MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS ELETRÔNICOS
PROFESSOR: **SAMIR BONHO**
UNIDADE CURRICULAR: **INTRODUÇÃO À PROGRAMAÇÃO ANDROID**

Projeto Final
Robô mutuamente controlado por dispositivos Android através da comunicação por sockets UDP

Victor Augusto dos Santos
Florianópolis, 31 de maio de 2014

ÍNDICE

Objetivo

Este trabalho tem como objetivo aplicar os conceitos abordados na cadeira optativa de Programação para Dispositivos Android, ministrada pelo professor Samir Bonho, em um projeto envolvendo técnicas de comunicação de rede por socket UDP, e conceitos como utilização de threads assíncronas, programação multi-threads, processamento digital de imagens e interface de controle utilizando múltiplas activities.

Metodologia

O diagrama de blocos abaixo demonstra como o sistema foi estruturado, tanto do ponto de vista de montagem e utilização de hardware quanto no que diz respeito à estrutura de rede empregada.

As ligaçoẽs marcadas na cor preta indicam as conexões de meio físico (ar ou cobre), enquanto as ligações em vermelho indicam as conexões lógicas entre os blocos.

As áreas tracejadas representam uma macro-visualização do projeto, com objetivo de facilitar o entendimento da solução completa. .

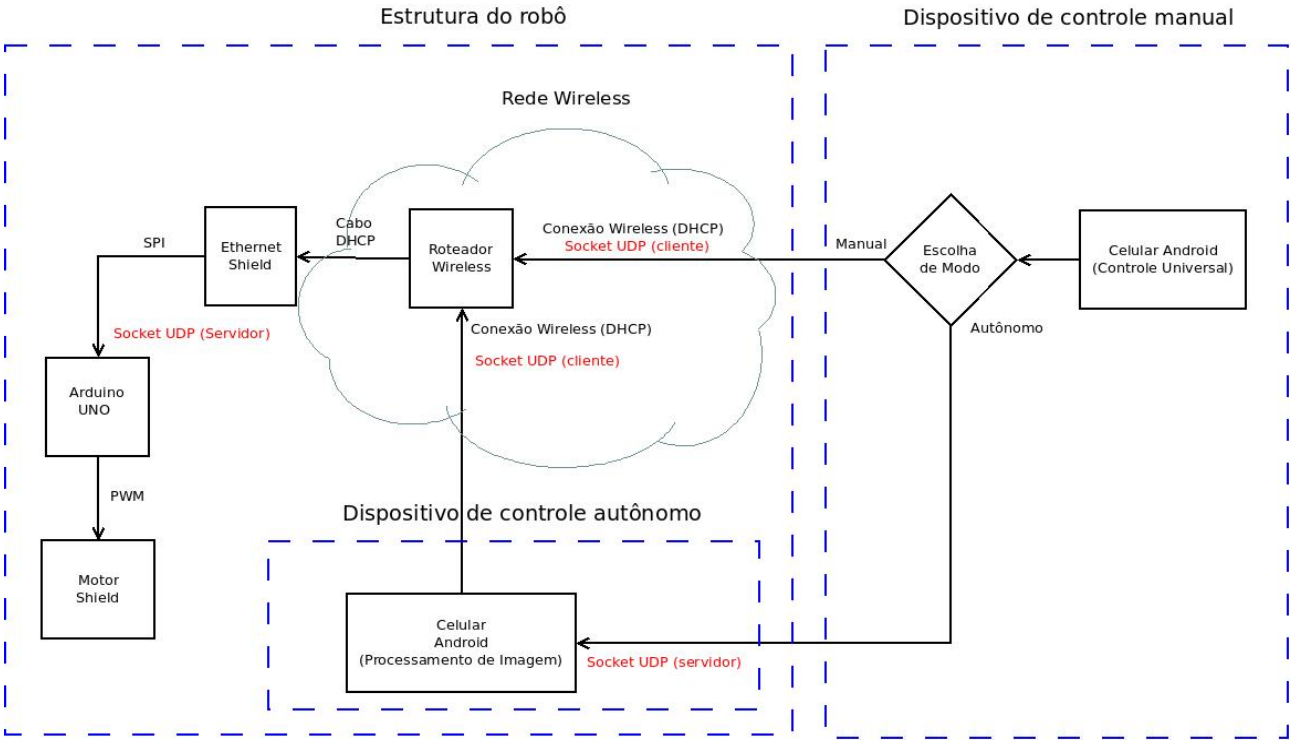

Figura 1: Diagrama de blocos do projeto

Apesar dos dispositivos wireless estarem recebendo IP por DHCP, os IP's que recebem estão associados aos endereços físicos de suas interfaces de rede (MAC binding).

