UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

LUIS GUILHERME MACHADO CAMARGO
PEDRO ALBERTO DE BORBA
RICARDO FARAH
STEFAN CAMPANA FUCHS
TELMO FRIESEN

# MAPEAMENTO DE AMBIENTES COM O ROBÔ BELLATOR

ANÁLISE TECNOLÓGICA

**CURITIBA**

**2013**

LUIS GUILHERME MACHADO CAMARGO
PEDRO ALBERTO DE BORBA
RICARDO FARAH
STEFAN CAMPANA FUCHS
TELMO FRIESEN

# MAPEAMENTO DE AMBIENTES COM O ROBÔ BELLATOR

Análise tecnológica apresentada à Unidade Curricular de Oficina de Integração 3 do Curso de Engenharia de Computação da Universidade Tecnológica Federal do Paraná como requisito parcial para aprovação.

**CURITIBA**

**2013**

**SUMÁRIO**

# 1 ANÁLISE TECNOLÓGICA

Nesta seção está explicitada, primeiramente, uma visão geral do projeto. Em seguida, há uma discussão detalhada a respeito dos requisitos de cada parte fundamental – estação base, sistema de comunicação e sistema embarcado. Por fim há uma enumeração das alternativas tecnológicas pesquisadas e das escolhidas para o preenchimento dos requisitos.

## 1.1 VISÃO GERAL DO PROJETO

O projeto, como foi idealizado de um ponto de vista geral, consiste em um robô controlado manualmente que seja capaz de efetuar mapeamento em duas dimensões de ambientes. Um usuário humano controlará e monitorará um computador – a estação base – através do qual poderão ser enviados comandos de movimentação ao robô (via teclado). Informações sobre o posicionamento do robô e dos obstáculos detectados por ele serão recebidas na estação base em tempo real. Imagens instantâneas de uma câmera posicionada no robô – aspecto explicado mais à frente – poderão ser visualizadas pelo utilizador.

O sistema de comunicação deverá ter, ao menos, alcance máximo de 20 metros. Visto que toda a comunicação entre a estação base e o sistema embarcado será feita por um único canal, a velocidade e o tipo de fluxo de transmissão de dados devem ser suficientes para o envio de comandos de movimentação ao robô, recebimento de dados de leituras de sensores e recebimento de imagens da câmera em tempo real.

O sistema embarcado é constituido, em suma, pelo robô. Ele deve ser capaz de se mover para frente e para trás e girar para a esquerda e direita. Uma visualização em tempo real do ambiente pelo utilizador, tendo o objetivo de facilitar o controle de movimentação manual, poderá ser feita através de imagens instantâneas geradas por uma câmera fixa instalada no robô.

O robô deve ser capaz de obter dados para cálculos (na estação base) de sua velocidade e deslocamento. Um aspecto a ser trabalhado em relação a isso é a atenuação de erros em decorrência de escorregamento, giros em falso ou trepidação de rodas, visando, dessa forma, a

utilização futura do robô em condições não ideais de terreno. Obstáculos próximos – em uma distância mínima de 30 cm e máxima de 150 cm – devem ser detectados de modo a possibilitar a confecção do mapa 2D em tempo real na estação base.

## 1.2 REQUISITOS

### 1.2.1 Estação base

Esta seção descreve os requisitos da estação base, que foram elaborados de forma a satisfazer os objetivos do projeto.

