General Purpose Bluetooth Control

General Purpose Bluetooth Control

F. S. Andrade, G. G. Bomfim, M. Torres, F. M. Milian and T. Pires

***Abstract*— This paper presents the development of a Bluetooth remote control in Java Micro Edition to command devices based on microcontrollers. The feasibility of the system was demonstrated by the develop of a Bluetooth remote control able to manipulate the movements of a toy car. This prototype uses a mobile device, a microcontroller, a Bluetooth module and two DC motors.**

***Keywords*— Bluetooth, Remote Control, microcontroller.**

I. INTRODUÇÃO

A BUSCA pela comunicação ágil e rápida alastrou-se e o mercado tem apresentado uma enorme quantidade e variedade de dispositivos móveis. Estes aparelhos cada vez mais robustos e próximos de pequenos computadores, possuem memória, processador, armazenamento de dados e diversos recursos agregados como câmeras digitais, dispositivos de comunicação, mensagens multimídia, áudio e vídeo, agenda eletrônica e até mesmo funções como GPS (Global Position System), além de poder acessar dados na Internet e se conectar a um computador ou a outro dispositivo para sincronizar informações. Diante desta situação, surge o Bluetooth, uma tecnologia com baixo custo, ideal para sistemas móveis e transmissões de curta distância. Podem ser utilizados em computadores, celulares e também em aparelhos eletroeletrônicos. Em um futuro próximo poderemos controlar os equipamentos de uma residência através de um computador utilizando um microcontrolador Bluetooth. No caso do Bluetooth IEEE 802.15.1, os dispositivos podem ser utilizados em diversas aplicações, como segurança e monitoramento na área de saúde, sensores de segurança e incêndio, detecção de falhas em sistemas e ambientes pervasivos, como também em ambientes cooperados de robôs. Seguindo esta tendência decidimos projetar um controle bluetooth para dispositivos de propósito geral, utilizando para demonstração deste funcionamento, um carro de controle remoto.

Na próxima seção apresentam-se os trabalhos relacionados ao nosso projeto. Na seção III veremos os materiais e métodos utilizados, na seção IV os resultados e discussão obtidos, finalizando a seção V as conclusões encontradas.

F. S. Andrade, Santa Cruz Tecnologia. 63.175.277/0001-75, Ilhéus - BA, fabriciosaand@gmail.com

G. G. Bomfim Junior, COOPEC LTDA. 32.615.247/0001-09, Ilhéus – BA, geraldo_bomfim@hotmail.com

M. Torres, Universidade Estadual de Santa Cruz (UESC), Ilhéus-Ba, mxtd2000@yahoo.com.br

F. M. Milian, Universidade Estadual de Santa Cruz (UESC), Ilhéus-Ba, felix_mas_milian@yahoo.com

T. Pires Junior, PMCC UFBA/UNIFACS/UEFS. 15.180.714/0001-04, Salvador – BA, teodoropires@gmail.com

II. TRABALHOS RELACIONADOS

A tecnologia utilizada no projeto é a bluetooth que possui sua descrição formal de protocolos em [1] e especificação de implementações Java em [2]. O projeto desenvolvido em [3], aborda aplicações interessantes para a tecnologia, como controle remoto de robôs e máquinas utilizando Bluetooth, Infrared (IR) e smartphones, serviu de base para o estudo de como estabelecer a conexão entre o aparelho celular e o modulo Bluetooth utilizados neste projeto. A pesquisa realizada em [4] também relata experimentos utilizando PDA's com bluetooth como controladores de pequenos robôs, demostrando a tendência do mercado em desenvolver aplicações voltadas a soluções sem fio, principalmente via Bluetooth. As pesquisas realizadas serviram para a aquisição do conhecimento necessário sobre a maneira de desenvolver controle remoto usando celular para controlar sistemas microcontrolados através de Bluetooth.

III. MATERIAIS E MÉTODOS

A. Componentes e dispositivos necessários

Dispositivo móvel - O projeto necessita de um dispositivo móvel compatível com a tecnologia Java ME com configuração CLDC e perfil MIDP, que suporte conexões via Bluetooth através da Java APIs for Bluetooth JSR-82. É importante salientar que para a aplicação realizar a comunicação com a placa, o dispositivo móvel, além de possuir a tecnologia Bluetooth deve ter implementada a API-82 em sua máquina virtual. Neste projeto foi utilizado o celular modelo SGH-E215L da Samsung especificado em [5], que possui todos os pré-requisitos descritos anteriormente. Fig. 1.

