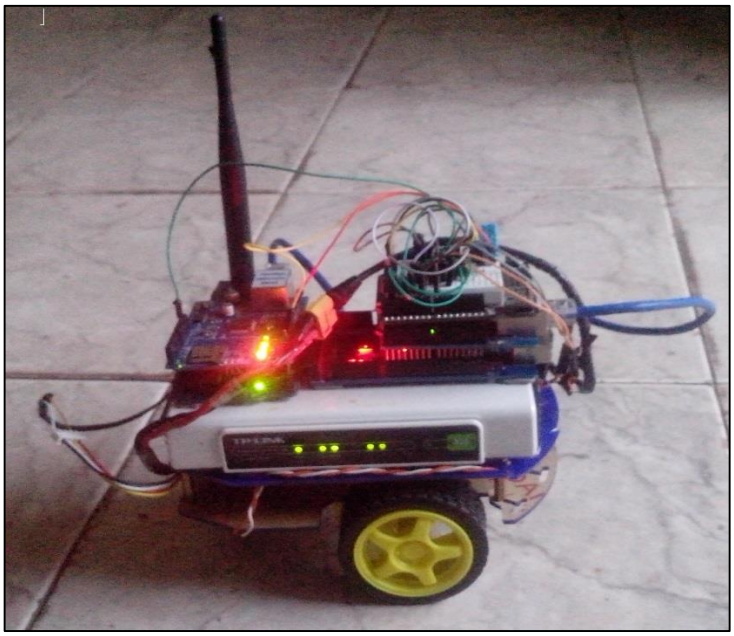

Figura 22 - Alcance do roteador com o robô móvel

Fonte: o Autor

O alcance do controles pode variar de acordo com o ambiente (ver Figura 22), pois como os testes tratam de comunicação sem fio muitos fatores vão influenciar, tais como: paredes, arvores conflitos de sinais. Foi observado também que o terreno em que o robô móvel vai se deslocar dever ser superfície plana, pois sua roda traseira, responsável pela direção a ser tomada pelo robô móvel, não consegue se mover em locais não planos.

A Tabela 15 representa os teste obtidos através do sensor DHT11 capturando a temperatura, umidade e ponto de orvalho dentro de uma casa das 10:36:04 da manhã até as 17:24:04 da tarde

Tabela 15 – Testes com informações capturadas dentro de casa

Data/Hora	Temperatura	Umidade		Ponto de Orvalho
22/02/14 | 10:36:04	28	33		9
22/02/14 | 10:39:47	29	30		9
22/02/14 | 11:56:16	30	30		10
22/02/14 | 12:12:26	31	31		11
22/02/14 | 12:46:20	33	28		12
22/02/14 | 13:42:37	32	29		11
22/02/14 | 14:22:30	32	31		12
22/02/14 | 15:12:39	30	30		10
22/02/14 | 16:25:23	29	31		10
22/02/14 | 17:24:04	28	30		8

A Tabela 16 mostra as informações capturadas pelo sensorDHT11 no pátio da Universidade Federal do Piauí campus Picos das 09:36:04 da manhã até 12:15:12, meio dia.

Tabela 16 – Teste com informações capturada no pátio da UFPI

Data/Hora	Temperatura	Umidade	Ponto de Orvalho
25/02/14 | 09:36:04	28	31	9
25/02/14 | 10:52:44	29	31	10
25/02/14 | 11:20:20	29	33	10
25/02/14 | 11:20:23	29	32	10
25/02/14 | 12:14:52	30	31	11
25/02/14 | 12:15:12	30	34	12

A Tabela 17 exibe informações referente a captura de dados pelo sensor DHT11 dentro do LIPPO com o Ar condicionado ligado às 09:31 da manhã.

Tabela 17 - Teste informações capturada dentro do Lippo com Ar ligado

Data/Hora	Temperatura	Umidade	Ponto de Orvalho
25/02/14 | 09:31:24	25	33	7
25/02/14 | 09:31:26	25	33	7
25/02/14 | 09:31:28	25	33	7
25/02/14 | 09:31:31	25	33	7
25/02/14 | 09:31:33	25	33	7
25/02/14 | 09:31:24	25	33	7

A Tabela 18 mostra as informações obtidas pelo sensor DHT11 em meio ao sol próximo ao pátio da UFPI as 09:33 até 12:15.

