

Universidade Federal de Mato Grosso do Sul

Instituto de Matemática

Programa de Pós-Graduação

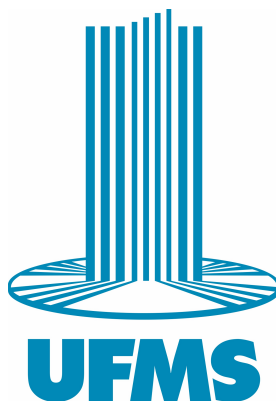
**Mestrado Profissional em
Matemática em Rede Nacional**

DONIZETH JACINTO DE SOUZA

**GEOMETRIA EM MOVIMENTO: ARDUINO E ROBÓTICA NO
ENSINO DE MATEMÁTICA**

Campo Grande - MS

Agosto - 2024



Universidade Federal de Mato Grosso do Sul

Instituto de Matemática

Programa de Pós-Graduação

Mestrado Profissional em

Matemática em Rede Nacional

DONIZETH JACINTO DE SOUZA

**GEOMETRIA EM MOVIMENTO: ARDUINO E ROBÓTICA NO
ENSINO DE MATEMÁTICA**

Orientadora: Profa. Dra. Rúbia Mara de Oliveira Santos

Dissertação apresentada ao Programa de Pós-Graduação
Mestrado Profissional em Matemática em Rede Nacional
do Instituto de Matemática da Universidade Federal de
Mato Grosso do Sul - INMA/UFMS, como parte dos re-
quisitos para obtenção do título de Mestre.

Campo Grande - MS

Agosto - 2024

GEOMETRIA EM MOVIMENTO: ARDUINO E ROBÓTICA NO ENSINO DE MATEMÁTICA

DONIZETH JACINTO DE SOUZA

Dissertação submetida ao Programa de Pós-Graduação (Mestrado Profissional em Matemática em Rede Nacional) do Instituto de Matemática da Universidade Federal de Mato Grosso do Sul - INMA/UFMS, como parte dos requisitos para obtenção do título de Mestre.

Banca examinadora:

Profa. Dra. Rúbia Mara de Oliveira Santos (Orientadora)
Universidade Federal de Mato Grosso do Sul - UFMS

Profa. Dra. Elen Viviani Pereira Spreafico
Universidade Federal de Mato Grosso do Sul - UFMS

Profa. Dra. Irene Magalhães Craveiro
Universidade Federal da Grande Dourados - UFGD

Campo Grande - MS
Agosto - 2024

Dedico este trabalho à minha família, cujo amor, apoio e sacrifício foram essenciais em cada passo deste caminho. Vocês são a minha inspiração e força.

Agradecimentos

Agradeço a Deus, por me dar força e sabedoria ao longo desta jornada acadêmica. Quero expressar minha profunda gratidão à Profa. Dra. Rubia Mara de Oliveira Santos, cuja orientação e apoio foram fundamentais para a realização deste trabalho.

Agradecemos à CAPES pelo incentivo e apoio financeiro e, por fim, à UFMS pela oportunidade de trilhar o caminho da pesquisa científica.

Resumo

Este trabalho investiga a integração de tecnologias educacionais, como o Arduino e a robótica, no ensino de geometria plana para alunos do 9º ano do ensino fundamental. A abstração dos conceitos geométricos frequentemente resulta em dificuldades de compreensão e falta de motivação. Para abordar esse desafio, este estudo propõe uma abordagem que utiliza plataformas de prototipagem rápida para facilitar a aplicação prática. Os objetivos específicos estão desenvolver atividades utilizando o Arduino e a robótica para ensinar conceitos de geometria, promover uma aprendizagem interdisciplinar integrando conhecimentos de matemática, robótica, e capacitar professores para utilizarem essas tecnologias em sala de aula. A metodologia inclui a criação de atividades que envolvem a montagem de robôs e a programação de tarefas geométricas.

Palavras-chave: Arduino, Robótica Educacional, Ensino de Geometria.

Abstract

This work investigates the integration of educational technologies, such as Arduino and robotics, in teaching plane geometry to 9th grade elementary school students. The abstraction of geometric concepts often results in difficulties in understanding and lack of motivation. To address this challenge, this study proposes an approach that uses rapid prototyping platforms to facilitate practical application. The specific objectives are to develop a activities using Arduino and robotics to teach geometry concepts, promote interdisciplinary learning by integrating knowledge of mathematics, robotics, and train teachers to use these technologies in the classroom. The methodology includes the creation of activities that involve assembling robots and programming geometric tasks.

Key words: Arduino, Educational Robotics, Teaching Geometry.

Sumário

Resumo	iii
Abstract	iv
Introdução	1
1 Referencial teórico	6
1.1 Conceitos básicos de geometria plana	6
1.1.1 Triângulos e algumas de suas propriedades na geometria Euclidiana	8
1.2 Polígonos	11
1.2.1 Polígonos regulares	11
2 Construção e montagem do robô	19
3 Integração da mat. e tec. no ensino	28
3.1 Atividade 1: Conhecendo o robô	29
3.2 Atividade 2: Reconhecimento da existência de triângulos	31
3.2.1 Utilizando o Arduino no reconhecimento da existência de triângulos	33
3.3 Atividade 3: cálculo de área e o perímetro	38
3.3.1 Utilizando o Arduino no cálculo de área e o perímetro	39
3.4 Atividade 4: classificação de triângulo	43
3.4.1 Utilizando o Arduino na classificação de triângulo	45
3.5 Atividade 5: cálculo de área e perímetro com o robô desenhador de polígonos	47
3.5.1 Aplicação em sala de aula	50
Conclusões e trabalhos futuros	51
Bibliografia	53

Lista de Abreviaturas e Siglas

LDB – Lei de Diretrizes e Bases da Educação nº 9.394/1996

TIC – Tecnologias da Informação e Comunicação

BNCC – Base Nacional Comum Curricular

AB – Seguimento de reta que parte do ponto A ao ponto B

$\widehat{BAC}$ – Ângulo formado pelos seguimentos de reta BA e AC

$m(\widehat{BAC})$ – Medida do ângulo $\widehat{BAC}$

LLL – Caso de congruência, como Lado-Lado-Lado

LAL – Caso de congruência, como Lado-Ângulo-Lado

ALA – Caso de congruência, como Ângulo-Lado- Ângulo

$\triangle ABC$ – Triângulo de vértices A , B e C

a_i – Ângulo interno de um polígono

a_e – Ângulo externo de um polígono

S_i – Soma dos ângulos internos de um polígono

S_e – Soma dos ângulos externos de um polígono

$A_{\triangle}$ – Área do triângulo equilátero

A_{p_n} – Área de um polígono de n lados

$\triangle ABC \cong \triangle DEF$ – Congruência dos triângulos $\triangle ABC$ e $\triangle DEF$

Lista de Figuras

1.1	Segmento de reta AB	7
1.2	Ângulo formado entre duas semirretas	7
1.3	Linha poligonal aberta e linha poligonal fechada	8
1.4	Caso LLL	9
1.5	Caso LAL	10
1.6	Caso ALA	10
1.7	Soma dos ângulos internos de um triângulo	12
1.8	Triângulo equilátero	13
1.9	Altura do triângulo equilátero	15
1.10	Triângulo isósceles no polígono regular de n lados.	17
2.1	Montagem do chassi	22
2.2	Arduino, tela LCD, Protoboard, mini drive motor	22
2.3	Teclado	23
2.4	Conexão dos cabos	24
2.5	Testes e ajustes do pincel	26
3.1	Projeção dos lados b e a no terceiro lado c	32
3.2	Seguimentos que não formam triângulo	32
3.3	Verificação da condição de existência do triângulo no código.	36
3.4	Função criada para leitura do lado a	37
3.5	Função criada para verificação da condição de existência de um triângulo.	37
3.6	Exemplo de código que pode ser explicado sobre cálculo de perímetro de polígonos regulares.	41
3.7	Exemplo de código que pode ser explicado sobre cálculo de área de polígonos regulares.	41

3.8	Exemplo de código para a explicação	46
3.9	Função <i>drawPolygon</i>	49
3.10	Menu principal do robô desenhador.	50
3.11	Robô desenhando polígonos.	51

Introdução

A geometria, uma das áreas fundamentais da matemática, enfrenta desafios significativos no contexto educacional atual, especialmente no ensino fundamental. A abstração dos conceitos geométricos frequentemente resulta em dificuldades de compreensão e falta de motivação por parte dos alunos. Neste cenário, a integração de tecnologias educacionais, como o Arduino e a robótica, apresenta uma oportunidade para transformar a didática tradicional, tornando o aprendizado mais interativo e envolvente.

O uso de plataformas de prototipagem como o Arduino no ensino de geometria pode melhorar a forma como os conceitos matemáticos são ensinados e aprendidos. Esta tecnologia facilita a aplicação prática de teorias abstratas, permitindo aos alunos visualizar e manipular formas geométricas de maneira concreta. Além disso, a robótica educativa promove o desenvolvimento de habilidades essenciais do século XXI, como a programação, o pensamento crítico e a resolução de problemas.

Conforme argumentado por Berlinghoff e Gouvêa [3], a introdução de ferramentas tecnológicas no ensino pode melhorar a compreensão dos alunos ao proporcionar experiências práticas e contextuais. A crescente importância da tecnologia no mundo moderno demanda que os alunos desenvolvam habilidades técnicas e digitais desde cedo. Integrar o Arduino e a robótica no currículo de matemática não apenas torna o aprendizado mais relevante e interessante, mas também prepara os alunos para futuros desafios acadêmicos e profissionais. Esta abordagem interdisciplinar enriquece o processo educativo e responde às diretrizes curriculares que incentivam o uso de tecnologias na educação. O problema central deste trabalho é como tornar o ensino de geometria mais acessível e compreensível para os alunos do 9º ano, utilizando tecnologias educacionais para facilitar a aprendizagem.

A metodologia deste trabalho envolve a criação de um conjunto de atividades práticas utilizando o Arduino e robôs educativos. As atividades podem ser aplicadas em aulas de

geometria e incluirão a montagem de robôs e programação de tarefas geométricas.

Arduino e Matemática

O uso da robótica na educação tem se mostrado uma ferramenta pedagógica poderosa para elevar o potencial dos alunos e promover a aprendizagem de conceitos científicos, especialmente na área da matemática. Ao integrar a robótica nas práticas pedagógicas, os estudantes têm a oportunidade de aplicar os conhecimentos matemáticos de forma prática e concreta, facilitando a compreensão e a internalização desses conceitos [1].

A robótica não se limita apenas à construção de robôs, mas também envolve o aprendizado de conceitos por meio da montagem e manipulação desses sistemas robóticos. Isso destaca a importância não apenas do produto, mas também do processo de construção e experimentação, que estimula o pensamento crítico, a resolução de problemas e o trabalho em equipe. Portanto, o uso da robótica educacional vai além de simplesmente ensinar robótica; ela proporciona uma abordagem interdisciplinar que integra matemática, ciência, tecnologia, enriquecendo o processo de ensino-aprendizagem e preparando os alunos para os desafios do mundo contemporâneo.

A versatilidade da robótica permite sua aplicação em uma ampla gama de disciplinas. Na matemática, os alunos podem explorar geometria plana ao projetar e programar robôs que realizam movimentos precisos e cálculos de ângulos. Em ciências, podem investigar fenômenos físicos através da criação de experimentos controlados por robôs. Na área de tecnologia e informática, a programação de algoritmos para automatizar tarefas reforça a compreensão de linguagens de programação e lógica computacional.

O Arduino, uma plataforma de prototipagem eletrônica, desempenha um papel importante nesse cenário. Ele permite que os alunos construam e programem dispositivos eletrônicos que podem realizar uma variedade de tarefas. No ensino de matemática, o Arduino pode ser utilizado para criar projetos que envolvem a medição de grandezas físicas, como tempo, distância e temperatura, integrando esses dados em cálculos matemáticos. Por exemplo, ao utilizar sensores de distância e motores, os alunos podem programar robôs que seguem trajetórias pré-definidas, aplicando conceitos de geometria e trigonometria [10].

A robótica, portanto, tem o potencial de desmistificar a matemática, tornando-a mais

acessível e menos intimidante. Ao instigar os alunos em atividades práticas que requerem pensamento crítico e resolução de problemas, ela pode ajudar a superar os bloqueios históricos e incentivar um interesse genuíno pela disciplina. Esta abordagem também prepara os alunos para um mundo cada vez mais digital e tecnológico, desenvolvendo habilidades que são essenciais para o século XXI [7].

Em suma, a robótica não apenas enriquece o ensino da matemática, mas também contribui para a formação de alunos mais confiantes, criativos e preparados para os desafios futuros. Ao integrar a tecnologia no ensino de maneira significativa, ela promove uma aprendizagem mais profunda e duradoura.

Diretrizes curriculares, competências e tecnologias na educação matemática

A Lei de Diretrizes e Bases da Educação Nacional (LDB), Lei nº 9.394/1996, é um marco fundamental na educação brasileira, definindo diretrizes que orientam a educação em todos os níveis e modalidades. A LDB diz que a educação básica deve garantir a formação integral do cidadão, destacando a importância das áreas de conhecimento, como Matemática e Tecnologia.

No que se refere ao ensino de Matemática, a LDB estabelece que o currículo da educação básica deve incluir conteúdos que promovam o desenvolvimento de competências e habilidades essenciais para a formação do aluno. A Matemática é considerada uma disciplina importante para a formação do pensamento lógico, crítico e para a resolução de problemas. Segundo o artigo 26 da LDB, os currículos da educação básica devem ter uma base nacional comum, complementada por uma parte diversificada, que deve ser elaborada em função das características regionais e locais da sociedade, da cultura, da economia e da clientela [5].

A LDB fala sobre a importância da interdisciplinaridade no ensino da Matemática, integrando-a com outras áreas do conhecimento e aplicando conceitos matemáticos em contextos práticos e reais. Esta abordagem visa tornar a aprendizagem mais significativa e relevante para os alunos, permitindo que eles compreendam a aplicação prática dos conceitos matemáticos em seu cotidiano e em diversas áreas profissionais.

A Tecnologia, por sua vez, é reconhecida pela LDB como uma área de crescente impor-

tância na educação. A lei incentiva o uso das Tecnologias da Informação e Comunicação (TIC) como ferramentas pedagógicas que enriquecem o processo de ensino-aprendizagem. O artigo 32, parágrafo 4º, destaca a importância de preparar os alunos para o uso competente das tecnologias modernas, integrando-as ao currículo escolar de forma transversal e interdisciplinar [5].

No contexto do Ensino Fundamental, a Base Nacional Comum Curricular (BNCC) desempenha um papel importante ao estabelecer diretrizes específicas para o desenvolvimento de competências e habilidades em todas as áreas do conhecimento, incluindo Matemática e Tecnologia. A BNCC, aprovada pelo Conselho Nacional de Educação e homologada pelo Ministério da Educação, é um documento normativo que define o conjunto de aprendizagens essenciais que todos os alunos devem desenvolver ao longo da Educação Básica [4].

No ensino de Matemática, a BNCC enfatiza a necessidade de desenvolver o pensamento crítico, a capacidade de resolver problemas e o raciocínio lógico-matemático. Ela propõe que os alunos sejam capazes de utilizar a Matemática para interpretar e atuar no mundo, promovendo a aprendizagem de conteúdos de forma contextualizada e significativa. Ela também sugere o uso de tecnologias digitais como ferramentas para potencializar a aprendizagem matemática, integrando recursos tecnológicos no processo educacional.