Figura 1. SGH-E215L.

Placa com módulo Bluetooth – Para a comunicação do microcontrolador com o dispositivo móvel é necessário a utilização de um módulo bluetooth. Foi utilizada uma placa Cerne Bluetooth PICLAB16F876A. Essa placa possui 1

conversor AD, 1 display LCD, 4 LEDs, 1 Trimpot para o AD do PIC, 1 Conector DB9, 1 Receptor Infrared (IR), 4 Botões, microcontrolador PIC16F876A e um módulo Bluetooth KC-21 pelo qual é feita a comunicação com o dispositivo móvel. Para este projeto foram necessários apenas o modulo KC-21 detalhado em [6] o microcontrolador PIC16F876A e o display LCD (opcional), não sendo utilizados os demais componentes presentes na placa Cerne Bluetooth PICLAB16F876A [7]. Fig. 2.

Figura 2. Placa Cerne Bluetooth.

Ponte H - Para podermos controlar os motores DC através do microcontrolador, foi necessária a utilização de uma construção elétrica conhecida como “Ponte H” [8]. Ponte H é um circuito eletrônico que permite que um microcontrolador controle um motor DC.

O nome ponte H é dado pela forma que assume o circuito quando montado. O circuito é construído com quatro "chaves" (S1-S4) que são acionadas de forma alternada (S1 e S4 ou S2 e S3). Para cada configuração das chaves o motor gira em um sentido. As chaves S1 e S2 assim como as chaves S3 e S4 não podem ser ligadas ao mesmo tempo porque podem gerar um curto circuito. Fig. 3.

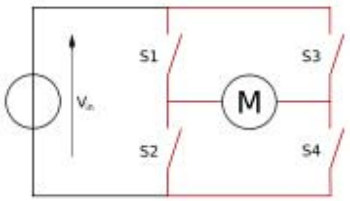

Figura 3. Esquema elétrico Ponte H.

Em nosso projeto foi utilizado um circuito pronto. O CI L293D descrito em [9]. Fig. 4. O L293D é uma ponte H dupla, isto é, controla até dois motores e faz com que estes rodem nas duas direções.

Figura 4. CI L293D.

A Fig. 5 mostra a configuração e o esquema elétrico do L293D. No esquema abaixo M corresponde aos motores A e B (direção e tração); PINxx correspondem aos pinos no microcontrolador; V2+, GND e 5v correspondem também aos respectivos pinos no microcontrolador. O pino 8(V2+) do CI é o responsável pela alimentação dos motores, sendo ligado ao Vcc do conjunto de baterias destinadas a esta finalidade. Os pinos 3, 6 e 11,14 do CI foram ligados aos motores e serão os responsáveis pelo sentido do giro dos motores. Os pino 2,7,10 e 15 do CI podem ser utilizados como PWM, em nosso projeto utilizamos o PWM apenas no motor responsável pela tração do conjunto. Fig. 5.

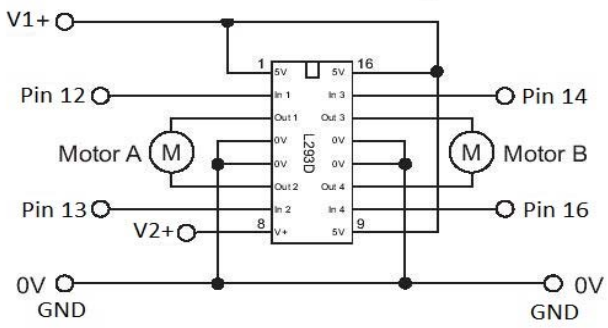

Figura 5. Esquema elétrico CI L293D.

Para uma melhor visualização dos pinos acionados pelo microcontrolador, foram utilizados os mesmos pinos que acendem os Leds da placa Cerne. Que são os pinos 12,13,14 e16 da porta C RC1, RC2, RC3 e RC5 respectivamente. Como visto no esquema elétrico da placa. Fig. 6.