Tabela 18 - Teste informações capturada em meio ao sol na UFPI

Data/Hora	Temperatura	Umidade	Ponto de Orvalho
25/02/14 | 09:33:01	29	37	12
25/02/14 | 09:33:03	30	37	13
25/02/14 | 09:33:05	30	37	13
25/02/14 | 09:33:12	31	37	14
25/02/14 | 09:33:14	31	37	14
25/02/14 | 12:15:37	33	29	12
25/02/14 | 12:15:39	34	30	13
25/02/14 | 12:15:43	35	29	14
25/02/14 | 12:15: 45	36	29	15
25/02/14 | 12:15:47	38	28	16

6 CONCLUSÂO

Este projeto teve como propósito desenvolver um protótipo de robô móvel com baixo custo de construção e estrutura capaz de realizar seu controle remoto via *web* e receber dados nessa mesma aplicação *web*.

É importante enfatizar que o sistema de controle, juntamente com o robô móvel, faz parte de um protótipo funcional. No entanto, alguns itens podem ser melhorados, de forma que no futuro, este possa ser adaptado a um projeto comercial de baixo custo e grande eficiência.

Os resultados apresentados nos experimentos foram satisfatórios devido a roda cega possibilita o robô de andar apenas em terrenos planos então o solo pode influenciar em relação aos movimentos do robô, assim como temperaturas elevadíssimas ou muito baixas, pois o sensor não é tão preciso. O robô apresentou os resultados esperados obedecendo a cada comando enviado e recebidos pela aplicação *web*.

Como trabalhos e melhorias futuras, podem ser citados alguns itens como: acoplamento de uma câmera, desenvolvimento do sistema de adaptação a solos irregulares, adaptação de sensores de presença ao robô móvel e sistemas inteligentes capaz de ser autonomos ao servidor remoto, desenvolvimento de módulo robusto.

REFERÊNCIAS

ARDUINO E COMPANHIA.2013. Disponivel em: ethernet- shield-wiznet-w5100-parte-1.html> Acesso em 09 de Jan. 2014.

ARDUINO. Arduino. 2013. Disponível em: <http://www.arduino.cc>. Acesso em: 15 de Dez. 2013.

BANZI, Massimo. Primeiros passos com o Arduíno. Novatec Editora, 2012.

DROBOTICS ONLINE 2013.Disponivel em <http://www.droboticsonline.com/index.php/arduino/sensor.html>. Acesso em 08 de Dez. 2013.

ELETRONICA DIDÁTICA .2013. Disponível em <http://www.eletronicadidaticas.com.br>. Acesso em 14 de Jan.2014.

FERNANDES, A. M. R. Inteligência Artificial: noções gerais. 2. ed. Florianópolis: VisualBooks Editora, 2005.

HEINEN, F. Controle Híbrido Inteligente de Robôs Autônomos: Dissertação de mestrado, Mestrado em Computação Aplicada / Unisinos, 2002.

INSTITUTO FEDERAL (UFRS). Disponível em <http://www.if.ufrgs.br>. Acesso em: 16 de Jan.2014.

MANUAL DO PHP .2014. Disponível em: <www.php.net/manual/pt_BR>. Acesso em: 12 de Jan.2014.

NILSSON, N. A mobile automaton: An application of artificial intelligence techniques. In Proceedings of the 1st. International Joint Conference on, 1969. Artificial Intelligence (1969), pp. 509–520.

OLIVEIRA, Emerson de. Protótipo de um Robô Rastreador de Objetos. Monografia Apresentada à Universidade Regional de Blumenau no curso de ciências da computação. Blumenau-SC, 2001.

RUSSELL, S; NORVIG, P. Inteligência Artificial. 5. ed. Rio de Janeiro: Elsevier, 2004.

SCRIMGER, ROB. TCP/IP a Bíblia. Tradução de Edson Furmankievicz, DocWare. 7ª Reimpressão. Rio de Janeiro: Elsevier, 2002.

TANENBAUM, Andrew S.; STEEN, Maarten Van. Sistemas Distribuido. 2. ed. São Paulo: ARLETE SIMILLE MARQUES, 2007.

TANENBAUM, ANDREW S.; WETHERALL, David. Redes de Computadores. 5. ed. São Paulo:PRENTICE HALL, 2011.

