UNIVERSIDADE DE ARARAQUARA – UNIARA

Programa de Pós-graduação em Processos de Ensino, Gestão e Inovação

Mauro de Lucca

A CONSTRUÇÃO DE UM MANUAL DIDÁTICO: A ROBÓTICA PEDAGÓGICA COMO FERRAMENTA PARA A APRENDIZAGEM DE LÓGICA DE PROGRAMAÇÃO PARA ALUNOS DO ENSINO MÉDIO PROFISSIONALIZANTE

ARARAQUARA - SP
2018

Mauro de Lucca

A CONSTRUÇÃO DE UM MANUAL DIDÁTICO: A ROBÓTICA PEDAGÓGICA COMO FERRAMENTA PARA A APRENDIZAGEM DE LÓGICA DE PROGRAMAÇÃO PARA ALUNOS DO ENSINO MÉDIO PROFISSIONALIZANTE

Dissertação apresentada ao Programa de Pós-graduação em Processos de Ensino, Gestão e Inovação da Universidade de Araraquara – UNIARA – como parte dos requisitos para obtenção do título de Mestre em Processos de Ensino, Gestão e Inovação.

Linha de pesquisa: Processos de Ensino

Orientador: Prof. Dr. Fábio Tadeu Reina

ARARAQUARA - SP
2018

FICHA CATALOGRÁFICA

L967c de Lucca, Mauro
A construção de um manual didático: a Robótica Pedagógica como ferramenta para a aprendizagem de lógica de programação para alunos do Ensino Médio Profissionalizante /Mauro de Lucca
Araraquara: Universidade de Araraquara – UNIARA
2018.
80f

Dissertação (Mestrado) – Programa de Pós-graduação em Processos de Ensino, Gestão e Inovação da Universidade de Araraquara

Orientador: Prof. Dr. Fábio Tadeu Reina

1. Palavra-chave. 2. Palavra-chave. 3. Palavra-chave.
4. Palavra-chave. 5. Palavra-chave.

CDU 370

REFERÊNCIA BIBLIOGRÁFICA

DE LUCCA, M. **A construção de um manual didático:** a Robótica Pedagógica como ferramenta para a aprendizagem de lógica de programação para alunos do Ensino Médio Profissionalizante. 2018. 80 páginas. Dissertação do Programa de Pós-graduação em Processos de Ensino, Gestão e Inovação da Universidade de Araraquara – UNIARA, Araraquara-SP.

ATESTADO DE AUTORIA E CESSÃO DE DIREITOS

NOME DO AUTOR: Mauro de Lucca
TÍTULO DO TRABALHO: A construção de um manual didático: a Robótica Pedagógica como ferramenta para a aprendizagem de lógica de programação para alunos do Ensino Médio Profissionalizante
TIPO DO TRABALHO/ANO: Dissertação / 2018

Mauro de Lucca
E-mail: mauro.lucca@ifsp.edu.br

UNIVERSIDADE DE ARARAQUARA - UNIARA
PROGRAMA DE PÓS-GRADUAÇÃO EM PROCESSOS DE ENSINO, GESTÃO E INOVAÇÃO, ÁREA DE EDUCAÇÃO

FOLHA DE APROVAÇÃO

Dissertação apresentada ao Programa de Pós-graduação em Processos de Ensino, Gestão e Inovação da Universidade de Araraquara – UNIARA – para obtenção do título de **Mestre em Processos de Ensino, Gestão e Inovação.**

Área de Concentração: Educação e Ciências Sociais.

NOME DO AUTOR: **MAURO DE LUCCA**

TÍTULO DO TRABALHO: **: "A CONSTRUÇÃO DE UM MANUAL DIDÁTICO: A ROBÓTICA PEDAGÓGICA COMO FERRAMENTA PARA A APRENDIZAGEM DE LÓGICA DE PROGRAMAÇÃO PARA ALUNOS DO ENSINO MÉDIO PROFISSIONALIZANTE".**

Assinatura dos Examinadores: Conceito:

Prof. Dr. Fabio Tadeu Reina (orientador) (X) Aprovado ( ) Reprovado
Universidade de Araraquara – UNIARA

Profa. Dra. Dirce Charara Monteiro (X) Aprovado ( ) Reprovado
Universidade de Araraquara – UNIARA

Prof. Dr. José Henrique Mazon (X) Aprovado ( ) Reprovado
Universidade Paulista - UNIP

Versão definitiva revisada pelo orientador em: 25/05/18

Prof. Dr. Fábio Tadeu Reina (orientador)

À Juliana, ao Felipe e ao Guilherme, vocês são toda a força de que eu preciso.

AGRADECIMENTOS

Aos meus pais, Elizabeth e Marcos, pelos valores que me passaram e pela fé que sempre tiveram em mim.

Aos meus irmãos, Marcelo e Marcos, que sempre me incentivaram e aconselharam.

Ao meu orientador, Prof. Dr. Fábio Tadeu Reina pelas contribuições valorosas neste novo mundo que se abre.

Ao casal de amigos, Cíntia e Marcel, que disponibilizaram tempo, ajuda e ouvidos quando precisei.

A todos os amigos e familiares, que de forma direta ou indireta contribuíram para meu crescimento pessoal.

A todos os alunos e colegas do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo – *campus* Araraquara, que foram importantes demais no meu caminho da docência.

RESUMO

Atualmente, muitos recursos digitais são utilizados como ferramentas no auxílio dos processos de ensino e aprendizagem. As Tecnologias de Informação e Comunicação (TICs) são cada vez mais utilizadas por professores que buscam novos meios de tornar suas aulas mais interessantes aos alunos e novas formas de ensinar. Mas o uso de tecnologias na educação não pode ser somente pensado como forma de modernizar a aula, mas sim como ferramentas que podem melhorar o rendimento e o desempenho escolar. Ao se pensar nos Cursos Técnicos Integrados ao Ensino Médio, podemos imaginar que o uso de tecnologias é algo corriqueiro e trivial. Mas, índices levantados pelo Ministério da Educação mostram que a formação de alunos no ensino fundamental ainda está abaixo do que se espera. Muitos alunos trazem dificuldades em interpretação de textos e matemática. Isso acarreta em dificuldades no ensino e aprendizagem de conhecimentos técnicos fundamentais, especialmente nesses cursos, ocasionando grande número de retenções e evasão nessas instituições. Pensando nessas dificuldades, o presente trabalho tem o objetivo de apresentar um manual que utilize a robótica pedagógica como meio de mitigar as dificuldades com a abstração de conceitos fundamentais aos processos de ensino e aprendizagem da programação de computadores e sua lógica. Por meio da revisão na literatura, foi possível traçar uma estratégia de construção deste manual com o ideal de guiar o professor na execução de um projeto de robótica pedagógica, na própria disciplina ou como atividade extracurricular. O manual contém propostas de trabalho e exercícios que convergem com os temas centrais da disciplina de Algoritmos e Programação do Curso Técnico em Informática Integrado ao Ensino Médio do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo – IFSP – *campus* Araraquara. Espera-se que este manual possibilite aos docentes e discentes a utilização da robótica pedagógica como meio de construção de um novo aprendizado que apoie o aprendizado da lógica de programação, em torno dessas situações-problema apresentadas.

Palavras-chaves: Robótica pedagógica; Lógica de Programação; Produto Educacional.

ABSTRACT

Currently, many digital resources are used as tools to assist teaching and learning processes. Information and Communication Technologies (ICTs) are increasingly used by teachers who are seeking new ways to make their classes more interesting to students and new ways to teach. But the use of technologies in education can not only be thought of as a way to modernize the class, but rather as tools that can improve the income and school performance. When thinking about the Technical Courses Integrated to High School, we can imagine that the use of technologies is commonplace and trivial. But indexes raised by the Ministry of Education show that the education of students in elementary school is still below expectations. Many students bring difficulties in interpretation of text and mathematics. This leads to difficulties in the teaching and learning of fundamental technical knowledge, especially in these courses, resulting in a large number of deductions and evasion in these institutions. Thinking about these difficulties, the present study aims to present a manual that uses pedagogical robotics as a means to mitigate difficulties with the abstraction of fundamental concepts to the teaching and learning processes of computer programming and its logic. Through the literature review, it was possible to trace a strategy for constructing this manual with the ideal of guiding the teacher in the execution of a pedagogical robotics project, either in the discipline itself or as an extracurricular activity. The manual contains work proposals and exercises that converge with the central themes of the Algorithms and Programming Course of the Technical Course in Integrated Computer Science at the Federal Institute of Education, Science and Technology of São Paulo - IFSP - Araraquara campus. It is hoped that this manual allows teachers and students to use educational Robotics as a means of building a new learning that supports the learning of programming logic, around these problem situations presented.

Keywords: Educational robotics; Programming logic; Educational Product.

ÍNDICE DE FIGURAS

ÍNDICE DE QUADROS

SUMÁRIO

INTRODUÇÃO

O uso de equipamentos, ferramentas e materiais para auxiliar a educação é quase tão antigo quanto a própria educação. Podemos considerar desde o uso de uma vara para escrever na areia, utilizado pelos Jesuítas, passando por lousas e giz, livros, mimeógrafos, retroprojetores até o uso de computadores, internet e lousa digital como tecnologias na educação. Pois, o que são as tecnologias senão as ferramentas, técnicas e aplicações utilizadas na solução de um problema? A robótica pedagógica é mais uma tecnologia que visa a facilitação nos processos de ensino e aprendizagem dos educandos e pode ser aplicada em qualquer disciplina. O encantamento que a construção de robôs tem sobre as pessoas é um grande aliado dos processos educativos.

Neste momento, apresento minha trajetória desde a formação como Tecnólogo em Processamento de Dados até a sala de aula, como Professor do Ensino Básico, Técnico e Tecnológico do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo. Além da descrição dessa trajetória, mostro em seguida a centelha motivadora desta pesquisa. Finalizo este capítulo com o objetivo deste trabalho e sua organização.

A respeito da trajetória profissional, concluí minha graduação no final do ano 2000, pela Faculdade de Tecnologia de Taquaritinga – FATEC – no curso de Tecnologia em Processamento de Dados. Ano após ano, galguei degraus de forma a me tornar um Analista Desenvolvedor de Sistemas. No final do ano de 2011, prestei um concurso, despretensiosamente, para ingressar no quadro docente do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP), para o qual fui nomeado e empossado em fevereiro de 2014 para vaga no campus Araraquara. Após o ingresso como docente dessa instituição, minha vida mudou totalmente. Fiz uma pós-graduação lato sensu em Formação de Docentes para o Ensino Superior e passei a me interessar pelos assuntos inerentes à prática docente. No mesmo ano de 2014, assumi os cargos de coordenação da área de informática e coordenação do curso técnico em informática, nas modalidades concomitante, quando o aluno cursa o ensino médio em outra instituição de ensino, e subsequente, quando o aluno já concluiu o ensino médio. Além disso, comecei a trabalhar numa comissão que formularia o Projeto Pedagógico de Curso (PPC) do Curso Técnico em Informática na modalidade Integrado ao Ensino Médio.

Nesse desafio pude participar da construção e elaboração de um curso a partir do zero. Dessa maneira, pude entender o significado e a importância da formação integral do ser humano, da interdisciplinaridade, multidisciplinaridade e, inclusive, sugerir práticas que pudessem contribuir para esses objetivos do PPC.

No ano de 2016, teve início no *campus* Araraquara do IFSP a primeira turma do Curso Técnico em Informática Integrado ao Ensino Médio e fui professor da disciplina de Algoritmos e Programação para metade desses alunos. Daí começam observações a respeito das dificuldades de abstração e também da interpretação de enunciados. A maioria dos alunos provem do ensino fundamental em escolas públicas municipais ou estaduais e traz deficiências em suas formações iniciais. Essas dificuldades podem ser observadas, além do olhar do professor em sala de aula, em levantamentos feitos e divulgados pelo próprio Ministério da Educação (MEC), através do Ideb – Índice de Desenvolvimento da Educação Básica – que é obtido a partir de cálculos de resultados de desempenho como Prova Brasil, Censo Escolar e Saeb (Sistema de Avaliação da Educação Básica).

Tive um contato inicial com a Robótica Pedagógica por meio do texto Estudos sobre dispositivos robóticos na educação: sobre a exploração do fascínio humano pela robótica no ensino-aprendizagem (MILL; CÉSAR, 2013), como aluno especial numa disciplina do Programa de Pós-graduação em Ciência, Tecnologia e Sociedade, na Universidade Federal de São Carlos – UFSCar – ministrada pelo próprio professor Daniel Mill, um dos autores do texto. Até então, a Robótica Pedagógica me parecia algo distante e só foi despertar meu interesse num curso de extensão realizado no Instituto de Ciências Matemática e de Computação da Universidade de São Paulo – ICMC/USP – intitulado: “O uso das tecnologias na aprendizagem: Excel, Word, PowerPoint e Robótica”. Nesse curso tive contato com um kit de Robótica Pedagógica fabricado na cidade de São Carlos e comecei a procurar outras alternativas também.

Nesse período, fiquei intrigado com a crescente demanda das escolas particulares quanto à Robótica Pedagógica e comecei a me perguntar se não haveria alguma maneira de levar essa metodologia às escolas públicas. Dessa maneira, verificando as dificuldades dos alunos com os enunciados e com a abstração dos conceitos de lógica de programação, comecei a pensar sobre a introdução de um projeto que aliasse o conteúdo dessa disciplina a um projeto de Robótica Pedagógica de forma a trazer significado aos conceitos introduzidos nessa disciplina.

O objetivo do trabalho foi a criação de um produto pedagógico, um manual na forma de sequência didática que possa direcionar e dar o aporte teórico necessário para a execução de um projeto de robótica pedagógica para a disciplina de Algoritmos e Programação, que pode ser trabalhado internamente à disciplina ou como projeto extracurricular, visando dirimir as dificuldades de alunos quanto à abstração de conceitos e à interpretação de enunciados, especialmente nos conceitos da Lógica de Programação. Sua construção foi calcada nos conceitos fundamentais para o aprendizado destes conhecimentos e a elaboração foi pensada baseada em propostas de trabalhos e exercícios, com dificuldade crescente.

Este manual é destinado aos professores de lógica de programação e tem a finalidade de auxiliá-los a trabalhar em conjunto com os alunos a construção do pensamento abstrato, necessário à programação de computadores, por meio da aplicação concreta encontrada na robótica pedagógica. Espera-se que sua utilização ajude na melhora no desempenho escolar como um todo, mas, mais especificamente nas áreas de lógica matemática e de programação.

Esse trabalho apresenta uma revisão da literatura, onde encontramos o aporte teórico necessário às questões centrais desse tema. Na metodologia, estão inseridos os materiais utilizados e a maneira que a construção do manual foi pensada. Ao término do trabalho, teremos o resultado que é o manual construído *per se*, e as considerações finais, as quais ficaram destacadas algumas questões acerca do manual e sua construção.