Funcionamento

O funcionamento do robô é simples, o dispositivo móvel Android responsável pela interface de controle (com o aplicativo de controle instalado) se conecta à rede estabelecida pelo roteador wireless (vs_robot) e entra na interface de escolha de modos de controle: manual ou autônomo.

No modo manual o usuário controla o robô por meio de botões implementados na interface de controle, ao apertar cada botão um socket UDP é aberto e os dados são roteados pelo roteador wireless para o IP e porta do server UDP instanciado no Arduino, através da interface de rede do shield Ethernet conectado.

O firmware do Arduino trata a mensagem recebida e aciona os motores de acordo com o comando especificado pelo usuário.

Ao escolher a opção de controle por modo autônomo o usuário deve apertar o botão “enter in AM” na interface de controle para instruir o robô a se comportar de maneira autônoma.

No modo autônomo o robô utiliza a câmera do dispositivo Android em sua estrutura (com o devido software instalado) para seguir um objeto (no caso foi utilizada uma bola de plástico na cor rosa), se movendo de acordo com a posição relativa da bola em relação aos limites da tela do dispositivo. Ao encontrar o objeto, um socket UDP é aberto para o IP e porta do Arduino para efetuar o envio do comando de movimentação e consequentemente efetuar o acionamento dos motores.

O algoritmo de processamento de imagem foi implementado utilizando a biblioteca OpenCV.

Desenvolvimento de Software Java/Android

Software de Controle do robô

Para a implementação do controle do robô, foi utilizado puramente a API de desenvolvimento Java/Android utilizando a IDE de programação Eclipse, onde foi criado um aplicativo dividido em três activies:

- **MainActivity (Tela inicial do aplicativo):** Onde o usuário pode escolher a maneira a qual deseja controlar o robô, sendo esta manual ou autônoma. A escolha dos modos é implementada utilizando o conceito de comunicação entre Activities por meio de Intents.

- **ManualActivity:** Se trata da tela onde o usuário tem acesso ao controle do robô de forma totalmente manual. Desta forma ele pode mandar comandos de movimentação para o robô por meio das teclas "*up*", "*down*", "*left*" e "*right*" e voltar à tela inical pressionando a tecla "*exit*".
  Quando um botão é apertado, uma tarefa assíncrona é executada para abrir o socket UDP para o Arduino e passar os comandos referentes à movimentação desejada.Quando o comando é passado o socket é fechado.

- **AutonomousActivity:** A interface de acionamento de modo autônomo é bem simples, faz uso de dois botões, o "*enter in AM*" e o botão "*exit*".
  O botão de "exit" tem a mesma funcionalidade antes explicada para a ManualActivity. Ao apertar o botão "enter in AM" o usuário aciona o algoritmo de processamento de imagem em execução no dispositivo Android embarcado na estrutura do robô.

Software de Processamento de Imagem

O sofware de processamento de imagem faz uso da biblioteca OpenCV portada para Java.

Quando a aplicação se inicia, uma thread assíncrona é iniciada e fica esperando receber uma mensagem específica para acionar o modo autônomo.
Ao receber esta mensagem, um callback para eventos de câmera é ativado para que o algoritmo de processamento seja aplicado a cada frame obtido pela câmera.

O frame é então recebido, convertido para o espaço de cores HSV, filtrado de forma paramétrica visando encontrar objetos de cor rosa, filtrado novamente utilizando um filtro Gaussiano de kernel 3x3 para suavizar os contornos da imagem resultante e por fim os contornos do objeto encontrado são delimitados por um bounding-box ao redor do objeto (caixa retangular na cor vermelha).

Um padrão de movimentação foi criado de acordo com o posicionamento relativo entre o objeto encontrado e a tela do dispositivo, ou seja, se o objeto está sendo visualizado à direita da tela no dispositivo Android, uma mensagem será enviada para o Arduino por meio de socket UDP, para movimentar o robô para direita.