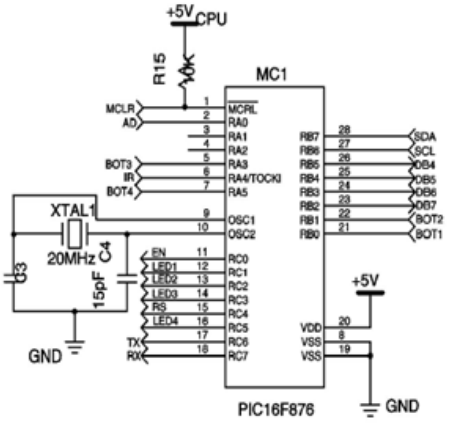

Figura 6. Esquema elétrico Cerne Bluetooth.

Para conectar os motores à placa CERNE, foi construída uma placa de circuito impresso. Feita em fenolite cobreado, que abriga um soquete para o L293D e terminais para acesso aos pinos do CI. Fig. 7.

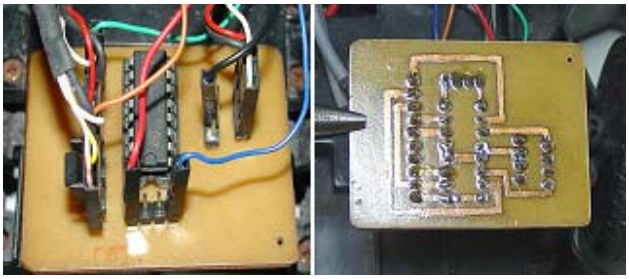

Figura 7. Placa Fenolite Cobreada.

Motores DC – Foram utilizados dois motores DC, do mesmo mostrado na Fig. 8, para realizar os movimentos do carro.

Figura 8. Motor DC.

Um motor é responsável pela tração, e outro pela direção. Para o controle da aceleração utilizamos o PWM descrito em [10], PWM é uma tecnologia que permite controlar o período cíclico da frequência da alimentação. Com isso, se tem o total domínio da potência designada e é possível controlar a velocidade de motores de baixa potência, sem perder o torque, podendo inclusive parar o motor na metade do percurso, mantendo estável a carga. O PWM está disponível no microcontrolador Pic16F876A no pino 13 RC2, detalhado em [11].

B. Desenvolvimento dos softwares

O software utilizado no microcontrolador foi desenvolvido em linguagem C para microcontrolador, e possui a rotina de funcionamento descrita abaixo.

1- Declaração, definição, inicialização das variáveis e constantes, configuração dos pinos de entrada e saída, prototipagem de funções e rotinas para inicialização do LCD.

2-Rotina principal (void main) –

2.1-Habilitação de interrupções,

2.2-Habilitação da função PWM no pino RC2 setup_ccp1(CCP_PWM); Configura CCP1 em modo PWM;

2.3-Setup_timer_2 (T2_DIV_BY_4, 255, 1); determina o período do sinal PWM set_pwm1_duty(0); zero igual a desligado,

2.4-Declaração de variável auxiliar e de controle, inicialização do LCD,

2.5-Envio dos comandos para habilitação e configuração do modulo bluetooth através da comunicação serial USART.:

2.5.1-AT(AT+ZV DefaultLocalName Modulo 1 (define o nome do modulo);

2.5.2-AT+ZV EnableBond (habilita o modulo para conexão)

3-Laço principal (while infinito) - Executa a função de rotina de recepção serial e exibe o caractere recebido no visor LCD e atribui o valor “0”(zero) à variável de controle(zero indica que nenhum caractere foi recebido). A função possui cinco tarefas que controlam a movimentação do carro, sendo estas:

Primeira: se o caractere recebido for igual a 2, é enviado um sinal de controle com o valor 1, ao pino 14 do microcontrolador, o RC3, fazendo com que o motor de tração seja acionado no sentido horário e que o carro se desloque para frente, ao mesmo tempo em que é ativado o modo PWM tornando este deslocamento gradativo, uma vez que o PWM ativará o motor na potência mínima e ira incrementando esta a cada 50 milisegundos até que este alcance a potência máxima.

