UNIVERSIDADE FEDERAL DO PAMPA
CURSO DE MATEMÁTICA - LICENCIATURA

TATIANE ZAGO BONORINO

TRANSFORMAÇÕES GEOMÉTRICAS AFINS: UMA ABORDAGEM UTILIZANDO GEOGEBRA E PYTHON

Itaqui
2022

TATIANE ZAGO BONORINO

TRANSFORMAÇÕES GEOMÉTRICAS AFINS: UMA ABORDAGEM UTILIZANDO GEOGEBRA E PYTHON

Trabalho de Conclusão de Curso apresentado ao Curso de Matemática-Licenciatura da Universidade Federal do Pampa, como requisito parcial para a obtenção do Título de Licenciada em Matemática.

Orientadora: Prof[a]. Dr[a]. Daiane Campara Soares

Itaqui
2022

Ficha catalográfica elaborada automaticamente com os dados fornecidos
pelo(a) autor(a) através do Módulo de Biblioteca do
Sistema GURI (Gestão Unificada de Recursos Institucionais) .

B712t Bonorino, Tatiane Zago
Transformações Geométricas Afins: Uma abordagem utilizando Geogebra e Python / Tatiane Zago Bonorino.
91 p.

Trabalho de Conclusão de Curso(Graduação)-- Universidade Federal do Pampa, MATEMÁTICA, 2022.
"Orientação: Daiane Campara Soares".

1. Transformações Geométricas. 2. GeoGebra. 3. Linguagem de Programação. 4. Python. I. Título.

TATIANE ZAGO BONORINO

TRANSFORMAÇÕES GEOMÉTRICAS AFINS: UMA ABORDAGEM UTILIZANDO GEOGEBRA E PYTHON

Trabalho de Conclusão de Curso apresentado ao Curso de Matemática-Licenciatura da Universidade Federal do Pampa, como requisito parcial para a obtenção do Título de Licenciada em Matemática.

Trabalho de Conclusão de Curso defendido e aprovado em: 13 de agosto de 2022.

Banca examinadora:

Daiane Campara Soares

Profª. Drª. Daiane Campara Soares
Orientadora

Elisa Regina Cara

Profª. Drª. Elisa Regina Cara
Universidade Federal do Pampa

Leonel Delatorre

Prof. Dr. Leonel Giacomini Delatorre
Universidade Federal do Pampa

AGRADECIMENTO

Agradeço primeiramente a Deus, pela força e sabedoria que me deu durante todo o curso de graduação, agradeço a Ele também por ter colocado cada pessoa na minha vida, tanto alguns professores quanto colegas.

A minha família por todo apoio e incentivo, caminhando junto comigo nesse momento, me dando conselhos e me ouvindo nos momentos em que eu estava cansada de tanto estudar. Agradeço também ao meu noivo Lucas pelo carinho e amor, nele encontro harmonia e tranquilidade para reestruturar-me cada vez mais.

Agradeço a minha orientadora Daiane que não mediu esforços para realizarmos esse trabalho, virando noites e estudando junto comigo nesse momento. Agradeço a ela e ao professor Leonel por terem me convidado para participar juntamente com a Maria Eduarda de um grupo de pesquisa, a partir desse momento conseguimos realizar grandes trocas que levarei para a vida.

Ao Guilherme que esteve junto com a professora Daiane me ajudando com a linguagem e a programação, dando todo o suporte e apoio para nós duas. Cada dica e conselho fizeram toda a diferença em nosso trabalho, mesmo que fosse domingo tirava todos as minhas dúvidas (que não eram poucas!) com muita sabedoria e paciência.

Agradeço a minha colega Maria Eduarda Berro por toda a amizade e cumplicidade nesses cinco anos de graduação, sendo minha parceira de estudo, trabalhos e projetos da faculdade, cada incentivo fez com que eu crescesse como pessoa e profissional.

E por fim, agradeço ao curso de graduação de Matemática-Licenciatura da Universidade Federal do Pampa (Unipampa) - Campus Itaqui/RS, por todo suporte que a universidade nos proporciona, aos professores e todos os colegas que de algum modo fizeram parte desses momentos comigo.

RESUMO

Este trabalho propõe o estudo de transformações geométricas, em particular as transformações afins, tendo como objetivo geral realizar um estudo das principais definições e resultados relacionados a transformações afins no plano e no espaço. Visando investigar de um ponto algébrico/analítico as transformações geométricas e ainda utilizar de recursos tecnológicos como um meio para visualizá-las. Diante disso, durante a execução do trabalho desenvolvemos uma aplicação – utilizando Python – no intuito de explorar/interpretar as consequências geométricas decorrentes da aplicação das transformações de translação e rotação no plano. Depois de concluída a aplicação foram convidados alguns discentes do curso de Matemática – Licenciatura da UNIPAMPA - Campus Itaqui com alguma experiência/familiaridade com a linguagem de programação. Com os acadêmicos que se mostraram interessados, foi realizado um encontro via Google Meet no qual foram propostas atividades a serem realizadas no Geogebra e também no Google Colab (utilizando Python). Após a conclusão das atividades os participantes foram convidados a responder um formulário elaborado para validar a ferramenta desenvolvida, classificando-a quanto à usabilidade e comparando ambos os recursos utilizados. A análise dos dados coletados seguiu uma abordagem qualitativa e, dentre os resultados apresentados, foi possível observar um interesse na aplicação desenvolvida considerando principalmente que não eram necessárias configurações avançadas na máquina pessoal para executar a aplicação. Além disso, podemos constatar uma maior familiaridade dos participantes com o GeoGebra e a falta de oportunidade em aprimorar os conhecimentos relacionados a algum tipo de linguagem de programação, por exemplo, Python, visto que algoritmos e programação tem sido, em geral, tópicos abordados nos últimos semestres do curso.

Palavras-chave: Transformações Geométricas. GeoGebra. Linguagem de Programação. Python.

ABSTRACT

This work proposes the study of geometric transformations, in particular affine transformations, with the general objective of carrying out a study of the main definitions and results related to affine transformations in the plane and in space. Aiming to investigate from an algebraic/analytical point the geometric transformations and still use technological resources as a means to visualize them. Therefore, during the execution of the work we developed an application – using Python – in order to explore/interpret the geometric consequences arising from the application of translation and rotation transformations in the plane. After completing the application, some students from the Mathematics course - UNIPAMPA - Campus Itaqui with some experience/familiarity with the programming language were invited. With the academics who were interested, a meeting was held via Google Meet in which activities were proposed to be carried out in Geogebra and also in Google Colab (using Python). After completing the activities, the participants were invited to answer a form designed to validate the developed tool, classifying it in terms of usability and comparing both resources used. The analysis of the collected data followed a qualitative approach and, among the results presented, it was possible to observe an interest in the developed application, mainly considering that advanced configurations were not necessary in the personal machine to run the application. In addition, we can see a greater familiarity of the participants with GeoGebra and the lack of opportunity to improve knowledge related to some type of programming language, for example, Python, since algorithms and programming have been, in general, topics covered in the last semesters of the course.

Keywords: Geometric Transformations. GeoGebra. Programming language. Python.

LISTA DE FIGURAS

Figura 1 Reflexão em torno da reta $y = -x$ ... 13
Figura 2 Cisalhamento na direção do eixo x ... 15
Figura 3 Dilatação de um vetor na direção do eixo x $(\alpha > 1)$ ... 16
Figura 4 Reflexão em torno do plano $y = xz$ ... 17
Figura 5 Rotação em torno do eixo x de um ângulo θ_x ... 18

Figura 6 Instruções iniciais no Google Colab ... 26
Figura 7 Exemplo de informação solicitada ao usuário ... 26
Figura 8 Menu do Gráfico - Colab ... 27
Figura 9 Construção do quadrado ... 28
Figura 10 Translação do quadrado ... 28
Figura 11 Construção do triângulo ... 29
Figura 12 Rotação do triângulo definido ... 29
Figura 13 Início da Seção Ponto ... 31
Figura 14 *Feedback* na Seção Ponto ... 31
Figura 15 Ponto escolhido ... 32
Figura 16 Exemplo de Triângulo construído pela autora ... 32
Figura 17 Rotação do triângulo por um ângulo definido pelo usuário ... 33
Figura 18 Quadrado ... 34
Figura 19 Translação do quadrado ... 34

Figura 20 Respostas da Questão 7 ... 36
Figura 21 Respostas da Questão 8 ... 37
Figura 22 Respostas da Questão 17 ... 38
Figura 23 Respostas da Questão 11 ... 38
Figura 24 Respostas da Questão 15 ... 40
Figura 25 Respostas da Questão 16 ... 40
Figura 26 Respostas da Questão 13 ... 41
Figura 27 Visualização do quadrado pelo usuário em tela *Widescreen* (16:9) ... 42
Figura 28 Visualização do quadrado pelo usuário em tela proporcional à escala ... 42
Figura 29 Visualização da rotação do triângulo pelo usuário em tela *Widescreen* ... 43

Figura 30 Pintura Rupestre de El Buey na Bolívia ... 86
Figura 31 Tapete Pazyryk - A mais antiga peça de tapeçaria do mundo ... 86
Figura 32 Cerâmica Chinesa - 3.000 a. C ... 87
Figura 33 Cerâmica Marajoara - Brasil ... 87
Figura 34 Simetrias presentes na Natureza ... 88
Figura 35 Simetrias presentes na Natureza ... 88
Figura 36 Desenho em tecidos ... 89
Figura 37 O homem Vitruviano - Leonardo Da Vinci ... 89

SUMÁRIO

1 INTRODUÇÃO **8**
2 REFERENCIAL TEÓRICO **10**
2.1 Transformações Afins **10**
2.1.1 Transformações Lineares 12
2.1.2 Combinação de Transformações Afins 19
2.2 Recursos Tecnológicos **20**
3 METODOLOGIA **23**
4 APLICAÇÃO **25**
4.1 Formas de Interação do Usuário com a Aplicação desenvolvida no Google Colab **25**
4.2 Sequência de Atividades **27**
5 RESULTADOS E DISCUSSÕES **36**
6 CONSIDERAÇÕES FINAIS **45**
REFERÊNCIAS **48**
APÊNDICE A — APLICAÇÃO NO GOOGLE COLAB **50**
APÊNDICE B — TUTORIAL (COM AS ATIVIDADES) - GEOGEBRA **64**
APÊNDICE C — TUTORIAL - GOOGLE COLAB **74**
APÊNDICE D — QUESTIONÁRIO PROPOSTO AOS ACADÊMICOS **77**
ANEXO A — EXEMPLOS DE TRANSFORMAÇÕES GEOMÉTRICAS AO LONGO DA HISTÓRIA **86**

1 INTRODUÇÃO

Este trabalho propõe o estudo de transformações geométricas, em particular as transformações afins. Podemos pensar em uma transformação geométrica como uma função bijetiva entre duas figuras de modo a alterar o lugar geométrico dos pontos no plano (ou no espaço).

O registro de alguns indícios de transformações geométricas faz parte da história da humanidade, há mais tempo do que se possa imaginar. Destacamos: a pintura rupestre de El Buey na Bolívia; a cerâmica chinesa, que remonta ao período Neolítico (3.000 a.C.), usando as transformações geométricas na sua decoração; a cerâmica marajoara (Brasil) que é considerada uma das cerâmicas mais antigas do continente Americano; o tapete Pazyryk (Sibéria) datado do século V a.C.; aplicações na tecelagem, etc (ver Anexo A). Além disso, pode-se observar que simetrias estão presentes na natureza, como em flores, frutas e nos animais.

Com o passar do tempo, o estudo das transformações geométricas tornou-se mais frequente e, no período do Renascimento, entre os séculos XV e XVI, artistas e arquitetos se interessaram pela representação plana de figuras espaciais a partir do ponto de vista criado pelo próprio olho. Na obra de Leonardo da Vinci (1452-1519), pode-se perceber a relação entre a arte e a matemática (ver Anexo A). Entre o século XVI e XVII, depois dos gregos, surgiram outras contribuições em relação aos rumos da Geometria, sendo, por exemplo, creditado a René Descartes (1596-1650) e Pierre de Fermat (1601-1665) a origem da geometria analítica (MABUCHI, 2000).

Atualmente, com a utilização de Recursos Tecnológicos Educacionais para o ensino de matemática, a investigação de fenômenos geométricos pode ser realizada a partir de ambientes de aprendizagem de geometria. Nesses ambientes os objetos não permanecem de forma estática, ou seja, o usuário consegue interagir com as construções geométricas, realizando movimentos como translação, rotação, modificação de tamanho, além de outras possibilidades.

Na Geometria Analítica, as retas e as superfícies que estudamos são descritas por meio de equações. Isso possibilita um olhar do ponto de vista algébrico para muitas questões geométricas e, da mesma forma, pode-se interpretar geometricamente determinados contextos algébricos. Essa interconexão entre Geometria e Álgebra foi responsável por consideráveis avanços na Matemática e suas aplicações (LIMA, 2013).

Em sua análise sobre a Base Nacional Comum Curricular (BNCC), Souza et al.

(2020) afirmam que a Matemática, em particular no Ensino Fundamental, deve articular as unidades temáticas – dentre elas a Geometria e a Álgebra – de modo a estimular nos estudantes a habilidade de identificar situações-problemas e resolvê-las, aplicando conceitos e métodos estudados em sala de aula.

Outro aspecto bastante enfatizado na BNCC é o pensamento computacional. O uso de algoritmos possibilita que os estudantes desenvolvam a habilidade de escrever uma sequência finita de ações executáveis no intuito de resolver algum problema. A linguagem utilizada para a construção de algoritmos se assemelha à linguagem algébrica, sobretudo no que se refere à noção de variável e ao encontro de determinados padrões através dos quais, algumas vezes, é possível desenvolver conjecturas (SOUZA et al., 2020).

Motivados pelo exposto acima, nosso trabalho propõe a seguinte temática de pesquisa: *Explorar – algébrica e geometricamente – conceitos relacionados à transformações afins por meio de recursos tecnológicos*.

A fim de buscar respostas para essa problemática, este trabalho estabelece como objetivo geral: realizar um estudo das principais definições e resultados relacionados à transformações afins no plano e no espaço, visando investigar de um ponto de vista algébrico/analítico as transformações geométricas e ainda utilizar de recursos tecnológicos como um meio para visualizá-las.

Este trabalho de conclusão de curso se justifica ao reforçar importantes aspectos da BNCC e ao estabelecer conexões entre os conteúdos trabalhados em vários componentes curriculares ao longo da formação da licencianda em Matemática, principalmente, nos componentes relacionados à Geometria e à Álgebra Linear, bem como no componente curricular de Algoritmos e Programação.

O presente trabalho de conclusão de curso (TCC) apresenta-se da seguinte forma a partir desta introdução: o segundo capítulo intitulado Referencial Teórico descrevemos – em suas seções e subseções – os conceitos relacionados à Transformações Afins e as ferramentas que serão utilizadas, logo após, o terceiro capítulo apresenta a metodologa. No quarto capítulo temos as aplicações. No quinto capítulo temos os resultados e breves discussões, apresentados com base nos dados coletados. No sexto capítulo apresentamos as considerações finais. A seguir são apresentadas as referências bibliográficas, os Apêndices A, B, C e D e, por fim, o Anexo A.

2 REFERENCIAL TEÓRICO

Neste capítulo destacamos os principais conceitos de transformações afins tanto no plano quanto no espaço. Além disso, faremos uma breve explanação sobre os recursos educacionais tecnológicos escolhidos para visualização e aplicação dos conceitos estudados.

2.1 Transformações Afins

Nesta seção apresentamos a definição e alguns exemplos de Transformações Afins. Para isso, utilizamos essencialmente as referências: Battaiola (2003) e Lima (2013).

Definição 2.1.1. *Dado um vetor $v' \in \mathbb{R}^n$, definimos uma transformação afim como toda transformação de coordenadas de $\mathbb{R}^n$ em $\mathbb{R}^n$ do tipo*

$$\mathbf{v}' = A\,\mathbf{v} + \mathbf{b} \tag{1}$$

onde A é uma matriz $n \times n$ com entradas reais, cujo determinante é não-nulo, e B é um vetor coluna fixo.

Se $n = 1$, obtemos $A = a \neq 0$ e a transformação afim $v' = av + b$ é a função afim. Observe que se $n = 2$, então

$$\mathbf{v} = \begin{bmatrix} x \\ y \end{bmatrix},\ A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} (a_{11}a_{22} - a_{12}a_{21} \neq 0) \text{ e } \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

de modo que:

$$A\mathbf{v} + \mathbf{b} = \begin{bmatrix} a_{11}x + a_{12}y + b_1 \\ a_{21}x + a_{22}y + b_2 \end{bmatrix}$$

Ou ainda, se $n = 3$, podemos escrever:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Neste caso, as coordenadas (x', y', z') do vetor $\mathbf{v}'$, que definem um ponto no espaço, estão colocadas em função das coordenadas (x,y,z) e das entradas a_{ij} e b_i $(i,j=1,2,3)$, que são constantes reais determinadas pelo tipo de transformação (lembrando que $\det A \neq 0$). As transformações afins têm a função de modificar a posição dos pontos ou dos objetos no espaço. Essas transformações tem a característica geral de transformar linhas paralelas em linhas paralelas e mapear pontos finitos em pontos finitos. O grupo de transformações afins do espaço define a geometria afim, que estuda as razões e proporções entre objetos geométricos.

Exemplo 2.1.1. *Uma transformação afim muito importante na computação gráfica é a translação que consiste em adicionar um vetor* $\mathbf{b} = (\alpha, \beta)$ *não-nulo ao vetor* (x,y)*, isto é,* $T : \mathbb{R}^2 \to \mathbb{R}^2$ *é dada por*

$$T\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \alpha \\ \beta \end{bmatrix}.$$

Cabe observar que a transformação translação é uma transformação afim mas não é uma transformação linear pois não transforma o vetor nulo nele mesmo. E, não sendo linear, esta transformação não pode ser escrita como produto de uma matriz (que estaria associada à transformação) pelo vetor (x,y).

Exemplo 2.1.2. *A transformação de translação no espaço é uma função* $T : \mathbb{R}^3 \to \mathbb{R}^3$*, tal que*

$$T\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix},$$

onde

$$\mathbf{b} = \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix}$$

é um vetor não-nulo.

Observe que se tomarmos $\mathbf{b} = \mathbf{0}$ em (1) temos, como um caso particular das transformações afins, as transformações lineares.

2.1.1 Transformações Lineares

Nesta subseção, apresentamos definições, resultados e exemplos da Álgebra Linear. Tais conceitos foram extraídos de literatura pertinente ao tema, dos quais destacamos: Boldrini et al.(1980), Lay (1999), Silva (2013) e Steinbruch e Winterle (1987).

Primeiramente, vamos conceituar transformações lineares e observar que, de modo geral, podemos associá-las à matrizes. Logo após, apresentaremos uma visão geométrica das transformações lineares no plano, dando alguns exemplos para ilustrar que certas "deformações" podem ser descritas por transformações lineares.

Definição 2.1.2. *Sejam X e Y dois espaços vetoriais. Uma função $T : X \to Y$ é chamada transformação linear de X em Y se satisfaz as seguintes condições:*

(i) Para quaisquer $\mathbf{u}, \mathbf{v} \in X$, temos $T(\mathbf{u}+\mathbf{v}) = T(\mathbf{u}) + T(\mathbf{v})$.

(ii) Para quaisquer $k \in \mathbb{R}$ e $\mathbf{v} \in X$, temos $T(k\mathbf{v}) = kT(\mathbf{v})$.

A Definição 2.1.2 afirma que se $T : X \to Y$ é linear ela preserva as duas operações básicas de um espaço vetorial, isto é, a adição de vetores e a multiplicação por escalar.

Conforme descreve Lay (1999), toda transformação linear do $\mathbb{R}^n$ para o $\mathbb{R}^m$ pode ser vista, na verdade, como uma transformação matricial $x \to Ax$ em que propriedades importantes da transformação T estão relacionadas a propriedades conhecidas de A. Para se determinar A é importante observar que T fica completamente determinada pela sua ação nas colunas da matriz.

Entenderemos por transformações lineares planas as transformações de $\mathbb{R}^2$ para $\mathbb{R}^2$. Vejamos alguns exemplos:

Exemplo 2.1.3. *A transformação de escala é uma função $T : \mathbb{R}^2 \to \mathbb{R}^2$ dada por $T(x,y) = (s_1x, s_2y)$ onde a abscissa x é multiplicada por um fator s_1, a ordenada y por um fator s_2, isto é,*

$$T \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_1x \\ s_2y \end{bmatrix} = \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

Na transformação de escala os escalares s_1 e s_2 são números reais positivos. Quando eles variam no intervalo $(0,1)$ temos uma redução da dimensão correspondente e se s_1 e s_2 são maiores que 1 temos um aumento. Se s_1 e s_2 fossem negativos, os pontos do plano seriam espelhados em torno de um eixo correspondente (para mais detalhes desse último caso, veja o próximo exemplo).