Discussão sobre os resultados

O sistema se comportou conforme o esperado, entretanto, alguns problemas retardaram a conclusão do projeto.

O circuito integrado presente no Ethernet Shield do Arduino responsável por implementar a camada física de rede, Wiznet W5100, parece não se perder ao receber sequências rápidas de pacotes de rede.

Isto fez com que fosse necessário implementar o conceito de time-sharing no firmware do Arduino para que o servidor recebesse os pacotes de forma assíncrona dentro uma uma janela de tempo pré-determinada.

Além disso foi necessário implementar um buffer circular para armazenar as mensagens recebidas pelo Arduino, pois a função que implementa o recebimento dos pacotes de rede UDP é orientada a byte.

Foi adotado o uso do envio de mensagens por tarefas assíncronas para aumentar a segurança do protocolo de comunicação, o socket é aberto somente quando uma mensagem é enviada.

A maior dificuldade encontrada foi na implementação do software de processamento de imagem, já que é o bloco mais complexo do projeto.

Alguns dos recursos do Android utilizados:

1. **Leitura do estado da interface de rede e SSID:** Testa se o celular está conectado à à rede sem-fio do robô, do contrário o aplicativo retorna um erro e volta para o sistema operacional;

2. **Configuração de Wake-Lock:** O serviço nativo de wake-lock foi habilitado de modo a evitar o bloqueio por inatividade da tela do celular;

3. **Gerenciamento do callback da câmera:** O callback da câmera foi gerenciado de forma a ser ativado somente quando o modo autônomo for habilitado, de modo a economizar bateria;

Devido ao fato do shield Ethernet do Arduino não conseguir lidar com bursts de pacotes de rede, foi necessário implementar um filtro decimador de quinta ordem para reduzir a amostragem dos pacotes de rede enviados, reduzindo o *throughput*.

