

UNIVERSIDADE FEDERAL DO TOCANTINS
CAMPUS UNIVERSITÁRIO DE ARAGUAÍNA
CURSO DE LICENCIATURA EM MATEMÁTICA

GABRIEL COSTA MOREIRA

**O JABUTI: UMA PROPOSTA DE ATUALIZAÇÃO ACESSÍVEL DA TARTARUGA
DE PAPERT**

ARAGUAÍNA
2019

GABRIEL COSTA MOREIRA

**O JABUTI: UMA PROPOSTA DE ATUALIZAÇÃO ACESSÍVEL DA TARTARUGA
DE PAPERT**

Trabalho apresentado a Universidade Federal do Tocantins, no Curso de Licenciatura em Matemática como requisito parcial para obtenção do título de licenciado em Matemática.

Orientador: Prof. Dr. Deive Alves

ARAGUAÍNA
2019

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da Universidade Federal do Tocantins

- M838j Moreira, Gabriel Costa.
 O JABUTI: UMA PROPOSTA DE ATUALIZAÇÃO ACESSÍVEL DA
 TARTARUGA DE PAPERT . / Gabriel Costa Moreira. – Araguaína, TO,
 2019.
 80 f.

 Monografia Graduação - Universidade Federal do Tocantins –
 Câmpus Universitário de Araguaína - Curso de Matemática, 2019.
 Orientador: Deive Alves

 1. Robótica educacional. 2. Construcionismo. 3. Ambiente de
 aprendizagem. 4. Assimilação conceitual. I. Título

CDD 510

TODOS OS DIREITOS RESERVADOS – A reprodução total ou parcial, de qualquer forma ou por qualquer meio deste documento é autorizado desde que citada a fonte. A violação dos direitos do autor (Lei nº 9.610/98) é crime estabelecido pelo artigo 184 do Código Penal.

Elaborado pelo sistema de geração automática de ficha catalográfica da UFT com os dados fornecidos pelo(a) autor(a).

GABRIEL COSTA MOREIRA

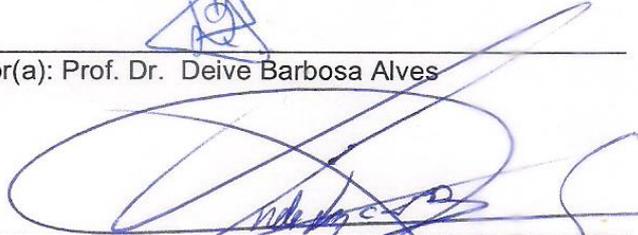
O JABUTI: UMA PROPOSTA DE ATUALIZAÇÃO ACESSÍVEL DA TARTARUGA DE
PAPERT

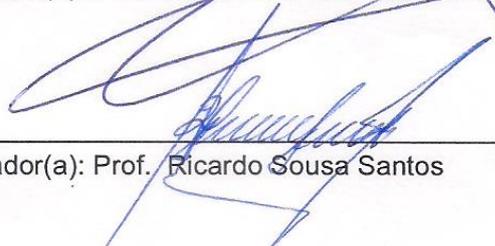
Monografia foi avaliada e apresentada à
UFT – Universidade Federal do Tocantins
– Campus Universitário de Araguaína,
Curso de Matemática para a obtenção do
título de Licenciado em Matemática e
aprovada em sua forma final pelo
orientador e pela Banca Examinadora.

Aprovada em: 18/ 12/ 2019

Banca examinadora:


Orientador(a): Prof. Dr. Deive Barbosa Alves


Avaliador(a): Prof. Msc. André Luiz Ortiz da Silva


Avaliador(a): Prof. Ricardo Sousa Santos

Dedico este trabalho à minha família e amigos, em especial ao meu falecido Pai José Bonifácio de Sousa Moreira, eterno “especialista em motores elétricos”, que enquanto vivo me guiou e abriu caminho para meu ingresso na Universidade, e a minha mãe Maria Costa Ribeiro, que com muito esforço me possibilitou concluir o curso de Licenciatura em matemática. Essas duas pessoas maravilhosas me ensinaram o valor do esforço, dedicação e humildade para contornar as adversidades proporcionadas pela vida.

“Às vezes, quando você inova, você comete erros. É melhor admiti-los rapidamente, e seguir em frente para melhorar suas outras inovações”.
Steve Jobs (1955-2011).

“A mente que se abre a uma nova ideia, jamais voltará ao seu tamanho original”.
Albert Einstein (1879-1955).

RESUMO

O ambiente de aprendizagem robótica educacional, baseado no construcionismo e filosofia Logo de Papert, está sendo a aposta atual de grande parte dos educadores para favorecer uma aprendizagem ativa e multidisciplinar para assimilação conceitual e depuração do processo mental em construções concretas ou mentais. Porém, esse ambiente está restrito a escolas com boa infraestrutura ou com profissionais sobressalentes da área. Nesse trabalho estou preocupado se é possível atualizar a tartaruga mecânica de Papert para ensinar conceitos formais de geometria plana? Com isso, o objetivo geral deste trabalho é apresentar um projeto robótico em desenvolvimento, chamado de Jabuti Geométrico, acessível as escolas brasileiras em geral, para o ensino de geometria. Trata-se de uma atualização da tartaruga de Papert, para superação de algumas limitações identificadas que dificultavam as construções e manipulações da máquina. A metodologia adotada fundamentou-se em estudo teórico e estudo de caso para descrever a construção do projeto robótico com um eixo a mais para movimentar a caneta. A análise e comparação do funcionamento do robô mostrou que as limitações foram superadas, além de obter flexibilidade em algumas construções geométricas. Porém deve-se continuar o aperfeiçoamento do projeto em algumas peças, pois, em decorrência de falhas mecânicas apresenta imprecisões que podem ser superadas com calibrações futuras. Embora apresente as mesmas propriedades da tartaruga de Papert muito bem fundamentada, e com vastas pesquisas de sua utilização no ensino, essa ferramenta necessita de um estudo de sua aplicação em sala de aula, para validar a eficiências das atualizações para assimilações conceituais no ensino e facilidade no controle e manipulação dos objetos geométricos.

Palavras-chave: Robótica educacional; Construcionismo; Ambiente de aprendizagem; Assimilação conceitual.

ABSTRACT

The robotic educational learning environment, based on the constructionism and philosophy of Papert's Logo, is currently the bet of most educators to favor active and multidisciplinary learning for conceptual assimilation and purification of the mental process into concrete or mental constructions. However, this environment is restricted to schools with good infrastructure or with spare professionals in the area. In this paper I am concerned if it is possible to upgrade Papert's mechanical turtle to teach formal concepts of flat geometry? Thus, the general objective of this paper is to present a robotic project under development, called Geometric Jabuti, accessible to Brazilian schools in general, for the teaching of geometry. This is an update of the Papert tortoise to overcome some identified limitations that made the construction and manipulation of the machine difficult. The methodology adopted was based on theoretical study and case study to describe the construction of the robotic project with one more axis to move the pen. The analysis and comparison of the robot's operation showed that the limitations were overcome, besides obtaining flexibility in some geometric constructions. However, the design improvement must be continued in some parts, because due to mechanical failures it presents inaccuracies that can be overcome with future calibrations. While presenting the same properties as the well-grounded Papert tortoise, and with extensive research on its use in teaching, this tool needs a study of its classroom application to validate the efficiencies of upgrades to conceptual assimilation in teaching and ease in the control and manipulation of geometric objects.

Keywords: Educational robotics; Constructionism; Learning environment; Conceptual assimilation.

LISTA DE ILUSTRAÇÕES

| | |
|--|----|
| Figura 1 – Tartaruga de solo de Papert..... | 17 |
| Figura 2 – Tartaruga de Callahan..... | 18 |
| Figura 3 – Tartaruga Irving..... | 19 |
| Figura 4 - Interface do programa Superlogo 3.0..... | 20 |
| Figura 5 – Lista de comandos básicos do Superlogo 3.0..... | 21 |
| Figura 6 - Construção de triângulo no Logo..... | 22 |
| Figura 7 – Construção do quadrado no Logo..... | 23 |
| Figura 8 – Construção hexágono no Logo..... | 23 |
| Figura 9 – Circunferência da tartaruga..... | 24 |
| Figura 10 – Circunferência da tartaruga ampliada..... | 25 |
| Figura 11 – Tartaruga de Grey Walter..... | 29 |
| Figura 12 – Veiculos de Braitenberg..... | 30 |
| Figura 13 – Exemplo de motor simples..... | 34 |
| Figura 14 – Motor CC torque zero..... | 34 |
| Figura 15 – Motor CC arranjo de bobinas..... | 35 |
| Figura 16 – Motor de passo simples..... | 36 |
| Figura 17 – Sequência de ativações de bobinas..... | 36 |
| Figura 18 – Tamanho do passo do motor..... | 37 |
| Figura 19 – Grau de liberdade de um objeto no espaço..... | 38 |
| Figura 20 – Comparação do GDL de um helicóptero e um carro..... | 39 |
| Figura 21 – Motor 28BYJ-48 e Micro servo SG90 Tower Pro..... | 41 |
| Figura 22 – Rotação eixo externo do motor e sistema de redução..... | 41 |
| Figura 23 – Driver ULN2003..... | 43 |
| Figura 24 – Componentes de um Arduino Uno..... | 44 |
| Figura 25 – Site do Arduino..... | 47 |
| Figura 26 – Interface do IDE Arduino..... | 49 |
| Figura 27 – Barra de Ferramentas..... | 49 |
| Figura 28 – Tartaruga atualizada de Papert e plotadora caseira 2D..... | 51 |
| Figura 29 – Área de trabalho pesado..... | 53 |
| Figura 30 – Área para montagem, testes e programação..... | 54 |
| Figura 31 – Eixo da caneta inicial..... | 55 |
| Figura 32 – Protótipo inicial..... | 56 |
| Figura 33 – Chassi bandeja de impressora..... | 56 |
| Figura 34 – Projeto com funcionamento errado..... | 57 |
| Figura 35 – Suporte para eixo da caneta versão 3..... | 57 |
| Figura 36 – Pipira versão estendida..... | 58 |
| Figura 37 – Chassi versão 3..... | 58 |
| Figura 38 – Jabuti Geométrico versão 3..... | 59 |
| Figura 39 – Chassi final sem modificação..... | 59 |
| Figura 40 – Atuadores das rodas do Jabuti Geométrico..... | 60 |
| Figura 41 – Atuadores ativos e passivos do eixo da caneta..... | 61 |
| Figura 42 – Rodas do Jabuti..... | 61 |
| Figura 43 – O eixo da caneta..... | 62 |
| Figura 44 – Eixo Z da caneta..... | 62 |
| Figura 45 – Mini protoboard..... | 63 |
| Figura 46 – Estrutura mecânica montada..... | 64 |
| Figura 47 – Circuito atuadores eixo da caneta..... | 64 |

| | |
|---|----|
| Figura 48 – Circuito atuadores rodas | 65 |
| Figura 49 – Ligações realizadas no projeto..... | 66 |
| Figura 50 – Jabuti geométrico versão final..... | 66 |
| Figura 51 – Circunferência pelo Jabuti..... | 73 |
| Figura 52 – Paralelepípedo do Jabuti..... | 74 |

SUMÁRIO

| | | |
|-------|---|----|
| 1 | INTRODUÇÃO | 10 |
| 2 | FUNDAMENTAÇÃO TEÓRICA | 13 |
| 2.1 | O Construcionismo | 13 |
| 2.2 | A filosofia Logo | 14 |
| 2.3 | Robótica Educacional | 16 |
| 3 | A GEOMETRIA DA TARTARUGA DE PAPERT | 18 |
| 3.1 | Tartaruga mecânica de Papert | 18 |
| 3.2 | Geometria da Tartaruga | 19 |
| 3.3 | Apresentação básica do Superlogo 3.0 | 20 |
| 3.4 | Breve análise da geometria da tartaruga | 21 |
| 3.4.1 | A limitação conceitual da geometria da tartaruga | 25 |
| 4 | INTRODUÇÃO A ROBÓTICA | 27 |
| 4.1 | Definição de robô | 27 |
| 4.1.1 | A tartaruga de Grey Walter | 28 |
| 4.1.2 | Veículos de Braitenberg..... | 29 |
| 4.2 | Componentes de um robô | 31 |
| 4.2.1 | Corporalidade | 31 |
| 4.2.2 | Componentes de ação..... | 32 |
| 4.2.3 | Autonomia..... | 32 |
| 4.3 | Efetadores e Atuadores | 33 |
| 4.3.1 | Tipos de atuadores | 33 |
| 4.3.2 | Motores CC..... | 34 |
| 4.3.3 | Motores de Passo | 35 |
| 4.3.4 | Servo motores..... | 37 |
| 4.4 | Graus de liberdade | 37 |
| 5 | DESCRIÇÃO DOS ATUADORES E CONTROLADORES DO PROJETO | 41 |
| 5.1 | Motor de passo 28BYJ-48 | 41 |
| 5.2 | Micro servo motor sg90 | 42 |
| 5.3 | Controladores secundários | 43 |
| 5.4 | Controlador principal: Arduino | 43 |
| 5.5 | O IDE do Arduino | 47 |
| 5.5.1 | Instalação no Windows 10 | 47 |
| 5.5.2 | Interface do IDE | 48 |
| 6 | CONSTRUÇÃO DO JABUTI GEOMÉTRICO | 51 |

| | | |
|-------|---|-----------|
| 6.1 | Lista de materiais | 52 |
| 6.2 | As Ferramentas e bancada de trabalho | 53 |
| 6.3 | O Chassi do Jabuti | 55 |
| 6.4 | Os Atuadores | 60 |
| 6.4.1 | Atuadores das rodas..... | 60 |
| 6.4.2 | Atuadores do eixo da caneta | 60 |
| 6.5 | Os Efetadores | 61 |
| 6.5.1 | Atuador e Efetador da caneta | 62 |
| 6.6 | Os Controladores | 63 |
| 6.7 | Montagem do Jabuti Geométrico | 63 |
| 7 | PROGRAMAÇÃO DO JABUTI GEOMÉTRICO | 67 |
| 8 | ANÁLISE DO JABUTI GEOMÉTRICO | 72 |
| 8.1 | Jabuti Geométrico e suas características | 72 |
| 8.2 | Distinções entre a Tartaruga de Papert e o jabuti Geométrico | 72 |
| 8.2.1 | A circunferência do Jabuti Geométrico | 73 |
| 8.2.2 | Outras ampliações | 74 |
| 9 | CONSIDERAÇÕES FINAIS | 76 |
| | REFERÊNCIAS | 78 |

1 INTRODUÇÃO

Como um entusiasta das novas tecnologias e graduando do curso de licenciatura em matemática na Universidade Federal do Tocantins – UFT, campus Araguaína, me vi motivado a pesquisar sobre a aplicação das mesmas no ensino de matemática. Minha pesquisa teórica inicial me deixou preocupado quanto ao quadro da utilização do computador na educação, pois, enquanto professores e pesquisadores concordam que o paradigma¹ educacional instrucionista² já não é suficiente para atender as exigências de formação do cidadão na sociedade da informação³, onde a função do profissional mudou de um simples seguidor de instrução para alguém capaz de lidar com situações diversas na realidade tecnológica vigente, os mesmos parecem não encontrar o caminho para o desenvolvimento do paradigma educacional contemporâneo, da educação 4.0⁴, necessário para a realidade atual e a iminente que será provocada pela indústria 4.0⁵.

Atualmente para mudar esse quadro ineficiente⁶ do uso das novas tecnologias na educação, está sendo desenvolvida uma série de métodos de ensino baseados na inserção da robótica no contexto escolar, denominada robótica educacional. Surge cada vez mais artigos sobre o assunto, relatando suas vantagens e mostrando o quanto esse ambiente de aprendizagem está se tornando o alicerce do paradigma da educação 4.0.

Porém existem alguns obstáculos para a efetivação desse ambiente nas escolas brasileiras. Devido a robótica educacional ser comumente trabalhada com Kits da LEGO® Education e os mesmos apresentarem alto custo, tornam-se inacessíveis

¹ Thomas Kuhn introduziu a noção de paradigma na sua obra *The Structure of Scientific Revolution*, para indicar novos conjuntos de conceitos e de métodos, além de práticas, artefatos culturais e valores que caracterizam um determinado período na área da ciência. Valente (1999, p.32)

² O instrucionismo é uma corrente pedagógica baseada na teoria didática tecnicista sustentada pela teoria da aprendizagem comportamentalista (behaviorista). Valoriza-se a estrutura curricular, é estabelecida uma aprendizagem mecânica de assimilação e repetição de informações. Pimentel (2012, p.on-line)

³ A expressão “sociedade da informação” passou a ser utilizada, nos últimos anos desse século, como substituto para o conceito complexo de “sociedade pós-industrial” e como forma de transmitir o conteúdo específico do “novo paradigma técnico-econômico”. Werthein (2000, p.71)

⁴ É uma proposta de educação diferenciada, que preza por colocar os estudantes, professores e diretores em conexão com as novas tecnologias. Ela prioriza a prática e experimentação dos alunos.

⁵ Conhecida também como a quarta revolução industrial, se trata da evolução e integração profunda de tecnologias na indústria. Se caracteriza, por um conjunto de tecnologias que permitem a fusão do mundo físico, digital e biológico.

⁶ Ineficiente no sentido de não proporcionar a revolução do ensino necessária nos dias atuais. O termo não é usado no sentido de que não há vantagens em usar o computador como instrutor informatizado.

para a maioria das instituições públicas de ensino. O acesso a esse ambiente, com raras exceções, está sendo realizado apenas pelas escolas particulares. Para reverter este quadro alguns professores da rede pública vêm tentando construir o ambiente de aprendizagem com sucatas de aparelhos eletroeletrônicos, e o estão fazendo com excelência, tendo reconhecimento internacional, como a professora que ensina robótica, na periferia de São Paulo, com sucatas. Mas, o ambiente ainda é restrito a alguns com recursos ou sobressalentes.

Ainda existe o problema da forma como a robótica é aplicada, embora estudiosos (Seymour Papert (1985); Valente (1999); entre outros) defendam que o ambiente criado pela robótica contribui para que o aluno desenvolva um pensamento reflexivo e depurativo, sobre as disciplinas curriculares. Seu uso prático está sendo limitado a uma atividade extracurricular em alguns casos ou extraclasse em outras.

Ao ler os escritos de Seymour Papert, educador matemático, me deparei com sua posição futurista do uso do computador e da robótica na educação. Fiquei maravilhado com sua visão sobre a importância dessas máquinas no desenvolvimento cognitivo do aluno. Ao conhecer a Geometria da tartaruga criada por esse educador, me pareceu incrível a ideia de proporcionar possibilidades para a assimilação dos conceitos, e não o simples decorar de informações, por meio das novas tecnologias. Porém, parei um pouco minha admiração para uma avaliação crítica, como resultado encontrei uma limitação conceitual na construção da circunferência na tartaruga de Papert, decorrentes da época em que fora desenvolvida. Em minha visão, dificulta a conceitualização da Geometria Euclidiana, e vi a necessidade de atualizar a tartaruga mecânica de Papert, para que possa ser usada nas aulas de Geometria plana na assimilação de conceitos, avaliação e depuração de pensamentos pelo aluno.

Dessa forma, como amante confesso das novas tecnologias, decidi fazer o meu trabalho de conclusão de curso focado no desenvolvimento de uma ferramenta robótica que possa ser acessível pelos professores em geral da rede pública e que possa ser usado nas aulas curriculares do ensino de matemática, especificamente nas aulas de Geometria plana.

Nesse trabalho estou preocupado se **é possível atualizar a tartaruga mecânica de Papert para ensinar conceitos formais de geometria plana?** Assim pretendo relatar o desenvolvimento de um robô, que chamei de Jabuti Geométrico, com alguns componentes eletrônicos e peças mecânicas retiradas de sucatas. O projeto robótico apresenta, um eixo a mais para movimentar a caneta. Para essa construção utilizei a

placa de prototipagem Arduino⁷ como controlador do hardware e Arduino IDE⁸ como software de programação. Quanto a linguagem de programação o IDE do Arduino utiliza a C++ simplificada, mas estarei adaptando parte do código⁹ para a lógica da linguagem LOGO desenvolvida por Seymour Papert.

