# ERUS – Team Description Paper – IEEE Open 2013

Roberta Lima Gomes[1], André Georghton Cardoso Pacheco [1], Lucas Catabriga Rocha[1], Juan França Muniz de Souza[1], Ebenézer Nogueira da Silva[1], Guilherme Artém dos Santos[1], Vanderlei Vieira de Souza Filho[1], Renan Sarcinelli[1], Igor Oliveira Nunes[1], Thaylo Xavier de Freitas[1]

*Departamento de Informática [1]*

*Centro Tecnológico, Universidade Federal do Espírito Santo*
*Av. Fernando Ferrari 514 – Goiabeiras, Vitória ES*

**Abstract—** This article reports the work of the team ERUS building a robot to solve the challenge posed in IEEE Open 2013 category. The robot collects empty cans on an irregular terrain made of sand and deposits them in a trash bin. The system built uses an Android phone connected to an Arduino board.

**Keywords—** Robotics, Android, Arduino.

**Resumo—** Este artigo relata o trabalho desenvolvido pela equipe ERUS na construção de um robô para solucionar o desafio proposto na categoria IEEE Open 2013. O robô construído coleta latas de refrigerante em um terreno irregular feito de areia e as deposita em uma lixeira. O sistema construído é composto por um celular Android conectado a uma placa Arduino.

**Palavras-chave—** Robótica, Android, Arduino, LARC.

## 1 Introdução

O desafio IEEE OPEN 2013 proposto na Competição Latino Americana de Robótica (em inglês *Latin American Robotics Competition abreviado por LARC*) tem como temática a limpeza de resíduos em uma praia. A arena da competição consiste em uma circunferência de areia cercada por uma região azul que representa o mar, simulando assim uma ilha. Nesta arena um robô, completamente autônomo, deverá coletar latas de alumínio espalhadas pela mesma e depositá-las em uma lixeira também localizada na arena. O robô não poderá tocar em nenhum obstáculos presentes na arena, obstáculos estes que serão: uma cadeira, uma sombrinha de praia e um manequim. Além disso o robô não pode adentrar a região azul, que representa o mar. Caso descumpra uma das normas acima, a equipe sofrerá punições na pontuação final e/ou sofrerá reinício da rodada.

A Equipe ERUS (*Equipe de Robótica da Universidade Federal do Espírito Santo*), formada por alunos de engenharias da UFES, desenvolveu um robô para solucionar este desafio. O robô, completamente projetado em softwares de desenvolvimento 3D, foi desenvolvido para navegar em ambiente com terreno irregular, assim como os terrenos existentes em praias. Sua estrutura física é composta, principalmente, por peças de acrílico e plástico. A parte eletrônica do robô é composta por um Arduino Mega ADK que se comunica com um celular com sistema operacional Android. Este celular é utilizado como "cérebro" central do robô. Nele são capturadas e processadas as imagens da arena. Para isso, as linguagens de programação utilizadas foram C, C++ e Java.

A seguir são apresentados mais detalhes sobre o desafio e as tecnologias utilizadas no desenvolvimento do robô.

## 2 Objetivo

Este desafio tem como objetivo a construção e programação de um robô que trabalhe de forma completamente autônoma. O robô deve ser capaz de identificar todos os objetos inclusos, tais como: Cadeira, sombrinha, manequim, lixeira e latas. Uma vez identificado o obstáculo, o robô deve possuir inteligência suficiente para discernir quais destes objetos são obstáculos e quais são lixo. Além disso, o robô deverá navegar somente dentro do espaço permitido na arena, ou seja, na areia, e coletar o máximo de latas encontradas. Após coletadas, as mesmas deverão ser depositadas em um lugar específíco demarcado na arena.

## 3 Considerações do Ambiente

O cenário proposto simula uma ilha. Para isso, é construído uma arena, com forma circular, cujo as bordas são demarcadas pela cor azul. O tamnaho total da arena possui 6 metros quadrados. Des-

tes, 5,5 metros quadrados são preenchidos por areia. O restante representa o mar e será preenchido por um material de cor azul.

