UNIVERSIDADE FEDERAL DO VALE DO SÃO FRANCISCO

MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL – PROFMAT

FRANCISCO ALVES DOS SANTOS

APLICAÇÃO DE VETORES À COMPUTAÇÃO GRÁFICA: Um estudo de caso

JUAZEIRO-BA

2018

FRANCISCO ALVES DOS SANTOS

APLICAÇÃO DE VETORES À COMPUTAÇÃO GRÁFICA: Um estudo de caso

Trabalho apresentado a Universidade Federal do Vale do São Francisco-UNIVASF, Campus Juazeiro, como requisito para a obtenção do título de Mestre em matemática.

Orientador: Prof. Dr. Alexandre Ramalho Silva

JUAZEIRO-BA

2018

Santos, Francisco Alves

S237a APLICAÇÃO DE VETORES À COMPUTAÇÃO GRÁFICA: Um estudo de caso / Francisco Alves dos Santos. – Juazeiro - BA, 2018.

xii, 84f.: il. ;29 cm.

Dissertação (Mestrado Profissional em Matemática em Rede Nacional - Profmat) - Universidade Federal do Vale do São Francisco, Campus, Juazeiro - BA, 2018.

Orientador: Prof. Dr. Alexandre Ramalho Silva

1. Vetores. 2. Programação. I. Título. II Silva, Alexandre Ramalho. III Universidade Federal do Vale do São Francisco

CDD: 515.63

Ficha catalográfica elaborada pelo Sistema Integrado de Biblioteca SIBI/UNIVASF
Bibliotecária: Ana Paula Lopes da Silva CRB/3: 1269

UNIVERSIDADE FEDERAL DO VALE DO SÃO FRANCISCO
MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL - PROFMAT

FOLHA DE APROVAÇÃO

Francisco Alves dos Santos

APLICAÇÃO DE VETORES À COMPUTAÇÃO GRÁFICA: Um estudo de caso

Dissertação apresentada como requisito parcial para obtenção do título de Mestre em Matemática, pela Universidade Federal do Vale do São Francisco.

Aprovada em: 26 de outubro de 2018.

Banca Examinadora

Prof. Dr. Alexandre Ramalho Silva, PROFMAT/UNIVASF

Prof. Dr. Lino Marcos da Silva, PROFMAT/UNIVASF

Prof. Dr. Jorge Luis Cavalcanti Ramos, CECOMP/UNIVASF

AGRADECIMENTOS

Agradeço primeiramente ao meu Deus, sem o qual nenhuma conquista seria possível.

Aos meus familiares, especialmente aos meus pais Antônio Francisco e Joana Maria, e à minha querida esposa Mara Dalila Moura, que sempre me apoiaram e me incentivaram a dar prosseguimento nos meus projetos de vida, em especial, esse.

Aos professores que compõem o quadro do profmat, na Univasf campus Juazeiro, pelos conhecimentos transmitidos.

Ao professor Alexandre Ramalho Silva por ter orientado e ajudado na construção desse trabalho.

De maneira não menos especial, ao um grande amigo, matemático e programador, professor Olávio Arruda, por ter me ajudado a compreender o uso dos vetores para dar movimentos a seres em ambientes virtuais, e que me auxiliou na realização desse projeto.

E por último, agradeço de maneira grandemente especial, aos meus amados irmãos em Cristo Jesus, que oraram a Deus por mim, pela concretização desse sonho.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001