1. REVISÃO DE LITERATURA

Neste capítulo apresentaremos o aporte teórico utilizado na construção do manual. Foram leituras importantes para sua elaboração e que procuraram elucidar os conceitos utilizados e aprendidos desde as tecnologias na educação e robótica e textos que influenciaram e motivaram o trabalho. Descreveremos, também, os Institutos Federais e o *campus* Araraquara e sobre Seymour Papert, considerado o pai da robótica pedagógica e grande entusiasta e incentivador do uso de tecnologias na educação.

Este início traz a importância do uso das tecnologias na educação para os processos de ensino e aprendizagem. Frisa-se que normalmente são confundidas com as tecnologias digitais, mas seu uso, como veremos, é tão antigo quanto o próprio ato de ensinar.

Se nos ampararmos em suas definições, apenas, veremos que o uso de tecnologias acompanha a docência desde o início: a tecnologia é sempre vista como a aplicação de conhecimentos científicos (ou a aplicação da ciência), o que nem sempre é verdade. Desde muito tempo utilizamos a tecnologia, mesmo antes da ciência. A tecnologia sempre foi utilizada e criada por conhecimentos empíricos e da tentativa e erro, sempre impulsionada pelas necessidades humanas. Assim, não pode ser considerada meramente como uma aplicação de leis científicas (RIBEIRO; OLIVEIRA; MILL, 2013). Sua utilização na educação vem muito antes das tecnologias digitais ou das tecnologias de informação e comunicação (TICs), tão em voga com o advento da informática e internet e sua popularização nas escolas. Seriam, dessa forma, giz e lousa, livros impressos, carteiras e cadeiras, materiais de apoio tecnologias voltadas à educação.

Para Ribeiro, Oliveira e Mill (2013, pg. 146):

> No entanto, se antes o domínio de tecnologias complexas pelos professores, tais como as TIC, era facultativa, hoje se tornou mandatório. Embora possamos lastimar sua natureza invasiva e questionar sua necessidade real, não podemos negar a presença da tecnologia nas mais variadas atividades humanas e sua importância como mediadora da comunicação entre as pessoas na contemporaneidade. Nessa direção, as TIC são um elemento importante na educação no mundo atual, já que a educação é essencialmente um processo de comunicação – não só de teorias, conceitos e habilidades, mas também de atitudes profissional e socialmente desejáveis.

Desde o final dos anos de 1990 e início dos anos 2000, com o aumento da disponibilidade de computadores, *internet*, *tablets*, *smartphones* e sua consequente redução de custos, essas TICs vêm sendo utilizadas nas escolas como ferramentas no auxílio dos processos de ensino e aprendizagem. Junto a esse fenômeno, faz-se necessária a inclusão digital das pessoas que não tinham uma vivência naturalizada ou nativa com toda essa tecnologia digital e trabalhos específicos na formação de professores para o uso de tais tecnologias nos processos de ensino e aprendizagem.

Nesse ínterim, surgem em 2008 os Institutos Federais de Educação, Ciência e Tecnologia, estruturados sobre as instalações dos Centros Federais de Educação Tecnológica e outras escolas técnicas e agrotécnicas federais, distribuídos em 38 instituições *multicampi*, espalhadas por todo o Brasil (PACHECO, 2011).

Estes Institutos Federais atuam desde a Educação Básica até os níveis superiores de educação e tem por finalidade, dentre outras coisas, de acordo com sua lei de criação de nº 11.892/2008:

> I - Ofertar educação profissional e tecnológica, em todos os seus níveis e modalidades, formando e qualificando cidadãos com vistas na atuação profissional nos diversos setores da economia, com ênfase no desenvolvimento socioeconômico local, regional e nacional;
> II - Desenvolver a educação profissional e tecnológica como processo educativo e investigativo de geração e adaptação de soluções técnicas e tecnológicas às demandas sociais e peculiaridades regionais; (BRASIL, 2008)

Em 2010, com a expansão dos Institutos Federais por todo o Brasil, é inaugurado o *campus* Araraquara do IFSP. Esse campus possui três eixos de conhecimento: Informática, Indústria e Matemática. Além desses três eixos formativos, há ainda um eixo de formação geral destinado a ministrar as disciplinas propedêuticas aos cursos em que forem necessárias. Assim, hoje existem cursos superiores de Bacharelado em Engenharia Mecânica, Licenciatura em Matemática e Tecnologia em Análise e Desenvolvimento de Sistemas. Há cursos técnicos, integrados ao Ensino Médio, em Mecânica e Informática, que são o foco do presente trabalho. Tem, ainda, cursos técnicos concomitantes e subsequentes em Mecatrônica e Informática.

É importante falar um pouco sobre a construção dos PPCs dos Cursos Técnicos Integrados ao Ensino Médio. Esses cursos foram pensados e discutidos em medidas que pudessem levar ao estudante do ensino médio conhecimentos que fossem ensinados e

aprendidos de forma integrada, como são no mundo real. Assim como vivemos numa realidade difusa, onde os conceitos se misturam em informações e conhecimentos das mais diversas áreas. De tal maneira seria esse Curso Técnico Integrado: o conhecimento técnico não se desprende de forma alguma das disciplinas propedêuticas. Assim, português, matemática, sociologia e todas as demais disciplinas não se desprendem umas das outras e nem das disciplinas de formação técnica e devem se integrar durante todo o processo formativo do aluno, fortalecendo e dando significado real ao que ele aprende.

Sendo assim, mostra-se necessário conhecermos o objetivo do Curso Técnico em Informática Integrado ao Ensino Médio, conforme nos traz o seu PPC (IFSP, 2015): "O Curso Técnico em Informática Integrado ao Ensino Médio tem como objetivo geral, associando a base nacional comum com o ensino tecnológico, proporcionar aos alunos formação profissional aliada a uma cultura geral".

A robótica pedagógica aparece como um meio educacional que proporciona a interdisciplinaridade e até mesmo a multidisciplinaridade. Essa tecnologia mostra-se como uma ferramenta essencialmente competente a auxiliar nesse objetivo de formação do curso, da forma como foi proposto.

O manual pretende ser uma ferramenta de auxílio ao professor dos anos iniciais do curso como uma ferramenta mediadora na construção de conhecimentos da programação de computadores e sua lógica e poderá auxiliar no conhecimento de outras disciplinas indiretamente.

Assim, é essencial, a partir de agora, adentrarmos e aprofundarmos os conhecimentos e termos em torno da robótica, separando o que for de interesse para a robótica pedagógica.

Iniciaremos esta etapa falando como a ficção científica impulsionou a ciência. Karel Capek e Isaac Asimov são exemplos de como a tecnologia, em especial a robótica, sempre fascinou a humanidade e são autores que impulsionaram a ciência por meio de sua ficção.

A palavra robô tem sua origem na palavra *robota,* foi inventada pelo escritor tcheco Karel Capek, e traduz a ideia de trabalho forçado por homens-máquinas em sua obra de ficção de 1920 (GIRALT, 1997). Os robôs, como autômatos, foram popularizados por meio dos livros e contos do escritor de ficção científica e bioquímico Isaac Asimov. Asimov criou em seus livros as três leis da robótica, que seriam os princípios que regeriam a robótica:

> 1ª Lei: Um robô não pode ferir um ser humano ou, por inação, permitir que um ser humano sofra algum mal.
>
> 2ª Lei: Um robô deve obedecer as ordens que lhe sejam dadas por seres humanos exceto nos casos em que tais ordens entrem em conflito com a Primeira Lei.
>
> 3ª Lei: Um robô deve proteger sua própria existência desde que tal proteção não entre em conflito com a Primeira ou Segunda Leis. (ASIMOV, 2014)

Saindo do campo ficcional e partindo para o campo científico, buscamos o conceito de Matarič (2014, p. 21), que afirma que "um robô é um sistema autônomo que existe no mundo físico, pode sentir o seu ambiente e pode agir sobre ele para alcançar alguns objetivos". Assim sendo, não podemos descartar para nosso estudo as formas como o robô interage com o ambiente e como ele toma suas decisões para a solução de problemas propostos. Dessa maneira, podemos concluir que a robótica, por definição, é o estudo dos robôs e tudo que os envolve.

Para falar sobre a robótica pedagógica, precisamos passar pelo professor Seymour Papert e suas contribuições à educação, especialmente sobre o uso de tecnologias na educação.

Papert foi bacharel em Filosofia, *Ph.D* em Matemática e trabalhou com Jean Piaget de1958 a 1963, em Genebra, na perspectiva do uso da matemática para o entendimento da aprendizagem de crianças. Em 1964, ingressou no MIT (*Massachusetts Institute of Technology*) onde trabalhou no laboratório de Inteligência Artificial iniciando, juntamente com Marvin Minsky, diversos programas de pesquisa envolvendo teoria da computação, robótica, percepção humana e psicologia da criança. Durante a década de 60, desenvolveu com outros pesquisadores a linguagem LOGO – uma linguagem de programação voltada para crianças, no apoio aos processos de ensino e aprendizagem. Por meio do estudo de autores como Dewey, Piaget, Montessori e Paulo Freire, Papert desenvolveu a teoria do construcionismo (CAMPOS, 2013).

O contrucionismo, segundo Papert (1994, pg.127), é sua reconstrução pessoal do contrutivismo de Piaget. No construcionismo, além das construções mentais feitas pelo indivíduo no processo de aprendizagem, as construções materiais apoiam e sedimentam o conhecimento adquirido.

Sendo assim, temos o apoio que precisamos no contrucionismo, e por consequência na robótica pedagógica, para as construções mentais necessárias à abstração que sentimos

falta nos processos de ensino e aprendizagem da lógica de programação e programação de computadores.

A robótica pedagógica se apoia justamente nesse mecanismo de construção de conhecimento, de acordo com Mill e César (2013, p.272):

> [...] consideramos que robótica pedagógica é uma denominação para o conjunto de processos e procedimentos envolvidos em propostas de ensino e aprendizagem que tomam os dispositivos robóticos como tecnologia de mediação para a construção do conhecimento. Dessa forma, quando nos referimos à robótica pedagógica, não estamos falando da tecnologia dos artefatos robóticos em si nem do ambiente físico onde as atividades são desenvolvidas. Não estamos nos referindo a outra coisa senão à proposta de possibilidades metodológicas de uso das tecnologias informáticas e robóticas no processo de ensino e aprendizagem.

Aqui vemos que não só importa a aprendizagem inerente ao funcionamento dos robôs, mas sim, todo o conjunto dos processos e procedimentos envolvidos.

Sendo assim, D'Abreu (1999) indica que ambientes de aprendizagens baseados no uso de dispositivos robóticos, tem possibilitado de forma barata, rápida e segura a disponibilização do uso de recursos tecnológicos, não só para o aprendizado da robótica em si, mas de ciências em geral. O que nos é importante quando falamos da construção interdisciplinar de conhecimentos.

No desenvolvimento das propostas do manual, indico o trabalho em grupo em todas as propostas. Pois, dessa maneira podemos pensar numa forma de construção de conhecimento pelos próprios alunos, com a mediação do professor. Para Curcio (2010) as tecnologias como a robótica educacional podem ser utilizadas como um recurso pedagógico para o aluno realizar a construção do seu próprio conhecimento, por meio de investigações e simulações, ampliando-se as possibilidades de aprendizagem. E para complementar essa informação, em Silva *et al*.(2008, p. 2), vemos que:

> O trabalho com a robótica no âmbito educacional estimula a curiosidade, empolgação, concentração, orgulho e prazer na realização das atividades, possuindo metodologias específicas, as quais possibilitam o relacionamento com conteúdos curriculares, como os de matemática, artes, física, ciências, dentre outros

Dessa maneira, após dois anos lecionando disciplinas voltadas ao ensino de lógica de programação de computadores, entendo que a robótica pedagógica pode ser usada como apoio *construcionista* às construções dos conhecimentos envolvidos.

Baseado no conteúdo programático da disciplina de Algoritmos e Programação, ministrada aos alunos do Curso Técnico em Informática Integrado ao Ensino Médio, foi construído um manual pedagógico para apoiar os professores das disciplinas iniciais de programação de computadores.

Esse manual poderá ser utilizado em qualquer modalidade de ensino e poderá ser adaptado a outras disciplinas, pois a robótica pedagógica pode ser utilizada em qualquer disciplina ou área intelectual e interage com uma diversidade de situações (DOS SANTOS; POZZEBON; FRIGO, 2013), inclusive criando um ambiente onde é possível trabalhar a transdisciplinaridade ou interdisciplinaridade, de acordo com Mill e César (2013):

> Projetos com robótica pedagógica geram situações de aprendizagem pela resolução de problemas inter ou transdisciplinares, que podem ser mais simples ou mais complexos, dependendo do nível de ensino em que forem aplicados. Dessa forma, são criadas oportunidades de construção do conhecimento pelas situações geradas pelo projeto de robótica pedagógica[...].

A seguir, na metodologia, mostro como foi a elaboração desse manual e os dispositivos utilizados para essa criação e o plano de ensino da disciplina de Algoritmos e Programação que norteou e indicou os conceitos que foram utilizados.

2. METODOLOGIA

Veremos agora alguns fatores relevantes à elaboração do manual e qual o percurso de construção a que ele foi submetido, bem como suas motivações.

O resultado deste trabalho pode ser entendido como um produto educacional de uma pesquisa qualitativa. Qualitativa, pois não há um dado concreto ou estatístico que possa ser mensurado, mas uma hipótese de melhoria nos processos de ensino e aprendizagem da lógica de programação e programação de computadores baseada num levantamento literário fundamentado no campo da robótica pedagógica.

2.1.A Construção de um Manual de Robótica Pedagógica

O manual foi feito na forma de sequência didática visando a orientação e execução de um projeto de robótica pedagógica como um material didático de apoio ao professor durante esse processo. O mesmo, também poderá ser utilizado como um material didático dentro da própria disciplina de Algoritmos e Programação.

Sua confecção partiu de uma sequência de decisões e motivações. O primeiro passo foi definir qual o tipo de robô que teríamos para o desenvolvimento do trabalho. Optou-se por um robô móvel na forma de um carrinho de três rodas: duas tracionadas por motores elétricos e um rodízio independente conhecido como roda boba. Nesse robô móvel foram utilizados sensores simples para que o aprendizado de seu funcionamento não se tornasse um problema e seria utilizada a placa de prototipação Arduino, escolhida pelo baixo custo e pela facilidade de programá-la. Além disso, sua linguagem é baseada nas linguagens C/C++, que são as mesmas utilizadas na disciplina em questão, sendo necessários somente alguns ajustes.