Para implementar os princípios da LDB e as diretrizes da BNCC no ensino de Matemática e Tecnologia, algumas práticas pedagógicas recomendadas incluem o uso de metodologias ativas, como a resolução de problemas, projetos colaborativos e estudos de caso. Essas metodologias envolvem o uso de ferramentas tecnológicas e matemáticas de forma integrada, promovendo a aprendizagem ativa e significativa. O uso de softwares educacionais, aplicativos e plataformas digitais é uma estratégia eficaz para auxiliar na compreensão de conceitos e no desenvolvimento de habilidades práticas.

O uso de robôs com Arduino para desenhar polígonos e calcular perímetros e áreas exemplifica a aplicação prática dos conceitos matemáticos em um contexto tecnológico e interdisciplinar. As atividades sugeridas permitem que os alunos explorem a geometria de forma ativa e prática, utilizando tecnologias modernas para resolver problemas reais e desenvolver habilidades essenciais. Dessa forma, o projeto não apenas segue as diretrizes da LDB e da BNCC, mas também promove uma abordagem criativa e envolvente para o ensino de Matemática e Tecnologia, preparando os alunos para enfrentar os desafios do

século XXI com competência e criatividade.

Objetivos e organização do trabalho

O objetivo geral é explorar a aplicação de Arduino e robótica no ensino de geometria para alunos do 9º ano do ensino fundamental, proporcionando uma abordagem prática e interativa que facilite a compreensão dos conceitos geométricos. Entre os objetivos específicos estão desenvolver atividades utilizando o Arduino e a robótica para ensinar conceitos de geometria, promover uma aprendizagem interdisciplinar integrando conhecimentos de matemática, robótica e engenharia, e capacitar professores para utilizarem essas tecnologias em sala de aula.

A estrutura da dissertação é composta pelos seguintes capítulos: Onde serão apresentados a relevância, justificativa, problema, objetivos e metodologia do trabalho; 2. Arduino e Matemática, que descreve a abordagem metodológica adotada para o desenvolvimento das atividades, explora o alinhamento das atividades com as diretrizes curriculares e as competências desenvolvidas, detalha o processo de construção do robô e sua integração com o Arduino, e discute a aplicação dos conceitos geométricos através do robô; 3. Construção do robô, apresentando os passos da montagem; 4. Integração da matemática e tecnologia no ensino fundamental e 5. Considerações Finais, fornecendo as conclusões do trabalho e sugestões para futuras pesquisas e aplicações.

A integração de Arduino e robótica no ensino de geometria representa uma abordagem prática e eficaz para melhorar a compreensão dos conceitos geométricos entre os alunos do 9º ano do ensino fundamental. Como enfatizado por Stewart [12], a utilização de métodos práticos e tecnológicos no ensino pode fomentar uma aprendizagem mais profunda e significativa. Este trabalho visa não apenas facilitar o aprendizado de matemática, mas também promover habilidades essenciais para o século XXI, preparando os alunos para um futuro cada vez mais tecnológico.

Capítulo 1

Referencial teórico

Na robótica, especialmente com robôs que desenham polígonos e calculam áreas e perímetros, a geometria plana é essencial para a modelagem dos movimentos do robô, simulação de trajetórias e programação de comportamentos. Através dessas atividades, os alunos aplicam conceitos geométricos de forma prática, reforçando seu entendimento e habilidade em resolver problemas matemáticos. Este capítulo consiste na apresentação dos conceitos utilizados para as atividades propostas neste trabalho. Todos os postulados, definições e embasamento teórico deste subtópico se encontra referenciado em: [3], [6], [7], [8], [9], [11], [12], [13], [14] e [15].

1.1 Conceitos básicos de geometria plana

Quando foi estabelecida pelo matemático grego por volta de 300 a.C. em sua obra “Os Elementos”, Euclides formulou um sistema lógico de axiomas e teoremas que formam a estrutura da geometria plana.

Os axiomas de Euclides, também conhecidos como postulados, são proposições que Euclides assumiu como verdadeiras sem prova. Eles servem como fundamentos para deduzir outros teoremas e propriedades geométricas.

Postulado 1.1.1 *Por dois pontos distintos, passam uma única reta.*

Este axioma estabelece a ideia de que dois pontos determinam uma linha reta única, formando a base para o conceito de linha reta na geometria.

Postulado 1.1.2 *Dados dois pontos distintos A e B , sempre existem: um ponto C entre A e B e um ponto D , tal que B está entre A e D .*

CAPÍTULO 1. REFERENCIAL TEÓRICO

Esse axioma define a natureza infinita das linhas retas, permitindo que elas se prolonguem sem limites.

Os axiomas de Euclides são fundamentais porque estabelecem um sistema lógico a partir do qual é possível derivar todas as propriedades e teoremas da geometria plana. Com base nesses axiomas, pode-se provar teoremas sobre a soma dos ângulos internos de um triângulo, a relação entre os ângulos de um polígono, e muitas outras propriedades geométricas essenciais como, por exemplo, o conceito de segmento de reta.

Definição 1.1.1 *O conjunto formado pelos pontos A e B e por todos os pontos que se encontram entre eles é o segmento de reta que parte do ponto A ao ponto B , denotado por AB e representado pela Figura 1.1.*

Figura 1.1: Segmento de reta AB

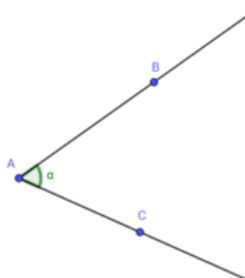


Fonte: arquivo pessoal

Definição 1.1.2 *Se A e B são pontos distintos, o conjunto formado pelos pontos pertencentes a AB e por todos os pontos C , tais que B está entre A e C , constitui a semirreta de origem A passando pelo ponto B , denotada por S_{AB} .*

Definição 1.1.3 *Ângulo é a figura formada por duas semirretas de mesma origem.*

Figura 1.2: Ângulo formado entre duas semirretas



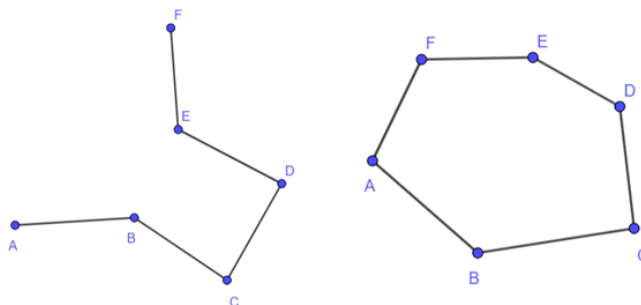
Fonte: arquivo pessoal

CAPÍTULO 1. REFERENCIAL TEÓRICO

A Figura 1.2 mostra a formação de um ângulo α a partir de duas semirretas com origem no ponto A. O ângulo α também pode ser escrito na forma $\widehat{BAC}$ ou $\widehat{CAB}$ e sua medida como $m(\widehat{BAC})$ ou $m(\widehat{CAB})$.

Definição 1.1.4 *Linha poligonal é uma figura geométrica composta por uma sequência de segmentos de linha conectados, onde cada segmento se encontra exatamente com dois outros segmentos. Esses segmentos não estão todos em linha reta, mas conectam-se em uma série de vértices, formando uma figura fechada ou aberta.*

Figura 1.3: Linha poligonal aberta e linha poligonal fechada



Fonte: arquivo pessoal

Para Dante (2005), as linhas poligonais, representadas na Figura 1.3, são importantes no estudo das propriedades geométricas porque elas ajudam a entender a formação e a estrutura dos polígonos. Por exemplo, um triângulo é uma linha poligonal fechada com três segmentos de linha, enquanto um quadrado possui quatro segmentos.

A conexão entre a linha poligonal e os polígonos é direta e fundamental. Quando uma linha poligonal é fechada, ela forma um polígono. Polígonos são figuras planas delimitadas por linhas poligonais fechadas.

1.1.1 Triângulos e algumas de suas propriedades na geometria Euclidiana

Definição 1.1.5 *Um triângulo é um polígono, ou seja, uma linha poligonal fechada que possui três lados e, conseqüentemente, três ângulos. O próprio nome “triângulo” indica essa característica: “tri” (três) e “ângulo”.*

Definição 1.1.6 (Congruência de triângulos) *Congruência de triângulos é a relação de*

CAPÍTULO 1. REFERENCIAL TEÓRICO

igualdade entre dois triângulos em termos de forma e tamanho, onde todos os lados correspondentes e ângulos são congruentes.

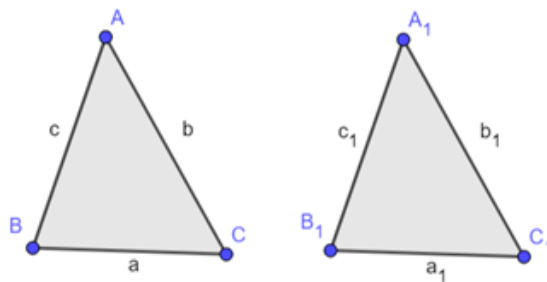
Esta definição estabelece a base para a aplicação dos critérios clássicos de congruência, como Lado-Lado-Lado (*LLL*), Lado-Ângulo-Lado (*LAL*) e Ângulo-Lado-Ângulo (*ALA*). Como destacado por Coxeter [6], “estes critérios não apenas identificam triângulos congruentes, mas também são essenciais para a prova de teoremas geométricos fundamentais”.

Além dos contextos aplicados, a congruência de triângulos desempenha um papel central na resolução de problemas geométricos complexos. Como observado por Euclides em seus “Elementos” [9], a congruência de triângulos permite desdobrar questões complexas em passos lógicos, utilizando a equivalência geométrica como fundamento.

Os critérios de congruência de triângulos são fundamentais na geometria euclidiana para determinar quando dois triângulos são idênticos em forma e tamanho. Cada critério utiliza diferentes combinações de lados e ângulos dos triângulos para estabelecer essa congruência.

Critério Lado-Lado-Lado (*LLL*): Este critério afirma que dois triângulos são congruentes se os três lados de um triângulo são respectivamente iguais aos três lados correspondentes do outro triângulo. Na construção civil, este critério é usado para garantir que estruturas triangulares tenham dimensões exatas em cada lado, assegurando a estabilidade e equilíbrio da estrutura.

Figura 1.4: Caso *LLL*



Fonte: arquivo pessoal

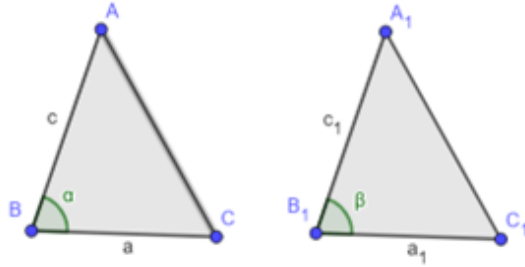
Tomando $a = a_1$, $b = b_1$ e $c = c_1$ pode-se concluir que os triângulos $\triangle ABC$ e $\triangle A_1B_1C_1$ da Figura 1.4 são congruentes pelo caso *LLL*.

Critério Lado-Ângulo-Lado (*LAL*): No critério *LAL* (figura 1.5), dois triângulos são congruentes se dois lados e o ângulo formado por esses lados em um triângulo são iguais

CAPÍTULO 1. REFERENCIAL TEÓRICO

aos dois lados e ao ângulo correspondente no outro triângulo. Este critério é aplicável em situações em que a orientação ou a direção dos triângulos é relevante, como na navegação e em sistemas de posicionamento global (GPS).

Figura 1.5: Caso LAL

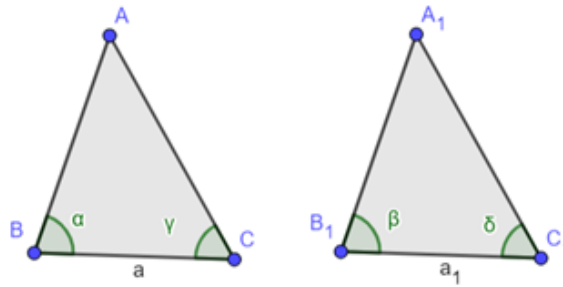


Fonte: arquivo pessoal

Tomando $a = a_1$ e $c = c_1$ e $\alpha = \beta$ pode-se concluir que os triângulos $\triangle ABC$ e $\triangle A_1B_1C_1$ da Figura 1.5 são congruentes pelo caso LAL .

Critério Ângulo-Lado-Ângulo (ALA): O critério ALA (figura 1.6) estabelece que dois triângulos são congruentes se dois ângulos e o lado entre eles em um triângulo são iguais aos dois ângulos e ao lado correspondente no outro triângulo. Este critério é utilizado na engenharia mecânica para projetar componentes que se encaixam perfeitamente, garantindo compatibilidade e eficiência na montagem.

Figura 1.6: Caso ALA



Fonte: arquivo pessoal

Tomando $a = a_1$, $\alpha = \beta$ e $\gamma = \delta$ pode-se concluir que os triângulos $\triangle ABC$ e $\triangle A_1B_1C_1$ da Figura 1.6 são congruentes pelo caso ALA .

1.2 Polígonos

A seguinte definição estabelece mais formalmente o conceito de polígono.

Definição 1.2.1 *Um polígono é uma linha poligonal fechada em que as seguintes condições são satisfeitas:*

- (a) *Os lados da poligonal se interceptam somente em suas extremidades;*
- (b) *Cada vértice é extremidade de dois lados;*
- (c) *Dois lados com uma mesma extremidade não pertencem a uma mesma reta.*

1.2.1 Polígonos regulares

Os polígonos regulares são figuras geométricas fundamentais caracterizadas por lados e ângulos internos de mesma medida. Exemplos incluem o triângulo equilátero, o quadrado e o hexágono regular. A simetria dos polígonos regulares facilita a introdução dos alunos a conceitos de simetria, ângulos e proporções, além de serem úteis na exploração de tesselações e padrões repetitivos.

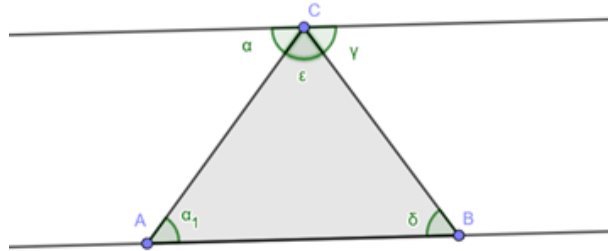
Além das propriedades matemáticas, os polígonos regulares têm significativas aplicações práticas, desde o design e engenharia até a robótica e computação gráfica. Dante [8] sugere que o ensino dos polígonos regulares deve incluir atividades práticas e visuais, como o uso de tecnologias como o Arduino para programar robôs que desenharam polígonos, tornando o aprendizado mais interativo e envolvente.

Os polígonos regulares são essenciais para o desenvolvimento do pensamento matemático, com aplicações práticas amplas. A tecnologia e atividades práticas no ensino desses polígonos pode enriquecer a experiência de aprendizagem, proporcionando uma compreensão mais profunda e duradoura dos princípios geométricos.

Teorema 1.2.1 *A soma dos ângulos internos de um triângulo, na geometria plana, é sempre igual a 180 graus.*

Demonstração *Considere um triângulo $\triangle ABC$ qualquer de ângulos internos iguais a α , β e γ . Tome uma reta que passa pelo vértice C do triângulo e que é paralela a uma reta que contém a base AB do triângulo, como é mostrado na Figura 1.7*

Figura 1.7: Soma dos ângulos internos de um triângulo



Fonte: arquivo pessoal

Os ângulos α e α_1 são congruentes (alternos internos) assim como os ângulos δ e γ (alternos internos). Sabe-se que o ângulo $\alpha + \varepsilon + \gamma$ é raso, ou seja, a soma $\alpha + \varepsilon + \gamma$ deve ser igual a 180 graus. Logo, a soma dos ângulos internos do triângulo $\triangle ABC$ é igual a 180 graus.

Proposição 1.2.1 Os ângulos internos de um triângulo equilátero são iguais a 60 graus e todos os ângulos externos iguais a 120 graus.

