ROBOTOY: APLICAÇÃO PARA PROGRAMAÇÃO E SIMULAÇÃO DE ROBÔS

João Paulo Machado

Profa. Joyce Martins – Orientadora

1 INTRODUÇÃO

Atualmente, as pessoas vivem em uma era onde são surpreendidas no dia-a-dia pelos avanços tecnológicos e pelas facilidades proporcionadas. Nessa abordagem, o ambiente escolar é importante para a busca de um novo conhecimento. Para Azevedo, Aglaé e Pitta (2010), "a escola precisa estar em consonância com a sociedade emergente, não podendo permanecer com as mesmas funções através do tempo e do espaço".

A inclusão da robótica como ferramenta pedagógica e educacional possibilita o desenvolvimento do raciocínio lógico e criativo. A robótica educacional ou robótica pedagógica apresenta ao educando e ao educador um instrumento ideal, por ser desafiador e lúdico na solução de problemas propostos, bem como na inovação de hardware e software (MIRANDA; SAMPAIO; BORGES, 2010). Essa ideia surgiu por volta da década de 1960, pelo protagonismo do matemático americano Seymour Papert, que aplicava a teoria do construcionismo com base nos pensamentos do epistemólogo suíço Jean Piaget.

Papert (1986 apud GOMES et al., 2010) propôs o uso de computadores, robôs e programação como forma motivadora e que atraía as crianças para o âmbito educacional. Essa abordagem passa a considerar a interdisciplinaridade no currículo escolar, proporcionando o conhecimento em diversas áreas, tais como: linguagem, matemática, física, entre outras. Mas, para a inserção da robótica no ambiente educacional, se faz necessário o uso de materiais alternativos como sucata ou de kits específicos, como por exemplo, o Lego Mindstorms e o Arduino.

Com o objetivo de simplificar a programação para Lego Mindstorms NXT, Torrens (2014) desenvolveu uma linguagem e uma ferramenta de programação acessíveis para alunos das séries iniciais da educação básica, denominada Robotoy. Contudo, os kits Lego Mindstorms possuem um alto custo para aquisição. Assim, Batista (2016) adaptou a ferramenta Robotoy para permitir o uso de robôs Arduinos. O destaque do Arduino sobre outras plataformas de desenvolvimento, segundo McRoberts (2011, p. 20), "é a facilidade de sua utilização; pessoas que não são da área técnica podem, rapidamente, aprender o básico e criar seus próprios projetos em um intervalo de tempo relativamente curto". Outra proposta para viabilizar o uso da linguagem Robotoy por escolas com poucos recursos financeiros, dispensando a aquisição e o uso de um kit de robótica e, portanto, alavancando a inclusão de

mais usuários, foi desenvolvida por Silva (2016). Este implementou um simulador 2D, que permite criar e editar cenários para a resolução de exercícios.

Com base nesses argumentos, propõe-se a integração destas três ferramentas em uma única aplicação, adicionando a entrada de comando por voz, como melhoria no quesito de usabilidade, para programação e simulação dos robôs.

1.1 OBJETIVOS

O objetivo deste trabalho é disponibilizar um ambiente integrado para a programação e a simulação de robôs na linguagem de programação Robotoy.

Os objetivos específicos do trabalho são:

a) disponibilizar uma interface para a elaboração de programas na linguagem Robotoy;
b) disponibilizar uma interface para a elaboração dos cenários 2D;
c) possibilitar a simulação e a programação de robôs através de comandos por voz;
d) gerar código para o simulador, para robôs Lego Mindstorms NXT e para robôs Arduino a partir dos programas escritos em Robotoy;
e) executar os programas gerados nos cenários 2D criados, nos robôs Lego Mindstorms NXT e nos robôs Arduino.

2 TRABALHOS CORRELATOS

São apresentados três trabalhos correlatos, que possuem características semelhantes à proposta deste trabalho. A seção 2.1 detalha o RoboEduc (SILVA, 2009), software educacional para a programação de robôs desenvolvido na Universidade do Rio Grande do Norte. A seção 2.2 descreve o Webots (CYBERBOTICS, 2016), um ambiente de desenvolvimento para modelagem, simulação e programação de robôs. Por fim, a seção 2.3 aborda o RoboMind FURB (BENITTI et al., 2008), um *plugin* para o RoboMind que permite o envio dos programas a robôs Lego MindStorm NXT.

2.1 ROBOEDUC

Silva (2009) descreve o RoboEduc como um software educacional que serve como mediador em atividades de robótica educativa. A utilização do RoboEduc inicia-se com a seleção do protótipo do robô, podendo ser um dos modelos previamente cadastrados, tais como robô carregador, robô segurador, robô lixeira ou robô batedor, ou através da construção

de um novo modelo, a partir da escolha dos componentes de hardware, como atuadores, sensores e bases que deverão compor o protótipo.

Em seguida, o usuário pode determinar a forma de comandar o robô, entre controlar ou programar. Ao ser selecionada a opção para controlar o robô, inicia-se a comunicação de forma imediata entre o robô e o RoboEduc, sendo apresentados um conjunto de comandos para movimentação, em forma de ponteiros indicando as direções que o robô deve percorrer. Já a opção para programar o robô permite a elaboração de um conjunto de instruções a serem compiladas sem a necessidade de uma interação do usuário com o robô. As plataformas de robôs suportadas pelo software são: Lego Mindstorms NXT, Lego RCX e H-EDUC, protótipo de plataforma desenvolvida pelos idealizadores do RoboEduc.

Conforme afirma Silva (2009), o ambiente possibilita a criação de programas através de componentes gráficos, que representam cada ação executada pelo robô, ou pela escrita de código nas linguagens de programação RoboEduc ou C. O software possui cinco níveis de programação, sendo que os níveis 1, 2 e 4 permitem a programação através de componentes gráficos por meio do mecanismo arrasta-e-solta. O nível 1 incorpora comandos idênticos à opção para controlar o robô, sendo que cada instrução é armazenada para posterior compilação e envio ao robô. O nível 2, apresentado na Figura 1, inclui as estruturas de controle fluxo e o nível 4 funções semelhantes às do RoboLab, ambiente de programação vendido com os kits Lego Mindstorms. Já os níveis 3 e 5 englobam a programação textual, sendo que no nível 3 pode-se programar em RoboEduc e no nível 5 em C.

Figura 1 – Nível 2 para programação de robôs

Fonte: Silva (2009, p. 62).

A linguagem RoboEduc foi especificada com o intuito de diminuir a complexidade encontrada nas linguagens de programação já existente. Conforme Barros (2011, p. 21), "ao contrário de outras linguagens, esta [RoboEduc] permite a pessoa programar sem a necessidade de muitos conhecimentos prévios". Os programas desenvolvidos na linguagem RoboEduc são traduzidos para C. Posteriormente, o software envia o código C para o compilador BrickOS para que possa então ser enviado aos robôs. O Quadro 1 apresenta um programa desenvolvido na linguagem RoboEduc, onde a movimentação do robô é efetuada através da indicação do tempo que o robô deve executar determinado comando. Assim, `esquerda 1 segundos` comanda o robô para virar à esquerda durante um segundo. Comandos de repetição e seleção podem ser adicionados, contudo o código para interação com sensores deve ser escrito em C.