Exemplo 2.1.4. *Geometricamente a transformação linear de espelhamento ou reflexão realiza movimentos no plano que preservam as distâncias entre dois pontos, isto é, estas transformações mudam a posição dos objetos mantendo a sua forma e o seu tamanho originais, dando origem a figuras congruentes. Por isso estas transformações (bem como as translações e rotações), são também chamadas de isometrias (do grego: mesmo tamanho) ou movimentos rígidos. Algebricamente essa transformação pode ser descrita como uma função* $T : \mathbb{R}^2 \to \mathbb{R}^2$ *e, definindo a matriz:*

$$M = \begin{bmatrix} \lambda_1 & \lambda_2 \\ \lambda_3 & \lambda_4 \end{bmatrix},$$

temos que:

• *Se* $\lambda_1 = 1,\ \lambda_4 = -1,\ \lambda_2 = \lambda_3 = 0,$ *então a matriz* M *está associada à uma reflexão em torno do eixo* x.

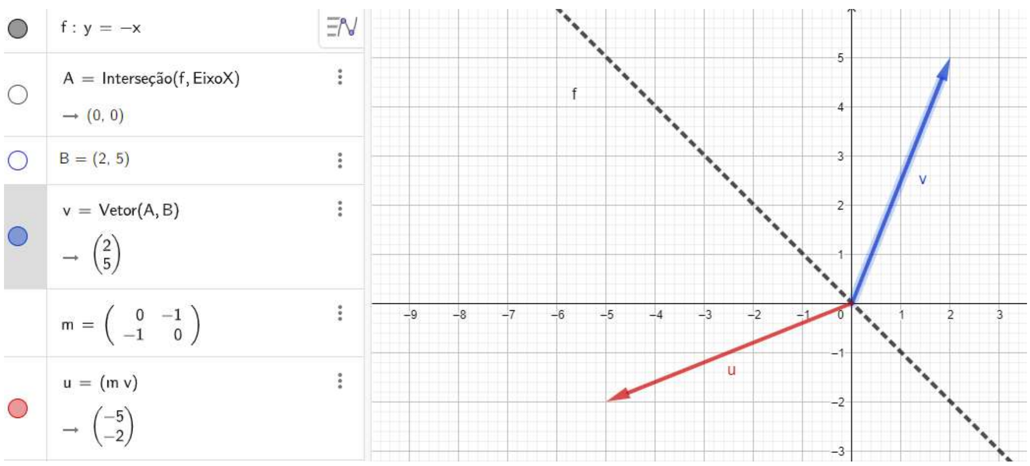
• *Se* $\lambda_1 = -1,\ \lambda_4 = 1,\ \lambda_2 = \lambda_3 = 0,$ *então a matriz* M *está associada à uma reflexão em torno do eixo* y.

• *Se* $\lambda_1 = \lambda_4 = -1,\ \lambda_2 = \lambda_3 = 0,$ *então a matriz* M *está associada à uma reflexão em torno da origem.*

• *Se* $\lambda_1 = \lambda_4 = 0,\ \lambda_2 = \lambda_3 = 1,$ *então a matriz* M *está associada à uma reflexão em torno da reta* $y = x$.

• *Se* $\lambda_1 = \lambda_4 = 0,\ \lambda_2 = \lambda_3 = -1,$ *então a matriz* M *está associada à uma reflexão em torno da reta* $y = -x$.

Figura 1 – Reflexão em torno da reta $y = -x$



Fonte: Autora.

Exemplo 2.1.5. *A transformação de rotação no plano permite rotacionar um vetor em torno da origem de um ângulo θ (no sentindo anti-horário) através da multiplicação da matriz de transformação de rotação pela coordenada do vetor que queremos rotacionar. Algebricamente é uma função $T : \mathbb{R}^2 \to \mathbb{R}^2$ dada por*

$$T\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x\cos(\theta) - y\,sen(\theta) \\ y\cos(\theta) + x\,sen(\theta) \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -sen(\theta) \\ sen(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

Exemplo 2.1.6. *A transformação linear de cisalhamento é uma função $T : \mathbb{R}^2 \to \mathbb{R}^2$ que consiste em preservar uma coordenada e mover a outra, porém, esse movimento depende do valor da coordenada que não foi alterada. Além disso, temos a transformação de cisalhamento em relação ao eixo x e em relação ao eixo y, como apresentamos a seguir:*

(a) ***Cisalhamento na direção do eixo*** **x***:* *A transformação é dada por*

$$T\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + \alpha y \\ y \end{bmatrix} = \begin{bmatrix} 1 & \alpha \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

Esse cisalhamento é também chamado cisalhamento horizontal de fator α.

(b) ***Cisalhamento na direção do eixo*** **y***:* *A transformação é dada por*

$$T\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ y + \alpha x \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

Esse cisalhamento é também chamado cisalhamento vertical de fator α.

Exemplo 2.1.7. *Nosso último exemplo de transformações lineares no plano apresenta as transformações de dilatação e contração. Vejamos alguns casos:*

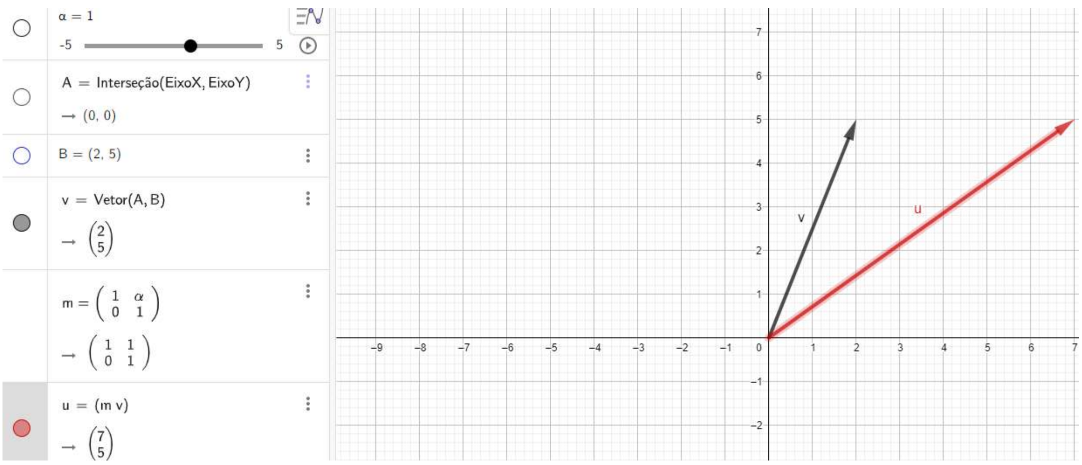
(a) ***Dilatação ou contração na direção do vetor:*** *A transformação é dada por*

$$T\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \alpha x \\ \alpha y \end{bmatrix} = \begin{bmatrix} \alpha & 0 \\ 0 & \alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

Observamos que:

- *Se $|\alpha| > 1$, T dilata o vetor;*
- *Se $|\alpha| < 1$, T contrai o vetor;*
- *Se $\alpha = 1$, T é a identidade I;*
- *Se $\alpha < 0$, T troca o sentido do vetor.*

Figura 2 – Cisalhamento na direção do eixo *x*



Fonte: Autora.

(b) ***Dilatação ou contração na direção do eixo x:*** *A transformação é dada por*

$$T\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \alpha x \\ y \end{bmatrix} = \begin{bmatrix} \alpha & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}.$$

(c) ***Dilatação ou contração na direção do eixo y:*** *A transformação é dada por*

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ \alpha y \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \alpha \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}.$$

Em ambos os itens (b) e (c) consideramos $\alpha > 0$ e observamos que:

• *Se $\alpha > 1$, T dilata o vetor;*

• *Se $0 < \alpha < 1$, T contrai o vetor.*

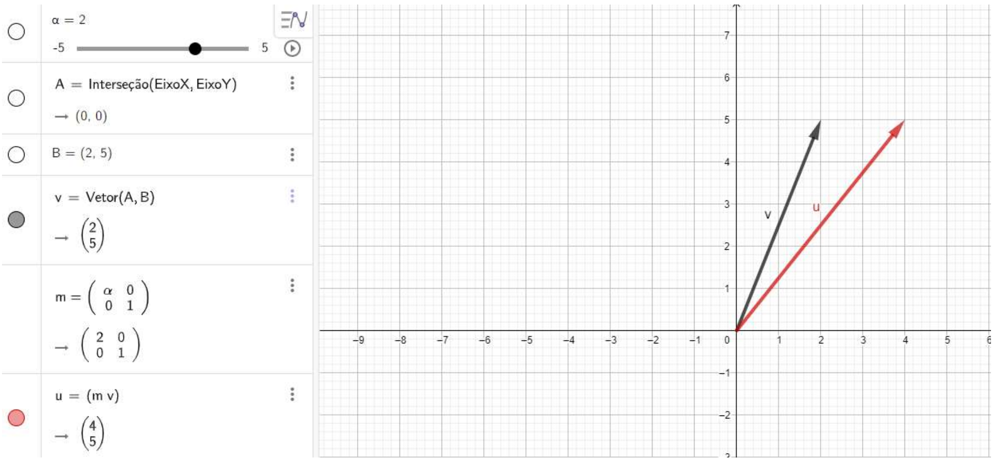
Se, no item (b) fizéssemos $\alpha = 0$ teríamos que T seria a projeção ortogonal no plano sobre o eixo x. Da mesma forma, se $\alpha = 0$ no item (c), teríamos T a projeção ortogonal do plano sobre o eixo y.

A fim de facilitar a visualização das transformações de Reflexão, Translação e Rotação no plano, sugerimos o seguinte objeto de aprendizagem: <https://www.geogebra.org/m/Xb8u2DQb>.

Entendemos por transformações lineares no espaço as transformações de $\mathbb{R}^3$ em $\mathbb{R}^3$. Vejamos alguns exemplos:

Exemplo 2.1.8. *A transformação de escala **T** no espaço é uma função $T : \mathbb{R}^3 \to \mathbb{R}^3$, tal*

Figura 3 – Dilatação de um vetor na direção do eixo x $(\alpha > 1)$



Fonte: Autora.

que

$$T\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \alpha_1 x \\ \alpha_2 y \\ \alpha_3 z \end{bmatrix} = \begin{bmatrix} \alpha_1 & 0 & 0 \\ 0 & \alpha_2 & 0 \\ 0 & 0 & \alpha_3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix},$$

com $\alpha_1,\ \alpha_2,\ \alpha_3 > 0$*.*

Exemplo 2.1.9. *A transformação de reflexão no espaço pode ser descrita como uma função* $T : \mathbb{R}^3 \to \mathbb{R}^3$ *e, definindo a matriz:*

$$M = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix},$$

temos que:

• *Se* $\lambda_1 = \lambda_2 = 1$ *e* $\lambda_3 = -1$*, então a matriz* M *está associada à uma reflexão em relação ao plano* xy*.*

• *Se* $\lambda_1 = \lambda_3 = 1$ *e* $\lambda_2 = -1$*, então a matriz* M *está associada à uma reflexão em relação ao plano* xz*.*

• *Se* $\lambda_2 = \lambda_3 = 1$ *e* $\lambda_1 = -1$*, então a matriz* M *está associada à uma reflexão em relação ao plano* yz*.*

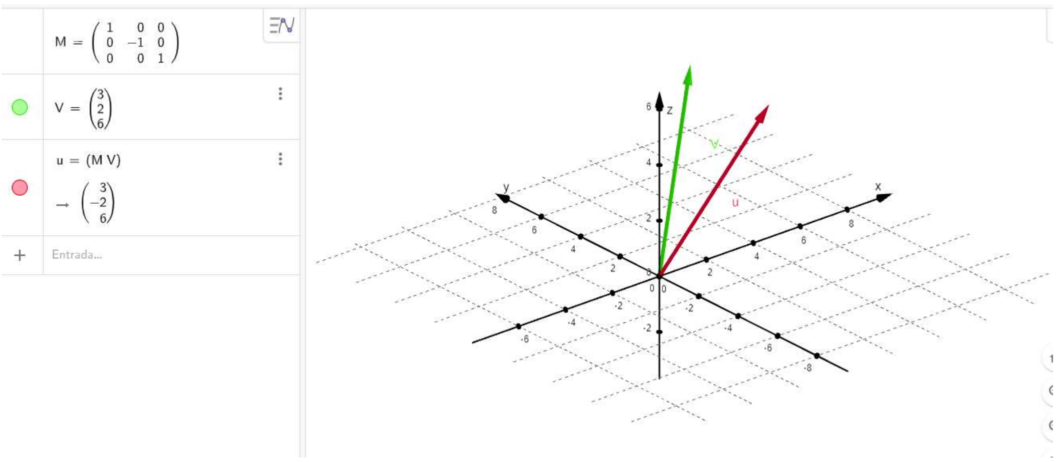
• *Se* $\lambda_1 = 1$ *e* $\lambda_2 = \lambda_3 = -1$*, então a matriz* M *está associada à uma reflexão em relação ao eixo* x*.*

• *Se* $\lambda_2 = 1$ *e* $\lambda_1 = \lambda_3 = -1$*, então a matriz* M *está associada à uma reflexão em relação* *ao eixo y.*

• *Se* $\lambda_3 = 1$ *e* $\lambda_1 = \lambda_2 = -1$, *então a matriz M está associada à uma reflexão em relação ao eixo z.*

• *Se* $\lambda_1 = \lambda_2 = \lambda_3 = -1$, *então a matriz M está associada à uma reflexão em relação a origem.*

Figura 4 – Reflexão em torno do plano $y = xz$



Fonte: Autora.

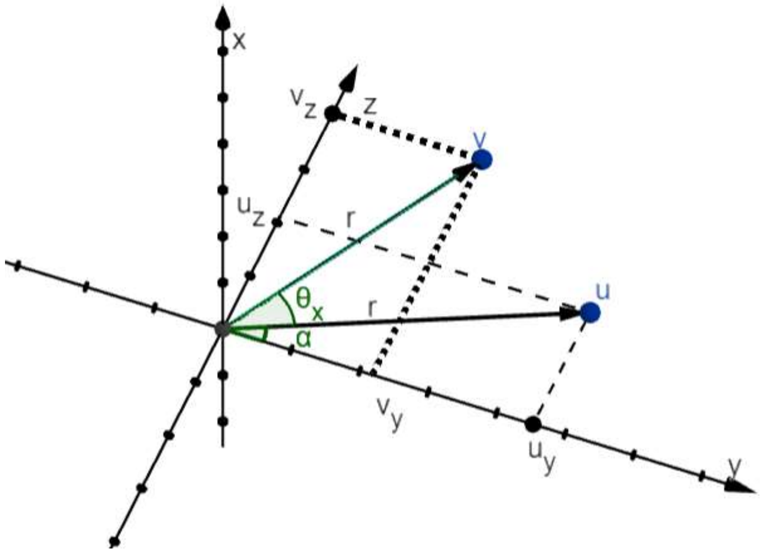
As matrizes de Transformação de Rotação no Espaço não são tão simples de definir como no Plano. Podemos simplificar o processo considerando a rotação de um objeto no espaço a partir de três rotações em torno de cada um dos eixos cartesianos e observando a ordem das rotações, pois se alterarmos a ordem em que elas são aplicadas obtemos resultados diferentes.

Exemplo 2.1.10. *Vamos considerar a seguinte ordem de rotação: primeiro em torno do eixo x de um ângulo* θ_x, *segundo em torno do eixo y de um ângulo* θ_y *e, por último, em torno do eixo z de um ângulo* θ_z. *Mostremos que a matriz de uma transformação de rotação no espaço em relação ao eixo x é:*

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -sen(\theta_x) \\ 0 & sen(\theta_x) & \cos(\theta_x) \end{bmatrix}.$$

De fato, seja o vetor $\mathbf{u} = (u_x, u_y, u_z) \in \mathbb{R}^3$ *tal que* $||\mathbf{u}|| = r$, *vamos rotacionar* $\mathbf{u}$ *em torno do eixo x de um ângulo* θ_x, *obtendo assim um vetor* $\mathbf{v} = (v_x, v_y, v_z)$. *Assim, de acordo com a figura 5*

Figura 5 – Rotação em torno do eixo x de um ângulo θ_x



Fonte: Autora. Disponível em: <https://www.geogebra.org/m/xzhkry8e>

temos que: $u_y = r\cos(\alpha)$*,* $u_z = r\,sen(\alpha)$*,* $v_y = r\cos(\alpha+\theta_x)$ *e* $v_z = r\,sen(\alpha+\theta_x)$*. Utilizando as expressões para o seno e o cosseno da soma de dois vetores, temos que*

$$\begin{aligned} v_y &= r\cos(\alpha+\theta_x) \\ &= r\left[\cos(\alpha)\cos(\theta_x) - sen(\alpha)sen(\theta_x)\right] \\ &= r\cos(\alpha)\cos(\theta_x) - rsen(\alpha)sen(\theta_x) \\ &= u_y\cos(\theta_x) - u_z\,sen(\theta_x) \end{aligned}$$

e

$$\begin{aligned} v_z &= r\,sen(\alpha+\theta_x) \\ &= r\left[sen(\alpha)\cos(\theta_x) + sen(\theta_x)\cos(\alpha)\right] \\ &= r\,sen(\alpha)\cos(\theta_x) + rsen(\theta_x)\cos(\alpha) \\ &= u_z\cos(\theta_x) + u_y\,sen(\theta_x), \end{aligned}$$

logo,

$$\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} u_x \\ u_y\cos(\theta_x) - u_z\,sen(\theta_x) \\ u_y\,sen(\theta_x) + u_z\cos(\theta_x) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -sen(\theta_x) \\ 0 & sen(\theta_x) & \cos(\theta_x) \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix},$$

portanto, a matriz de rotação no espaço em relação ao eixo x de um ângulo θ_x *(no sentido* *anti-horário) é:*

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -sen(\theta_x) \\ 0 & sen(\theta_x) & \cos(\theta_x) \end{bmatrix},$$

como queríamos.

Da mesma forma que mostramos acima, podemos concluir que:

- *A matriz de rotação no espaço em relação ao eixo y de um ângulo θ_y é*

$$T = \begin{bmatrix} \cos(\theta_y) & 0 & sen(\theta_y) \\ 0 & 1 & 0 \\ -sen(\theta_y) & 0 & \cos(\theta_y) \end{bmatrix}.$$

- *A matriz de rotação no espaço em relação ao eixo z de um ângulo θ_z é*

$$T = \begin{bmatrix} \cos(\theta_z) & -sen(\theta_z) & 0 \\ sen(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

2.1.2 Combinação de Transformações Afins

Para exemplificar como podemos combinar transformações, vamos analisar um exemplo em particular: Suponha que queiramos rotacionar um objeto (como uma figura geométrica por exemplo) em torno de um ponto arbitrário $P(x,y)$. Se conhecermos apenas a rotação em torno da origem e a translação (apresentadas anteriormente), podemos converter o problema original (a princípio desconhecido) em três problemas diferentes (conhecidos). Assim, para rotacionar um objeto em torno de P podemos sequencialmente:

- Transladar P até a origem;
- Rotacionar o objeto em relação à origem e, por fim,
- Transladar da origem até P.

Se estivermos um objeto no plano, na primeira translação $\mathbf{b} = (-x,-y)$ e a última translação é o inverso $\mathbf{b} = (x,y)$. Podemos nos perguntar se a composição dessas transformações continua sendo uma transformação afim e, para responder essa questão, apresentamos o seguinte resultado:

Teorema 2.1.1. *A composição de duas transformações afins é uma transformação afim.*

Prova. Sejam T e S duas transformações afins de $\mathbb{R}^n$ em $\mathbb{R}^n$ tais que $T(v') = Av + b$ e $S(v') = Cv + d$. Então,

$$T \circ S(v') = T(S(v')) = AS(v') + b = A(Cv + d) + b = ACv + (Ad + b)$$

E, como $\det(AC) = \det(A) \cdot \det(C) \neq 0$, temos que $T \circ S$ é uma transformação afim. □

2.2 Recursos Tecnológicos

Os recursos tecnológicos escolhidos para visualização e aplicação de algumas das transformações afins apresentadas – sob um ponto de vista algébrico e também geométrico – foram o *software* Geogebra e a linguagem de programação Python.

O GeoGebra foi criado em 2001 pelo pesquisador Markus Hohenwarter. Uma das principais diferenças entre ele e outros *softwares* de geometria dinâmica é que ele possui licença livre e é escrito em linguagem JAVA, possibilitando sua disponibilidade em várias plataformas (SILVA, 2012).

Segundo Silva (2012), a principal característica do Geogebra é a combinação de geometria, álgebra e cálculo em um único ambiente. Além disso, conseguimos observar no programa que a visualização de uma figura possui um correspondente geométrico e outro algébrico.

Destacamos ainda que as principais razões da escolha desse *software* para este trabalho são:

- o fato de ele ser gratuito, fazendo com que qualquer pessoa possa baixá-lo em seu tablet, computador ou celular, ou ainda, rodar o programa diretamente na Web através do endereço: <https://www.geogebra.org/classic?lang=pt>;
- a grande comunidade adepta a ele que cresce mais a cada dia, tornando-se uma grande rede de troca de informações e conhecimentos;
- as diversas ferramentas presentes na aplicação e suas possibilidades;
- a existência de um contato prévio com o Geogebra na elaboração de atividades a serem desenvolvidas em sala de aula, por meio de projetos ou mesmo dos componentes curriculares de estágios. Outros componentes curriculares como Informática na Educação Matemática e alguns dos Laboratórios de Ensino de Matemática

também apresentaram e utilizaram essa ferramenta a fim de facilitar a visualização geométrica.