Ao me dispor a realizar esse trabalho notei que as informações sobre a construção de robôs são bastante dispersas, e a maioria dos fóruns mencionam sobre o assunto como se o leitor fosse um expert ou estivesse por dentro da área de robótica. Perdi vastas horas tentando entender o que hoje classifico como conceitos simples da robótica. Por isso nesse trabalho utilizo uma linguagem simples para que o leitor iniciante possa reproduzir este projeto robótico, e com a experiência faça suas próprias modificações.

No próximo Capítulo faço uma breve apresentação dos escritos de Seymour Papert sobre a filosofia construcionista e Logo. Também defino o ambiente de aprendizagem robótica educacional baseado nas duas filosofias criadas pelo educador matemático Sul africano. Essas linhas teóricas justificam a construção do projeto robótico Jabuti Geométrico.

No Capítulo 3 descrevo brevemente a Geometria da Tartaruga e aproveito para olhar analiticamente a mesma. O Capítulo 4 trata de definições de conceitos introdutórios da robótica, apresentando os componentes pertinentes ao projeto, por meio do livro Introdução à Robótica de Mataric (2014).

Estruturei o Capítulo 5 para ampliar as informações sobre os principais componentes do projeto robótico, no intuito de fornecer informações importante sobre os mesmos. No Capítulo 6 descrevo a construção do projeto e relato as dificuldades e contratemplos encontrados no caminho. No Capítulo 7 abordo a programação do robô. O Capítulo 8 é para a conceitualização da geometria possível com o Jabuti Geométrico e comparação com a Geometria da Tartaruga de Papert para validar as ampliações conceituais. No último Capítulo faço as considerações finais sobre o projeto promessa de atualizações e continuação do estudo.

⁷ Arduino é uma plataforma de prototipagem eletrônica de hardware livre e de placa única, projetada com um microcontrolador Atmel AVR com suporte de entrada/saída embutido, uma linguagem de programação padrão, a qual tem origem em Wiring, e é essencialmente C/C++. Arduino (2019)

⁸ O Arduino Integrated Development Environment (IDE) é um aplicativo de plataforma cruzada (para Windows, macOS, Linux) escrito em funções de C e C ++. É usado para escrever e fazer upload de programas em placas compatíveis com Arduino, mas também, com a ajuda de núcleos de terceiros, outras placas de desenvolvimento de fornecedores. Arduino IDE (2019)

⁹ Código é um termo usado pelos programadores para fazer referência a uma seção de um programa ou, genericamente, ao que é escrito quando um programa é criado. Monk (2014, p.13)

2 FUNDAMENTAÇÃO TEÓRICA

Este trabalho é baseado no ambiente de aprendizagem robótica educacional fundamentado no Construcionismo de Papert e na filosofia Logo. Desse modo pretendo nesse capítulo abordar primeiramente sobre o Construcionismo e sobre a filosofia Logo utilizando os escritos do educador matemático Seymour Papert e por fim irei apresentar uma definição do ambiente de aprendizagem robótica educacional.

2.1 O Construcionismo

Antes de mais nada devo definir o que é Instrucionismo, onde o sufixo *ismo* é um marcador do abstrato Papert (1994, p.124), a palavra não é tomada apenas como um ato de ensinar, mas sim como uma ideologia que expressa a crença de que o caminho para melhorar a aprendizagem está no aperfeiçoamento da forma da instrução. O Construcionismo é assim uma filosofia que nega essa ideologia. Não se trata da proibição do ensino, mas da minimização do mesmo ao ponto de ter um ambiente que proporcione a maior aprendizagem no mínimo possível de ensino. Essa filosofia do educador matemático parte da teoria construtivista de Piaget, a qual defende que as crianças aprenderão melhor descobrindo por si mesmas o conhecimento específico que lhes faz sentido, que lhes pareça útil para determinada situação em que é submetida. A chave principal é que se desenvolva conhecimento sobre a arte de aprender, denominada por ele Matética. Quanto a isso Papert (1994, p.126) afirma que

No contexto de uma sociedade dominada pela escola, o princípio mais importante da Matética pode ser o incitamento à revolta contra a sabedoria aceita que vem de saber que você pode aprender sem ser ensinado e, com frequência, aprender melhor quando é menos ensinado.

Carregando assim as ideias construtivistas de Piaget em que o conhecimento não pode simplesmente ser transferido para outra pessoa. Mesmo fazendo isso ao olhar o processo mental da mesma, iríamos perceber que ela estaria dando ressignificação pessoal a tudo o que está aprendendo. Nas palavras de Papert (1994, p.127) a sua filosofia fica definida sinteticamente

Assim, o construcionismo, minha reconstrução pessoal do Construtivismo, apresenta como principal característica o fato de que examina mais de perto do que outros – *ismos* educacionais a idéia da construção mental. Ele atribui especial importância ao papel das construções no mundo como um apoio para o que ocorreu na cabeça, tornando-se, desse modo menos uma doutrina puramente mentalista. Também leva mais a sério a idéia de construir na cabeça reconhecendo mais de um tipo de construção (algumas delas tão

afastadas de construções simples como cultivar um jardim) e formulando perguntas a respeito dos métodos e materiais usados.

Com essa definição fica evidente que essa linha teórica carrega, tanto, a conotação de conjunto de construção no sentido literal quanto mental, partindo dos kits Lego, até os micromundos como o da linguagem de programação Logo.

Assim o Construcionismo é uma filosofia que diz respeito ao desenvolvimento do conhecimento por meio de uma construção mental ou no mundo concreto, desenvolvido com o auxílio do computador, e que seja de interesse de quem o produz. Seja um texto, uma imagem, um mapa conceitual, uma apresentação em slides - deve ter vínculo com a realidade da pessoa ou com o local onde será produzido e utilizado. Esse ambiente implica numa interação aluno-objeto, mediada por uma linguagem de programação, como é o caso da Logo.

2.2 A filosofia Logo

Trata-se de uma filosofia educacional que desenvolve a ideia de que as crianças podem aprender a usar e se comunicar com o computador habilmente e isso pode mudar a maneira como elas aprendem sobre as outras coisas. No intuito de criar um ambiente de aprendizagem construcionista Papert e a equipe de colaboradores do Massachusetts Institute of Technology – MIT, desenvolveram a linguagem de programação Logo. As palavras de Papert (1994, p.193) sintetizam o objetivo

Meus objetivos são educacionais, não a simples compreensão. Assim, na minha própria reflexão, coloquei uma ênfase maior em duas dimensões implícitas, mas não elaboradas no trabalho de Piaget: um interesse em estruturas intelectuais que poderiam se desenvolver, em oposição às que realmente se desenvolvem presentemente na criança, e o planejamento de ambientes educacionais que estivessem em consonância com aquelas estruturas.

A preocupação está em modificar as frágeis estruturas mentais que se formavam na mente da criança, pelo contato com ideias abstratas nos ambientes tradicionais de ensino, por uma estrutura que dê acesso a forma concreta das ideias aprendidas além de desenvolver um ambiente de aprendizagem que ampare essa nova estrutura.

Papert e seus colaboradores viam a linguagem de programação como uma maneira de dar acesso à criança à sua linha de pensamento e dessa forma ela poderia refletir sobre os próprios processos mentais. Trata-se da criação da Matelândia, denominada como o lugar especial para aprender a linguagem matemática. A analogia é bem simples, se você quiser aprender um idioma de outro país você pode ir a um

cursinho ou ir ao país predominante do idioma escolhido, embora a pessoa que escolheu aprender pelo cursinho possa se dedicar o máximo possível, ela nunca irá assimilar o idioma tão bem quanto a que escolheu ir ao país de origem do mesmo. Na falta de um país que se comunique com a linguagem matemática, o Logo¹⁰ se torna essa Matelândia.

A proposta trabalha ainda a ideia da criação de micromundos, que servem para descrever certos ambientes de aprendizagens exploratórios próprios da criança, “No trabalho com computador pode se tornar mais evidente que a criança constrói seus próprios micromundos pessoais” (PAPERT, 1994, p.194). Para o educador o mais importante na tarefa é a ideia de refletir sobre procedimentos, fazendo a sistematização e depuração do pensamento.

As primeiras versões do ambiente Logo não tinham a parte gráfica, com o objetivo de estender a aprendizagem dessa linguagem, até mesmo a crianças da pré-escola, foi desenvolvido o micromundo artificial da tartaruga, ampliando as possibilidades de aprendizagem por meio de depuração do pensamento. Surgindo assim a geometria da tartaruga que inspirou o presente trabalho. Mais tarde Papert desenvolveu a tartaruga de solo, no intuito de mesclar as duas conotações de construção em sua filosofia.

Segundo o autor é uma nova maneira de fazer geometria, baseada no Construcionismo e enriquecida com a Logo, onde o elemento gerador indivisível das demais formas na geometria de Euclides (o ponto), se confunde com a tartaruga na geometria computacional. No micromundo da tartaruga deve-se comandar um cursor, que possui a forma do animal, com o objetivo de criar desenhos, programas ou formas, com possibilidade de definir novos comandos e funções que podem ser usados exatamente como as primitivas da linguagem. O grau de sofisticação desses desenhos ou programas depende do nível do usuário, e podem ensinar ao cursor como desenhar um simples quadrado ou como plotar um gráfico complexo.

De forma descritiva, esse ambiente de aprendizagem consiste no aluno por meio do computador utilizar uma linguagem de programação como a Logo, tendo assim acesso a Matelândia, e ter de resolver um problema, através do controle do cursor ou da tartaruga de solo, nesse contato o aluno deve ensinar a tartaruga, a realizar

¹⁰ Nesse trabalho o artigo definido “o” antes de Logo implica que estou me referindo ao ambiente de aprendizagem, o artigo definido “a” indica que estou me referindo a linguagem de programação. Ao me referir a filosofia deixarei explícito para evitar confusões.

movimentos, por uma série de comandos sistematizados, que cumpram o desafio proposto, não deve haver instrução o aluno deve chegar ao resultado de forma empírica. Como resultado ao ser resolvido o problema dos comandos da linguagem de programação, temos na tela do computador a descrição da linha de raciocínio do aluno.

Quando a tartaruga se move, trata-se da execução dos procedimentos criados pelo próprio aprendiz, e assim a junção desses procedimentos é exposto por meio de uma tela ou no próprio papel um resultado gráfico do que fora idealizado. A visualização, reflexão e depuração do pensamento sobre o resultado se torna possível, essa depuração mental é necessária para o desenvolvimento cognitivo. Valente no prefácio de Papert (1985, p.9-10) explica que

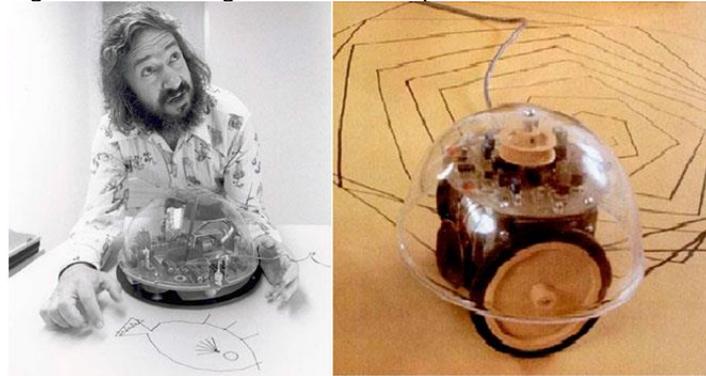
O computador é a ferramenta que propicia à criança as condições de entrar em contato com algumas das mais profundas idéias em ciência, matemática e criação de modelos. Segundo a filosofia Logo, o aprendizado acontece através do processo *de a criança inteligente “ensinar” o computador burro, ao invés de o computador inteligente ensinar a criança burra*. Com esta proposta, Papert inverte o atual quadro de uso do computador na escola. O computador deixa de ser o meio de transferir informação, e passa a ser a ferramenta com a qual a criança pode formalizar os seus conhecimentos intuitivos. O programa (a sequência de ações ao computador) que a criança elabora é o espelho que reflete o seu conhecimento sobre um determinado assunto e o seu estilo de pensamento. Este programa quando usado como objeto de reflexão, se torna uma poderosa fonte de aprendizagem.

Assim a Logo facilita o acesso do aluno aos seus processos mentais por meio de uma linguagem simples, precisa e formal, dando horizontes para modificar a forma como aprende as outras áreas do conhecimento.

2.3 Robótica Educacional

O marco inicial desse ambiente de aprendizagem ocorreu por volta de 1970, quando Papert e a equipe de colaboradores do MIT, criaram a tartaruga de solo, um robô programado pela linguagem Logo, que podia desenhar variadas formas geométricas, bastando a criança programar seus movimentos, a máquina podia deixar rastros por onde passava. Apresentava uma estrutura metálica com dois motores de passo para as rodas e um suporte para a caneta no centro da estrutura.

Figura 1 – Tartaruga de solo de Papert



Fonte: Matte (2012)

Com as inovações tecnológicas esse ambiente evoluiu exponencialmente e está ganhando força quanto a sua utilidade na educação. Essa abordagem carrega as duas conotações do Construcionismo, onde a criança pode aprender construindo um produto palpável, no caso um robô, geralmente com kits da LEGO® Education, ou sucatas de equipamentos eletrônicos. E também pode se tratar da construção de estruturas mentais através do controle, manipulação e depuração das ações do robô para resolver situações a que é colocada em contato. “A aprendizagem consiste em construir um conjunto de materiais e ferramentas que podem ser manejadas e manipuladas. Talvez, em essência, se trate de um processo de trabalhar com o que se tem”. (PAPERT, 1994, p. 206).

Em suma, a robótica educacional visa favorecer para que o aluno desenvolva a aprendizagem por assimilações concretas, no sentido de que o aluno possa atribuir conceitos ao que está em contato, evitando o abstrato, de forma multidisciplinar, com estímulo a sistematização e depuração do pensamento, além de criatividade para confirmar ou refutar hipóteses, ao construir ou manipular sistemas eletrônicos por meio de uma linguagem de programação para favorecer o acesso do aluno a Matelândia necessária para dominar a linguagem matemática.

3 A GEOMETRIA DA TARTARUGA DE PAPERT

Nesse capítulo faço uma breve apresentação do robô idealizado por Papert. Posteriormente exponho alguns aspectos da geometria criada para esse sistema mecânico.

3.1 Tartaruga mecânica de Papert

A primeira da série de tartarugas educacionais fora construída por Tom Callahan no MIT em 1969-70. A estrutura mecânica não era denominada de tartaruga na época.

Figura 2 – Tartaruga de Callahan



Fonte: The Logo Turtle (2010)

Esse sistema primário, apresenta mobilidade de um triciclo e utiliza a cauda para estabilidade da movimentação. Apenas em 1971, Mike Paterson introduziu a tartaruga robô controlada pela linguagem Logo, sob a orientação de Papert. O sistema robótico era conectado fisicamente ao computador por meio de linhas de arame. À medida que a tartaruga se movia, ela arrastava o fio de conexão. O aluno podia controlar esse sistema solicitando que andasse para frente ou para trás, especificada por programação, poderia solicitar que a máquina vire para esquerda ou direita em um ângulo específico.

Em 1972, Paul Wexelblat, projetou e construiu a primeira tartaruga sem fio, chamada *Irving*. Um robô com uma grande estrutura, e resistente o suficiente para crianças pequenas andar nela. Controlada por rádio, ele respondia aos comandos dados ao intérprete de Logo em execução no terminal. Com este robô, as crianças

podiam desenhar no chão e acompanhar o movimento, e concretização de seus pensamentos, criando um ambiente de aprendizado interativo.

Figura 3 – Tartaruga Irving



Fonte: The Logo Turtle (2010)

O robô apresentava a mesma estrutura no decorrer dos tempos, motor de passo para as rodas e caneta no meio para a plotagem, porém, as tartarugas passaram por uma série de atualizações e modificações. A ideia de Papert concretizada pela equipe de colaboradores do MIT se tornou um ambiente de aprendizagem rico em possibilidades para o ensino construcionista com aprendizagem ativa.

3.2 Geometria da Tartaruga

Considerada por seu criador como uma geometria que se distingue das demais, em Papert (1985, p.77)

A geometria da Tartaruga é um estilo diferente de geometria, da mesma forma que o estilo axiomático de Euclides é bem diferente do estilo analítico de Descartes. O de Euclides é lógico, o de Descartes é algébrico. A geometria da Tartaruga é um estilo computacional de geometria.

Essa distinção se define tanto por parte de alguns elementos, quanto no foco que cada geometria dá as suas construções. Em analogia ao ponto, o elemento indivisível e gerador de todos os outros objetos, da geometria euclidiana temos a tartaruga na de Papert. Desse modo argumenta-se que o ponto é um objeto estático, possui apenas uma posição, enquanto que a tartaruga é um objeto dinâmico, que possui uma posição e além disso possui uma orientação, esse elemento está sempre direcionado para onde aponta a cabeça da tartaruga. Não se pode atribuir associações ao cotidiano que se conhece ao ponto, enquanto para a tartaruga existe essa possibilidade.

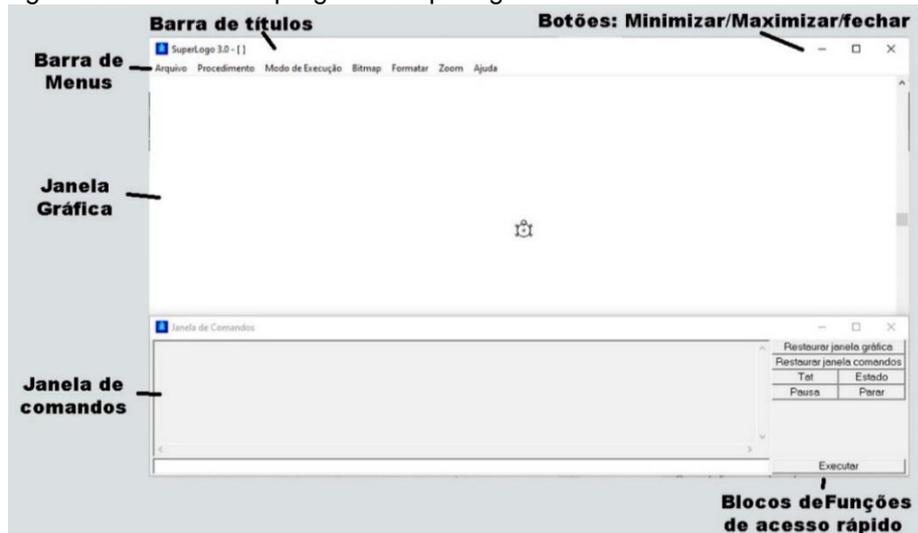
Em relação ao foco da construção, a geometria de Euclides tem foco nas relações lógicas axiomáticas para estabelecer um novo objeto, a de Descartes objetiva a análise algébrica dos objetos, enquanto que a de Papert está preocupada com a descrição do processo e no desenvolvimento cognitivo ao construir objetos.

Como as demais geometrias já são bem definidas e conhecidas, pretendo focar a partir de agora apenas em dar forma a geometria possível da tartaruga com exemplos no software Superlogo 3.0 do Núcleo de Informática Aplicada à Educação – NIED da Universidade Estadual de Campinas – UNICAMP. Porém o foco não será em descrever o programa, ou oferecer um minicurso ou mini tutorial da linguagem Logo. Tenho o objetivo de expor algumas construções da geometria da tartaruga e suas peculiaridades, para dar forma a mesma. Para isso irei exibir a tela do software com suas descrições e uma outra imagem com os comandos do Superlogo 3.0, e então estudar a geometria da tartaruga.

3.3 Apresentação básica do Superlogo 3.0

O ambiente do superlogo é disposto no sistema double-window, onde há duas janelas, isso significa que as duas podem ser redimensionadas e reposicionadas de forma independente, dando ao usuário a oportunidade de configurar o ambiente do modo que lhe deixar mais confortável.

Figura 4 - Interface do programa Superlogo 3.0



Fonte: o autor

Na janela de comandos temos os blocos de função de acesso rápido, vamos entender o que cada uma dessas funções faz:

Restaurar Janela gráfica: apaga o desenho realizado, restaurando as condições iniciais do programa;

Restaurar janela de comandos: Apaga os comandos digitados.

Tat: apaga todos os desenhos da janela gráfica e coloca a tartaruga na sua posição inicial.