RESUMO

O presente trabalho descreve a aplicação e as conclusões obtidas da aplicação de uma oficina sobre vetores aplicados à computação gráfica. A mesma foi desenvolvida em uma turma do curso de licenciatura em matemática de uma instituição de ensino da rede federal da cidade de São Raimundo Nonato-PI, que já havia cursado a disciplina de geometria analítica. Na oficina, foi mostrada a aplicação de vetores para simular movimentos de objetos em um ambiente virtual programa *processing 3.3.5* (gratuito para *download*). Os objetivos dessa pesquisa são investigar a satisfação e o desempenho dos discentes durante a aplicação da oficina, e verificar qual melhoria no aprendizado dos mesmos, no estudo da geometria analítica. A metodologia usada traz em sua composição, uma avaliação diagnóstica, uma avaliação intermediária, e um questionário final, com perguntas abertas com intuito de verificar a compreensão da aplicação de vetores em computação gráfica por parte dos alunos, bem como se concordam que a contextualização é importante no processo de ensino-aprendizagem. Pode-se afirmar que os alunos compreenderam como as operações com vetores são aplicadas na programação, em especial em movimentos de objetos em ambientes virtuais e demonstraram interesse em aprofundar seus conhecimentos em geometria analítica. Concluindo, podemos afirmar que a contextualização dos conteúdos matemáticos e sua aplicação, por parte do docente, favorecem, considerável e positivamente, o processo ensino-aprendizagem.

Palavras chave: vetor, movimentos, ambiente virtual, programação, ensino da matemática.

ABSTRACT

This work describes the application and conclusions obtained from the application of a workshop on vectors applied to computer graphics. The workshop was developed for a group of mathematics undergraduate students of a federal educational in São Raimundo Nonato city (PI). These students had already studied analytic geometry. In the workshop, it was shown the application of vectors to simulate object movements in a virtual environment program processing 3.3.5 (free to download). The objectives of this research are investigate students' satisfaction, analyse their performance during the application of the workshop and verify the improvement in their learning in the study of analytical geometry. The methodology is based on a diagnostic evaluation, an intermediate evaluation, and a final questionnaire, with objetive questions in order to verify the students' understanding of vectors application in computer graphics, as well as if they agree that contextualization is important in the teaching-learning process. It can be said that the students obtained a satisfactory understanding of the role that the operations with vectors perform in the programming, in what refers to movements to objects in virtual environments and showed interest in deepening their knowledge in analytical geometry. In conclusion, we can affirm that the contextualization of the mathematical contents and their application, by the teacher, favor, considerably and positively, the teaching-learning process.

Keywords: Vector, movements, virtual environment, programming, Mathematics Teaching.

LISTA DE FIGURAS

Figura 1. Segmento orientado.....pág. 22
Figura 2. Segmento orientados AB e ED opostos.....pág. 22
Figura 3. Vetores representantes de $\overrightarrow{AB}$.....pág. 23
Figura 4. EF ≡ OP, Exemplo 2.....pág. 24
Figura 5. Representação gráfica de um vetor no plano.....pág. 24
Figura 6. Soma de vetores.....pág. 25
Figura 7. Soma de vetores no plano.....pág. 26
Figura 8. Soma de vetores: regra do polígono.....pág. 26
Figura 9. Soma de vetores: regra do paralelogramo.....pág. 26
Figura 10. Diferença de vetores.....pág. 28
Figura 11. Multiplicação de um vetor por um escalar.....pág. 29
Figura 12. Módulo ou comprimento de um vetor.....pág. 31
Figura 13. Representação de um vetor no espaço.....pág. 31
Figura 14. Objeto se movimentando na tela segundo uma velocidade atribuída a ele.....pág. 35
Figura 15. Bola de salto com vetores.....pág. 39
Figura 16. Cálculo da magnitude de um vetor.....pág. 41
Figura 17. Cálculo da magnitude de um vetor 2.....pág. 41
Figura 18. Magnitude de um vetor.....pág. 42
Figura 19. Magnitude de um vetor representado por retângulo de altura fixa.....pág. 44
Figura 20. Vetor normalizado.....pág. 45
Figura 21. Bola em movimento segundo uma velocidade atribuída a ela.....pág. 48
Figura 22. Bola se movimentando sob efeito de uma aceleração.....pág. 50
Figura 23. Seguindo o mouse.....pág. 52
Figura 24. Objeto em movimento sob efeitos do vento e da gravidade.....pág. 56
Figura 25. Gravidades iguais aplicadas a vários objetos.....pág. 59
Figura 26. Gravidades diferentes aplicadas a vários objetos.....pág. 60
Figura 27. Atrito entre um corpo em movimento e sua superfície de apoio.....pág. 61
Figura 28. Atrito aplicado a objetos em movimento.....pág. 63