Tomadas essas decisões, foram escolhidos quais seriam os conteúdos trabalhados no manual. Escolhemos as estruturas principais de um programa de computador, bem como os conceitos fundamentais para o seu funcionamento. Seriam as estruturas sequencial, condicional e de repetição. Trabalharia, ainda, com declaração de variáveis e constantes e funções. Optamos por não trabalhar os conceitos de vetores e matrizes, mas se os conteúdos trabalhados forem bem assimilados, os mais avançados, certamente, serão mais facilmente aprendidos.

É importante ressaltar que o manual indica algumas propostas para o trabalho destes conteúdos, mas deixa em aberto para o professor sentir o trabalho da turma e desenvolver outras propostas de trabalho junto aos alunos no decorrer das aulas.

Após tomadas as decisões para a elaboração do manual, partimos para a construção do robô para que pudesse programá-lo e testar o funcionamento dos programas e do robô. O robô móvel sobre alguns eixos e passamos a desenvolver as propostas de trabalho e programá-lo. Após cada programa, eram feitos testes para verificarmos o funcionamento.

No próximo subcapítulo, serão descritos em detalhes a construção do *kit* robótico e falaremos sobre as opções dos sensores escolhidos e as motivações para escolhê-los.

2.1.1. Kit de Robótica e Linguagem para o Arduino

Para a construção deste manual foi utilizado um *kit* de robótica de baixo custo e fácil acesso. Foram utilizados uma placa de prototipação Arduino Uno, sensores e motores para possibilitarem desde a construção do robô até a proposição de situações-problema para atingir os objetivos propostos no manual.

O Arduino é uma placa de prototipação eletrônica de arquitetura aberta, podendo ser construída gratuitamente em casa ou comprada em lojas de eletrônicos, e pode sentir e controlar o ambiente de forma autônoma, através de sua programação e sensores, ou de forma auxiliar a um computador. É muito utilizada por designers, arquitetos, artistas ou entusiastas da computação, robótica ou eletrônica. Sua programação é feita por uma linguagem própria baseada nas linguagens C e C++, e o ambiente de programação pode ser baixado gratuitamente em seu site (ARDUINO, 2017).

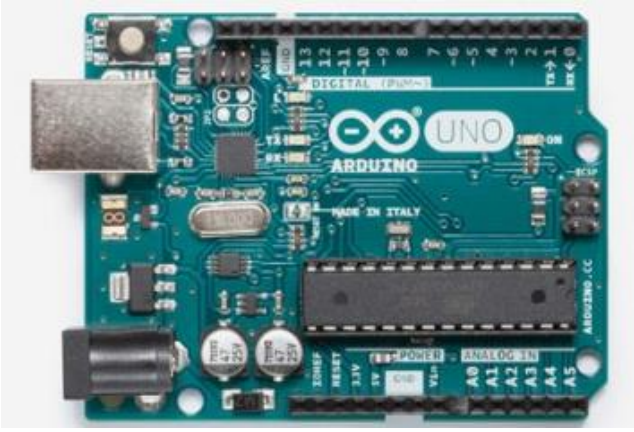
A linguagem C é amplamente utilizada para o aprendizado de lógica de programação, como é o caso da nossa disciplina, restando ao professor apenas alguns ajustes para a adequação à linguagem nativa do Arduino.

A construção do manual foi feita com base na construção do *kit* robótico pretendido, utilizando-se os equipamentos descritos com mais detalhes, a seguir:

O *kit* é formado por um chassi para robô móvel com tração em duas rodas, composto por dois motores DC, duas rodas de borracha, chapa de acrílico usinada, rodízio

universal (roda boba), suporte de bateria 9V com chave, Arduino Uno, Arduino Motor *Shield* L293, dois sensores infravermelhos de obstáculo, e três módulos com sensores ópticos de contraste (seguidores de linha), uma mini *protoboard* de 170 pontos e um *buzzer* (alto-falante) ativo. A escolha desse *kit* robótico foi feita com dois objetivos em mente: facilidade na montagem e custo-benefício.

Figura 1: Arduino UNO



Nessa imagem podemos ver as entradas/saídas que conectam o Arduino UNO ao mundo real: numeradas de 0 a 13, na parte superior e A0 a A5 na parte inferior. O Arduino Motor *Shield*, mostrado abaixo, é acoplado nessas entradas e disponibiliza automaticamente os controles dos motores (peças azuis, acima e abaixo), entrada de corrente de alimentação (peça azul sozinha na lateral) e portas A0 a A5, terra e alimentação +5V ao lado dos controles de motores, no lado inferior. Para utilização dessas portas, há a necessidade de soldar pinos conectores com estanho – o que não requer muita prática, porém pode ser feito com auxílio de um técnico de laboratórios de eletrônica.

Figura 2: Arduino Motor Shield

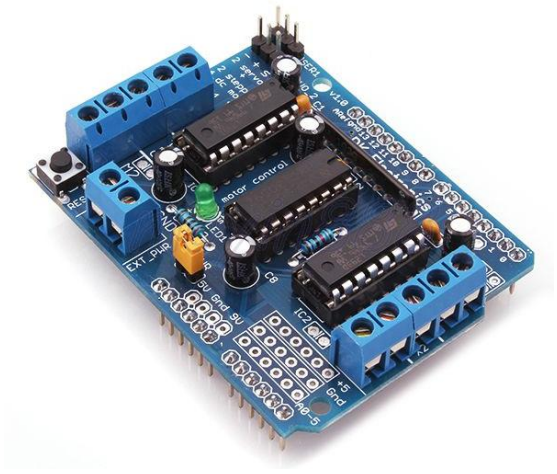


Figura 3: Chassi, motores e rodas

Figura 4: Módulo de sensor óptico de contraste

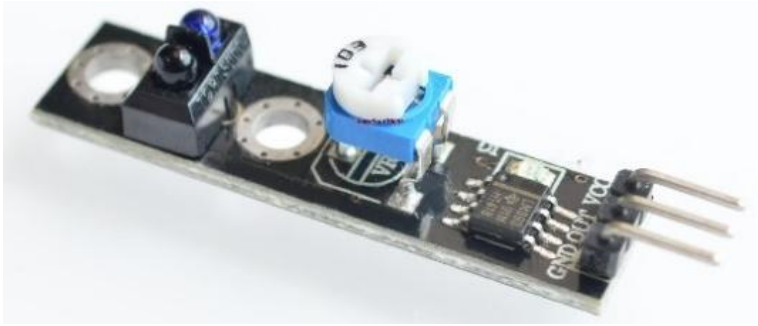


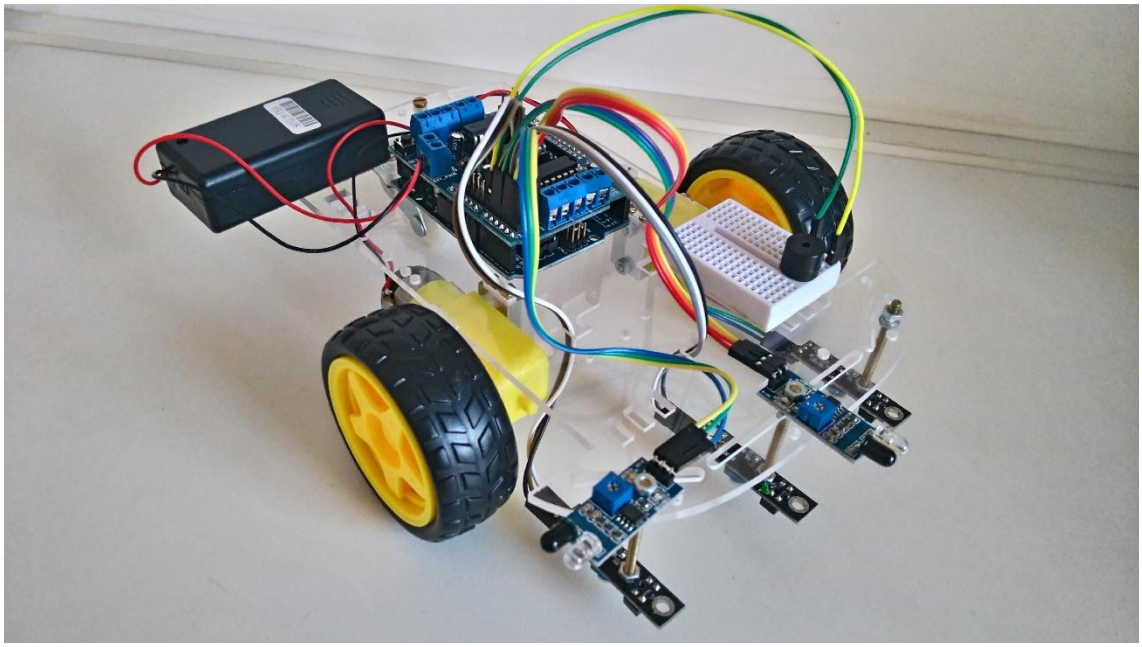
Figura 5: Módulo de sensor infravermelho

Os motores, sensores, a mini *protoboard* e o Arduino UNO são afixados no chassi de acrílico com parafusos e porcas. O *buzzer* foi afixado na mini *protoboard*. O Arduino Motor *Shield* é um módulo que vai acoplado ao Arduino UNO, com a finalidade de alimentar e controlar os motores, além de alimentar o Arduino UNO, o *buzzer* e os sensores através do suporte de bateria 9V ligado a ele. A alimentação e comunicação dos sensores e do *buzzer* são feitas através de fios condutores ligados às portas A0 até A5, conexões de 5V e terra, no Arduino Motor *Shield*.

O modo correto de montagem, no nosso caso, seria: fixamos os motores no chassi através de eixos que vem no kit e, após isso, fixamos as rodas de borracha nos motores, de maneira que fiquem na lateral, mais próximas à frente do chassi. O rodízio universal é fixado atrás com parafusos. O chassi montado com as rodas e rodízio devem ficar semelhantes à figura abaixo, exceto pelo módulo de pilhas, que não utilizaremos:

Figura 6: Chassi do robô móvel montado

Figura 7: Robô móvel já montado com todas as peças propostas no manual



Feita a apresentação do *kit* robótico utilizado na construção do manual e as explicações detalhadas dos componentes utilizados, passamos em seguida a apresentação da disciplina de Algoritmos e Programação, ministrada aos alunos do primeiro ano do Curso Técnico em Informática Integrado ao Ensino Médio do IFSP – *campus* Araraquara.

2.1.2. A disciplina de Algoritmos e Programação

Apresento nesse tópico o plano de ensino da disciplina de Algoritmos e Programação, que aborda os conceitos fundamentais para a programação de computadores e, por consequência, do *kit* robótico. Segue, abaixo, a transcrição da ementa da referida disciplina:

> A disciplina aborda a construção de algoritmos em português estruturado e em linguagem de programação estruturada, como a Linguagem C. Trabalha com o desenvolvimento de algoritmos usando estrutura sequencial, estrutura condicional e estruturas de repetição. Abrange ainda variáveis compostas homogêneas uni e bidimensionais e testes de mesa. (IFSP, 2015)

Essa disciplina possui quatro aulas semanais e tem a função de fazer o contato inicial dos alunos, em sua grande maioria ao menos, com a lógica de programação.

Abaixo, segue o plano de ensino da disciplina, conforme o Projeto Pedagógico de Curso (IFSP, 2015):

Quadro 1: Plano de ensino da disciplina de Algoritmos e Programação, ementa.
Fonte: IFSP Araraquara

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA SÃO PAULO	***CAMPUS*** *Araraquara*

1- IDENTIFICAÇÃO		
Curso: Técnico em Informática Integrado ao Ensino Médio		
Componente curricular: Algoritmos e Programação		
1º ano	**Código: ALP**	
Nº de aulas semanais: 04	**Total de aulas: 160**	**Total de horas: 133**
Abordagem Metodológica: T ( ) P (X) T/P ( )	**Uso de laboratório ou outros ambientes além da sala de aula?** (X) SIM ( ) NÃO Qual(is)? Laboratório de Informática	
2 - EMENTA: A disciplina aborda a construção de algoritmos em português estruturado e em linguagem de programação estruturada, como a Linguagem C. Trabalha com o desenvolvimento de algoritmos usando estrutura sequencial, estrutura condicional e estruturas de repetição. Abrange ainda variáveis compostas homogêneas uni e bi-dimensionais e testes de mesa.		

Quadro 2: Plano de ensino da disciplina de Algoritmos e Programação, objetivo e conteúdo
Fonte: IFSP Araraquara

3-OBJETIVOS:
Interpretar e desenvolver algoritmos e programas seguindo o paradigma de linguagem estruturada utilizando a Linguagem C.

4-CONTEÚDO PROGRAMÁTICO:

1º Bimestre
- Introdução à lógica de programação
- Construção de algoritmos em português estruturado
- Estrutura sequencial

2º Bimestre
- Estrutura condicional
- Estruturas de repetição.
- Variáveis compostas homogêneas

3º Bimestre
- Unidimensionais
- Bidimensionais

4º Bimestre
- Testes de mesa.
- Linguagem de programação estruturada

Quadro 3: Plano de ensino da disciplina de Algoritmos e Programação, bibliografia básica
Fonte: IFSP Araraquara

5- BIBLIOGRAFIA BÁSICA:

ASCENCIO, A. F. G.; CAMPOS, E. A. V. **Fundamentos da Programação de Computadores:** Algoritmos, Pascal, C/C++ e Java. 2. ed. São Paulo: Pearson Prentice Hall, 2007. 434 P.

DEITEL, Paul; DEITEL, Harvey. **C: como programar. Traduzido do original: C: how to program**; Tradução: Daniel Vieira; Revisão técnica: César Augusto Cardoso Caetano. 6a ed.. São Paulo: Pearson, 2011. 818 p.

FARRER, H.; BECKER, C. G. et. al. **Algoritmos Estruturados**. 3. ed. Rio de Janeiro: LTC, 2010.

MIZRAHI, Victorine Viviane. **Treinamento em linguagem C.** 2a ed. São Paulo: Pearson Prentice Hall, 2008. 405 p.

Quadro 4: Plano de ensino da disciplina de Algoritmos e Programação, bibliografia complementar
Fonte: IFSP Araraquara

6-BIBLIOGRAFIA COMPLEMENTAR:

FARRER, Harry et al. **Pascal estruturado**. 3. ed. Rio de Janeiro: LTC, 1999. 279 p.

GUIMARÃES, A. M.; LAGES, N. A. C. **Algoritmos e Estruturas de Dados**. São Paulo: LTC, 2011. 216 p.

LOPES, Anita; GARCIA, Guto. **Introdução à programação:** 500 algoritmos resolvidos**.** Rio de Janeiro: Elsevier, 2002.