Estado: mostra informações referentes à posição da tartaruga;

Parar: interrompe a execução de um programa;

Pausa: interrompe temporariamente a execução de um programa;

Executar: executa o código digitado na linha de comandos.

Embora as instruções aqui estejam sendo abordadas por alto, a utilização das mesmas é bem intuitiva e simples, por isso vou informar apenas os comandos básicos do Superlogo 3.0 e deixar o melhor entendimento de como funciona esse ambiente durante os argumentos cabíveis a geometria da tartaruga.

Figura 5 – Lista de comandos básicos do Superlogo 3.0

| | Comando | Significado |
|--|---------------|---|
| Comandos para movimentar e controlar a tartaruga | pf | para a frente (passos de tartaruga) |
| | pt | para trás (passos de tartaruga) |
| | pd | para a direita (valor do ângulo em graus) |
| | pe | para a esquerda (valor do ângulo em graus) |
| | ul | use lapis |
| | ub | use borracha |
| | un | use nada |
| | ljc | limpe janela de comandos |
| | pc | movimenta a tartaruga para o centro sem alterar a sua direcção |
| | at | apareça tartaruga |
| | dt | desapareça tartaruga |
| | ad | apaga todos os desenhos deixados na zona gráfica sem modificar a direcção da tartaruga |
| | tat | apaga todos os desenhos deixados na zona gráfica e coloca a tartaruga na posição inicial |
| | repita | executa repetições ou seja repete o número de vezes do(s) comando(s) especificados. Ex: repita 3 [pf 100 pd 45] |

Fonte: Martins (2009)

3.4 Breve análise da geometria da tartaruga

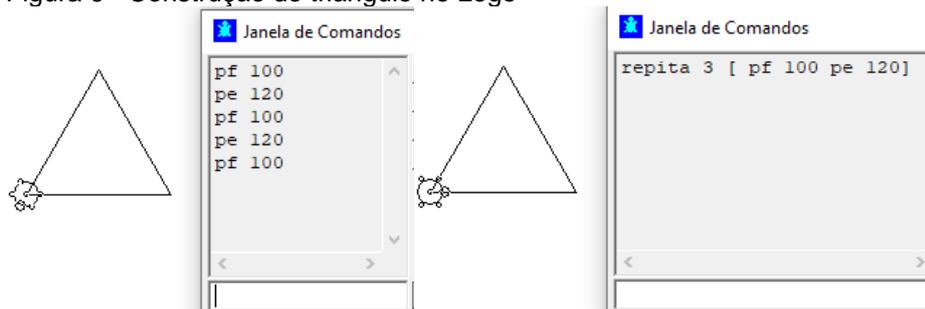
O elemento gerador de todos os outros é a tartaruga nessa construção teórica. Ela possui posição, direção e sentido, ou seja, tem propriedade de um vetor. A tartaruga é capaz de aceitar comandos ou listas de comandos, em sua “linguagem da tartaruga”, e executá-los de forma expressa, categorizando esse elemento como dinâmico. As diferentes representações desse elemento, tartaruga mecânica de solo e tartaruga luminosa de tela, apresentam isomorfismo matemático.

Essa geometria se resume ao estudo dos procedimentos na construção de objetos possíveis de se fazer movimentando o cursor. Assim o interesse da geometria da tartaruga está no processo de construção dos objetos dissociado dos teoremas

formais, mas que possa estabelecer relação com os objetos da geometria euclidiana, com foco no aprendizado por assimilação, reflexão e depuração.

O cursor sempre inicia na posição central da tela, na direção vertical sentido pra cima. Para fazer uma forma geométrica nesse ambiente, deve-se ordenar que ela se mova de acordo com as propriedades da estrutura do elemento geométrico. Vou pontuar algumas construções de objetos geométricos no Logo.

Figura 6 - Construção de triângulo no Logo

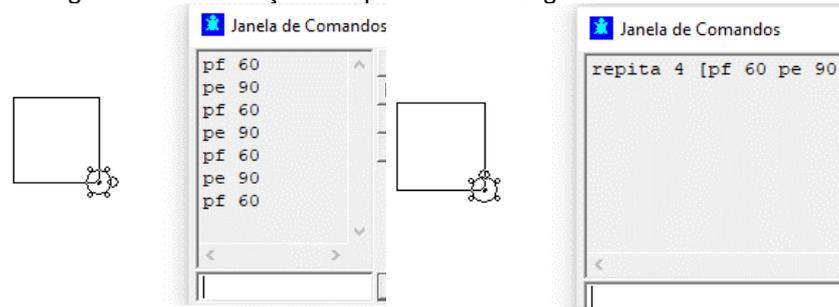


Fonte: o autor

Na figura acima estão dispostos os códigos para traçar o triângulo, a leitura deve ser feita de cima para baixo. Para se obter o triângulo é preciso antes de mais nada definir o tamanho dos lados, para obter um segmento da medida desejada, no caso da imagem acima foi de cem passos de tartaruga, que é a unidade de medida nessa geometria. Para mover o elemento devemos usar o comando *pf* (*para frente*) ou *pt* (*para trás*) seguidos da distância desejada, por serem relacionados a mudança de sentido, com relação aos comandos *pe* (*para a esquerda*) ou *pd* (*para a direita*) como estão relacionados a mudança de direção devemos usar o ângulo que deseja girar, a tartaruga faz um giro no próprio eixo não deixando nenhum traço novo.

O procedimento da esquerda na imagem acima pede para a tartaruga andar cem passos e virar cento e vinte graus para a esquerda, como há a noção intuitiva que um triângulo equilátero tem três lados igual, repete-se essas linhas mais duas vezes, tendo assim um triângulo equilátero de lado cem. Seria cansativo ter que construir formas geométricas desse modo sempre, então os idealizadores do projeto decidiram inserir a função *repita* e aprenda. A função *repita* garante a repetição em sequência de outros comandos ou códigos, e até a utilização de *repita* dentro de *repita* garantindo maior variação para sua utilização. Os códigos a direita na imagem 6 mostram a construção do mesmo triângulo utilizando a função *repita*.

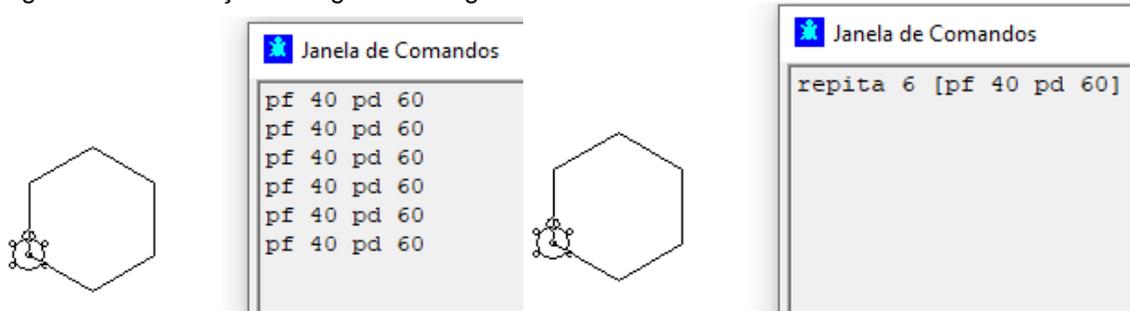
Figura 7 – Construção do quadrado no Logo



Fonte: o autor

Para o quadrado basta lembrar que se trata de uma figura que tem quatro lados e ângulos iguais, assim o código para traçar o quadrado de lado sessenta é exemplificado na imagem acima que informa que a tartaruga deve andar sessenta passos pra frente e virar noventa graus repetindo esses comandos quatro vezes.

Figura 8 – Construção hexágono no Logo

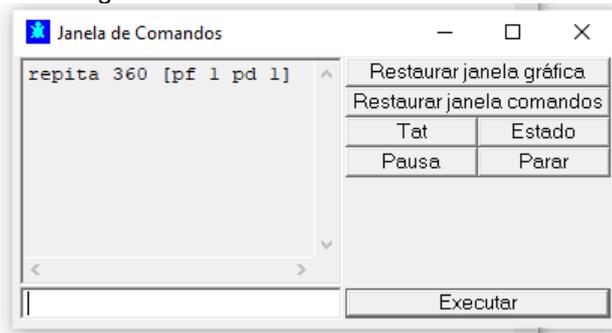
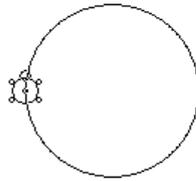


Fonte: o autor

Uso a construção do hexágono para exemplificar a ideia de como construir as figuras geométricas com qualquer número de lados no Logo. Perceba que o número de lados da figura determina quantas vezes os comandos deverão ser seguidos na função repita, e com isso o grau que a tartaruga deve virar em cada processo fica definido se observar que basta dividir 360° pela quantidade de lados, no exemplo $360^\circ/6 = 60^\circ$. Assim para desenhar um octógono, por exemplo, deve-se seguir essa mesma ideia. O comando pra frente simplesmente indica qual o tamanho dos lados da figura que se deseja obter.

Com o exposto acima vem a pergunta, como se faz para desenhar uma circunferência? Existe um problema, que será melhor abordado em tópico específico, na construção dessa figura. O processo para fazer algo que se assemelhe a circunferência consiste em desenhar uma figura com vários lados ao ponto de se assemelhar com a mesma.

Figura 9 – Circunferência da tartaruga



Fonte: o autor

Note que nesse caso estamos fazendo um polígono com 360 lados em que cada lado mede 1 passo. Esses simples objetos podem ser combinados para obter diferentes formas na tela. Mas, seria cansativo construir esses objetos toda vez que precisasse usá-los, a função aprenda foi desenvolvida para auxiliar em casos assim, o aluno pode guardar as construções na memória do sistema, para usá-las posteriormente em outros projetos, e assim aumentar o repertório da linguagem da tartaruga com a função aprenda. Não vou estender sobre a mesma, apenas deixo subentendido que ela existe e que pode adicionar novos comandos para serem executados. Com ela podemos rotacionar a figura geométrica simples que criamos dando-lhe formas mais complexas, o rearranjo de posição é essencial para tal.

Acredito que até aqui tenha se confirmado que a geometria da tartaruga está em sua maior parte dissociada das demais geometrias, confirmamos tal argumento nas palavras de Papert (1985, p. 92)

A geometria da Tartaruga pertence a uma família de geometrias com propriedades não encontradas nos sistemas euclidianos ou cartesianos. Essas são as geometrias diferenciais desenvolvidas desde a época de Newton e que viabilizaram a maior parte da física moderna.

O educador matemático Sul africano ainda complementa que o efeito que seu esse ambiente proporciona é puramente afetivo e relacional. Se trata de uma geometria para o desenvolvimento do aprendizado e reflexão dos processos cognitivos. Não tem foco em formalizações, e sim em associações e assimilações, dentro de um micromundo desenvolvido exatamente para esse objetivo, para um contato inicial, e não um contato corrente.

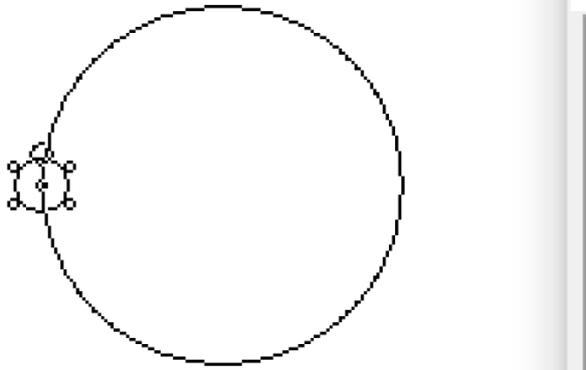
3.4.1 A limitação conceitual da geometria da tartaruga

A limitação mais grave na geometria da tartaruga, onde o próprio Papert (1985) perde quase metade do capítulo, do livro Logo: computadores e educação, tentando contornar a situação e no final chega à conclusão que se pode aprender outras coisas, como cálculo diferencial, em sua construção, é a circunferência da tartaruga.

E com isso a formalização de conceitos geométricos circulares como circunscrição e inscrição ficam comprometidos e dificultados, uma vez que a manipulação e maior grau de liberdade sobre a própria circunferência se faz necessária.

Essa construção está totalmente dissociada de sua definição formal, trata-se de uma variação de grandezas como grau e passos, fugindo, assim, de qualquer relação com a definição de circunferência que de forma direta entendo como *uma linha curva, fechada, cujos pontos são equidistantes de um ponto fixo, o centro*. A definição formal será vista no capítulo de comparações mais à frente. O que estamos realmente construindo fica claro se aplicarmos um zoom na tela.

Figura 10 – Circunferência da tartaruga ampliada



Fonte: o autor

Esse efeito que parece a pixelização¹¹ não se trata de um defeito no programa, e sim a descrição escondida do que se está construindo com o código acima, a circunferência aqui é uma variação de grandezas entre ir pra frente e virar tantos graus repetidas vezes, o que quero dizer é que ao digitar tal comando, não estamos realmente construindo uma circunferência, e nem de perto estamos nos aproximando do conceito da mesma, mas trata-se uma *debugging*¹² de construção visual para

¹¹ Pixelização é qualquer técnica usada na edição de imagens ou vídeo, em que uma imagem é desfocada, exibindo parte ou todo ele em uma resolução significativamente menor.

¹² Ao invés de erro, em LOGO diz-se que o programa tem um bug - a formiguinha no bolo. Achar ou eliminar o bug é um processo de debugging. Como em português não existe um termo especial para

disfarçar tal limitação, e o controle sobre a mesma está limitado. Meu argumento se confirma nas palavras do próprio Papert (1985, p.90) “Para o aluno, desenhar um círculo com a Tartaruga é mais do que usar "bom senso" para desenhar um círculo. Coloca a criança em contato com um conjunto de ideias que estão na base do cálculo diferencial”.

No livro: “computadores e conhecimento” de Valente (1998) em um análise geral do ambiente Logo e geometria da tartaruga encontramos vários educadores brasileiros que defendem esse ambiente, garantindo que em suas experiências em sala de aula ele tem potencial para proporcionar o envolvimento ativo dos alunos da formulação de hipóteses, favorecer a assimilação de situações reais levando a estabelecer conexão com modelos matemáticos e além disso a depuração do pensamento constante, em função de resultados conflitantes quando comparados com colegas. O foco central desse ambiente é o aprender fazendo, a atividade deve estar intimamente relacionada com conceitos matemáticos e assim o aluno possa dar significado aos conceitos e depurar processos mentais sobre as atividades, como vimos essa etapa é importante para o desenvolvimento cognitivo.

Um estudo atual se faz necessário para confirmar as vantagens e desvantagens dessa geometria, e como visto em citações acima admitidas pelo próprio Papert esse ambiente possui limitações, foi criado para um contato primeiro com a geometria, e não para a constante utilização no decorrer do processo de cognição dessa área, dificultando a possibilidade de utilização dessa ferramenta robótica no ensino de conteúdos matemáticos da grade curricular. Além disso a própria linguagem Logo limita a movimentação e conseqüentemente impossibilita trabalhar com alguns conteúdos da geometria euclidiana.

Portanto, um meio termo entre uma geometria facilitadora de assimilação inicial de conceitos e uma formal deve ser desenvolvido para dar possibilidade de utilização efetiva em sala de aula. Essa é a proposta deste trabalho com o projeto Jabuti Geométrico, uma atualização da tartaruga mecânica de Papert para resolver a limitação conceitual da circunferência e manter as propriedades didáticas da mesma.

Antes de abordar sobre a construção do projeto robótico que pode ser uma atualização para as limitações da geometria da tartaruga de Papert, no próximo capítulo faço uma breve introdução à robótica.

denotar o erro no mesmo estilo que o bug o 'faz, usaremos o termo bug e debugging, sem tradução (N.T.). Papert (1985, p.84).

4 INTRODUÇÃO A ROBÓTICA

Nesse capítulo defino conceitos básicos da robótica, abordo apenas os aspectos necessários para a construção do Jabuti Geométrico. A base teórica para o capítulo é o livro Introdução à robótica de Mataric (2014).

4.1 Definição de robô

O surgimento da ideia de robô não tem data registrada. Mas, a palavra foi popularizada pelo dramaturgo tcheco Karel Capek em 1921. De lá até aqui houveram algumas modificações conceituais.

A noção de robô foi se modificando ao longo dos tempos, antes entendia-se que esse termo se referia a toda máquina capaz de realizar um trabalho obrigatório, na forma de tarefas repetitivas. Porém, se olhar bem para essa afirmação verá que uma máquina de lavar poderia se enquadrar nela, ou até mesmo um ventilador. Houve uma época em que esse termo era reconhecido apenas como uma máquina humanoide. Atualmente a definição é bem mais complexa e considera algumas características importantes para um sistema mecânico ser considerado um robô.

Em Mataric (2014, p.19) encontramos uma definição ampla, bem explicada e exemplificada cada uma de suas partes. Farei o mesmo nesse trabalho de forma resumida. *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.* (MATARIC, 2014, p.19).

Primeiro vamos entender que **sistema autônomo** está relacionado ao fato de conseguir fazer tarefas sozinho, sem o controle externo de um ser humano, assim essa parte da definição engloba toda máquina capaz de realizar uma ação sozinha, ou seja, até aqui ventiladores e maquinas de lavar também entram na definição, assim como, software de Inteligência Artificial – IA, que faz processamento de dados e controle de outras maquinas automaticamente.

Mas as próximas palavras-chave da definição começam a fazer a filtragem. **Existe no mundo físico** isso desclassifica os softwares, pois, o mundo dos softwares é uma realidade simulada, o robô deve existir na mesma realidade irregular e incerta que nós humanos e poder agir sobre ela. Assim uma estrutura que habite no espaço com as verdadeiras propriedades do mundo real se faz necessário.

Sentir seu ambiente e agir sobre ele, referente à capacidade de coletar informações do mundo físico, através dos sensores. Seria o mesmo que fazemos

quando enxergamos ou escutamos, estamos recebendo informações do mundo exterior e interpretando e com isso guiando nossas ações. O robô não precisa ter a capacidade tão complexa de enxergar, mas deve ter sensores para sentir o ambiente e obter informações para poder operar sobre o mesmo, e realizar ação com base na informação recebida.

E finalmente **alcançar alguns objetivos**, trata-se da capacidade de memorizar e processar informações, ou seja, a inteligência propriamente dita. Assim espera-se que o robô possa processar internamente as informações e consiga usar o resultado do processamento para guiar suas ações para cumprir objetivos.

Com toda essa sofisticação na definição fica evidente que máquinas de lavar e ventiladores não se enquadram como robôs, eles possuem, características e funções distintas que não se resume ao simples movimento repetitivo mecânico é uma junção da estrutura com a inteligência artificial da máquina para cumprir objetivos no mundo real.

E dessa forma Mataric (2014, p. 21) define robótica “é o estudo dos robôs, o que significa que é o estudo da sua capacidade de sentir e agir no mundo físico de forma independente”. A robótica é um campo em constante crescimento e desenvolvimento.

Antes de continuar com as definições acho interessante trazer informações sobre dois projetos da robótica a tartaruga de Grey Walter e os veículos de Braitenberg.