Demonstração: Sabendo que o triângulo equilátero é um polígono que possui todos os lados de mesma medida, que todos os ângulos são congruentes e que a soma dos ângulos internos de um triângulo é 180 graus, tem-se que

$$a_i + a_i + a_i = 3a_i = 180^\circ$$

$$a_i = \frac{180^\circ}{3} = 60^\circ$$

sendo a_i o ângulo interno. Tome a_e como o ângulo externo. Sabe-se que

$$a_i + a_e = 180^\circ$$

ou seja,

$$a_e = 180^\circ - 60^\circ = 120^\circ.$$

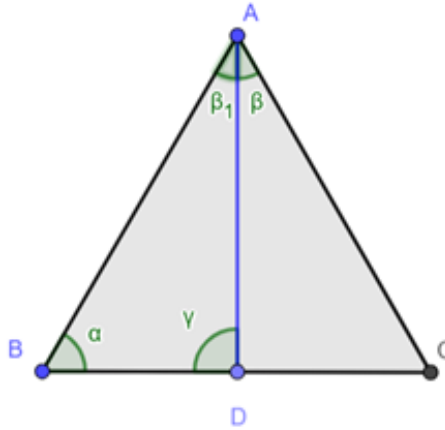
Logo, os ângulos internos de um triângulo equilátero medem 60 graus e os ângulos externos medem 120 graus.

■

Teorema 1.2.2 *No triângulo equilátero, a altura (ou altura relativa a um lado), a bissetriz (ou bissetriz interna do ângulo) e a mediana (ou mediana relativa a um lado) coincidem.*

Demonstração: *Considere o triângulo da Figura 1.8.*

Figura 1.8: Triângulo equilátero



Fonte: arquivo pessoal

Tome o seguimento AD , comum aos dois triângulos $\triangle ABD$ e $\triangle ACD$, como bissetriz do ângulo $\widehat{BAC}$, ou seja, $\beta_1 = \beta$. Os seguimentos AB e AC são congruentes. Logo, os triângulos $\triangle ABD$ e $\triangle ACD$ são congruentes pelo caso de congruência LAL , pois $AB = AC$, $\beta_1 = \beta$ e AD está nos dois triângulos.

A partir daí, pode-se concluir que $BD = CD$, logo AD é mediana e mediatriz.

Como os ângulos $\widehat{ADB}$ e $\widehat{ADC}$ são congruentes devido a congruência dos triângulos e que $\widehat{ADB} + \widehat{ADC} = 180^\circ$ chega-se em $\gamma = 90^\circ$ e, portanto, o seguimento AD também é altura.

De modo análogo se demonstra que a bissetriz relativa à base de medida diferente aos lados congruentes é altura, mediana e mediatriz de um triângulo isósceles.

A soma dos ângulos internos e externos de um polígono é um conceito central para este projeto, pois a rotação do robô utiliza os ângulos externos dos polígonos regulares para desenhar suas formas corretamente.

Proposição 1.2.2 *A medida do ângulo externo a_e de um polígono regular de n lados, $n \geq 3$, é igual a*

$$a_e = \frac{360^\circ}{n}.$$

CAPÍTULO 1. REFERENCIAL TEÓRICO

Demonstração: A medida do ângulo interno de um polígono regular é dada pela fórmula

$$a_i = \frac{(n-2) \cdot 180^\circ}{n}$$

e que

$$a_i + a_e = 180^\circ.$$

O que implica em

$$\begin{aligned} a_e &= 180^\circ - a_i = 180^\circ - \frac{(n-2) \cdot 180^\circ}{n} \\ &= \frac{(180^\circ \cdot n - n \cdot 180^\circ + 2 \cdot 180^\circ)}{n} = \frac{360^\circ}{n} \end{aligned}$$

Logo,

$$a_e = \frac{360^\circ}{n}.$$

■

Definição 1.2.2 *Perímetro é definido como a soma dos comprimentos dos lados de uma figura geométrica.*

O perímetro de um triângulo é a soma das medidas de seus três lados, enquanto o perímetro de um quadrado é quatro vezes a medida de um de seus lados. Para polígonos regulares, onde todos os lados têm a mesma medida, o cálculo do perímetro é simplificado pela multiplicação do comprimento de um lado pelo número total de lados.

Teorema 1.2.3 *Para um polígono regular com n lados de comprimento l , o perímetro P é dado pela fórmula*

$$P = n \cdot l.$$

Demonstração: Como o polígono é regular e o perímetro é soma das medidas dos lados desse polígono, tem-se que

$$\underbrace{l + l + l \cdots + l}_{n \text{ vezes}} = n \cdot l$$

Logo, o perímetro de um polígono regular é multiplicação da quantidade de lados n pelo comprimento l do lado.

■

Definição 1.2.3 *Área é a medida da superfície interna de uma figura geométrica.*

CAPÍTULO 1. REFERENCIAL TEÓRICO

A área quantifica o espaço bidimensional ocupado pela figura. Ela é expressa em unidades quadradas, como centímetros quadrados (cm^2) ou metros quadrados (m^2). As fórmulas para calcular a área variam de acordo com a forma da figura. Por exemplo, a área de um retângulo é obtida multiplicando-se o comprimento pela largura, enquanto a área de um triângulo é metade do produto da base pela altura.

A área de um polígono regular é uma medida que representa a extensão da superfície ocupada pela figura. O cálculo da área desses polígonos envolve tanto conhecimentos básicos de geometria quanto aplicações práticas em diversos campos. Conhecendo o Teorema 1.2.2 é possível demonstrar a fórmula do cálculo da área de um triângulo equilátero conhecendo apenas a medida do seu lado.

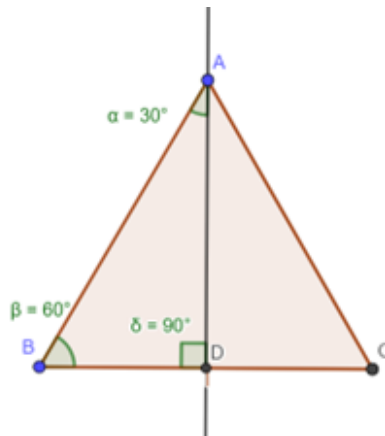
Teorema 1.2.4 *A área de um triângulo equilátero é dada pela fórmula*

$$A_{\Delta} = \frac{l^2 \cdot \sqrt{3}}{4}$$

onde l é a medida do lado do triângulo.

Demonstração: *Pelo teorema 1.2.2 sabe-se que a altura de um triângulo equilátero coincide com a bissetriz e a mediana. Considere o triângulo da Figura 1.9*

Figura 1.9: Altura do triângulo equilátero



Fonte: arquivo pessoal

Sendo a medida $h = AD$ a altura do triângulo retângulo ABD, então

$$\text{sen}(60^\circ) = \frac{h}{l} \implies h = \frac{l \cdot \sqrt{3}}{2}.$$

Sabendo que a área de um triângulo qualquer é dada pela fórmula

$$A_{\Delta} = \frac{l \cdot h}{2}$$

CAPÍTULO 1. REFERENCIAL TEÓRICO

e substituindo $h = \frac{l\sqrt{3}}{2}$ obtém-se

$$A_{\Delta} = \frac{l}{2} \frac{l \cdot \sqrt{3}}{2}$$

Logo, a área de um triângulo equilátero pode ser dada apenas em função do lado l pela fórmula

$$A_{\Delta} = \frac{l^2 \cdot \sqrt{3}}{4}$$

■

Outro resultado importante é a fórmula de Heron [8], que permite calcular a área de um triângulo a partir das medidas dos lados:

$$A = \sqrt{p \cdot (p - a) \cdot (p - b) \cdot (p - c)},$$

onde p é a metade do perímetro P , ou seja, $p = P/2$.

Utilizando uma demonstração parecida com a do teorema 1.2.4 pode-se obter uma fórmula geral para o cálculo da área de um polígono regular qualquer de n lados apenas em função da quantidade de lados e do tamanho do lado l .

Teorema 1.2.5 *A área de um polígono regular de n lados é dada pela fórmula*

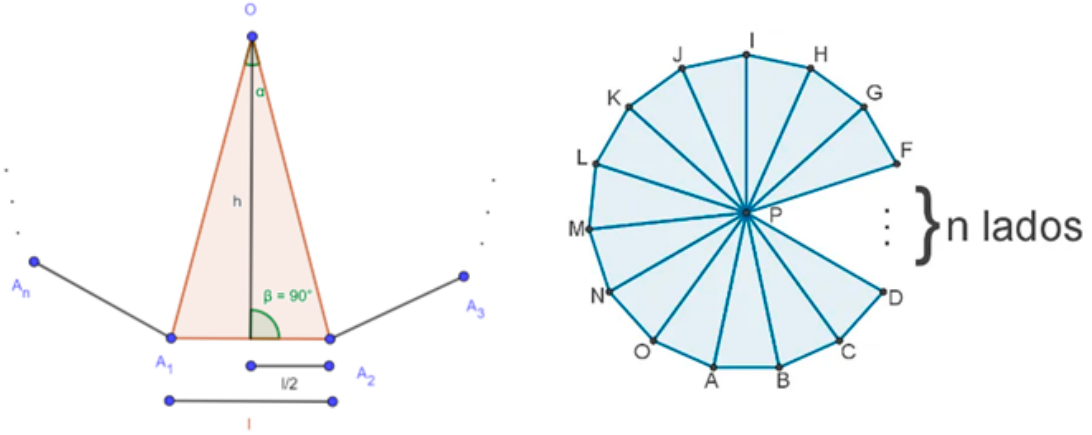
$$A_{p_n} = \frac{n \cdot l^2}{4 \cdot \operatorname{tg}\left(\frac{\pi}{n}\right)}$$

sendo l a medida do lado do polígono.

Demonstração: *Divide-se o polígono em n triângulos congruentes, neste caso, triângulos isósceles cuja altura relativa à base de medida l (lado do polígono de n lados) coincide com a bissetriz, a mediatriz e a mediana como já foi demonstrado anteriormente no teorema 1.2.2 para o caso em que o triângulo é equilátero, entretanto a prova é análoga para o triângulo isósceles, cujos ângulos da base são congruentes. Deve-se calcular a área de apenas um desses triângulos e em seguida multiplicá-la pela quantidade total de triângulos o que resultará na área total do polígono.*

CAPÍTULO 1. REFERENCIAL TEÓRICO

Figura 1.10: Triângulo isósceles no polígono regular de n lados.



Fonte: arquivo pessoal

O triângulo $\triangle A_1OA_2$ na Figura 1.10 é isósceles, pois os segmentos de reta A_1O e A_2O são congruentes uma vez que representam o raio da circunferência circunscrita ao polígono de n lados.

O ângulo central de um polígono de n lados é dado por $2\pi/n$. Como a altura do triângulo é bissetriz pode-se concluir que o ângulo central $2\pi/n = 2 \cdot \alpha$ e, portanto, $\alpha = \pi/n$.

A altura h é dada por

$$\operatorname{tg}\left(\frac{\pi}{n}\right) = \frac{l}{2h} \implies h = \frac{l}{2 \cdot \operatorname{tg}\left(\frac{\pi}{n}\right)}.$$

A área do triângulo $\triangle A_1OA_2$ é dada por

$$A_{\triangle A_1OA_2} = \frac{l \cdot h}{2} = \frac{l \cdot \frac{l}{2 \cdot \operatorname{tg}\left(\frac{\pi}{n}\right)}}{2} = \frac{l^2}{4 \cdot \operatorname{tg}\left(\frac{\pi}{n}\right)}.$$

Portanto, a área do polígono de n lados é dada por

$$A_{p_n} = n \cdot A_{\triangle A_1OA_2} = \frac{n \cdot l^2}{4 \cdot \operatorname{tg}\left(\frac{\pi}{n}\right)}.$$

■

Note que a fórmula depende apenas da quantidade de lados n e do tamanho do lado l . A área A_{p_n} será utilizada na programação do robô no cálculo de áreas de polígonos regulares.

CAPÍTULO 1. REFERENCIAL TEÓRICO

Portanto, compreender a área de polígonos regulares é essencial para aprofundar o conhecimento em geometria plana e suas aplicações práticas. O estudo das fórmulas e métodos para calcular a área de polígonos regulares não apenas reforça a compreensão teórica dos alunos, mas também lhes permite visualizar e aplicar esses conceitos em contextos do mundo real, como na arquitetura, design e engenharia.

Capítulo 2

Construção e montagem do robô

O Arduino é uma plataforma de prototipagem rápida, versátil e de baixo custo, ideal para iniciantes e profissionais em eletrônica e programação. Desenvolvido pelo Ivrea Interaction Design Institute na Itália, o Arduino democratiza a prototipagem eletrônica, permitindo o desenvolvimento de projetos inovadores com sensores, LEDs, motores e mais. O modelo mais popular, o Arduino UNO, é amplamente utilizado devido à sua simplicidade e robustez, sendo uma excelente base para aprender sobre microcontroladores e eletrônica [2].

O Arduino é hardware e software livre, facilitando modificações e adaptações para diversas necessidades, e possui uma grande comunidade de suporte. Embora o design seja aberto, o nome “Arduino” é uma marca registrada, controlando seu uso comercial. A combinação do Arduino UNO com sua IDE permite criar protótipos funcionais de maneira rápida, aplicando conceitos matemáticos na prática.

A utilização do Arduino em projetos educacionais, como robôs que desenham polígonos regulares, ajuda os alunos a consolidar conhecimentos teóricos em geometria e matemática, proporcionando uma compreensão concreta dos conceitos abstratos através da interação com tecnologias educacionais.

A montagem de um robô educativo é uma prática pedagógica moderna que integra diversos conhecimentos, promovendo aprendizado dinâmico e interativo. Para montar um robô, é essencial entender os componentes básicos: um chassi motorizado com rodas, uma placa Arduino Uno, motores, um protoboard, sensores, atuadores, um teclado 4x4 e uma tela LCD 16x2.

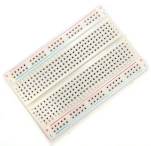
Os componentes utilizados no robô educativo desempenham papéis essenciais, desde a

CAPÍTULO 2. CONSTRUÇÃO E MONTAGEM DO ROBÔ

estrutura física até a lógica de controle. A Tabela 2.1 abaixo explica o conceito de cada componente, destacando sua função no projeto.

Tabela 2.1: Componentes para montagem do robô.

(continua)

COMPONENTE	DESCRIÇÃO
Chassi 	Tela que exibe informações como o polígono selecionado e comandos operacionais, tornando a interface do robô mais amigável e interativa.
Fonte: [16]	
Placa Arduino Uno 	Microcontrolador que atua como o "cérebro" do robô, executando a lógica de controle e recebendo comandos para coordenar sensores e motores.
Fonte: [16]	
Protoboard 	Placa de ensaio que permite a montagem de circuitos temporários sem solda, facilitando conexões rápidas entre os componentes do sistema.
Fonte: [16]	
Teclado 4x4 	Dispositivo de entrada composto por uma matriz de 4 linhas por 4 colunas (16 botões), permitindo a inserção de comandos diretamente no robô.
Fonte: [16]	
Tela LCD 16x2 	Tela que exibe informações como o polígono selecionado e comandos operacionais, tornando a interface do robô mais amigável e interativa.
Fonte: [16]	
Cabos Jumper 	Fios flexíveis que conectam os pinos da placa Arduino e do protoboard aos diferentes componentes, como sensores, motores, teclado e LCD.
Fonte: [16]	

CAPÍTULO 2. CONSTRUÇÃO E MONTAGEM DO ROBÔ

Tabela 2.1 Componentes para montagem do robô.