MANZANO, J. A. N. G. **Algoritmos:** lógica para desenvolvimento de programação**.** São Paulo: Erica, 1996.

MANZANO, José Augusto N. G.; YAMATUMI, Wilson Yoshiteru. **Free pascal: programação de computadores:** guia básico de orientação e desenvolvimento para programação em Linux e MS-Windows. 2. ed. São Paulo: Érica, 2010.

SALVETTI, D. D.; BARBOSA, L. M. **Algoritmos**. São Paulo: Makron Books, 1998.

SCHILDT, Herbert. **C completo e total. Traduzido do original:** C the complete reference; Tradução e revisão técnica: Roberto Carlos Mayer. 3. ed. rev. e atualizada. São Paulo: Pearson, 1997. 827 p.

Os conceitos abordados nos dois primeiros bimestres são essenciais ao desenvolvimento da lógica de programação e são os conceitos nos quais os alunos mais apresentam dificuldades de aprendizado. São eles: a construção de algoritmos e seu funcionamento, estrutura sequencial, estrutura condicional, estruturas de repetição e, ainda, definição e declaração de variáveis e tabela-verdade.

No início, abordamos a construção de algoritmos e a forma lógica da estruturação de comandos simples para a resolução de problemas. São sempre utilizados exemplos práticos cotidianos como: trocar uma lâmpada, trocar um pneu, fritar um ovo, tomar banho, etc. Esses exemplos são utilizados para conceituar essa estruturação de tarefas.

Após essa abordagem inicial, passamos aos programas de computadores e mostramos Estrutura Sequencial, que nada mais é que a utilização dos comandos de uma linguagem de computador que executadas em sequência trazem resultados à resolução de um problema proposto.

Em seguida, adicionamos variáveis a este problema, sempre abordando a temática inicial da programação: Entrada → Processamento → Saída. Busca-se na matemática a conceituação de variáveis e passamos aos tipos básicos de dados como números reais, números inteiros, caracteres e textos.

Os passos seguintes são os que apresentam os maiores problemas de conceituação, especialmente se os conceitos anteriores não foram completamente compreendidos. Na

Estrutura Sequencial, os comandos seguem uma ordem predeterminada até o final da execução do algoritmo e se tem contato, quase exclusivamente, com operadores aritméticos – Soma, Subtração, Multiplicação e Divisão, além dos comandos de entrada e saída de informações. Já nas Estruturas Condicionais e de Repetição, há de se ter a compreensão de desvios condicionais ou a repetição de comandos para a automatização de tarefas. Para tanto, é absolutamente necessário que se tenha um real aprendizado dos conceitos de variáveis e tabela-verdade, pois essas estruturas dependem de operadores lógicos – E, OU e NÃO – e Lógica Booleana – Verdadeiro e Falso. (ASCENCIO; CAMPOS, 2012)

Todos esses conceitos podem ser abordados de forma simples e rápida utilizando-se equipamento robótico, no caso o *kit* sugerido. Dessa forma, os conceitos saem do abstrato e são executados de forma concreta, voltando ao abstrato após a execução: “Penso, faço, testo e refaço”.

No próximo capítulo, mostro o resultado dessa pesquisa que é o manual. Este manual alinha todos os conceitos levantados e traz as propostas de trabalho com que se pretende executar e responder, futuramente, as ideias levantadas até o momento.

3. RESULTADOS

Após o levantamento de uma inquietação, que foi a observação das dificuldades lógico-matemáticas e de abstração por parte dos alunos, uma busca por respostas na literatura e a ligação entre teoria e prática com a robótica, temos como resultado desse trabalho um manual de robótica pedagógica destinado à transposição dos conceitos fundamentais da lógica de programação e da programação de computadores à programação de um robô móvel.

No próximo item, podemos contemplar um manual que abrange desde a apresentação da linguagem de programação do Arduino, passando pelas principais estruturas de um programa de computador, até o trabalho com funções. Cada tópico do manual foi destinado a um tema específico da disciplina de Algoritmos e Programação e apresenta, normalmente, três propostas de trabalho e mais algumas propostas de exercícios. Sempre caminhando da forma mais simples até a mais complexa.

Vale ressaltar que as propostas foram pensadas como temas de trabalho, mas o professor pode e deve trabalhar sempre como mediador dos grupos de alunos, avançando e discutindo novas formas de trabalho, com novas situações-problema propostas entre o grupo.

3.1. O Manual de Robótica Pedagógica

Segue o manual de robótica pedagógica para apoio à disciplina de Algoritmos e Programação ou disciplinas afins de um curso técnico de nível médio.

Este manual traz propostas de trabalho com um robô móvel para apoiar a construção de algoritmos em um ambiente de construção concreta para conceitos abstratos como estruturas de repetição e decisão, variáveis de ambiente, constantes, lógica *booleana* e funções.

Espera-se que com o uso do manual os alunos possam programar um robô móvel de maneira que o mesmo possa se mover, tomar decisões e executar ações de forma autônoma, e que esses mesmos alunos possam compreender tais conceitos nos programas de computador.

3.1.1.Introdução ao Ambiente Arduino

Objetivo

Mostrar aos discentes o Ambiente de Desenvolvimento Integrado para Arduino (Arduino IDE) e dar os primeiros passos na programação do dispositivo.

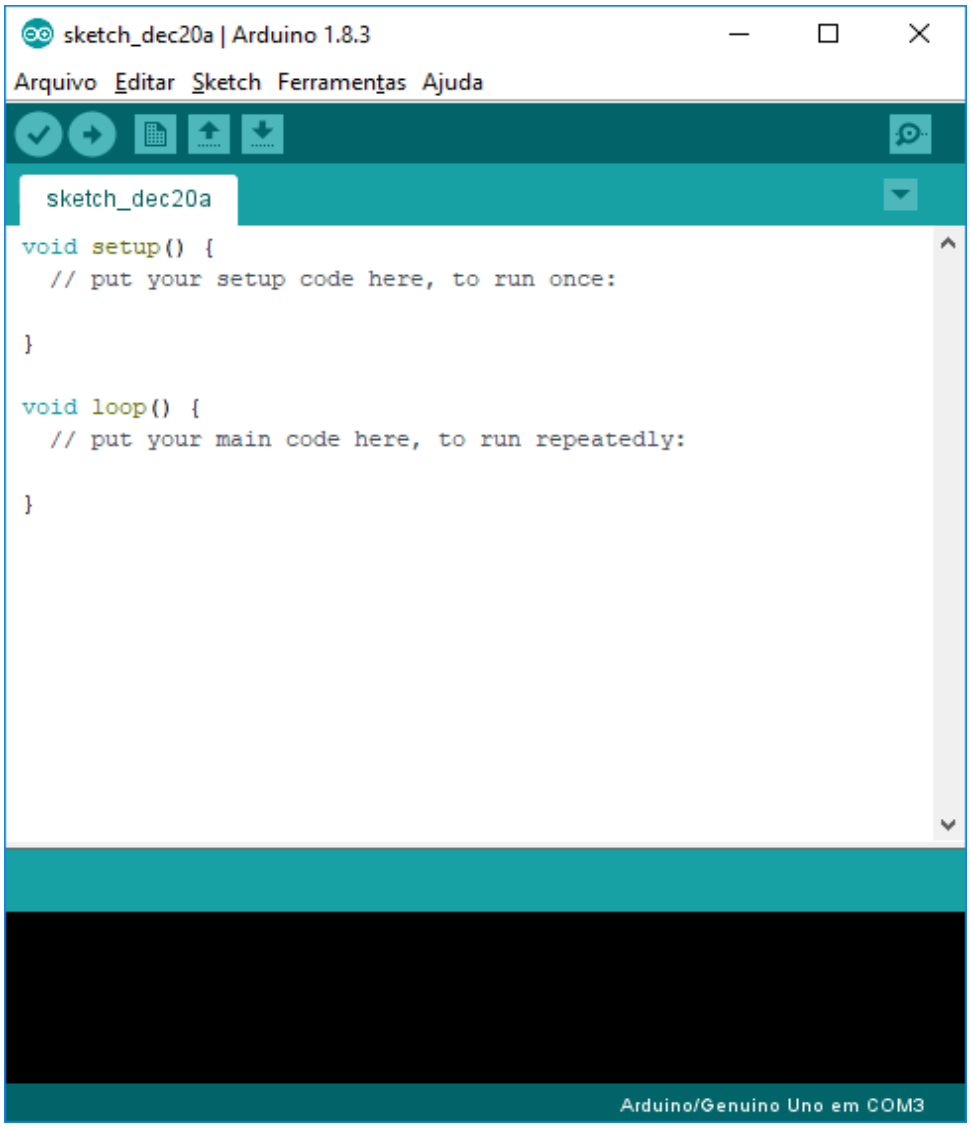
Recursos

Utilizar computador conectado em projetor multimídia.

Desenvolvimento

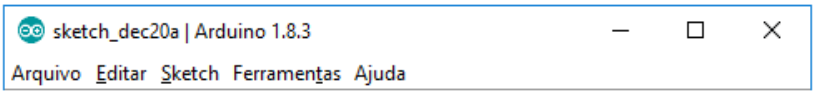
Projetar o ambiente de desenvolvimento integrado para Arduino. A linguagem é baseada em C/C++. A versão utilizada neste manual é a 1.8.3:

Figura 8: Janela do Arduino IDE



Vale ressaltar que o Arduino IDE é dividido em três partes: A barra de menus, onde você abrirá seus programas, configurará a porta de comunicação correta do Arduino e todas as outras ferramentas:

Figura 9: Barra de menus - Arduino IDE

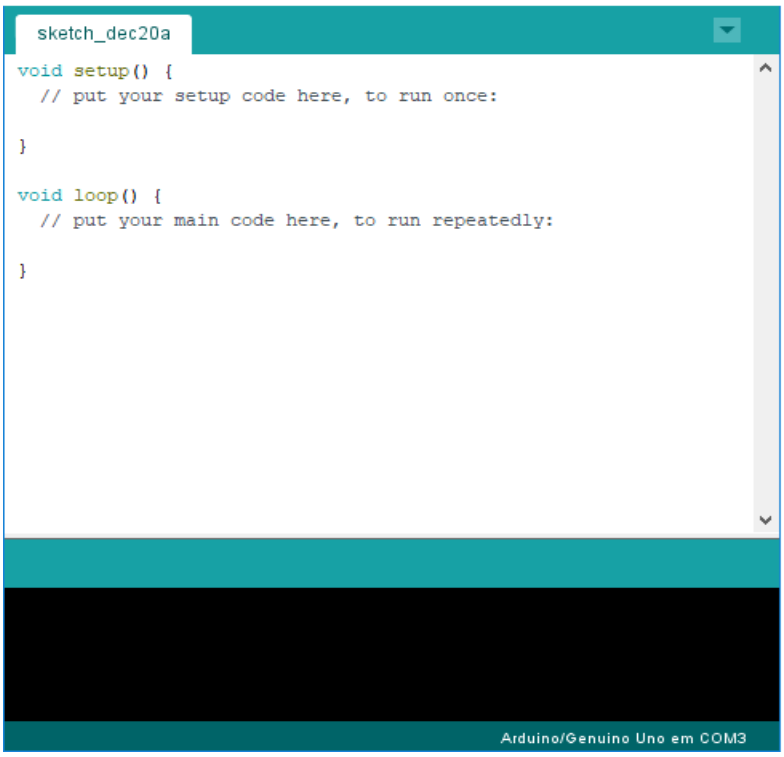


A barra de ferramentas, que possui os botões para a compilação dos programas, o botão para envio do programa ao Arduino, Novo Programa, Abrir, Salvar e, ao final da barra, o botão para abertura do monitor:

Figura 10: Barra de ferramentas - Arduino IDE

Temos, ainda, a área de edição, que é a área onde os programas são editados. Acima da área, numa aba, fica o nome do programa (chamado pelo Arduino IDE de *sketch*). Ao centro, a área de edição propriamente dita, já com as funções setup( ) e loop( ). Abaixo da área de edição estão a monitor de compilação (área em cor preta) e o *status* de comunicação com a placa, bem abaixo:

Figura 11: Área de edição - Arduino IDE



Como dito acima, a linguagem utilizada é baseada em C/C++ e é importante comparar esse ambiente a um programa em C, linguagem utilizada na disciplina de Algoritmos e Programação. A intenção do manual não é ensinar a programar o Arduino e, sim, utilizar a programação Arduino para aprender a programar o robô e, por consequência, aprender os conceitos básicos da lógica de programação. Para melhor compreensão do funcionamento da linguagem, utilizaremos um exemplo muito simples de programa.

Acima da função setup( ), podemos declarar constantes, variáveis de escopo global e a inclusão de bibliotecas, como na linguagem C. Nesse primeiro exemplo, só colocaremos o nome do programa e do autor, em forma de comentários.

A função setup( ) tem o objetivo de configurar o sistema. Ela é executada somente uma vez e deve ser utilizada para configurar os "canais" de comunicação da placa Arduino com o "mundo exterior".