Quanto ao Python, o mesmo foi desenvolvido em 1989 pelo programador holandês Guido Van Rossum. Lançado em 1991, é uma das linguagens mais conhecidas atualmente, além de ser livre e de código aberto (SILVA, 2019).

> O Python possibilita ao usuário criar programas, sejam jogos ou códigos científicos para cálculos avançados, isso para vários ambientes, desde desktop, web ou até mesmo móvel, além de que, sua sintaxe é uma das suas mais simples dentre todas as linguagens de programação. E tem uma ampla biblioteca de suporte para programar junto com uma comunidade enorme, que pode auxiliar ao usuário vários materiais em fóruns, redes sociais etc. Sem falar que atualmente, o Python possui grande mercado, contando com grandes empresas usuárias de seus serviços (SILVA, 2019, p.31).

Além do que já foi mencionado acima, outras justificativas da escolha do Python como linguagem de programação se devem ao fato de:

- ter sido a linguagem adotada no componente curricular de Algoritmos e Programação cursado pela autora deste TCC e presente nas componentes obrigatórias do curso de Matemática - Licenciatura da UNIPAMPA – Campus Itaqui;
- possuir diversas bibliotecas que oferecem recursos para a realização das principais tarefas comuns no desenvolvimento de *software* e, além disso, visto que possui código aberto, tais bibliotecas podem ser estendidas através da contribuição de terceiros;
- o conjunto de regras que definem como um programa em Python será escrito e interpretado (sintaxe) ser simples e geral, o que possibilita que com uma pequena quantidade de código implementado seja possível desenvolver diversas aplicações.

Cabe ainda salientar que para podermos implementar qualquer algoritmo utilizando Python é necessário entender as funcionalidades e potencialidades da linguagem, bem como explorar algumas bibliotecas que possam ser utilizadas a fim de atender nossos objetivos. Para isso é necessário um estudo, seguido de diversas tentativas para resolver cada problema que surge na busca por um código que contemple minimamente o que desejamos.

Considerando tudo que acabamos de mencionar e também observando que um dos objetivos específicos deste trabalho é a utilização de recursos tecnológicos para mobilizar conhecimentos sobre transformações afins, justificamos o longo período utilizado para testar e comparar implementações em algumas bibliotecas do Python. A fim de verificar se essas atendiam (pelo menos parcialmente) as necessidades impostas pelo ambiente e também esperadas na construção das atividades. De forma mais específica, analisamos o comportamento da biblioteca dentro do ambiente ao renderizar gráficos e animações.

Para utilizar uma linguagem de programação, de modo geral, precisamos de interpretadores/compiladores, um ambiente para que o código seja digitado e algum mecanismo para executá-lo. Muitas vezes essas ferramentas exigem instalação e, em alguns casos particulares como, por exemplo, o Python, são necessários requisitos mínimos na máquina para que se possa utilizar a linguagem, quase sempre precisando de um conhecimento médio e/ou avançado sobre computação.

A fim de contornar as (possíveis) dificuldades apresentadas e também buscando propor atividades que possam ser desenvolvidas por qualquer pessoa em qualquer máquina, utilizamos como alternativa um ambiente que permite escrever e executar Python no navegador sem a necessidade de configurações. Para isso, escolhemos o Google Colaboratory.

O ambiente Google Colaboratory ou Colab é um serviço de nuvem gratuito hospedado pelo Google para incentivar a pesquisa de aprendizado em máquina e inteligência artificial. Por meio dos recursos disponibilizados, ele facilita o processo de programação e torna mais prático para o usuário aprender (SILVA, 2019).

Além disso, destacamos que o Colab possui acesso gratuito a unidades de processamento gráfico e é de compartilhamento fácil. Por fim, podemos mencionar que outra motivação para escolha deste ambiente é que o mesmo foi utilizado no componente curricular de Algoritmos e Programação cursado pela autora, aumentando assim a familiaridade com a ferramenta.

3 METODOLOGIA

A presente pesquisa organiza-se em torno de conceitos e propriedades relacionados às transformações afins. Com a finalidade de contemplarmos a temática proposta, foi necessário um estudo aprofundado sobre definições e resultados básicos de Álgebra Linear e Geometria, evidenciando assim uma forte conexão entre aspectos algébricos e geométricos. Neste trabalho, desenvolvemos uma aplicação – utilizando Python – com objetivo de explorar/interpretar as consequências geométricas decorrentes da aplicação das transformações exemplificadas no capítulo 2. Além disso, também utilizamos o Geogebra para construção e visualização de exemplos.

Depois de concluída a aplicação, foi elaborado pela autora uma sequência de atividades no GeoGebra e também no Google Colab e, após diversos testes, foram convidados alguns discentes do curso de Matemática – Licenciatura da UNIPAMPA - Campus Itaqui com alguma experiência/familiaridade com a linguagem de programação para participarem de um encontro. A finalidade da reunião era propor aos acadêmicos as atividades previamente desenvolvidas e após, solicitar que respondessem a um formulário, validando a aplicação (classificando-a quanto à usabilidade) e comparando as atividades desenvolvidas nos recursos utilizados, explorando, eventualmente, vantagens e/ou desvantagens das ferramentas.

A fim de validar a aplicação utilizamos a Escala Likert[1] que não se utiliza de questões nas quais as pessoas escolhem apenas entre sim e não mas, em geral, é constituída de questões com afirmações autodescritivas que tem por opções de respostas uma escala que varia de "concordo totalmente" até "discordo totalmente".

Nesse sentido, em relação aos procedimentos técnicos, podemos identificar (inicialmente) nessa pesquisa o caráter bibliográfico, uma vez que foram realizados estudos bibliográficos, em obras científicas adequadas ao tema – livros, artigos, revistas científicas, anais de eventos científicos, dissertações e teses disponíveis em acervo convencional ou virtual – de forma a compreender conceitos, resultados e propriedades relacionadas às transformações geométricas, transformações afins ou, mais particularmente ainda, transformações lineares. Ademais, também foram necessárias leituras a respeito dos recursos tecnológicos utilizados. A partir disso, quando investigamos relações entre entes geométricos por meio dos diferentes tipos de transformações afins, entendemos que o trabalho

[1]Desenvolvida nos Estados Unidos na década de 30 é a escala mais utilizada em pesquisas de opinião pois os participantes de pesquisas que utilizam essa escala especificam seu nível de concordância com uma afirmação.

adquiriu características associadas à uma pesquisa experimental, visto que Gil (2019, p. 30) nos diz que "[...] a pesquisa experimental consiste em determinar um objeto de estudo, selecionar as variáveis que seriam capazes de influenciá-lo, definir as formas de controle e de observação dos efeitos que a variável produz no objeto".

Em relação aos objetivos apresentados, entendemos que este trabalho se caracterizou como uma pesquisa exploratória, visto que, conforme GIL (2019, p. 26): "Estas pesquisas têm como objetivo proporcionar maior familiaridade com o problema, com vistas a torná-lo mais explícito ou a constituir hipóteses. Pode-se dizer que estas pesquisas têm como objetivo principal o aprimoramento de ideias ou a descoberta de intuições".

4 APLICAÇÃO

Depois de finalizada a construção de um roteiro de atividades relacionadas à temática deste TCC, utilizando a linguagem de programação Python e o Geogebra, no dia 16 de junho de 2022, realizou-se um encontro – em sala virtual do Google Meet – a fim de propor que os discentes previamente convidados desenvolvessem tais atividades. O encontro teve duração de duas horas e contou com a participação de sete discentes, todos com os conhecimentos prévios de Algoritmos e Programação por já terem cursado o componente curricular vinculado ao curso de Matemática.

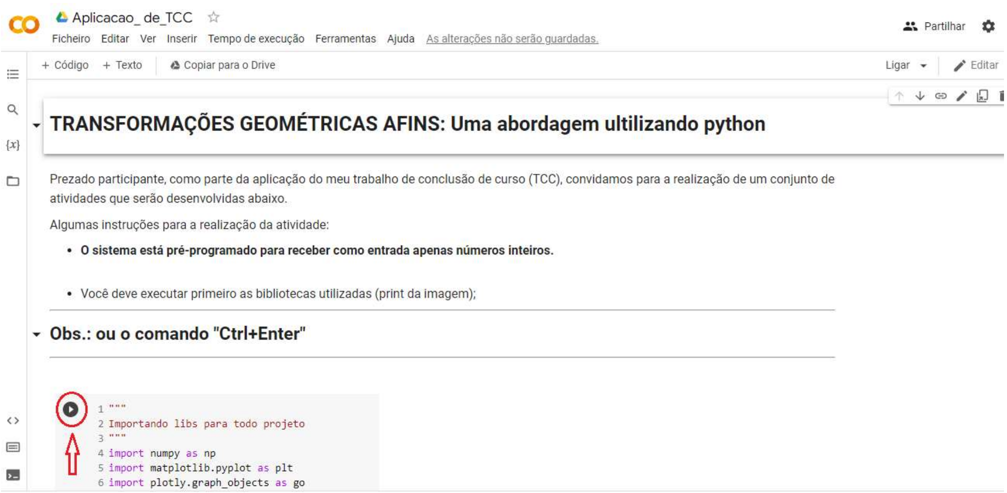
No início da reunião foi apresentado um breve resumo sobre a pesquisa e seus objetivos. A seguir, cada discente foi instruído quanto a sua participação, foi esclarecido que eles deveriam consentir a apresentação parcial/total dos dados coletados neste trabalho (ver primeira seção do questionário no Apêndice D), reforçando que a identidade de cada participante seria preservada e que, caso não houvesse interesse em participar da pesquisa, a qualquer momento, seria possível enviar uma mensagem no *chat* e sair da sala virtual.

Posteriormente foi apresentado aos acadêmicos a sequência das atividades que seriam desenvolvidas por eles, indicando que fossem realizadas primeiro no Geogebra e, só após, no Google Colaboratory (Colab). Destacamos que, ao concluir a atividade, foi solicitado que os(as) participantes encaminhassem um link ou *print* como registro. Além disso, também foi explicado que o instrumento de avaliação/validação do roteiro proposto seria um questionário elaborado pela autora (ver Apêndice D), a ser respondido ao término das atividades.

4.1 Formas de Interação do Usuário com a Aplicação desenvolvida no Google Colab

Inicialmente, o usuário deveria executar a caixa de código destacada na Figura 6. Tal ação é necessária pois, se não fossem pré-carregadas as bibliotecas que seriam utilizadas no decorrer da aplicação, o sistema não iria executar corretamente as instruções presentes nos próximos blocos de código.

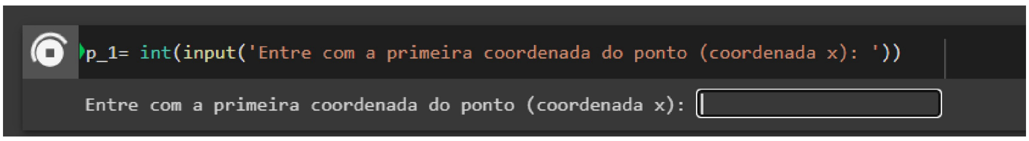
Figura 6 – Instruções iniciais no Google Colab



Fonte: Autora.

Algumas instruções iriam solicitar que o usuário fornecesse dados para o sistema que devem ser inseridos dentro de caixas de texto. Por exemplo, “Entre com a primeira coordenada do ponto (coordenada *x*)”.

Figura 7 – Exemplo de informação solicitada ao usuário



Fonte: Autora.

Após carregar a caixa de código e inserir a informação solicitada, o usuário precisaria teclar *Enter* para que o dado fosse armazenado. A seguir, os dados seriam processados e o sistema forneceria ao usuário *feedbacks* que seriam avaliados como esperado/desejado ou não, indicando, por exemplo, se a informação inserida está correta/incorreta/incompleta, exibindo um ponto escolhido, dentre outras opções (dependendo de cada atividade).

Finalizando todas as instruções dentro de uma seção (por exemplo, a seção ponto) seria plotado o gráfico na tela do dispositivo no qual estava sendo acessado a atividade (celular, máquina pessoal, notebook, tablet, etc). Ao clicarmos em cima do gráfico seria exibido no canto superior direito um menu – como na Figura 8 – apresentando opções como: *download* da imagem no formato PNG, zoom, deslocamento o gráfico, dentre outras.

Figura 8 – Menu do Gráfico - Colab

Fonte: Autora.

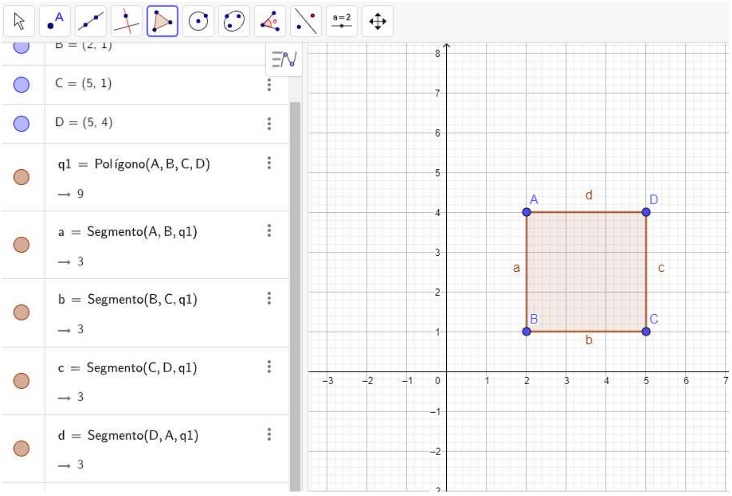
Para que o usuário conseguisse utilizar as ferramentas sugeridas na realização das atividades, foram desenvolvidos dois breves tutoriais: um deles explicando qual a sequência de atividades deveria ser desenvolvida no Geogebra e como realizá-la e o outro com uma breve orientação de como acessar o Google Colaboratory (Apêndices B e C). Para visualizar a aplicação desenvolvida, acesse: <https://colab.research.google.com/drive/1DF11HbmE-RO4svK091x71EddZN1ghNAJ?usp=sharing>.

4.2 Sequência de Atividades

As atividades do Geogebra consistiam em: 1) Criar um quadrado e transladá-lo; e 2) Construir um triângulo e rotacioná-lo em relação a um ponto fixo. Com a utilização do passo a passo, disponibilizado minutos antes da aplicação, os(as) participantes poderiam construir tais figuras geométricas. Indicamos o acesso ao Geogebra online, com link previamente disponibilizado no passo a passo, evitando que o participamente precisasse fazer o *download* e a instalação do Geogebra em seu dispositivo de acesso.

Na primeira atividade, o quadrado deveria ser criado no plano e as medidas seriam delimitadas pelo usuário.

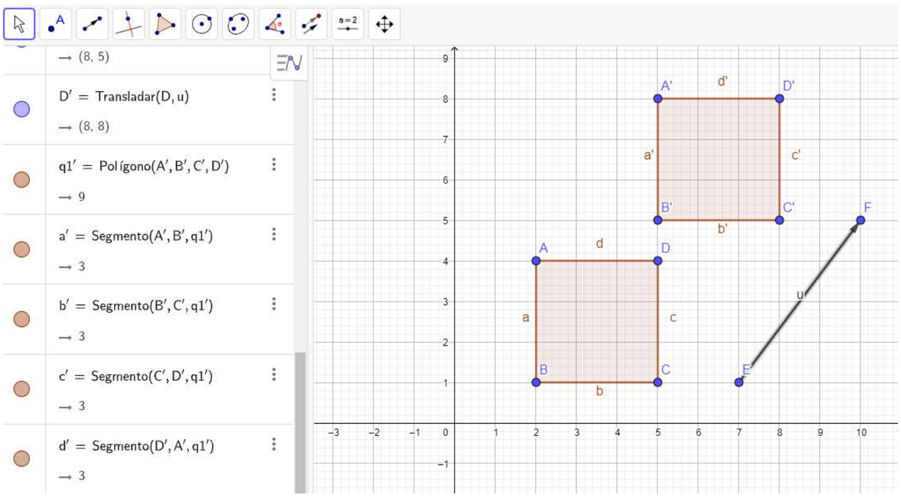
Figura 9 – Construção do quadrado



Fonte: Autora.

Com o quadrado construído, os(as) participantes deveriam criar um vetor, a partir de dois pontos de sua escolha. Cabe destacar que poderiam ser escolhidos quaisquer direção e sentido. Por fim, era necessário fazer a translação da figura geométrica, conforme representada na Figura 10.

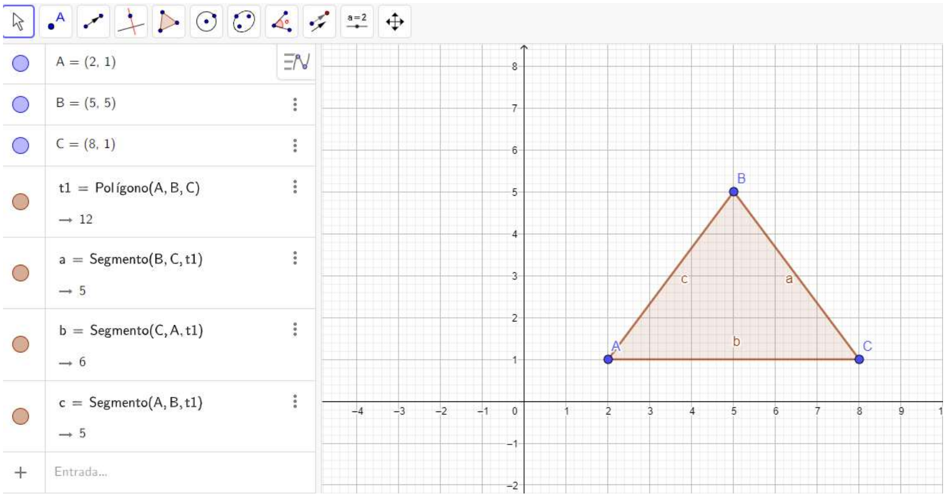
Figura 10 – Translação do quadrado



Fonte: Autora.

A segunda atividade consistia em criar um triângulo com medidas a serem escolhidas por cada participante.

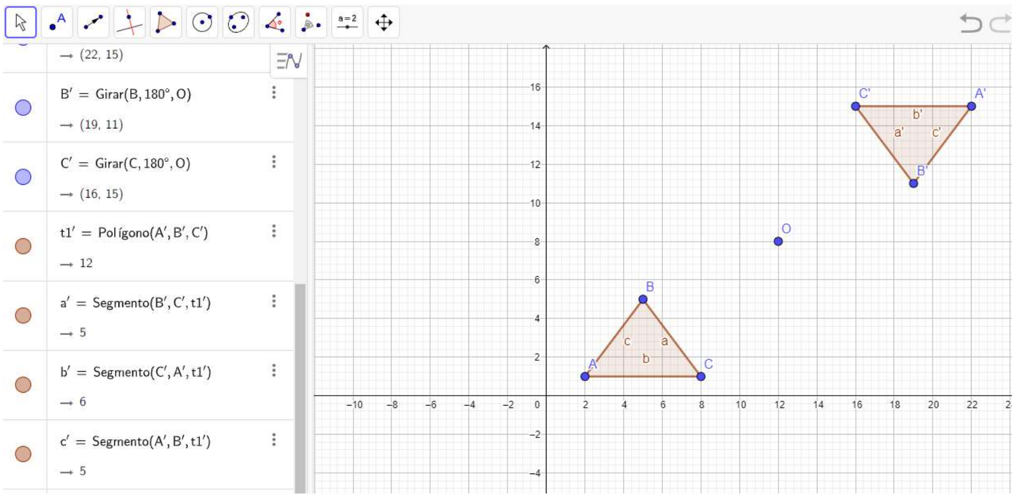
Figura 11 – Construção do triângulo



Fonte: Autora.

Logo após, como pré-estabelecido no passo a passo disponibilizado, foi solicitado que cada usuário escolhesse e plotasse um ponto – denominado O – a fim de rotacionar o triângulo construído a um ângulo de 180º com relação a O, como podemos visualizar na Figura 12.

Figura 12 – Rotação do triângulo definido



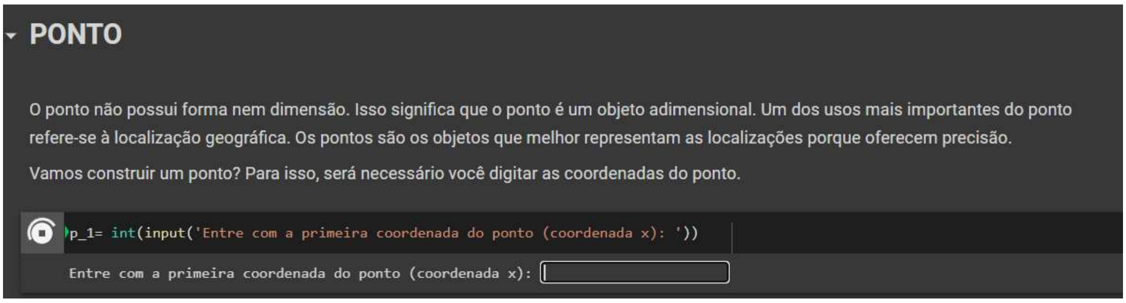
Fonte: Autora.