LISTA DE CÓDIGOS

Código 1. Bola se movimentando sem o uso de vetores.....pág. 35
Código 2. Bola se movimentando com o uso de vetores.....pág. 39
Código 3. Comprimento de um vetor.....pág. 40
Código 4. Comprimento fixo de um veto.....pág. 42
Código 5. Comprimento de um vetor representado por um retângulo.....pág. 43
Código 6. Normalização de um vetor.....pág. 45
Código 7. Movimento com velocidade.....pág. 47
Código 8. Movimento com aceleração.....pág. 49
Código 9. Objeto seguindo o mouse.....pág. 52
Código 10. Vento e gravidade aplicados ao objeto.....pág. 55
Código 11. Gravidades iguais aplicadas a objetos diferentes.....pág. 58
Código 12. Gravidades aplicadas a vários objetos variando de acordo com sua massa.....pág. 60
Código 13. Atrito.....pág. 62

LISTA DE GRÁFICOS

Gráfico 1. Resposta dos discentes ao questionário sobre aplicação de vetores.....pág. 69

Gráfico 2. Reposta dos discentes ao questionário sobre a relevância do estudo dos vetores no curso de licenciatura em matemática.....pág. 70

Gráfico 3. Resposta dos discentes quando questionados se o professor da disciplina geometria analítica mostrou onde eles aplicariam vetores.....pág. 71

Gráfico 4. Reposta dos discentes à pergunta voltada à manipulação dos comandos que fazem o objeto se mover pela tela.....pág. 73

SUMÁRIO

1 INTRODUÇÃO.....pág. 13
2 FUNDAMENTAÇÃO TEÓRICA.....pág. 16

2.1 DIFICULDADES NO PROCESSO ENSINO-APRENDIZAGEM, COMO SANÁ-LAS? .....pág. 16

2.2 VETORES.....pág. 21
2.2.1 Breve contexto histórico.....pág. 21
2.2.2 Segmentos Orientados.....pág. 21
2.2.3 Vetores no plano.....pág. 23
2.3 OPERAÇÕES COM VETORES.....pág. 25
2.3.1 Soma de vetores.....pág. 25
2.3.2 Diferença de vetores.....pág. 28
2.3.3 Multiplicação de um vetor por um escalar.....pág. 29
2.3.4 Módulo de um vetor.....pág. 30
2.4 VETORES NO ESPAÇO.....pág. 31
2.4.1 Soma de vetores e multiplicação de vetores por um escalar.....pág. 32
2.5 COMO A APLICAÇÃO DOS VETORES PODE DAR MOVIMENTOS AOS SERES EM AMBIENTE VIRTUAL.....pág. 33
2.5.1 Bola de salto quicando sem vetores.....pág. 34
2.5.2 Bola de saltar com vetores (Pvectors).....pág. 38
2.5.3 Magnitude de um vetor.....pág. 40
2.6 VETORES NORMALIZADOS.....pág. 44
2.6.1 Aplicação da velocidade ao objeto.....pág. 45
2.6.2 Aplicação da aceleração ao objeto.....pág. 48
2.6.3 Objeto seguindo o mouse.....pág. 50
2.7 FORÇAS E LEIS DE NEWTON.....pág. 53
2.7.1 Gravidade aplicada a um objeto.....pág. 54
2.7.2 Gravidade aplicada a mais de um objeto.....pág. 57
2.7.3 Gravidade modelando uma força.....pág. 59
2.7.4 Atrito.....pág. 61
3 METODOLOGIA.....pág. 64

4 RESULTADOS E DISCUSSÕES.....pág. 68
5 CONSIDERAÇÕES FINAIS.....pág. 75
6 REFERÊNCIAS BIBLIOGRÁFICAS.....pág. 77
7 APÊNDICES.....pág. 80

1. INTRODUÇÃO

O estudo sobre vetores ou grandezas vetoriais, é introduzido no ensino por meio das disciplinas de exatas desde o início do ensino médio, quando se estuda, na física, sobre velocidade, aceleração e forças, por exemplo. Para aqueles que optam por um curso superior na área de exatas, tais como engenharias, licenciaturas ou bacharelados em matemática, esse estudo é mais aprofundado nas disciplinas de álgebra linear e geometria analítica, importantes componentes curriculares do referido curso. A geometria analítica, por sua vez, traz estudos mais aprofundados sobre o conteúdo em questão.