Segunda: se o caractere recebido for igual a 4, é enviado um sinal de controle com o valor 1, ao pino 13 do microcontrolador, o RC2, fazendo com que o motor de direção seja acionado em sentido anti-horário e o carro se vire para a esquerda.

Terceira: se o caractere recebido for igual a 5, é enviado um sinal de controle com o valor 0, a todos os pinos, fazendo com que o carro pare.

Quarta: se o caractere recebido for igual a 6, é enviado um sinal de controle com o valor 1, ao pino 12 do microcontrolador, o RC1, fazendo com que o motor de direção seja acionado em sentido horário e o carro vire para a direita.

Quinta: se o caractere recebido for igual a 8, é enviado um sinal de controle com o valor 1, ao pino 16 do microcontrolador, o RC5, fazendo com que o motor de tração seja acionado em sentido anti-horário e o carro se desloque para trás.

A rotina de recepção serial escuta a porta serial e se houver interrupção, pega o caractere recebido e armazena em uma variável, compara a variável e executa a tarefa correspondente, atribui a variável de controle o valor 1 (1 indica que um caractere foi recebido).

4-Tarefas – se o caractere recebido pela porta serial do microcontrolador for igual a “2”, envia sinal de controle ao pino RC2, o qual controla o movimento de tração para frente.

```
if (dado == '2'){//caractere recebido pela porta serial
output_low(pin_c1);//pino 1 = zero
output_low(pin_c3); //pino 3 = zero
```

```
output_low(pin_c5); //pino 5 = zero
while (value < 255){//valor máximo PWM 255
setup_ccp1(CCP_PWM);
value++; //incrementa variavél
set_pwm1_duty(value);//PWM ativado
delay_ms(50);//atraso de 50 milisegundos}
```

Se o caractere recebido pela porta serial do microcontrolador for igual a “4”, envia sinal de controle ao pino RC3, o qual controla o movimento de direção à esquerda.

```
if (dado == '4'){//caractere recebido pela porta serial
output_low(pin_c1); );//pino 1 = zero
output_high(pin_c3); );//pino 3 = 1 }
```

Se o caractere recebido pela porta serial do microcontrolador for igual a “6”, envia sinal de controle ao pino RC1, o qual controla o movimento de direção à direita.

```
if (dado == '6'){//caractere recebido pela porta serial
output_high(pin_c1); );//pino 1 = 1
output_low(pin_c3); );//pino 3 = zero }
```

Se o caractere recebido pela porta serial do microcontrolador for igual a “8”, envia sinal de controle ao pino RC5, o qual controla o movimento de tração para trás.

```
if (dado == '8'){//caractere recebido pela porta serial
output_low(pin_c1);
setup_ccp1(CCP_OFF);//PWM desligado
output_low(pin_c2); //pino 2 = 0
output_low(pin_c3); //pino 3 = 0
output_high(pin_c5); //pino 5 = 0
```

Qualquer caractere diferente dos citados acima desliga os motores.

```
output_low(pin_c1); //pino 1 = 0
output_low(pin_c3); //pino 3 = 0
output_low(pin_c2); //pino 2 = 0
setup_ccp1(CCP_OFF);//desativa o PWM
output_low(pin_c5); //pino 5 = 0
```

Para a comunicação entre o dispositivo móvel e a placa Cerne utilizamos a plataforma JME com a API Bluetooth JSR-82. Para estabelecer a conexão Bluetooth foi utilizado o perfil SPP(Serial Port Profile). Esse perfil define os requisitos necessários para a criação de uma conexão serial emulada utilizando o protocolo RFCOMM entre os dois dispositivos. RFCOMM é um protocolo de transporte simples, com provisões adicionais para emular os nove circuitos das portas seriais RS-232. O protocolo RFCOMM suporta até 60 conexões simultâneas (canais RFCOMM) entre dois dispositivos Bluetooth.

Uma aplicação Bluetooth consiste de duas partes: Aplicação cliente e aplicação servidora. A aplicação cliente roda no dispositivo móvel usando JME. A aplicação servidora roda na placa CERNE usando a linguagem C.