Finalizando as atividades no Geogebra, os discentes deveriam (caso julgassem necessário e/ou em caso de dúvidas) ler o tutorial que descrevia brevemente como acessar o Google Colab, e ainda, como fazer uma cópia da aplicação. A sequência de atividades proposta utilizando Python era composta de um conjunto de instruções e organizadas por seções, a saber

- PONTO;
- SEGMENTO;
- RETA;
- TRIÂNGULO;
- ROTAÇÃO DO TRIÂNGULO;
- QUADRADO;
- TRANSLAÇÃO DO QUADRADO.

No início das seções apresentamos uma noção intuitiva (ou, quando oportuno, definição matemática) do ente geométrico (ponto, segmento, reta, triângulo e quadrado) ou ainda comentários, explicações, exemplos ou fórmulas matemáticas envolvendo as transformações escolhidas, utilizando a linguagem de marcação LaTeX a fim de destacar informações pertinentes para a compreensão do usuário. A seguir (como já mencionamos na interação com o usuário) eram solicitadas informações através de caixas de texto, como pode ser visualizado na Figura 13.

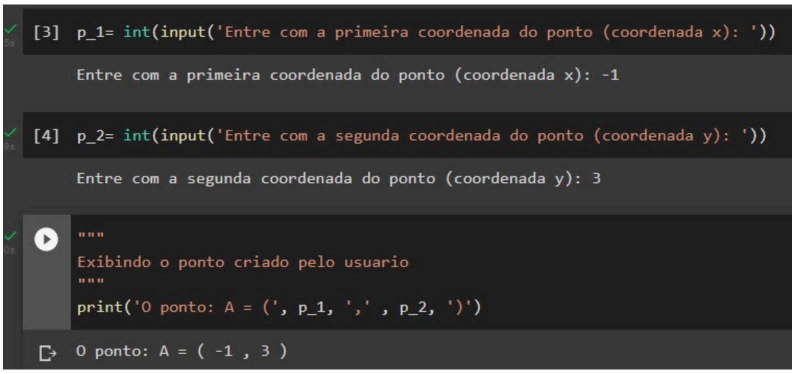
Figura 13 – Início da Seção Ponto



Fonte: Autora.

Durante essa interação eram fornecidos alguns *feedbacks* como, por exemplo, na seção ponto é exibido o par ordenado escolhido, conforme a Figura 14.

Figura 14 – *Feedback* na Seção Ponto



Fonte: Autora.

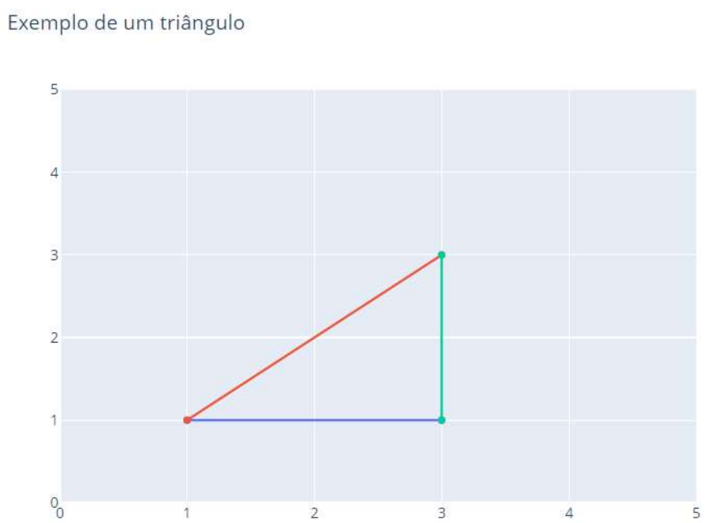
E, para finalizar cada seção, era representado graficamente o ente/figura geométrico/a, de acordo com a Figura 15.

Figura 15 – Ponto escolhido

Fonte: Autora.

Em particular, podemos destacar algumas seções. Na seção Triângulo, a autora apresentou um exemplo a partir dos pontos (1,1), (1,3) e (3,3), conforme a Figura 16. Nessa seção foi explorado o recurso de importar imagens/GIFs externos a fim de facilitar a compreensão do usuário e explorar tal possibilidade.

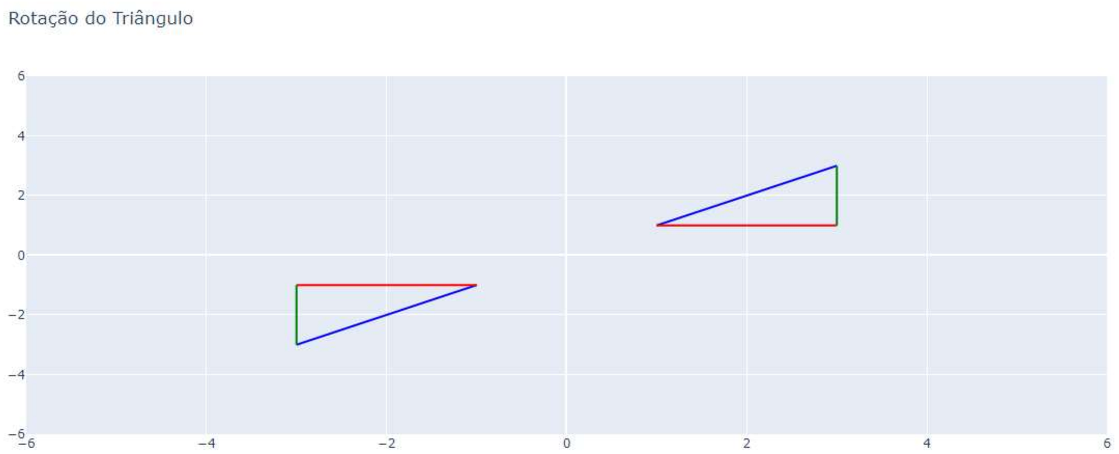
Figura 16 – Exemplo de Triângulo construído pela autora



Fonte: Autora.

Na seção Rotação do Triângulo, definimos a rotação no plano como uma transformação linear, apresentando a matriz de rotação e foi solicitado que o usuário escolhesse um ângulo para rotacionar a figura apresentada previamente como exemplo e visualizasse a rotação como na Figura 17.

Figura 17 – Rotação do triângulo por um ângulo definido pelo usuário



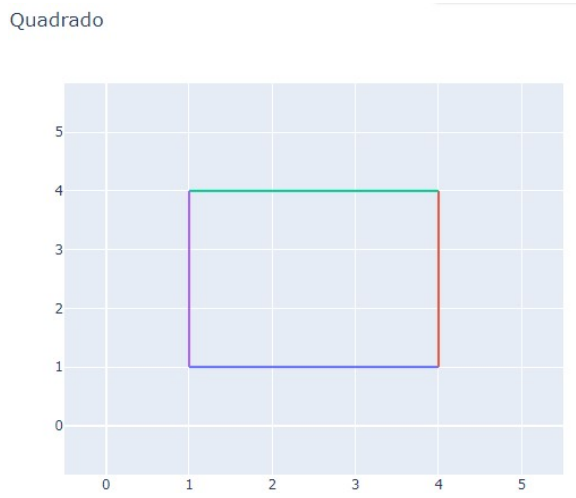
Fonte: Autora.

Na seção Quadrado, pedimos que o usuário inserisse 4 pontos – de sua escolha – de modo a formar um quadrado, a partir de seus vértices em sentido horário ou anti-horário (a restrição de sentido foi indicada ao usuário durante a reunião via Google Meet, nas orientações iniciais para a execução das atividades).

Nesse momento, a aplicação utiliza-se de equações paramétricas (previamente definidas pela autora) para descrever o segmento que une os pontos inseridos dois a dois (considerando a ordem que os pontos foram inseridos no sistema, entendendo essa como a orientação definida pelo usuário), formando um polígono de quatro lados. A fim de garantir que a distância entre vértices inseridos era a mesma e também que o ângulo entre os segmentos era de 90 graus, implementamos uma validação – em duas etapas – baseada nesses conceitos matemáticos.

De forma prática, se os pontos dois a dois estivessem a mesma distância entre si, o participante poderia continuar a atividade, já se o *feedback* exibido fosse: “não possui a mesma distância”, uma mensagem exibida pelo sistema instruiria o usuário a inserir novos valores para as coordenadas dos pontos. Da mesma forma, para a verificação dos ângulos. Com os segmentos congruentes e os ângulos todos medindo 90 graus, seria possível visualizar no plano o quadrado construído.

Figura 18 – Quadrado



Fonte: Autora.

Na seção Translação do Quadrado, apresentamos a translação no plano como uma transformação geométrica e foi solicitado que o usuário realizasse uma translação, a partir da inserção das coordenadas do ponto de translação. Além, da visualização da translação na Figura 19, na próxima etapa foi implementada uma animação.

Figura 19 – Translação do quadrado

Fonte: Autora.

Ao final da página do Google Colab, disponibilizamos um link do Google Forms com o questionário, a fim de validar esta pesquisa. A avaliação segue uma abordagem qualitativa. O formulário dispunha de dezenove questões, nas quais dezessete eram com base na Escala Likert (no sentido já especificado na metodologia) e duas eram questões abertas, em que o grupo de participantes poderia apresentar sugestões, dúvidas, críticas e/ou elogios sobre a aplicação desenvolvida em Python.

Salientamos que as atividades foram escolhidas por – em nossa opinião – estabelecerem uma sequência lógica de construção, aumentando gradativamente a complexidade envolvida em cada seção e oportunizando ao usuário adquirir maior familiaridade com as construções realizadas em cada etapa.

Por fim, explicitamos a relação entre a teoria desenvolvida no Capítulo 2 e as atividades elaboradas quando destacamos na seção Rotação do Triângulo e Translação do Quadrado o uso de Transformações Geométricas. Na aplicação desenvolvida utilizamos figuras e não mais retas, vetores ou funções (como no TCC I) no intuito de tornar mais atrativo, do ponto de vista visual, para o usuário.

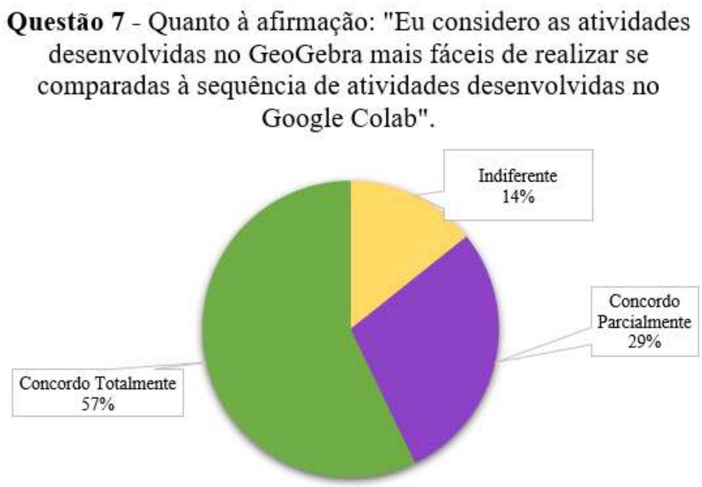
Além disso, deixamos em aberto (para trabalhos futuros) a possibilidade de explorar mudança de sistemas de coordenadas e aplicações relacionadas à Computação Gráfica (cujo o objetivo central é mover e deformar objetos) visto que, um dos casos particulares, é justamente, explorar transformações geométricas.

5 RESULTADOS E DISCUSSÕES

Neste capítulo apresentamos alguns resultados – a partir de dados obtidos – seguidos de breves discussões. A coleta de dados realizada por meio de questionário tinha por objetivo validar a aplicação desenvolvida em Python e também realizar um comparativo entre as sequências de atividades propostas em cada um dos recursos tecnológicos e (em determinados aspectos) dos recursos em si, sempre procurando verificar quanto a usabilidade.

Podemos observar, pelo exposto no capítulo anterior, que o roteiro das atividades em cada ferramenta, embora tenha o mesmo objetivo que é a visualização de duas transformações afins no plano (a rotação e a translação), não é exatamente igual. A fim de verificar qual seria a percepção dos usuários sobre a ordem/instruções/facilidade das atividades, destacamos as questões 7 e 8.

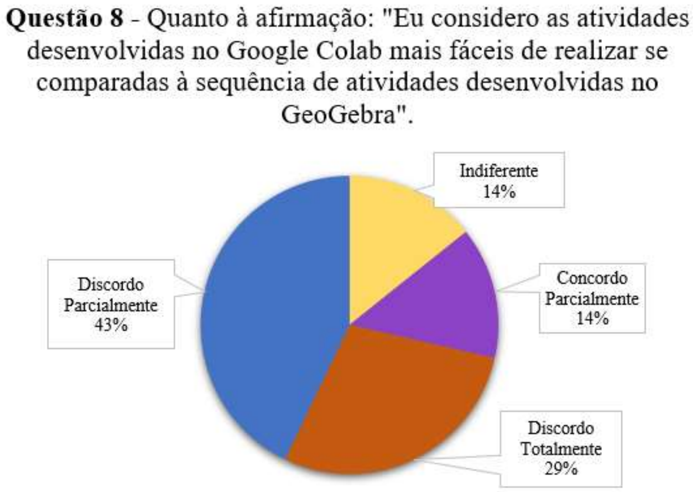
Figura 20 – Respostas da Questão 7



Fonte: Autora.

Podemos observar que todos os participantes responderam positivante, mostrando conforto e pouca dificuldade em executar as atividades no Geogebra. Quando questionamos a afirmação contrária, 72% dos discentes retornaram – possivelmente – algum desconforto/dificuldade ou mesmo falta de familiaridade para realizar a sequência de atividades proposta no Google Colab.

Figura 21 – Respostas da Questão 8



Fonte: Autora.

Acreditamos que um ponto estratégico dessa discussão (e, de certa forma, comparação entre as ferramentas) seja justamente a familiaridade dos participantes (pensamos ser possível inclusive generalizar para a maioria dos discentes do curso de Matemática - Licenciatura da UNIPAMPA - Campus Itaqui) com o GeoGebra e a falta de oportunidade em aprimorar os conhecimentos relacionados a algum tipo de linguagem de programação, por exemplo, Python, ao longo dos componentes curriculares estudados durante o curso.

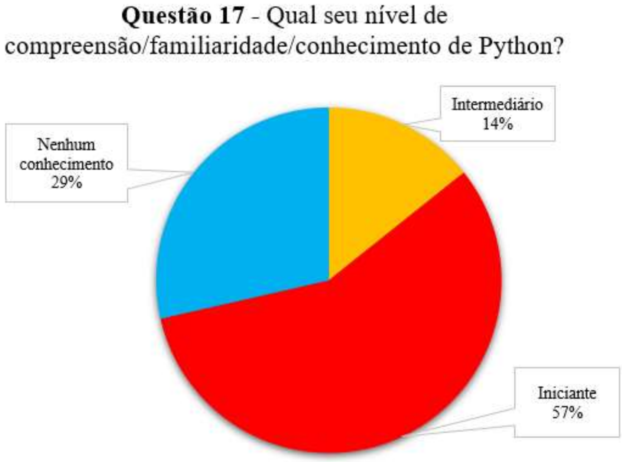
Não estamos aqui dizendo que não se deve utilizar o GeoGebra como recurso tecnológico durante a graduação. Mas, que oportunizar outras ferramentas pode diversificar e mobilizar outros conhecimentos, relacionando a outros conteúdos do ensino básico, outros componentes curriculares estudados ou mesmo aprofundando tópicos para uma futura pós-graduação na respectiva área de interesse.

O GeoGebra é muito utilizado desde o primeiro semestre do curso, podendo ser explorado em Teoria Elementar das Funções, Trigonometria, Geometria Plana, Geometria Espacial, na maioria dos Laboratórios e também nos Estágios. Já o primeiro contato com alguma linguagem de programação ocorre no último ano do curso quando os discentes matriculam-se em Algoritmos e Programação (Projeto Pedagógico do Curso - PPC, 2016).

Considerando que o curso de graduação em questão tem 9 semestres, é realmente muito difícil explorar um pouco (ou adquirir uma maior familiaridade com) alguma linguagem de programação se não for através de projetos extracurriculares ou um TCC. Fazendo uma breve pesquisa em todos os PPCs desde 2014, vemos que o componente nunca foi trabalhado antes do sexto semestre, explicando um pouco – na nossa visão –

sobre o possível desconforto dos alunos em utilizarem esse recurso, reforçando também os dados obtidos em outra questão, onde 86% dos participantes apontam seu nível de conhecimento em Pyhton como nenhum ou iniciante.

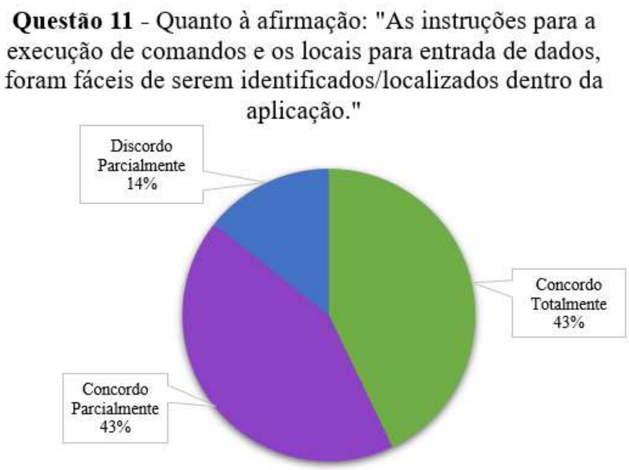
Figura 22 – Respostas da Questão 17



Fonte: Autora.

Quanto, especificamente, à validação da ferramenta, podemos destacar três questões: a primeira delas em relação às instruções para a execução dos comandos e aos locais para entrada de dados, buscando compreender se os acadêmicos tiveram dificuldades em localizar ou mesmo se a disposição escolhida poderia ter, de algum modo, atrapalhado ou confundido o usuário durante a realização da atividade.

Figura 23 – Respostas da Questão 11



Fonte: Autora.

Pelas respostas apresentadas na Figura 23, podemos concluir que para 86% dos participantes foi fácil identificar as caixas de execução de comandos ou mesmo as caixas de texto para interação do usuário com a aplicação. Tal resultado – aliado ao fato de que não interferirmos trazendo orientações complementares ao longo da aplicação – nos permite concluir que as instruções foram suficientes para a realização das atividades.

Além disso, podemos analisar a percepção/satisfação (de modo geral) em relação à aplicação desenvolvida no Google Colab, podemos destacar que:

- 100 % dos discentes ficaram satisfeitos em não ter a necessidade de baixar e/ou instalar qualquer software/bibliotecas em sua máquina pessoal. Observemos o relato de um dos participantes:

  > O ponto mais importante da aplicação desenvolvida em Python para mim foi a possibilidade de realização através do Google Colaboratory (serviço online), pois utilizo um Chromebook da Samsung nos meus estudos e o sistema operacional desse aparelho não é compatível com muitos dos softwares usualmente escolhidos para esse tipo de pesquisa.

- 85,7 % dos acadêmicos avaliaram como positivo a interação com as atividades, a possibilidade de inserir instruções/textos/conteúdos explicativos e de integração com LaTeX. Além disso, obtivemos a mesma porcentagem significativa quando considerada a possibilidade do usuário poder editar o código afim de corrigí-lo/melhorá-lo.
- 71,4 % dos usuários acharam atrativo e interativo a possibilidade de inserir imagens, links e gifs.
- 57,1 % dos participantes apontaram como ponto positivo a possibilidade de explorar gráficos e animações e a facilidade/rapidez com que se pode testar muitos exemplos sem a necessidade de refazer contas ou gráficos.

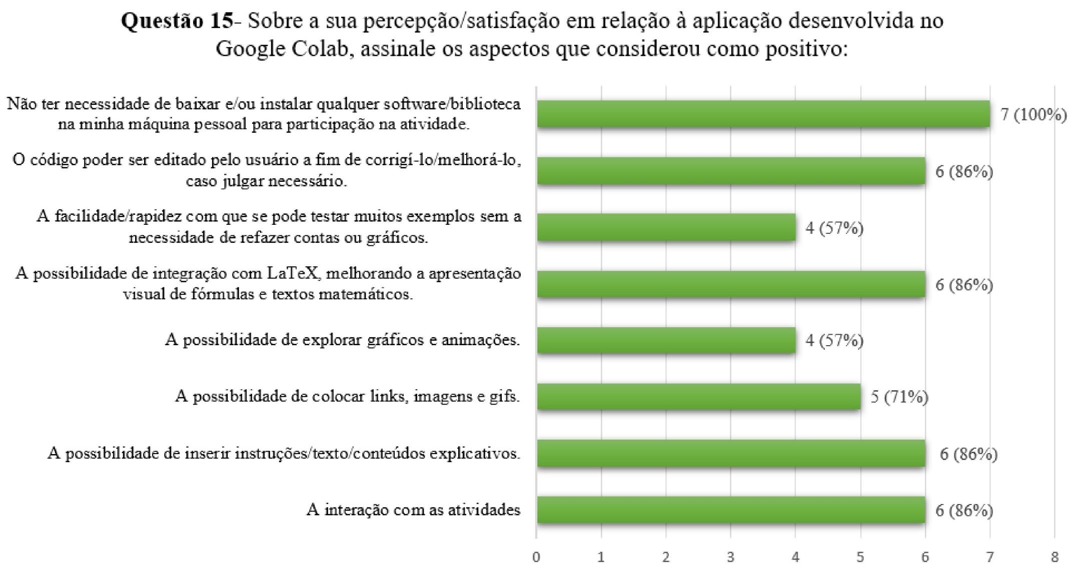
Quanto aos aspectos avaliados pelos acadêmicos participantes como negativos, destacamos:

- A falta de tratamento e prevenção de erros na construção da aplicação, com 28.6%.
- A possibilidade do usuário modificar acidentalmente as caixas de comando, impossibilitando a execução da aplicação, com 85,7%.