Esse trabalho relata os resultados de uma pesquisa sobre aplicação de vetores geométricos no campo da computação gráfica. É uma introdução à lógica de programação usada nos jogos de computadores para dar movimentos a objetos em um ambiente virtual, por meio de comandos que envolvem operações com vetores, entre outras.

Foi, ao participar de um minicurso sobre a aplicação de vetores geométricos em jogos de computadores em um evento de matemática (III Encomat), na instituição da qual sou docente, que surgiu a ideia de trabalhar, em forma de minicurso, com os alunos da mesma instituição de ensino, que já haviam cursado a disciplina geometria analítica, a aplicação de tal componente matemático, na computação gráfica. Isso se deve à busca por métodos de ensino que sejam capazes de despertar no discente, o interesse pelo aprender matemática, e tornam a aprendizagem mais dinâmica e prazerosa.

As experiências adquiridas como professor de matemática nos ensinos fundamental, médio e superior, desde 2002 até a presente data, me permitiram observar que muitas metodologias utilizadas no ensino da matemática são ineficazes e levam a um distanciamento, por parte do aluno, da disciplina em questão. Para dar fim, ou diminuir às consequências desse “mito” de que aprender matemática é para poucos, vem se pensando, ao longo dos anos, em métodos que façam com que o aluno passe de sujeito passivo a sujeito ativo no processo ensino-aprendizagem.

O aluno será capaz de aprender, quando for capaz de construir seu conhecimento, assimilando e associando aquilo que aprende com o mundo a sua

volta, com seu cotidiano. Não obstante, trabalhar a aplicação da matemática em outros componentes curriculares de maneira interdisciplinar, mostra ao aluno que essa disciplina tem um vasto campo de aplicações práticas. Mostra-se ainda que a matemática enquanto conhecimento, ultrapassa o campo da abstração, contrariando o que muitos discentes julgam acerca disso, por não conhecerem a relação entre a matemática e as outras áreas do conhecimento. Em síntese, mostrar essa relação é uma maneira de dinamizar o ensino, e isso colabora com o aprendizado do aluno. Os PCNs acrescentam que

> O professor, considerando a multiplicidade de conhecimentos em jogo nas diferentes situações, pode tomar decisões a respeito de suas intervenções e da maneira como tratará os temas, de forma a propiciar aos alunos uma abordagem mais significativa e contextualizada. (BRASIL 1997, pág. 44)

O presente trabalho traz como proposta, mostrar a aplicação de vetores geométricos na computação gráfica. Em suma, de forma não aprofundada, é trabalhada a aplicação de vetores para dar movimento a objetos em um ambiente virtual. Para isso, se faz necessário conhecer algumas propriedades dos vetores e trabalhar algumas operações que envolvem os mesmos.

Quando comecei a ministrar aulas de matemática na instituição de ensino da qual hoje sou professor, no ensino superior, pude ver o quão grande é a deficiência dos discentes em conteúdos matemáticos básicos. Pude também presenciar o desestímulo, por parte de alguns discentes, em estudar algumas disciplinas da grade curricular, por não conhecerem sua aplicação prática, ou que relação há entre os conteúdos estudados e seu dia a dia.

Uma das disciplinas que mostra-se bastante distante de sua aplicação prática quando é lecionada, é a geometria analítica. Os vetores, por sua vez, apesar de terem aplicações em uma infinidade de campos, na maior parte das vezes é trabalhado fora do contexto.

Considerando essas premissas, elaboramos uma oficina que mostra a aplicação de operações entre vetores à computação gráfica.

Durante a aplicação desse projeto, os estudantes envolvidos foram submetidos a avalições e questionários, que foram usadas para coleta dos dados, e que auxiliaram no resultado pesquisa.