(conclusão)

COMPONENTE	DESCRIÇÃO
Driver de Motor (L298N)  Fonte: [16]	Controla a direção e a velocidade dos motores de corrente contínua, permitindo que o Arduino controle o movimento do robô.
Motores DC  Fonte: [16]	Motores responsáveis por movimentar o robô, controlados pela placa Arduino através do driver de motor.
Pincel  Fonte: arquivo pessoal	Componente mecânico fixado no chassi para desenhar polígonos regulares conforme os comandos do robô.

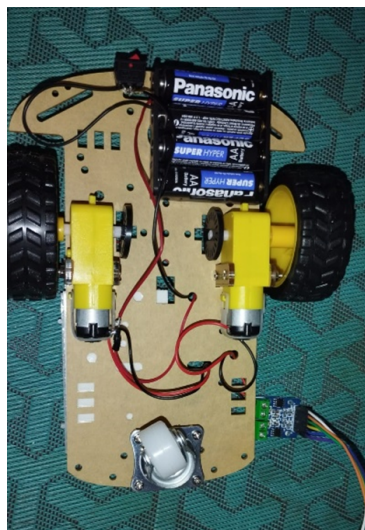
Fonte: elaboração própria.

O processo de montagem do robô educativo envolve diversas etapas, cada uma fundamental para garantir a funcionalidade e precisão do sistema. a seguir temos os detalhes dessas etapas, incluindo a descrição de cada uma, as figuras correspondentes e as fontes das informações.

Iniciando, temos o chassi, a estrutura base do robô, suporta todos os componentes e garante precisão e estabilidade na movimentação quando os motores e rodas são fixados corretamente. Ele está representado na Figura 2.1.

CAPÍTULO 2. CONSTRUÇÃO E MONTAGEM DO ROBÔ

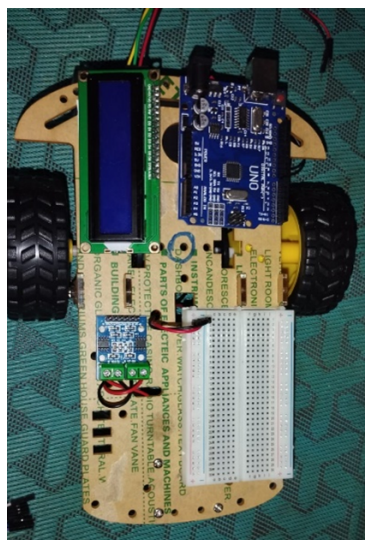
Figura 2.1: Montagem do chassi



Fonte: arquivo pessoal

Depois de construir o chassi, a placa Arduino Uno é conectada a ele, conforme a Figura 2.2. A placa Arduino atua como o cérebro do robô, implementando a lógica de controle e programação. Conectar a placa aos motores e ao protoboard permite ligar os sensores e atuadores. A programação do Arduino, usando uma linguagem de código, define o comportamento do robô em resposta aos comandos recebidos.

Figura 2.2: Arduino, tela LCD, Protoboard, mini drive motor



Fonte: arquivo pessoal

O protoboard facilita a adição de componentes como o teclado 4x4 e a tela LCD 16x2. O teclado permite que os alunos entrem com comandos diretamente no robô, selecionando

CAPÍTULO 2. CONSTRUÇÃO E MONTAGEM DO ROBÔ

diferentes polígonos para serem desenhados. A tela LCD exibe informações importantes, como o tipo de polígono selecionado e instruções operacionais, proporcionando uma interface amigável para os usuários.

A montagem do robô envolve a integração de todos os componentes em um sistema funcional, programado para executar tarefas como desenhar polígonos regulares. O teclado 4x4 (Figura 2.3) é prático nesse processo, atuando como dispositivo de entrada. Com 16 botões dispostos em uma matriz 4x4, ele permite a interação direta dos alunos com o robô. Cada botão é identificado por sua posição na matriz, e a conexão ao Arduino utiliza 8 pinos digitais. A leitura dos botões é realizada através de varredura, ativando linhas e lendo colunas para detectar quais botões estão pressionados.

Figura 2.3: Teclado



Fonte: arquivo pessoal

No contexto do robô educativo, o teclado 4x4 é utilizado para permitir que os alunos selecionem diferentes tipos de polígonos para serem desenhados pelo robô. A tela LCD 16x2, conectada ao sistema, exibe as opções de polígonos regulares, como triângulos, quadrados, pentágonos, hexágonos, heptágonos e octógonos.

Os alunos podem então usar o teclado para inserir o número correspondente ao polígono desejado. Por exemplo, ao pressionar o botão “3” no teclado, o robô é programado para desenhar um triângulo, configurando a lógica de controle para rotacionar 120° após cada movimento em linha reta. Da mesma forma, pressionar o botão “4” faz com que o robô desenhe um quadrado, rotacionando 90° após cada movimento. Este processo não só torna a interação com o robô mais dinâmica, mas também reforça a compreensão dos alunos sobre os ângulos internos e externos dos polígonos.

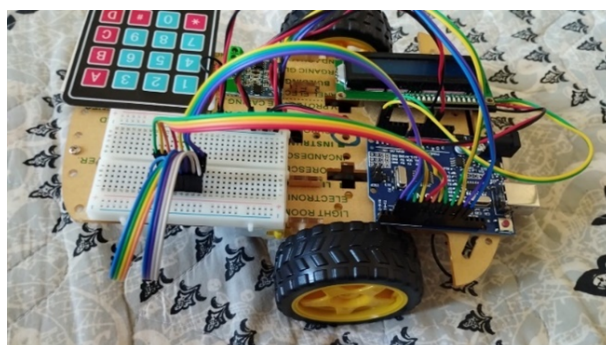
A inclusão do teclado 4x4 no projeto oferece múltiplos benefícios educacionais. Pri-

CAPÍTULO 2. CONSTRUÇÃO E MONTAGEM DO ROBÔ

meiramente, o teclado torna a experiência de aprendizado mais interativa, permitindo que os alunos tomem decisões ativas sobre as tarefas que o robô executa. Em segundo lugar, a seleção e desenho de polígonos através do teclado ajuda os alunos a aplicar conhecimentos teóricos de geometria de forma prática e concreta.

Finalmente, a utilização do teclado para programar e controlar o robô promove o desenvolvimento de habilidades em eletrônica e programação, áreas fundamentais para a compreensão da tecnologia moderna. A utilização do teclado 4x4 exemplifica este princípio, proporcionando aos alunos uma ferramenta prática para explorar e solidificar seu entendimento da geometria plana.

Figura 2.4: Conexão dos cabos



Fonte: arquivo pessoal

No que se refere aos cabos no projeto de montagem do robô educativo, a conexão adequada dos cabos, como na Figura 2.4 entre o protoboard e os componentes é essencial para garantir o funcionamento correto do sistema. O protoboard, uma placa de ensaio, permite a construção e teste de circuitos temporários sem a necessidade de soldagem, facilitando a montagem e modificação do robô.

O primeiro passo na montagem é conectar a placa Arduino ao protoboard. A placa Arduino serve como o cérebro do robô, controlando os movimentos dos motores e a interação com os sensores e atuadores. A conexão entre a placa Arduino e o protoboard é realizada utilizando cabos jumper, que são fios flexíveis com terminais que se encaixam nos pinos do Arduino e nas colunas do protoboard.

Para conectar o teclado 4x4 ao protoboard e à placa Arduino, são necessários 8 pinos digitais: 4 para as linhas e 4 para as colunas. Os cabos jumper são utilizados para ligar os pinos de saída do teclado às linhas e colunas correspondentes no protoboard. A partir do protoboard, outros cabos jumper conectam essas linhas e colunas aos pinos digitais

CAPÍTULO 2. CONSTRUÇÃO E MONTAGEM DO ROBÔ

apropriados no Arduino, permitindo a leitura dos inputs do teclado pelo microcontrolador.

A tela LCD 16x2, que exibe as opções de polígonos, também é conectada ao protoboard e à placa Arduino. Para a tela LCD, geralmente são utilizados 6 a 10 pinos dependendo do modo de comunicação (4 ou 8 bits). Os pinos de alimentação (VCC e GND) do LCD são conectados às trilhas de alimentação do protoboard, que são alimentadas pelo Arduino. Os pinos de controle (RS, E, e os pinos de dados D4-D7) são conectados aos pinos digitais do Arduino através do protoboard, utilizando cabos jumper. Além disso, um potenciômetro pode ser conectado ao LCD para ajustar o contraste da tela, sendo seus terminais conectados ao protoboard e alimentados pelo Arduino.

Os motores, responsáveis pelos movimentos do robô, também são conectados ao protoboard e controlados pelo Arduino. Motores de corrente contínua (DC) ou servomotores são comumente usados. Para motores DC, é necessário um driver de motor, como o L298N, que permite controlar a direção e a velocidade do motor. Os pinos de entrada do driver de motor são conectados aos pinos digitais do Arduino através do protoboard, enquanto os pinos de saída do driver são conectados aos terminais dos motores. A alimentação dos motores é fornecida por uma fonte de energia externa, conectada ao driver de motor e ao protoboard para distribuição.

Além dos componentes principais, outros elementos como resistores, capacitores e LEDs podem ser adicionados ao circuito no protoboard para sinalização e proteção. Os resistores são frequentemente usados para limitar a corrente elétrica e proteger os componentes, sendo conectados em série com LEDs ou outros dispositivos sensíveis. Capacitores podem ser utilizados para estabilizar a tensão de alimentação e filtrar ruídos.

As conexões dos cabos no protoboard seguem uma organização lógica para facilitar a identificação e modificação dos circuitos. As linhas de alimentação positiva (VCC) e negativa (GND) são distribuídas ao longo das trilhas laterais do protoboard, fornecendo uma referência comum para todos os componentes. Os pinos de sinal dos componentes são conectados às colunas centrais do protoboard, permitindo uma configuração modular e flexível dos circuitos.

A correta conexão dos cabos e componentes no protoboard é essencial para o funcionamento do robô educativo, garantindo que os sinais elétricos sejam transmitidos corretamente entre a placa Arduino e os diversos dispositivos.

A integração de um pincel em um chassi motorizado para o projeto do robô desenhista,

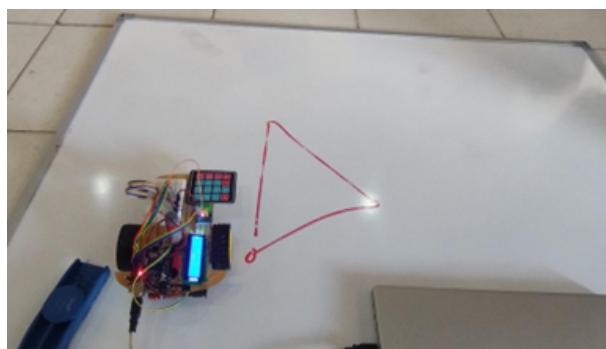
CAPÍTULO 2. CONSTRUÇÃO E MONTAGEM DO ROBÔ

como na Figura 2.5 envolveu um processo meticuloso para garantir que o pincel ficasse corretamente posicionado no eixo de rotação. Esta precisão é essencial para permitir que o robô desenhe polígonos regulares com exatidão.

Primeiramente, foi necessário adaptar o chassi motorizado, que já estava equipado com motores e rodas, para acomodar o pincel. Foi analisado que o pincel deveria ser colocados diretamente no eixo de rotação do robô, pois isso garantiria que todos os movimentos realizados pelo robô fossem transferidos de forma precisa para o papel.

Para realizar o furo no chassi, foi utilizado uma parafusadeira com uma ponta especialmente adaptada. Essa ponta era ligeiramente maior que o diâmetro do pincel, permitindo um encaixe firme, mas ao mesmo tempo ajustável. A escolha de uma broca com diâmetro um pouco maior que o do pincel foi estratégica, pois proporcionava a flexibilidade necessária para realizar pequenos ajustes e garantir que o pincel estivesse perfeitamente alinhado.

Figura 2.5: Testes e ajustes do pincel



Fonte: arquivo pessoal

O processo de perfuração começou com a marcação do ponto exato no chassi onde o furo seria feito. Esta etapa foi crítica, pois qualquer desvio na posição do furo poderia comprometer a precisão dos desenhos. Após a marcação, a parafusadeira foi usada com cuidado para fazer o buraco. A profundidade do furo foi cuidadosamente controlada para assegurar que o pincel ficasse bem encaixado, sem se mover durante a operação do robô.

Uma vez realizado o furo, o pincel foi inserido no chassi e ajustado até que ficasse centralizado no eixo de rotação. Foram realizados alguns testes preliminares para confirmar que o pincel se movia conforme esperado quando o robô era acionado. O resultado foi um robô capaz de traçar polígonos regulares com precisão. O pincel, estando perfeitamente alinhado e fixado no eixo de rotação, assegurava que cada movimento do robô fosse repro-

CAPÍTULO 2. CONSTRUÇÃO E MONTAGEM DO ROBÔ

duzido fielmente no papel, facilitando a visualização prática dos conceitos geométricos por parte dos alunos. Este ajuste minucioso do pincel no chassi motorizado destacou a importância da precisão e da atenção aos detalhes no desenvolvimento de projetos educacionais de robótica.

Capítulo 3

Integração da matemática e tecnologia no ensino

A integração da matemática e da robótica no ensino fundamental proporciona uma abordagem inovadora e prática ao aprendizado. Utilizando kits de robótica, os alunos podem aplicar conceitos matemáticos em projetos reais, como construção e programação de robôs. Essa metodologia estimula o raciocínio lógico e a resolução de problemas, além de desenvolver habilidades de colaboração e criatividade. A robótica torna a matemática mais tangível e relevante, incentivando o interesse e a curiosidade dos estudantes pela ciência e tecnologia, preparando-os para as demandas do futuro. Serão apresentadas algumas formas de utilização da robótica em sala de aula, demonstrando seu potencial pedagógico. Todos as definições, teoremas e embasamento teórico deste subtópico se encontra referenciado em: [1], [2], [7], [8], [10] e [11].

Neste capítulo, são apresentadas atividades didáticas que integram conhecimentos de geometria, robótica e programação para professores do ensino fundamental. A metodologia bibliográfica envolveu uma revisão de literatura sobre os conceitos de geometria plana, incluindo ângulos internos e externos de polígonos regulares, cálculo de perímetros e áreas, e o uso do Arduino como ferramenta educacional.

O conteúdo foi selecionado com base nas diretrizes curriculares nacionais para o ensino da geometria no 9º ano do ensino fundamental. As atividades foram projetadas para abordar conceitos geométricos fundamentais, tais como ângulos, perímetros, áreas, e coordenadas, utilizando o robô como uma ferramenta prática de ensino.

Foram consultadas diversas fontes acadêmicas, incluindo livros, artigos científicos e

materiais didáticos, para fundamentar o trabalho e garantir uma abordagem pedagógica eficaz e atualizada. A Atividade 1 consiste em uma apresentação dos componentes, enquanto as atividades seguintes ocorreram em quatro etapas, sendo: teoria, prática, computacional e uso do robô arduino.

Inicialmente, é trabalhada a teoria necessária para desenvolvimento das atividades. Nessa fase apresentamos as teorias formais dos conteúdos trabalhados, como definições da geometria plana, assim ficam familiarizado com os conceitos matemáticos que serão necessários para o desenvolvimento das atividades com o robô. Em seguida temos a prática que é a aplicação da teoria, reforçando a aprendizagem com uso de exercícios abstratos e lúdicos. Nessa fase podemos usar palitos e barbantes como recursos didáticos para melhorar e facilitar a aprendizagem .