WALTER, W Grey, Máquina Speculatrix .1950

LIMA, Thiago José Barbosa. **PARADINHA: Um robô móvel inteligente**.2013.monografia

APÊNDICE A – Sketch responsável por enviar os dados para o arquivo PHP.

```
 #include <SPI.h>
 #include <Ethernet.h>
#define esq 45
#define dir 135
#define frt 90
#define dis 30

int pinMotorEsq = 5;
int pinELE = 4;
int pinELD = 3;

int pinMotorDir = 9;
int pinDLE = 8;
int pinDLD = 6;

int velE = 220;
int velD = 220;

int add = 15;

 //Configurações do Ethernet Shield
 byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
 //byte ip[] = { 10,180,14,99 }; // ip que o arduino assumirá
 byte ip[] = { 10,180,16,120};
 //byte gateway[] = { 192,168,2, 1 }; // ip do roteador
 //byte subnet[] = { 255, 255, 0, 0 };

 // String que representa o estado dos dispositivos
 char Luz[7] = "0000L#";

 EthernetServer server(8080); // Cria o servidor na porta 8081
```

```
// String onde é guardada as msgs recebidas
char msg[7] = "0000L#";

void setup() {
  Ethernet.begin(mac, ip);
  server.begin();
    pinMode(pinMotorEsq, OUTPUT);
 pinMode(pinMotorDir, OUTPUT);
 pinMode(pinELE, OUTPUT);
 pinMode(pinELD, OUTPUT);
 pinMode(pinDLE, OUTPUT);
 pinMode(pinDLD, OUTPUT);
  pinMode(7,OUTPUT);
}
void loop() {
EthernetClient client = server.available();
// SE receber um caracter...
if (client) {
// guarda o caracter na string 'msg'
msg[1]=msg[2];
msg[2]=msg[3];
msg[3]=msg[4];
msg[4]=msg[5];
msg[5]=msg[6];
msg[6] = client.read();

if (msg[6]=='#') {
switch(msg[5]) {

case 'R': {
  client.write(Luz);
 break;
}
```

```
case 'F': {
  frente();
  break;
}

case 'T': {
  re();
  break;
}

case 'D': {
  direita();
  break;
}

 case 'E': {
  esquerda();
  break;
}

 case 'P': {
  parar();
 }
  break;
}
}
}
}

void frente(){
 analogWrite(pinMotorEsq, velE);
 digitalWrite(pinELE, 1);
 digitalWrite(pinELD, 0);
```

```
  analogWrite(pinMotorDir, velD);
  digitalWrite(pinDLE, 1);
  digitalWrite(pinDLD, 0);
}

void re(){
  analogWrite(pinMotorEsq, velE);
  digitalWrite(pinELE, 0);
  digitalWrite(pinELD, 1);
  analogWrite(pinMotorDir, velD);
  digitalWrite(pinDLE, 0);
  digitalWrite(pinDLD, 1);
}

void parar(){
  analogWrite(pinMotorEsq, LOW);
  digitalWrite(pinELE, LOW);
  digitalWrite(pinELD, LOW);
  analogWrite(pinMotorDir, LOW);
  digitalWrite(pinDLE, LOW);
  digitalWrite(pinDLD, LOW);
}

void esquerda(){
  analogWrite(pinMotorEsq, velE + add);
  digitalWrite(pinELE, 0);
  digitalWrite(pinELD, 1);
  analogWrite(pinMotorDir, velD + add);
  digitalWrite(pinDLE, 1);
  digitalWrite(pinDLD, 0);
}

void direita(){
  analogWrite(pinMotorEsq, velE + add);
```

```
  digitalWrite(pinELE, 1);
  digitalWrite(pinELD, 0);
  analogWrite(pinMotorDir, velD + add);
  digitalWrite(pinDLE, 0);
  digitalWrite(pinDLD, 1);
}
```