Quadro 1 – Programa na linguagem RoboEduc

```
tarefa RoboMexer
inicio
   para i := 0 ate 2 faca
        frente 3 segundos
        esquerda 1 segundos
   fimpara
fim
```

Fonte: elaborado pelo autor.

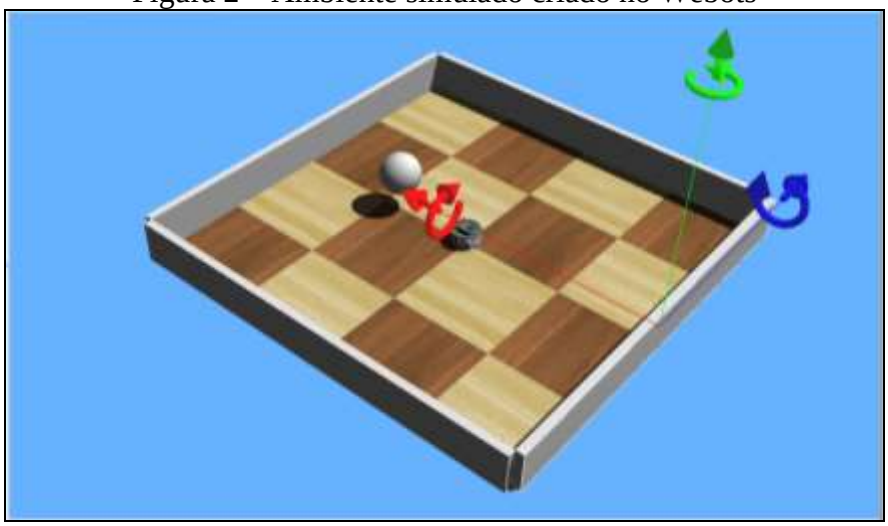
2.2 WEBOTS

O Webots é um ambiente de desenvolvimento tridimensional criado pela CyberBotics (2016) que possibilita modelagem, programação e simulação de robôs. Possui um ambiente virtual fisicamente realista, ou seja, o cenário simulado aplica propriedades da física, como por exemplo, distribuição de massa, coeficiente de atrito e inércia para torná-lo mais próximo da realidade. O ambiente de simulação permite a inclusão de objetos com diversas formas geométricas em uma estrutura de grafo de cena para interação com os robôs a serem desenvolvidos. Estes objetos podem ser classificados em: segmentos de entrada, semáforos e luzes, objetos da natureza, construções e seus elementos. Na Figura 2, pode-se observar um cenário contendo um objeto na forma de bola em uma estrutura de tabuleiro.

Denominados como controladores, os programas desenvolvidos no Webots devem ser escritos nas linguagens C, C++, Java, Python, URBI ou MATLAB$_{TM}$. Os controladores podem ser desenvolvidos no ambiente integrado ao software que, após a verificação de erros, libera a opção para envio imediato dos programas a diversas plataformas de robôs, tais como Khepera, Hemisson e Lego Mindstorms (WING, 2011). Diferente dos softwares para robótica educacional, a complexidade na elaboração dos programas torna-se alta, pois pressupõe-se que o usuário possua conhecimento na linguagem de programação a ser utilizada. No Quadro

2 é apresentado um programa em C para apresentação de uma mensagem simples. O objetivo do código mostrado é o envio da mensagem "Olá Mundo utilizando o Webots" para o console do simulador, exibindo a mensagem a cada intervalo de um segundo.

Figura 2 – Ambiente simulado criado no Webots



Fonte: CyberBotics (2016).

Quadro 2 – Programa em C desenvolvido no Webots

```
#include <webots/robot.h>
#include <stdio.h>

int main(){
   wb_robot_init();
   while(1){
      printf("Olá Mundo utilizando o Webots!\n");
      wb_robot_step(1000);
   }
   return 0;
}
```

Fonte: elaborado pelo autor.

A modelagem de robôs no Webots permite a criação de inúmeros protótipos de robôs com diferentes tipos de sensores e atuadores. Pode-se citar como exemplo o uso de sensores de distância e toque, câmeras, entre outros.

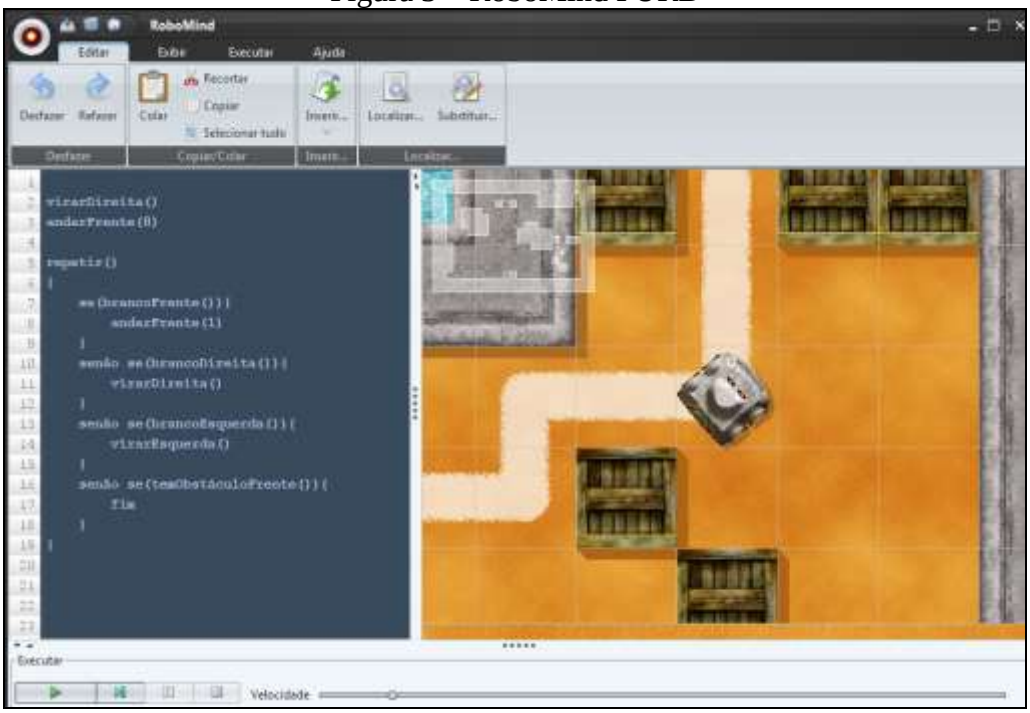
2.3 ROBOMIND FURB

Elaborado por Arvid Hama na Universidade de Amsterdam (BENITTI et al., 2008), o RoboMind é um software educacional que possui uma linguagem de programação simples, denominada Robo, para a programação e o desenvolvimento de robôs em um ambiente bidimensional. Inspirado pela linguagem Logo, Robo foi projetada com o intuito de permitir a programação de forma imediata. No que se refere à modelagem dos robôs, no RoboMind é possível adicionar sensores, como o ultrassônico para identificação de obstáculos à frente, bem como atuadores, que permitem a manipulação de objetos no cenário (ROBOMIND ACADEMY, 2014).