Esses dados podem ser observados nas Figuras 24 e 25. Visivelmente nosso principal objetivo no desenvolvimento da aplicação em Python, que era selecionar um ambiente colaborativo e/ou bibliotecas que não necessitassem de nenhum tipo de *download*, foi atingido e bem avaliado pelos participantes. A porcentagem dos participantes que consideraram positivo a exploração de gráficos e animações pode ter relação com o que já argumentamos antes, visto que muitos estudantes não conseguem compreender porque precisariamos criar/desenvolver uma nova ferramenta se o GeoGebra, por exemplo, apresenta essa funcionalidade.

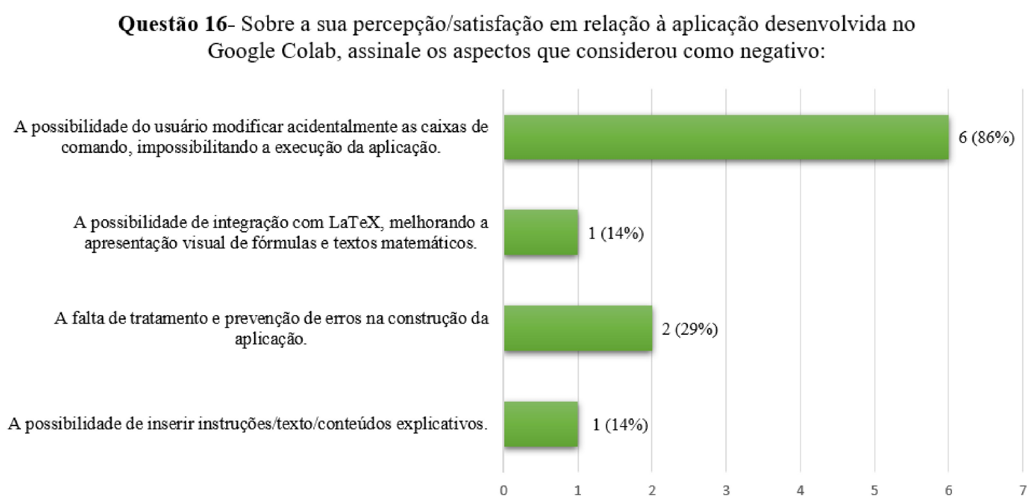
Nesse trabalho buscamos mostrar que as funcionalidades podem ser parecidas mas que a aplicação desenvolvida, ao mesmo tempo que mobiliza diversos conceitos relacionados a algoritmos e programação, também apresenta uma certa generalização. Assim, toda vez que mudamos algum ponto/vértice em uma determinada figura, não precisamos refazer todas as contas e/ou gráficos, sendo suficiente executar novamente os comandos que o gráfico é automaticamente atualizado.

Figura 24 – Respostas da Questão 15



Fonte: Autora.

Figura 25 – Respostas da Questão 16



Fonte: Autora.

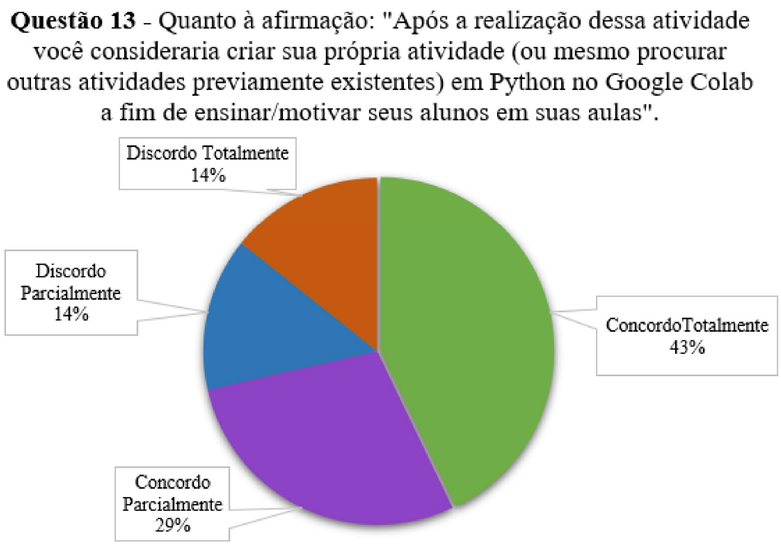
O aspecto negativo mais pontuado pelos discentes diz respeito à possibilidade do usuário modificar acidentalmente as caixas de comando, impossibilitando a execução da aplicação. Durante o planejamento pensamos que isso poderia acontecer e avaliando tal cenário pensamos em possibilidades:

- inserir como comentário o código original para que, caso o usuário alterasse por engano a caixa de código, pudesse olhar o comentário logo abaixo e corrigir o erro.
- deixar a versão original salva e apresentar uma cópia para os participantes pois, caso ocorresse alguma "quebra" no código, bastava apenas compartilhar outra cópia.

A fim de evitar poluição visual, escolhemos a segunda opção. Justamente por essa razão foi desenvolvido um tutorial para acessar o Google Colab, realizar uma cópia e compartilhá-la com a autora (para coleta de dados), antes de iniciar as atividades.

Um contraponto interessante em nossa discussão é que, apesar de concluirmos que a maioria dos participantes demonstrou maior facilidade em utilizar o GeoGebra, foi possível observar o interesse de 71% dos participantes em criar ou mesmo procurar outras atividades previamente existentes em Python no Google Colab para utilizar durante suas aulas. Acreditamos que isso se deve ao fato de considerarem a aplicação desenvolvida interativa, podendo ser utilizada para contribuir no desenvolvimento de habilidades e/ou competências nos alunos do ensino básico e/ou superior.

Figura 26 – Respostas da Questão 13



Fonte: Autora.

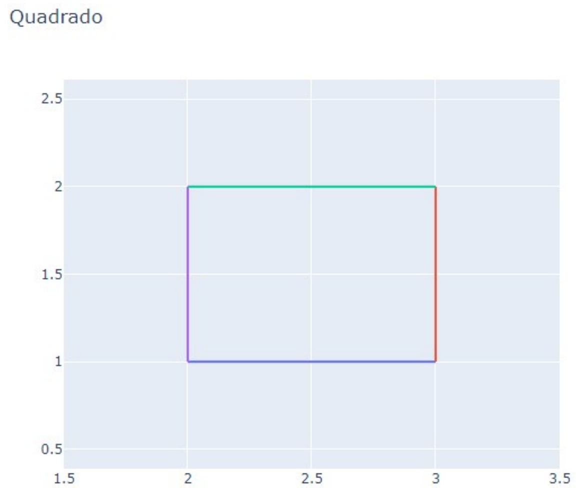
As duas últimas questões do formulário eram abertas na intenção que os participantes apresentassem dúvidas, contribuições, sugestões, elogios e/ou críticas a cerca da aplicação em Python. Na última questão em específico foi dado a seguinte sugestão: “Um ponto a ressaltar que talvez melhore, é deixar as escalas do gráfico igual, pois quando estava construindo “dava” uma percepção de não ser congruentes os lados do quadrado.”

Quando construímos o código percebemos essa distorção de imagem, assim, inserimos uma verificação (utilizando da teoria/Matemática) durante a atividade a fim de confirmar se os quatro lados eram congruentes e se todos os ângulos eram retos, caso não fossem, o discente precisaria escrever novos valores. Essa distorção de imagem está relacionada a escala e às configurações de tela do dispositivo utilizado, como podemos ver nas Figuras 27 e 29.

Figura 27 – Visualização do quadrado pelo usuário em tela *Widescreen* (16:9)

Fonte: Autora.

Figura 28 – Visualização do quadrado pelo usuário em tela proporcional à escala

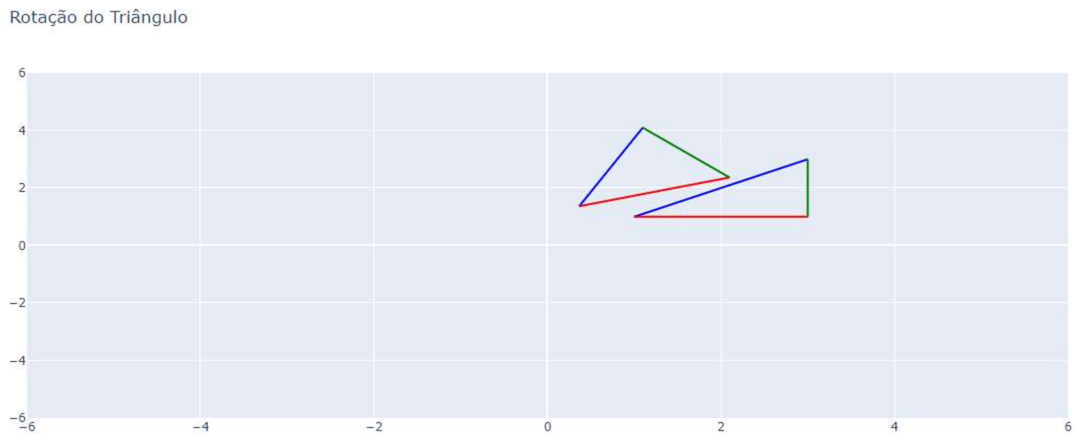


Fonte: Autora.

Quando visualizado em tela *Widescreen*, com proporção 16:9, o quadrado fica parecendo um retângulo justamente porque a escala foi configurada de forma automática, variando conforme a resolução da tela. Em tela proporcional, observamos um quadrado normalmente. A escolha da escala automática é condizente com a proposta do usuário poder inserir um quadrado qualquer, sem limitação de tamanho. Porém, poderiam ter sido realizados ajustes, utilizando mais linhas de código para contemplar os dois cenários e não obtermos distorções na imagem, por questão de tempo as correções não foram viáveis nesse momento e, muito provavelmente, sejam desenvolvidas em trabalho futuro.

Outra distorção também pode ser observada na seção Rotação do Triângulo. Se o usuário escolhesse os ângulos de 180° e 90° para realizar a rotação, a visualização no gráfico ficava correta, mas quando era escolhido outro ângulo, como por exemplo, 30°, as figuras ficavam visualmente diferentes do esperado.

Figura 29 – Visualização da rotação do triângulo pelo usuário em tela *Widescreen*



Fonte: Autora.

Com isso, realizamos alguns cálculos, a fim de verificar que a distância (e também o ângulo) entre dois pontos (segmentos) eram de fato a mesma. Tomando A$(1,1)$, B$(3,1)$, C$(3,3)$ e A'$(0.3660, 1.3660)$, B'$(2.0980, 2.3660)$ e C'$(1.0980, 4.0980)$ os pontos respectivos do triângulo rotacionado (30°) observamos, utilizando a fórmula da distância, que

$$d(A,B) = \sqrt{(1-1)^2-(3-1)^2} = 2.$$
$$d(A',B') = \sqrt{(2.3660-1.3660)^2-(2.0981-0.3660)^2} \approx 2.$$
$$d(B,C) = \sqrt{(3-1)^2-(3-1)^2} = \sqrt{8} \approx 2,8284.$$
$$d(B',C') = \sqrt{(4.0980-1.3660)^2-(1.0981-0.3660)^2} \approx 2,8284.$$
$$d(A,C) = \sqrt{(3-1)^2-(3-3)^2} = 2.$$
$$d(A',C') = \sqrt{(4.0980-2.3660)^2-(1.0981-2.0981)^2} \approx 2.$$

Além disso, encontrando os vetores $\overrightarrow{AB}$, $\overrightarrow{AC}$ (respectivamente $\overrightarrow{A'B'}$, $\overrightarrow{A'C'}$), $\overrightarrow{AB}$, $\overrightarrow{BC}$ (respectivamente $\overrightarrow{A'B'}$, $\overrightarrow{B'C'}$) e $\overrightarrow{BC}$, $\overrightarrow{CA}$ (respectivamente $\overrightarrow{B'C'}$, $\overrightarrow{C'A'}$), utilizamos a fórmula para encontrar o ângulo entre dois vetores (fórmula do cosseno) – que aprendemos em Geometria Analítica – para obter os ângulos de 45°, 90° e 45°, respectivamente em cada caso. Destacamos, apesar de omitirmos aqui os detalhes, que utilizamos nesses cálculos aproximações com quatro casas decimais.

Podemos perceber que de fato os triângulos são congruentes e que o erro em questão está novamente relacionado à escala de visualização do gráfico. Fizemos algumas tentativas para tentar contornar o problema, mas, por falta de tempo, tais ajustes também foram listados para trabalhos futuros. Não podemos afirmar que a mesma correção que precisa ser feita para resolver o problema de escala no caso quadrado seria suficiente para solucionar também essa questão, para verificarmos isso são necessários alguns testes.

Em resumo, sobre as atividades realizadas via Google Meet, observamos que os discentes realizaram com maior rapidez e desenvoltura as atividades propostas no Geogebra, bem como manifestaram preferência por essa ferramenta se comparada à utilização da linguagem de programação. Isso pode estar relacionado à resistência com relação a novos recursos tecnológicos, à diversas experiências com o Geogebra e a pouca exploração do uso de algoritmos e de programação. Apesar da insegurança demonstrada nos dados obtidos do formulário, todos os participantes conseguiram realizar as atividades, alguns deixaram considerações/sugestões e elogios.

6 CONSIDERAÇÕES FINAIS

Neste trabalho propomos a utilização de linguagem de programação, mais especificamente Python, na elaboração de atividades envolvendo figuras e transformações geométricas. Entendemos que foi desafiador a construção de uma aplicação com essa temática, visto que conceitos relacionados à programação são vistos de forma isolada e em poucos componentes curriculares do curso de graduação que se está concluindo, indicando assim, que as maiores dificuldades encontradas na realização do trabalho não estiveram presentes em sua escrita em si, mas, sim, na sua execução (na prática).

Quando estamos programando, de certo modo, estamos transformando padrões em código, de modo que um algoritmo possa ser executado sem a necessidade de repetirmos – a cada etapa – um mesmo procedimento (uma espécie de generalização) e isso pode nos conduzir a resultados equivocados. Sendo necessário buscar o conhecimento teórico (no caso, na Matemática) para encontrarmos os possíveis erros e tornarmos funcional qualquer aplicação que estivermos tentando desenvolver.

Desde o ensino básico somos pouco incentivados a aplicar conceitos matemáticos no cotidiano e a fazer (cor)relações com outras áreas do conhecimento (como a computação por exemplo). A fragmentação dos conteúdos trabalhados em sala de aula, bem como a extensa matriz curricular que deve ser cumprida, acaba desmotivando o aluno. Uma estratégia para nós, futuros professores, é oportunizarmos o desenvolvimento de habilidades como a percepção, ao trabalharmos, por exemplo, com as transformações geométricas, fazendo uso de recursos tecnológicos.

No processo de construção das atividades buscou-se conceituar e interpretar as transformações geométricas através da manipulação de dados e o deslocamento dinâmico das figuras, de forma a contribuir para uma aprendizagem significativa por meio da visualização. Dessa forma, buscando tornar um pouco mais concretos alguns conceitos que são abstratos, o que pode proporcionar aos estudantes o desenvolvimento de novas habilidades.

A ferramenta foi desenvolvida para aprimorar os conhecimentos relacionados a algoritmos e programação da autora, mas também como sugestão de uma sequência de atividades que possa ser aplicada em sala de aula. Inicialmente, no âmbito do ensino superior e, com as devidas adaptações, posteriormente no âmbito da educação básica.

Quanto às dificuldades, podemos listar a falta de literatura específica relacionada ao assunto. Em relação às bibliotecas disponíveis e como determinadas funcionalidades deveriam ser utilizadas em cada uma delas, encontrávamos, algumas vezes, exemplos – em sua grande maioria em inglês. Contudo, nenhum deles muito parecido com o que estávamos tentando desenvolver, sendo sempre necessários ajustes e adaptações que exigiam conhecimentos (nem que mínimos) de programação.

Em relação à aplicação em si, quase não encontramos experiências relatadas em trabalhos científicos que pudessem nos dar direções ou indicassem alternativas quanto à estrutura, bibliotecas, layout, etc. Algumas referências encontradas até utilizavam de programação (outras linguagens) para o ensino das transformações geométricas (em geral no plano). Porém, eram trabalhos focados em apresentar uma ferramenta do ponto de vista educacional, sem muitas vezes explorar os conceitos matemáticos que justificavam as construções/conclusões/atividades realizadas.

Destacamos que um diferencial em nosso trabalho foi a preocupação em utilizarmos bibliotecas que não necessitassem de nenhum tipo de instalação, *downloads* e/ou atualizações. Isso nos trouxe desvantagens do ponto de vista visual (comentadas no capítulo anterior) mas, por outro lado, permite que qualquer pessoa, independente das configurações de sua máquina pessoal, possa fazer uso da aplicação e contribuir para aprimorá-la.

Como trabalho futuro pretendemos buscar, analisar e implementar soluções/melhorias, a fim de corrigir, por exemplo, problemas visuais como os que destacamos aqui e considerar as sugestões apresentadas pelos acadêmicos durante a aplicação das atividades. Planejamos publicar em revista científica e/ou evento científico os resultados obtidos, bem como em momento posterior, disponibilizar o código para que possa ser utilizado como recurso didático para o ensino de transformações geométricas, inicialmente, no ensino superior, sob a ótica de futuros professores.

Enquanto licencianda em Matemática, este trabalho trouxe grandes contribuições para minha formação acadêmica, pois possibilitou não apenas um estudo, mas uma aplicação das transformações afins utilizando de linguagem de programação. Dessa forma, retomando e ampliando vários conceitos trabalhados durante a graduação e sobre uma temática que eu também devo abordar na escola, visto que transformações geométricas estão previstas como conteúdo programático na BNCC.

Reforço aqui a dificuldade que sempre tive com tópicos relacionados à Geometria e como este trabalho me permitiu visualizar de forma prática a relação entre a Geometria e a Álgebra, se tornando cada vez mais interessante à medida que conseguia implementar os códigos de forma a obter o resultado desejado.

Por fim, foi possível perceber que não é apenas o Geogebra que auxilia na passagem do conceito abstrato para a visualização de forma concreta mas que, utilizando Python, além de visualizarmos figuras e animações (inclusive) também podemos explorar um pouco de algoritmos e de programação, o que traz algumas possibilidades a mais e pode ser utilizado para desenvolver conceitos envolvendo muitos outros tópicos como Estatística, Matemática Financeira e até mesmo aplicações da matemática em outras áreas do conhecimento.

REFERÊNCIAS

BATTAIOLA, A. L. **Notas de aula do Curso de Computação Gráfica**. [Adaptado por: Prof. Dr. José Hiroki Saito]. Departamento de Computação da Univesidade Federal de São Carlos, 2003. Disponível em: <http://www2.dc.ufscar.br/~saito/download/comp-grafica/apostila.pdf>. Acesso em: 27 jan. 2022.

BOLDRINI, José Luiz *et al.* **Álgebra Linear**. 3. ed. São Paulo: Harper e Row do Brasil, 1980.

GIL, A. C. **Como elaborar projetos de pesquisa.** 6. ed. São Paulo: Atlas, 2019.

LAY, D. C. **Álgebra Linear e suas aplicações.** Tradução: Ricardo Camelier e Valéria de Magalhães Iório. Rio de Janeiro: LTC, 1999.

LIMA, E. L. **Coordenadas no Plano**. 6. ed. Rio de Janeiro: SBM, 2013.

MABUCHI, S. T. **Transformações geométricas: a trajetória de um conteúdo ainda não incorporado às práticas escolares e nem à formação de professores.** Dissertação (Mestrado em Educação Matemática) - PUC/São Paulo. São Paulo, 2000. Disponível em: <https://tede2.pucsp.br/handle/handle/11217>. Acesso em: 27 jan. 2022.

UNIVERSIDADE FEDERAL DO PAMPA. **Projeto Pedagógico do Curso de Matemática - Licenciatura**. [Itaqui: UNIPAMPA], 2016. Disponível em: <https://dspace.unipampa.edu.br//handle/riu/120>. Acesso em: 03 ago. 2022.

SILVA, G. H. G. **Ambientes de Geometria Dinâmica: Potencialidades e Imprevistos.** Revista Brasileira de Ensino de Ciência e Tecnologia, 2012. Disponível em: <https://periodicos.utfpr.edu.br/rbect/article/view/900>. Acesso em: 28 jan. 2022.