Como já mencionado, a arena deverá conter latas de alumínio, pintadas de preto, que representarão o lixo da praia (que serão colocadas de forma aleatória por toda a arena); um manequim, que representará um banhista; uma cadeira e um guarda sol, que representarão o material utilizado pelo banhista; e por fim uma bandeja circular vermelha, que representará um depósito de lixo onde as latas de alumínio deverão ser recolhidas e depositadas pelo robô.

Um esboço da arena é representado na figura 1:

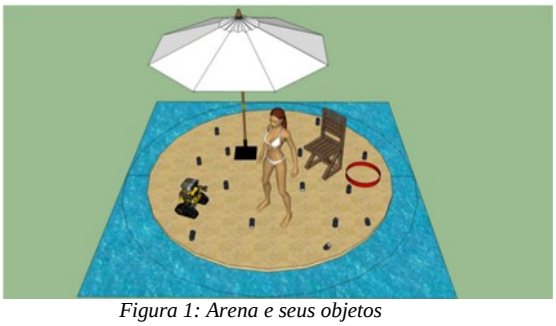

*Figura 1: Arena e seus objetos*

Um fator extremamente importante a ser considerado no ambiente é luminosidade do local. A arena poderá estar tanto em um ginásio aberto, e exposta ao sol, quanto em um ginásio fechado. Portanto fica a cargo da equipe solucionar este problema.

## 4 Desenvolvimento do robô

A seguir será apresentado todo o desenvolvimento do robô divido em três partes: Estrutura, Eletrônica e Programação.

### *4.1 Estrutura do robô*

A primeira parte projetada na contrução do robô foi a sua estrutura. Para auxiliar nesta fase, foram utilizados softwares de modelagem 3D como *OpensCad* [1] e *SolidWorks* [2]. Portanto, toda estrutura do robô antes de ser físicamente construída foi modelada em 3D para se obter uma melhor performace na construção.

Vale a pena ressaltar que algumas das peças modeladas nos softwares citados acima foram impressas para facilitar a construção física da estrutura. Isso apenas intensifica a importância de uma modelagem 3D antes da construção física.

A estrutura física do robô possui basicamente 3 partes principais: garra, caçamba e a base. Cada uma destas partes será brevemente detalhadas a seguir:

- Garra: A garra do robô consiste, basicamente, em uma “vassoura”. A ideia é que o robô possa “varrer” as latas encontradas para dentro de sua caçamba, para posteriormete depositá-las na lixeira. A garra foi construída utilizando acrílico e peças 3D modeladas e impressas. Para movimentar a garra é utilizado apenas um motor.
- Caçamba: Com intuito de armazenar as latas antes das mesmas serem depositadas, a caçamba do robô foi contruída utilizando acrílico. Esta caçamba foi modelada e contruída com uma inclinação de 20º para que possa ser aproveitada a gravidade para as latas caírem. Sendo assim, o único movimento necessário para depositar as latas é abrir a comporta traseira que as prendem. Para isso é utilizado um *servo motor* que realiza o movimento necessário para abrir e fechar a comporta.
- Base: Toda base do robô também foi construída com acrílico. A base sustenta toda estrutura do robô, como a caçamba e a garra citadas acima. Também, nela, são posicionados toda eletrônica, motores e celular pertinentes ao robô.

Como o robô deverá se movimentar em terreno irregular, ele foi construído com tração 4x4 para evitar que o mesmo fique preso em alguma irregularidade da arena.