- O *software* será executado em um computador pessoal.
  - O programa deverá ser executado razoavelmente em computadores com recursos de *hardware* comparáveis aos padrões atuais (pelo menos 2GB de memória RAM DDR2 ou melhor e processador *dual-core*).
  - O *software* deverá ser multiplataforma, ou seja, executar em diferentes sistemas operacionais (no mínimo Linux e Windows).
  - Preferencialmente bibliotecas e ferramentas livres (e gratuitas) deverão ser utilizadas no desenvolvimento do *software*.
- O *software* deve possuir uma interface gráfica.
  - Um utilizador, através da interface gráfica, será capaz de controlar o robô enviando comandos de movimentação especificados pelo teclado.
  - O usuário receberá a imagem em tempo real (preferencialmente com atrasos não muito consideráveis) de uma câmera fixa instalada no robô.
  - Os dados instantâneos de velocidade e posição do robô serão mostrados ao usuário na interface gráfica.
  - Um mapa 2D do caminho percorrido e dos obstáculos detectados pelo robô será gerado, na interface gráfica, à medida em que houver movimentação do mesmo. O caminho percorrido pelo robô será representado visualmente por pontos, gradualmente posicionados no mapa. Os obstáculos serão representados por marcações nas localidades onde houve detecção de objetos por sensores. Todos os pontos representados no mapa serão gerados a partir de amostras obtidas em intervalos de tempo discretos.

  - O mapa 2D gerado na interface poderá ser salvo em um arquivo, podendo ser posteriormente carregado.

### 1.2.2 Sistema de comunicação

Esta seção descreve os requisitos do sistema de comunicação entre a estação base e o sistema embarcado.

- Distância entre robô e estação base.
  - O sistema de comunicação deve possuir, ao menos, alcance máximo de 20 metros sem fios – de modo que ambientes de tamanho razoável possam ser mapeados.
- Velocidade e direção do fluxo de transmissão de dados.
  - Toda a comunicação entre a estação base e o sistema embarcado será feita por um único canal e, portanto:
  - A velocidade de transmissão do canal de comunicação deve ser suficiente para o envio de comandos de movimentação ao robô, recebimento de dados de leituras de sensores e recebimento de imagens da câmera em tempo real;
  - O fluxo de dados deve ser bidirecional (*full-duplex*).
- Protocolo de transporte.
  - A tecnologia utilizada para a comunicação deve permitir fácil utilização do protocolo de transporte TCP. Como as leituras de sensores devem obrigatoriamente ser recebidas na estação base na mesma ordem em que forem enviadas pelo robô (e também os comandos de movimentação enviados pela estação base devem chegar ao robô em ordem), o uso desse protocolo de transporte simplificará muito a implementação do protocolo em alto nível entre os dois pontos. Outro aspecto positivo do TCP é que além de garantir a ordem de chegada, existem mecanismos de detecção de perdas de pacotes – que efetuam o reenvio caso necessário.

### 1.2.3 Sistema embarcado

Esta seção descreve os requisitos do sistema embarcado (robô).

- Movimentação do robô.

  - O robô deve ser capaz de mover-se para frente, para trás e girar para a esquerda e direita.

- Controle de posicionamento e velocidade.

  - O robô deve ser capaz de obter dados que permitam calcular sua velocidade (linear e angular) e posição atual (deslocamento e rotação em relação à posição inicial). Deve ser capaz de enviar os dados à estação base.

- Detecção de obstáculos.

  - O robô deverá ser capaz de detectar obstáculos próximos – com distância de no mínimo 30 cm e no máximo 150 cm – localizados ao seu redor, determinando a distância de cada objeto detectado.

## 1.3 ANÁLISE DE OPÇÕES TECNOLÓGICAS

Nesta seção está apresentada a análise das opções tecnológicas plausíveis para o atendimento dos requisitos. As alternativas pesquisadas e as escolhidas para cada parte do projeto estão explicitadas a seguir.

### 1.3.1 Estação base

As alternativas pesquisadas para a estação base estão apresentadas nesta subseção.

#### 1.3.1.1 Biblioteca para desenhos 2D

Tendo em vista que um dos requisitos é a geração de um mapa em duas dimensões na estação base, deve-se escolher uma biblioteca que permita realizar o desenho de formas geométricas diversas e que possa ser integrada facilmente à interface gráfica. Ela deve também possuir meios simples de obter informações do mouse e teclado, para interatividade com o usuário.