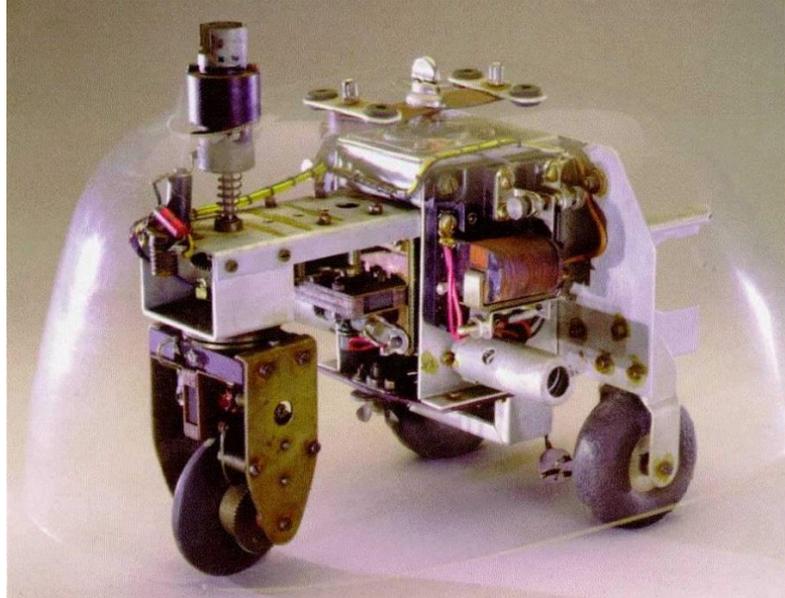
4.1.1 A tartaruga de Grey Walter

Matáric (2014, p.32) informa que “William Grey Walter (1910-1977) foi um neurofisiologista inovador, interessado no estudo do funcionamento do cérebro humano”. Walter estudou o funcionamento do cérebro por meio da construção e análise de máquinas para terem comportamento semelhante ao dos animais. Máquinas com tais comportamento são denominadas de *biomiméticas*¹³. Por volta de 1940 Walter construiu uma série de máquinas inteligentes para adotar tais comportamentos, as quais denominou de tartarugas. Se tratavam de robôs simples, construídos com três rodas, uma dianteira e duas traseiras, uma célula fotoelétrica (sensor de luz), um sensor de colisão, uma bateria recarregável, três motores (um para cada roda), um circuito eletrônico com dois tubos de vácuo (o cérebro do robô) o

¹³ Uma máquina que imita comportamento de sistemas biológicos.

sistema foi coberto com um plástico transparente para dar uma estética mais biológica.

Figura 11 – Tartaruga de Grey Walter



Fonte: Robô (2019)

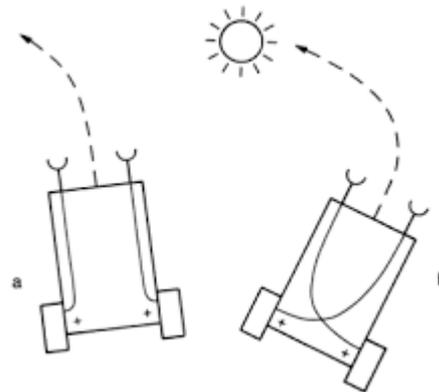
Embora pareça uma construção sofisticada, é na verdade uma máquina com funções simples. Como a máquina possuía sensor de luz e de contato, ela tinha de operar com base nessas informações, e com isso podia apresentar comportamentos como procurar a luz, ir em direção a luz, afastar-se da luz, desviar para evitar obstáculos, recarregar bateria.

O interessante da experiência é que as tartarugas apresentaram uma série de comportamentos complexos que não foram programados previamente, esses comportamentos serviram para exemplificar um comportamento animal típico. Assim Grey Walter tentava entender o funcionamento do cérebro dos animais, tentando recriar seus comportamentos através das máquinas.

4.1.2 Veículos de Braitenberg

Após o trabalho de Walter, influenciado pelo mesmo, Valentino Braitenberg escreveu um livro intitulado *vehicles* em 1984, segundo Mataric o livro descreve uma série de ideias, sobre como projetar robôs simples e leva-los a apresentar comportamentos que simule o de um animal. Vários projetos de robótica usaram esse livro como inspiração.

Figura 12 – Veículos de Braitenberg



Fonte: Mataric (2014)

Além das mesmas funções das tartarugas eles apresentavam alguns comportamentos extras, que auxiliava no estudo do comportamento animal. Os sensores eram ligados diretamente nas rodas como mostra o esquema acima, e então poderia programar se o aumento da percepção de luz iria acelerar (conexão excitatória) ou desacelerar (conexão inibitória) as rodas, ou se iria em direção contrária (fotofóbico) ou na mesma (fotolítico) que o sinal luminoso. Variando as conexões e programações entre os sensores e motores, vários comportamentos do mundo animal podiam ser gerados, e dando assim um entendimento maior sobre os mesmos.

Segundo Matáric (2014, p.32) “O livro de Braitenberg descreve como tais mecanismos simples podem ser utilizados para armazenar informação, construir memória e até mesmo conseguir que um robô aprenda”. Inspirando o novo campo emergente, IA, a dirigir seu foco para o estudo da inteligência.

O motivo de ter informado sobre esses dois projetos pioneiros antes de seguir com as definições e conceitos da robótica, foi para evidenciar um aspecto importante que a robótica proporciona, ou seja, por meio de seu desenvolvimento ela dá um entendimento maior sobre nosso próprio mundo, e sobre nós mesmos enquanto seres biológicos, quando nos preocupamos em programar o robô para uma determinada tarefa temos que analisar profundamente, para entender como ela funciona e assim possa instruir o robô a executá-la.

Não comentarei sobre IA, uma vez que nossa preocupação nesse trabalho é quanto ao uso da cibernética¹⁴ para o Jabuti Geométrico. O próximo tópico aborda sobre os componentes necessários a um robô.

4.2 Componentes de um robô

A definição de robô dá uma ideia do que essa máquina precisa ter, serão utilizados nesta sessão termos técnicos que podem ser desconhecidos pelo leitor, porém utilizaremos uma linguagem simples e clara, buscando evidenciar cada uma das partes.

Esse sistema antes de mais nada precisa ter um corpo físico para que possa existir e agir no mundo real, é claro. Esse corpo físico é desenvolvido levando em consideração a ação que o robô será programado para executar e o modo de execução da mesma. Além disso deve possuir sensores como dito anteriormente, para poder obter informações do ambiente a sua volta ou sobre si mesmo. Necessita de efetadores e atuadores para realizar a ação propriamente dita, sem eles seria imóvel. E por fim, um controlador, um dispositivo que faz o controle dos pulsos elétricos e coordena todas os demais dispositivos, fazendo literalmente a função de cérebro.

4.2.1 Corporalidade

A corporalidade refere-se a ter um **corpo material**, estar sujeito as mesmas leis que os objetos físicos obedecem. O corpo indica o espaço de ocupação da máquina e deve ser levado em conta para evitar colisões. Além disso ele indica as limitações do robô, no planejamento, já deve estar explícito o que irá ou não fazer com tal estrutura. O corpo determina também o quão rápido um robô pode se movimentar ou o peso que pode suportar. Assim na corporalidade do robô temos o chassi, os atuadores, os efetadores, os sensores e os controladores.

O chassi é a estrutura que irá suportar todos os demais, fazendo analogia a nosso corpo, trata-se do esqueleto do robô. Os demais serão apresentados em tópicos separados para melhor explicação, com exceção dos sensores como o projeto não terá sensor, deixo aqui apenas a definição encontrada em Mataric (2014, p.43)

¹⁴ A cibernética é um campo de estudo que tem como base o estudo e comparação dos processos de comunicação e controle nos sistemas biológicos e artificiais. Matáric (2014, p.26)

“sensores são dispositivos físicos que permitem a um robô perceber seu ambiente físico, a fim de obter informações sobre si mesmo e sobre os objetos que o cercam”.

4.2.2 Componentes de ação

Os **efetadores** são as partes que efetuam a ação final. Por exemplo, para se mover os robôs usam rodas ou patas mecânicas, esses seriam os efetadores dos robôs. efetuator depende da ação a ser realizada, se for para manipular objetos as garras seriam efetadoras, como se fosse nossas mãos. Para cada ação exige um tipo de efetuator. Mas os mesmos não funcionam de forma independente, uma roda não faz o movimento de rotação sozinha, nem uma garra se movimenta por ela mesma, para que um efetuator faça uma ação ele necessita de **atuadores** para fazer o trabalho real. Pense neles como os músculos e nervos, sem os mesmos eu não teria nem mesmo digitado esse trabalho.

Quando fazemos uma ação geralmente ela é classificada em dois tipos: locomoção e manipulação, que correspondem aos dois principais subcampos da robótica. Mataric (2014, p.47) define os dois

1. robótica móvel: referente aos robôs que se movem, principalmente no solo, mas também no ar e debaixo da água;
2. manipulador robótico: relacionado basicamente com os braços robotizados de vários tipos.

Nesse trabalho estou interessado em descrever a robótica móvel, por ser exatamente o subcampo específico do projeto Jabuti Geométrico.

4.2.3 Autonomia

A autonomia está relacionada aos controladores que desempenham a função do cérebro e sistema nervoso no robô. Segundo Mataric (2014, p.49)

Autonomia é a capacidade de tomar as próprias decisões e agir de acordo com elas. Para os robôs, autonomia significa que as decisões são tomadas e executadas pelo próprio robô, e não por operadores humanos. A autonomia pode ser completa ou parcial.

Assim mesmo sistemas semiautônomos podem ser considerados robôs, informação importante para quando definir o Jabuti Geométrico.

4.3 Efetadores e Atuadores

Em Mataric (2014, p.51) “um **efetuador** é um dispositivo do robô que exerce um efeito (impacto ou influência) sobre o ambiente”. Veja os efetadores como a parte que irá interagir fisicamente com o mundo externo e assim produzir a ação. Como rodas, garras, braços mecânicos, pontas de canetas ou lápis ou qualquer coisa projetada no robô para entrar em contato com objetos externos e efetivar assim sua programação. Compare os efetadores como nossas mãos, dedos, pés e língua quando for pegar, tocar, andar e lambe algo respectivamente. Embora outras partes do corpo sejam responsáveis de levar o efetuator até a posição necessária é por meio desse dispositivo que a ação final é realizada.

Assim temos em Mataric (2014, p.51) que “Todos os efetadores têm algum mecanismo que os permite fazer o seu trabalho. Um **atuador** é um mecanismo que permite que o efetuator execute uma ação ou movimento”. Compare o atuador com os músculos e tendões ao executar uma ação. Eles são os geradores de força e direcionadores para executar um movimento atuando assim no efetuator lhe dando movimento para efetivar ação.

Existem atuadores ativos e passivos. Os ativos são aqueles que produzem a própria força para mediar o movimento do efetuator, como os motores. Enquanto os passivos retiram a força do externo, onde esse atuador externo pode ser até mesmo a própria gravidade.

4.3.1 Tipos de atuadores

Entre os atuadores ativos e passivos existem vários modelos e designs para os diferentes e mais criativos modos de aplicação, que abrangem: motores elétricos, dispositivos hidráulicos, dispositivos pneumáticos, materiais foto reativos, materiais quimicamente reativos, materiais termicamente reativos e materiais piezoelétricos. Nesse trabalho falarei apenas sobre o primeiro tendo em vista que é esse tipo de atuador que irei utilizar na construção do projeto robótico.

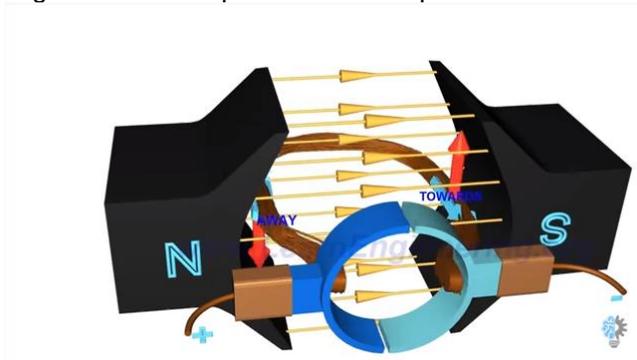
Motores são arranjos eletromagnéticos que produzem força de rotação. Três tipos de motores são os mais comuns na robótica: Motores de corrente contínua, Motor de passo e Servo motor. Geralmente encontramos siglas diferentes para denominar o mesmo tipo de motor (DC ou CC¹⁵), nesse trabalho adotarei a sigla CC.

¹⁵ DC da expressão em inglês Direct Current e CC da expressão Corrente Contínua.

4.3.2 Motores CC

Um motor de corrente contínua é um manipulador de polaridade magnética para gerar força rotacional. Como o nome sugere a armadura do motor é conectada a uma fonte CC, o motor utiliza imã, bobinas e corrente para gerar campos magnéticos cuja força eletromotriz faz girar o motor.

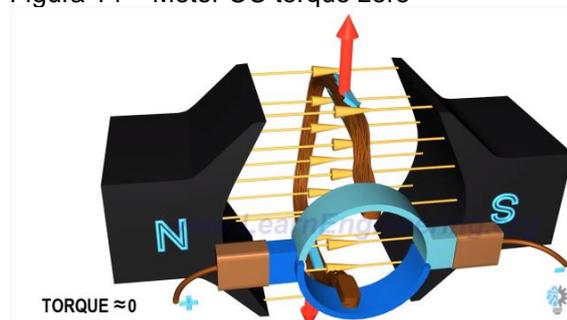
Figura 13 – Exemplo de motor simples



Fonte: Learn Engineering (2016)

A energia circula pela bobina gerando um campo magnético com polaridade, o estator também é polarizado e os polos respectivamente opostos irão se atrair gerando torque e força rotacional. Porém há um problema quando a bobina entra no campo sem polaridade, o torque nesse caso é zero, a bobina continua rodando apenas pela força inercia até voltar aos campos magnéticos polarizados. O que torna a rotação do motor irregular.

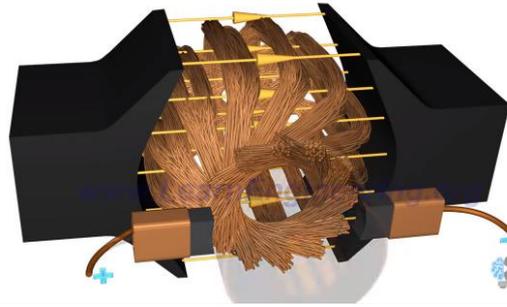
Figura 14 – Motor CC torque zero



Fonte: Learn Engineering (2016)

Para reverter essa situação e deixar a rotação mais suave, monta-se um sistema com mais bobinas, onde quando uma bobina estiver na área sem polaridade as demais estarão polarizadas isso faz com que o movimento do motor seja constante e regular.

Figura 15 – Motor CC arranjo de bobinas



Fonte: Learn Engineering (2016)

Além disso devemos notar que, para funcionar, o motor precisa ser alimentado com uma faixa de tensão correta por uma fonte de energia de corrente contínua. A corrente gerada é proporcional ao trabalho realizado. Da física tiramos que o trabalho é o produto da força pelo deslocamento. Desse modo, quando um robô está empurrando algum objeto ou preso em algo que o imobilize, os motores tracionam as rodas e aumentam o consumo de energia consumindo mais da bateria do que normalmente consome, esse fato é importante para o planejamento do robô, deixando claro que deve levar em conta evitar topar em obstáculos para poupar a bateria.

O funcionamento do motor é mais complexo do que essa noção intuitiva que estou apresentando, porém, a mesma se faz necessário para entender a diferença entre os dois primeiros tipos de motores apresentados nesse tópico, e por que o segundo motor é importante para o projeto robótico a ser construído nesse trabalho.

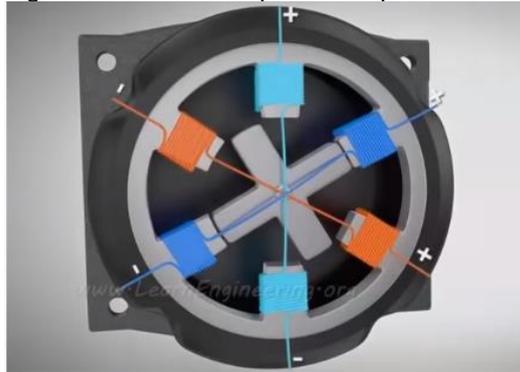
Note que não existe uma forma de controlar a posição do eixo, o motor cc se resume a manter a rotação ou parar de se mover variando a força e velocidade da corrente, para projetos que precisam de precisão no movimento, como o desse trabalho, esse tipo de motor não é o mais indicado.

4.3.3 Motores de Passo

A ideia do motor de passo é ter um atuador que consiga dar precisão nos movimentos de rotação com alto torque, porém baixa velocidade. Ele funciona com corrente contínua também, mas seu diferencial está em poder posicionar o eixo precisamente em qualquer posição, já que a sua rotação é dividida em vários passos.

Vamos analisar um exemplo simples, para entender até como a versão mais sofisticada do equipamento funciona. O conceito básico é que as bobinas dessa vez ficam no estator enquanto que no centro fica o rotor.

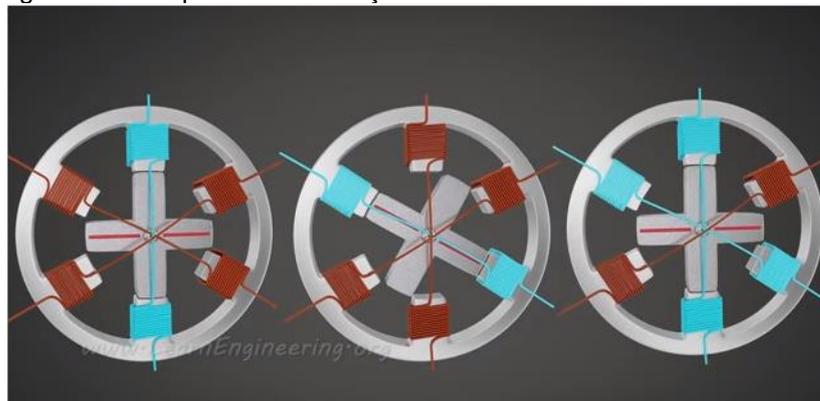
Figura 16 – Motor de passo simples



Fonte: Learn Engineering (2016)

Note que o rotor (a cruz no centro da imagem) tem menos dentes do que o estator, esse projeto visa garantir que apenas um par de dentes esteja alinhado com o estator. Para se ter o controle da rotação, esse motor funciona através de uma sequência de ativações e desativações de bobinas, onde ao ligar uma bobina o rotor será atraído pra ela, assim basta desativar a bobina novamente e ativar a próxima. Segue exemplo das ativações.

Figura 17 – Sequência de ativações de bobinas



Fonte: Learn Engineering (2016)

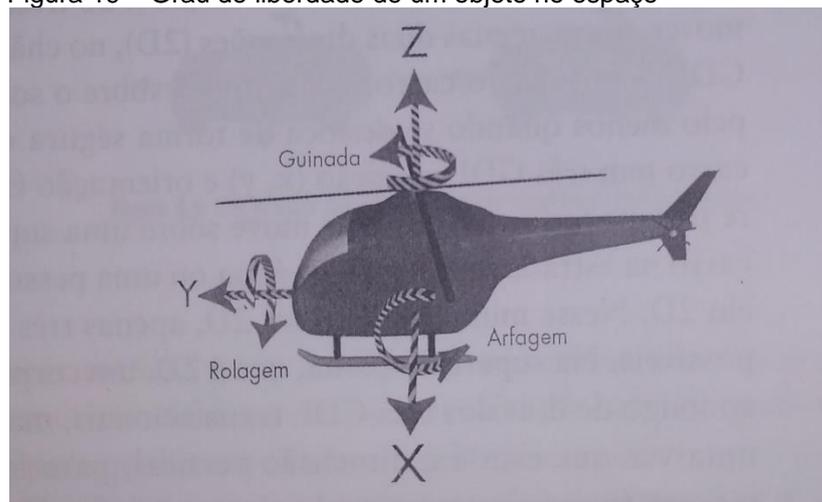
A distância que o rotor se movimenta fica definida pela construção do motor e pelo modo de ativação. Quanto a construção temos **motores de relutância variável**: estator com enrolamentos e rotor com várias polaridades. **Imã Permanente**: Parecido com o motor de relutância variável, porém possui um imã fixo no rotor. **Híbrido**: imã permanente no eixo e rotor *multidentado*. É o tipo de motor de passo mais comum em indústrias.

O modo de ativação se resume ao passo completo (full step) uma bobina é ativada por vez ou duas bobinas para aumentar o torque. E meio passo (half-mode) deve-se alternar entre ativar uma única bobina por vez e ativar duas bobinas de uma vez consecutivamente, dobrando a posição controláveis do rotor.

completamente o movimento de um sistema mecânico”. Em termos mais simples, define todos os modos como o robô pode se mover ou agir no ambiente.