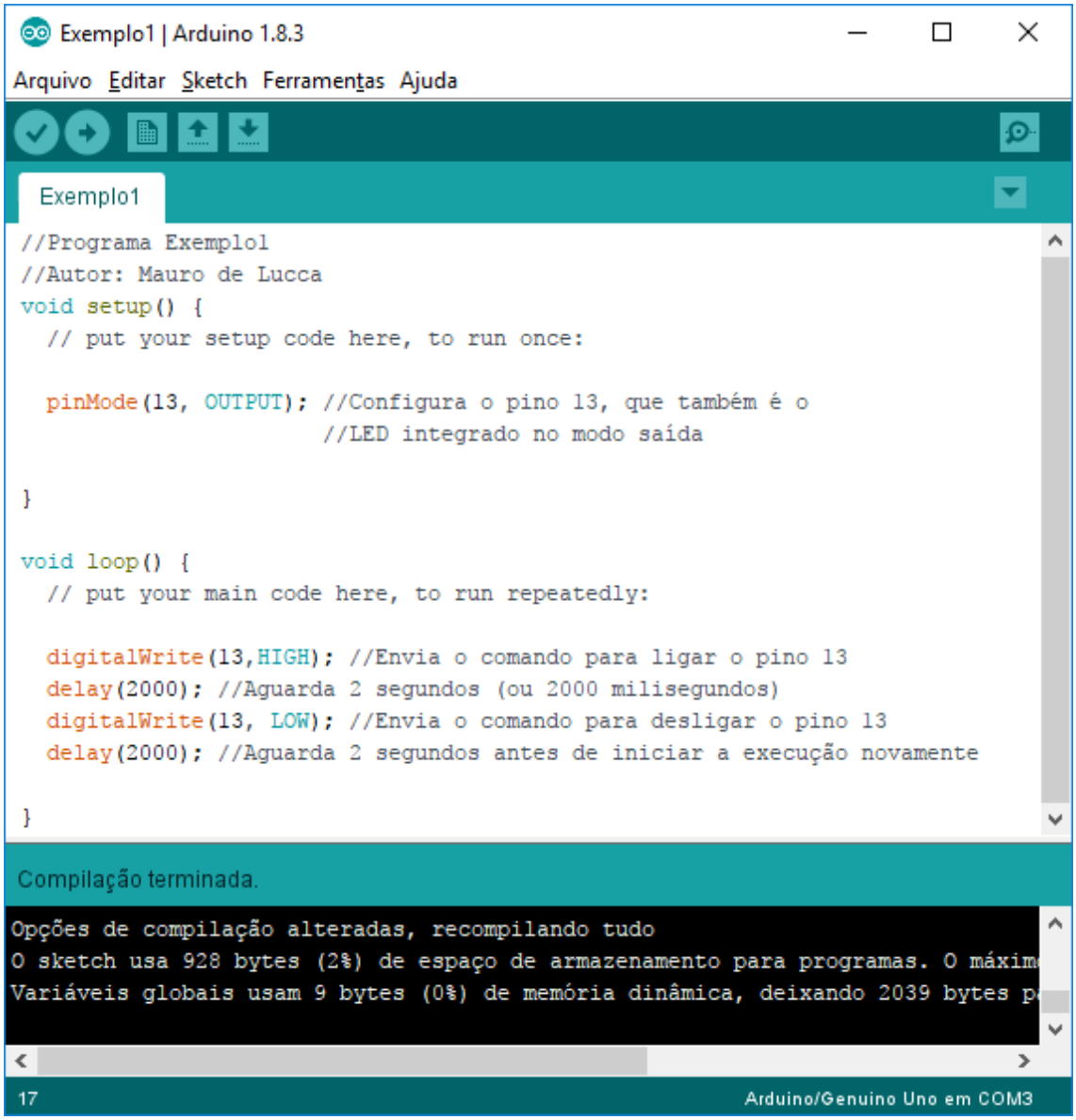
Na função loop( ) devemos colocar o programa que desejamos que o Arduino execute até ser desligado. Vale ressaltar que ela tem um funcionamento de repetição eterna (ou loop infinito). Nessa função, colocaremos, no decorrer do projeto, toda a "inteligência" que nosso robô deverá ter para funcionar autonomamente. Essa função seria a equivalente a função main( ) na linguagem C, porém com a diferença de executar infinitamente, ao chegar ao final. Ou seja, acabando o código, ele é executado novamente até que o Arduino seja desligado.

Nesse primeiro exemplo, apenas faremos um *led* interno da placa Arduino Uno piscar intermitentemente a cada 2 segundos. Toda a documentação necessária para a programação encontra-se no site do Arduino em:

https://playground.arduino.cc/Portugues/Referencia.

Segue o exemplo:

Figura 12: Programação do Arduino para piscar LED integrado



Ao conectar o Arduino Uno à porta USB e configurar adequadamente a porta de comunicação pelo menu Ferramenta >> Porta, você poderá enviar o programa à placa clicando no botão carregar (seta à direita, segundo botão na barra de ferramentas) ou pelo menu Sketck >> Carregar e verificar o resultado. Na imagem abaixo, há a indicação do *led* que deverá piscar a cada 2 segundos:

Figura 13: Arduino UNO com o LED integrado circulado

3.1.2. Estrutura Sequencial – Declaração de Constantes e Atribuição de Valores

Objetivo

Iniciar o trabalho com o robô móvel e incorporar os conceitos de declaração constantes.

Recursos

Aula prática com a utilização do robô móvel e Arduino IDE em laboratório de informática ou robótica. Utilizar grupos de quatro ou cinco discentes, preferencialmente.

Desenvolvimento

Iniciaremos a programação do robô móvel, trabalhando os conceitos de Estrutura Sequencial, a princípio utilizando a declaração de constantes e atribuição de valores.

Proposta 1: Fazer o robô se mover para a frente e parar a cada 5 segundos

Com o Arduino Motor Shield, devemos declarar a biblioteca AFMotor.h, da mesma forma em que declaramos na Linguagem C, no início do programa:

```
#include<AFMotor.h>
```

Também deveremos declarar os motores, conforme a ligação no *shield*, na área de declaração de constantes e variáveis de escopo global:

```
AF_DCMotor motor1(1); // Define o motor1 ligado a conexão M1
AF_DCMotor motor2(2); // Define o motor2 ligado a conexão M2
```

Na função de configuração setup( ), utilizaremos a função da biblioteca AFMotor setSpeed( ), onde passaremos, inicialmente a velocidade de funcionamento dos motores (valores de 0 a 255):

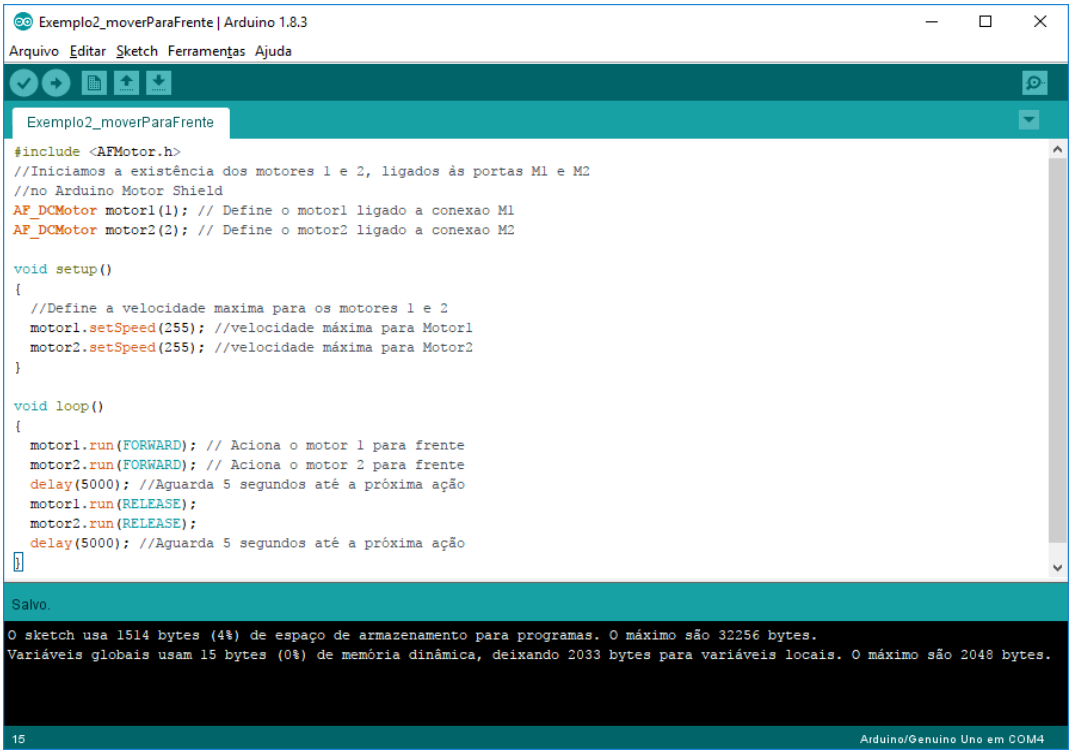
```
void setup()
{
  //Define a velocidade máxima para os motores 1 e 2
  motor1.setSpeed(255); //velocidade máxima para Motor1
  motor2.setSpeed(255); //velocidade máxima para Motor2
}
```

Para finalizar, passaremos a função loop( ), nossa função principal, lembrando que ela se repete até que o Arduino deixe de receber eletricidade. Nela colocaremos a programação que moverá o robô para frente e parará a cada 5 segundos. É importante verificar que utilizaremos a função run( ) para os motores, onde informaremos as constantes FORWARD e RELEASE para mover os motores para frente e parar, respectivamente:

```
void loop()
{
  motor1.run(FORWARD); // Aciona o motor 1 para frente
  motor2.run(FORWARD); // Aciona o motor 2 para frente
  delay(5000); //Aguarda 5 segundos até a próxima ação
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  delay(5000); //Aguarda 5 segundos até a próxima ação
}
```

Nossa programação, no Arduino IDE ficou assim:

Figura 14: Exemplo 2 - Mover Robô para frente



Proposta 2: A partir do exemplo anterior, utilizar a constante BACKWARD e fazer o robô se mover para trás após ir para frente por 5 segundos e parar.

O desenvolvimento é o mesmo, porém inclui-se aqui a constante BACKWARD para a função run( ). Após o carro ir para a frente por 5 segundos e parar, incluir a função run( ) para os dois motores com o valor BACKWARD. Como o restante do código é igual, colocarei aqui a função loop( ) alterada:

```
void loop()
{
  motor1.run(FORWARD); // Aciona o motor 1 para frente
  motor2.run(FORWARD); // Aciona o motor 2 para frente
  delay(5000); //Aguarda 5 segundos até a próxima ação
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  delay(5000); //Aguarda 5 segundos até a próxima ação
  motor1.run(BACKWARD); //Aciona o motor 1 para trás
  motor2.run(BACKWARD); //Aciona o motor 1 para trás
```

```
  delay(5000); //Aguarda 5 segundos até a próxima ação
}
```

Proposta 3: Declarar constantes para velocidades baixa e altas e mover o robô para frente com essas duas velocidades.

Para o desenvolvimento dessa proposta, utilizaremos a declaração de duas constantes: baixa e alta, com os valores de 100 e 255, respectivamente. Para utilizarmos estas constantes não incluiremos a função setSpeed( ) dentro da função setup( ), mas sim na função loop( ), no desenvolvimento do programa.

Segue abaixo o código:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); // Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); // Define o motor2 ligado a conexao M2

//Declaramos as constantes para as velocidades baixa e alta
//Essas constantes não poderão ser alteradas no decorrer do programa
const int baixa = 100;
const int alta = 255;

void setup()
{
}

void loop()
{
  motor1.setSpeed(baixa); // Configura velocidade baixa para motor1
  motor2.setSpeed(baixa); // Configura velocidade baixa para motor2
  motor1.run(FORWARD); // Aciona o motor 1 para frente
  motor2.run(FORWARD); // Aciona o motor 2 para frente
  delay(5000); // Aguardar 5 segundos
```

```
  motor1.run(RELEASE); // Para o motor 1
  motor2.run(RELEASE); // Para o motor 2
  delay(2000); // Aguardar 2 segundos

  motor1.setSpeed(alta); // Configura velocidade alta para motor1
  motor2.setSpeed(alta); // Configura velocidade alta para motor2
  motor1.run(FORWARD); // Aciona o motor 1 para frente
  motor2.run(FORWARD); // Aciona o motor 2 para frente
  delay(5000); // Aguardar 5 segundos
  motor1.run(RELEASE); // Para o motor 1
  motor2.run(RELEASE); // Para o motor 2
  delay(2000); // Aguardar 2 segundos
}
```

Propostas de exercícios:

1) Peça aos alunos para fazerem o robô móvel girar no sentido horário por cinco segundos, parar e girar no sentido anti-horário por cinco segundos.
2) Peça aos alunos para fazerem o robô móvel fazer um caminho de círculo para a esquerda.
3) Peça, agora para fazerem o robô móvel fazer um caminho em círculo para a direita.

3.1.3. Estrutura Condicional Simples – Entrada de Dados, Declaração de Variáveis e Lógica *Booleana*

Objetivo

Trabalhar conceitos de Estrutura Condicional Simples (Se... Então), entrada de dados, declaração de variáveis e lógica *booleana* através do uso de sensores no robe móvel.

Recursos

Aula prática com a utilização do robô móvel, sensores de obstáculos e Arduino IDE em laboratório de informática ou robótica. Utilizar grupos de quatro ou cinco discentes, preferencialmente.

Desenvolvimento

Programação do robô móvel, trabalhando os conceitos de Estrutura Condicional Simples, entrada de dados, declaração de variáveis e lógica *booleana*, utilizando os sensores de obstáculos.

Proposta 1: Fazer o robô se mover para a frente e parar se encontrar algum obstáculo à frente

Continuaremos com as declarações da biblioteca AFMotor.h e todas as outras relativas aos motores. Incluiremos nesse código, a declaração da porta em que está ligada ao sensor de obstáculos frontal (no caso do robô móvel utilizado para os testes, a porta A4). Também declararemos uma variável para captar o estado do sensor – 0 para encontrou obstáculo e 1 para não encontrou obstáculo.

A lógica utilizada foi a de que os motores são impulsionados para a frente e, se um obstáculo for encontrado o robô móvel deverá ficar parado até que o obstáculo seja retirado. É preciso colocar um comando de espera para que a leitura do sensor seja processada.

O código desenvolvido foi esse:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2


int estdSensorFr; //Declaração da variável para o estado do sensor
//frontal, do tipo inteiro


void setup()
{
  //Define a velocidade para os motores 1 e 2
  motor1.setSpeed(180); //velocidade para Motor1
  motor2.setSpeed(180); //velocidade para Motor2
  //Configura Sensor de Obstáculo em A4
  pinMode(A4, INPUT); //Porta em que está configurada o Sensor frontal //-
A4 como entrada de dados
```

```
}


void loop()
{
  motor1.run(FORWARD); //Aciona o motor 1 para frente
  motor2.run(FORWARD); //Aciona o motor 2 para frente


  estdSensorFr = digitalRead(A4); //estado do sensor frontal recebe a
//leitura deste sensor


  if (estdSensorFr == 0) //Se encontrar um obstáculo (se estdSensorFR //for
igual zero)
  {
   motor1.run(RELEASE); //Parar motor 1
   motor2.run(RELEASE); //Parar motor 2
   delay(500); //tempo de espera para a leitura do estado do sensor
//frontal - meio segundo basta
  }
}
```

Proposta 2: Fazer o robô móvel girar quando encontrar um obstáculo – princípio do carrinho de bate-e-volta.

Agora, a proposta é fazer o robô móvel dar meia volta quando o sensor de obstáculo frontal detectar um obstáculo à frente.

Para realizar essa proposta, faremos algumas mudanças no código anterior. Encaminharemos os motores para frente e, se o estado do sensor for igual a 0, faremos girar, calibrando o tempo necessário para que ele dê meia volta.