São necessárias quatro etapas para estabelecer uma conexão entre o cliente e o servidor: Inicialização da pilha de protocolos, localizar dispositivos, localizar serviços e estabelecer conexão. A API JSR-82 fornece as classes LocalDevice e RemoteDevice, essenciais para a conexão Bluetooth. A classe LocalDevice representa o dispositivo Bluetooth local. A partir dessa classe é possível gerenciar e obter as propriedades do dispositivo. Por ser uma classe de construtor privado só haverá um objeto LocalDevice na memória, obtido através do método estático getLocalDevice(). Fig. 9.

Figura 9. Tela software. Procurando.

A classe RemoteDevice representa um dispositivo Bluetooth remoto, onde é possível obter várias informações a respeito do dispositivo, como endereço e friendly name. Para iniciar a busca por dispositivos utilizamos o método startInquiry passando dois argumentos. O primeiro argumento é um inteiro que especifica o modo de conectividade definido para a busca. Utilizamos o método DiscoveryAgent.GIAC para ter acesso ilimitado. O segundo parâmetro é um objeto que implementa DiscoveryListener. É através desse objeto que serão notificados os dispositivos que forem descobertos. Como a própria classe implementa esse objeto, passamos a palavra chave “this” no segundo parâmetro. Cada vez que um dispositivo é encontrado o método deviceDiscovered é chamado passando dois parâmetros. O primeiro é um objeto da classe RemoteDevice que representa o dispositivo encontrado. O segundo é um objeto da classe DeviceClass que permite determinar o tipo de dispositivo encontrado. Depois de localizado o dispositivo realizamos a busca por serviços utilizando a classe DiscoveryAgent. Para iniciar o processo de busca utilizamos o método searchServices da classe

DiscoveryAgent passando quatro parâmetros. O primeiro é um array de inteiros que especifica os atributos dos serviços que interessam. O segundo é um array de identificadores de serviço que permite especificar os serviços que o cliente está interessado. Fig. 10.

Figura 10. Tela software. Dispositivo localizado.

A classe UUID representa identificadores únicos universais. Para esse parâmetro o valor de UUID utilizado foi "0x1101" que corresponde ao perfil SPP – Perfil de Porta Serial. O terceiro argumento indica o dispositivo remoto onde será realizada a busca por serviços. O quarto parâmetro é um objeto DiscoveryListener que será usado para notificar os eventos da busca por serviços. Como a própria classe implementa esse objeto, passamos a palavra chave this nesse parâmetro. O método servicesDiscovered notifica os serviço encontrados, passando dois parâmetros. O primeiro é um inteiro que identifica o processo de busca. O segundo é um array de objetos ServiceRecord. Para abrir a conexão com o dispositivo remoto utilizamos o método estático open() da classe javax.microedition.io. Connector, passando como parâmetro a url de conexão. A url de conexão foi obtida durante o processo de busca por serviços, através do método getConnectionURL() da classe ServiceRecord passando dois parâmetros. O primeiro indica se deve autenticar e/ou criptografar a conexão. Utilizamos ServiceRecord. NOAUTHENT IC ATE _ NOENCRYPT nesse parâmetro para não autenticar e não criptografar a conexão. O segundo é um booleano que indica se o dispositivo deve ser mestre (true) ou não se não importa se é mestre ou escravo (false). Utilizamos false nesse parâmetro. O método open devolve um objeto distinto segundo o tipo de protocolo usado. No caso do SPP devolverá um StreamConnection. A partir do StreamConnection obtemos o fluxo de entrada e saída. Com a conexão estabelecida é possível ler e escrever informações.

Para realizar o controle do carro utilizamos as teclas direcionais de um aparelho celular (UP, DOWN, LEFT, RIGHT e SELECT). Fig. 11. Essas teclas foram mapeadas com números da seguinte forma: UP = 2, DOWN = 8, LEFT = 4, RIGHT = 6 e SELECT = 5.

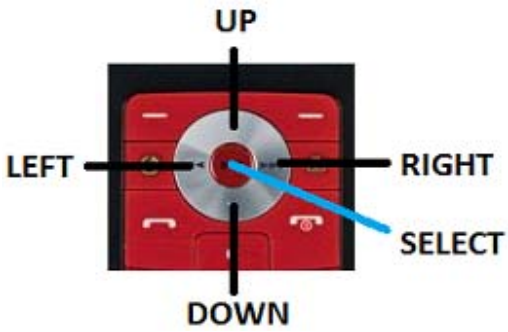

Figura 11. Teclado do celular.