Em geral um corpo no espaço tridimensional apresenta seis GDL. São classificados em translação e rotação. **GDL de translação** está relacionado a alterar a posição em cada um dos eixos tridimensionais, ou seja, o movimento em X de ir pra frente e pra trás é 1 GDL, a ação em Y de movimentar para direita e esquerda, e no eixo Z de deslocar para cima e para baixo, ambos representam uma unidade de grau de liberdade respectivamente. Dessa forma existem três GDL de translação. **GDL de Rotação** como o nome sugere trata-se da possibilidade de efetuar o movimento de rotação em torno dos três eixos do espaço. Existem assim três GDL de rotação, sendo seis no total. Para ficar mais claro vamos utilizar o exemplo do livro de Mataric (2014, p.63)

Figura 19 – Grau de liberdade de um objeto no espaço



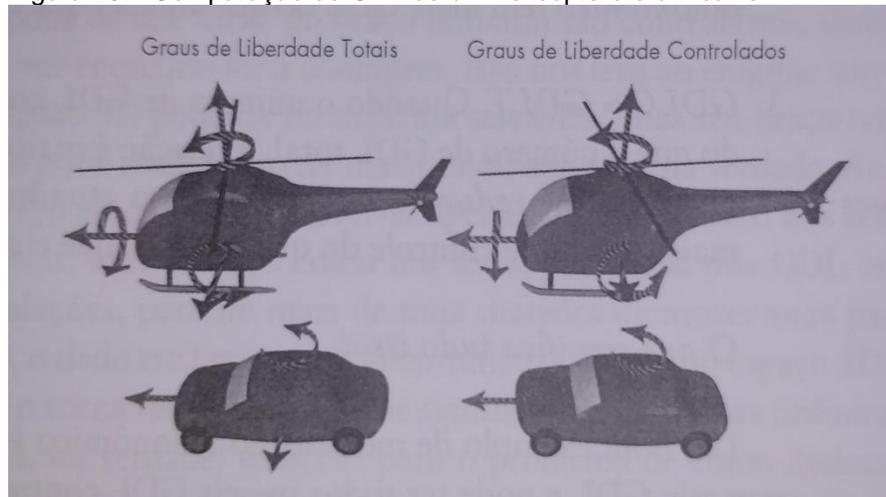
Fonte: Mataric (2014)

Imagine o movimento do helicóptero quando está voando, ele pode transladar e rotacionar em todos os eixos do espaço. O movimento de rotação em torno do eixo Z, Y e X é denominado Guinada, Rolagem e Arfagem respectivamente.

Um efetuator pode ter qualquer quantidade de GDL, uma roda tem um grau de liberdade, pois, ela só pode se deslocar em um eixo por vez. Porém, existem efetutores passivos como mencionado acima, e com isso existem também GDL que não são ativamente controlados. Assim GDL controláveis são os que possuem um atuador que possibilite o controle do movimento na direção do mesmo. Enquanto que GDL incontroláveis consiste na limitação do atuador ou efetuator em executar o movimento na direção de algum eixo que a estrutura física pode se movimentar.

A comparação entre os movimentos possíveis de um carro e um helicóptero torna claro a ideia. Um helicóptero por se mover no espaço tem possibilidade de se movimentar pela translação e rotação, logo possui seis GDL, essa máquina tem o controle sobre todos os seus movimentos, logo todos os seus graus de liberdade são controláveis. Já um carro por estar limitado, em situação normal, ao plano bidimensional, tem possibilidade de se movimentar por dois graus de translação (ele não pode ir pra cima), e 1 grau de rotação (pode apenas girar para esquerda ou direita), porém, quando dirigimos um carro só temos como ir pra frente e virar, não existe um mecanismo que o faça andar de lado. Logo um carro possui três GDL, mas apenas dois são controláveis.

Figura 20 – Comparação do GDL de um helicóptero e um carro



Fonte: Mataric (2014)

A relação dos graus de liberdade com os controladores define se o movimento do robô será mais complicado ou mais simples de se executar. Se ele puder controlar todos os GDL que lhe são possíveis, é denominado robô holonômico, e é simples de movimentar, possui apenas uma forma de realizar os movimentos em cada direção, mas não apresenta dificuldades na movimentação. Se a máquina não controla todos os GDL então é classificada como não holonômica e apresenta dificuldades em sua movimentação, para quem dirige sabe muito bem dessa dificuldade ao ir fazer a baliza, se o carro tivesse controle sobre o movimento lateral dirigir se tornaria bem mais fácil.

Agora se o robô pode controlar mais do que os possíveis para sua realidade então são classificados como redundante, e possui várias formas de executar uma mesma ação. A título de curiosidade nosso braço, excluindo a mão, possui sete graus

de liberdade e todos controláveis, por isso podemos pegar as coisas de diferentes maneiras e posições.

Embora apresentados de formas bem resumida esses conceitos introdutórios da robótica, são necessários para o capítulo que descreverei a construção do robô e estarei trazendo mais informações sobre os mesmos.

5 DESCRIÇÃO DOS ATUADORES E CONTROLADORES DO PROJETO

Esse projeto robótico tem dois tipos diferentes de atuadores e três funções distintas que podem ser exercidas por esse sistema. Eles precisam atuar para movimentar o robô e movimentar a caneta em dois eixos. Desse modo referente aos tipos de atuadores ativos no projeto usei motores de passo 28BYJ-48 e um servo motor SG90 Tower Pro.

Figura 21 – Motor 28BYJ-48 e Micro servo SG90 Tower Pro



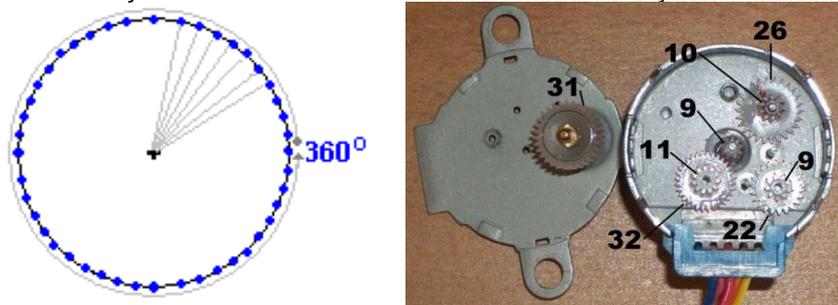
Fonte: Oliveira (2017)

5.1 Motor de passo 28BYJ-48

É um motor de fabricação chinesa, a construção é do tipo relutância variável, unipolar e funciona com uma tensão de 5~12V contínuos. É um dispositivo barato e com uma boa precisão, por isso muito usado em projetos robóticos de baixo custo, que inclusive é a proposta desse trabalho, por isso a escolha desse modelo.

Ele possui um mecanismo de redução de 1/64, ou seja, a Rotação Por Minuto - RPM do eixo interno do motor é reduzida em 64 vezes. Para melhor entendimento vou exemplificar.

Figura 22 – Rotação eixo externo do motor e sistema de redução



Fonte: Murta (2018)

Esse motor internamente não possui uma precisão muito alta, por isso para aumentar a mesma, desenvolveram a redução por engrenagens, na figura acima

mostra o esquema de redução, os números são referentes a quantidade de dentes de cada engrenagem. Essa disposição de engrenagens faz com que o motor interno precise girar, nesse modelo, 64 voltas para que a engrenagem que move o eixo externo gire uma, isso significa que, se dividirmos os 360° por 64 teremos $5,625^\circ$ que é exatamente o grau que esse motor desloca no eixo externo a cada 64 voltas do interno. Como vimos no capítulo anterior, temos controle sobre o eixo interno do motor, dessa forma podemos ainda ter o controle sobre a revolução externa dependendo do modo de operação pode chegar a $0,088^\circ$ por passo.

Assim para o eixo externo rodar 360° , o interno vai precisar dar 4096 passos, ou no modo de ativação passo completo, esse número cai pela metade. Como dito no capítulo anterior e no parágrafo acima, os motores de passo possuem certos modos de ativação que modificam seu torque, rpm e passo. São eles o Passo completo com alto torque (*Full step*), Passo completo com baixo torque (*Wave step*), Meio Passo (*half step*) e Micro Passo (*Microstep*).

Para entender a ideia deve lembrar do capítulo anterior sobre o funcionamento do motor de passo. *Full step* é o acionamento de duas fases (bobinas) simultaneamente garantindo um alto torque e passos mais largos. *Wave step* é o acionamento de uma fase por vez tem o mesmo passo que o anterior. *Half step* é o acionamento alternado entre uma fase e duas fases ao mesmo tempo dobrando o número de passos controláveis. *Microstep* é o modo de ativação com maior precisão, infelizmente não se aplica ao nosso dispositivo, pois o mesmo não permite controle de corrente.

5.2 Micro servo motor sg90

É um dispositivo de fabricação da empresa Tower Pro e funciona com uma tensão de 3~7,2V contínuos, apresenta uma velocidade de 0,12 seg/60Graus. Uma precisão excelente variando de 0° a 180° .

Não pretendo me estender sobre o Micro servo uma vez que a função dele no sistema robótico é apenas baixar e levantar a caneta, por isso dispenso maior atenção e foco nos motores de passo, eles são os atores principais desse projeto.

O jabuti tem 2 atuadores para as rodas e 5 atuadores para o eixo de plotagem, variando entre ativos e passivos.

O projeto possui controladores secundários e o principal. Os controladores secundários são os drivers dos motores de passo, e o principal é a placa Arduino. Vamos explorar cada um deles.

5.3 Controladores secundários

Os motores de passo não podem ser conectados diretamente a placa Arduino, tem o risco de danificar o equipamento, então a conexão desses motores é mediada por Shields reguladores de tensão. Para o modelo 28BYJ-48 a fabricante desenvolveu os drivers ULN2003.

Figura 23 – Driver ULN2003



Fonte: o autor

Esse chip possui um conjunto de sete drivers de transistores, que permitem o acionamento de cargas indutivas. Todas as saídas tem o coletor aberto e diodos de supressão. Os transistores suportam tensões de até 50V e correntes de até 500 mA. Todas as entradas IN1, IN2, IN3 e IN4, tem limite de 5V. O pino comum tem que ser conectado na tensão de alimentação do motor. Nesse caso é conectado no 5V. De forma simples é um controlador regulador de tensão projetado para os motores 28BYJ-48.

Os pinos IN1, IN2, IN3 e IN4 são conectados ao Arduino. A alimentação pode ser feita pelo Arduino, nesse caso deve deixar o jump que está do lado superior direito da imagem 23, ou por fonte externa, deve-se remover o jump e conectar os fios a seus respectivos polos.

5.4 Controlador principal: Arduino

O principal controlador desse projeto robótico é o microcontrolador ATmega328 contido na placa de prototipagem Arduino. Em grupooitoarduino (2015) encontramos que

O Arduino foi criado em 2005 pelo professor Massimo Banzhi na Itália. Banzhi queria ensinar para seus alunos conceitos de programação e de eletrônica, porém enfrentava um problema, não havia placas de baixo custo no mercado, e, portanto, isso dificultaria a aquisição do produto por todos os seus alunos. Com isso em mente Banzhi decidiu criar uma placa de baixo custo que fosse semelhante a estrutura de um computador para que seus alunos tivessem a oportunidade de aprendizado. A sua placa, nomeada de Arduino, foi um sucesso, recebendo uma menção honrosa na categoria Comunidades Digitais em 2006.

O objetivo era elaborar um dispositivo que fosse ao mesmo tempo barato, funcional e fácil de programar, sendo dessa forma acessível a estudantes e projetistas amadores. Foi adotado o conceito de *Open Source*, o que significa que qualquer um pode montar, modificar, melhorar e personalizar o Arduino. Por conta disso há vários modelos no mercado, o modelo que utilizarei no projeto é o Arduino Uno.

Monk (2014, p.20) informa que “na realidade, temos um pequeno computador em um chip. Ele tem tudo que os primeiros computadores pessoais continham, além de outras coisas mais”. Se trata de um pequeno computador que você pode programar para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele.

Figura 24 – Componentes de um Arduino Uno



Fonte: Monk (2014) adaptado

Essa placa possui pinos de entrada e saída, para estabelecer a conexão entre o microcontrolador e o resto dos componentes eletrônicos. Não se trata de uma placa nova no mercado, mas é muito utilizada em projetos robóticos de entrada, vários posts

sobre o funcionamento dessa placa são encontrados facilmente, por isso para não estender o trabalho irei falar apenas das partes e funções necessárias para a construção do Jabuti.

Na figura 24 está listado os componentes fixos da placa, o leitor novato nessa área deve se orientar pela imagem para visualizar e entender de qual parte estarei falando. Antes, devo mencionar que como todo dispositivo eletrônico o Arduino deve ser alimentado com uma faixa de tensão certa para funcionar, essa alimentação é feita por uma fonte externa que deve estar entre os limites de 6V¹⁶ a 20V. Porém, se alimentada com uma tensão acima de 12V, o regulador de tensão da placa pode superaquecer e gerar danos, por isso o recomendado é que se utilize fontes de alimentação de até 12V.

O **conector USB** serve tanto para alimentar o Arduino com 5V, quanto para fazer a conexão da mesma com o computador, é a partir desse componente que é feita a comunicação entre os dois dispositivos, e com o IDE para programar e reprogramar a placa. Isso é possível graças ao microcontrolador ATMEL ATMEGA16U2 presente no Arduino, este microcontrolador é o responsável pela forma transparente como funciona a placa Arduino UNO, possibilitando o upload do código binário gerado após a compilação do programa feito pelo usuário.

Soquete **Jack CC** tem a função única de alimentar a placa com diferentes faixas de tensão que devem estar na faixa de segurança para não danificar o dispositivo. Para a alimentação também podemos usar o **pino vin**, ele dá acesso tanto a tensão alimentada da placa como torna possível energizar todo o dispositivo por ele.

A chave **reset** ao ser acionada, envia uma carga para o microcontrolador obrigando o mesmo a executar o código desde o início. As estrelas dessa placa para o projeto são os pinos.

Os pinos controladores de alimentação, são especificamente necessários para manipular a corrente elétrica dos componentes eletrônicos. Os nomes e posição podem ser conferidos na imagem acima, vejamos sobre a função deles:

IOREF – É usado para indicar a tensão que o Arduino trabalha. É útil para fornecer uma tensão de referência para que shields¹⁷ possam selecionar o tipo de interface apropriada.

¹⁶ Símbolo V refere-se a unidade de tensão elétrica Volts

¹⁷ Um Shield para Arduino nada mais é que uma placa de circuito impresso com conectores que se encaixam na parte superior de uma placa arduino.

RESET – tem a mesma função da chave reset, ao aplicar uma tensão momentânea de +5V nesse pino o controlador ATmega328P irá ser obrigado a resetar.

Os pinos 3.3V e 5V - têm a tensão regulada para fornecer essa faixa independente da tensão da fonte externa.

GND – a mesma função do fio terra, significa que possui 0V.

Temos os pinos de **entrada analógica**, o conjunto de pinos ao lado do de alimentação. São seis pinos analógicos denominados “Analog in 0 to 5”. Além disso o conjunto de pinos de **entrada Digital**, na parte de cima da imagem denominados “Digital (pwm~)”. As entradas digitais só podem assumir dois estados, HIGH e LOW, ou seja, 5 V ou 0 V, em valor lógico significa 1 e 0 respectivamente, ou seja, se o dispositivo conectado a elas estará ligado ou desligado. Não regulam a tensão para outros valores já as entradas analógicas possuem essa função. Porém as entradas digitais pwm, são as que possuem o “~” na frente podem ser usadas como portas analógicas. A regulação de tensão é útil para efetuar controle dos dispositivos podendo alterar a força dos mesmos.

Para que todos esses componentes tenham um bom funcionamento e sincronia o Arduino conta com alguns microcontroladores sendo o principal o ATmega328. Na imagem é o dispositivo retangular preto com 28 pinos. Seu encaixe é feito em um soquete do tipo dual-in-line – DIL, podendo ser substituído facilmente. Ele conta com uma unidade central de processamento – UCP, que controla tudo o que acontece dentro do dispositivo. A UCP busca dados na memória de acesso aleatório – RAM, interpreta e modifica os dados e depois volta a armazená-los. Assim ele faz o controle de tudo o que acontece na placa, podendo alterar funções dos pinos programáveis e alterar a tensão dos mesmos.

Por meio desse microcontrolador o usuário pode programar sua placa a vontade para atender aos objetivos do seu projeto específico. Mas para isso é preciso de um software compatível, no caso dessa placa, o oficial é o IDE do Arduino. Por isso vou abordar brevemente sobre a mesma e finalmente descrever a construção do Jabuti Geométrico.

5.5 O IDE do Arduino

O software Arduino (IDE) de código aberto facilita a gravação e o upload do código na placa. É executado no Windows, Mac OS X e Linux. Este software pode ser usado com qualquer placa Arduino.

5.5.1 Instalação no Windows 10

Para iniciar a instalação, antes de mais nada, é necessário baixar o instalador do IDE no site www.arduino.cc.

Figura 25 – Site do Arduino



Fonte: o autor

Ao entrar no site, posicione o cursor do mouse no menu programas na barra de opções, localizado na parte superior. Selecione transferências para ser direcionado para a página da imagem acima. Nessa página escolha a opção compatível com seu sistema operacional, para o exemplo é o Windows 10, a versão até o momento da escrita desse texto é a 1.8.10, mas todas as versões apresentam uma boa compatibilidade, fazendo com que projetos possam ser utilizados em qualquer versão desse software. Será direcionado para uma outra página que tem duas opções *contribuir e download* destacado e apenas download, a contribuição é opcional e serve para dar recursos para a equipe de desenvolvimento continuar atualizando e disponibilizando as mesmas.

Após o download dependerá mais uma vez de sua escolha, se escolheu baixar o aplicativo do Windows, será direcionado para a loja do Windows onde poderá clicar

em obter, para abrir o Microsoft Store e então basta clicar em instalar para o software ficar disponível para uso. Porém, se escolheu baixar a versão zip terá que ir no diretório onde salva os downloads em seu computador e extrair os arquivos, uma pasta chamada Arduino-1.8.10, será extraída, ao entrar nela deve clicar no arquivo arduino do tipo aplicativo para começar a usar o software.

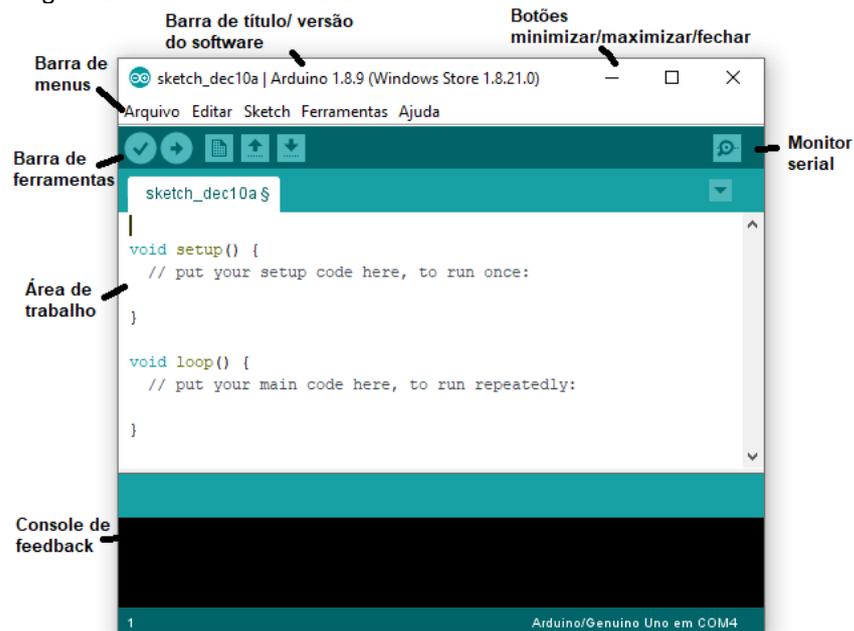
Agora terá que conectar a placa Arduino ao seu computador usando o cabo USB. O LED verde de energia, rotulado ON, deve acender. O Windows irá iniciar seu processo de instalação do driver automaticamente. Após alguns instantes, o processo falhará. Não é problema, basta ir no *painel de controle*, que pode ser acessado pelo menu *Iniciar*, navegue para *Sistema e Segurança*, em seguida, clique em *Sistema*, e por fim abra o *Gerenciador de dispositivos*. Na seção *Outros Dispositivos* terá um ícone de exclamação amarelo em *Outros Dispositivos* é o driver do Arduino que precisa ser atualizado. Clique com o botão direito em *Dispositivo Desconhecido* e selecione *Atualizar Driver* e escolha a opção *pesquisar automaticamente software de driver atualizado*, clique em *Avançar* e finalmente irá exibir uma mensagem de confirmação da instalação, o Arduino está pronto para ser usado, basta abrir o IDE e começar a programar.