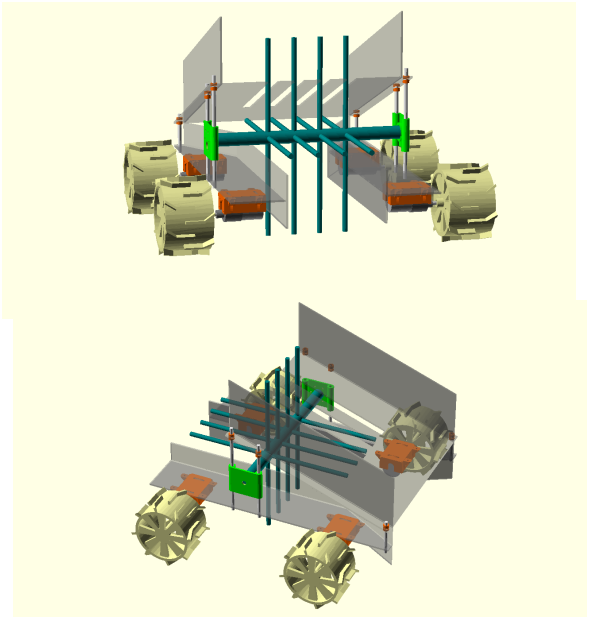

*Figura 2: Modelagem 3D do robô*

### *4.2 Eletrônica*

Paralelamente a estrutura, foi desenvolvido toda eletrônica do robô. Para controlar o robô foram utilizados um *Arduino Mega ADK* [3] e um Celular com *Android* [4] conectados. A ideia principal é processar a imagem do ambiente no *Android* e, baseado, nesse processamento enviar as informações para *Arduino* para que este comande os movimentos do robô.

#### *4.2.1 Arduino*

Como já mencionado foi utilizado um microprocessador *Arduino Mega ADK* para realizar as

operações de baixo nível necessárias para o funcionamento do robô. A grande vantagem deste microcontrolador é que ele já implementa uma interface de comunicação USB evitando, assim, a utilização de um *shield USB [5]* para a comunicação com o *Android.*

*O Arduino* é responsável por controlar todos os motores, servo motor, sensores de distância, bem como os botões de *on/off* e *start.* Sendo assim, para movimentação dos motores elétricos foi necessário a utilização de duas pontes H, definindo assim a direção dos mesmos.

O controle executado pelo *Arduino* sofre forte influência das informações de imagens processadas pelo *Android.* De certo modo, pode-se afirmar que existe uma relação de *Mestre e Escravo* entre o *Arduino* e o *Android.* O *Arduino* obedece os comandos enviados pelo *Android*, podendo apenas realizar interrupções geradas pelos sensores de distância, caso o mesmo consiga detectar um obstáculo não encontrado pela câmera.

4.2.2 *Android*

Aparelhos celulares modernos possuem grande poder de processamento, além disso possuem vários sensores embutidos, como uma câmera de vídeo, bússola e acelerômetro. Isto cria uma ambiente poderoso para o desenvolvimento de robôs móveis e por conta disso este sistema foi escolhido pela ERUS como sistema principal a ser utilizado no robô. O celular utilizado foi um *Motorola Milestone I*, rodando *Android v2.2*.

O *Android* recebe periodicamente valores dos sensores de bússola e acelerômetro além de processar a imagem capturada pela câmera de vídeo existente no celular. De acordo com as informações processadas, o *Android* envia as ordens para o *Arduino* por meio de uma troca de informações e um protocolo criado pela equipe.

*4.3 Programação*

*Ne*sta seção será apresentado tudo o que tange a programação no desenvolvimento do robô. Fica a cargo da programação o controle do *Android* e do *Arduino* e a comunicação de ambos. Além disso, aqui é criada uma interface de comunicação entre o *Android* e um computador, denominado Fiscal. Como o próprio nome sugere, o Fiscal é utilizado como uma fonte de auxílio para depuração de erros e deve se deixar bem claro que essa funcionalidade não é utilizada durante a execução oficial do robô, que funciona de maneira completamente autônoma. Para criar todas as funcionalidade citadas acima, foi necessário desenvolvimento nas seguintes linguagens de programação: C e C++, para o desenvolvimento no *Arduino*, e Java, para desenvolvimento no *Android e* Fiscal.

*4.3.1 Comunicação Arduino - Android*

A comunicação entre o Arduino e o Android ocorre através de um cabo USB. Essa comunicação é facilitada pelo fato do modelo do Arduino utilizado já realizar uma interface de comunicação USB sem necessidade de eletrônica adicional.