Assim quando o usuário pressiona uma tecla no celular enviamos o número correspondente a essa tecla para o dispositivo móvel, facilitando o processo de comunicação. Para cada tecla pressionada a tela do dispositivo móvel é atualizada mostrando a direção que o usuário escolheu. Figs. 12,13 e 14.

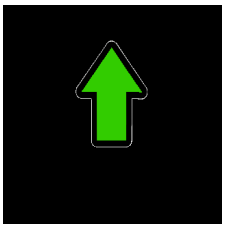
Figura 12. Tela software. Direção para frente.

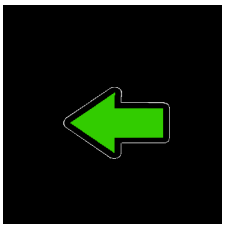
Figura 13. Tela software. Direção para esquerda.

Figura 14. Tela software. parado.

IV. Resultados e Discussão

A configuração do sistema ficou da seguinte forma, como mostra Fig. 15. Esta configuração poderá ser utilizada em outros sistemas que possuam os mesmos princípios.

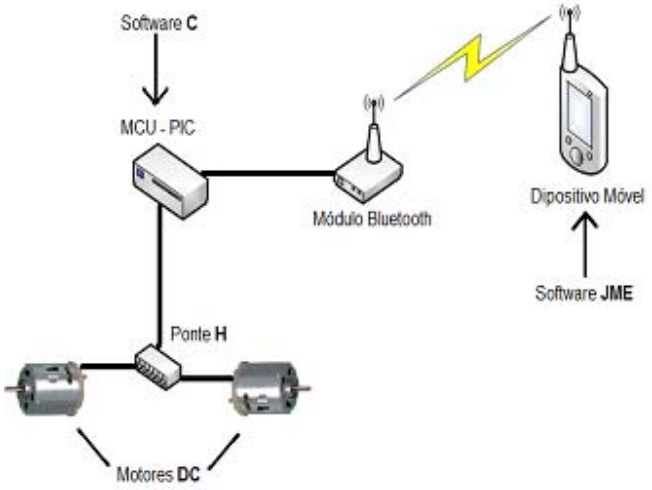

Figura 15. Esquemático.

O celular estabelece conexão com o módulo bluetooth KC-21 que esta ligado serialmente com o microcontrolador PIC 16F876A, que por sua vez esta ligado ao circuito que contém o CI L293D, que esta ligado aos motores DC. Podemos observar esta montagem nas figuras a seguir. Figs. 16, 17 e 18.

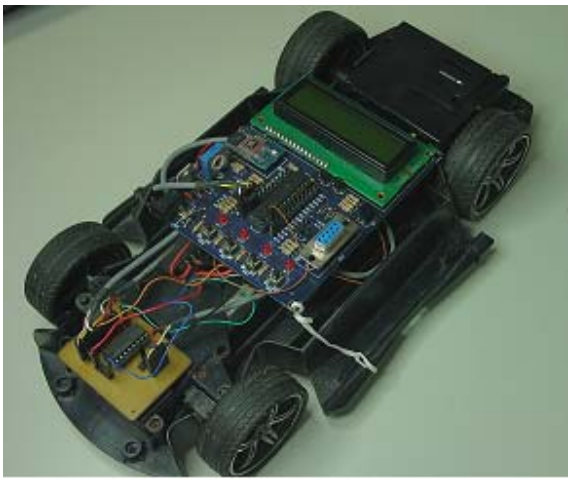

Figura 16. Carro montado.