O objetivo geral deste trabalho é analisar o desempenho e satisfação do aluno em uma oficina em que conceitos básicos sobre vetores e geometria analítica são trabalhados usando um software de computador, onde são simulados movimentos de objetos em ambientes virtuais por meio de operações com vetores.

São objetivos específicos: investigar o desempenho dos alunos durante a realização da oficina supra citada, acerca da aplicação de vetores, quando usados para introduzir movimentos a objetos dentro de um ambiente virtual e; verificar qual a melhoria no aprendizado do aluno, dos conceitos de geometria analítica após a realização da oficina.

A presente pesquisa traz em sua composição, além dessa introdução, um referencial teórico que a embasa, por meio da visão de alguns autores, e traz vários exemplos sobre o uso dos vetores geométricos por meio das operações básicas de adição, subtração, normalização e multiplicação de um vetor por escalar, seguidos de explicações que visam simplificar a compreensão do leitor quanto ao uso dos vetores para tal fim.

Na sequência, trazemos a metodologia da pesquisa de forma detalhada, mostrando os passos de desenvolvimento desse trabalho.

Em seguida os resultados e as discussões baseados na coleta dos dados obtidos durante o desenvolvimento do projeto; e

Por fim, são apresentadas algumas considerações finais, que buscam comparar a questão norteadora, com os resultados obtidos e analisados na disquisição.

2 FUNDAMENTAÇÃO TEÓRICA

Neste tópico do trabalho, será apresentado o referencial teórico desta pesquisa, trazendo a visão de alguns autores que embasaram o desenvolvimento da mesma, e exemplos que tratam do uso de vetores para dar movimentos a objetos dentro de um ambiente virtual.

2.1 DIFICULDADES NO PROCESSO ENSINO-APRENDIZAGEM, COMO SANÁ-LAS?

Levando em consideração o crescente desinteresse dos alunos das escolas do tempo presente, e os fatores que levam a isso, vem se pensando em métodos e técnicas que despertem neles o prazer pela aprendizagem da matemática. Essa falta de interesse, por parte dos alunos, associada à falta de métodos dinâmicos no ensino da disciplina em questão, tem levado à deficiência no ensino dessa disciplina e tem criado cada vez mais, lacunas nesse processo tão importante na relação aluno-escola-professor.

De acordo com os Parâmetros Curriculares Nacionais - PCNs (BRASIL, 1997, P. 21) “mesmo os alunos que conseguem completar os nove anos do ensino fundamental acabam dispondo de menos conhecimento do que se espera de quem concluiu a escolaridade obrigatória “. Os PCNs (pág. 10) definem escolaridade obrigatória como sendo os nove anos do ensino fundamental.

Ao final da escolaridade obrigatória, os alunos aprenderam pouco, e muitas vezes o que aprenderam não facilita sua inserção e atuação na sociedade. Dentre outras deficiências do processo de ensino e aprendizagem, são relevantes o desinteresse geral pelo trabalho escolar, a motivação dos alunos centrada apenas na obtenção de notas e na promoção, o esquecimento precoce dos conteúdos estudados e os problemas de indisciplina. Pensando em mudar essa visão, são muitas as maneiras de se dinamizar o ensino da Matemática, como por exemplo, a utilização de jogos como ferramenta de auxílio ao ensino da disciplina em questão, e sua relação com o dia a dia do aluno.

O panorama atual requer propostas desafiadoras e um tanto atrativas para os alunos do Ensinos Fundamental e Médio, e mesmo, para os alunos do ensino superior. O trabalho escolar deve ser pensado e elaborado de acordo com a necessidade do corpo discente, tendo em vista alargar o seu leque de conhecimento da disciplina matemática por meio da sua aplicação prática, pois não é prazeroso aprender aquilo que não tem “utilidade”. O aluno necessita conhecer a profundidade daquilo que está aprendendo, e saber em que momento da vida esse conhecimento lhe será útil. Ele precisa saber que o aprender matemática não consiste em memorização de regras e fórmulas que serão usados em determinados momentos e depois serão esquecidos. A esse respeito os PCNs acrescentam que:

> Os Conteúdos do ensino correspondem aos conhecimentos e valores sociais acumulados pelas gerações passadas como verdades acabadas, e, embora a escola vise à preparação para a vida, não busca estabelecer relação entre os conteúdos que se ensinam e os interesses dos alunos, tampouco entre esses e os problemas reais que afetam a sociedade. (BRASIL, 1997, P. 27)

É vasto o campo de aplicação dos conhecimentos matemáticos adquiridos em sala de aula, mas o aluno “não” sabe disso, pois na sala de aula, seu aprendizado é limitado, priorizando algebrismos, memorização e o trabalho descontextualizado.

Para despertar o interesse no aluno pelo aprender matemático, é necessário mostrar ao mesmo que a referida disciplina tem aplicação prática, e não subsiste apenas de teoria, problematizando os conteúdos adequadamente e mostrando suas áreas de aplicação. Torres acrescenta:

> Assim, surge à necessidade de favorecer um ensino que leve o educando a problematizar os mais variados conceitos abordados na referida disciplina escolar. O homem como ser social e integrante de uma cultura necessita do conhecimento matemático na sua relação interpessoal, muitas vezes não sendo perceptível a sua utilização. No entanto, ele o utiliza assim como utiliza a sua língua materna. (TORRES 2016, p. 2)

Assim, faz-se necessário avançar do campo teórico para o prático, para mostrar aos discentes alguns ramos da aplicação dos conteúdos ministrados em sala de aula.

Diante de tais discussões surge a necessidade de se buscar ensinar a matemática de modo que mesma venha a favorecer ao aluno a aplicar seus conceitos, seja em conexão com outras áreas como a física, a geografia, a computação gráfica; seja no cotidiano. Segundo Menezes

> À medida que se defende a necessidade da utilização da matemática no cotidiano, inevitavelmente deve-se perceber que existem sujeitos envolvidos no processo de ensino e aprendizagem, de um lado temos o sujeito que ensina ou media e do outro o que aprende ou que constrói o conhecimento junto com o sujeito mediador. (MENEZES, 2016 apud CONCEIÇÃO, 2015, p. 2)

Alguns professores de matemática ainda estão presos a um modelo de ensino que não promove no aluno a mobilização da aprendizagem e nem a motivação para perceber que essa disciplina está presente em suas relações sociais. Assim, esse componente curricular torna-se, muitas vezes, distante da realidade do aluno. Isso está relacionado à maneira como ela é trabalhada em sala de aula. Além disso, a ausência de relacionamento da teoria com a prática no ensino da disciplina em questão, associada à sua não aplicação no cotidiano do discente, leva o aluno a não despertar em si o interesse pela disciplina.

Alguns profissionais da educação ainda fazem uso do tradicionalismo no processo ensino-aprendizagem, que coloca o professor como centro desse processo e detentor do conhecimento, e não como mediador, e tinha como objetivo principal preparar os alunos para que esses assumissem papéis na sociedade. Segundo Queiroz e Moita

> No ensino tradicional o principal objetivo da escola era preparar os alunos para assumirem papéis na sociedade, já que quem tinha acesso às escolas eram os filhos dos burgueses e a escola tomava como seu papel principal, fazer o repasse do conhecimento moral e intelectual porque através deste estaria garantida a ascensão dos burgueses e, consequentemente, a manutenção do modelo social e político vigente. (QUEIROZ e MOITA 2007, p.5)

No ensino tradicional, a proposta de educação era absolutamente centrada no professor, figura incontestável, único detentor do saber que deveria ser repassado para os alunos. Para os PCNs (BRASIL, 1997, p.27): A “pedagogia tradicional” é uma proposta de educação centrada no professor, cuja função se define como a de vigiar e aconselhar os alunos, corrigir e ensinar a matéria.

Metodologias tradicionais de ensino tem se mostrado ineficazes na aquisição de conhecimento por parte dos discentes. Assim, é necessário utilizar métodos e técnicas que possibilitem ao aluno exercer seu papel de sujeito ativo e construtor do conhecimento no processo ensino-aprendizagem.