Segue o código:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2
```

```
int estdSensorFr; //Declaração da variável para o estado do sensor
//frontal, do tipo inteiro


void setup()

{

  //Define a velocidade para os motores 1 e 2

  motor1.setSpeed(180); //velocidade para Motor1

  motor2.setSpeed(180); //velocidade para Motor2

  //Configura Sensor de Obstáculo em A4

  pinMode(A4, INPUT); //Porta em que está configurada o Sensor frontal //-
A4 como entrada de dados

}


void loop()

{

  motor1.run(FORWARD); //Aciona o motor 1 para frente

  motor2.run(FORWARD); //Aciona o motor 2 para frente


  estdSensorFr = digitalRead(A4); //estado do sensor frontal recebe a
//leitura deste sensor


  if (estdSensorFr == 0) //Se encontrar um obstáculo (se estdSensorFR //for
igual zero)

  {

    motor1.run(BACKWARD); //Aciona o motor 1 para trás

    motor2.run(FORWARD); //Aciona o motor 2 para frente

    delay(500); //Tempo para execução do giro

  }

}
```

Proposta 3: Fazer o robô móvel seguir uma parede utilizando os dois sensores de obstáculo.

Para a concretização dessa proposta, é necessário que os alunos já tenham contato com a Tabela-verdade. Utilizaremos o sensor frontal e o sensor lateral (no robô exemplo, ligado à porta A5). O robô móvel seguirá a parede até encontrar um obstáculo.

No desenvolvimento, podemos encontrar algum problema com o rodízio universal. Não esquecer de alinhá-lo antes de colocar o robô para se mover. Para o caso do robô se distanciar da parede, programei para que pare, também.

Utilizaremos os códigos desenvolvidos até agora para continuar. Aqui, faremos a leitura dos dois sensores no início do código. Utilizaremos os operadores lógicos E (&&) e OU (||) para testar as condições das Estruturas Condicionais Simples utilizadas no programa.

SE o estado do sensor lateral detectar a parede E o estado do sensor frontal não detectar obstáculos, os motores devem ser impulsionados para frente.

SE o estado do sensor frontal detectar obstáculo OU o estado do sensor lateral deixar de detectar a parede, os motores deverão ser parados.

Segue o código:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2

//Declaração das variáveis para o estado dos 2 sensores, do tipo inteiro
int estdSensorFr, estdSensorLt;

void setup()
{
  //Define a velocidade para os motores 1 e 2
  motor1.setSpeed(180); //velocidade para Motor1
  motor2.setSpeed(180); //velocidade para Motor2
  //Configura Sensor de Obstáculo em A4
  pinMode(A4, INPUT); //Porta em que está configurada o Sensor frontal //-
A4 como entrada de dados
  //Configura Sensor de Obstáculo em A5
  pinMode(A5, INPUT); //Porta em que está configurada o Sensor lateral //-
A5 como entrada de dados
}
```

```
void loop()
{
  estdSensorFr = digitalRead(A4); //estado do sensor frontal recebe a
//leitura deste sensor
  estdSensorLt = digitalRead(A5); //estado do sensor lateral recebe a
//leitura deste sensor
//Se Sensor lateral identificar a parede E Sensor frontal não identificar
//obstáculo
  if (estdSensorLt == 0 && estdSensorFr == 1)
  {
    motor1.run(FORWARD); //Aciona o motor 1 para frente
    motor2.run(FORWARD); //Aciona o motor 2 para frente
  }

//Se encontrar um obstáculo (se estdSensorFR for igual zero
//OU deixar de detectar a parede (|| estdSensorLt for igual a 1
  if (estdSensorFr == 0 || estdSensorLt == 1)
      {
        motor1.run(RELEASE); //Parar o motor 1
        motor2.run(RELEASE); //Parar o motor 2
        delay(500); //Tempo para processar a leitura
      }
}
```

Propostas de Exercícios

1) Fazer o robô móvel dar marcha-à-ré quando encontrar um obstáculo.
2) Criar um programa que faça o robô móvel contornar uma barreira.
3) Criar um mecanismo que corrija o traçado do robô móvel quando ele se “descolar” da parede.

3.1.4. Estrutura Condicional Composta

Objetivo

Trabalhar conceitos de Estrutura Condicional Composta (Se... Então... Senão) e os conceitos aprendidos anteriormente.

Recursos

Aula prática com a utilização do robô móvel, sensores de obstáculos e Arduino IDE em laboratório de informática ou robótica. Utilizar grupos de quatro ou cinco discentes, preferencialmente.

Desenvolvimento

Programação do robô móvel, trabalhando os conceitos de Estrutura Condicional Composta, entrada de dados, declaração de variáveis e lógica *booleana*, utilizando os sensores de obstáculos e saída de dados, utilizando o *buzzer*.

Proposta 1: Fazer com que o robô móvel se movimente para a frente sempre que não houver obstáculos.

Essa proposta já foi efetuada acima, utilizando a Estrutura Condicional Simples, porém, com a Estrutura Condicional Composta a programação fica simplificada, pois, ganhamos a opção de dois caminhos a seguir: movimentar para frente ou ficar parado conforme a condição imposta seja verdadeira ou falsa, respectivamente.

Utilizaremos o teste de condição (existir obstáculo) para o sensor frontal. Nesse caso, o robô móvel deverá se movimentar sempre que o estado do sensor frontal indicar o caminho livre, ou seja, estdSensorFr deve ser igual a 1 (estdSensorFr == 1, no código). Quando esta condição não for atendida, estdSensorFr for diferente de 1, o robô móvel deverá permanecer parado.

Segue o código:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2

//Declaração das variáveis para o estado do sensor, do tipo inteiro
int estdSensorFr;

void setup()
{
```

```
  //Define a velocidade para os motores 1 e 2
  motor1.setSpeed(180); //velocidade para Motor1
  motor2.setSpeed(180); //velocidade para Motor2
  //Configura Sensor de Obstáculo Frontal em A4
  pinMode(A4, INPUT); //Configurar porta A4 como entrada de dados
}

void loop()
{
//estado do sensor frontal recebe a leitura deste sensor
  estdSensorFr = digitalRead(A4);

//Se Sensor frontal não identificar obstáculo
  if (estdSensorFr == 1)
  {
    motor1.run(FORWARD); //Aciona o motor 1 para frente
    motor2.run(FORWARD); //Aciona o motor 2 para frente
  }
//Senão (Sensor frontal identificou obstáculo)
//Negação da condição testada acima
  else
  {
    motor1.run(RELEASE); //Parar o motor 1
    motor2.run(RELEASE); //Parar o motor 2
  }
}
```

Proposta 2: Recriar o carro de bate-e-volta, utilizando Estrutura Condicional Composta.

Mais uma vez, a proposta segue simplificada pelo fato de utilizarmos uma única condição para movermos o robô móvel para a frente e a negação dessa condição para fazermos o robô móvel dar a meia volta. O código é muito parecido com o anterior, mas, desta vez deveremos calibrar o giro do robô móvel para a continuidade de seu movimento. Nesse caso, moveremos o Motor 1 (Esquerda) para trás por 1 segundo, para realizar o giro.

Lembrando sempre que o código se repete quando chega ao final.

Eis nosso exemplo:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2

//Declaração das variáveis para o estado do sensor, do tipo inteiro
int estdSensorFr;

void setup()
{
  //Define a velocidade para os motores 1 e 2
  motor1.setSpeed(100); //velocidade para Motor1
  motor2.setSpeed(100); //velocidade para Motor2
  //Configura Sensor de Obstáculo Frontal em A4
  pinMode(A4, INPUT); //Configurar porta A4 como entrada de dados
}

void loop()
{
  estdSensorFr = digitalRead(A4);
//estado do sensor frontal recebe a leitura deste sensor

//Se Sensor frontal não identificar obstáculo
  if (estdSensorFr == 1)
  {
    motor1.run(FORWARD); //Aciona o motor 1 para frente
    motor2.run(FORWARD); //Aciona o motor 2 para frente
  }
//Senão (Sensor frontal identificou obstáculo)
//Negação da condição testada acima
  else
  {
```

```
    motor1.run(BACKWARD); //Move o motor 1 para trás
    delay(1000); //Durante 1 segundo
  }
}
```

Proposta 3: Fazer o robô móvel dar marcha-à-ré e emitir um som antes de dar a meia volta.

Essa proposta é parecida com a anterior, mas, a intenção desse exercício é trabalhar a saída de informações, no caso o som, juntamente com o movimento robô móvel.

Utilizaremos, para essa proposta, uma função nova: digitalWrite( ). Essa função tem o objetivo de enviar um comando de saída para a porta indicada. Como vamos utilizar a porta A0, no caso, como digital, ela poderá assumir os valores Ligado (HIGH) ou Desligado (LOW) e, dessa forma, emitir um som através do *buzzer*. Seu funcionamento é digitalWrite(porta de saída, modo) – no exemplo, digitalWrite(A0, HIGH) para emitir o som e digitalWrite(A0, LOW) para desligar o som.

Assim, temos o seguinte programa:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2
//Declaração das variáveis para o estado do sensor, do tipo inteiro
int estdSensorFr;

void setup()
{
  //Define a velocidade para os motores 1 e 2
  motor1.setSpeed(120); //velocidade para Motor1
  motor2.setSpeed(120); //velocidade para Motor2
  //Configura Sensor de Obstáculo Frontal em A4
  pinMode(A4, INPUT); //Configurar porta A4 como entrada de dados
  //Configura o buzzer na porta A0
  pinMode(A0, OUTPUT); //Configurar porta A0 como saída de dados
```

```
}

void loop()
{
  estdSensorFr = digitalRead(A4);
//estado do sensor frontal recebe a leitura deste sensor

//Se Sensor frontal não identificar obstáculo
  if (estdSensorFr == 1)
  {
    motor1.run(FORWARD); //Aciona o motor 1 para frente
    motor2.run(FORWARD); //Aciona o motor 2 para frente
  }
//Senão (Sensor frontal identificou obstáculo)
//Negação da condição testada acima
  else
  {
    digitalWrite(A0, HIGH); //Ativa o som no Buzzer
    motor1.run(BACKWARD); //Move o motor 1 para trás
    motor2.run(BACKWARD); //Move o motor 2 para trás
    delay(1000); //Durante 1 segundo
    digitalWrite(A0, LOW); //Desativa o som no Buzzer
    motor2.run(FORWARD); //Aciona o motor 2 para frente
    delay(1000); //Por 1 segundo
  }
}
```

Propostas de exercícios

1) Fazer o robô móvel contornar um obstáculo pequeno, utilizando a Estrutura Condicional Composta.
2) Fazer o robô móvel seguir a parede até encontrar um obstáculo, utilizando os dois sensores de obstáculos.
3) Fazer o robô móvel seguir a parede e emitir um aviso sonoro, caso “descole” da parede.

3.1.5. Estrutura Condicional Composta Encadeada

Objetivo

Trabalhar conceitos de Estrutura Condicional Composta Encadeada (Se... Senão-Se) e os conceitos aprendidos anteriormente.

Recursos

Aula prática com a utilização do robô móvel, sensores de obstáculos, sensores de contraste, *buzzer* e Arduino IDE em laboratório de informática ou robótica. Utilizar grupos de quatro ou cinco discentes, preferencialmente.

Desenvolvimento

Programação do robô móvel, trabalhando os conceitos de Estrutura Condicional Composta encadeada, entrada de dados, declaração de variáveis e lógica *booleana*, utilizando os sensores de obstáculos, sensores de contraste e saída de dados, utilizando o *buzzer*.

Proposta 1: Fazer com que o robô móvel decida para qual lado virar, caso encontre um obstáculo.

Movimentaremos o robô móvel até que ele encontre algum obstáculo. Quando encontrar, pararemos os motores e aguardaremos para que o robô tome uma decisão, conforme sua programação: Se o sensor lateral, que fica à direita da parte frontal do robô, não encontrar nenhum obstáculo, ele virará à direita. Caso haja obstáculo também à direita, virará à esquerda.

Encadearemos as estruturas condicionais de forma que o robô só tomará a decisão para qual lado virará quando encontrar um obstáculo à frente. Assim teremos uma estrutura condicional composta principal, que guia o robô adiante até que encontre um obstáculo, e outra estrutura condicional composta, dentro dessa primeira estrutura, que guiará o robô móvel para qual lado virar, direita ou esquerda.

Sendo assim, teremos o seguinte código:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
```

```
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2

//Declaração das variáveis para o estado dos 2 sensores, do tipo inteiro
int estdSensorFr, estdSensorLt;

void setup()
{
  //Define a velocidade para os motores 1 e 2
  motor1.setSpeed(100); //velocidade para Motor1
  motor2.setSpeed(100); //velocidade para Motor2
  //Configura Sensor de Obstáculo em A4
  pinMode(A4, INPUT); //Configura porta A4 como entrada de dados
  //Confirgura Sensor de Obstáculo em A5
  pinMode(A5, INPUT); //Configura porta A5 como entrada de dados
}

void loop()
{
  estdSensorFr = digitalRead(A4); //Leitura do estado do sensor frontal

//
  if (estdSensorFr == 1)
  {
    motor1.run(FORWARD); //Aciona o motor 1 para frente
    motor2.run(FORWARD); //Aciona o motor 2 para frente
  }
  else
  {
    //Parar o robô móvel
    motor1.run(RELEASE);
    motor2.run(RELEASE);
    delay(2000); //Tempo de espera 2 segundos

    //Leitura do sensor lateral
    estdSensorLt = digitalRead(A5); //estado do sensor lateral
```

```
    //Se direita estiver livre
    if (estdSensorLt == 1)
    {
      //Virar à direita
      motor1.run(FORWARD);
      motor2.run(BACKWARD);
      delay(1000);
    }
    //Se direita não estiver livre
    else
    {
      //Virar à esquerda
      motor1.run(BACKWARD);
      motor2.run(FORWARD);
      delay(1000);
    }
  }
}
```

Proposta 2: Utilizar o sensor de contraste do meio para fazer o robô móvel seguir uma faixa branca (ou clara) no solo. Quando acabar a faixa, o robô deve continuar se movendo e emitindo um bip até encontrar um obstáculo.

Para essa proposta, utilizaremos o sensor de contraste do meio, ligado à porta A3, o *buzzer* ligado à porta A0 e o sensor de obstáculo frontal ligado à porta A4. Faremos a leitura do estado do sensor de contraste e armazenaremos na variável estdSensorMd. É importante salientar, nesse momento, que todos os sensores que utilizamos e utilizaremos no restante do manual são binários – só assumem os estados 0 (zero) e 1 (um) ou verdadeiro ou falso. Isso foi feito pensando em facilitar o entendimento dos conceitos, em especial nos testes de condições.

O sensor de contraste assume os estados 0 (zero) para faixa clara e 1 (um) para faixa escura. Se o estado for igual a zero, os motores deverão impulsionar o robô móvel para frente e essa é a condição única para o robô iniciar seu movimento. Caso venha a perder a faixa clara, o robô deverá continuar se movendo para frente até encontrar um obstáculo. Nesse

movimento, sem a faixa clara para seguir, o robô deverá emitir um bip intermitente. Quando encontrar um obstáculo, deverá parar.