Já o RoboMind FURB, conforme descreve Benitti et al. (2008), é uma extensão do software original. Desenvolvido no âmbito do Departamento de Sistemas e Computação (DSC) da Universidade Regional de Blumenau (FURB), a extensão elaborada promove o envio dos códigos aos robôs para execução da tarefa. Antes disso, deve ser feita a seleção, dentre vários robôs desenvolvidos, de um modelo para realizar o experimento.

Os programas devem ser desenvolvidos no editor de texto do ambiente, que dispõe, à sua direita, de um cenário para simulação dos comandos. Na Figura 3 é apresentada a interface gráfica do RoboMind FURB.

Figura 3 – RoboMind FURB



Fonte: Benitti et al. (2008).

Os comandos a serem usados variam em função do modelo de robô selecionado, mas toda a mecânica tem como base a indicação de quantidade de casas que o robô deve percorrer. No Quadro 3 é apresentado um conjunto de instruções que podem ser utilizadas para desenhar a letra “T” no cenário. Primeiramente é ativada a função do pincel branco para colorir o chão através do comando `pintarBranco`, as próximas instruções realizam a movimentação do robô, e, por fim, a função `pararPintar` desativa o recurso de colorir o chão.

Quadro 3 – Instruções do RoboMind FURB

```
pintarBranco()
andarFrente(3)
virarDireita()
andarFrente(2)
andarTrás(4)
pararPintar()
```

Fonte: elaborado pelo autor.

3 FERRAMENTAS ATUAIS

Desenvolvido por Torrens (2014), o Robotoy é um software educacional que tem objetivo permitir que crianças elaborem programas para robôs Lego Mindstorms NXT. Com uma linguagem de programação simplificada, a ferramenta evidencia o foco na solução dos problemas propostos.

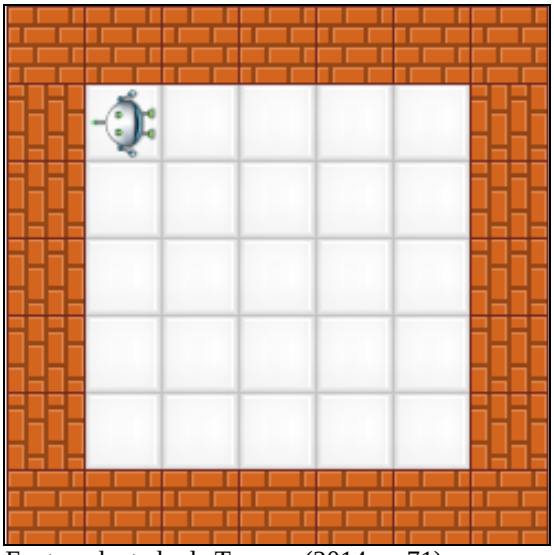
No Quadro 4, encontra-se um programa desenvolvido na linguagem Robotoy. O programa soluciona o seguinte problema: o robô deve mostrar no display a quantidade de colunas e de linhas que um cenário possui (TORRENS, 2014, p. 71). A Figura 4, traz um possível cenário, com o estado inicial do robô.

Quadro 4 – Programa em Robotoy para contagem de colunas e linhas

```
número linhas <- 1
número colunas <- 1

enquanto não tem obstáculo
   andar para frente 1
   colunas <- colunas + 1
fim do enquanto

virar para a direita 1
enquanto não tem obstáculo
   andar para frente 1
   linhas <- linhas + 1
fim do enquanto

texto qtdLinhas <- "Linhas: " . linhas
texto qtdColunas <- "Colunas: " .
colunas
escrever qtdLinhas
escrever qtdColunas
emitir som
```

Fonte: Torrens (2014, p. 71).

Figura 4 – Cenário para contagem de colunas e linhas



Fonte: adaptado de Torrens (2014, p. 71).

Para desenvolvimento da ferramenta Robotoy, Torrens (2014) definiu alguns requisitos, sendo que os principais são:

> a) disponibilizar uma linguagem de programação para que o usuário possa programar (Requisito Funcional - RF);
> b) disponibilizar um software para que o usuário possa manipular os programas definidos na linguagem (RF);
> c) disponibilizar uma camada intermediária que viabilize o suporte a várias plataformas de robôs (RF);
> [...]
> e) permitir que programas definidos na linguagem possam ser implantados em robôs LEGO Mindstorms NXT (RNF);
> f) realizar a implantação de código no robô LEGO Mindstorms NXT através da API leJOS (RNF);
> g) possuir a inteligência do robô LEGO Mindstorms NXT implementada com a API leJOS (RNF); [...] (TORRENS, 2014, p. 27).

A linguagem Robotoy foi definida para viabilizar a execução em várias plataformas de robôs, bem como permitir a montagem de qualquer tipo de robô, sendo o único componente obrigatório o *brick* para robôs Lego Mindstorms NXT. Segundo LEGO Group (2016), o kit

Lego Mindstorms NXT 2.0 possui, além do *brick*, 3 servo motores, 1 sensor ultrassônico, 2 sensores de toque, 1 sensor de cor e outras 612 peças para montagem dos robôs, conforme ilustra a Figura 5.

Figura 5 – Kit Lego Mindstorms NXT

Fonte: Plug (2016).

A partir dessa característica, Batista (2016) estendeu a ferramenta desenvolvida por Torrens (2014) para permitir "[...] que um mesmo programa possa ser executado tanto em robôs do tipo Lego Mindstorms NXT como em robôs construídos com base na placa Arduino Mega 2560 [...]" (BATISTA, 2016, p. 6). Isto é, foi adicionado suporte à programação de robôs Arduino. Nesse sentido, foram incluídas na ferramenta as seguintes funcionalidades: (a) escolha da plataforma de robô a ser utilizada (Lego Mindstorms NXT ou Arduino); (b) possibilidade de configuração do robô Arduino; (c) tradução direta dos comandos da linguagem Robotoy para a linguagem do Arduino. Dessa forma, estabeleceu-se os seguintes requisitos principais para estender a Robotoy:

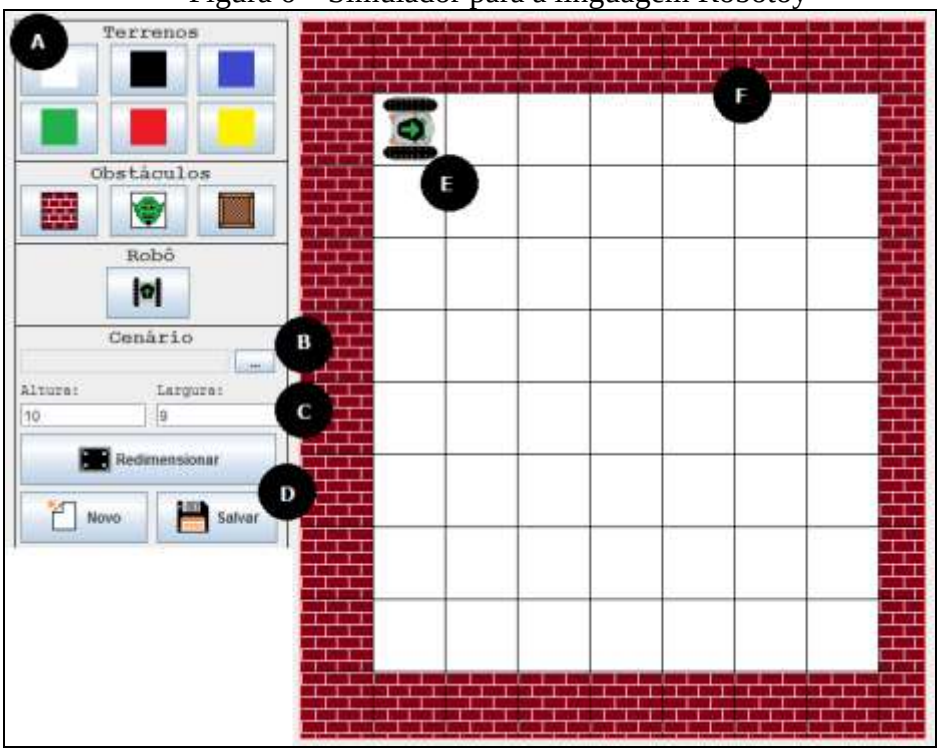
> a) gerar código para Arduino a partir dos comandos da linguagem Robotoy (Requisito Funcional - RF);
> b) disponibilizar uma tela para que o usuário escolha para qual plataforma será gerado o programa: Lego Mindstorms NXT ou Arduino (RF);
> c) disponibilizar uma tela de configuração para o robô Arduino (RF);
> d) fazer o envio do programa traduzido para o Arduino (RF);
> e) executar os programas em robôs montados com base no Arduino (Requisito Não Funcional - RNF); [...] (BATISTA, 2016, p. 24).

No entanto, independentemente da plataforma de robô usada[1], o custo ainda pode ser elevado para adoção da Robotoy como ferramenta de ensino-aprendizagem por escolas de educação básica. Assim, para viabilizar o uso da Robotoy, Silva (2016) desenvolveu um simulador 2D, facilitando a validação dos exercícios em cenários complexos. Para que isso ocorra, implementou-se um conversor para uma linguagem intermediária em formato script, simulando um interpretador de comandos (SILVA, 2016).

Na Figura 6, pode-se observar que o simulador contém um editor de cenário para a inclusão de obstáculos, alteração de terreno e redimensionamento do campo. Além disso, o cenário só pode conter apenas um robô que, após ser selecionado, pode ser rotacionado por inteiro ou apenas a sua cabeça. Foram definidos como principais requisitos do Simulador 2D:

> a) possuir uma interface para a elaboração de programas na linguagem Robotoy (Requisito Funcional - RF);
> b) possuir uma interface para criar e editar cenários 2D a fim de serem a base das simulações (RF);
> c) traduzir os programas Robotoy em um *script* que possa ser interpretado pelo simulador 2D (RF);
> d) executar os programas traduzidos no simulador 2D (RF);
> [...]
> h) utilizar a linguagem Robotoy como linguagem de alto nível para programar as ações dos robôs que serão simuladas (RNF). (SILVA, 2016, p. 23).

Figura 6 – Simulador para a linguagem Robotoy



Fonte: Silva (2016, p. 40).

[1] Batista (2016) gastou aproximadamente R$480,00 para adquirir um kit Arduino Mega 2560 (placa e demais componentes). Enquanto um kit Lego Mindstorms EV3, similar ao NXT, conforme afirma a autora, é vendido na loja virtual oficial da Lego por $349,99, ou seja, aproximadamente R$1.133,93.

4 PROPOSTA DA FERRAMENTA

Este capítulo tem como objetivo apresentar a justificativa para elaboração deste trabalho, assim como os requisitos e metodologia de desenvolvimento.

4.1 JUSTIFICATIVA

O Quadro 5 apresenta de forma comparativa as características dos trabalhos correlatos e da ferramenta proposta a ser estendida. Pode-se observar que os softwares Webots, Robotoy e RoboEduc permitem a customização de robôs, ou seja, o usuário pode modificar a estrutura dos robôs adaptando a cada cenário proposto, enquanto o RoboMind FURB utiliza modelos de robôs pré-definidos, limitando assim o número de problemas e cenários onde os robôs possam ser inseridos. Todas as ferramentas dispõem de um ambiente para simulação com o intuito de tornar desnecessária a aquisição de um kit de robótica, porém somente o ambiente de simulação do Webots permite a inclusão de múltiplos robôs, proporcionando a interação entre os robôs na busca da solução a partir da programação separada de cada um.

Quadro 5 – Comparativo entre os trabalhos correlatos e a ferramenta Robotoy

características / trabalhos	RoboEduc (SILVA, 2009)	Webots (CYBERBOTICS, 2016)	RoboMind FURB (BENITTI et al. 2008)	Robotoy (TORRENS, 2014; BATISTA, 2016; SILVA, 2016)
customização de robôs	X	X		X
múltiplos robôs		X		
plataforma de robô suportada	Lego NXT, Lego RCX, H-EDUC	Khepera, Hemisson, Lego NXT, entre outros	Lego NXT	Lego NXT, Arduino
linguagem de programação própria	X		X	X
tipo de programação	textual, gráfica	textual	textual	textual
ambiente de simulação	X	X	X	X
comando por voz				

Fonte: elaborado pelo autor.

Referente à linguagem de programação, exceto pelo Webots, os demais contam com uma linguagem própria, motivando assim os usuários que não possuem conhecimento prévio em linguagens de programação comerciais o estudo da lógica de programação. Contudo, esses ambientes que possuem uma linguagem de programação própria contêm um conjunto menor

de instruções em comparação ao Webots. Ainda no quesito linguagem de programação, apenas o RoboEduc tem-se a possibilidade de programação com o uso de componentes gráficos, incluindo um nível inicial intuitivo aos alunos no princípio do desenvolvimento da lógica de programação.

Por fim, todos incluem suporte à programação de kit Lego Mindstorms NXT, sendo que RoboEduc e Robotoy possibilitam o envio de programas a kits de baixo custo, como H-EDUC e Arduino, respectivamente. Essa alternativa permite que as instituições de ensino possam determinar o kit adequado a sua realidade financeira, além de abranger um maior público a partir dos kits de baixo custo. No entanto, nenhum deles permite programar os robôs com comando por voz, funcionalidade proposta nesse trabalho como extensão do Robotoy.

Foram apresentadas duas categorias de ambientes para desenvolvimento e programação de robôs. Os softwares RoboEduc, RoboMind FURB e Robotoy possuem a característica comum de serem ambientes educativos, no qual os alunos efetuam exercícios para o desenvolvimento do processo de ensino-aprendizagem. Por outro lado, o Webots é uma aplicação comercial para desenvolvimento de robôs comerciais, sendo necessário um ambiente fisicamente realista para modelagem, programação e simulação dos robôs.