A ferramenta Computacional feita na sala de informática pode ser feita em python ou outros programas, onde programamos a aplicação do arduino para executar as atividades. Os alunos observam recortes específicos da programação, que são de melhor compreensão, com uso das fórmulas. Além da aplicação do arduino para desenvolver a atividade, é trabalhado o uso da placa física do arduino. Portanto ocorre uma interação com o objeto para compreensão dessa teoria. Nessa etapa, a placa arduino faz tarefas onde foram pré-programadas na fase anterior: fazer cálculos da área e perímetro de polígonos, identificar se as medidas colocadas no teclado formam ou não formam um triângulo e fazer desenhos de polígonos. Assim, consolidamos a prática a partir das teorias da geometria plana com o uso da tecnologia.

3.1 Atividade 1: Conhecendo o robô

O objetivo da atividade é apresentar aos alunos os componentes e o funcionamento do robô, destacando sua aplicação no aprendizado de conceitos geométricos. A atividade utiliza como materiais um robô educativo montado, protoboard, placa Arduino Uno, teclado 4x4, tela LCD 16x2, motores DC, pincel, além de um computador com a IDE do Arduino instalada.

Inicia com uma breve introdução sobre robótica educativa e sua relevância para o aprendizado, explicando como o robô em questão auxilia na compreensão prática da geometria. Em grupo recebe componentes do robô, como o protoboard, Arduino, teclado e

CAPÍTULO 3. INTEGRAÇÃO DA MAT. E TEC. NO ENSINO

outros. A função de cada componente é explicada, utilizando o robô montado como exemplo. Os grupos são incentivados a manusear os componentes, compreendendo melhor seu funcionamento, desde a base do chassi, que suporta todos os elementos, até a placa Arduino, que controla todo o sistema, passando pelo teclado 4x4, que insere comandos, e a tela LCD, que exibe informações.

Após essa apresentação, ocorre uma demonstração prática do robô em funcionamento. Utilizando o teclado 4x4, o instrutor escolhe um polígono para o robô desenhar, explicando cada etapa do processo, desde a inserção do comando até a execução da tarefa pelo robô. Os grupo são incentivados a sugerir diferentes polígonos, observando o robô em ação.

Nessa fase os alunos são orientados pelo professor como são conectados os componentes e como é essa interação entre as partes eletrônicas, mostrando a conexão entre cada componente e os cabos. Mostrando a placa do arduino que é a parte mais importante do robô e é onde são programados as atividades e feitos os cálculos, sendo o microcontrolador que atua como o "cérebro" do robô, executando a lógica de controle e recebendo comandos para coordenar sensores e motores. Temos o chassi que é a estrutura base do robô que suporta todos os componentes e garante estabilidade e precisão durante o movimento dos motores e rodas. A placa protoboard que permite a montagem de circuitos temporários sem solda, facilitando conexões rápidas entre os componentes do sistema. O teclado que é o dispositivo de entrada composto por uma matriz de 4 linhas por 4 colunas (16 botões), permitindo a inserção de comandos diretamente no robô. O display que é a tela que exibe informações como o polígono selecionado e comandos operacionais, tornando a interface do robô mais amigável e interativa. Os cabos jumper que são fios flexíveis que conectam os pinos da placa Arduino e do protoboard aos diferentes componentes, como sensores, motores, teclado e LCD. Os motores responsáveis por movimentar o robô, controlados pela placa Arduino através do driver de motor. E o pincel que fixado no chassi para desenhar polígonos regulares conforme os comandos do robô. É mostrado também como o robô faz a interação com a programação, observando as ordens pré-programadas que o robô irá executar de acordo com a programação, observando partes da programação onde o robô efetua cálculos e faz o desenho dos polígonos. Na programação temos também o Driver de Motor (L298N) que controla a direção e a velocidade dos motores de corrente contínua, permitindo que o Arduino controle o movimento do robô. Assim teremos uma discussão em que os alunos compartilham suas impressões sobre o funcionamento do robô, o que

acharam mais interessante e como acreditam que essa ferramenta pode aprimorar o entendimento da geometria. O instrutor conclui resumindo os pontos principais abordados, reforçando a importância da robótica como uma ferramenta educacional, e destacando como o conhecimento adquirido pode ser aplicado em projetos futuros. A participação dos alunos, sua curiosidade e suas contribuições durante a discussão são consideradas na avaliação da atividade.

3.2 Atividade 2: Reconhecimento da existência de triângulos

Esta atividade tem como objetivo compreender os critérios que determinam a existência de triângulos, explorar as propriedades matemáticas fundamentais que definem sua existência e aprimorar a compreensão da geometria através da aplicação prática de tecnologia utilizando o Arduino.

Em um primeiro momento, são realizadas aulas de revisão sobre o que é um triângulo e como verificar se três segmentos de reta formam um através da condição de existência de triângulos seguida de exercícios ao final de cada explicação. Em seguida, inicia-se a parte prática, onde os alunos aplicam o conteúdo aprendido utilizando o Arduino. Eles terão aulas básicas de programação no Arduino, onde aprenderão a escrever códigos que implementam os critérios de existência de triângulos. Esse processo permitirá que os alunos visualizem e testem suas soluções, consolidando o conhecimento teórico por meio de experiências práticas e interativas.

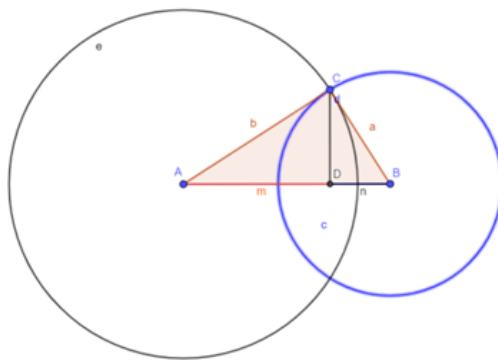
Condição de existência: Três segmentos de retas de medidas a , b e c formam um triângulo se, e somente se,

$$a + b > c; a + c > b; b + c > a$$

Essas condições podem ser visualizadas geometricamente, para isso o professor deve utilizar régua e compasso. O processo começa desenhando um segmento de reta AB com uma medida qualquer c . Em seguida, desenha-se outro segmento de reta BC com uma medida a . Finalmente, desenha-se um segmento de reta CA com uma medida b . Com o compasso, a ponta seca é colocada no ponto A e aberta para a medida do lado b . Desenha-se um arco que intersecta o segmento AB . Em seguida, a ponta seca do

compasso é colocada no ponto B , abrindo-o para a medida do lado a e desenhando um arco que intersecta o segmento AB , conforme a Figura 3.1.

Figura 3.1: Projeção dos lados b e a no terceiro lado c



Fonte: arquivo pessoal

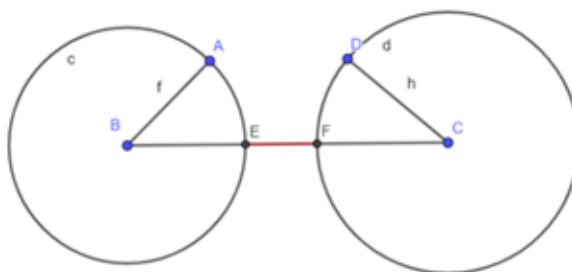
Observando a Figura 3.1 é possível ver que a soma das medidas a e b será maior que lado de medida c . De fato, como $b > m$ e $a > n$, então

$$b + a > m + n = c \implies a + b > c$$

Com isso os alunos podem visualizar e entender de forma geométrica as condições de existência dos triângulos, reforçando os conceitos teóricos de maneira concreta e intuitiva. De modo análogo é possível para mostrar para as demais.

Essas desigualdades garantem que os segmentos possam se conectar de maneira a formar uma figura fechada com três lados e três ângulos internos, caso contrário, o triângulo não se conecta nas extremidades (veja a Figura 3.2).

Figura 3.2: Seguintos que não formam triângulo



Fonte: arquivo pessoal

Como atividade o professor poderá dividir a sala em grupos e distribuir conjuntos de segmentos de papel ou cartão com comprimentos variados para cada equipe, incluindo

medidas como 3cm , 4cm , 5cm ; 2cm , 6cm , 7cm , entre outras. Os alunos terão a tarefa de escolher três segmentos e verificar se eles irão formar um triângulo ou não, aplicando o conteúdo principal desta. Após alguns minutos de trabalho, cada grupo poderá apresentar seus resultados onde serão discutidos e analisados, verificando possíveis os erros de interpretação, para possível intervenção do professor em eventuais dúvidas. Em seguida, cada grupo poderá escolher um conjunto de segmentos que cumpriu as desigualdades para desenhá-los em papel milimetrado utilizando régua e compasso formando um triângulo como já foi explicado anteriormente. Pode ser proposto, como tarefa para casa, que os alunos encontrem exemplos de triângulos em suas casas ou em livros e validem as medidas utilizando as desigualdades triangulares.

3.2.1 Utilizando o Arduino no reconhecimento da existência de triângulos

Para fortalecer essa compreensão, na próxima etapa, os alunos terão acesso a robôs equipados com teclado 4x4, uma tela LCD 16x2 e dois motores, todos devidamente integrados à placa principal Arduino Uno e pré-programados. Os alunos serão desafiados a desenvolver, no ambiente de programação do Arduino, um código responsável pela verificação e reconhecimento da condição de existência de triângulos com auxílio do professor.

Para essa atividade sugerisse levar os alunos para a sala de informática ou poderá ser disponibilizado notebooks na sala de aula com acesso à internet e ao ambiente de programação do Arduino, podendo o professor fazer as adaptações de acordo com a realidade da escola.

Nesta parte da atividade os alunos deverão ter acesso as instruções detalhadas em um arquivo no formato PDF além da explicação do professor. Primeiramente, deverá ser explicado os conceitos básicos da programação no Arduino e como implementar as condições de existência de um triângulo no código. Posteriormente, os alunos irão utilizar o ambiente de programação do Arduino, previamente instalado, para iniciar a implementação dos seus próprios códigos.

O foco principal será a integração de uma funcionalidade que verifique a existência de triângulos com base em princípios geométricos, como a desigualdade triangular. Eles serão guiados a escrever as condições em código, com auxílio do professor, que refletem esses conceitos matemáticos, aplicando assim a lógica de programação para resolver um

problema real. Durante o processo, eles terão acesso contínuo à internet para consulta e pesquisa, o que facilitará a exploração autônoma de soluções alternativas e o refinamento do código.

A seguir, o detalhamento de como o código responsável pela verificação da condição de existência dos triângulos deve ler, interpretar e processar os dados inseridos. Os alunos deverão ser guiados na criação dessa parte do código, que reconhece se a soma de quaisquer dois lados era maior que o terceiro, uma condição essencial para a formação de um triângulo. Eles irão aprender como inserir os dados (os lados do triângulo) pelo teclado e fazer a leitura deles através da programação no Arduino, como esses dados serão processados pela placa Arduino utilizando um reconhecimento por meio da condição de existência e como exibir os resultados no display do robô.

Uma forma ideal e mais complexa de leitura de dados inseridos através do teclado pode ser realizada através da criação de uma função do tipo *displayInputTriangleSide1()* que tem a responsabilidade de preparar o sistema para receber a entrada do primeiro lado do triângulo. Quando chamada, essa função limpa o display LCD através do comando *lcd.clear()*, exibe a mensagem “Lado 1:” e atualiza o estado atual da máquina de estados para *INPUT_TRIANGLE_SIDE1*, indicando que o sistema está pronto para receber a entrada do valor para o primeiro lado. Além disso, a variável *userInput* é inicializada como uma string vazia para armazenar o valor digitado pelo usuário.

Uma segunda função, *handleInputTriangleSide1(charkey)*, lida com a entrada do usuário enquanto o sistema está no estado *INPUT_TRIANGLE_SIDE1*. Esta função verifica se a tecla pressionada é o símbolo '#', utilizado para confirmar a entrada. Se o usuário pressionar '#', a função verifica se algum valor foi digitado, caso positivo, a entrada é convertida para um inteiro e armazenada na variável *side1* e o sistema é direcionado para a próxima etapa, que envolve a entrada do segundo lado. Se nenhum valor foi digitado, a função exibe uma mensagem de “Entrada inválida” no display, aguarda dois segundos, e retorna para o estado de solicitação do primeiro lado. Além disso, se a tecla pressionada for um dígito numérico, ela é adicionada à variável *userInput* e exibida no display, permitindo ao usuário ver o número que está digitando.

Esta explicação proporciona uma compreensão clara de como o sistema reconhece a entrada dos segmentos de reta, assegurando que os dados inseridos sejam corretos antes de verificar se formam um triângulo. Isso é útil, pois os alunos irão programar inicialmente

apenas a verificação da condição de existência de triângulos, mantendo o código simples e exigindo que os alunos alterem os valores diretamente no código sempre que mudarem as medidas dos lados.

Com esta explicação, o professor pode propor uma atividade que tenha o objetivo de incentivar os alunos a criar uma forma de realizar a leitura dos dados diretamente no robô, sem precisar alterar o código. Dessa forma, a inserção dos valores deverá ocorrer por meio do teclado, sendo visível na tela LCD, sem a necessidade de realizar o upload do código repetidamente, pois essa parte estará no *void loop()* e será executada continuamente enquanto o robô estiver ligado.

Logo após, será explicado como o código faz a verificação da condição de existência de triângulos, através na função *checkTriangle(intside1, intside2, intside3)*.

A função *checkTriangle(intside1, intside2, intside3)* verifica se os três segmentos de reta, representados pelas variáveis *side1*, *side2* e *side3*, podem formar um triângulo. Esta função é chamada após a inserção dos três lados do triângulo.

Primeiramente, a função começa limpando o display LCD utilizando *lcd.clear()*, o que garante que qualquer mensagem anterior seja removida, preparando o display para exibir o novo resultado.

Em seguida, a função verifica se os três segmentos podem formar um triângulo. Para isso, ela utiliza a condição *if(side1 + side2 > side3 && side1 + side3 > side2 && side2 + side3 > side1)*. Esta condição aplica a desigualdade triangular, que é uma regra fundamental na geometria. Se todas as três condições da desigualdade forem verdadeiras, então os segmentos podem formar um triângulo.

Se a condição for satisfeita, a função exibe a mensagem “*Forma triângulo*” no display utilizando *lcd.print(“Forma triângulo”)*. Caso contrário, se qualquer uma das condições da desigualdade não for satisfeita, a função exibe “*Nao forma*” no display com *lcd.print(“Nao forma”)*.

Após exibir a mensagem correspondente, a função aguarda dois segundos, *delay(2000)*, permitindo que o usuário leia o resultado antes de prosseguir. Finalmente, a função chama *displayMainMenu()* para retornar ao menu principal do sistema, permitindo ao usuário iniciar uma nova verificação ou realizar outras operações. Na Figura 3.3 se encontra a função *checkTriangle(intside1, intside2, intside3)*.

Esta função exemplifica como os conceitos de programação e geometria podem ser

Figura 3.3: Verificação da condição de existência do triângulo no código.

```

392 void checkTriangle(int side1, int side2, int side3) {
393     lcd.clear();
394     if (side1 + side2 > side3 && side1 + side3 > side2 && side2 + side3 > side1) {
395         lcd.print("Forma triangulo");
396     } else {
397         lcd.print("Nao forma");
398     }
399     delay(2000);
400     displayMainMenu();
401 }
402

```

Fonte: arquivo pessoal

integrados para resolver problemas práticos. Os alunos poderão ter contato direto com a lógica condicional e como ela é aplicada para verificar propriedades geométricas, reforçando seu entendimento tanto de programação quanto de geometria.

Com esse entendimento, como forma de atividade, os grupos de alunos podem começar a desenvolver seus próprios códigos para verificar a condição de existência de triângulos conforme as Figuras 3.4 e 3.5. Utilizando o ambiente de programação do Arduino, eles escreverão as instruções necessárias para implementar a verificação baseada na desigualdade triangular. Após eles poderão desenvolver os códigos e inserir os dados dos segmentos de reta que receberam no início da atividade. Dessa forma, eles podem testar seus códigos para identificar possíveis erros e garantir que os segmentos formassem triângulos corretamente. Utilizando o robô programado, os alunos puderam verificar de maneira rápida e eficiente a validade dos triângulos formados pelos segmentos de reta fornecidos.