O celular Android suporta o ADB (em inglês Android Debug Bridge) [6], uma ferramenta capaz de se comunicar com o Android através da interface USB. Para que o Arduino seja capaz se de comunicar com o *ADB*, é necessário implementar um protocolo de comunicação em baixo nível. Este protocolo é implementado pela biblioteca *MicroBridge* [7] e, portanto, foi utilizada no código de comunicação entre as duas tecnologias.

Nesta configuração a conexão é vista como uma conexão padrão através de um Socket pelo código que roda no *Android*. No *Arduino*, a conexão é realizada através de funções da *MicroBridge*. Além do protocolo implementado pela *MicroBridge*, é necessário, ainda, um outro protocolo de comunicação básico, porém, mais alto nível. Este último protocolo foi elaborado para que mensagens fossem trocadas entre o *Android* e o *Arduino.*

*4.3.2 Comunicação Android – Computador*

Como já descrito, esta conexão não é utilizada no momento da competição, e sim na depuração de erros, monitoramento dos sensores do robô durante os treinos e calibração de valores. Decidiu-se implementar esta funcionalidade para facilitar o desenvolvimento e acelerar a calibração de valores. É possível gravar dados de calibração em um cartão SD no celular através da conexão com o computador e, assim, quando o robô estiver operando autonomamente o cartão SD é lido e os últimos dados enviados pelo computador são utilizados. Também é possível executar o robô em um modo de monitoramento, no qual todos os valores de sensores e variáveis importantes são enviadas ao computador, sendo possível observar possíveis falhas em tempo real.

Para esta finalidade foi desenvolvida, em Java, uma interface gráfica de monitoramento denominada Fiscal. O meio utilizado para esta comunicação é a interface *Wireless* disponível no celular e em computadores. O protocolo utilizado é o mesmo que protocolo criado para troca de mensagens entre o *Arduino* e o *Android*, sendo que algumas mensagens podem ser repassadas desde o *Arduino* até o computador e vice-versa.

*4.3.3 Visão Computacional*

A visão computacional é toda realizada dentro do celular *Android.* Entretanto, o *Android* não possui nenhuma biblioteca para tratamento de imagens. Sendo assim, se fez necessário a utilização da biblioteca *OpenCV para Android* [8].

Para o processamento da imagem, é necessário a implementação de algoritmos que identifiquem os objetos que estão na arena. Para auxiliar neste ponto, a equipe utilizou o *Color Blob Detector*, um algoritmo que é capaz de detectar as diferentes cores nas imagens.

Inicialmente, ao se receber a imagem, é realizada uma conversão de RGB para HSV para facilitar o processamento da mesma. São especificadas as

cores a serem reconhecidas a um raio mínimo pré-estabelecido. Após a identificação é retornado a imagem com os objetos detectados atraveś de um contorno.

Identificado o ambiente, o *Android* deve tomar decisões sobre o mesmo: Continuar a explorar, coletar lata, depositar lata, dentre outras decisões possíveis. Tomada essa decisão, uma mensagem de ordem é enviada ao Arduino e logo após executada pelo robô.

## 5 Conclusão

Com o avanço da tecnologia, os celulares estão se tornando mais poderosos, em nível de processamento, e também cada vez mais completos, visto que neles há inserção de vários sensores. O uso integrado de celulares com sistemas operacionais, como o *Android*, e microcontroladores, como o *Arduino*, demostrou ser bastante eficiente e confiável para o desenvolvimento de robótica autônoma. Além disso, a utilização de um celular também demostrou ser uma opção com um ótimo custo benefício, devido ao fato do mesmo já possuir diversos sensores que se fossem adquiridos de forma separada aumentaria, e muito, o custo total do projeto. Por fim, pode-se afirmar, devido aos fatos já expostos, que a a robótica tem se tornado mais acessível para a maioria das pessoas.

## Referências Bibliográficas


[1] http://www.openscad.org/

[2] http://www.solidworks.com/

[3] http://arduino.cc/en/Main/ArduinoBoardADK

[4] http://www.android.com/

[5]http://www.circuitsathome.com/products-page/arduino-shields/usb-host-shield-2-0-for-arduino

[6] http://developer.android.com/tools/help/adb.html

[7] http://code.google.com/p/microbridge/

[8] http://opencv.org/platforms/android.html