Todavia, o ambiente educativo Robotoy passou por adaptações, porém tais transformações foram implementadas separadamente. A partir disso, propõe-se a integração em um único ambiente para programação e simulação de robôs Arduinos e Lego Mindstorms NXT. Desta forma, tem-se a intenção de proporcionar uma maior inserção do Robotoy nas instituições de ensino. Além disso, pretende-se adicionar o recurso de entrada de comando por voz, adicionando outra forma de interação e aumentando a usabilidade. No campo tecnológico, o trabalho torna-se relevante, uma vez que se propõe a elaboração de um estudo sobre as ferramentas de processamento de linguagem natural, com o propósito de utilizar a *engine* Julius encapsulada pela API LabSAPI para o reconhecimento dos comandos por voz.

4.2 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

A aplicação descrita neste trabalho deverá:

a) possuir um módulo para elaboração e compilação dos programas na linguagem Robotoy (Requisito Funcional - RF);
b) conter um módulo para criação, edição e simulação 2D dos programas elaborados (RF);
c) possibilitar a programação dos robôs através de comando por voz, tanto no editor de programas quanto no simulador (RF);

d) permitir o envio dos programas para as plataformas Lego Mindstorms NXT e Arduino (RF);
e) executar os programas nas plataformas Lego Mindstorms NXT e Arduino (Requisito Não Funcional - RNF);
f) utilizar a Application Programming Interface (API) LabSAPI juntamente com a *engine* Julius para reconhecimento de comandos por voz (RNF);
g) ser implementado na linguagem Java no ambiente de desenvolvimento Eclipse (RNF).

4.3 METODOLOGIA

O trabalho será desenvolvido observando as seguintes etapas:

a) levantamento bibliográfico: realizar levantamento bibliográfico sobre robótica educacional e processamento de linguagem natural, mais especificamente reconhecimento de voz e *engine* Julius;
b) elicitação de requisitos: detalhar e reavaliar os requisitos e, se necessário, especificar outros a partir das necessidades observadas durante a revisão bibliográfica;
c) especificação e análise: formalizar as funcionalidades da ferramenta através dos diagramas de classe e de atividades da Unified Modeling Language (UML), utilizando a ferramenta Star UML;
d) implementação da ferramenta: implementar a ferramenta proposta, utilizando a linguagem de programação Java no ambiente de desenvolvimento Eclipse. Para tanto, serão incorporadas às funcionalidades já existentes nos trabalhos de Torrens (2014), Batista (2016) e Silva (2016). Além disso, será desenvolvido o módulo de reconhecimento de voz para captação de comandos da linguagem Robotoy utilizando a *engine* Julius através da LabSAPI;
e) testes: elaborar testes para validar se os comandos da linguagem Robotoy estão sendo executados de forma correta nos robôs e no simulador, além de validar a ferramenta como um todo, passando pelas etapas de programação, simulação e execução.

As etapas serão realizadas nos períodos relacionados no Quadro 6.

Quadro 6 – Cronograma

	2017									
	fev.		mar.		abr.		maio		jun.	
etapas / quinzenas	1	2	1	2	1	2	1	2	1	2
levantamento bibliográfico										
elicitação de requisitos										
especificação e análise										
implementação da ferramenta										
testes										

Fonte: elaborado pelo autor.

5 REVISÃO BIBLIOGRÁFICA

Este capítulo está divido em duas seções. A seção 5.1 aborda uma visão geral sobre a robótica educacional, enquanto a seção 5.2 apresenta a evolução e as dificuldades relacionadas à área de reconhecimento de voz, bem como descreve as características e funcionalidades da *engine* Julius.

5.1 RÓBOTICA EDUCACIONAL

Define-se a robótica educacional como um ambiente de aprendizagem que utiliza a tecnologia como instrumento para que os participantes possam vivenciar experiências semelhantes ao cotidiano. Neste ambiente, segundo Maisonnette (2002), o aluno é levado a observar, abstrair e inventar, criando modelos a partir de peças de brinquedos ou eletrodomésticos danificados, de peças de kits Lego ou Arduino e circuitos eletrônicos.

De acordo com Quintanilha (2008), a utilização da robótica como ferramenta pedagógica é extremamente difundida em determinados países, “Holanda e a Alemanha já têm a robótica pedagógica em 100% das escolas públicas. Inglaterra, Itália, Espanha, Canadá e Estados Unidos caminham na mesma direção”. Ainda conforme afirma o autor, países da América Latina, como por exemplo, México e Peru, deviam chegar a 3 mil escolas públicas com aulas de robótica em 2008.

Neste contexto, é importante tornar o aluno um agente ativo de conhecimento, possibilitando-o construir todo o seu processo de aprendizagem. Defende-se a inclusão da robótica educacional como ferramenta desta transformação, já que o aluno tem a responsabilidade da busca do entendimento necessário para que seu projeto alcance o objetivo proposto (CASTILHO, 2009).

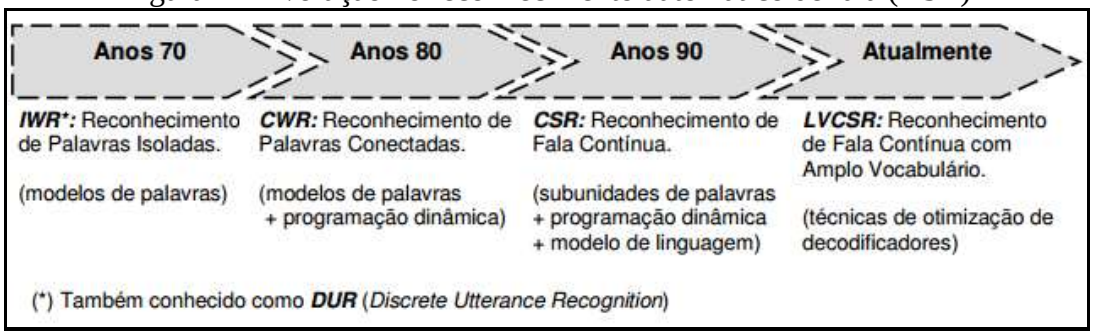
Contudo, pode-se identificar a robótica educacional como um método de aprendizagem que ultrapassa os limites da sala de aula, resolvendo assim problemas curriculares de aprendizagem contextualizada e duradoura. Além disso, a robótica educacional interfere de forma significativa na motivação dos alunos, uma vez que ensina de forma prática a aplicação

da tecnologia em problemas corriqueiros (FORNAZA; WEBBER, 2014). O ensino de robótica educacional permite a elaboração de um currículo interdisciplinar, visto que para solucionar um problema o aluno deve ser capaz de relacionar diversos conhecimentos e competências, como por exemplo, ler, articular e interpretar símbolos ou código em diferentes linguagens. Diferentemente da metodologia tradicional de programação, o desenvolvimento do raciocínio lógico torna-se dinâmico e desafiador proporcionando a criação de uma solução com o propósito de resolver problemas rotineiros. (SANTOS et al., 2010; SCHIVANI, 2014).

5.2 RECONHECIMENTO DE VOZ

A construção de sistemas para reconhecimento de voz por uma máquina tem o propósito da intepretação do sinal acústico, sendo o objeto de estudo de pesquisadores por mais de quatro décadas. A partir da década 70 do século passado, a área de reconhecimento automático de fala teve um significativo avanço, influenciado pelas áreas de processamento de sinais, algoritmos, arquitetura de software e hardware (TEVAH, 2006). A Figura 7 apresenta resumidamente os avanços das áreas nas últimas quatro décadas.