Fotos do projeto e exemplos de utilização

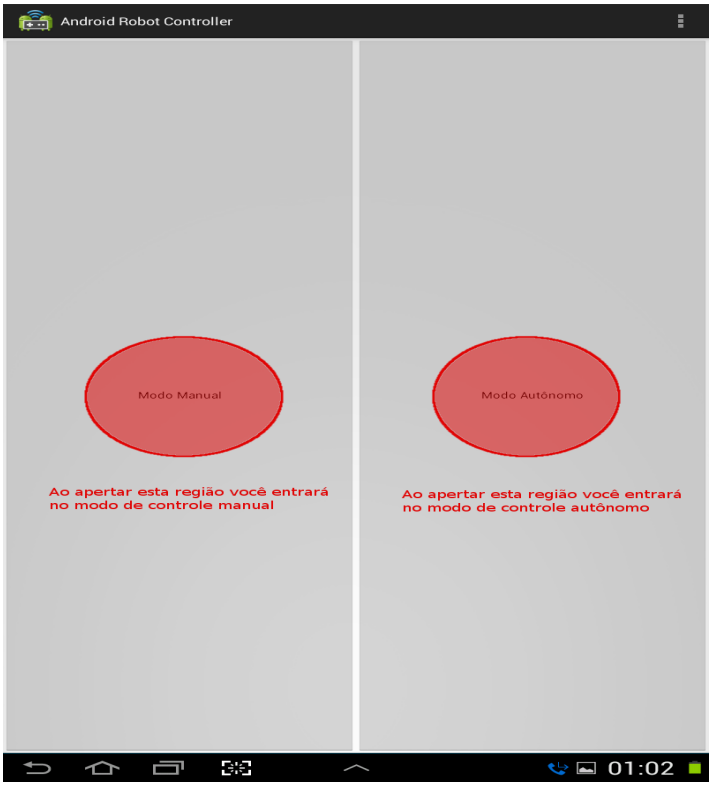

Figura 2: Tela de controle: Tela Inicial

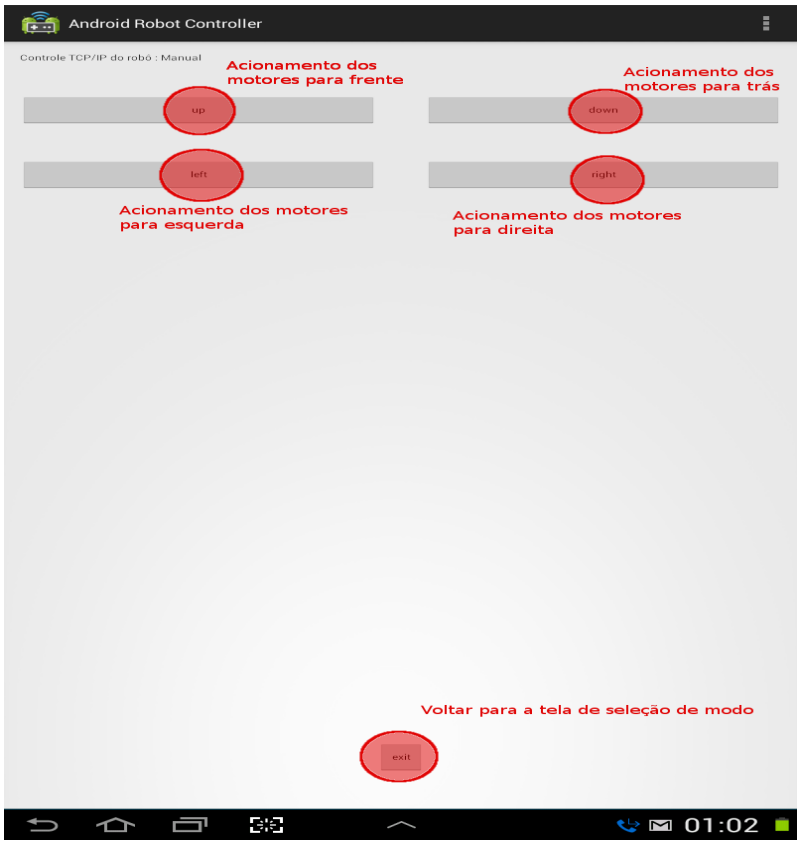

Figura 3: Tela de controle: Modo Manual

Figura 4: Tela de controle: Modo Autônomo

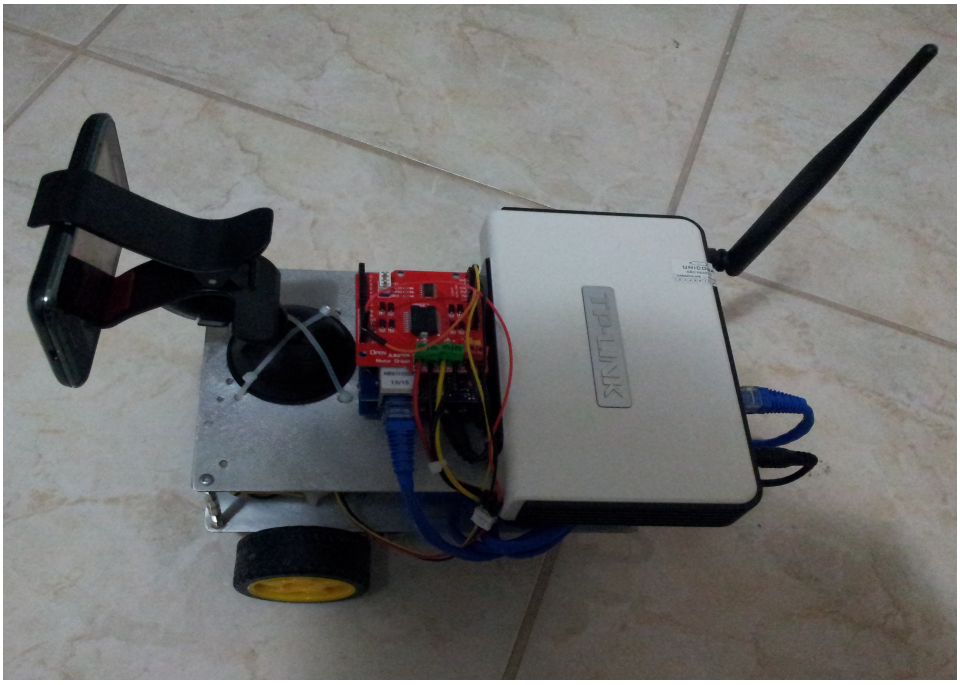

Figura 5: Robô: Vista superior

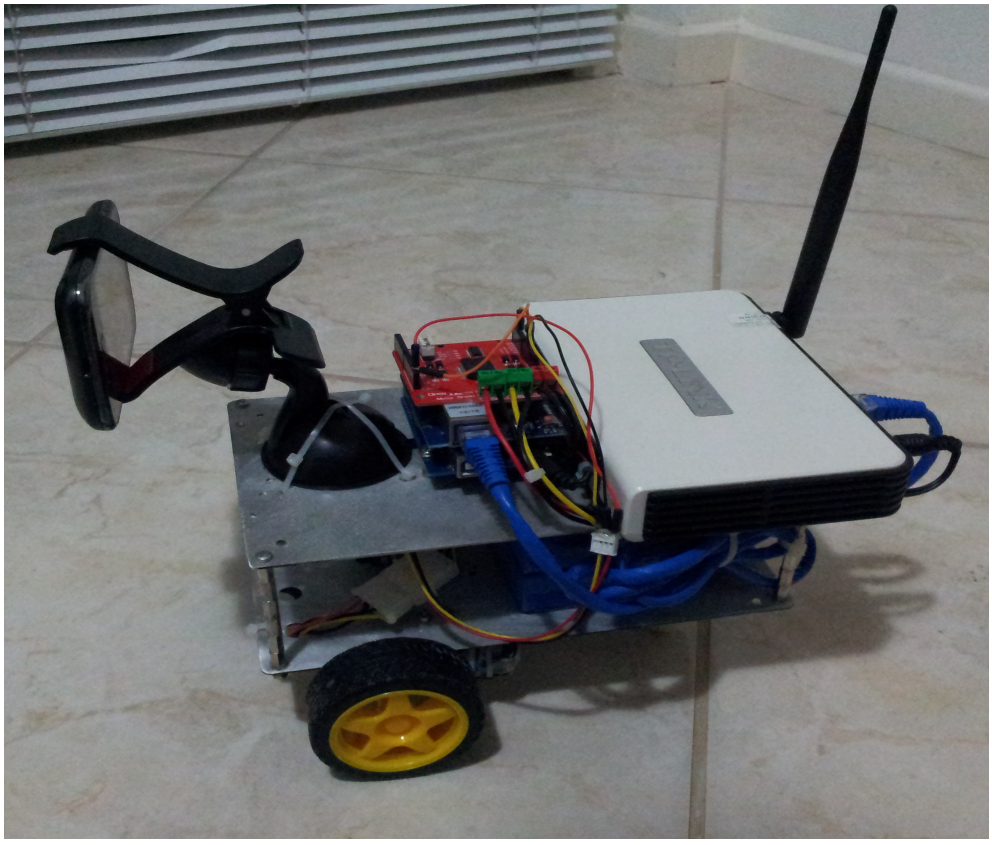

Figura 6: Robô: Vista lateral

Conclusão

Além do objetivo inicial deste projeto, de atender implementar um projeto utlizando recursos do sistema operacional Android, um outro objetivo foi alcançado, que é o de aprender a desenvolver software utilizando a plataforma Android.

Desenvolver para uma plataforma não é apenas desenvolver o código fonte, compilar (ou interpretar) e gerar um arquivo binário, é entender como a plataforma funciona, entender a documentação que lhe é oferecida, aprender a inspecionar as mensagens de erro através da ferramenta de depuração (logCat neste caso), aprender a utilizar os métodos fornecidos pela API desta plataforma. Tudo isto foi de extrema importância para que a implementação deste projeto tenha se concluído de forma bem sucedida.

Entretanto, são necessários trabalhos futuros de melhoria, como:

1. Melhoria da interface gráfica de controle;
2. Implementação de um feedback gráfico para o modo autônomo;
3. Melhoria na precisão do algoritmo detector de objetos;
4. Acionamento automático pelo Arduino do dispositivo Android quando o robô for ligado;
5. Utilização do GPS e da bússola para auxiliar o modo autônomo;
6. Gerenciamento aprimorado de energia;
7. Implementação de um método de depuração por Wifi;
8. Redundância de rede;

Referências

- Stack Overflow.http://stackoverflow.com/.
  Acessado em 31 de maio de 2014.

- Android Developers. http://developer.android.com/reference/android/package-summary.html.
  Acessado em 31 de maio de 2014.

Código Fonte

Todos os arquivos deste projeto estão disponibilizados na íntegra no meu github:

https://github.com/victorsantosdev/pga_robot.git