APÊNDICE B – Código responsável por receber os dados do arduino e pelos botões de controle.

```
<html>
 <head></head>
<body>

 <?php
 $sock = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
 // Se conecta ao IP e Porta:
socket_connect($sock,'10.180.15.108', 8081);
 // Executa a ação correspondente ao botão apertado.
 if(isset($_POST['bits'])) {
        $msg = $_POST['bits'];
        if(isset($_POST['Frente'])){ $msg = 'F#'; }
        if(isset($_POST['Re' ])){ $msg = 'T#'; }
        if(isset($_POST['Direita' ])){ $msg = 'D#'; }
        if(isset($_POST['Esquerda' ])){ $msg = 'E#'; }
        if(isset($_POST['Parar' ])){ $msg = 'P#'; }

        socket_write($sock,$msg,strlen($msg));
 }

 socket_write($sock,'R#',2); //Requisita o status do sistema.

 // Espera e lê o status e define a cor dos botões de acordo.

 $status = socket_read($sock,6);


 echo "<form method =\"post\" action=\"proj.php\">";
 echo "<input type=\"hidden\" name=\"bits\" value=\"$status\">";
 echo "<button style=\"width:90;font: bold 14px Arial\" type = \"Submit\"Name =
\"Frente\">FRENTE</button></br></br>";
```

```
 echo "<button style=\"width:90;font: bold 14px Arial\" type = \"Submit\"Name = \"Re\">RE</button></br></br>";
 echo "<button style=\"width:90;font: bold 14px Arial\" type = \"Submit\"Name = \"Direita\">DIREITA</button></br></br>";
 echo "<button style=\"width:90;font: bold 14px Arial\" type = \"Submit\"Name = \"Esquerda\">ESQUERDA</button></br></br>";
 echo "<button style=\"width:90;font: bold 14px Arial\" type = \"Submit\"Name = \"Parar\">PARAR</button></br></br>";
 echo "</form>";
 socket_close($sock);
 ?>
 </body>
 </html>
```

APÊNDICE C – Sketch responsável por enviar os dados do sensor na página PHP

```
#include <SPI.h>
#include <Ethernet.h>
#include <dht11.h>
dht11 DHT11;
#define DHT11PIN 2
byte mac[] = {  0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress server(10,180,14,138);
EthernetClient client;
//char buffer[256];

void setup()
{
  Serial.begin(9600);
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
  }
  delay(1000);
}

void loop(){
  Serial.println("connecting...");
  if (client.connect(server, 8080)){

  }

  int chk = DHT11.read(DHT11PIN);

  int hum=DHT11.humidity;
  int temp=DHT11.temperature;
  int dew=DHT11.dewPoint();
  if (client.connected()){
```

```
    client.print(hum,DEC);
    client.print("&temp=");
    client.print(temp,DEC);
    client.print("&dew=");
    client.print(dew,DEC);
    client.println(" HTTP/1.1");
    client.println("Host: arduino1");
    client.println();
    client.print("GET /dht11_php_funcionando/gravar.php?hum=");

  }
  while(client.available()) {
    char c = client.read();
  }
  delay(1000);
  client.stop();
  delay(100);
}
```

APÊNDICE D – Código em HTML responsável por mostrar os dados na página PHP

```
<html>
<head>
<title>Digital TEMP-H</title>
</head>
<body>
<br><br> <center> <h2>  SENSORIAMENTO REMOTO - T.H.P.</h2> <center>
<br><br>
<center>
<iframe Marginheight='16' align='bottom' width='600' height='200' frameborder='2'
src='http://10.180.14.138/dht11_php_funcionando/conteudo.php'></iframe>
</center>
</body>
</html>
```

APÊNDICE E – Código em PHP responsável pela conexão com banco de dados

```
<?php
$local_serve = "localhost";      // local do servidor
$usuario_serve = "root";         // nome do usuario
$senha_serve = "";               // senha
$banco_de_dados = "bancoth";     // nome do banco de dados

$conn = @mysql_connect($local_serve,$usuario_serve,$senha_serve) or die ("O servidor não responde!");
// conecta-se ao banco de dados
$db = @mysql_select_db($banco_de_dados,$conn)
  or die ("Não foi possivel conectar-se ao banco de dados!");
  ?>
```

APÊNDICE F – Banco de Dados

```
create database bancoth;
create table dados(
Temperatura int not null,
Humidade int not null,
Ponto_Orvalho int not null,
Data varchar(8),
Hora varchar(8))
create table atualiza(
Temperatura int not null,
Humidade int not null,
Ponto_Orvalho int not null)
insert into atualiza values(2,3,4)
```