Figura 7 – Evolução no reconhecimento automático de fala (ASR)



Fonte: Tevah (2006, p. 2).

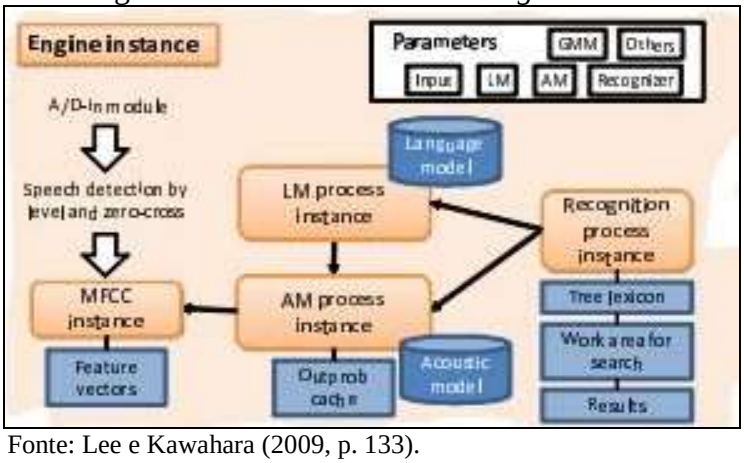
Martins (1997) afirma que as pesquisas na área de reconhecimento automático de fala têm como propósito o desenvolvimento de uma máquina capaz de transcrever a fala, em um alto nível de precisão independente de ambiente e locutor, possibilitando a comunicação entre homem e máquina de forma similar ao ser humano. Este processo tem como entrada a forma acústica produzida pelo ser humano gerando dados na qual os processadores de dados possam interpretar. Segundo Bresolin (2003), um dos aspectos mais difíceis no desenvolvimento do reconhecimento da voz pela máquina é sua interdisciplinaridade natural com várias áreas do conhecimento. Contudo, existem atualmente diversos sistemas disponíveis com alto nível de precisão para reconhecimento da fala, atendendo a necessidade de compreender diversos idiomas. Porém estes softwares são demasiadamente caros e de código fechado. Como

alternativa as essas ferramentas, pode-se citar a *engine* Julius, uma ferramenta de código aberto para reconhecimento da fala contínua (PERICO; SHINOHARA; SARMENTO, 2014).

A *engine* Julius, conforme descreve Lee (2009), é um software decodificador para reconhecimento da fala, com boa performance para grandes vocabulários. Desenvolvida na Kyoto University em 1998, está em aperfeiçoamento devido a desafios técnicos e as solicitações de melhorias dos usuários. Dispõe de versão tanto para Windows como para Linux, sendo que o reconhecimento de fala pode ocorrer tanto em tempo real através de um dispositivo como o microfone ou através de arquivos de áudio. O motor da aplicação foi escrito na linguagem C, podendo ser incorporado por outras aplicações através das manipulações dos eventos (LEE; KAWAHARA, 2009).

O funcionamento da *engine* Julius independe do idioma. Lee (2009) explica que, ao ser inicializada, a *engine* carrega o dicionário fonético, o modelo acústico e o modelo de linguagem, que são indicados através do arquivo de configuração. Após carregar esses arquivos, uma análise estatística é efetuada a partir das opções fornecidas no dicionário fonético, determinando qual a sequência de fonemas será disponibilizada para os usuários. A cada execução do decodificador são disponibilizados três resultados que representam o comando de voz em texto, o nível de confiança e o coeficiente de Vertebi, ou seja, a constante do algoritmo de cálculo de fonemas (PERICO; SHINOHARA; SARMENTO, 2014). Na Figura 8 apresenta-se os módulos internos da *engine*: entrada de áudio (`A/D in module`), detecção de voz (`Speech detection`), extração de características (`MFCC instance`), modelo de linguagem (`LM process instance`), modelo acústico (`AM process instance`) e o processo de reconhecimento de palavras (`Recognition process instance`).

Figura 8 – Estrutura interna da *engine* Julius

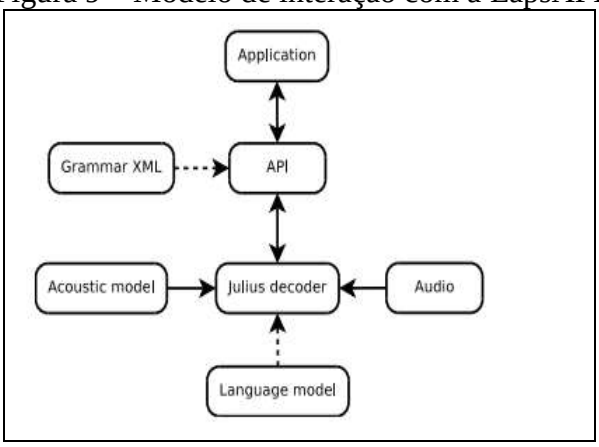


Fonte: Lee e Kawahara (2009, p. 133).

Com o intuito de simplificar o uso da *engine* Julius, o Laboratório de Processamento de Sinais (LAPS) da Universidade Federal do Pará desenvolveu uma API denominada

LapSAPI, que possibilita a manipulação em tempo real da mesma. Conforme apresenta-se na Figura 9, a API realiza uma abstração para o programador dos controles de operação em baixo nível da *engine* (KLAUTAU et al., 2012). Essa API foi desenvolvida em C++ com a especificação Common Language Runtime (CLR) para permitir a comunicação com todas as linguagens disponíveis da plataforma .NET. Do ponto de vista computacional, a API consiste em uma classe, que provê todo o controle da *engine* Julius, possibilitando que a aplicação carregue os modelos acústico e de linguagem. Além disso, conforme descreve Klautau et al. (2012), é possível carregar a gramática especificada no formato Microsoft Speech API (SAPI), que posteriormente é convertida para o formato suportado pela *engine*.

Figura 9 – Modelo de interação com a LapsAPI



Fonte: Klautau et al. (2012).

REFERÊNCIAS

AZEVEDO, Samuel; AGLAÉ, Akynara; PITTA, Renata. Minicurso: introdução a robótica educacional. In: REUNIÃO ANUAL DA SOCIEDADE BRASILEIRA PARA O PROGRESSO DA CIÊNCIA, 62., 2010, Natal. **Anais eletrônicos...** São Paulo: SBPC/UFRN, 2010. Não paginado. Disponível em: <http://www.sbpcnet.org.br/livro/62ra/minicursos/MC%20Samuel%20Azevedo.pdf>. Acesso em: 07 set. 2016.

BARROS, Renata P. **Evolução, avaliação e validação do software RoboEduc**. 2011. 92 f. Dissertação (Mestrado em Engenharia da Computação) – Programa de Pós-Graduação em Engenharia Elétrica e de Computação, Universidade Federal do Rio Grande do Norte, Natal. Disponível em: <http://repositorio.ufrn.br:8080/jspui/handle/123456789/15375>. Acesso em: 07 set. 2016.