Figura 3.4: Função criada para leitura do lado a .

```

void ValorLadoA() {
    lcd.clear();
    lcd.print("Lado 1:");
    currentState = inserir_ladoA_triangulo;
    userInput = "";
}

void VerificaLadoA(char key) {
    if (key == '#') {
        if (userInput.length() > 0) {
            a = userInput.toInt();
            ValorLadoB();
        } else {
            lcd.clear();
            lcd.print("Entrada invalida");
            delay(2000);
            ValorLadoA();
        }
    } else if (isdigit(key)) {
        userInput += key;
        lcd.print(key);
    }
}
}

```

Fonte: arquivo pessoal

Figura 3.5: Função criada para verificação da condição de existência de um triângulo.

```

void VerificaCondicaoExistencia(int a, int b, int c) {
    lcd.clear();
    if (a+b>c && a+c>b && b+c>a) {
        lcd.print("Forma triangulo");
    } else {
        lcd.print("Nao forma");
    }
    delay(2000);
    displayMainMenu();
}

```

Fonte: arquivo pessoal

Dessa forma, a conexão entre a teoria e a prática não só consolidará os conhecimentos adquiridos, mas também mostrará a relevância e a aplicabilidade da geometria em diferentes contextos.

Cada grupo irá de forma colaborativa, discutir entre si e inserir diferentes combinações de segmentos. O robô, ao receber os dados, realizará a validação e imediatamente mostrar no display se os segmentos formavam um triângulo ou não.

A atividade reforça o conhecimento teórico sobre a condição de existência de triângulos, e permite que os alunos vejam uma aplicação prática e tecnológica desse conhecimento. Eles vão poder observar o processo de validação em tempo real, o que facilitará a compreensão e a fixação dos conceitos. Além disso, o uso da tecnologia e a interatividade proporcionada pelos robôs tornaram a experiência mais envolvente e significativa para os alunos.

3.3 Atividade 3: cálculo de área e o perímetro

O objetivo dessa atividade é fornecer uma explicação clara e concisa sobre o perímetro e a área de polígonos, especialmente polígonos regulares, ensinando como calcular essas medidas. Além disso, visa demonstrar como esses conceitos podem ser aplicados através da programação utilizando o Arduino. Para alunos do 9º ano do ensino fundamental, compreender esses conceitos é de grande importância, pois eles constituem a base para tópicos mais avançados em matemática e outras disciplinas relacionadas, como física e engenharia.

Inicialmente, é realizada uma revisão do conteúdo sobre o perímetro e a área de polígonos, explorando suas definições, propriedades, fórmulas e a importância desses conceitos no contexto educacional. de polígonos seguida de exercícios. Em seguida, inicia-se a parte prática, onde os alunos aplicam o conteúdo aprendido utilizando o Arduino. Eles terão aulas básicas de programação no Arduino, onde aprenderão a escrever códigos que implementam o cálculo de perímetro e área. Esse processo permitirá que os alunos visualizem e testem suas soluções, consolidando o conhecimento teórico por meio de experiências práticas e interativas.

Ao ensinar perímetro e área, é importante usar exemplos concretos e problemas práticos que envolvam situações reais. Isso ajuda os alunos a compreender a relevância desses conceitos e a aplicá-los de maneira eficaz. Além disso, a visualização das figuras geométricas e a manipulação de modelos físicos podem facilitar a compreensão dos conceitos abstratos, pois um desafio comum é a dificuldade em visualizar a relação entre as medidas lineares e as medidas de superfície.

Uma estratégia eficaz é a resolução de problemas em grupo, onde os alunos poderão discutir e compartilhar diferentes abordagens para resolver questões relacionadas a perí-

metro e área. Essa colaboração promove um ambiente de aprendizado ativo e engajado, onde os alunos podem aprender uns com os outros e desenvolver um entendimento mais profundo dos conceitos.

Uma proposta de atividade é dividir os alunos em grupos onde cada grupo irá receber materiais como réguas, transferidores, compasso e segmentos de papel de diferentes comprimentos para construir polígonos regulares, além do material impresso com detalhamento das atividades e exercícios. Os alunos irão construir os polígonos de acordo com o material impresso. Eles irão medir os lados e calcular os perímetros utilizando as fórmulas específicas para cada polígono.

Após calcular o perímetro e a área, os grupos poderão comparar os resultados entre si, verificando se os cálculos estavam corretos. Poderão ser discutidos os diferentes métodos e estratégias utilizadas para medir e calcular, bem como os desafios enfrentados e as soluções encontradas. Para contextualizar o aprendizado, os grupos poderão resolver problemas práticos. Exemplos de problemas: Calcular a quantidade de material necessário para cercar um jardim de formato triangular. Determinar a quantidade de tinta necessária para pintar uma parede quadrada. Verificar se determinada quantidade de azulejos seria suficiente para cobrir o piso de um hexágono.

3.3.1 Utilizando o Arduino no cálculo de área e o perímetro

Para complementar essas abordagens, o professor pode levar os alunos para fazer a atividade em uma sala de informática ou na própria sala de aula, com acesso à internet e computadores, motivando-os a utilizarem a lógica de programação do Arduino, aplicando os conceitos de perímetro e área, e posteriormente tendo acesso ao robô. Seguindo a metodologia utilizada na atividade de validação de triângulos, os alunos irão programar códigos simples para calcular o perímetro e a área de diferentes figuras geométricas. Essa abordagem didática permitirá aos alunos aplicar diretamente os conceitos teóricos em um contexto tecnológico, reforçando o aprendizado e desenvolvendo habilidades em programação.

Nesta etapa da atividade os alunos serão guiados a utilizar o Arduino na escola para desenvolver códigos que calculem área e perímetro de diferentes figuras geométricas. Essa atividade tem como objetivo principal integrar o aprendizado teórico dos conceitos de perímetro e área com uma aplicação prática através do Arduino.

Com os alunos já instruídos sobre as fórmulas matemáticas necessárias para calcular perímetro e área de figuras como retângulos, quadrados, triângulos e polígonos regulares em aulas anteriores eles irão aprender a utilizar variáveis, operadores matemáticos e funções de maneira adequada para realizar os cálculos necessários. Além disso, explorarão como organizar seu código de forma clara e eficiente, seguindo os princípios da programação estruturada.

Durante a execução dos códigos, os alunos poderão testar diferentes valores de entrada para verificar se os resultados dos cálculos estão corretos.

Com base nesse entendimento teórico, os alunos utilizarão os recursos disponíveis para escrever códigos simples no ambiente do Arduino. Como sugestão de atividade, o professor pode pedir para cada grupo de alunos programar funções específicas para calcular o perímetro e a área das figuras geométricas mencionadas. Os códigos das Figuras 23 e 24 poderão ser usados como explicação de como utilizar o ambiente do Arduino para realizar cálculo de perímetro e área de um polígono regular de n lados e de lado l de medida usando o ambiente do Arduino.

Os códigos das Figuras 3.6 e 3.7 utilizam um Arduino em conjunto com um display LCD I2C para calcular e exibir o perímetro e área de um polígono regular, neste caso um quadrado, cujo número de lados é 4 e o comprimento de cada lado é 5.0 cm.

Figura 3.6: Exemplo de código que pode ser explicado sobre cálculo de perímetro de polígonos regulares.

```
1  #include <Wire.h>
2  #include <LiquidCrystal_I2C.h>
3
4  LiquidCrystal_I2C lcd(0x27, 16, 2);
5
6  const int numLados = 4;
7  const float comprimentoLado = 5.0;
8
9  float perimetro;
10
11 void setup() {
12     lcd.begin(16, 2);
13     lcd.backlight();
14
15     perimetro = numLados * comprimentoLado;
16
17     // Exibe os resultados no LCD
18     lcd.setCursor(0, 0);
19     lcd.print("Perimetro: ");
20     lcd.print(perimetro);
21     lcd.print(" cm");
22 }
23
24 void loop() {
25     // O loop está vazio porque todos os cálculos são feitos no setup
26 }
27
```

Fonte: arquivo pessoal

Figura 3.7: Exemplo de código que pode ser explicado sobre cálculo de área de polígonos regulares.

```
1  #include <Wire.h>
2  #include <LiquidCrystal_I2C.h>
3
4  LiquidCrystal_I2C lcd(0x27, 16, 2);
5
6  const int numLados = 4;
7  const float comprimentoLado = 5.0;
8
9  float area;
10
11 void setup() {
12     lcd.begin(16, 2);
13     lcd.backlight();
14
15     area = (numLados * pow(comprimentoLado, 2)) / (4 * tan(PI / numLados));
16
17     lcd.setCursor(0, 0);
18     lcd.print("Area: ");
19     lcd.print(area);
20     lcd.print(" cm2");
21 }
22
23 void loop() {
24     // O loop está vazio porque todos os cálculos são feitos no setup
25 }
26
```

Fonte: arquivo pessoal

Primeiramente, são incluídas as bibliotecas necessárias: *Wire.h* para comunicação I2C e *LiquidCrystal_I2C.h* para controle do display LCD. O display é inicializado com o endereço I2C padrão 0x27 e configurado para ter 16 colunas por 2 linhas.

No setup do Arduino, que é executado apenas uma vez ao ligar o dispositivo, a comunicação com o LCD é iniciada e a luz de fundo é ligada. Em seguida, o perímetro do quadrado é calculado multiplicando-se o número de lados (*numLados*) pelo comprimento de um lado (*comprimentoLado*).

O resultado do cálculo, ou seja, o valor do perímetro, é então exibido no display LCD. Primeiro, o cursor é posicionado na posição (0, 0) do display usando *lcd.setCursor(0, 0)*. Depois, o texto “Perimetro:” é impresso seguido pelo valor calculado do perímetro e a unidade de medida, neste caso “cm”.

O loop principal (*loop()*) não contém nenhuma instrução, pois todos os cálculos e exibições são realizados no setup e não há necessidade de repeti-los continuamente.

Esses códigos demonstram como utilizar um Arduino para realizar cálculos simples e exibir resultados em um display LCD I2C de forma clara e eficiente, sendo útil para aplicações onde a interação com o usuário através de um display é necessária para mostrar informações calculadas.

Para permitir que os alunos insiram diretamente os dados sem a necessidade de alterar as variáveis no código, é possível utilizar um teclado 4x4 e colocar a lógica principal no *loop()*, onde o Arduino aguardará a entrada do usuário e realizará os cálculos dinamicamente.

No setup do Arduino, a comunicação com o LCD é iniciada, a luz de fundo é ativada e uma mensagem solicitando que o usuário digite o número de lados é exibida. O loop principal então aguarda a entrada do usuário através do teclado. Quando uma tecla é pressionada, a função *keypad.getKey()* captura a tecla pressionada. Se a tecla # for pressionada, o programa verifica se o número de lados já foi inserido, utilizando a variável *ladoInserido*. Se foi inserido, a string de entrada é convertida para um inteiro, armazenado na variável *numLados*, e o programa solicita o comprimento do lado. Caso o número de lados já tenha sido inserido, a string de entrada é convertida para um *float*, armazenada na variável *comprimentoLado*, e a função *calcularArea()* é chamada.

A função *calcularArea()* limpa o display LCD, calcula a área do polígono utilizando a fórmula apropriada e exibe a área calculada no display LCD. Este código torna o processo

de inserção de dados interativo e dinâmico, permitindo que os alunos insiram diretamente os valores necessários para calcular a área de um polígono regular, sem a necessidade de alterar variáveis no código. De modo análogo pode-se criar um código parecido para o cálculo do perímetro.

Ao final da atividade, os alunos poderão compartilhar seus códigos com os colegas e discutir suas abordagens para resolver os desafios encontrados. Essa colaboração poderá promover um ambiente de aprendizado interativo, onde todos podem aprender uns com os outros e refinarem suas habilidades de programação e compreensão dos conceitos matemáticos.

3.4 Atividade 4: classificação de triângulo

A classificação dos triângulos é um tema importante na geometria. No 9º ano, é essencial que os estudantes compreendam as diferentes formas de classificação dos triângulos, seja pelos seus lados ou pelos seus ângulos. Esse conhecimento não só fortalece a base da geometria, mas também prepara os alunos para tópicos mais avançados. Os triângulos podem ser classificados de duas maneiras principais: pela medida de seus lados e pela medida de seus ângulos.

Classificação pelos Lados:

O triângulo equilátero é caracterizado por possuir três lados de mesma medida. Consequentemente, todos os ângulos internos são congruentes, medindo 60 graus cada um. Este tipo de triângulo é simétrico e apresenta propriedades únicas, como a igualdade de seus ângulos e a mediana que coincide com altura, bissetriz e mediatriz;

O triângulo isósceles é definido como um triângulo com dois lados de mesma medida. Os ângulos opostos aos lados congruentes também possuem a mesma medida. Este tipo de triângulo é importante para estudar simetria e a relação entre lados e ângulos;

O triângulo é chamado de escaleno quando possui todos os lados e ângulos com medidas diferentes. Este triângulo não apresenta simetrias específicas, tornando-o um bom exemplo para a aplicação de teoremas gerais da geometria como a fórmula de Heron.

Classificação pelos Ângulos:

Um triângulo é denominado acutângulo se possui todos os ângulos internos com medidas menores que 90 graus. Este triângulo pode ser isósceles, equilátero ou escaleno,

dependendo das medidas dos lados;

O triângulo retângulo é definido pela presença de um ângulo interno de 90 graus. Os triângulos retângulos são a base para o estudo da trigonometria e do Teorema de Pitágoras. Os lados que formam o ângulo reto são chamados de catetos, e o lado oposto ao ângulo reto é a hipotenusa.

O triângulo obtusângulo apresenta um ângulo interno maior que 90 graus.

Os triângulos retângulos e obtusângulos não podem possuir mais de um ângulo igual a 90 graus ou maior, uma vez que a soma dos ângulos internos de um triângulo sempre deve ser igual a 180 graus.

Para cada tipo de triângulo, há propriedades e teoremas específicos que podem ser explorados. Por exemplo, no triângulo retângulo, o Teorema de Pitágoras estabelece uma relação entre os lados, enquanto nos triângulos isósceles e equiláteros, a simetria pode ser usada para deduzir propriedades dos ângulos e das alturas.

É importante que os professores incentivem os alunos a identificar e classificar triângulos em diversas situações práticas e teóricas. Isso pode ser feito através de atividades que envolvam a medição de ângulos e lados, bem como a construção e a análise de triângulos em diferentes contextos.

Ao dominar a classificação dos triângulos, os alunos desenvolvem uma compreensão sólida das bases da geometria, que é essencial para o estudo de figuras mais complexas e para a resolução de problemas geométricos mais avançados. A prática contínua e a aplicação desses conceitos em problemas variados ajudam a consolidar o aprendizado e a desenvolver habilidades analíticas importantes para o avanço nos estudos matemáticos.

Para ensinar a classificação dos triângulos de forma prática e envolvente, propomos uma atividade para o professor que combina construção, medição e uso de tecnologia. Esta atividade pode ajudar os alunos a compreenderem as propriedades dos triângulos de maneira dinâmica e interativa.

No início da aula, deve haver uma breve introdução sobre os diferentes tipos de triângulos, tanto pela classificação dos lados quanto pela classificação dos ângulos. Utilize exemplos visuais e desenhos no quadro para ilustrar cada tipo. Em seguida, divida os alunos em grupos e distribua os materiais necessários, como régua, compasso, transferidor, papel milimetrado, lápis e borracha. Cada grupo deve construir três tipos de triângulos diferentes: um triângulo equilátero com lados de 6 cm, um triângulo isósceles com lados

de 5 cm, 5 cm e 8 cm, e um triângulo escaleno com lados de 3 cm, 4 cm e 5 cm.