Uma biblioteca interessante disponível em Java que possui o recurso de produzir desenhos dinâmicos (e integrá-los a interfaces gráficas) é o Processing (PROCESSING, 2013), *open-source*. Essa biblioteca foi a principal encontrada que seria capaz de satisfazer as necessidades de desenho do mapa 2D de forma simples. Por possuir inúmeras funções de desenho em alto nível, o trabalho de renderização dos gráficos seria consideravelmente simplificado. Além

disso, na biblioteca existem recursos que permitem o recebimento de informações de posicionamento do mouse e de comandos do teclado. Por ser constituído basicamente de um *Applet* Java, o Processing pode facilmente ser integrado a componentes do Swing – biblioteca de interface gráfica (GUI) do Java.

Outra biblioteca para a confecção de desenhos em 2D é o Cairo (CAIRO, 2013), que é *open-source*, disponível nas linguagens C e C++. Ele possui recursos em alto nível para renderização de formas e interação com o usuário, assim como o Processing. Nos aspectos gerais as duas ferramentas não muito semelhantes. A integração do Cairo com a interface gráfica, porém, é dependente na biblioteca externa de GUI utilizada para tal.

Um aspecto importante a notar é que ambas as bibliotecas foram desenvolvidas e otimizadas para terem bom desempenho em máquinas atuais – o que é desejavel tendo em vista os requisitos. Na Tabela 1 está presente uma comparção entre as duas bibliotecas.

**Tabela 1**: Comparação entre Bibliotecas para desenhos 2D.

| **Característica** | **Cairo** | **Processing** |
|---|---|---|
| Linguagem de programação | C e C++ | Java |
| Integração com interface gráfica | Sim (depende da biblioteca de GUI utilizada) | Sim (na biblioteca Swing do Java) |
| Ferramentas de interação com o usuário | Sim | Sim |

A escolha da biblioteca de desenhos foi feita em conjunto com a escolha de linguagem de programação. A biblioteca escolhida, dentre as duas opções, foi o Processing, visto que a integração a interfaces gráficas do Java é muito simples.

#### 1.3.1.2 Linguagem de programação

Nessa etapa de avaliação das opções, a escolha de uma boa linguagem de programação que atenda aos requisitos é fundamental. Abaixo está presente uma lista dos aspectos desejáveis da linguagem:

- Deve ser multiplataforma (ao menos compatível com Linux e Windows sem muitas modificações);
- Deve possuir orientação a objetos;
- Deve possuir recursos multiplataforma e *open-source* para o desenvolvimento de interfaces gráficas;

- Deve ter a disponibilidade de ferramentas *open-source* e multiplataforma para a criação visual da interface gráfica, dessa forma agilizando o processo de desenvolvimento;
- Deve possuir recursos, integrados ou em bibliotecas *open-source*, para o desenvolvimento de desenhos dinâmicos (para a geração do mapa 2D). Os desenhos devem ser facilmente integráveis à interface gráfica.

Abaixo está presente uma descrição das duas linguagens, o C++ e Java, atualmente utilizadas em inúmeras aplicações, e que são potenciais alternativas ao projeto. A Tabela 2 sumariza os recursos de cada uma.

**Java**

O Java (JAVA, 2013) é uma linguagem concebida de início como sendo orientada a objetos. A maneira com que é feita a compilação e execução do código permite que muito facilmente programas sejam rodados em diferentes plataformas (Linux, Windows, Mac, entre outros). O processo de compilação do código gera os chamados *bytecodes*, que são instruções a serem interpretadas pela *Java Virtual Machine* (JVM). A grande vantagem é que o JVM possui disponibilidade multiplataforma, e a manutenção pelos desenvolvedores é frequente.

Há disponibilidade, na API do Java, da biblioteca Swing – que contém recursos completos para a criação de interfaces gráficas (GUI) interativas. Existem ferramentas visuais de código aberto que consideravelmente agilizam o processo de desenvolvimento de interfaces Swing, entre elas o NetBeans (NETBEANS, 2013) e o Eclipse (ECLIPSE, 2013), através de plugins ou extensões.

Para o preenchimento do requisito de confecção de desenhos em 2D com integração à interface gráfica, a biblioteca do Processing (explicada anteriormente na Subseção 1.3.1.1) está disponível nessa linguagem.