BATISTA, Juliana C. **Robotoy:** ferramenta para ensino de programação para crianças usando robô Arduino. 2016. 64 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

BENITTI , Fabiane B. V. et al. **Bem vindo ao RoboLAB**: robótica educativa na FURB. [Blumenau?], [2008]. Disponível em: <http://robolab.inf.furb.br/>. Acesso em: 21 ago. 2016.

BRESOLIN, Adriano A. **Estudo do reconhecimento de voz para o acionamento de equipamentos elétricos via comandos em português**. 2003. 107 f. Dissertação (Mestrado em Automação Industrial) – Programa de Pós-Graduação em Automação Industrial, Universidade do Estado de Santa Catarina, Joinville. Disponível em: <http://www.tede.udesc.br/tde_arquivos/8/TDE-2006-05-19T093420Z-193/Publico/Adriano%20de%20Andrade%20Bresolin.pdf>. Acesso em: 02 nov. 2016.

CASTILHO, Maria I. **Robótica na educação:** com que objetivos? [Porto Alegre], [2009]. Disponível em: <http://www.pucrs.br/eventos/desafio/2007/mariaines.php#raclog>. Acesso em: 15 set. 2016.

CYBERBOTICS. **Webots robot simulator** [S.l.], 2016. Disponível em: <https://www.cyberbotics.com>. Acesso em: 23 ago. 2016

FORNAZA, Roseli; WEBBER, Carine G. Robótica educacional aplicada à aprendizagem em física. **Renote**, [Porto Alegre], v. 12, n. 1, p. 1-10, jul. 2014. Disponível em: <http://seer.ufrgs.br/index.php/renote/article/view/50275/31405>. Acesso em: 02 nov. 2016.

GOMES, Cristiane G. et al. A robótica como facilitadora do processo de ensino-aprendizagem de matemática no ensino fundamental. In: PIROLA, Nelson A. (Org). **Ensino de ciências e matemática IV**: temas de investigação. São Paulo: UNESP/Cultura Acadêmica, 2010. p. 205-221. Disponível em: <http://books.scielo.org/id/bpkng/pdf/pirola-9788579830815-11.pdf>. Acesso em: 28 ago. 2015.

KLAUTAU, Aldebaro et al. **Reconhecimento de voz para o português brasileiro**. [S.l.], 2012. Disponível em: <http://www.laps.ufpa.br/falabrasil/downloads.php>. Acesso em: 21 out. 2016.

LEE, Akinobu; KAWAHARA, Tatsuya. **Recent development of open-source speech recognition engine Julius**. In: ASIA-PACIFIC SIGNAL AND INFORMATION PROCESSING ASSOCIATION ANNUAL SUMMIT AND CONFERENCES, 1., 2009, Hokkaido. **Proceedings...** Hokkaido: APSIPA, 2009. p. 131-137. Disponível em: <https://julius.osdn.jp/paper/julius-APSIPA2009.pdf>. Acesso em: 15 set. 2016.

LEE, Akinobu. **The Julius book**. [S.l.], 2009. Disponível em: <http://osdn.dl.sourceforge.jp/julius/37581/Juliusbook-part-4.1.2-en.pdf>. Acesso em: 22 out. 2016.

LEGO GROUP. **Lego Mindstorms NXT 2.0**. [S.l.], 2016. Disponível em: <https://shop.lego.com/en-US/LEGO-MINDSTORMS-NXT-2-0-8547>. Acesso em: 02 nov. 2016.

MAISONNETTE, Rogers. **A utilização dos recursos informatizados a partir de uma relação inventiva com a máquina**: a robótica educativa. [Paraná], [2002]. Disponível em: <http://www.proinfo.gov.br/upload/biblioteca.cgd/192.pdf>. Acesso em: 02 nov. 2016.

MARTINS, José A. **Avaliação de diferentes técnicas para reconhecimento de fala**. 1997. 161 f. Tese (Doutorado em Engenharia Elétrica e de Computação) – Faculdade de Engenharia Elétrica e de Computação, Universidade Estadual de Campinas, Campinas. Disponível em: <http://www.bibliotecadigital.unicamp.br/document/?code=vtls000127015>. Acesso em: 15 set. 2016.

MCROBERTS, Michael. **Arduino básico**. São Paulo: Novatec, 2011.

MIRANDA, Leonardo C.; SAMPAIO, Fábio F.; BORGES, José A. S. RoboFácil: especificação e implementação de um kit de robótica para a realidade educacional brasileira. **Revista Brasileira de Informática na Educação**, [Porto Alegre?], v. 18, n. 3, p. 46-58, 2010. Disponível em: <http://www.br-ie.org/pub/index.php/rbie/article/view/1275>. Acesso em: 07 set. 2016.

PERICO, Alisson; SHINOHARA, Cindi S.; SARMENTO, Cristiano D. **Sistema de reconhecimento de voz para automatização de uma plataforma elevatória**. 2014. 96 f. Trabalho de Conclusão de Curso (Engenharia Industrial Elétrica - Ênfase em Automação) – Departamento Acadêmico de Eletrotécnica, Universidade Tecnológica Federal do Paraná, Curitiba. Disponível em: <http://nupet.daelt.ct.utfpr.edu.br/tcc/engenharia/doc-equipe/2012_2_15/2012_2_15_monografia.pdf>. Acesso em: 02 nov. 2016.

PLUG. **Como arrumar as peças?** [S.l.], 2016. Disponível em: <http://www.plug.pt/forum/index.php?topic=61.270>. Acesso em: 02 nov. 2016.

QUINTANILHA, Leandro. Irresistível robô. **ARede**, São Paulo, n. 34, não paginado, mar. 2008. Disponível em: <www.revista.arede.inf.br/site/edicao-n-34-marco-2008/3920-irresistivel-robo>. Acesso em: 23 out. 2016.

ROBOMIND ACADEMY. **Introduction:** what is Robo/RoboMind? [S.l.], 2014. Disponível em: <http://www.robomind.net/en/introduction.htm>. Acesso em: 02 set. 2016.

SANTOS, Franklin L. et al. REDUC: a robótica educacional como abordagem de baixo custo para o ensino de computação em cursos técnicos e tecnológicos. In: WORKSHOP DE INFORMÁTICA NA ESCOLA, 16., 2010, Santos. **Anais eletrônicos...** [S.l.]: SBC, 2010. P. 1304-1313. Disponível em: <http://www.br-ie.org/pub/index.php/wie/article/view/2053>. Acesso em: 05 nov. 2016.

SCHIVANI, Milton. **Contextualização no ensino de física à luz da teoria antropológica do didático: o caso da robótica educacional**. 2014. 220 f. Tese (Doutorado em Educação) – Faculdade de Educação, Universidade de São Paulo, São Paulo. Disponível em: <http://www.teses.usp.br/teses/disponiveis/48/48134/tde-01122014-104322/>. Acesso em: 05 nov. 2016.

SILVA, Alzira F. **RoboEduc**: uma metodologia de aprendizado com robótica educacional. 2009. 135 f. Tese (Doutorado em Engenharia da Computação) – Faculdade de Engenharia Elétrica e de Computação, Universidade Federal do Rio Grande do Norte, Natal. Disponível em: <https://repositorio.ufrn.br/jspui/handle/123456789/15128>. Acesso em: 07 set. 2016.