SILVA, J. E. L. **Comparando as linguagens de programação Fortran e Python através de problemas matemáticos.** Monografia (Conclusão do Curso de Licenciatura em Matemática) - Universidade Federal de Campina Grande, Campus Cuité. Cuité - PB, 2019. Disponível em: <http://dspace.sti.ufcg.edu.br:8080/jspui/handle/riufcg/8382>. Acesso em: 28 jan. 2022.

SILVA, R. F. **Transformações geométricas no plano e no espaço.** Dissertação (Mestrado em Matemática) - Programa de Mestrado Profissional em Matemática em Rede Nacional (PROFMAT). Universidade Federal da Paraíba. Paraíba, 2013. Disponível em: <https://repositorio.ufpb.br/jspui/handle/tede/7476>. Acesso em: 21 dez. 2021.

SOUZA, Marcela L. V.; LOPES, Sérgio A. A.; NASCIMENTO, Kleber G. **Álgebra: Proposta da unidade temática na BNCC e desafios por sua trajetória ao longo dos nove anos do Ensino Fundamental**. 1. ed. Rio de Janeiro: Associação Nacional dos Professores de Matemática na Educação Básica, 2020. Disponível em: Link. Acesso em: 28 jan. 2022.

STEINBRUSH, Alfredo; WINTERLE, Paulo. **Álgebra Linear.** São Paulo: Makron Books, 1987.

APÊNDICE A — APLICAÇÃO NO GOOGLE COLAB

TRANSFORMAÇÕES GEOMÉTRICAS AFINS: Uma abordagem utilizando python

Prezado participante, como parte da aplicação do meu trabalho de conclusão de curso (TCC), convidamos para a realização de um conjunto de atividades que serão desenvolvidas abaixo.

Algumas instruções para a realização da atividade:

- **O sistema está pré-programado para receber como entrada apenas números inteiros.**

- Você deve executar primeiro as bibliotecas utilizadas (print da imagem);

```
"""
Importando libs para todo projeto
"""
import numpy as np
import matplotlib.pyplot as plt
import plotly.graph_objects as go
```

- As atividades foram organizadas em seções, de modo que você pode realizá-las de forma independente, ou seja, qualquer ordem desejada. Sugerimos seguir a ordem pré-estabelecida; (print seção comprimida e expandida);

TRANSFORMAÇÕES GEOMÉTRICAS AFINS: Uma abordagem ultilizando python

PONTO

SEGMENTO

RETA

TRIÂNGULO

ROTAÇÃO DO TRIÂNGULO

QUADRADO

TRANSLAÇÃO DO QUADRADO

Respondendo ao Formulário.

Secção

- A cada interação que solicitar uma informação você deve digitá-la no local indicado pelo cursor e pressionar ENTER, fazendo com que o sistema armazene-a;

```
p_1= int(input('Entre com a primeira coordenada do ponto (coordenada x): '))

Entre com a primeira coordenada do ponto (coordenada x): 2
```

Primeiro, vamos importar todas as bibliotecas utilizadas:

```
"""
Importando libs para todo projeto
"""
import numpy as np
import matplotlib.pyplot as plt
import plotly.graph_objects as go
```

PONTO

O ponto não possui forma nem dimensão. Isso significa que o ponto é um objeto adimensional. Um dos usos mais importantes do ponto refere-se à localização geográfica. Os pontos são os objetos que melhor representam as localizações porque oferecem precisão.

Vamos construir um ponto? Para isso, será necessário você digitar as coordenadas do ponto.

```
p_1= int(input('Entre com a primeira coordenada do ponto (coordenada x): '))
```

```
p_2= int(input('Entre com a segunda coordenada do ponto (coordenada y): '))
```

```
"""
Exibindo o ponto criado pelo usuario
"""
print('O ponto: A = (', p_1, ',' , p_2, ')')
```

Abaixo, será exibido o ponto escolhido no plano cartesiano:

```
"""
Criando grafico com Ploty
"""
fig = go.Figure(
    data=[go.Scatter(x=[p_1], y=[p_2], text=["Ponto A"])],
    layout=go.Layout(
        xaxis=dict(range = [-10, 10], autorange = True),
```

```
        yaxis=dict(range = [-10, 10], autorange = True),
        title="Ponto Definido pelo Usuário"))

"""
Plotando (Desenhando) grafico definido por fig.
"""
fig.show()
```

SEGMENTO

Um segmento de linha é uma parte de uma linha que é limitada por dois pontos: ponto final e ponto inicial, chamados de "extremos". Da geometria sabemos que por dois pontos passa um único segmento, então podemos construir um segmento a partir de dois pontos A e B.

Você deve digitar abaixo as coordenadas (x,y) do ponto inicial A:

```
a_1= int(input('Entre com a primeira coordenada do ponto (coordenada x): '))
```

```
a_2= int(input('Entre com a segunda coordenada do ponto (coordenada y): '))
```

Agora deve digitar a segunda coordenada (x,y) do ponto final B:

```
b_1= int(input('Entre com a primeira coordenada do ponto (coordenada x): '))
```

```
b_2= int(input('Entre com a segunda coordenada do ponto (coordenada y): '))
```

Visualize o segmento no gráfico:

```
"""
Criando grafico com Ploty
Usando equacao parametrica de um segmento generica que passa por dois pontos dados
qualquer, A = (a_1, a_2) e B = (b_1, b_2)
"""

fig = go.Figure(
    data=[
          go.Scatter(x = [a_1, b_1], y = [a_2, b_2])
          ],
    layout=go.Layout(
        xaxis=dict(range=[-20, 20], autorange=True),
        yaxis=dict(range=[-20, 20], autorange=True),
        title="Segmento"))

"""
Plotando (Desenhando) grafico definido por fig.
```

```
"""
fig.show()
```

RETA

A reta é formada por infinitos pontos que estão alinhados. Ela é ilimitada nos dois sentidos. Quando construímos uma reta devemos utilizar letras minúsculas para representá-la. Da geometria sabemos que por dois pontos passa uma única reta, então podemos construir uma reta a partir de dois pontos A e B.

Você deve digitar abaixo as coordenadas (x,y) do ponto A (primeiro ponto em que a reta deve passar):

```
a_1= int(input('Entre com a primeira coordenada do ponto (coordenada x): '))
```

```
a_2= int(input('Entre com a segunda coordenada do ponto (coordenada y): '))
```

Você deve digitar abaixo as coordenadas (x,y) do ponto B (segundo ponto em que a reta deve passar):

```
b_1= int(input('Entre com a primeira coordenada do ponto (coordenada x): '))
```

```
b_2= int(input('Entre com a segunda coordenada do ponto (coordenada y): '))
```

Visualização da reta no gráfico:

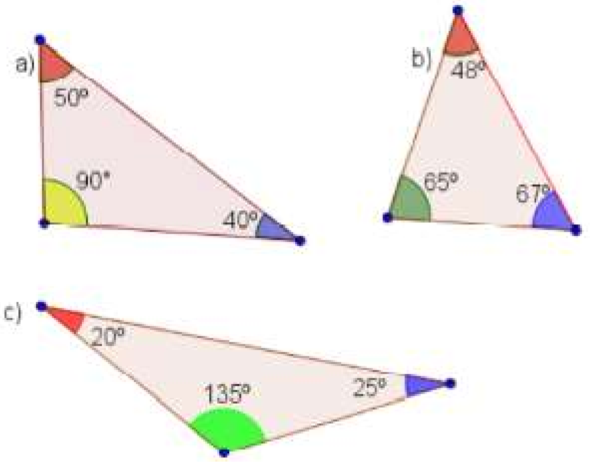
```
"""
Criando grafico com Ploty
Usando equacao parametrica de uma reta generica que passa por dois pontos dados
qualquer, A = (a_1, a_2) e B = (b_1, b_2)
"""

t = np.arange(-20, 20)
fig = go.Figure()
fig.add_trace(go.Scatter(x = (b_2 - a_2) * t + a_1, y = (b_1 - a_1) * t + b_1))
fig.update_layout(title='Reta',
xaxis_title='x',
xaxis=dict(range=[-5, 5], autorange=True),
yaxis=dict(range=[-5, 5], autorange=True),
yaxis_title='f(x)',
template = 'plotly_white') #adicionando o template

```

```
"""
Plotando (Desenhando) grafico definido por fig.
"""

fig.show()
```

TRIÂNGULO

No plano, o triângulo é a figura geométrica que ocupa o espaço interno limitado por três segmentos de reta que concorrem, dois a dois, em três pontos diferentes formando três lados e três ângulos internos que somam 180°.

Na figura abaixo, podemos observar os ângulos internos de alguns triângulos. Lembre que a soma destes ângulos sempre resultará 180º.




```
"""
Criando um triangulo em Plotly
"""

fig = go.Figure(
    data=[go.Scatter(x=[1, 3], y=[1, 1]), go.Scatter(x=[1, 3], y=[1, 3]), go.Scatter(x=
    layout=go.Layout(
        xaxis=dict(range=[0, 5], autorange=False),
        yaxis=dict(range=[0, 5], autorange=False),
        title="Exemplo de um triângulo")

)


"""
Plotando (Desenhando) grafico definido por fig.
"""

fig.show()
```

ROTAÇÃO DO TRIÂNGULO

A transformação de rotação no plano permite rotacionar uma reta/um segmento/um gráfico, dentre outros entes geométricos em torno de um ponto (no nosso exemplo, faremos a rotação em torno da origem) um ângulo θ (no sentindo anti-horário). Matematicamente, podemos representar a rotação através de uma transformação linear $T : \mathbb{R}^2 \to \mathbb{R}^2$ dada por

$$T\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x\ \cos(\theta) - y\,\mathrm{sen}(\theta) \\ y\ \cos(\theta) + x\,\mathrm{sen}(\theta) \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\mathrm{sen}(\theta) \\ \mathrm{sen}(\theta) & \cos(\theta) \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}.$$

Na definição acima multiplicamos a matriz de rotação por um vetor a ser rotacionado. Para a nossa aplicação fizemos as devidas adaptações para rotacionarmos os segmentos de reta que formam o triângulo.

Escolha um ângulo para rotacionarmos o triângulo:

```
angulo= int(input('Ângulo de rotação (um número inteiro): '))
```

```
"""
Criando grafico com Ploty
Usando equacao parametrica do quadrado e a matriz de rotação
"""

#============= Definindo os parâmetros ==============
t=np.linspace(1,3)
s=np.linspace(1,3)
r=np.linspace(1,3) #intervalo

theta = 2*(np.pi/360)*angulo

reta_1=np.array([t, t]) #Escrevendo os pontos da reta como um vetor
reta_2=np.array([s, 1 + s*0])
reta_3=np.array([3 + r*0 ,r])


matriz=np.array([[np.cos(theta) , -np.sin(theta)], [np.sin(theta) , np.cos(theta)]]) #c


reta_t=np.dot(matriz, reta_1)
reta_s=np.dot(matriz, reta_2)
reta_r=np.dot(matriz, reta_3)

fig = go.Figure(
    data=[go.Scatter(x=reta_1[0], y=reta_1[1], line=dict(width=2, color="blue")),
          go.Scatter(x=reta_t[0], y=reta_t[1], line=dict(width=2, color="blue")),
          go.Scatter(x=reta_2[0], y=reta_2[1], line=dict(width=2, color="red")),
```

```
            go.Scatter(x=reta_s[0], y=reta_s[1], line=dict(width=2, color="red")),
            go.Scatter(x=reta_3[0], y=reta_3[1], line=dict(width=2, color="green")),
            go.Scatter(x=reta_r[0], y=reta_r[1], line=dict(width=2, color="green")),
            ],
      layout=go.Layout(
          xaxis=dict(range=[-6, 6], scaleratio = 1, autorange=False),
          yaxis=dict(range=[-6, 6], scaleratio = 1, autorange=False),
          title="Rotação do Triângulo"))

"""
Plotando (Desenhando) grafico definido por fig.
"""

fig.show()

```

QUADRADO

O quadrado é um polígono que possui quatro lados de mesma medida e quatro ângulos retos. A seguir você deve inserir quatro pontos que irão formar quatro segmentos (de acordo com o mesmo conceito matemático utilizado na construção de um segmento).

Digite as cordenadas do ponto A:

```
x_1= int(input('Entre com a primeira coordenada do ponto A: '))
```

```
y_1= int(input('Entre com a segunda coordenada do ponto A: '))
```

```
v_1 = np.array([x_1,y_1]) #lista com os elementos
print('Temos o ponto: A = (', v_1[0], ',' , v_1[1], ')')
```

Digite as coordenadas do ponto B:

```
x_2= int(input('Entre com a primeira coordenada do ponto B: '))
```

```
y_2= int(input('Entre com a segunda coordenada do ponto B: '))
```

```
v_2 = np.array([x_2,y_2]) #lista com os elementos
print('Temos o ponto: B = (', v_2[0], ',', v_2[1], ')')
```

Digite as coordenadas do ponto C:

```
x_3= int(input('Entre com a primeira coordenada do ponto C: '))
```

```
y_3= int(input('Entre com a segunda coordenada do ponto C: '))
```

```
v_3 = np.array([x_3,y_3]) #lista com os elementos
print('Temos o ponto: C = (', v_3[0], ',', v_3[1], ')')
```

Digite as coordenadas do ponto D:

```
x_4= int(input('Entre com a primeira coordenada do ponto D: '))
```

```
y_4= int(input('Entre com a segunda coordenada do ponto D: '))
```

```
v_4 = np.array([x_4,y_4]) #lista com os elementos
print('Temos o ponto: D = (', v_4[0], ',', v_4[1], ')')
```

Precisamos identificar se esses quatro segmentos formam um quadrado, para isso vamos verificar se os quatro lados são congruentes e se os quatro ângulos são retos.

Para verificar se os lados são congruentes basta verificar o ¨tamanho¨ do segmento $\overline{AB}$, a partir da fórmula da distância:

$$d(A,B) = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}$$

Fazemos o mesmo procedimento para $\overline{BC}$, $\overline{CD}$ e $\overline{DA}$.

Se $d(A,B) = d(B,C) = d(C,D) = d(D,A)$, então temos os quatro lados congruentes.

```
"""
Verificando se os pontos tem a mesma distancia entre si.
"""

def dist(A, B):
  return ((B[1] - A[1])**2 + (B[0] - A[0])**2)**(1/2)

def comparedist():
 if(dist(v_1,v_2)==dist(v_2,v_3) and dist(v_2,v_3)==dist(v_3,v_4) and dist(v_3,v_4)==di
    return print('Os pontos A,B,C,D tem a mesma distância entre si')
 else:
    return print('Os pontos A,B,C,D não tem a mesma distância entre si. Retorne com out

comparedist()
```

Para verificar se os ângulos são retos, inicialmente construímos os vetores $\overrightarrow{AB}$ e $\overrightarrow{BC}$:

$$\overrightarrow{AB} = B - A = (x_2 - x_2, y_2 - y_1)$$
$$\overrightarrow{BC} = C - B = (x_3 - x_2, y_3 - y_2).$$

Agora, utilizaremos conceitos de geometria analítica, mais especificamente, o produto escalar entre dois vetores para verificar se o ângulo entre eles é de 90º.

$$\begin{aligned}\overrightarrow{AB} \cdot \overrightarrow{BC} &= (x_2 - x_1\ ,\ y_2 - y_1) \cdot (x_3 - x_2\ ,\ y_3 - y_2)\\ &= (x_2 - x_1) \cdot (x_3 - x_2) + (y_2 - y_1) \cdot (y_3 - y_2).\end{aligned}$$

Se este produto escalar for igual a zero, o ângulo entre os vetores será de 90º.

```
"""
Verificando se os segmentos tem o mesmo angulo entre si, utilizando produto escalar.
"""

def escalar(A,B,C):
  V=[B[0] - A[0], B[1] - A[1]]
  W=[B[0] - C[0], B[1] - C[1]]
  return np.dot(V, W)

escalar(v_1,v_2,v_3)==escalar(v_2,v_3,v_4)

def compareangulos():
 if(escalar(v_1,v_2,v_3)==escalar(v_2,v_3,v_4) and escalar(v_2,v_3,v_4)==escalar(v_3,v_
    return print('Os 4 segmentos têm o mesmo ângulo entre sim.')
 else:
    return print('Os 4 segmentos não têm o mesmo ângulo entre sim. Retorne com outros v

compareangulos()
```

Observe abaixo o quadrado construído:

```
"""
Criando grafico com Ploty, utilizando a equação parametrica
"""

h=np.linspace(0,1)
f=np.linspace(0,1)
g=np.linspace(0,1)
j=np.linspace(0,1)

reta_h = np.array([(v_2[0] - v_1[0]) * h + v_1[0], (v_2[1] - v_1[1]) * h + v_1[1]])
reta_f = np.array([(v_3[0] - v_2[0]) * f + v_2[0], (v_3[1] - v_2[1]) * f + v_2[1]])
reta_g = np.array([(v_4[0] - v_3[0]) * g + v_3[0], (v_4[1] - v_3[1]) * g + v_3[1]])
reta_j = np.array([(v_1[0] - v_4[0]) * j + v_4[0], (v_1[1] - v_4[1]) * j + v_4[1]])

fig = go.Figure(
    data=[
          go.Scatter(x = reta_h[0], y = reta_h[1]),
```

```
            go.Scatter(x=reta_f[0], y=reta_f[1]),
            go.Scatter(x=reta_g[0], y=reta_g[1]),
            go.Scatter(x=reta_j[0], y=reta_j[1])
            ],
      layout=go.Layout(
          xaxis=dict(range=[0, 5], autorange=True),
          yaxis=dict(range=[0, 5], autorange=True),
          title="Quadrado")

)

"""
Plotando (Desenhando) grafico definido por fig.
"""
fig.show()
```

TRANSLAÇÃO DO QUADRADO

Uma transformação afim muito importante na computação gráfica é a translação. Matematicamente ela consiste em adicionar um vetor $\mathbf{b} = (\alpha, \beta)$ não-nulo ao vetor (x, y), isto é, $T : \mathbb{R}^2 \to \mathbb{R}^2$ é dada por

$$T \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \alpha \\ \beta \end{bmatrix}.$$

Para nossa aplicação fizemos as devidas adaptações (matemática e computacionalmente) para transladarmos os segmentos de reta que formam o quadrado.

Escolha um ponto em relação ao qual o quadrado será transladado.