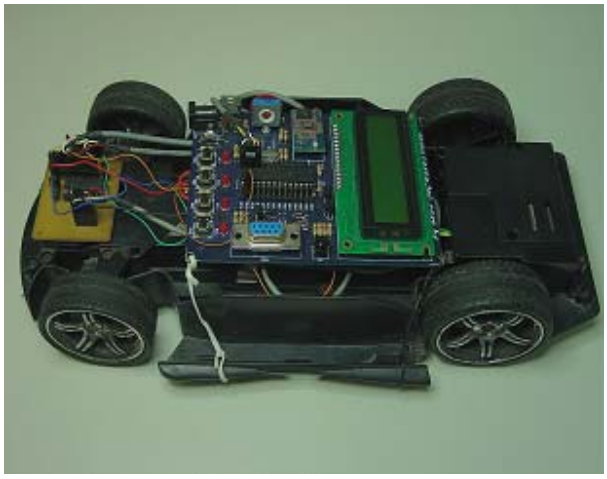

Figura 17. Visão lateral.

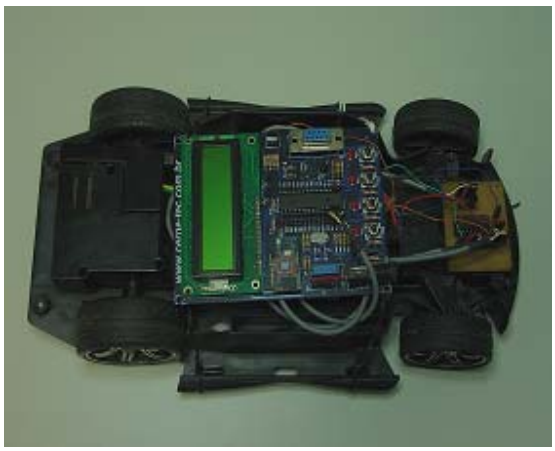

Figura 18. Visão superior do carro.

Foram realizados vários testes de distância e autonomia. Com relação à distância esta se mostrou limitada a dez metros, visto o modulo Bluetooth utilizado ser de classe II. Se for utilizado um modulo classe I essa distância aumenta para 100 metros. Os motores consomem bastante energia, o que torna a autonomia bastante reduzida, cerca de uma hora de uso contínuo.

O projeto desenvolvido cumpriu os objetivos propostos uma vez que o carro responde bem aos comandos sem atrasos ou interrupções e possui uma velocidade de deslocamento dentro do esperado. A comunicação entre o celular e o modulo Bluetooth é bem estável e de confiabilidade, o que torna a execução dos comandos precisos. Um ponto negativo é o fato da limitação da distância, visto que se for executado um comando e o carro sair da área de atuação do Bluetooth a conexão é perdida não dando a possibilidade de interromper ou parar sua execução.

V. CONCLUSÃO

O objetivo inicial do projeto foi alcançado, foram concluídos todos os passos propostos na elaboração deste trabalho. Além do produto final alcançado, este trabalho nos trouxe um maior entendimento e conhecimento das técnicas utilizadas. Como a aplicação é um controle de dispositivos de

propósito geral, esta configuração poderá ser utilizada em outros sistemas que possuam os mesmos princípios como aparelhos de ar-condicionado, TVs, computadores, portões automáticos, robôs industriais, etc. A tecnologia Bluetooth se apresentou como uma boa alternativa para comunicação sem fio entre dispositivos eletrônicos por ser uma especificação aberta, de baixo custo, baixo consumo de energia e está presente na maioria dos dispositivos eletrônicos .

O desenvolvimento mostrou a factibilidade do uso do celular como controle remoto, mas o uso da plataforma JME implica um incremento no desenvolvimento do software dado que o lema da linguagem Java (Write Once Run Anywhere – Escrever uma vez e executar em qualquer lugar) não se aplica nestes casos sendo necessário realizar diferentes mudanças de acordo com o tipo de celular a ser utilizado.

AGRADECIMENTOS

Agradecemos ao Ministério de Ciência e Tecnologia e ao Conselho Nacional de Pesquisa pelo apoio ao projeto através do Edital MCT/CNPq nº 11 /2007 - Extensão Inovadora 2007.

Agradecemos ao Dr. César Alberto Bravo Pariente pela sua ajuda com a editoração final do artigo.

REFERÊNCIAS

[1] Bluetooth Specification. Acesso em: abril de 2011: https://www.bluetooth.org/Technical/Specifications/adopted.htm.

[2] JavaTM APIs for Bluetooth. Acesso em: abril de 2011: http://www.jcp.org/en/jsr/detail?id=82.