5.5.2 Interface do IDE

O software contém um editor de texto para escrever códigos que serão enviados ao Arduino, uma área de mensagens de feedback para conferir o status do processo realizado, um console de texto, uma barra de ferramentas com botões para funções comuns e uma série de menus. Através dele pode fazer upload de programas ao Arduino e manter a comunicação com ele.

A área de trabalho é onde irá escrever os programas para enviar à placa. É composta por padrão pelas funções *setup* e *loop*, elas configuram toda a inicialização do sistema montado e as ações repetitivas da placa respectivamente.

Figura 26 – Interface do IDE Arduino



Fonte: o autor

Ao escrever um código fora dessas funções estará acessando a área de configuração global que geralmente é utilizada para inserir bibliotecas e variáveis que serão usadas em toda a programação e não em apenas uma função específica.

Figura 27 – Barra de Ferramentas

- 
Verificar
 Verifica se há erros ao compilar seu código.
- 
Upload
 Compila seu código e o envia para a placa configurada. Veja o **upload** abaixo para obter detalhes.
 Nota: Se você estiver usando um programador externo com sua placa, mantenha pressionada a tecla "shift" no seu computador ao usar este ícone. O texto mudará para "Upload using Programmer"
- 
Novo
 Cria um novo esboço.
- 
Abrir
 Apresenta um menu de todos os esboços no seu caderno de rascunhos. Ao clicar em um deles, será aberto na janela atual, substituindo o conteúdo.
 Nota: devido a um erro em Java, este menu não rola; se você precisar abrir um esboço no final da lista, use o arquivo | Menu Sketchbook .
- 
Salvar
 Salva seu esboço.
- 
Monitor serial
 Abre o [monitor serial](#) .

Fonte: Software Arduino (2015)

A barra de ferramentas é composta por botões usuais de acesso rápido, como pode ser visto pela descrição da imagem acima, eles são as ferramentas necessárias para criar, verificar e compilar um programa para o Arduino. O monitor serial exibe dados programados para serem enviados para a placa, muito úteis durante a verificação da programação para saber se algumas funções estão sendo lidas

corretamente, ou para verificação de hardware que podem ser obtidas pela computação dos dados exibidos pelo monitor serial.

A barra de menus como o nome sugere possui alguns menus uteis para a programação, como:

- Arquivo, serve para criar, abrir, salvar programas e mudar as preferências do software.
- Editar, contém funções para ajudar a editar o código, como localizar, desfazer, refazer, entre outras.
- Sketch, além de ter funções de compilar e fazer upload do código, há o menu de inclusão de novas bibliotecas. As bibliotecas são códigos que adicionam funções ao programa para facilitar a utilização de algum componente.
- Ferramentas, contém algumas ferramentas uteis para a programação, como o monitor serial. E o mais importante, é por meio deste menu que se configura qual placa se está usando e em qual porta.

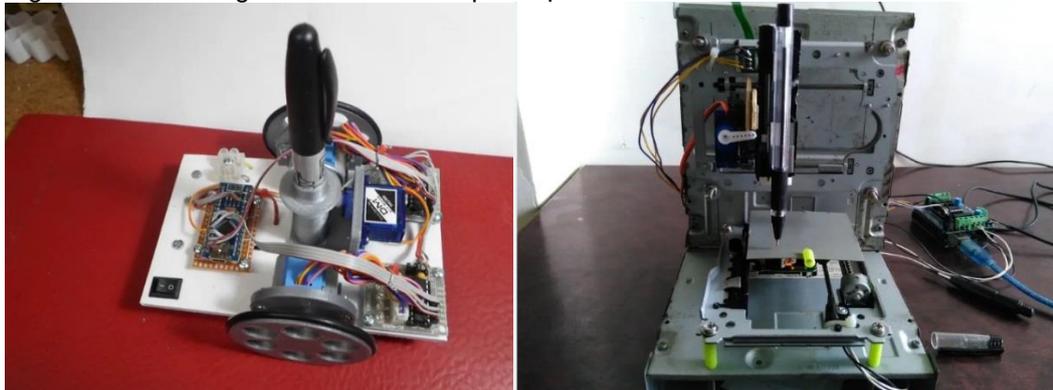
O Console de feedback é onde o programa informa o status da compilação, onde ao término da compilação aparece um relatório sobre o código na parte preta, que contém o espaço ocupado pelo código nas memórias do Arduino, se apresentar algum erro de construção, o mesmo é exibido por meio dessa tela. Acredito que essa apresentação básica proporcione a noção necessária para entender o código no capítulo de programação do Jabuti geométrico mais à frente.

6 CONSTRUÇÃO DO JABUTI GEOMÉTRICO

Classifico o processo de construção desse robô como uma aventura longa, exaustiva, propensa a diversidades, estressante, enfim simplesmente uma atividade incrível para a aprendizagem significativa que em minha opinião todos deveriam experimentar.

Quando entreguei o pré-projeto do TCC estava focado em selecionar projetos de robótica prontos e testar suas potencialidades no ensino de matemática. Mas, durante as férias ao entrar em contato com uma versão atual da tartaruga de Papert me veio à mente as possibilidades daquela máquina no ensino. Porém o hardware como se apresentava era limitado com 2 GDL, a translação em X e rotação em torno de Z, como visto no capítulo de introdução a robótica isso representa um obstáculo na movimentação. Conheci simultaneamente as máquinas CNC¹⁸ caseiras de plotagem 2D, elas apresentavam a translação em X e Y contendo assim 2GDL, com uma área bem limitada, cerca de 4 cm. Pensei que uma fusão das duas máquinas seria possível e que assim teria um projeto perfeito, com controle sobre os 3GDL do plano 2D, para ensinar geometria, removendo as limitações da tartaruga de Papert e implementando a liberdade da máquina plotadora.

Figura 28 – Tartaruga atualizada de Papert e plotadora caseira 2D



Fonte: Pavleski (2019)

Voltei das férias com a ideia de mudar o projeto de TCC, o orientador primeiramente resistiu a mesma, mas com muita insistência ele cedeu. Assim começava a experiência mais desgastante e fantástica de minha vida acadêmica até o momento e que com certeza mais aprendi.

¹⁸ Referente a expressão Controle Numérico Computadorizado.

O problema estava tanto nas informações na internet sobre construção de robô, que são muito fragmentadas, quanto em minha própria falta de experiência na área, em decorrência disso tive problemas com meus planejamentos. Como um amador, tive que refazer a estrutura do robô muitas vezes para chegar em um modelo consistente. O orientador teve que pedir algumas peças faltando um mês para o término do prazo para a entrega do trabalho. E além disso tive problemas em relação a placa controladora, classifiquei como incompatibilidade de hardwares.

Mas como disse no início do trabalho, sou um entusiasta das novas tecnologias e mesmo esses problemas não me abalaram e continuei contornando as adversidades com criatividade e disposição, a finalização do robô propriamente dito ocorreu faltando alguns dias para encerrar o prazo da entrega do trabalho e por isso adianto desculpas quanto a partes que não ficarem claras. Sem me alongar mais, passarei a descrever o processo de fabricação do Jabuti Geométrico e suas adversidades.

6.1 Lista de materiais

Alguns materiais que usei nesse projeto não são fáceis de se encontrar, farei uma tabela de materiais, contendo os preços dos equipamentos importados que foram utilizados no projeto e outra coluna com o preço das peças que podem ser compradas nacionalmente para fornecer dados para comparar o preço total do projeto nos dois casos.

Tabela 01 – Materiais usados e segunda opção

| Qtd. | Preço Importados | Preço | Qtd. | Preço Nacional | Preço |
|--------------|-------------------------|--------------|--------------|------------------------|---------------|
| 03 | Motor 28BYJ-48 + Drive | 22,11 | 03 | Motor 28BYJ-48 + Drive | 45,00 |
| 01 | Micro servo SG90 | 5,33 | 01 | Micro servo SG90 | 10,70 |
| 01 | Arduino Uno | 13,98 | 01 | Arduino Uno | 22,00 |
| 01 | Kit correia e polia | 00,00 | 01 | Kit correia e polia | 14,50 |
| 01 | Barra de aço 4mm | 2,50 | 01 | Barra de aço 4mm | 11,97 |
| Total | | 43,92 | Total | | 106,67 |

Fonte: o autor

Optei por comprar um kit importando, a tabela da direita exibe o preço dos produtos se fossem comprados nacionalmente, o total evidencia as diferenças entre

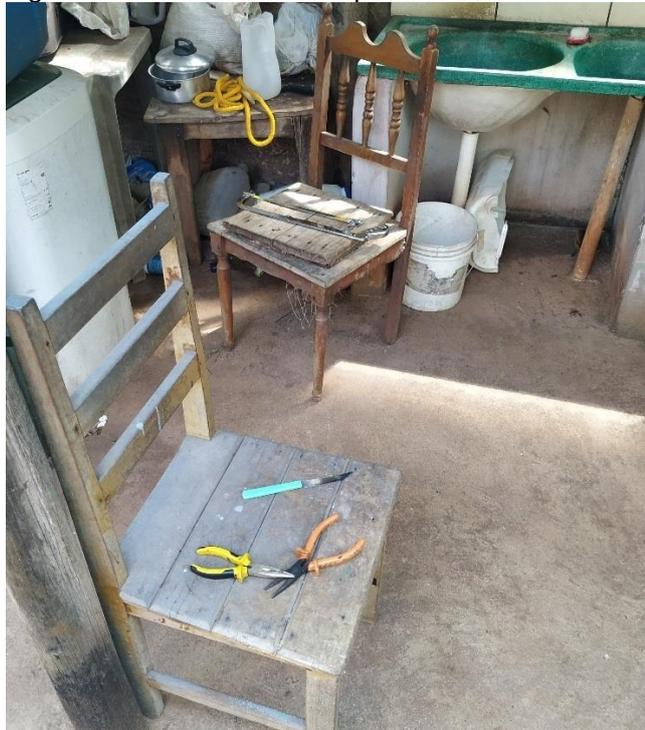
o gasto total com o projeto, importar as peças apresenta uma redução superior a 50% do gasto total, aumentando o custo benefício do projeto.

Os outros materiais como: Acrílico pro chassi, estrutura do eixo da caneta, parafusos e a peça para movimentar a caneta no eixo Z foram retiradas de sucatas, e são fáceis de localizar. Eu coloquei o preço da barra que usei da Norte Inox, porém não tive gastos com a mesma, uma vez que o vendedor me deu ela de presente.

6.2 As Ferramentas e bancada de trabalho

Atualmente a maioria dos robôs caseiros, são produzidos por meio de impressoras 3D ou maquinas CNC de precisão para corte e fabricação de peças. Porém, não tive acesso a nenhum desses equipamentos, na realidade, não me dispunha de muitas ferramentas, então acho importante exibir meu arsenal de ferramentas para que possa ficar registrado que qualquer um com muita disposição e entusiasmo consegue criar um robô hoje em dia.

Figura 29 – Área de trabalho pesado



Fonte: o autor

Como a imagem sugere, quando precisava fazer cortes em peças grandes ou peças metálicas, tinha um espaço no quintal da Kitnet¹⁹ que alugo, para efetuar os

¹⁹ Kitnets são imóveis bem populares no Brasil e possuem uma estrutura básica: um cômodo que une quarto e sala, banheiro e uma pequena cozinha.

cortes, contava apenas com um arco de serra, dois alicates e a *faquinha faz tudo*, essa faca de serra realmente participou mais do processo de construção que qualquer outra ferramenta. Como não tinha uma morsa para segurar as peças, muitas vezes eu tive que me colocar em diversas posições com os pés e mãos para apoiar as peças na cadeira que mais pareciam ensaios do Kama Sutra. E ainda assim alguns cortes eram bem difíceis de se realizar, alguns arranhões acabaram acontecendo no processo.

Uma vez cortadas as peças primárias ou metálicas, eu tinha uma área específica para fazer a montagem e estruturação, design, programação e testes. Tudo acontecia em minha escrivaninha no meu quarto.

Figura 30 – Área para montagem, testes e programação



Fonte: o autor

As ferramentas que eu usava contavam com chave de fenda e o kit de ferramentas de 31 chaves (o objeto amarelo na foto), o ferro de solda que era usado tanto para furar peças de plástico quanto para soldagem de fios, as fitas isolante e dupla face, cola Super Bonder, o multímetro e furadeira, para testes, ajustes finais e furos em peças metálicas e por fim mais uma vez a *faquinha faz tudo* ela era usada pra cortes pequenos, eu a esquentava no fogão e executava os cortes de modo a obter peças que seriam muito difíceis de se fazer pelo arco de serra, além de adaptações em peças já existentes.

Como descrito, eu não contava com um aparato de ferramentas de ponta e novas tecnologias para a construção do robô, mas isso era apenas um contratempo contornado com novas ideias e criatividade para fabricar partes mecânicas que

desempenhassem determinada função para o projeto. As peças eram tiradas e adaptadas de sucatas de impressora e unidade de leitora de CD/DVD principalmente. Como dito no início foi uma aventura por um caminho cheio de pedras e espinho para um ambiente repleto de aprendizagem e experiências enriquecedoras.

6.3 O Chassi do Jabuti

O chassi é o esqueleto dos carros robóticos, e dessa forma eles precisam ser projetados para suportar todos os componentes sem sofrer deformações, pois pode implicar em defeitos no sistema mecânico todo. Para ser sincero embora soubesse dessa informação estava tão animado em projetar o robô que ignorei completamente a mesma, criando verdadeiras atrocidades iniciais.

Eu tinha um vasto arsenal de acrílico, retirados de tanquinhos, da impressora, modems, unidade leitora de CD/DVD que inclusive para esse projeto é necessário pelo menos duas, para retirar a peça que movimenta o eixo da caneta no eixo X e Z, embora possa encontrar ou fabricar peças para efetuar a mesma função.

A ideia principal do projeto foi atualizar a tartaruga de Papert com um eixo a mais para dar maior liberdade e flexibilidade na construção de figuras geométricas. Por isso toda a estrutura deveria ser construída em torno do funcionamento dessa peça.

Figura 31 – Eixo da caneta inicial

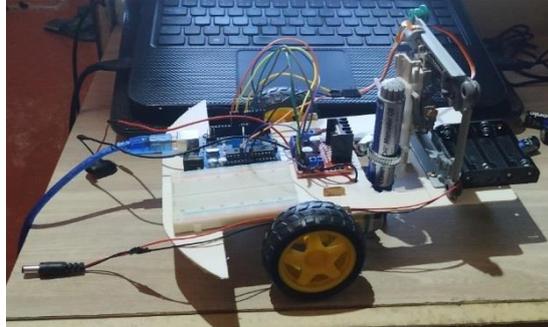


Fonte o autor

Como a peça para mover a caneta era tirada de uma unidade leitora e gravadora de CD/DVD antiga, baseada na plotadora 2D, eu estava com a mente fechada apenas para ela como possível para efetuar essa ação, nas primeiras versões não pensei em modifica-la para facilitar o desenvolvimento do chassi. E em meus projetos iniciais eu fazia buracos de 6 a 8 cm no chassi para fixação. O modelo do chassi foi copiado dos 2WD comumente vendidos para projetos variados de robótica. O projeto inicial estava sendo construído com motor cc, não tinha acesso ainda aos motores passo. Em

termos de organização do sistema e pela fragilidade do chassi, acabei abandonando essa versão.

Figura 32 – Protótipo inicial



Fonte: o autor

Encontrei então na minha caixa de sucata a bandeja da impressora com um simples corte para adaptação ela encaixou perfeitamente como novo chassi, mas ainda continuava focado na peça da plotadora 2D.

Figura 33 – Chassi bandeja de impressora

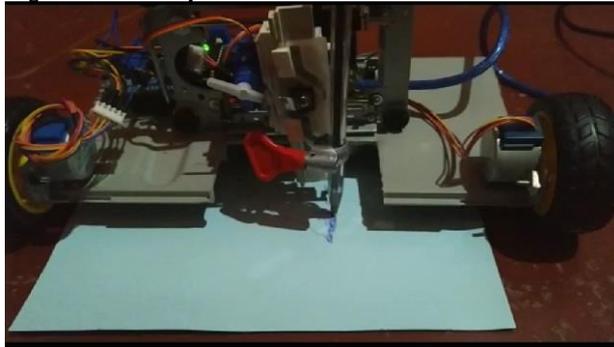


Fonte: o autor

Nesta versão recebi do orientador os equipamentos eletrônicos que faltavam para montar o robô, ou seja, os motores de passo e a placa controladora. Ao terminar a montagem não fiquei satisfeito com a área de plotagem do projeto, estava mais reduzida que o normal, mesmo o robô sendo móvel acredito que não iria alterar muito as limitações da tartaruga de Papert.

Além disso a plotagem estava toda bagunçada, fazendo com que eu chamasse o projeto de Pipira, em analogia ao pássaro que ao se alimentar faz um bom estrago na fruta e não come a mesma por inteiro.

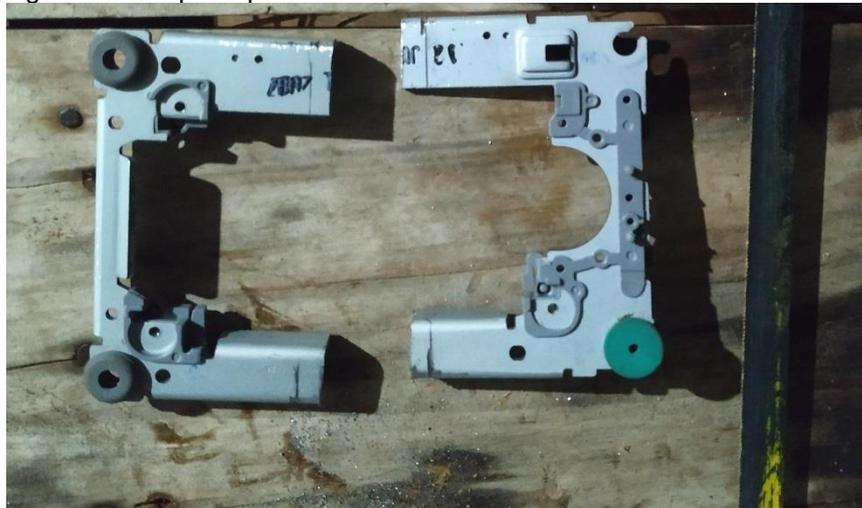
Figura 34 – Projeto com funcionamento errado



Fonte: o autor

Como não estava satisfeito com o resultado decidi finalmente fazer alterações na peça do leitor de DVD.

Figura 35 – Suporte para eixo da caneta versão 3



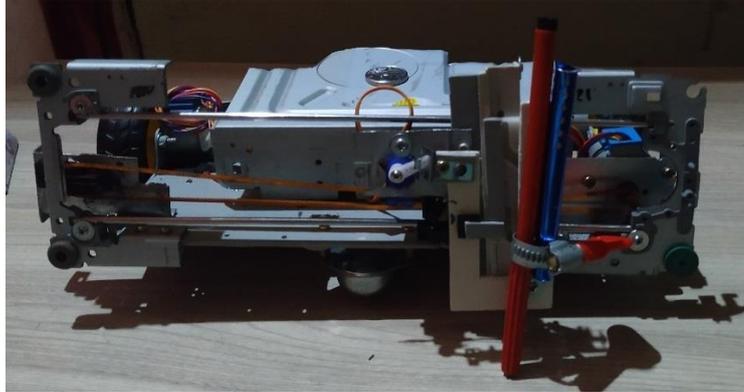
Fonte: o autor

Cortei a peça ao meio para expandir a área de plotagem. E isso me trouxe o problema ter de encontrar uma barra de aço de 4mm para substituir as barras da peça original. Passei a manhã toda circulando de ferraria em ferraria em Araguaína (TO) para finalmente encontrar, a bendita cuja, na Norte Inox, quando contei ao atendente que precisava da barra para fazer um projeto robótico para a universidade, ele decidiu nem me cobrar pela mesma, mesmo o preço da barra sendo baixo fiquei impressionado pela ação do atendente. Utilizando como suporte a estrutura de aço do leitor de DVD, fixando as barras de aço para movimentar a peça centra eu pude desenvolver o eixo estendido da caneta.