**C++**

O C++ é uma linguagem orientada a objetos, que foi desenvolvida a partir da linguagem C. A compilação de código no C++ deve ser feita especificamente para cada plataforma em que os programas desenvolvidos forem utilizados. De uma perspectiva prática, certas seções de código frequentemente necessitam de adaptações manuais para cada plataforma e sistema operacional, o que gera retrabalho e gastos de tempo adicionais.

Recursos para desenvolvimento visual de interfaces gráficas estão disponíveis através de bibliotecas e ferramentas externas. O C++ não possui recursos de interface gráfica na própria API. Deve-se notar que esse é um aspecto que adiciona complexidade ao portar programas entre

diferentes sistemas.

Para a confecção de desenhos 2D e incorporação dos mesmos à interface gráfica, a biblioteca Cairo (explicada anteriormente na Subseção 1.3.1.1) pode ser utilizada com essa linguagem. A possibilidade de haver integração com a interface, porém, é dependente da biblioteca de GUI utilizada.

**Escolha da equipe:** O Java foi a linguagem escolhida para o desenvolvimento do *software* da estação base, uma vez que preenche satisfatoriamente os requisitos do projeto. A escolha do Java foi feita em conjunto com a escolha da bilbioteca do Processing. Notavelmente, há a facilidade em portar, sem adaptações, programas para diferentes plataformas, processo este que é mais complexo no C++. Com relação ao quesito de desempenho em computadores atuais, a linguagem escolhida é satisfatória, visto que há manutenção constante da implementação das bibliotecas e da máquina virtual do Java pelos desenvolvedores – que buscam, entre outros aspectos, otimizar a linguagem para tecnologias atuais.

**Tabela 2**: Comparação entre linguagens de programação.

| **Característica** | **C++** | **Java** |
|---|---|---|
| Multiplataforma (Linux e Windows) | Sim (com adaptação) | Sim (sem adaptação) |
| Orientação a objetos | Sim | Sim |
| Recursos multi-plataforma e *open-source* para desenvolvimento de interface gráfica (GUI) | Sim (com bibliotecas externas) | Sim (integrado à API da linguagem) |
| Ferramentas *open-source* e multiplataforma para criação visual de interface gráfica | Sim (ferramentas externas) | Sim (ferramentas externas) |
| Recursos *open-source* para desenvolvimento de desenhos dinâmicos, facilmente integráveis à interface gráfica | Sim (biblitoeca externa, integração à interface gráfica dependente da GUI utilizada) | Sim (biblioteca externa) |

### 1.3.2 Sistema de comunicação

Na Tabela 3 está presente uma comparação entre diferentes tecnologias de comunicação sem fios. O Wi-Fi é o recurso mais atrativo em todos os aspectos que foram comparados, preenchendo satisfatoriamente os requisitos do sistema de comunicação. Sua velocidade e alcance são suficientes para satisfazer as necessidades, e o fluxo de dados pode ser *full-duplex*. Notavelmente, o Wi-Fi é o único sistema comparado que oferece a possibilidade (com simplicidade)

de uso do protocolo TCP – o que é um requisito importante para o desenvolvimento ágil e satisfatório do projeto.

**Tabela 3**: Comparação entre tecnologias de comunicação sem fios.

| **Característica** | **802.11g (Wi-Fi)** | **Rádio Frequência (RF)** | **Bluetooth** |
|---|---|---|---|
| Distância máxima de alcance | 50-100 metros | 30-100 metros | 10 metros |
| Velocidade de transmissão máxima | 54 Mbits/s | 2 Mbits/s | 1 Mbits/s |
| Fluxo de dados *full-duplex* | Sim | Sim | Sim |
| Possibilidade e simplicidade de uso de TCP | Sim | Não | Não |

# REFERÊNCIAS

CAIRO. 2013. Disponível em: <http://www.cairographics.org/>.

ECLIPSE. 2013. Disponível em: <http://eclipse.org/>.

JAVA. 2013. Disponível em: <www.java.com>.

NETBEANS. 2013. Disponível em: <http://netbeans.org/>.

PROCESSING. 2013. Disponível em: <www.processing.org>.