Instrua os alunos a usar régua e compasso para garantir a precisão nas construções. Após construírem os triângulos, peça para que identifiquem e anotem os ângulos internos utilizando o transferidor. Cada grupo deve medir e registrar os lados e ângulos dos triângulos construídos. Em seguida, compare os resultados com as propriedades teóricas: verificar se os ângulos de um triângulo equilátero são todos iguais a 60 graus, confirmar que os triângulos isósceles têm dois lados de mesma medida e os ângulos opostos a esses lados são congruentes, e observar que no triângulo escaleno, todos os lados e ângulos são diferentes e um deles mede 90 graus.

Pode ser introduzido o Teorema de Pitágoras através do triângulo escaleno que construíram. Eles devem calcular a hipotenusa e verificar se o valor obtido coincide com a medida real. Mostre, por meio das medidas, que o teorema não se aplica ao triângulo equilátero.

Após as atividades práticas, reúna os alunos para uma discussão em grupo. Peça a cada grupo para compartilhar suas observações e resultados. Destaque como os diferentes tipos de triângulos se comportam em relação às suas propriedades de lados e ângulos. Incentive os alunos a fazerem perguntas e a refletirem sobre o que aprenderam. Para concluir, proponha um desafio final onde os alunos devem identificar e classificar triângulos em diferentes contextos, como em figuras geométricas complexas ou em situações do mundo real. Eles podem desenhar ou fotografar triângulos que encontrarem em objetos do cotidiano e classificá-los de acordo com os critérios aprendidos.

3.4.1 Utilizando o Arduino na classificação de triângulo

Para a próxima etapa, é proposto uma atividade que integra conceitos de geometria com programação para ensinar a classificação dos triângulos utilizando o Arduino de forma prática e interativa. Os materiais necessários incluem uma placa Arduino Uno, um display LCD I2C 16x2, um protoboard e fios jumper. Além disso, os alunos precisarão de régua, transferidor, papéis e canetas.

O objetivo é que os alunos construam um sistema que leia as medidas dos lados de um triângulo e determine sua classificação com base nas medidas inseridas diretamente no código do Arduino. Eles visualizarão o tipo de triângulo no display LCD após a inserção dos comprimentos dos lados.

Logo após, deve-se explicar o código que será carregado no Arduino. O código permite que os alunos insiram os comprimentos dos lados do triângulo diretamente no código, e o Arduino exibirá o tipo de triângulo no display LCD.

Cada grupo pode construir três tipos de triângulos diferentes no papel: um triângulo equilátero com lados de 6 cm, um triângulo isósceles com lados de 5 cm, 5 cm e 8 cm, e um triângulo escaleno com lados de 3 cm, 4 cm e 5 cm. Régua e transferidor garantem precisão nas construções e medições.

Após construírem e medirem os triângulos, os alunos deverão inserir os valores dos lados diretamente no código do Arduino. O Arduino processa as entradas e exibe a classificação do triângulo no display LCD. Os alunos verificam se a classificação exibida corresponde aos triângulos construídos. A discussão dos resultados com a turma enfatiza as propriedades dos diferentes tipos de triângulos.

Figura 3.8: Exemplo de código para a explicação

```

24
25 void classificarTriangulo(int lado1, int lado2, int lado3) {
26     lcd.clear();
27
28     if (lado1 == lado2 && lado2 == lado3) {
29         lcd.print("Equilatero");
30     } else if (lado1 == lado2 || lado2 == lado3 || lado1 == lado3) {
31         lcd.print("Isosceles");
32     } else {
33         lcd.print("Escaleno");
34     }
35 }

```

Fonte: arquivo pessoal

O código da figura 25 começa incluindo as bibliotecas necessárias: *Wire.h* e *LiquidCrystal_I2C.h*. Em seguida, inicializamos o display LCD com o endereço *I2C0x27* e definimos que ele terá 16 colunas e 2 linhas.

As variáveis *lado1*, *lado2* e *lado3* armazenam os comprimentos dos três lados do triângulo. Estes valores são inseridos diretamente no código.

No *setup()*, inicializa-se a comunicação com o LCD e é ativado a luz de fundo. Em seguida, é exibido uma mensagem inicial no LCD e chamada a função *classificarTriangulo()* passando os comprimentos dos três lados como argumentos.

A função *classificarTriangulo()* limpa o display LCD e compara os comprimentos dos lados. Se todos os três lados forem iguais, o triângulo é classificado como equilátero

e essa mensagem é exibida no LCD. Se dois lados forem iguais, o triângulo é classificado como isósceles. Se todos os lados forem diferentes, o triângulo é classificado como escaleno. De modo análogo pode-se criar o código para classificação segundo os ângulos. Ao invés de ler as medidas dos lados o código deve ler as medidas dos ângulos. A função *classificarTriangulo()* seguirá o mesmo raciocínio só que desta vez deverá exibir se o triângulo é equilátero, isósceles ou escaleno.

Esta atividade prática permite que os alunos vejam, de forma interativa e visual, como a matemática pode ser aplicada na programação para classificar triângulos, reforçando o aprendizado teórico com uma experiência prática concreta.

Por fim, os alunos discutem as dificuldades encontradas e as lições aprendidas. Destaque a importância de integrar conhecimentos de diferentes áreas (matemática, eletrônica e programação) para resolver problemas práticos. Encoraje os alunos a pensar em outras aplicações possíveis para o Arduino em projetos matemáticos e científicos. Esta atividade proporciona uma experiência rica e multifacetada, permitindo que os alunos vejam a aplicabilidade da matemática em contextos tecnológicos e práticos. Ao usar o Arduino, os alunos não apenas reforçam seus conhecimentos sobre triângulos, mas também desenvolvem habilidades em programação e eletrônica, tornando o aprendizado mais dinâmico e significativo.

3.5 Atividade 5: cálculo de área e perímetro com o robô desenhador de polígonos

Nesta atividade prática, os alunos escolherão o polígono a ser desenhado pelo robô desenhador. As opções incluem triângulo, quadrado, pentágono, hexágono, heptágono e octógono. Após selecionar o polígono, o robô começará a desenhar a figura no chão usando um pincel fixado nele. Os alunos observarão o robô mover-se para frente e realizar as rotações necessárias para completar o polígono. Depois, eles usarão uma régua ou fita métrica para medir o comprimento de cada lado da figura desenhada. A partir disso, eles poderão calcular área e perímetro do polígono, usando a fórmula específica para a figura geométrica selecionada.

Após concluírem os cálculos, os alunos deverão comparar os resultados obtidos com os valores calculados pelo robô. Finalmente, os resultados obtidos pelos grupos serão

CAPÍTULO 3. INTEGRAÇÃO DA MAT. E TEC. NO ENSINO

apresentados para a turma, explicando o processo seguido, as medições realizadas, os cálculos do perímetro e da área, e as conclusões tiradas. Esta etapa permitirá que os alunos desenvolvam suas habilidades de comunicação e compartilhem suas experiências.

O código fornecido é para um robô que desenha diferentes polígonos e calcula áreas e perímetros usando um display LCD, um teclado numérico 4x4, e dois motores para movimentação. Logo a seguir, uma explicação do código em anexo de forma detalhada.

Primeiro, o código inclui as bibliotecas necessárias: *Wire.h*, *LiquidCrystal_I2C.h*, *Keypad.h* e *math.h*.

Em seguida, o display LCD é inicializado com o endereço I2C definido e tamanho 16x2. A configuração do teclado numérico é feita, definindo as teclas e os pinos aos quais estão conectadas. A biblioteca *Keypad* é usada para mapear as teclas e facilitar a leitura dos inputs.

Os pinos dos motores são definidos: quatro pinos no total, dois para cada motor, que controlam a direção de rotação dos motores. Um enum *MenuState* é definido para gerenciar o estado atual do menu do robô. Os estados incluem a tela de boas-vindas, menus principais e específicos para desenho e cálculos, e estados de entrada para receber os valores dos lados dos polígonos.

No setup, o display LCD é inicializado e a luz de fundo é ligada. A mensagem de boas-vindas é exibida e, após um breve delay, o menu principal é mostrado. Os pinos dos motores são configurados como saída.

No loop, o código monitora as teclas pressionadas no teclado numérico. Dependendo do estado atual do menu, diferentes funções são chamadas para processar a entrada do usuário.

A função *displayWelcomeMessage* limpa o display, mostra uma mensagem de boas-vindas e depois chama *displayMainMenu*, que apresenta as opções de “Desenho” e “Calculadora”. A função *handleMainMenu* processa a escolha do usuário: se “Desenho” é selecionado, chama *displayDrawMenu*, que exibe opções para diferentes polígonos, atualizando o display a cada dois segundos para mostrar mais opções.

Na função *handleDrawMenu*, o código define o número de lados e o comprimento dos lados do polígono a ser desenhado com base na escolha do usuário. A função *drawPolygon* é então chamada, que controla os motores para desenhá-lo. O robô move-se para frente por um determinado tempo, vira um ângulo calculado com base no número de lados

e repete esse processo até completar o desenho do polígono.

Figura 3.9: Função *drawPolygon*

```

187 void drawPolygon(int sides, int sideLength) {
188     lcd.clear();
189     lcd.print("Desenhando");
190     // Código para controlar os motores para desenhar o polígono
191     delay(2000);
192     for (int i = 0; i < sides; i++) {
193         moveForward(sideLength * 100); // Ajuste o tempo conforme necessário
194         delay(2000); // Ajuste o tempo conforme necessário
195         turn(360/sides); // Rotação do robô
196         delay(2000); // Ajuste o tempo conforme necessário
197     }
198     displayMainMenu();
199 }
200

```

Fonte: arquivo pessoal

As funções *moveForward*, *turn*, e *stopMotors* controlam o movimento do robô: *moveForward* ativa ambos os motores para mover o robô para frente, *turn* faz o robô girar em um ângulo específico, e *stopMotors* para o movimento desligando os motores.

Para os cálculos de área e perímetro, a função *displayResultMenu* apresenta opções no display. Quando uma dessas opções é selecionada, as funções *displayInputSides* e *displayInputSideLength* solicitam e recebem a entrada do usuário para o número de lados e o comprimento dos lados, respectivamente. As funções *handleInputSides* e *handleInputSideLength* processam essas entradas e, dependendo da opção selecionada, chamam *calculateArea* ou *calculatePerimeter*.

A função *calculateArea* usa a fórmula específica para calcular a área de um polígono regular:

$$A_{p_n} = \frac{n \cdot l^2}{4 \cdot \operatorname{tg}\left(\frac{\pi}{n}\right)},$$

onde n é o número de lados e l é o comprimento do lado. A função *calculatePerimeter* simplesmente multiplica o número de lados pelo comprimento do lado para obter o perímetro. Ambas as funções exibem os resultados no display LCD.

Todo o código está estruturado de forma a permitir interação contínua e intuitiva com o robô, proporcionando uma maneira prática e divertida de aprender conceitos matemáticos enquanto se trabalha com robótica e programação.

3.5.1 Aplicação em sala de aula

A atividade foi organizada com a divisão dos alunos em grupos, de modo que cada grupo recebeu um robô com todos os componentes necessários: dois motores para movimentação, um display LCD, um pincel fixado no eixo de rotação e um teclado numérico 4x4. Após uma breve introdução sobre a função de cada componente, os alunos ligaram o robô e visualizaram a mensagem de boas-vindas no display LCD. Em seguida, o robô apresentou o menu principal, mostrado na figura 3.10, com as opções “Desenho” e “Calculadora”.

Figura 3.10: Menu principal do robô desenhador.



Fonte: arquivo pessoal

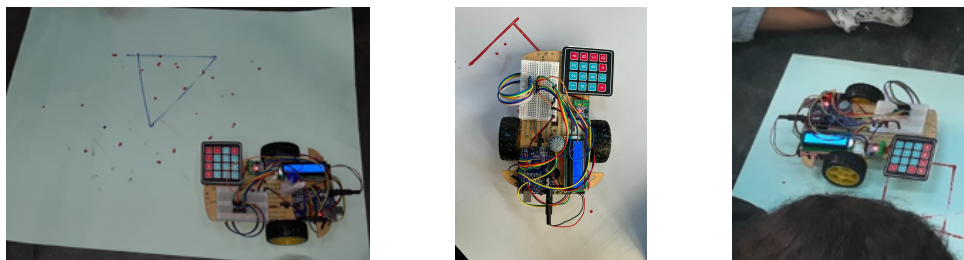
Os grupos selecionaram a opção “Desenho” no menu principal. No submenu de desenho, cada grupo escolheu um polígono a ser desenhado, conforme as instruções fornecidas em um arquivo PDF. Este PDF continha imagens e exemplos que facilitavam a compreensão da tarefa. Os alunos optaram por desenhar diferentes tipos de polígonos, como triângulos, quadrados e hexágonos.

Os robôs começaram a desenhar as figuras no chão usando o pincel. Os alunos acompanharam o processo, observando os robôs moverem-se e girarem conforme necessário para completar o polígono escolhido. A Figura 3.11 mostra o robô em funcionamento.

Após o desenho, os alunos utilizaram réguas e fitas métricas para medir o comprimento de cada lado da figura desenhada. Com essas medidas em mãos, calcularam o perímetro e a área dos polígonos utilizando as fórmulas apropriadas. Os cálculos foram registrados em uma folha de atividades para facilitar a organização dos dados.

Em seguida, os alunos utilizaram a função “Calculadora” do robô para inserir as medidas dos lados e comparar os cálculos de perímetro e área feitos manualmente com os valores fornecidos pelo robô. Durante essa etapa, discutiram as possíveis discrepâncias

Figura 3.11: Robô desenhando polígonos.



Fonte: arquivo pessoal

entre os resultados e refletiram sobre a precisão das medições e dos cálculos realizados.

Cada grupo apresentou seus resultados para a turma, explicando o processo seguido, as medições realizadas, os cálculos do perímetro e da área, e as conclusões tiradas. A atividade também possibilitou o desenvolvimento das habilidades de comunicação, pois os alunos compartilharam suas experiências e aprenderam a articular suas descobertas de forma clara e concisa.

A atividade foi um sucesso, proporcionando aos alunos uma experiência enriquecedora que uniu teoria e prática de forma dinâmica e envolvente. O uso do robô educativo não só ajudou a reforçar os conceitos matemáticos, mas também despertou o interesse dos alunos por robótica e programação. A abordagem prática facilitou a compreensão dos conceitos de perímetro e área, tornando o aprendizado mais significativo e divertido.

Conclusões e trabalhos futuros

Nessa dissertação, foi analisado o uso da robótica educacional no ensino, com foco no desenvolvimento dos conceitos da geometria plana. Para tanto, foram propostas atividades com os objetivos de aprofundar a compreensão de conceitos geométricos a partir da manipulação prática dos robôs, a fim de que os alunos possam visualizar concretamente os objetos geométricos estudados.

A aplicação das atividades propostas permitiu que os alunos desenvolvessem familiaridade com os tópicos abordados, entre eles a condição de existência e as classificações de triângulos, o conceito de polígonos e os cálculos de área e perímetro. Além disso, a abordagem prática ofereceu uma alternativa aos métodos tradicionais de ensino e proporcionou o contato dos alunos com o Arduino, de modo que eles pudessem usá-lo como ferramenta de resolução de problemas e desenvolver habilidades em programação. Assim, as aulas tornaram-se mais dinâmicas, garantindo que os alunos participassem ativamente do processo de aprendizado.