SILVA, Diogo. **Um simulador 2D para a linguagem Robotoy**. 2016. 54 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

TEVAH, Rafael T. **Implementação de um sistema de reconhecimento de fala contínua com amplo vocabulário para o português brasileiro**. 2006. 91 f. Dissertação (Mestrado em Engenharia Elétrica) – Programa de Pós-Graduação de Engenharia, Universidade Federal do Rio de Janeiro, Rio de Janeiro. Disponível em: <http://www.eletrica.ufrj.br/teses/textocompleto/2006053001.pdf >. Acesso em: 02 nov. 2016.

TORRENS, Maria G**. Robotoy:** ferramenta para uso de robótica no ensino de programação para crianças. 2014. 71 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

WING, Rowan. **Intro to controllers**. [Boulder?], 2011. Disponível em: <http://correll.cs.colorado.edu/?p=852>. Acesso em: 07 set. 2016.

ASSINATURAS

(Atenção: todas as folhas devem estar rubricadas)

Assinatura do(a) Aluno(a): _____

Assinatura do(a) Orientador(a): _____

Assinatura do(a) Coorientador(a) (se houver): _____

Observações do orientador em relação a itens não atendidos do pré-projeto (se houver):

FORMULÁRIO DE AVALIAÇÃO (PROJETO) – PROFESSOR TCC I

Acadêmico(a): _____

Avaliador(a): _____

	ASPECTOS AVALIADOS[1]	atende	atende parcialmente	não atende
ASPECTOS TÉCNICOS	1. INTRODUÇÃO O tema de pesquisa está devidamente contextualizado/delimitado?			
	O problema está claramente formulado?			
	2. OBJETIVOS O objetivo principal está claramente definido e é passível de ser alcançado?			
	Os objetivos específicos são coerentes com o objetivo principal?			
	3. TRABALHOS CORRELATOS São apresentados trabalhos correlatos, bem como descritas as principais funcionalidades e os pontos fortes e fracos?			
	4. JUSTIFICATIVA Foi apresentado e discutido um quadro relacionando os trabalhos correlatos e suas principais funcionalidades com a proposta apresentada?			
	São apresentados argumentos científicos, técnicos ou metodológicos que justificam a proposta?			
	São apresentadas as contribuições teóricas, práticas ou sociais que justificam a proposta?			
	5. REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO Os requisitos funcionais e não funcionais foram claramente descritos?			
	6. METODOLOGIA Foram relacionadas todas as etapas necessárias para o desenvolvimento do TCC?			
	Os métodos, recursos e o cronograma estão devidamente apresentados e são compatíveis com a metodologia proposta?			
	7. REVISÃO BIBLIOGRÁFICA Os assuntos apresentados são suficientes e têm relação com o tema do TCC?			
	As referências contemplam adequadamente os assuntos abordados (são indicadas obras atualizadas e as mais importantes da área)?			
ASPECTOS METODOLÓGICOS	8. LINGUAGEM USADA (redação) O texto completo é coerente e redigido corretamente em língua portuguesa, usando linguagem formal/científica?			
	A exposição do assunto é ordenada (as ideias estão bem encadeadas e a linguagem utilizada é clara)?			
	9. ORGANIZAÇÃO E APRESENTAÇÃO GRÁFICA DO TEXTO A organização e apresentação dos capítulos, seções, subseções e parágrafos estão de acordo com o modelo estabelecido?			
	10. ILUSTRAÇÕES (figuras, quadros, tabelas) As ilustrações são legíveis e obedecem às normas da ABNT?			
	11. REFERÊNCIAS E CITAÇÕES As referências obedecem às normas da ABNT?			
	As citações obedecem às normas da ABNT?			
	Todos os documentos citados foram referenciados e vice-versa, isto é, as citações e referências são consistentes?			

PARECER – PROFESSOR DE TCC I OU COORDENADOR DE TCC:

O projeto de TCC será reprovado se:

- qualquer um dos itens tiver resposta NÃO ATENDE;
- pelo menos **4 (quatro)** itens dos **ASPECTOS TÉCNICOS** tiverem resposta ATENDE PARCIALMENTE; ou
- pelo menos **4 (quatro)** itens dos **ASPECTOS METODOLÓGICOS** tiverem resposta ATENDE PARCIALMENTE.

PARECER: ( ) APROVADO ( ) REPROVADO

Assinatura: _____ Data: _____

[1] Quando o avaliador marcar algum item como atende parcialmente ou não atende, deve obrigatoriamente indicar os motivos no texto, para que o aluno saiba o porquê da avaliação.

FORMULÁRIO DE AVALIAÇÃO (PROJETO) – PROFESSOR AVALIADOR

Acadêmico(a): _____

Avaliador(a): _____

	ASPECTOS AVALIADOS[1]	atende	atende parcialmente	não atende
ASPECTOS TÉCNICOS	1. INTRODUÇÃO O tema de pesquisa está devidamente contextualizado/delimitado?			
	O problema está claramente formulado?			
	2. OBJETIVOS O objetivo principal está claramente definido e é passível de ser alcançado?			
	Os objetivos específicos são coerentes com o objetivo principal?			
	3. TRABALHOS CORRELATOS São apresentados trabalhos correlatos, bem como descritas as principais funcionalidades e os pontos fortes e fracos?			
	4. JUSTIFICATIVA Foi apresentado e discutido um quadro relacionando os trabalhos correlatos e suas principais funcionalidades com a proposta apresentada?			
	São apresentados argumentos científicos, técnicos ou metodológicos que justificam a proposta?			
	São apresentadas as contribuições teóricas, práticas ou sociais que justificam a proposta?			
	5. REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO Os requisitos funcionais e não funcionais foram claramente descritos?			
	6. METODOLOGIA Foram relacionadas todas as etapas necessárias para o desenvolvimento do TCC?			
	Os métodos, recursos e o cronograma estão devidamente apresentados e são compatíveis com a metodologia proposta?			
	7. REVISÃO BIBLIOGRÁFICA Os assuntos apresentados são suficientes e têm relação com o tema do TCC?			
	As referências contemplam adequadamente os assuntos abordados (são indicadas obras atualizadas e as mais importantes da área)?			
ASPECTOS METODOLÓGICOS	8. LINGUAGEM USADA (redação) O texto completo é coerente e redigido corretamente em língua portuguesa, usando linguagem formal/científica?			
	A exposição do assunto é ordenada (as ideias estão bem encadeadas e a linguagem utilizada é clara)?			

PARECER – PROFESSOR AVALIADOR:

O projeto de TCC serdeverá ser revisado, isto é, necessita de complementação, se:

- qualquer um dos itens tiver resposta NÃO ATENDE;
- pelo menos **5 (cinco)** tiverem resposta ATENDE PARCIALMENTE.

PARECER: ( ) APROVADO ( ) REPROVADO

Assinatura: _____ Data: _____

[1] Quando o avaliador marcar algum item como atende parcialmente ou não atende, deve obrigatoriamente indicar os motivos no texto, para que o aluno saiba o porquê da avaliação.