Entre com a coordenada x do ponto de translação

```
h_0= int(input('Entre com a primeira coordenada do ponto (coordenada x): '))
```

Entre com a coordenada y do ponto de translação

```
h_1= int(input('Entre com a segunda coordenada do ponto (coordenada y): '))
```

```
"""
Criando grafico com Ploty, utilizando a equação parametrica e sua translacao
"""

h=np.linspace(0,1)
f=np.linspace(0,1)
g=np.linspace(0,1)
j=np.linspace(0,1)

```

```
reta_h = np.array([(v_2[0] - v_1[0]) * h + v_1[0], (v_2[1] - v_1[1]) * h + v_1[1]])
reta_f = np.array([(v_3[0] - v_2[0]) * f + v_2[0], (v_3[1] - v_2[1]) * f + v_2[1]])
reta_g = np.array([(v_4[0] - v_3[0]) * g + v_3[0], (v_4[1] - v_3[1]) * g + v_3[1]])
reta_j = np.array([(v_1[0] - v_4[0]) * j + v_4[0], (v_1[1] - v_4[1]) * j + v_4[1]])

fig = go.Figure(
    data=[
          go.Scatter(x = reta_h[0], y = reta_h[1]),
          go.Scatter(x = reta_f[0], y = reta_f[1]),
          go.Scatter(x = reta_g[0], y = reta_g[1]),
          go.Scatter(x = reta_j[0], y = reta_j[1]),
          go.Scatter(x = reta_h[0] + h_0, y = reta_h[1] + h_1),
          go.Scatter(x = reta_f[0] + h_0, y = reta_f[1] + h_1),
          go.Scatter(x = reta_g[0] + h_0, y = reta_g[1] + h_1),
          go.Scatter(x = reta_j[0] + h_0, y = reta_j[1] + h_1)
          ],
    layout=go.Layout(
        xaxis=dict(range=[0, 10], autorange=True),
        yaxis=dict(range=[0, 10], autorange=True),
        title="Translação do quadrado")

)

"""
Plotando (Desenhando) grafico definido por fig.
"""

fig.show()
```

Vamos criar agora uma animação para a translação do quadrado.

```
"""
Criando animacao com Ploty
"""

h=np.linspace(0,1)
f=np.linspace(0,1)
g=np.linspace(0,1)
j=np.linspace(0,1)
q=np.linspace(0,20)
N=20

reta_h = np.array([(v_2[0] - v_1[0]) * h + v_1[0], (v_2[1] - v_1[1]) * h + v_1[1]])
reta_f = np.array([(v_3[0] - v_2[0]) * f + v_2[0], (v_3[1] - v_2[1]) * f + v_2[1]])
reta_g = np.array([(v_4[0] - v_3[0]) * g + v_3[0], (v_4[1] - v_3[1]) * g + v_3[1]])
reta_j = np.array([(v_1[0] - v_4[0]) * j + v_4[0], (v_1[1] - v_4[1]) * j + v_4[1]])

fig = go.Figure(
    data=[
          go.Scatter(x = reta_h[0], y = reta_h[1]),
          go.Scatter(x = reta_f[0], y = reta_f[1]),
```

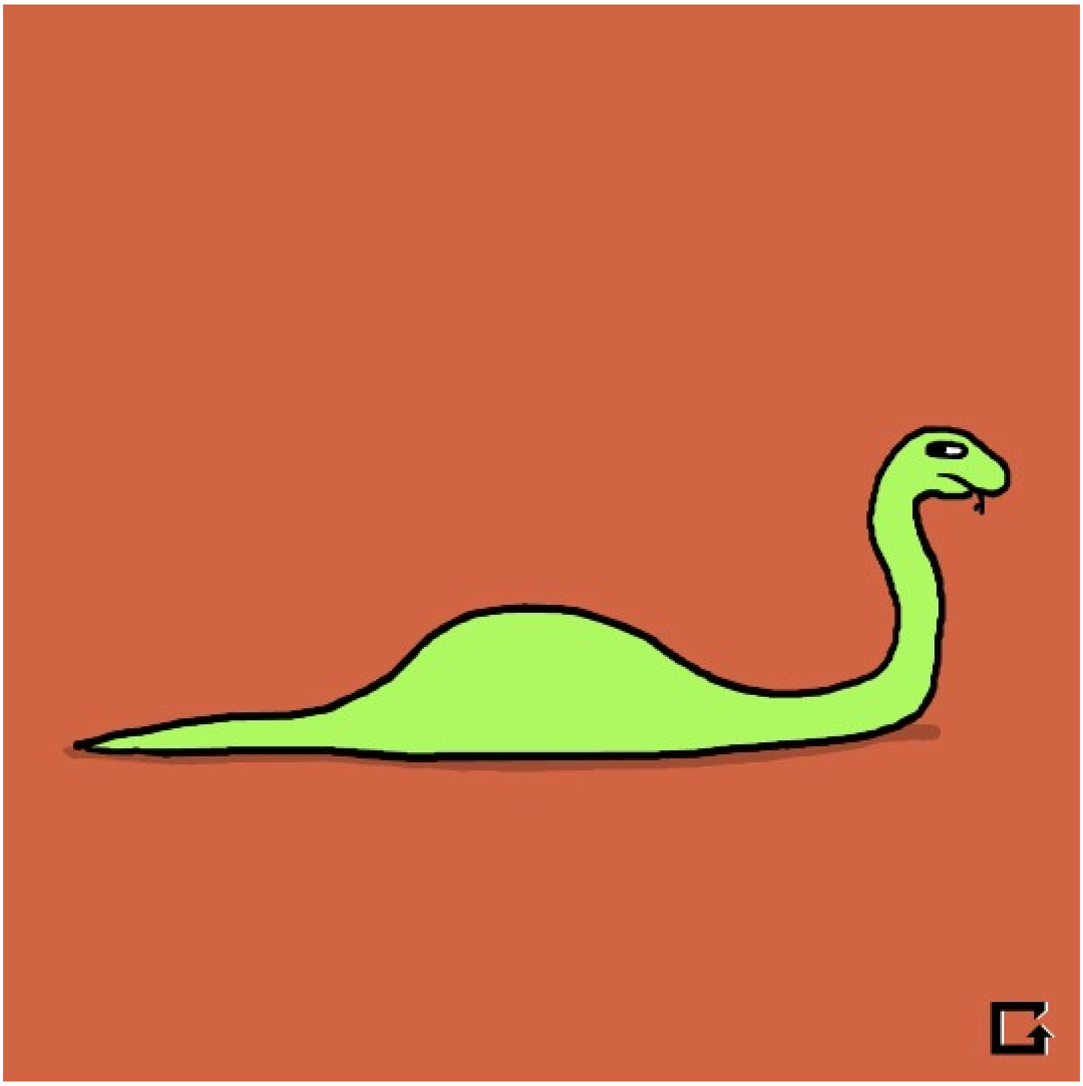
```
          go.Scatter(x = reta_g[0], y = reta_g[1]),
          go.Scatter(x = reta_j[0], y = reta_j[1]),
          ],
    layout=go.Layout(
        xaxis=dict(range=[0, 10], autorange=False),
        yaxis=dict(range=[0, 10], autorange=False),
        title="Animação da translação do quadrado",

updatemenus=[dict(type="buttons",
                          buttons=[dict(label="Play",
                                        method="animate",
                                        args=[None])])]),
    frames=[go.Frame(
        data=[
          go.Scatter(x = reta_h[0] + q[k], y = reta_h[1] + q[k]),
          go.Scatter(x = reta_f[0] + q[k], y = reta_f[1] + q[k]),
          go.Scatter(x = reta_g[0] + q[k], y = reta_g[1] + q[k]),
          go.Scatter(x = reta_j[0] + q[k], y = reta_j[1] + q[k])
          ])

        for k in range(N)]
)


"""
Plotando (Desenhando) grafico definido por fig.
"""

fig.show()
```

Respondendo ao Formulário.

> Bloco com avanço

[Formulario]

OBRIGADO POR PARTICIPAR!!



APÊNDICE B — TUTORIAL (COM AS ATIVIDADES) - GEOGEBRA

Translação de um quadrado no Geogebra

O GeoGebra (aglutinação das palavras Geometria e Álgebra) é um software matemático escrito na linguagem de programação Java, disponível de forma gratuita nas mais diversas plataformas. Criado com o intuito de ser utilizado em sala de aula, nos mais diversos níveis de educação, o programa apresenta, ao mesmo tempo, características geométricas e algébricas de um mesmo objeto, tanto em duas quanto em três dimensões, permitindo alterações dinâmicas nos objetos prontos, sejam eles pontos, retas, planos, vetores, matrizes ou funções que podem ser inseridos através de ferramentas, ou comandos. Isso permite que a ferramenta seja utilizada tanto no estudo de álgebra, quanto geometria e cálculo, auxiliando na visualização e interpretação do que está ocorrendo nos espaços bidimensionais e tridimensionais.

Vamos construir?

Atividade 1: A proposta da atividade é criar um quadrado no Geogebra e transladá-lo. Para isso está sendo disponibilizado esse breve tutorial.

Passo 1: Inicialmente, abra o Geogebra Web, acessando o link: https://www.geogebra.org/classic?lang=pt_PT.

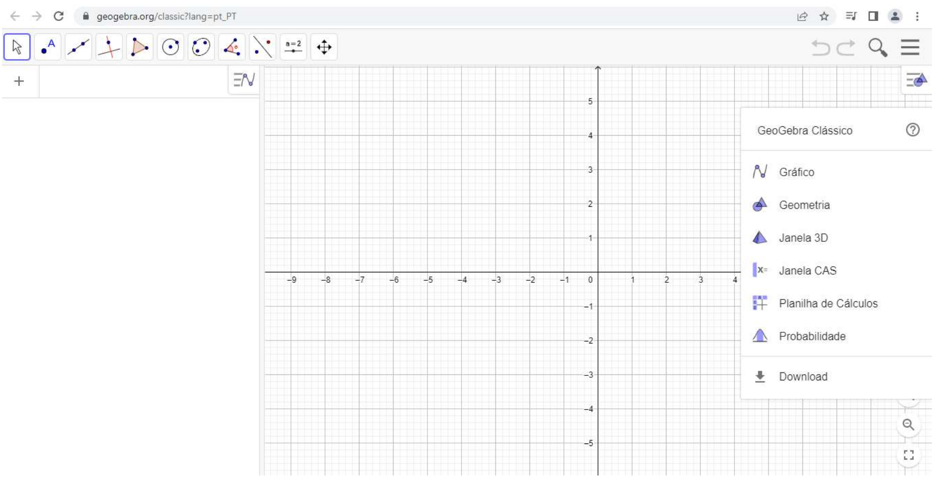


Figura 1: Layout do Geogebra Web.

Fonte: <https://www.geogebra.org/classic?lang=pt_PT>.

Passo 2: Crie um quadrado com as medidas que você desejar, para isso clique no quinto ícone que está na parte superior esquerda da tela (com a figura de um triângulo), selecione polígono e, após, os 4 vértices na malha quadriculada, como indicado na figura abaixo.

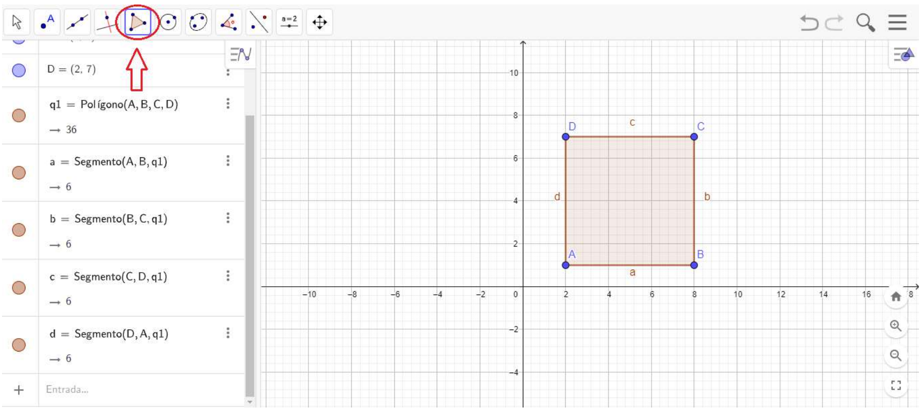


Figura 2: Construindo o quadrado.

Passo 3: Agora, você precisará criar um vetor, para isso clique no terceiro ícone no canto superior esquerdo da tela (com a figura de uma reta) e selecione Vetor.

Figura 3: Aba do vetor.

Passo 4: Com essa opção selecionada clique na malha quadriculada, escolhendo primeiramente a origem do vetor e depois sua extremidade. Cabe destacar que você pode escolher qualquer direção e sentido.

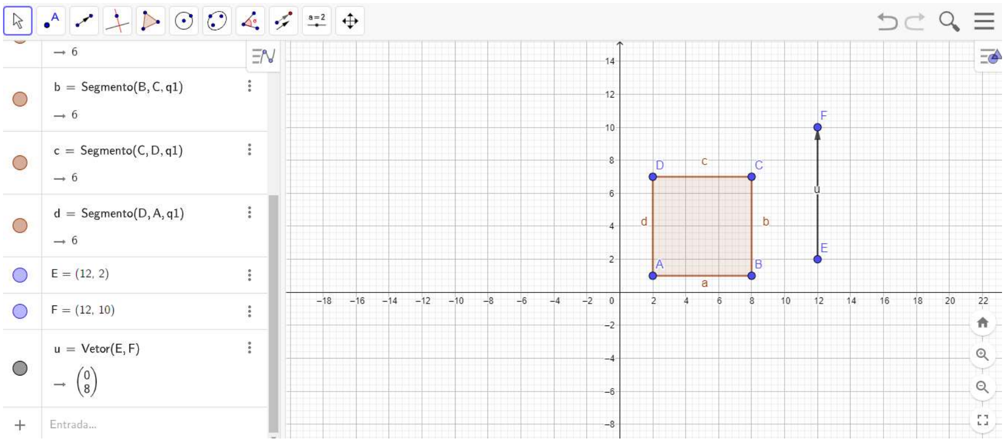


Figura 4: Construção do vetor.

Passo 5: Lembrando que nosso objetivo é realizar a translação dessa figura, selecionamos o nono ícone do canto superior esquerdo da tela (com a figura de uma reta e dois pontos) e clicamos em Translação por um Vetor.

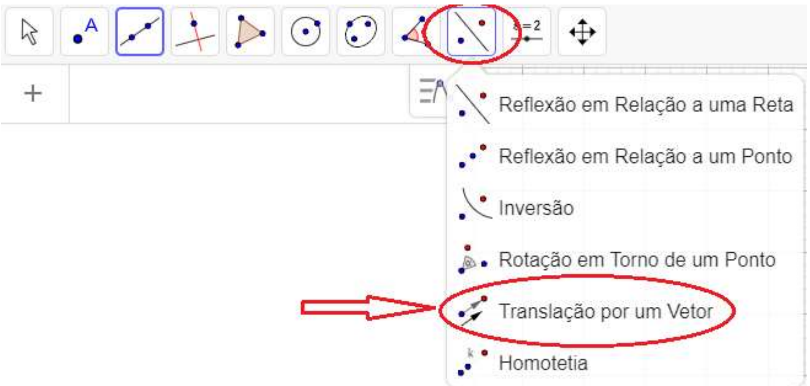


Figura 5: Aba de Translação por um vetor.

Passo 6: Logo após, selecione primeiro o objeto a ser transladado e depois o vetor.

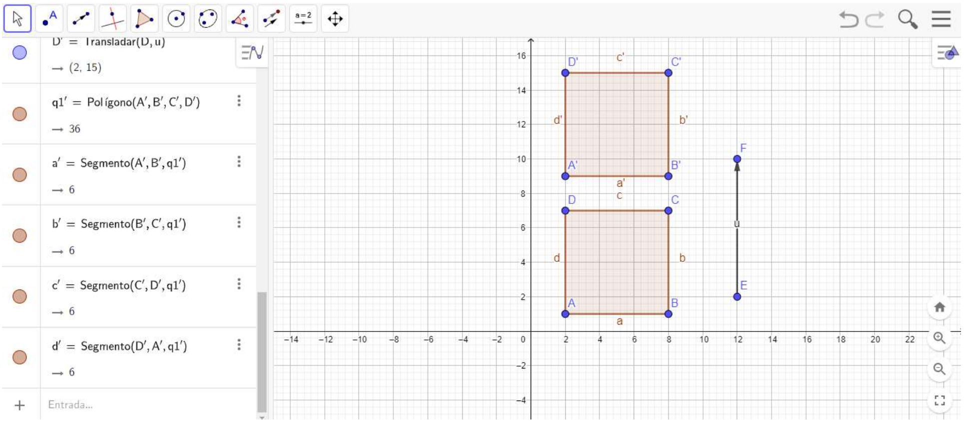


Figura 6: Transformação de translação de um quadrado.

Como resultado temos nosso quadrado original e o nosso quadrado transladado.

Passo 7: Caso você "deseje deixar a imagem mais limpa", pode esconder o nome de cada segmento, clicando no segmento com o botão direito do mouse e selecionando Exibir Rótulo.

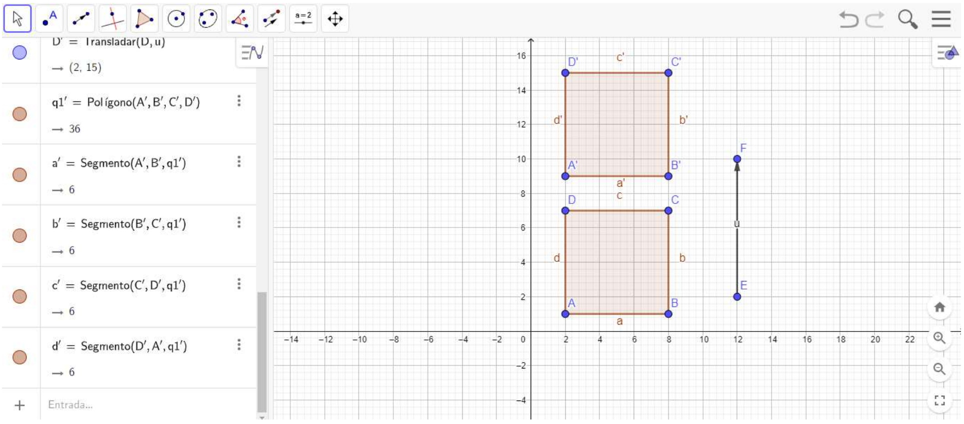


Figura 7: Transformação de translação em um quadrado sem rótulo.

Rotação de um triângulo no Geogebra

Atividade 2: A segunda atividade tem por objetivo construir um triângulo e rotacionar ele em relação a um ponto. Para essa atividade também disponibilizamos um passo a passo.

Passo 1: Inicialmente, abra o Geogebra Web, acessando o link: https://www.geogebra.org/classic?lang=pt_PT.

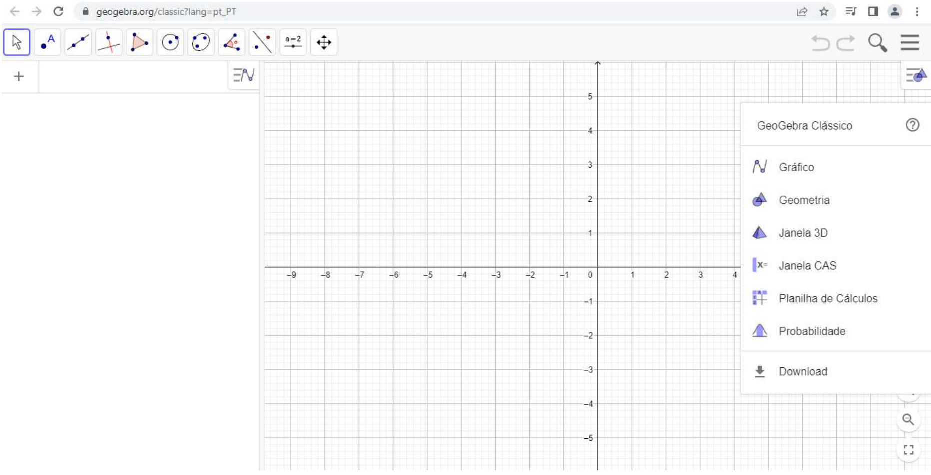


Figura 8: Layout do Geogebra.

Passo 2: Crie um triângulo com as medidas que você desejar, para isso clique no quinto ícone que está na parte superior esquerda da tela (com a figura de um triângulo) e selecione Polígono.

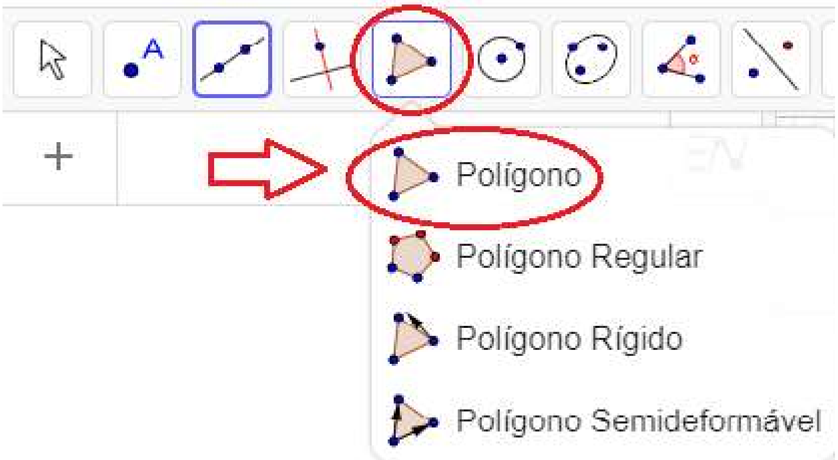


Figura 9: Aba de Polígono.

Passo 3: Escolha os 3 vértices do triângulo na malha quadriculada, como na figura abaixo.

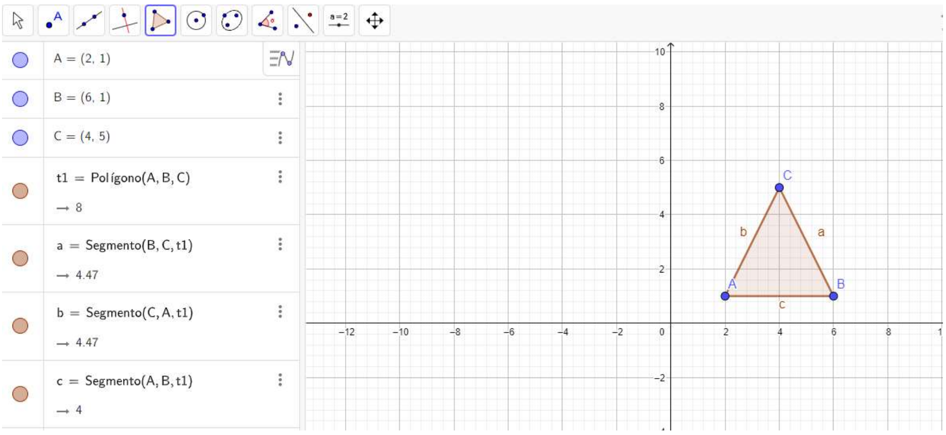


Figura 10: Construção do triângulo.

Passo 4: Agora, precisamos criar um ponto, clicando no segundo ícone do canto superior esquerdo da tela (com a figura de um ponto identificado pela letra A) e selecione a opção Ponto.

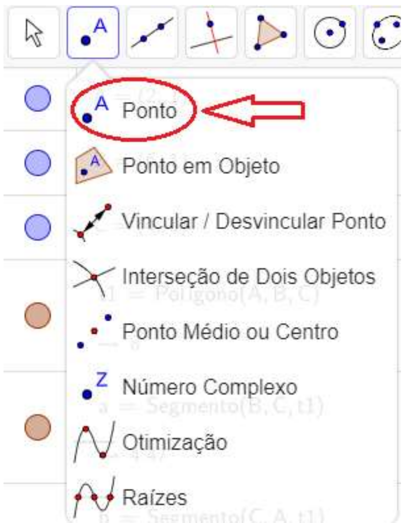


Figura 11: Aba de Ponto.