Assim finalmente eu tinha todo o necessário para montar a nova versão do projeto que chamei de pipira versão estendida. A área de plotagem passou de 4 cm para 23 cm no eixo X (movimentação da caneta). Porém ao término da montagem,

notei um problema de planejamento, que pode ser acompanhado visualmente pela imagem abaixo.

Figura 36 – Pipira versão estendida



Fonte: o autor

Falei anteriormente sobre a importância de projetar o chassi para suportar a estrutura, e mais uma vez volto ao assunto. Se olhar bem para imagem verá que pipira está desequilibrado em relação aos seus lados, isso ocorre pelo chassi ser longo ou o acrílico fino e frágil para suportar o vasto peso da peça de metal na frente, além dos motores, baterias e demais componentes. E mais uma vez tive que fazer uma alteração.

A organização anterior tinha me agradado, em razão disto, decidi que queria o chassi no mesmo formato só que menor, e foi isso que fiz, cortei mais uma parte da impressora, fazendo um retângulo de dimensões 20 cm de largura e 11 cm de comprimento. Além disso substituí de vez os pesados suportes de metal, por plásticos leves, chegando a um bom resultado com cerca de 18 cm de área de plotagem.

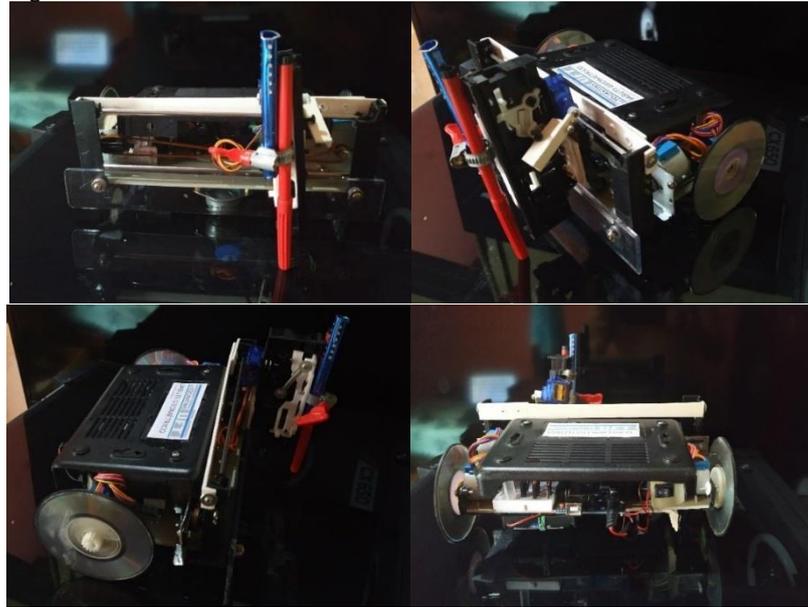
Figura 37 – Chassi versão 3



Fonte: o autor

Os buracos são para fixação das demais peças, e a parte do interruptor é fixa com cola Super Bonder. O resultado para esse chassi pode ser visto na imagem abaixo.

Figura 38 – Jabuti Geométrico versão 3



Fonte: o autor

Pela aparência dessa versão, alterei o nome do projeto para Jabuti²⁰ pois, sua silhueta lembrava a do animal, para quem tem mente aberta é claro. Além disso é como se eu estivesse anunciando essa é a versão brasileira da tartaruga de Papert.

Porém o eixo da caneta e a localização dos motores se tornou um problema para o controle das construções. O eixo da caneta deve ficar alinhado com as rodas para ter um controle maior do traço, e assim fiz a última versão do chassi um retângulo de 19,5 cm de largura e 15 cm de comprimento, que mais à frente ainda será apresentada uma modificação na estrutura do mesmo.

Figura 39 – Chassi final sem modificação



Fonte: o autor

Em decorrência dos atuadores e efetadores tive que fazer um corte no chassi que será mostrado a frente.

²⁰ Jabuti é a designação usual, utilizada no Brasil, para duas espécies de répteis providos de carapaça, exclusivamente terrestres, nativos da América do Sul.

6.4 Os Atuadores

Os atuadores precisariam ter força para movimentar toda a estrutura do sistema robótico e bom grau de precisão nos movimentos, para poder ter o controle do traço. Em decorrência disso os motores de passo 28BYJ-48 e o Micro servo SG90 são ótimos modelos para desempenharem essa função, por atenderem as necessidades do projeto e apresentar um preço bem reduzido em relação aos demais modelos. Mas, alguns atuadores auxiliares também são necessários ao projeto.

6.4.1 Atuadores das rodas

Os dois atuadores das rodas são ativos e funcionam de forma independente, tive que construir uma peça de plástico para fixação dos motores no chassi, adaptei de outro equipamento em minha caixa de sucatas por meio do arco de serra, faca quente e ferro de solda. Note que essa peça pode ser fabricada a partir de qualquer sucata que tenha ângulo de 90 graus entre seus lados, basta fazer adaptações.

Figura 40 – Atuadores das rodas do Jabuti Geométrico



Fonte: o autor

A peça deve ficar bem presa ao motor qualquer folga ou inclinação irá afetar o desempenho das rodas e bom funcionamento de todo o robô.

6.4.2 Atuadores do eixo da caneta

Já os atuadores do eixo da caneta apresentam características passivas e ativas, adaptei peças metálicas da impressora, a função desses atuadores é de exclusivamente proporcionar a força para movimentar a caneta pelo eixo X, como a peça é leve, e foi bem regulada não exige muita potência, mas a precisão como nas rodas é essencial.

O mecanismo consiste em um suporte para o motor (atuador ativo), outro para a polia (atuador passivo) e mais um da correia (atuador passivo) para puxar o suporte

da caneta (atuador passivo) e fazê-lo se mover no eixo. Os suportes foram retirados da impressora da peça que movia a cabeça de impressão.

Figura 41 – Atuadores ativos e passivos do eixo da caneta



Fonte: o autor

Não estou comentando sobre as dificuldades de construir tais peças pois, acredito que com a descrição das ferramentas disponíveis, já esteja subentendido o quão complicado foi adaptar as mesmas. Irei deixar pra exibir o atuador de baixar e levantar a caneta para o próximo tópico onde apresento os efetadores.

6.5 Os Efetuadores

Os efetadores nesse projeto robótico são as rodas e o dispositivo da caneta. O primeiro e mais fácil de se construir são as rodas, basicamente são dois minis CD's colados um no outro e uma liga de borracha colada em sua extremidade. Utilizei a cola Super Bonder para a fixação.

Figura 42 – Rodas do Jabuti



Fonte o autor

Para fixar as rodas no eixo do motor de passo colei engrenagens, retiradas da impressora, no centro dos minis CD's, de modo a deixar bem justo as rodas e não perder precisão de giro. O mini CD tem 8 cm de diâmetro, e o eixo do motor encaixa no ponto central, isso implica que o robô possui uma suspensão de cerca 4 cm em relação ao solo.

6.5.1 Atuador e Efetuador da caneta

Essa parte teve várias versões ao longo do desenvolvimento do projeto. Esta última consiste em duas bases de plástico rígido para suportar toda a estrutura, a barra de aço de 4 mm doada pela Norte Inox, o suporte do leitor de CD para movimentar a caneta no eixo, uma abraçadeira regulável para prender e soltar a caneta, peças de plástico de fixação e suporte para toda a estrutura e o Micro Servo.

Figura 43 – O eixo da caneta

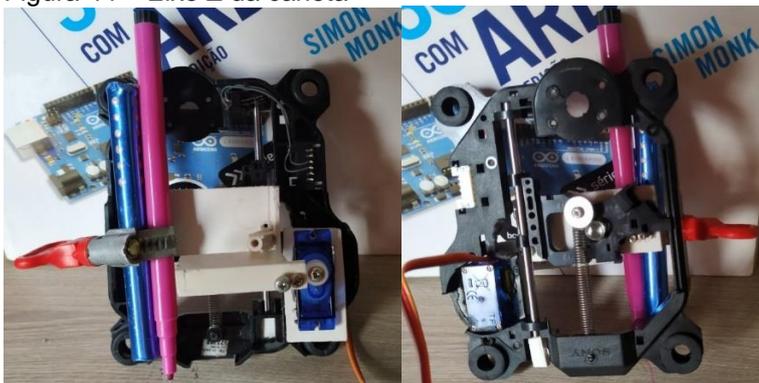


Fonte: o autor

Tive que cortar a barra várias vezes, adaptando a cada versão feita anteriormente, trabalho desgastante sem as ferramentas certas. Nesta versão os dois segmentos de aço possuem 17 cm. Os suportes laterais são de um leitor de DVD antigo, cortei ao meio e adaptei para fixar as barras e prender no chassi do robô.

O modo de operação é simples, o atuador ativo exerce força na correia que está presa no suporte da caneta, este está muito bem calibrado para poder correr livremente pelas barras, a correia por sua vez está firmemente conectada a polia podendo concretizar seu movimento de rotação. Assim o atuador ativo pode movimentar a caneta nos dois sentidos do eixo, bastando alterar a rotação do motor. Além disso existe a peça para mover a caneta no eixo Z retirada da unidade leitora de CD/DVD.

Figura 44 – Eixo Z da caneta



Fonte: o autor

A parte fixa de plástico preta e a móvel branca onde a abraçadeira está aparafusada, foram retiradas do mecanismo do leitor de CD/DVD, assim trata-se de uma reutilização e adaptação de uma peça que iria para o lixo. O Servo tem a função de levantar a peça branca ao mudar de ângulo levando sua hélice ao encontro do parafuso que está fixo no plástico branco fazendo subir, ao alterar o ângulo do servo motor novamente a peça irá descer, para garantir isso coloquei uma mola que puxa a peça pra baixo. Assim temos o movimento da caneta no eixo X e Z. A responsabilidade de movimentar no eixo Y é inteiramente das rodas e seus atuadores.

6.6 Os Controladores

A descrição dos controladores do projeto já fora feita, mas ainda precisava fixar no chassi e garantir que fosse possível fazer as ligações necessárias entre os mesmos. Com isso para fixar os controladores utilizei parafusos dos equipamentos eletrônicos que desmontei. Para fazer as ligações usei uma mini protoboard.

Se observar a descrição dos componentes desse projeto robótico irão perceber que o Arduino não tem portas suficientes para energizar e aterrar todos. Por isso utilizei a mini protoboard para fazer o processo de energização.

Figura 45 – Mini protoboard



Fonte: o autor

Muito mais poderia ser explicado. Mas, essa apresentação superficial do sistema mecânico já é suficiente para entender o processo de montagem.

6.7 Montagem do Jabuti Geométrico

Com todo o exposto acima é hora de dar forma ao projeto. O processo de montagem inicia com os atuadores das rodas na parte dianteira no chassi, depois aparafuso o eixo da caneta na parte frontal, em seguida fixo os atuadores passivos e ativos do eixo na parte traseira, para assim deixar a correia bem esticada. E então basta fazer a disposição dos controladores e da mini protoboard no espaço restante. A fixação dos componentes no chassi pode ser observada na imagem a seguir.

Em decorrência da caneta ter de ficar em linha reta com as rodas tive que fazer um corte de 3 cm no chassi como pode ser observado na imagem abaixo. O projeto pode ser alimentado tanto por pilhas quanto por bateria de 9V, o recomendado é que compre recarregável pois se trata de um sistema com três motores e um Micro servo, e por isso, consome uma carga considerável. Se usar carregadores portátil de celular poderá energizar o projeto pela porta USB.

Figura 46 – Estrutura mecânica montada

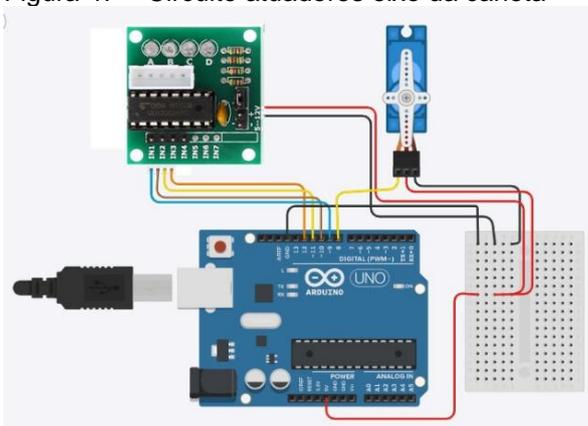


Fonte: o autor

Com todos os esquemas mecânicos montados é hora de fazer as ligações utilizei Jumpers²¹ coloridos para realizar as mesmas, inicialmente tentei padronizar as cores, mas como me faltaram materiais acabei modificando as cores em alguns circuitos.

Para ficar bem fácil o entendimento das ligações fiz um esquema de circuito eletrônico didático, dividindo em esquema dos atuadores do eixo da caneta e atuadores das rodas, para chegar à ligação completa basta efetuar todas as ligações no mesmo circuito, entenderá o motivo disso mais na frente.

Figura 47 – Circuito atuadores eixo da caneta



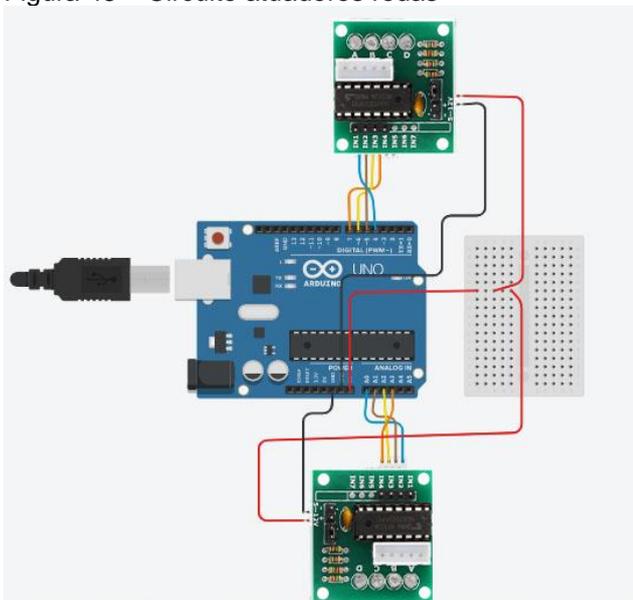
Fonte: o autor

²¹ São pequenos pedaços de fios que servem para fazer conexões entre as portas dos equipamentos eletrônicos.

Não tem como errar a conexão do motor com o driver ULN2003 uma vez que possui cavas que permite apenas a conexão correta por isso exibo na imagem apenas o driver. Note que eu conectei os pinos IN1, IN2, IN3 e IN4 em ordem crescente na placa, nos pinos 9, 10, 11 e 12 respectivamente, significa que irei programar a sequência correta de ativação do motor pelo IDE. O fio para controle do Micro servo é o 8. As conexões de energia dos dois atuadores são feitas pelo pino de 5V, pois eles não necessitam de muita potência para mover a caneta.

O circuito das rodas possui a mesma ligação do motor na etapa anterior, mas usando portas diferentes. Para a roda esquerda a conexão é feita nos pinos 4, 5, 6 e 7 e nas rodas direitas os pinos são A0, A1, A2 e A3 que possuem referência numérica 14,15,16 e 17 respectivamente no Arduino.

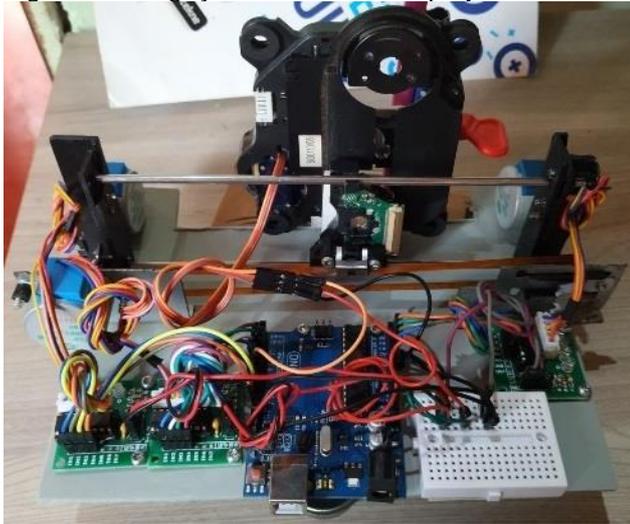
Figura 48 – Circuito atuadores rodas



Fonte: o autor

Agora fica aqui explícito o motivo de ter separado as ligações em partes. Para evitar que o esquema fique tão bagunçado quanto as ligações que fiz no robô. Embora fosse visualmente interessante organizar os fios, digamos que pela minha falta de tempo essa seja a melhor organização possível no momento.

Figura 49 – Ligações realizadas no projeto



Fonte: o autor

Dessa forma o projeto já está completamente funcional, se energizar e programar é claro, mas ainda falta a estética, então adicionei o casco para finalmente apresentar a versão final do Jabuti Geométrico, como segue na imagem.

Figura 50 – Jabuti geométrico versão final



Fonte: o autor

Faltando apenas programar o robô, para que ele execute as funções projetadas no processo de construção do mesmo. por isso no próximo capítulo apresento o programa do jabuti pôr extenso para não estender muito o trabalho.

7 PROGRAMAÇÃO DO JABUTI GEOMÉTRICO

A programação do projeto foi feita de forma sucinta, irei exibir o código completo, pois, programei as partes de forma didática, com fácil identificação e assimilação do que cada parte faz para passar para a análise e comparação da geometria possível do Jabuti geométrico. O código inteiro pode ser copiado a seguir para utilização ou modificação.

```
//=====
//=====Início do Programa do Jabuti=====
// Lista de bibliotecas
#include <Servo.h>
#include <AFMotor.h>
//=====
//-----#Configurações Globais#-----
// setup servo
int servoPin = 8;
int RISCAR = 70; // angulo do servo quando a caneta está baixa
int NAO_RISCAR = 20; // angulo do servo quando a caneta está alta
Servo caneta;
//fim do setup servo

//Especificações do robô
float diam_roda=80; // # diâmetro da roda em milímetro
float base_roda=220; // # Distância entre as duas rodas em milímetro
int passo_rev=512; // # 512 para 64x na caixa de velocidade, 128 para
16x na caixa de velocidade
int delay_time=2; // #tempo entre etapas em microssegundos
//Fim das especificações

// Definição dos motores atuadores
int MotorE[] = {4, 5, 6, 7}; //Motor da roda esquerda
int MotorD[] = {14, 15, 16, 17}; // Motor da Roda direita
int MotorC[] = {9, 10, 11, 12}; // Motor da Caneta
//fim da definição

//Sequência de ativação motores
int fwd_mask[][4] = {{1, 1, 0, 0},
                    {0, 1, 1, 0},
                    {0, 0, 1, 1},
                    {1, 0, 0, 1}};

int rev_mask[][4] = {{1, 0, 0, 1},
                    {0, 0, 1, 1},
                    {0, 1, 1, 0},
                    {1, 1, 0, 0}};
//Fim da definição de sequência

//Setup para a inicialização do Jabuti
void setup() {
  randomSeed(analogRead(1));
  Serial.begin(9600);
  for(int pin=0; pin<4; pin++){
    pinMode(MotorE[pin], OUTPUT);
    digitalWrite(MotorE[pin], LOW);
  }
}
```

```

    pinMode(MotorD[pin], OUTPUT);
    digitalWrite(MotorD[pin], LOW);
    pinMode(MotorC[pin], OUTPUT);
    digitalWrite(MotorC[pin], LOW);
}
caneta.attach(servoPin);
Serial.println("setup");
caminha();
nao_riscar();
delay(1000);
}
//Fim do setup de inicialização

//=====
//====Área de manipulação livre do Jabuti=====
void caminha(){
//Área de acesso do aluno

    fim();      // libera o motor de passo
    while(1);   // aguarda o reset
}

//=====Fim da área de manipulação livre=====
//=====
//=====manipulação com repetição do robô=====
void loop(){
    //Área de acesso do aluno

}

//=====Fim da área de manipulação com repetição=====
//=====
//=====Área restrita para a criação de funções globais=====

// ----- Funções Globais -----

//Definição da distância que o robô irá percorrer
int step(float distancia){
    int passos = distancia * passo_rev / (diam_roda * 3.1412); //25.12
    /*
    Serial.print(distancia);
    Serial.print(" ");
    Serial.print(passo_rev);
    Serial.print(" ");
    Serial.print(diam_roda);
    Serial.print(" ");
    Serial.println(passos);
    delay(1000);*/
    return passos;
}
//fim da função definição de distância

// Função de movimento para frente das rodas
void para frente(float distancia){
    int passos = step(distancia);
    Serial.println(passos);
    for(int step=0; step<passos; step++){
        for(int mask=0; mask<4; mask++){
            for(int pin=0; pin<4; pin++){