Notaremos, nesse programa, que após perder a faixa clara, o robô iniciará o bip. Quando encontrar um obstáculo, esse bip continuará mais lento, mas não parará até que o robô volte a se mover com a faixa clara por baixo do sensor. A próxima proposta será a de fazer o robô não se mover e nem fazer o bip após encontrar o obstáculo.

Nesse exemplo, haverá uma estrutura condicional composta que perguntará sobre o estado do sensor de contraste do meio e a outra encadeada, porém, agora uma estrutura condicional simples, perguntado sobre o estado do sensor de obstáculo frontal.

Abaixo, o código do programa:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2

//Declaração das variáveis para o estado dos 2 sensores, do tipo inteiro
int estdSensorFr, estdSensorMd;

void setup()
{
  //Define a velocidade para os motores 1 e 2
  motor1.setSpeed(100); //velocidade para Motor1
  motor2.setSpeed(100); //velocidade para Motor2
  //Configura Sensor de Obstáculo em A4
  pinMode(A4, INPUT); //Configura porta A4 como entrada de dados
  //Configura Sensor de Contraste em A1
  pinMode(A3, INPUT); //Configura porta A1 como entrada de dados
  //Configura Buzzer em A0
  pinMode(A0, OUTPUT); //Configura porta A0 como saída de dados
}
```

```
void loop()
{
  estdSensorMd = digitalRead(A3); //Leitura do estado do sensor de contraste

//Se encontrou o caminho claro
  if (estdSensorMd == 0) //Caminho claro -> estdSensorMd == 0
  {
    motor1.run(FORWARD); //Aciona o motor 1 para frente
    motor2.run(FORWARD); //Aciona o motor 2 para frente
  }
  //Se deixou de encontrar o caminho claro
  else
  {
    //Bip
    digitalWrite(A0,HIGH);
    delay(300);
    digitalWrite(A0, LOW);
    delay(300);

    //Faz a leitura do sensor de obstáculo
    estdSensorFr = digitalRead(A4);
    //Se encontrou obstáculo
    if (estdSensorFr == 0)
    {
      //Parar o robô móvel
      motor1.run(RELEASE);
      motor2.run(RELEASE);
      delay(1000); //Tempo de espera 1 segundo
    }
  }
}
```

Proposta 3: Fazer o mesmo que a proposta anterior, porém, nessa proposta o robô móvel deverá parar em repouso absoluto após encontrar o obstáculo.

Teremos, para essa proposta, basicamente o mesmo código, porém com uma variável que detectará o estado do robô. Nessa proposta, processaremos uma variável que determinará o estado em que o robô deverá iniciar e o estado em que ele ficará parado.

Para isso, declararemos uma variável com um valor inicial, que será testada no início do programa. Portanto, teremos mais uma estrutura condicional simples abrangendo o restante do programa. Nesse código, utilizaremos, também, o operador lógico NÃO (!, na programação em C ou em Arduino).

Segue o código:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2

//Declaração das variáveis para o estado dos 2 sensores, do tipo inteiro
int estdSensorFr, estdSensorMd;
//Declaração da variável de estado do robô, do tipo inteiro
//sendo iniciada com valor 1
int estdRobo = 1;

void setup()
{
  //Define a velocidade para os motores 1 e 2
  motor1.setSpeed(100); //velocidade para Motor1
  motor2.setSpeed(100); //velocidade para Motor2
  //Configura Sensor de Obstáculo em A4
  pinMode(A4, INPUT); //Configura porta A4 como entrada de dados
  //Configura Sensor de Contraste em A1
  pinMode(A3, INPUT); //Configura porta A1 como entrada de dados
  //Configura Buzzer em A0
  pinMode(A0, OUTPUT); //Configura porta A0 como saída de dados
}

void loop()
```

```
{
  //Só iniciará se o estado do robô for diferente de zero
  if (estdRobo != 0) //Se (estdRobo NÃO IGUAL a 0)
  {
    estdSensorMd = digitalRead(A3); //Leitura do estado do sensor de contraste

  //Se encontrou o caminho claro
    if (estdSensorMd == 0) //Caminho claro -> estdSensorMd == 0
    {
      motor1.run(FORWARD); //Aciona o motor 1 para frente
      motor2.run(FORWARD); //Aciona o motor 2 para frente
    }
    //Se deixou de encontrar o caminho claro
    else
    {
      //Bip
      digitalWrite(A0,HIGH);
      delay(300);
      digitalWrite(A0, LOW);
      delay(300);

      //Faz a leitura do sensor de obstáculo
      estdSensorFr = digitalRead(A4);
      //Se encontrou obstáculo
      if (estdSensorFr == 0)
      {
        //Parar o robô móvel
        motor1.run(RELEASE);
        motor2.run(RELEASE);
        delay(5000); //Tempo de processamento
        //Aqui atribuímos o valor zero para a variável estdRobo
        estdRobo = 0;
      }
    }
  }
}
```

Propostas de exercícios

1) Fazer um código de carro Bate-e-Volta que se desliga após encontrar o quinto obstáculo.
2) Fazer um robô que siga duas paredes em forma de L. Cuidado ao calibrar o tempo de giro.
3) Elaborar um código para o robô móvel que siga uma faixa clara e pare no momento em que a faixa clara abranger os três sensores de contraste.

3.1.6. Estruturas de Repetição

Objetivo

Trabalhar conceitos das Estruturas de Repetição: com teste no início, com teste no final e com número definido de repetições.

Recursos

Aula prática com a utilização do robô móvel, sensores de obstáculos, sensores de contraste, *buzzer* e Arduino IDE em laboratório de informática ou robótica. Utilizar grupos de quatro ou cinco discentes, preferencialmente.

Desenvolvimento

Programação do robô móvel, trabalhando os conceitos de Estruturas de Repetição, entrada de dados, declaração de variáveis e lógica *booleana*, utilizando os sensores de obstáculos, sensores de contraste e saída de dados, utilizando o *buzzer*.

Proposta 1: Desenvolver um robô móvel seguidor de faixa clara numa pista escura utilizando uma estrutura de repetição com teste no início (ENQUANTO... FAÇA)

Para o desenvolvimento dessa proposta, utilizaremos os três sensores de contraste e, por consequência, três variáveis para armazenar o estado de cada sensor. Sempre é válido lembrar que o conceito de repetição vem sendo trabalhado em todos os exemplos até aqui, pois o Arduino funciona em modo de repetição infinita. Não há um teste de condições na função loop( ).

Nesse exemplo, não configuraremos a velocidade dos motores na função setup( ), mas sim após as leituras de estado dos sensores, para que possamos corrigir o traçado dos motores.

A lógica utilizada, em princípio, é a de que o robô móvel deverá se mover mais rápido sempre que o sensor do meio detectar a faixa. Quando nenhum dos três sensores detectar a faixa, o robô deverá se mover mais devagar. E quando um dos sensores das extremidades detectar a faixa, o motor contrário a esse sensor deverá girar mais rápido para corrigir o traçado.

Segue o exemplo com a lógica dada:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2

//Declaração das variáveis para o estado dos 3 sensores, do tipo inteiro
int estdSensorMd = 0, estdSensorEq = 0, estdSensorDt = 0;

void setup()
{
  //Configura Sensores de Contraste
  //Sensor da Direita
  pinMode(A1, INPUT); //Configura porta A1 como entrada de dados
  //Sensor da Esquerda
  pinMode(A2, INPUT); //Configura porta A2 como entrada de dados
  //Sensor do Meio
  pinMode(A3, INPUT); //Configura porta A3 como entrada de dados
}

void loop()
{
  //Enquanto o estado do sensor do meio detectar a faixa clara
  while (estdSensorMd == 0)
  {
    //Os dois motores devem girar mais rápido
    motor1.setSpeed(150); //velocidade para Motor1
    motor2.setSpeed(150); //velocidade para Motor2
    motor1.run(FORWARD);
```

```
  motor2.run(FORWARD);
  //Leitura do sensor do meio para detectar seu estado
  estdSensorMd = digitalRead(A3);
}

//Quando o Sensor do meio deixar de detectar a faixa
//É feita a leitura dos outros dois sensores
estdSensorDt = digitalRead(A1);
estdSensorEq = digitalRead(A2);

if (estdSensorDt == 0) //Se o sensor direito detectar a faixa
{
  //O motor esquerdo deve girar mais rápido
  motor1.setSpeed(150); //velocidade para Motor1
  motor2.setSpeed(80); //velocidade para Motor2
  motor1.run(FORWARD);
  motor2.run(FORWARD);
}
else //Caso contrário
{
  if (estdSensorEq == 0) //Se o sensor esquerdo detectar a faixa
  {
    //O motor direito deverá girar mais rápido
    motor1.setSpeed(80); //velocidade para Motor1
    motor2.setSpeed(150); //velocidade para Motor2
    motor1.run(FORWARD);
    motor2.run(FORWARD);
  }
  else //Caso nenhum detecte a faixa
  {
    //O robô deverá se mover mais lento
    motor1.setSpeed(80); //velocidade para Motor1
    motor2.setSpeed(80); //velocidade para Motor2
    motor1.run(FORWARD);
    motor2.run(FORWARD);
    //E fazer a leitura do sensor do meio
```

```
      estdSensorMd = digitalRead(A3);
    }
  }
}
```

Proposta 2: Refazer o código anterior, utilizando no robô móvel uma estrutura de repetição com teste no final (FAÇA... ENQUANTO).

Neste próximo exemplo, a ideia é o aluno poder comparar as duas estruturas de repetição. Enquanto na estrutura com teste no início (ENQUANTO... FAÇA), temos que dar as condições para que a estrutura seja executada, na estrutura com teste no final o bloco de comandos será executado pelo menos uma vez e a condição será testada no final dessa execução.

A lógica de desenvolvimento é a mesma, com a diferença de que não precisaremos inicializar as variáveis, pois faremos a leitura na primeira execução da estrutura de repetição.

Segue o código do exemplo:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2

//Declaração das variáveis para o estado dos 3 sensores, do tipo inteiro
int estdSensorMd, estdSensorEq, estdSensorDt;

void setup()
{
  //Configura Sensores de Contraste
  //Sensor da Direita
  pinMode(A1, INPUT); //Configura porta A1 como entrada de dados
  //Sensor da Esquerda
  pinMode(A2, INPUT); //Configura porta A2 como entrada de dados
  //Sensor do Meio
```

```
  pinMode(A3, INPUT); //Configura porta A3 como entrada de dados
}

void loop()
{

  do //Faça
  {
    //Os dois motores devem girar mais rápido
    motor1.setSpeed(150); //velocidade para Motor1
    motor2.setSpeed(150); //velocidade para Motor2
    motor1.run(FORWARD);
    motor2.run(FORWARD);
    //Leitura do sensor do meio para detectar seu estado
    estdSensorMd = digitalRead(A3);
  } while (estdSensorMd == 0);
  //Enquanto o estado do sensor do meio detectar a faixa clara

  //Quando o Sensor do meio deixar de detectar a faixa
  //É feita a leitura dos outros dois sensores
  estdSensorDt = digitalRead(A1);
  estdSensorEq = digitalRead(A2);

  if (estdSensorDt == 0) //Se o sensor direito detectar a faixa
  {
    //O motor esquerdo deve girar mais rápido
    motor1.setSpeed(150); //velocidade para Motor1
    motor2.setSpeed(80); //velocidade para Motor2
    motor1.run(FORWARD);
    motor2.run(FORWARD);
  }
  else //Caso contrário
  {
    if (estdSensorEq == 0) //Se o sensor esquerdo detectar a faixa
    {
      //O motor direito deverá girar mais rápido
```

```
      motor1.setSpeed(80); //velocidade para Motor1
      motor2.setSpeed(150); //velocidade para Motor2
      motor1.run(FORWARD);
      motor2.run(FORWARD);
    }
    else //Caso nenhum detecte a faixa
    {
      //O robô deverá se mover mais lento
      motor1.setSpeed(80); //velocidade para Motor1
      motor2.setSpeed(80); //velocidade para Motor2
      motor1.run(FORWARD);
      motor2.run(FORWARD);
      //E fazer a leitura do sensor do meio
      estdSensorMd = digitalRead(A3);
    }
  }
}
```

Proposta 3: Fazer o robô móvel emitir uma quantidade determinada de bips quando encontrar um obstáculo.

A estrutura de repetição com número definido (PARA... FAÇA) é útil quando sabemos a quantidade específica de repetições que a estrutura deve realizar. Nessa situação, faremos uma proposta simples, porém eficaz, para o entendimento dessa estrutura. Faremos o robô emitir dez vezes o bip quando encontrar um obstáculo em sua frente.

A lógica para essa proposta é a de fazer o robô se mover até encontrar um obstáculo, com o sensor frontal. Quando isso ocorrer, o robô deverá emitir dez bips, fazer a meia volta e continuar.