[3] MOBILEROBOTICS. Página do MOBILEROBOTICS. Acesso em: abril de 2011: http://mobilerobotics.sourceforge.net/articles.php.

[4] Birth of the Bluetooth bots. Acesso em: abril de 2011: http://www.cs.uga.edu/~potter/robotics/TheFeatureArticle.htm.

[5] Celular SGH-E215. Acesso em: abril de 2011: http://www.samsung.com/br/consumer/cellular-phone/cellular-phone/music/SGH-E215WBLZTM/index.idx?pagetype=prd_detail..

[6] KC-21. Acesso em: abril 2011: http://kcwirefree.com/docs/KC21_Datasheet.pdf.

[7] Kit Didático Cerne Bluetooth. Acesso em: abril 2011: http://www.cerne-tec.com.br/detalhescerneblue.htm.

[8] Ponte H. Acesso em: abril 2011: http://resenhaeletronica.blogspot.com/2010/04/projetos-de-ponte-h.html.

[9] Circuito Integrado L293D. Acesso em: abril 2011: http://www.datasheetcatalog.net/pt/datasheets_pdf/L/2/9/3/L293D.shtml.

[10] Apostila sobre Modulação PWM. Acesso em: abril de 2011: http://www.eletronica.org/arq_apostilas/apostila_pwm.pdf.

[11] PIC 16F876A. Acesso em: abril 2011: http://www.datasheetcatalog.com/datasheets_pdf/P/I/C/1/PIC16F876ATI_ML.shtml.

Fabrício Santiago Andrade graduou-se em Ciência da Computação na Universidade Estadual de Santa Cruz em 2008. Finalizando o curso de especialização em Sistemas Embarcados para Aquisição de Dados Remotos na Universidade Estadual de Santa Cruz, ingresso em 2009. Seus interesses estão relacionados com sistemas embarcados, desenvolvimento Web e Mobile.

Geraldo Gomes Bomfim Junior graduou-se em Ciência da Computação na Universidade Estadual de Santa Cruz em 2008, pós-graduando em Sistemas Embarcados na Universidade Estadual de Santa Cruz ingresso em 2009. Seus interesses estão relacionados com robótica, eletronica e sistemas embarcados.

M. Torres Nasceu em Cali, Colômbia, aos 18 de Abril de 1968. Graduou-se em Engenharia Elétrica na Universidad Del Valle em 1991, Mestre em Sistemas Eletrônicos na Universidade de São Paulo em 1994 e Doutor em Sistemas Eletrônicos na Universidade de São Paulo em 1999. Trabalha no Departamento de Ciências Exatas e Tecnológicas da Universidade Estadual de Santa Cruz, Ilhéus, Bahia, onde é professora desde 2004 e coordena o Programa de Pós-Graduação em Sistemas Embarcados. Seus interesses estão relacionados com sistemas embarcados, computação de alto desempenho e bioinformática.

F. M. Milian, nasceu em Havana, Cuba, o 23 de novembro de 1977. Graduo-se em Licenciado em Física Nuclear no Instituo Superior de Ciências y Tecnologia Nucleares (ISCTN) Cuba em 2001, Mestre em Física Nuclear com ênfase em Instrumentação Nuclear na mesma instituição em 2002, e Doutor em Física na Universidade de São Paulo em 2006. Trabalha no Departamento de Ciências Exatas e Tecnológicas da Universidade Estadual de Santa Cruz, Ilhéus, Bahia, onde é professor desde 2008. Seus interesses estão relacionados com sistemas embarcados e simulação computacional de fenômenos fisicos.

Teodoro Pires Junior nasceu em Itabuna Bahia aos 14 de Abril de 1963. Graduou-se em Filosofia na Universidade Estadual de Santa Cruz em 2001, Especialista em Aplicações Pedagógicas de Computadores na Universidade Estadual de Santa Cruz em 2003, Mestre em Desenvolvimento Regional e Meio Ambiente na Universidade Estadual de Santa Cruz em 2007 e Doutorando em Ciência da Computação na Universidade Federal da Bahia. Vice-coordenador do Programa de Pós-Graduação em Sistemas Embarcados. Seus interesses estão relacionados com sistemas embarcados, redes sem fio e redes veiculares.