Passo 5: Crie um ponto na malha quadriculada.

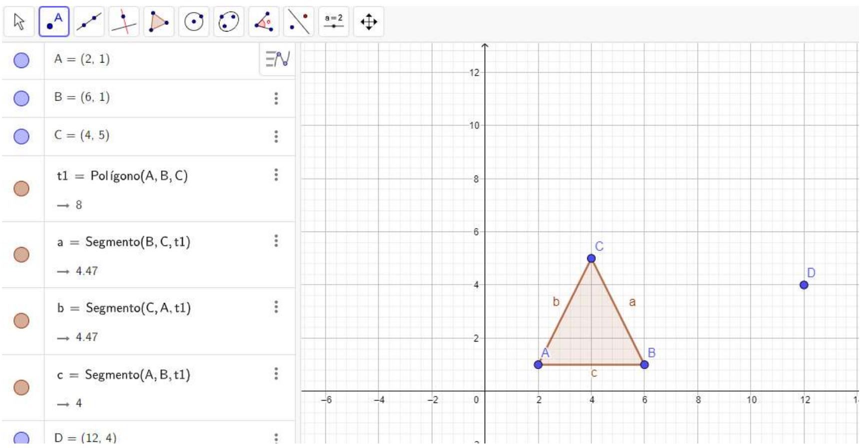


Figura 12: Criando um Ponto.

Passo 6: Clique com o botão direito do mouse em cima do Ponto D, e selecione renomear, a fim de que o ponto criado agora seja chamado: Ponto O.

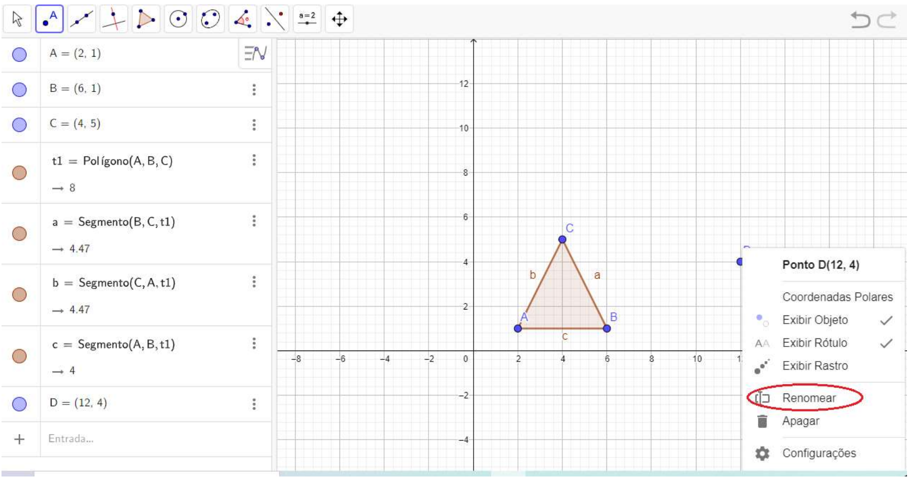


Figura 13: Renomeando o Ponto D.

Passo 7: Lembrando que nosso objetivo é realizar a rotação dessa figura, selecionamos o nono ícone do canto superior esquerdo da tela (com a figura de uma reta e dois pontos) e clicamos em Rotação em Torno de um Ponto.

Figura 14: Aba de Rotação em Torno de um Ponto.

Passo 8: Com a opção Rotação em Torno de um Ponto selecionada, clique primeiro no triângulo e depois no Ponto O. Abrirá uma janela para você escolher o ângulo dessa rotação. Você pode digitar o ângulo de 180° e clicar em OK.

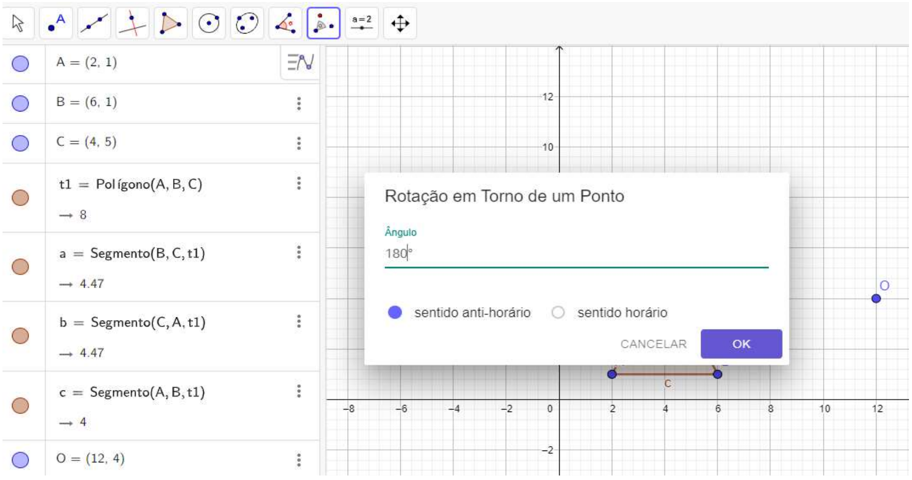


Figura 15: Janela para digitar o ângulo de rotação.

Passo 9: Como resultado temos nosso triângulo original e o nosso triângulo rotacionado.

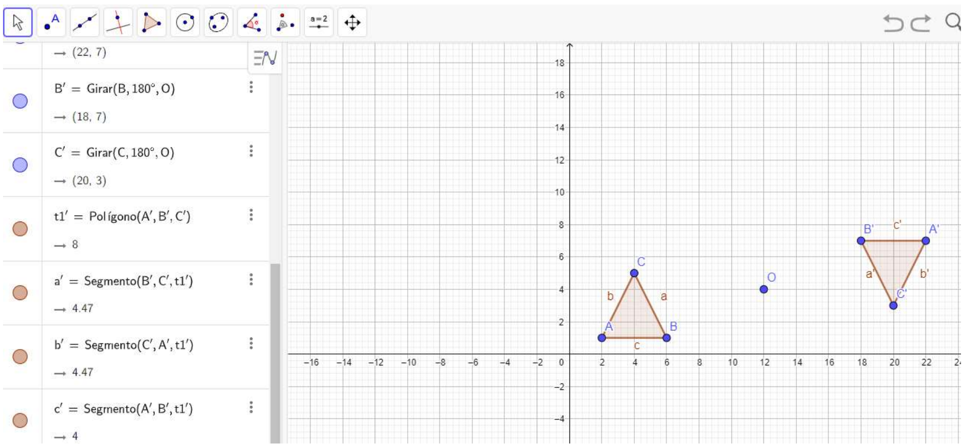


Figura 16: Transformação de rotação em um triângulo.

Passo 10: Caso você “deseje deixar a imagem mais limpa”, pode esconder o nome de cada um dos segmentos, clicando no segmento com o botão direito do mouse e selecionando Exibir Rótulo.

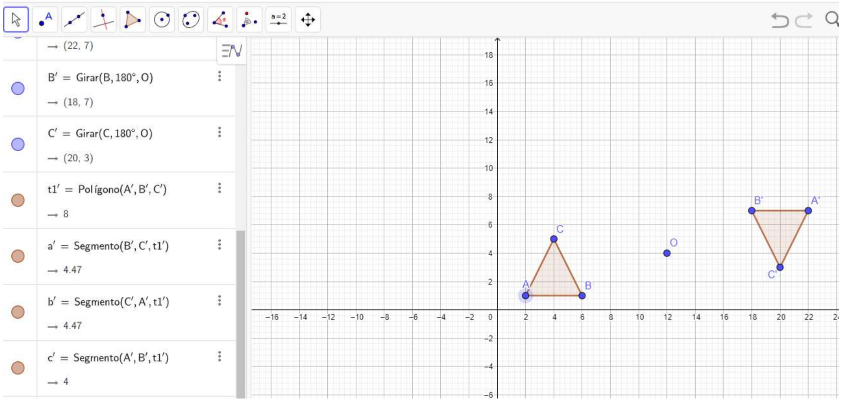


Figura 17: Transformação de rotação em um triângulo sem os rótulos.

APÊNDICE C — TUTORIAL - GOOGLE COLAB

Google Colaboratory

O Google Colaboratory — também conhecido como Google Colab — é uma ferramenta na nuvem que permite criar e executar códigos utilizando a linguagem Python. Com ele, você pode executar instruções diretamente do seu navegador, de forma simples e rápida.

Essa ferramenta oferece um ambiente bastante semelhante ao do software de código aberto Jupyter Notebook, ambos com a praticidade de não necessitar configurações — já que funciona inteiramente online. A escolha do Google Colab, em relação a outras ferramentas, se deve ao fato de possuir funcionalidades que podem auxiliar em uma melhor apresentação/desenvolvimento de atividades, no nosso caso particular, por exemplo, a integração com a linguagem de marcação LaTeX.

Como acessamos e utilizamos essa ferramenta? Para isso criamos um breve tutorial.

Passo 1: Inicialmente, você receberá por e-mail um convite para acessar o Google Colab, contendo instruções de como resolver as atividades propostas. Abrindo a página, você verá o início da aplicação intitulada: Aplicação de TCC, como na figura abaixo:

Figura 1: Visualização da Aplicação de TCC no Google Colab.

Passo 2: Dessa forma, você **deve ir** até **Ficheiro** e clicar em: **Guardar uma cópia no Drive.**

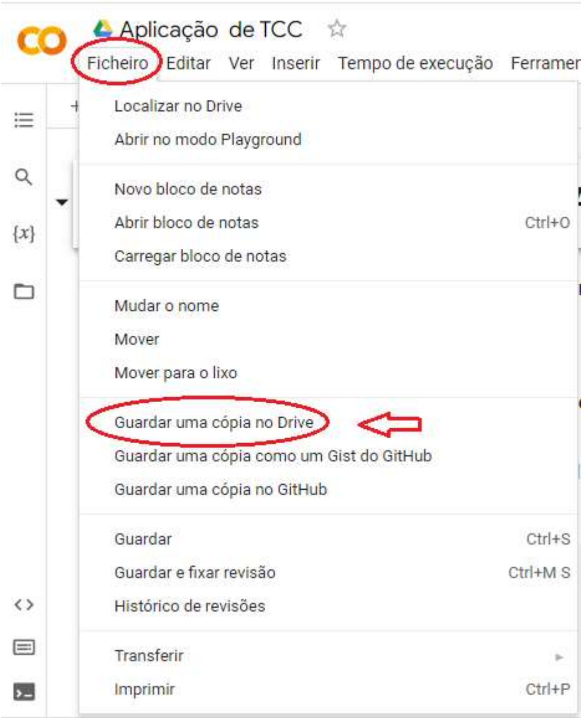


Figura 2: Aba Ficheiro no Google Colab, indicando a opção: Guardar uma cópia no Drive.

Passo 3: Você irá desenvolver a atividade na cópia salva no Drive.

Passo 4: Antes de iniciar a atividade, compartilhe com a autora desse trabalho. Para isso, clique em **Partilhar** e digite tatianebonorino.aluno@unipampa.edu.br. Como na figura abaixo:

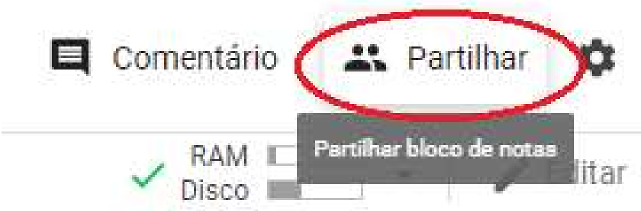


Figura 3: Aba de partilhar.

APÊNDICE D — QUESTIONÁRIO PROPOSTO AOS ACADÊMICOS

Pesquisa: "Transformações Geométricas Afins: Uma abordagem utilizando Geogebra e Python"

Após a realização de algumas atividades propostas como parte do Trabalho de Conclusão de Curso da acadêmica Tatiane Zago Bonorino, solicitamos que preencha o formulário abaixo no intuito de:

1) Validar a ferramenta desenvolvida pela acadêmica utilizando a linguagem Python.
2) Comparar os recursos tecnológicos utilizados para o desenvolvimento das atividades.

OBS: Tempo médio de resposta do formulário: 9 minutos.

---

*Obrigatório

1. Ao participar desta pesquisa, autorizo que os dados coletados sejam utilizados, preservando a identidade dos participantes, no Trabalho de Conclusão de Curso da acadêmica Tatiane Zago Bonorino. *

Marcar apenas uma oval.

- ( ) Concordo em participar da pesquisa
- ( ) Não concordo em participar da pesquisa

Formulário de avaliação das atividades

2. 1- Qual sequência de atividades você considera mais interativa? *

Marcar apenas uma oval.

- ( ) Sequência de atividades realizadas no Geogebra.
- ( ) Sequência de atividades realizadas no Python.
- ( ) Nenhuma das alterativas anteriores.

3. 2- O breve tutorial realizado para auxiliar nas atividades utilizando o Geogebra possui informações detalhadas suficientes para a execução das atividades propostas? *

Marcar apenas uma oval.

- Sim.
- Não.
- Indiferente.

4. 3- As instruções apresentadas, tanto no breve tutorial quanto durante a sequência de atividades no Google Colab, possuem informações detalhadas suficientes para realização das atividades propostas? *

Marcar apenas uma oval.

- Sim.
- Não.
- Indiferente.

5. 4- Quanto à afirmação: "Eu usaria o roteiro de atividade propostas no GeoGebra em uma possível aula sobre o tema." *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

6. 5- Quanto à afirmação: "Eu usaria o roteiro de atividade propostas (na aplicação desenvolvida em Python) no Google Colab em uma possível aula sobre o tema." *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

7. 6- Quanto à afirmação: "O roteiro das atividades desenvolvidas no Google Colab foram simples e de fácil compreensão." *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

8. 7- Quanto à afirmação: "Eu considero as atividades desenvolvidas no GeoGebra mais fácil de realizar se comparado a sequência de atividades desenvolvidas no Google Colab." *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

9. 8- Quanto à afirmação: "Eu considero a sequência de atividades desenvolvida no Google Colab mais fácil de realizar se comparado às atividades desenvolvidas no GeoGebra." *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

10. 9- Quanto à afirmação: "A sequência de atividades desenvolvidas em ambos os casos são igualmente simples e de fácil compreensão". *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

11. 10- Quanto à afirmação: "As definições/ilustrações que discutiam/apresentavam conhecimentos matemáticos no Google Colab auxiliaram na compreensão da atividade." *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

12. 11- Quanto à afirmação: "As instruções para a execução de comandos e os locais para entrada de dados, foram fáceis de serem identificados/localizados dentro da aplicação." *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

13. 12- Quanto à afirmação: " Considero a fonte, cores e layout apresentados na aplicação como adequados." *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

14. 13- Quanto à afirmação: "Após a realização dessa atividade você consideraria criar sua própria atividade (ou mesmo procurar outras atividades previamente existentes) em Pyhton no Google Colab a fim de ensinar/motivar seus alunos em suas aulas." *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

15. 14- Quanto à afirmação: "Os resultados alcançados em cada uma das atividades realizadas no Google Colab correspondem ao esperado." *

Marcar apenas uma oval.

- Discordo Totalmente.
- Discordo Parcialmente.
- Indiferente.
- Concordo Parcialmente.
- Concordo Totalmente.

16. 15- Sobre a sua percepção/satisfação em relação à aplicação desenvolvida no Google Colab, assinale os aspectos que considerou como positivo: *

Marque todas que se aplicam.

- [ ] A interação com as atividades.
- [ ] A possibilidade de inserir instruções/texto/conteúdos explicativos.
- [ ] A possibilidade de colocar links, imagens e gifs.
- [ ] A possibilidade de explorar gráficos e animações.
- [ ] A possibilidade de integração com LaTeX, melhorando a apresentação visual de fórmulas e textos matemáticos.
- [ ] A facilidade/rapidez com que se pode testar muitos exemplos sem a necessidade de refazer contas ou gráficos.
- [ ] O código poder ser editado pelo usuário a fim de corrigi-lo/melhorá-lo, caso julgar necessário.
- [ ] Não ter necessidade de baixar e/ou instalar qualquer software/biblioteca na minha máquina pessoal para participação na atividade.

17. 16- Sobre a sua percepção/satisfação em relação à aplicação desenvolvida no Google Colab, assinale os aspectos que considerou como negativo: *

Marque todas que se aplicam.

- [ ] A interação com as atividades.
- [ ] A possibilidade de inserir instruções/texto/conteúdos explicativos.
- [ ] A possibilidade de colocar links, imagens e gifs.
- [ ] A falta de tratamento e prevenção de erros na construção da aplicação.
- [ ] A possibilidade de integração com LaTeX, melhorando a apresentação visual de fórmulas e textos matemáticos.
- [ ] A facilidade/rapidez com que se pode testar muitos exemplos sem a necessidade de refazer contas ou gráficos.
- [ ] A possibilidade do usuário modificar acidentalmente as caixas de comando, impossibilitando a execução da aplicação.
- [ ] Não ter necessidade de baixar e/ou instalar qualquer software/biblioteca na minha máquina pessoal para participação na atividade.

18. 17- Qual seu nível de compreensão/familiaridade/conhecimento de Python? *

Marcar apenas uma oval.

- ( ) Avançado.
- ( ) Intermediário.
- ( ) Iniciante.
- ( ) Nenhum conhecimento.

19. 18- Deixe aqui seu comentário/dúvida/crítica/elogio sobre a aplicação desenvolvida em Python.

20. 19- Caso deseje, sinta-se à vontade para deixar aqui contribuições que acharia pertinente serem implementadas no código da aplicação ou mesmo seu endereço de e-mail para que possamos entrar em contato, agendar algum horário e discutir possíveis contribuições.

ANEXO A — EXEMPLOS DE TRANSFORMAÇÕES GEOMÉTRICAS AO LONGO DA HISTÓRIA

Figura 30 – Pintura Rupestre de El Buey na Bolívia

Fonte: Transformações Geométricas.
Disponível em: <http://www.mat.uc.pt/~mat0829/Transformacoesgeometricas-2.pdf>.

Figura 31 – Tapete Pazyryk - A mais antiga peça de tapeçaria do mundo

Fonte: Transformações Geométricas.
Disponível em: <http://www.mat.uc.pt/~mat0829/Transformacoesgeometricas-2.pdf>.

Figura 32 – Cerâmica Chinesa - 3.000 a. C

Fonte: Transformações Geométricas.
Disponível em: <http://www.mat.uc.pt/~mat0829/Transformacoesgeometricas-2.pdf>.

Figura 33 – Cerâmica Marajoara - Brasil

Fonte: Transformações Geométricas.
Disponível em: <http://www.mat.uc.pt/~mat0829/Transformacoesgeometricas-2.pdf>.

Figura 34 – Simetrias presentes na Natureza

Fonte: Transformações Geométricas.
Disponível em: <http://www.mat.uc.pt/~mat0829/Transformacoesgeometricas-2.pdf>.

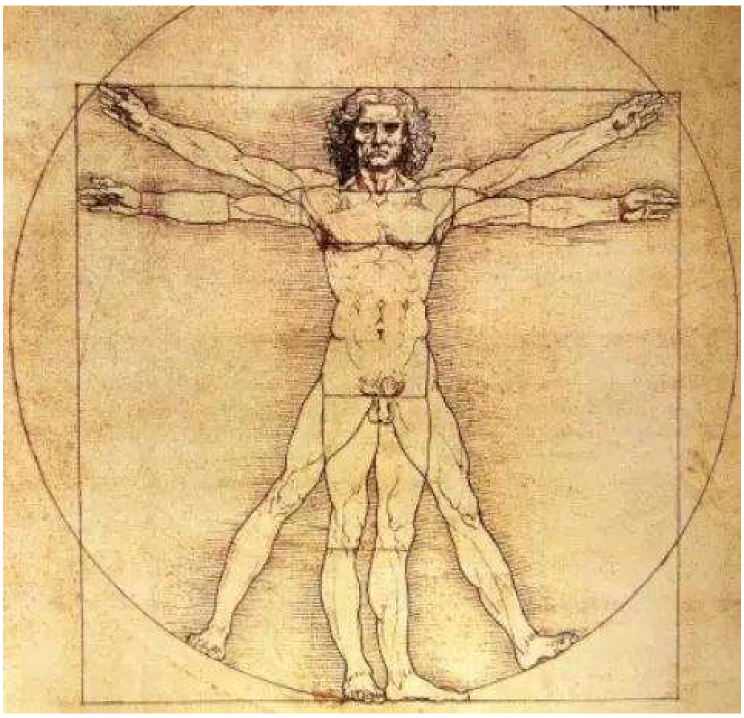
Figura 35 – Simetrias presentes na Natureza

Fonte: Transformações Geométricas.
Disponível em: <http://www.mat.uc.pt/~mat0829/Transformacoesgeometricas-2.pdf>.

Figura 36 – Desenho em tecidos

Fonte: Design de superfície têxtil.
Disponível em: <https://www.redalyc.org/pdf/5140/514054175006.pdf>

Figura 37 – O homem Vitruviano - Leonardo Da Vinci



Fonte: Imagens Google.