Segue o código comentado:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2
```

```
//Declaração das variáveis para o estado do sensor, do tipo inteiro
int estdSensorFr, estdSensorLt;

//Declaração de variável que servirá como contador de repetições
int i;

void setup()
{
  //Define a velocidade para os motores 1 e 2
  motor1.setSpeed(120); //velocidade para Motor1
  motor2.setSpeed(120); //velocidade para Motor2
  //Configura Sensor de Obstáculo Frontal em A4
  pinMode(A4, INPUT); //Configurar porta A4 como entrada de dados
  //Configura o buzzer na porta A0
  pinMode(A0, OUTPUT); //Configurar porta A0 como saída de dados
}

void loop()
{
  estdSensorFr = digitalRead(A4);
//estado do sensor frontal recebe a leitura deste sensor

//Se Sensor frontal não identificar obstáculo
  if (estdSensorFr == 1)
  {
    motor1.run(FORWARD); //Aciona o motor 1 para frente
    motor2.run(FORWARD); //Aciona o motor 2 para frente
  }
//Senão (Sensor frontal identificou obstáculo)
//Negação da condição testada acima
  else
  {
    motor1.run(RELEASE); //Parar o motor 1
    motor2.run(RELEASE); //Parar o motor 2
```

```
    //Para(contador i de 0 a 9 com incremento de 1 em 1) Faça
    for(i=0; i<10; i++)
    {
       digitalWrite(A0, HIGH); //Ativa o som no Buzzer
       delay(500); //Tempo do BIP
       digitalWrite(A0, LOW); //Desativa o som no Buzzer
       delay(500); //Tempo do BIP
    }
    //Meia volta
    motor1.run(FORWARD); //Move o motor 1 para trás
    motor2.run(BACKWARD); //Move o motor 2 para trás
    delay(1000); //Tempo da meia volta
  }
}
```

Propostas de Exercícios

1) Utilizar a estrutura de repetição com teste no início para desligar o robô móvel, quando encontrar o terceiro obstáculo.
2) Refazer o exercício anterior utilizando a estrutura de repetição com teste no final.
3) Utilizar uma estrutura de repetição para calcular quanto tempo, em segundos, um robô seguidor de faixa fica com os três sensores sem detectar a faixa. O robô móvel deverá parar quando encontrar uma faixa transversal. Nesse momento, utilizar o tempo calculado parar emitir bips.

3.1.7. Funções

Objetivo

Trabalhar a criação de funções juntamente com todos os conceitos aprendidos.

Recursos

Aula prática com a utilização do robô móvel, sensores de obstáculos, sensores de contraste, *buzzer* e Arduino IDE em laboratório de informática ou robótica. Utilizar grupos de quatro ou cinco discentes, preferencialmente.

Desenvolvimento

Para a construção de funções próprias, faremos uma programação para o robô móvel utilizando todos os sensores e o *buzzer*, utilizando todos os conhecimentos aprendidos. Faremos numa proposta única um robô móvel seguidor de faixa, que quando perde a faixa deve ser capaz de continuar se movendo com autonomia.

Proposta única: Criar um robô móvel seguidor de faixa, que se mova com autonomia quando perder a faixa.

Para a criação do código desse robô utilizaremos todos os recursos aprendidos até aqui, com a diferença de que tentaremos colocar os códigos dentro de funções específicas.

Nessa proposta, nosso objetivo é criar funções básicas como moverFrente( ), virarEsquerda( ), virarDireita( ), tocarBip( ), pararRobo( ) e todas as outras que acharmos necessárias para o funcionamento e organização do código para o problema proposto.

O robô móvel deverá seguir uma faixa clara ou se comportar autonomamente, em velocidade reduzida até encontrar novamente a faixa. O robô deverá parar,tocar um bip, pelo tempo que se desejar em segundos, e ser desligado quando encontrar uma faixa clara transversal, que ocupe os três sensores de contraste.

As funções criadas para o funcionamento do robô móvel serão declaradas após a inclusão das bibliotecas e declaração de variáveis globais e antes da função setup( ), conforme mostra o código a seguir:

```
#include <AFMotor.h>
//Iniciamos a existência dos motores 1 e 2, ligados às portas M1 e M2
//no Arduino Motor Shield
AF_DCMotor motor1(1); //Define o motor1 ligado a conexao M1
AF_DCMotor motor2(2); //Define o motor2 ligado a conexao M2

//Declaração de variáveis globais
int velocidade, estdSensorDt, estdSensorEq, estdSensorMd;
int estdSensorFr, estdSensorLt, estdRobo = 1;

//Construção das funções
void moverFrente(int v)
```

```
{
  //Configura velocidade dos motores
  motor1.setSpeed(v);
  motor2.setSpeed(v);
  //Aciona motores para frente
  motor1.run(FORWARD);
  motor2.run(FORWARD);
}
void virarEsquerda(int v)
{
  //Configura velocidade dos motores
  motor1.setSpeed(v/4);
  motor2.setSpeed(v);
  //Aciona motores para frente
  motor1.run(FORWARD);
  motor2.run(FORWARD);
}
void virarDireita(int v)
{
  //Configura velocidade dos motores
  motor1.setSpeed(v);
  motor2.setSpeed(v/4);
  //Aciona motores para frente
  motor1.run(FORWARD);
  motor2.run(FORWARD);
}
void pararRobo()
{
  //Parar os motores
  motor1.run(RELEASE);
  motor2.run(RELEASE);
}
void tocarBip(int t)
{
  int i;
  for(i=0; i<t; i++)
```

```
  {
    digitalWrite(A0, HIGH);
    delay(500);
    digitalWrite(A0, LOW);
    delay(500);
  }
}
void girarDireita()
{
  motor2.run(BACKWARD);
  delay(1000);
}
void girarEsquerda()
{
  motor1.run(BACKWARD);
  delay(1000);
}

void setup() {
//Configrar buzzer em A0 - Saída
pinMode(A0, OUTPUT);
//Configurar sensor de contraste direito em A1 - Entrada
pinMode(A1, INPUT);
//Configurar sensor de contraste esquerdo em A2 - Entrada
pinMode(A2, INPUT);
//Configurar sensor de contraste meio em A3 - Entrada
pinMode(A3, INPUT);
//Configurar sensor de obstáculo frontal em A4 - Entrada
pinMode(A4, INPUT);
//Configurar sensor de contraste lateral em A5 - Entrada
pinMode(A5, INPUT);

}

void loop() {
  while(estdRobo == 1)
```

```
  {
    velocidade = 250;
    estdSensorDt = digitalRead(A1);
    estdSensorEq = digitalRead(A2);
    estdSensorMd = digitalRead(A3);
    //Se encontrar a faixa transversal
    if ((estdSensorMd == 0) && ((estdSensorDt == 0) && (estdSensorEq ==
0)))
    {
      pararRobo();
      tocarBip(5);
      estdRobo = 0;
    }
    else //Caso contrário
    {
      //Se o sensor do meio detectar faixa clara
      if (estdSensorMd == 0)
      {
        moverFrente(velocidade);
      }
      else //Se deixar de detectar
      {
        //Se sensor direito detectar
        if (estdSensorDt == 0)
        {
          virarDireita(velocidade);
        }
        else
        {
          //Se sensor esquerdo detectar
          if (estdSensorEq == 0)
          {
            virarEsquerda(velocidade);
          }
          else //Se nenhum sensor detectar a faixa
          {
            //Programação sem detecção da faixa
```

```
            velocidade = velocidade / 2; //velocidade reduzida
            moverFrente(velocidade);

            estdSensorFr = digitalRead(A4);
            //Se encontrar obstáculo à frente
            if (estdSensorFr == 0)
            {
              estdSensorLt = digitalRead(A5);
              //Se encontrar obstáculo à direita
              if (estdSensorLt == 0)
              {
                girarEsquerda();
              }
              else //Se direita livre
              {
                girarDireita();
              }
            }
          }
        }
      }
    }
  }
}
```

Propostas de Exercícios

1) Criar um robô móvel que siga as paredes de uma sala e corrija o percurso, caso perca a detecção da parede, utilizando funções.
2) Recriar os códigos dos exercícios anteriores utilizando funções.

CONSIDERAÇÕES FINAIS

Por fim, finalizamos o trabalho com algumas considerações acerca de toda a sua construção, desde a concepção das ideias juntando os problemas de aprendizagem trazidos pelos alunos desde o ensino fundamental e a observação do crescente número de projetos de robótica nas escolas particulares até a finalização do manual.

Vislumbramos, no início, a possibilidade dos alunos que não conseguem abstrair os conceitos de programação de computadores testar seus códigos num ambiente concreto, diferente do virtual. Dessa forma, espera-se que ao término do trabalho com a robótica pedagógica, haja uma melhoria no rendimento escolar e na capacidade de abstração e resolução de problemas desses alunos, conforme Papert propõe com o Construcionismo. Essas possibilidades de melhorias de rendimento escolar serão testadas e avaliadas em estudos futuros.

Em princípio, enxergava-se a oportunidade de aplicação em disciplinas formativas de cursos ligados à computação e informática, mas espera-se que esse trabalho possa ser de fácil intercâmbio para qualquer outra disciplina que queira utilizar-se dos recursos da robótica pedagógica, alterando-se e adaptando as situações-problema propostas. Há, ainda, a possibilidade da construção interdisciplinar de uma oficina de robótica pedagógica.

Sobre o manual propriamente dito, seus códigos foram criados pensando sempre na relação entre professor e aluno, tendo o professor como mediador do conhecimento junto a grupos heterogêneos de alunos, de modo que haja uma construção do conhecimento de forma discutida e elaborada. Os códigos não podem ser vistos como peças centrais do conhecimento, mas sim como uma sugestão, um ponto de partida para a construção de novas formas de fazer o robô móvel se mover e, por consequência, aprender os conceitos básicos da lógica de programação.

Todos os códigos e o robô móvel podem e devem sofrer melhorias no processo de construção do conhecimento da lógica de programação. As propostas de exercícios devem ser discutidas entre o grupo todo. Muitas vezes, o andamento das aulas pode tomar um rumo diferente e é importante que os alunos participem das sugestões de novas propostas para o robô móvel. Uma das melhorias que podem contemplar o robô móvel é a troca da roda boba por um rodízio bola de metal, o que aumentaria a precisão dos movimentos.

Esse robô foi pensado para facilitar o primeiro contato, porém, há limitação de portas de comunicação por causa do Arduino Motor *Shield*. Para aumentar a capacidade de comunicação do robô com o mundo externo, podemos substituir o *shield* por uma controladora Ponte H, para controlar os motores, mas isso demandaria uma outra fonte de energia. Poderíamos, também, alterar a placa Arduino utilizada para a Arduino Mega, que possui mais portas de comunicação, mas é mais cara que a placa Arduino UNO utilizada. Poderíamos pensar numa evolução do robô móvel, trabalhando com um chassi 4 X 4 ou um chassi de esteiras. Além disso, poderíamos utilizar um sensor ultrassônico ligado a um servo motor no lugar dos sensores de obstáculos, mas, do meu ponto de vista, poderia ser uma continuidade de projeto, pois demandaria mais conhecimento da programação Arduino e inclusão de outras bibliotecas, o que foi descartado no início do projeto por se tratar do primeiro contato do professor e dos alunos com a robótica móvel e robótica pedagógica.

Existem muitas outras aplicações que poderíamos utilizar no decorrer do projeto, como um robô para competições de sumô, um robô por controle remoto e a utilização e sucatas eletrônicas, ou mesmo de materiais, para a construção do robô. É importante incluir conceitos que não somente os de programação – poderíamos incluir um *display* para que nosso robô se comunique com o mundo exterior. Enfim, criar várias outras aplicações no projeto de robótica pedagógica. Poderíamos, inclusive, basear as tarefas do robô ou sua construção com a inclusão de disciplinas propedêuticas. As aplicações possíveis são praticamente infinitas e, por isso mesmo, quanto mais discutirmos sobre o assunto, mais possibilidades se abrirão para as possíveis aplicações da robótica pedagógica.

REFERÊNCIAS

ARDUINO. **Language Reference**. Disponível em: <https://www.arduino.cc/>. Acesso em: 28 de dezembro de 2017.

ARDUINO E CIA. **Blog Arduino & Cia**. Disponível em: < https://www.arduinoecia.com.br>. Acesso em 28 de dezembro de 2017.

ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi. **Fundamentos da Programação de Computadores**: Algoritmos, Pascal, C/C++ e Java – 3ª edição – São Paulo: Pearson Prentice Hall, 2012

ASIMOV, Isáac. **Eu, robô.** São Paulo: Ed. Aleph, 2014.

BERTOLETI, Pedro; THOMSEN, Adilson; Et Al. **Blog Filipeflop**. Disponível em: < https://www.filipeflop.com/blog/>. Acesso em: 28 de dezembro de 2017.

BRASIL. **Lei no 11.892, de 29 de dezembro de 2008**. Institui a Rede Federal de Educação Profissional, Científica e Tecnológica, cria os Institutos Federais de Educação, Ciência e Tecnologia e dá outras providências. Diário Oficial da União, Brasília, 30 dez. 2008, Seção 1, p. 1.

CAMPOS, Flávio Rodrigues. **Paulo Freire e Seymour Papert:** educação, tecnologias e análise do discurso. Curitiba: Editora CRV, 2013

CURCIO, Christina Paula de Camargo. A robótica educacional como ferramenta facilitadora do processo de ensino e aprendizagem. **Revista Tecnologia Educacional ABT Científica do IFSC**, ISSN 0102-5503, a. XXXIX, n. 190, p. 45, julho/setembro de 2010.

D'ABREU, João Vilhete Viegas. Desenvolvimento de ambiente de aprendizagem baseados no uso de dispositivos robóticos. **Anais do X Simpósio Brasileiro de Informática na Educação – SBIE99**. Curitiba: Universidade Federal do Paraná (UFPR), 1999.

DOS SANTOS, Tatiana Nilson; POZZEBON, Eliane; FRIGO, Luciana Bolan. **A utilização de robótica nas disciplinas da educação básica.** Revista Técnico Científica do IFSC, v. 1, n. 5, p. 616, 2013.

GIRALT, Georges. **A Robótica**. Lisboa: Instituto Piaget, 1997.

IFSP - INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO. **Projeto Pedagógico do Curso Técnico em Informática Integrado ao Ensino Médio**: Campus Araraquara. Aprovado pela Resolução 90, de 29 de setembro de 2015. São Paulo: IFSP, 2015.

LEAL, Cristianni Antunes. **Vamos brincar de quê?:** Os jogos cooperativos no ensino de ciências. Dissertação (Mestrado Profissional em Ciências) Intituto Federal de Educação, Ciência e Tecnologia do Rio de Janeiro. Nilópolis, 2013

MATARIČ, Maja J. **Introdução à Robótica**. Tradução de Humberto Ferasoli Filho, José Reinaldo Silva, Silas Franco dos Reis Alves. São Paulo: Editora Unesp/Blucher, 2014.

MCROBERTS, Michael. **Arduino básico**. Tradução de Rafael Zanolli. São Paulo: Novatec, 2011.

MILL, Daniel; CÉSAR, Danilo Rodrigues. Estudos sobre dispositivos robóticos na educação: sobre a exploração do fascínio humano pela robótica no ensino-aprendizagem. In: MILL, Daniel. (org.). **Escritos sobre educação**: Desafios e possibilidades para ensinar e aprender com as tecnologias emergentes. São Paulo: Paulus, 2013

MIZRAHI, Victorine Viviane. **Treinamento em linguagem C**. 2ª Edição. São Paulo: Pearson Prentice Hall, 2008.

PACHECO, Eliézer (org.). **Institutos Federais - uma revolução na educação profissional e tecnológica**. São Paulo: Editora Moderna, 2011

PAPERT, Seymour. **A máquina das crianças:** repensando a escola na era da informática. Tradução: Sandra Costa. Porto Alegre: Artes Médicas, 1994.

RIBEIRO, Luis Roberto de Camargo; OLIVEIRA, Marcia Rozenfeld Gomes, MILL, Daniel. TECNOLOGIA E EDUCAÇÃO: aportes para a discussão sobre a docência na era digital. In: MILL, Daniel. (org.). **Escritos sobre educação:** Desafios e possibilidades para ensinar e aprender com as tecnologias emergentes. São Paulo: Paulus, 2013

SILVA, Alzira Ferreira da; Et al. Utilização da teoria de vygotsky em robótica educativa. **IX CONGRESO IBEROAMERICANO DE INFORMATICA EDUCATIVA RIBIE**. Caracas: Venezuela, 2008.