```

```

        digitalWrite(MotorE[pin], rev_mask[mask][pin]);
        digitalWrite(MotorD[pin], fwd_mask[mask][pin]);
    }
    delay(delay_time);
}
}
}
//fim da função de movimento para frente

//função de movimento para trás
void paratras(float distancia){
    int passos = step(distancia);
    for(int step=0; step<passos; step++){
        for(int mask=0; mask<4; mask++){
            for(int pin=0; pin<4; pin++){
                digitalWrite(MotorE[pin], fwd_mask[mask][pin]);
                digitalWrite(MotorD[pin], rev_mask[mask][pin]);
            }
            delay(delay_time);
        }
    }
}
//fim da função de movimento para trás

///função de rotação para direita
void direita(float degrees){
    float volta = degrees / 360.0;
    float distancia = base_roda * 3.1412 * volta;
    int passos = step(distancia);
    for(int step=0; step<passos; step++){
        for(int mask=0; mask<4; mask++){
            for(int pin=0; pin<4; pin++){
                digitalWrite(MotorE[pin], rev_mask[mask][pin]);
                digitalWrite(MotorD[pin], rev_mask[mask][pin]);
            }
            delay(delay_time);
        }
    }
}
//fim da função de rotação para direita

//Função de rotação para esquerda
void esquerda(float degrees){
    float volta = degrees / 360.0;
    float distancia = base_roda * 3.1412 * volta;
    int passos = step(distancia);
    for(int step=0; step<passos; step++){
        for(int mask=0; mask<4; mask++){
            for(int pin=0; pin<4; pin++){
                digitalWrite(MotorE[pin], fwd_mask[mask][pin]);
                digitalWrite(MotorD[pin], fwd_mask[mask][pin]);
            }
            delay(delay_time);
        }
    }
}
//fim da função de rotação para esquerda

//Função para desativar os motores e descansar a bateria
void fim(){ // unlock stepper to save battery
    for(int mask=0; mask<4; mask++){

```

```

        for(int pin=0; pin<4; pin++){
            digitalWrite(MotorE[pin], LOW);
            digitalWrite(MotorD[pin], LOW);
        }
        delay(delay_time);
    }
}
//fim da função de desativar os motores

//Função para movimentar eixo da caneta sentido positivo
void movecanetaP(float distancia){
    int passos = step(distancia);
    Serial.println(passos);
    for(int step=0; step<passos; step++){
        for(int mask=0; mask<4; mask++){
            for(int pin=0; pin<4; pin++){
                digitalWrite(MotorC[pin], rev_mask[mask][pin]);
            }
            delay(delay_time);
        }
    }
}
//fim da função de movimentar eixo da caneta sentido positivo

//Função para movimentar o eixo da caneta sentido negativo
void movecanetaN(float distancia){
    int passos = step(distancia);
    Serial.println(passos);
    for(int step=0; step<passos; step++){
        for(int mask=0; mask<4; mask++){
            for(int pin=0; pin<4; pin++){
                digitalWrite(MotorC[pin], fwd_mask[mask][pin]);
            }
            delay(delay_time);
        }
    }
}
//fim da função de movimentar o eixo da caneta sentido negativo

//função para movimentar a caneta para baixo
void riscar(){
    delay(250);
    Serial.println("RISCAR()");
    caneta.write(RISCAR);
    delay(250);
}
//fim da função de movimentar a caneta para baixo

//Função para movimentar a caneta para cima
void nao_riscar(){
    delay(250);
    Serial.println("NAO_RISCAR()");
    caneta.write(NAO_RISCAR);
    delay(250);
}
//fim da função de alimentar a caneta para baixo
//=====Fim do programa=====
//=====

```

Um programa com a interface que o aluno terá acesso ainda será projetado. Os comandos foram baseados na linguagem Logo com algumas alterações pertinentes. Os comandos que o Jabuti compreende são:

- **riscar()** abaixa a caneta e faz a estrutura riscar o papel;
- **nao_riscar()** sobe a caneta fazendo com que o movimento não risque o papel;
- **parafrente()** comanda o robô para andar para frente, os parênteses devem ser preenchidos com a distância da movimentação desejada;
- **paratras()** comando para mover para trás, os parênteses devem ser preenchidos com a distância da movimentação desejada;
- **direita()** faz o Jabuti rotacionar para direita, o parêntese deve ser preenchido com o grau da rotação desejada;
- **esquerda()** informa ao robô que deve se rotacionar para a esquerda, o parêntese deve ser preenchido com o grau da rotação desejada;
- **movecanetaP()** movimenta a caneta no sentido positivo.
- **movecanetaN()** movimenta a caneta no sentido negativo.
- **fim()** informa que acabou o movimento e desativa os motores para poupar a bateria.

Os códigos podem ser facilmente alterados para uma linguagem de fácil assimilação dos alunos. Além disso, o IDE possibilita a transição da linguagem de programação Logo para a C++ e dessa forma o aluno começa aprendendo por um micromundo simples inicial proporcionado pela Logo e posteriormente, dependendo do professor, pode ir fazendo a transição para linguagem mais complexas e assimilando a lógica das programações para aplicar em outras áreas, o IDE do Arduino tem a capacidade de imergir o aluno em um micromundo e o levar a transitar para fora dele com conhecimentos conceituais aplicáveis em outras áreas. Como dito acima ainda desenvolverei um software para o contato dos alunos com o Jabuti, por enquanto está exposto o resultado primário da programação.

8 ANÁLISE DO JABUTI GEOMÉTRICO

8.1 Jabuti Geométrico e suas características

O Jabuti tem uma estrutura física, ele funciona com base em um sistema por realimentação aberta, em outras palavras, é um sistema mecânico semiautônomo, onde após programar o controlador, ele irá executar a tarefa sem o auxílio de ações humanas. Com isso ele pode agir no ambiente para alcançar objetivos pré-programados. Porém, ele não possui sensor. Com base na definição de robô do capítulo 4, classifico o Jabuti como um projeto robótico semiautônomo, não entrando especificamente na categoria, optei por classifica-lo como um robô em desenvolvimento.

O jabuti apresenta o mesmo grau de liberdade que um carro para a movimentação, por isso, ele apresenta movimentação não holonômica, ver capítulo 4. Isso significa que tem limitações quanto ao seu deslocamento no plano. Quanto a manipulação da caneta ele tem controle sobre os três GDL possíveis para o plano 2D e por isso ele apresenta manipulação holonômica. Isso quer dizer que tem um amplo controle sobre a manipulação da mesma, fornecendo ótimos resultados e algumas flexibilidades para plotagem. Em comparação com a Tartaruga de Papert que possui apenas 2 GDL essa é a primeira e vital ampliação que foi obtida com essa estrutura mecânica

A caneta pode ser trocada e regulada para diferentes alturas. Isso permite que o Jabuti plote em áreas não planas, embora a área onde as rodas do jabuti passem deva ser plana. O limite de plotagem no eixo X é 10 cm, que é até onde o eixo da caneta pode se mover, porém o eixo Y das rodas está livre para levar a estrutura para qualquer local e posição ampliando a área de plotagem consideravelmente.

8.2 Distinções entre a Tartaruga de Papert e o jabuti Geométrico

Como apresentado no Capítulo 3 a tartaruga de Papert apresenta uma limitação conceitual para a construção da circunferência, onde segundo Dolce e Pompeo (2005, p.147) "Circunferência é um conjunto dos pontos de um plano cuja distância a um ponto dado desse plano é igual a uma distância (não nula) dada. O ponto dado é o centro e a distância dada é o raio da circunferência".

Vimos no terceiro capítulo que na geometria da tartaruga a circunferência é obtida através de uma variação sucessiva entre as variáveis ângulo e distância para chegar a um polígono de 360 lados que se assemelhe visualmente a circunferência.

A geometria do jabuti apresenta diferença crucial em suas construções fazendo aproximação conceitual com a geometria euclidiana. Criei um subtópico específico para analisar a construção da circunferência no Jabuti geométrico.

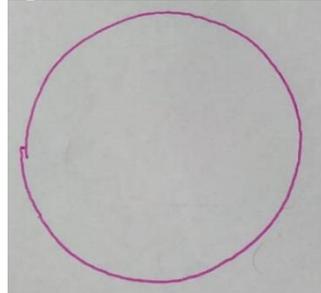
8.2.1 A circunferência do Jabuti Geométrico

Para se construir uma circunferência por meio desse robô, antes de mais nada deve definir o raio da mesma configurando o eixo da caneta para se mover a uma distância dada do centro da estrutura mecânica. Após isso basta comandar o movimento para a direita ou esquerda 360 graus para que o robô faça a rotação em torno do próprio eixo e assim tenha o conjunto de pontos equidistantes ao centro. O código explicativo para fazer isso pode ser acompanhado a seguir.

```
//=====
//====Área de manipulação livre do Jabuti=====
void caminha() {
  riscar(); //abaixa a caneta para poder riscar o papel
  movecanetaP(100); //Move a caneta 1 centímetro
  direita(360); //O Jabuti faz uma volta em torno do próprio eixo
  nao_riscar(); //Sobe a caneta para não riscar o papel
  fim(); // libera o motor de passo
  while(1); // aguarda o reset
}
//=====Fim da área de manipulação livre=====
```

Embora a circunferência não tenha fechado na imagem essa é uma falha mecânica. Em decorrência do prazo para a escrita do projeto não me disponho de tempo para resolver a mesma, nas considerações retorno a esse assunto.

Figura 51 – Circunferência pelo Jabuti



Fonte: o autor

Note que como a circunferência foi obtida por meio de uma distância dada (o raio) em relação ao centro e por revolução da estrutura para plotar pontos equidistantes em relação ao centro dado, essa construção está totalmente associada ao conceito formal da geometria euclidiana, superando as limitações conceituais da geometria da tartaruga de Papert.

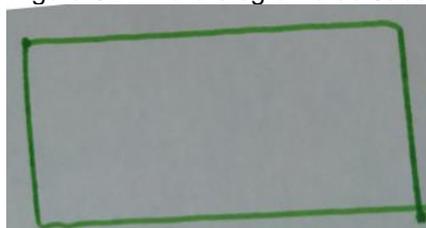
8.2.2 Outras ampliações

Em decorrência da possibilidade de construir a circunferência, em contraste com a definição da mesma, e com o movimento do eixo X da caneta se torna possível construir figuras inscritíveis, e desse modo toda a área da circunferência da geometria euclidiana pode ser trabalhada em consonância com seu conceito formal.

A estrutura robótica do jabuti por ser holonômica garante algumas flexibilidades de construções geométricas. Como os paralelogramos. Os paralelogramos podem ser obtidos pela manipulação dos ângulos e assim mantendo a mesma geometria da Tartaruga ou pode ser obtida através da plotagem de retas paralelas, por meio de uma junção de movimentos do eixo da caneta e das rodas. O código que descreve essa construção do jabuti pode ser acompanhado a seguir, assim como o resultado no papel.

```
//=====
//====Área de manipulação livre do Jabuti=====
void caminha(){
  riscar(); //abaixa a caneta para poder riscar o papel
  movecanetaP(137); //Move a caneta 3 centímetros sentido positivo
  para frente(158); //O Jabuti se move 4 centímetros para frente
  movecanetaN(137); // Move a caneta 3 centímetros sentido negativo
  para trás(158); // O Jabuti se move 4 centímetros para trás
  nao_riscar(); //Sobe a caneta para não riscar o papel
  fim(); // libera o motor de passo
  while(1); // aguarda o reset
}
//=====Fim da área de manipulação livre=====
```

Figura 52 – Paralelogramo do Jabuti



Fonte: o autor

Uma calibração será feita para contornar os erros por conta da estrutura do motor de passo. Dessa forma, traçar figuras e retas paralelas poderá ser feito de dois modos pelo Jabuti, ganhando uma flexibilidade e alternativas para construções pelo mesmo.

O objetivo desse trabalho é distinguir e validar as atualizações da tartaruga de Papert, exibindo que a remoção da limitação do sistema robótico proporcionou maior liberdade e controle ao projeto, além de torná-lo acessível as escolas, pois, se comprarem os motores no atacado ou importado, o preço apresentado terá uma redução de 50%, ampliando o custo/benefício do robô.

Um outro estudo para mostrar as variadas construções possíveis com esse projeto se fez necessário. Por enquanto essas construções realçam as ampliações do projeto. O jabuti pode desenhar as mais variadas figuras geométricas com um certo grau de flexibilidade em algumas delas, mas o essencial é que todas as construções geométricas do Jabuti estão em consonância com a definição formal facilitando assim assimilação conceitual e seu uso no cotidiano da sala de aula.

9 CONSIDERAÇÕES FINAIS

Com todo o exposto até aqui fica evidente que o Jabuti geométrico é uma atualização da Tartaruga de Papert, apresentando um eixo a mais para movimentar a caneta, ampliando o GDL do robô, fornecendo, com isso, maior flexibilidade nas construções geométricas.

O projeto está sustentado na mesma base teórica da Tartaruga, sendo assim é uma ferramenta robótica capaz de proporcionar um ambiente de aprendizagem construcionista, por meio de construções mentais para a assimilação conceitual ativa.

Além disso, está validado as ideias da filosofia Logo no Jabuti, pois, a parte do código que o aluno terá acesso é baseada na linguagem de programação desenvolvida por Papert e seus colaboradores. Dessa forma o projeto robótico desse trabalho facilita o acesso dos procedimentos mentais dos alunos nas construções por meio do IDE do Arduino, e dá horizontes para modificar o pensamento e modo de pensar, levando ao desenvolvimento cognitivo por meio de assimilações e depurações. Além disso fornece um caminho para fazer a transição da lógica presente no micromundo para um macro mundo, em relação ao anterior, da lógica de programação C++, para ampliar a área de aplicação dos conceitos aprendidos no ambiente de aprendizado proporcionado pelo robô Jabuti Geométrico e controlado pelo IDE do Arduino.

Por ser um projeto robótico o Jabuti, também, tem atributos dessa área, ou seja, ele favorece o desenvolvimento da aprendizagem por assimilações concretas, evitando o abstrato, no intuito de fortalecer a estrutura mental dos alunos por assimilações conceituais de algo manipulável que o aluno está em contato. Como a tartaruga de Grey Walter e os Veículos de Braitenberg foram usados para entender melhor o pensamento animal, o Jabuti carrega essa essência, onde não se pode chegar a um resultado dado sem conhecer completamente o que está fazendo e além disso consiga ensinar o robô como reproduzir o objeto de aprendizagem.

O Jabuti pode aproximar a geometria da tartaruga aos conceitos formais da euclidiana, além disso dá maior flexibilidade, controle e facilidade na manipulação de alguns objetos geométricos. Como fora apresentado no capítulo anterior, contornou a limitação da geometria de Papert na construção de uma circunferência, aproximando a mesma a sua noção conceitual e dando maior controle para a construção desse objeto geométrico.

O presente trabalho teve o objetivo de apresentar a atualização do sistema mecânico para transformar a estrutura limitada de Papert em uma ferramenta educacional acessível, efetiva e que favoreça a assimilação dos conceitos em contramão ao ensino instrucionista que está focado no decorar de regras e definições abstratas.

Trabalhos posteriores para aperfeiçoar a ferramenta e para analisar sua potencialidade em sala de aula se fazem necessário, e assim validar sua utilidade na assimilação de conceitos e desenvolvimento cognitivo do aluno. Estarei realizando os mesmos em um mestrado futuramente.

REFERÊNCIAS

ARDUINO. In: **Wikipédia**: a enciclopédia livre. Disponível em: < <https://pt.wikipedia.org/wiki/Arduino>>. Acesso em: 29 de novembro de 2019.

ARDUINO IDE. In: **Wikipédia**: a enciclopédia livre. Disponível em: < https://en.wikipedia.org/wiki/Arduino_IDE>. Acesso em: 29 de novembro de 2019.

DOLCE, Osvaldo; POMPEO, José Nicolau. **Fundamentos de matemática elementar** – vol. 9. 8. ed. São Paulo: Atual Editora LTDA, 2005.

MARTINS, Victor. Super Logo comandos para movimentar a tartaruga. **Slideshare**. 12 de fev de 2009. Disponível em: < <https://pt.slideshare.net/chedas/super-logo-comandos-para-movimentar-a-tartaruga/>>. Acesso em: 14 nov. 2019.

MATARIC, Maja J. **Introdução à robótica**. Tradução Humberto Ferasoli Filho, José Reinaldo Silva, Silas Franco dos Reis Alves. São Paulo: Editora Unesp/Bucher, 2014.

MATTE, Marília Luiza. **A linguagem Logo como possibilidade de aprendizagem em matemática**. 2011. Monografia (Licenciatura em matemática) – Universidade Federal do Rio Grande do Sul – UFRGS, Porto Alegre, 2000.

MONK, Simon. **30 projetos com Arduino**. 2. ed. Porto Alegre: Bookman, 2014.

MURTA, Gustavo. Guia completo do Motor de Passo 28BYJ-48 + Driver ULN2003. **Eletrogate**. 23 de julho de 2018. Disponível em: < <https://blog.eletrogate.com/?s=Guia+completo+do+Motor+de+Passo+28BYJ-48+%2B+Driver+ULN2003>>. Acesso em: 12 out. 2019.

OLIVEIRA, Euler. Como usar com Arduino Motor de Passo 28BYJ-48 com Driver ULN2003. **MasterWalker eletronic shop**. [s.d]. Disponível em: < <https://blogmasterwalkershop.com.br/?s=Como+usar+com+Arduino+%E2%80%93+Motor+de+Passo+28BYJ-48+com+Driver+ULN2003>>. Acesso em: 12 out. 2019.

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

PAPERT, Seymour. **Logo**: computadores e educação. Tradução José Armando Valente, Beatriz Bitelman, Afira Vianna Ripper. São Paulo: Editora brasiliense, 1985.

PAVLESKI, Mirko. Arduino Drawing Robot (Open Source Turtle Robot). **Arduino Project Hub**. 06 de janeiro de 2019. Disponível em: < https://create.arduino.cc/projecthub/mircemk/arduino-drawing-robot-open-source-turtle-robot-dd1878?ref=tag&ref_id=robots&offset=71>. Acesso em: 02 ago. 2019.

PIMENTEL. Perspectiva Instrucionista (computador como tutor). **Informática em educação**. [s.d]. Disponível em: <<https://sites.google.com/site/infoeducunirio/>>. Acesso em: 23 out. 2019.

ROBÔ. IN: **National Museum of American History**. Disponível em: <https://americanhistory.si.edu/collections/search/object/nmah_879329>. Acesso em: 05 dez. 2019.

SOFTWARE ARDUINO. In: **Arduino**. Disponível em: <file:///C:/Program%20Files/WindowsApps/ArduinoLLC.ArduinoIDE_1.8.21.0_x86__mdqgnx93n4wtt/reference/www.arduino.cc/en/Guide/Environment.html>. Acesso em: 28 set. 2019.

THE LOGO TURTLE. In: **Cyberneticzoo.com**. 2010. Disponível em: <<http://cyberneticzoo.com/cyberneticanimals/1969-the-logo-turtle-seymour-papert-marvin-minsky-et-al-american/>>. Acesso em: 02 dez. 2019.

VALENTE, José Armando (Org.). **O computador na sociedade do conhecimento**. Campinas, SP: Unicamp/NIED, 1999.

WERTHEIN, Jorge. A sociedade da informação e seus desafios. **Ciência da Informação**, [s.l.], v. 29, n. 2, p.71-77, ago. 2000. FapUNIFESP (SciELO). Disponível em: <<http://dx.doi.org/10.1590/s0100-19652000000200009>>. Acesso em: 10 out. 2019.

“How Does a Stepper Motor Work”, vídeo do Youtube, 5:53, postado por **Learn Engineering**, 19 de outubro de 2016. Disponível em: <<https://www.youtube.com/watch?v=eyqwLiowZiU>>. Acesso em: 21 nov. 2019.