Como desafios tivemos a montagem e programação do robô, que exigem paciência, atenção aos detalhes e conhecimentos técnicos, que podem ser um desafio para alguns alunos. Temos que fazer a gestão de recursos, pois a aquisição de materiais e a organização do tempo são fatores importantes para o sucesso do projeto.

Para trabalhos futuros existe a possibilidade de expansão do escopo para outras áreas da matemática, investigando a aplicabilidade da robótica em outras áreas da matemática, como álgebra e trigonometria. Podemos considerar a oportunidade de realizar a avaliação de longo prazo, acompanhando o desempenho dos alunos ao longo do tempo para verificar a efetividade da abordagem e identificar possíveis impactos na aprendizagem. Também é interessante avaliar a capacitação dos professores para o uso das tecnologias e desenvolver novos projetos mais complexos que envolvam a integração de diferentes sensores e atuadores, ampliando as possibilidades de aplicação da robótica na educação. Aumentando a

CAPÍTULO 3. INTEGRAÇÃO DA MAT. E TEC. NO ENSINO

quantidade de polígonos que podemos trabalhar no Arduino (nesse caso trabalhamos com 6 polígonos do robô desenhador), torna-se possível realizar cálculos mais elaborados, além do cálculo de área e perímetro, como também desenhar o círculo trigonométrico entre outros.

Referências Bibliográficas

- [1] ATTROT, Maria Augusta; AYROSA, Emilie Amorim. **Robótica Educacional e suas Potencialidades**. São Paulo: Editora Educacional, 2002.
- [2] **ARDUINO**. Arduino - Home. Disponível em: <https://www.arduino.cc/>. Acesso em: 2021.
- [3] BERLINGHOFF, William P.; GOUVÊA, Fernando Q. **Matemática através dos tempos: uma história concisa**. 2. ed. Rio de Janeiro: Elsevier, 2010.
- [4] BRASIL. **Base Nacional Comum Curricular**. Ministério da Educação, 2018. Disponível em: <http://basenacionalcomum.mec.gov.br/>. Acesso em: 28 jun. 2024.
- [5] BRASIL. **Lei de Diretrizes e Bases da Educação Nacional**, Lei nº 9.394, de 20 de dezembro de 1996. Disponível em: http://www.planalto.gov.br/ccivil_03/leis/l9394.htm. Acesso em: 20 jun. 2024.
- [6] COXETER, Harold Scott MacDonald. **Introduction to Geometry**. New York: Wiley, 1969.
- [7] CURTIS, Maureen. **Robótica Educativa: Uma Nova Abordagem para a Aprendizagem de STEM**. Nova York: Springer, 2014.
- [8] DANTE, Luiz Roberto. **Matemática: Contexto & Aplicações**. São Paulo: Ática, 2005.
- [9] EUCLIDES. **Os Elementos**. Tradução e notas de Irineu Bicudo. São Paulo: Edusp, 2019.
- [10] RICHARDSON, Michael; WALLACE, Shawn. **Primeiros Passos com Arduino: A Plataforma de Prototipagem Eletrônica de Código Aberto**. Sebastopol: O'Reilly Media, 2012.

REFERÊNCIAS BIBLIOGRÁFICAS

- [11] **SOCIEDADE BRASILEIRA DE MATEMÁTICA**. Profmat - Mestrado Profissional em Matemática em Rede Nacional. Aula 10: Teorema da Existência e Unicidade do Triângulo. São Paulo: SBM, 2018.
- [12] STEWART, Ian. **The Mathematical Universe**: An Alphabetical Journey Through the Great Proofs, Problems, and Personalities. Wiley, 2001.
- [13] ———, Ian. **A Matemática da Vida**. São Paulo: Companhia das Letras, 2001.
- [14] TOOMER, Gerald James. **Ptolemy's Almagest**. London: Duckworth, 1970-1990.
- [15] BARBOSA, Joao Lucas Marques. **Geometria Euclidiana Plana**. 11. ed. Rio de Janeiro: Sociedade Brasileira de Matemática, 2012.
- [16] MAKERHERO. Blog MakerHero, 2024. Disponível em: <https://www.makerhero.com/>. Acesso em: 10 set. 2024.

Anexo A

Código utilizado

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Keypad.h>
#include <math.h>

// Defina o endereço I2C do seu módulo LCD (normalmente é 0x27 ou 0x3F)
#define I2C_ADDR 0x27

// Inicialize o display LCD com o endereço I2C e o tamanho (16x2)
LiquidCrystal_I2C lcd(I2C_ADDR, 16, 2);

// Configuração do teclado numérico
const byte ROWS = 4; // quatro linhas
const byte COLS = 4; // quatro colunas

char keys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};

byte rowPins[ROWS] = {2, 3, 4, 5}; // Pinos de linha conectados ao
teclado
byte colPins[COLS] = {6, 7, 8, 9}; // Pinos de coluna conectados ao
teclado

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

// Configuração dos motores
const int motor1Pin1 = 10;
const int motor1Pin2 = 11;
```

ANEXO A. CÓDIGO UTILIZADO

```
const int motor2Pin1 = 12;
const int motor2Pin2 = 13;

// Estado atual do menu
enum MenuState {
    WELCOME,
    MAIN_MENU,
    DRAW_MENU,
    RESULT_MENU,
    INPUT_SIDES,
    INPUT_SIDE_LENGTH,
    INPUT_TRIANGLE_SIDES,
    INPUT_TRIANGLE_SIDE1,
    INPUT_TRIANGLE_SIDE2,
    INPUT_TRIANGLE_SIDE3
};

MenuState currentState = WELCOME;
String userInput = "";
int sides = 0;
int sideLength = 0;
char nextAction = '\\0';
int side1;
int side2;
int side3;

void setup() {
    // Inicialize o display LCD
    lcd.init();

    // Ligue a luz de fundo (backlight)
    lcd.backlight();
    displayWelcomeMessage();

    // Configuração dos pinos dos motores
    pinMode(motor1Pin1, OUTPUT);
    pinMode(motor1Pin2, OUTPUT);
    pinMode(motor2Pin1, OUTPUT);
    pinMode(motor2Pin2, OUTPUT);
}

void loop() {
    char key = keypad.getKey();

    if (key != NO_KEY) {
        switch (currentState) {
            case MAIN_MENU:
                handleMainMenu(key);
                break;
        }
    }
}
```

ANEXO A. CÓDIGO UTILIZADO

```
        case DRAW_MENU:
            handleDrawMenu(key);
            break;
        case RESULT_MENU:
            handleResultMenu(key);
            break;
        case INPUT_SIDES:
            handleInputSides(key);
            break;
        case INPUT_SIDE_LENGTH:
            handleInputSideLength(key);
            break;
        case INPUT_TRIANGLE_SIDE1:
            handleInputTriangleSide1(key);
            break;
        case INPUT_TRIANGLE_SIDE2:
            handleInputTriangleSide2(key);
            break;
        case INPUT_TRIANGLE_SIDE3:
            handleInputTriangleSide3(key);
            break;
        default:
            break;
    }
}
}

void displayWelcomeMessage() {
    lcd.clear();
    lcd.print("Bem Vindo!");
    delay(2000); // Espera 2 segundos
    displayMainMenu();
}

void displayMainMenu() {
    lcd.clear();
    lcd.print("1: Desenho");
    lcd.setCursor(0, 1);
    lcd.print("2: Calculadora");
    currentState = MAIN_MENU;
}

void handleMainMenu(char key) {
    switch (key) {
        case '1':
            displayDrawMenu();
            break;
        case '2':
            displayResultMenu();
    }
}
```

ANEXO A. CÓDIGO UTILIZADO

```
        break;
    default:
        // Opção inválida, não faz nada
        break;
    }
}

void displayDrawMenu() {
    lcd.clear();
    lcd.print("1: Triangulo");
    lcd.setCursor(0, 1);
    lcd.print("2: Quadrado");
    delay(2000); // Espera 2 segundos para exibir mais opções
    lcd.clear();
    lcd.print("3: Pentagono");
    lcd.setCursor(0, 1);
    lcd.print("4: Hexagono");
    delay(2000); // Espera 2 segundos para exibir mais opções
    lcd.clear();
    lcd.print("5: Heptagono");
    lcd.setCursor(0, 1);
    lcd.print("6: Octogono");
    currentState = DRAW_MENU;
}

void handleDrawMenu(char key) {
    switch (key) {
        case '1':
            sides = 3; // Define a quantidade de lados
            sideLength = 10; // Define o tamanho do lado para o triângulo
            break;
        case '2':
            sides = 4; // Define a quantidade de lados
            sideLength = 10; // Define o tamanho do lado para o quadrado
            break;
        case '3':
            sides = 5; // Define a quantidade de lados
            sideLength = 10; // Define o tamanho do lado para o pentágono
            break;
        case '4':
            sides = 6; // Define a quantidade de lados
            sideLength = 10; // Define o tamanho do lado para o hexágono
            break;
        case '5':
            sides = 7; // Define a quantidade de lados
            sideLength = 10; // Define o tamanho do lado para o heptágono
            break;
        case '6':
            sides = 8; // Define a quantidade de lados
```

ANEXO A. CÓDIGO UTILIZADO

```
        sideLength = 10; // Define o tamanho do lado para o octógono
        break;
    default:
        // Opção inválida, não faz nada
        return;
    }
    drawPolygon(sides, sideLength);
}

void drawPolygon(int sides, int sideLength) {
    lcd.clear();
    lcd.print("Desenhando");
    // Código para controlar os motores para desenhar o polígono
    delay(2000);
    for (int i = 0; i < sides; i++) {
        moveForward(sideLength * 100); // Ajuste o tempo conforme necessário
        delay(2000); // Ajuste o tempo conforme necessário
        turn(360/sides); // Rotação do robô
        delay(2000); // Ajuste o tempo conforme necessário
    }
    displayMainMenu();
}

void moveForward(int duration) {
    digitalWrite(motor1Pin1, HIGH);
    digitalWrite(motor1Pin2, LOW);
    digitalWrite(motor2Pin1, HIGH);
    digitalWrite(motor2Pin2, LOW);
    delay(duration);
    stopMotors();
}

void turn(int degrees) {
    // Ajuste o tempo de rotação baseado nos seus motores e configuração
    física
    // Assumindo que 1000ms é o tempo necessário para um giro de 360 graus.
    int turnTime = map(degrees, 0, 360, 0, 1500); // Ajuste conforme
    necessário

    // Motor 1 para frente, Motor 2 para trás
    digitalWrite(motor1Pin1, HIGH);
    digitalWrite(motor1Pin2, LOW);
    digitalWrite(motor2Pin1, LOW);
    digitalWrite(motor2Pin2, HIGH);

    delay(turnTime);
    stopMotors();
}
```

ANEXO A. CÓDIGO UTILIZADO

```
void stopMotors() {
    digitalWrite(motor1Pin1, LOW);
    digitalWrite(motor1Pin2, LOW);
    digitalWrite(motor2Pin1, LOW);
    digitalWrite(motor2Pin2, LOW);
}

void displayResultMenu() {
    lcd.clear();
    lcd.print("1: Area");
    lcd.setCursor(0, 1);
    lcd.print("2: Perimetro");
    delay(2000); // Espera 2 segundos para exibir mais opções
    lcd.clear();
    lcd.print("3: Triangulo?");
    currentState = RESULT_MENU;
}

void handleResultMenu(char key) {
    switch (key) {
        case '1':
        case '2':
            nextAction = key;
            displayInputSides();
            break;
        case '3':
            displayInputTriangleSide1();
            break;
        default:
            // Opção inválida, não faz nada
            break;
    }
}

void displayInputSides() {
    lcd.clear();
    lcd.print("Qtd. de lados:");
    lcd.setCursor(0, 1);
    currentState = INPUT_SIDES;
    userInput = "";
}

void handleInputSides(char key) {
    if (key == '#') {
        if (userInput.length() > 0) {
            sides = userInput.toInt();
            displayInputSideLength();
        } else {
            lcd.clear();
        }
    }
}
```

ANEXO A. CÓDIGO UTILIZADO

```
        lcd.print("Entrada invalida");
        delay(2000);
        displayInputSides();
    }
} else if (isdigit(key)) {
    userInput += key;
    lcd.print(key);
}
}

void displayInputSideLength() {
    lcd.clear();
    lcd.print("Tam. do lado:");
    lcd.setCursor(0, 1);
    currentState = INPUT_SIDE_LENGTH;
    userInput = "";
}

void handleInputSideLength(char key) {
    if (key == '#') {
        if (userInput.length() > 0) {
            sideLength = userInput.toInt();
            if (currentState == INPUT_SIDE_LENGTH && sides > 0) {
                // Chama a função correta baseado na ação desejada
                if (currentState == DRAW_MENU) {
                    drawPolygon(sides, sideLength);
                } else {
                    if (nextAction == '1') {
                        calculateArea(sides, sideLength);
                    } else if (nextAction == '2') {
                        calculatePerimeter(sides, sideLength);
                    }
                }
            }
        }
    } else {
        lcd.clear();
        lcd.print("Entrada invalida");
        delay(2000);
        displayInputSideLength();
    }
} else if (isdigit(key)) {
    userInput += key;
    lcd.print(key);
}
}

void displayInputTriangleSide1() {
    lcd.clear();
    lcd.print("Lado 1:");
}
```

ANEXO A. CÓDIGO UTILIZADO

```
    currentState = INPUT_TRIANGLE_SIDE1;
    userInput = "";
}

void handleInputTriangleSide1(char key) {
    if (key == '#') {
        if (userInput.length() > 0) {
            side1 = userInput.toInt();
            displayInputTriangleSide2();
        } else {
            lcd.clear();
            lcd.print("Entrada invalida");
            delay(2000);
            displayInputTriangleSide1();
        }
    } else if (isdigit(key)) {
        userInput += key;
        lcd.print(key);
    }
}

void displayInputTriangleSide2() {
    lcd.clear();
    lcd.print("Lado 2:");
    currentState = INPUT_TRIANGLE_SIDE2;
    userInput = "";
}

void handleInputTriangleSide2(char key) {
    if (key == '#') {
        if (userInput.length() > 0) {
            side2 = userInput.toInt();
            displayInputTriangleSide3();
        } else {
            lcd.clear();
            lcd.print("Entrada invalida");
            delay(2000);
            displayInputTriangleSide2();
        }
    } else if (isdigit(key)) {
        userInput += key;
        lcd.print(key);
    }
}

void displayInputTriangleSide3() {
    lcd.clear();
    lcd.print("Lado 3:");
    currentState = INPUT_TRIANGLE_SIDE3;
```

ANEXO A. CÓDIGO UTILIZADO

```
    userInput = "";
}

void handleInputTriangleSide3(char key) {
    if (key == '#') {
        if (userInput.length() > 0) {
            side3 = userInput.toInt();
            checkTriangle(side1, side2, side3);
        } else {
            lcd.clear();
            lcd.print("Entrada invalida");
            delay(2000);
            displayInputTriangleSide3();
        }
    } else if (isdigit(key)) {
        userInput += key;
        lcd.print(key);
    }
}

void checkTriangle(int side1, int side2, int side3) {
    lcd.clear();
    if (side1 + side2 > side3 && side1 + side3 > side2 && side2 + side3 >
side1) {
        lcd.print("Forma triangulo");
    } else {
        lcd.print("Nao forma");
    }
    delay(2000);
    displayMainMenu();
}

void calculateArea(int sides, int sideLenght) {
    lcd.clear();
    float area = (sides * sideLenght * sideLenght) / (4 * tan(PI / sides));
    lcd.print("Area: ");
    lcd.print(area);
    delay(2000);
    displayMainMenu();
}

void calculatePerimeter(int sides, int sideLenght) {
    lcd.clear();
    int perimeter = sides * sideLenght;
    lcd.print("Perimetro: ");
    lcd.print(perimeter);
    delay(2000);
    displayMainMenu();
}
```