É necessário despertar no aluno o interesse e a curiosidade pela disciplina, por meios que o levem a conhecer a matemática como algo dinâmico e aplicável. O educando deve ser levado a conhecer a relação entre a matemática e as outras áreas do conhecimento, saber que ela está em tudo a sua volta. Deve-se levar em consideração que para tal, é necessário que o educando assuma o papel de sujeito ativo nesse processo. Queiroz e Moita acrescentam que

> O professor e os conteúdos disciplinares devem sair do centro do processo pedagógico e o aluno deve ser colocado como fundamental, que deve ter sua curiosidade, criatividade, inventividade, estimulados pelo professor, que deve ter o papel de facilitador do ensino. A escola deve possibilitar a aprendizagem pela descoberta, focada no interesse do aluno, garantindo momentos para a experimentação e a construção do conhecimento, que devem partir do interesse do aluno. （QUEIROZ e MOITA 2007, p.8)

Assim, a forte relação entre a matemática e todas as áreas de conhecimento, facilita mostrar ao aluno essa relação da disciplina com o mundo em sua volta, elevando seu nível de conhecimento e despertando nele maior curiosidade pela aplicação da mesma.

A matemática mantém grande relação com a área da computação. A própria linguagem de computadores (binária) é um exemplo disso. Por trás daquilo que aparece na tela do computador temos pura aplicação matemática. Ressaltamos que

o ensino de linguagem de programação é importante, e auxilia no aprendizado do aluno. Segundo Reif

> Nos dias de hoje, aprender a programar é um diferencial comparado ao que foi aprender inglês há alguns anos, e deveria ser tão importante quanto ler ou escrever. Essa importância não se limita apenas às oportunidades de trabalho, ela possibilita ver o mundo de novas maneiras. (REIF, 2017, p. 13)

Devido a sua imensa importância e à falta de profissionais com habilidades em programação, muitos países perceberam que este aprendizado deve ser incorporado ao currículo das crianças ainda na fase escolar. Dessa maneira, desde cedo elas têm acesso a linguagem de programação, que auxilia no aprendizado de outras disciplinas assim como o desenvolve diversas habilidades pessoais como raciocínio lógico, trabalho em equipe, tomada de decisões e criatividade.

Para Reif (2017), inserção da linguagem de programação por meio da aplicação na matemática, reforça os ideais da prevalência de uma nova postura metodológica em detrimento da postura tradicional de ensino, ainda dominante no ambiente escolar. É importante introduzir nas escolas metodologias que possibilitem o aprendizado do aluno. Técnicas voltadas para a aplicação matemática contribuem para o aprendizado da mesma.

Na seção seguinte, veja a grande importância de se mostrar como se dá aplicação dos vetores de forma a dar movimentos a seres no ambiente virtual. Comandos gerados por meio de linguagens de programação Java permitem aplicar esse tão importante componente matemático na área da computação de modo a dar movimento a objetos na tela do computador.

2.2 VETORES

2.2.1 Breve contexto histórico

O estudo sobre vetores iniciou-se no século XIX com a necessidade de se representar geometricamente um número complexo. Alguns matemáticos como Caspar Wessel (matemático dinamarquês-norueguês, 1745-1818), Jean Robert Argand (um livreiro e matemático amador francês, nascido na suíça, 1768 – 1822), Carl Friedrich Gauss (matemático, astrônomo e físico alemão que contribuiu muito em diversas áreas da ciência, dentre elas a teoria dos números, estatística, análise matemática, geometria diferencial, goedésia, geofísica, eletrostática, astronomia e óptica, 1777 - 1855), entre outros, conceberam números complexos como pontos no plano bidimensional, isto é, como vetores bidimensionais.

August Ferdinand Möbius publicou, em 1827, o livro *The Barycentric Calculus*, no qual introduziu diretamente segmentos de reta que eram representados por letras do alfabeto; representação de vetores, mas não no nome.

Um trabalho foi publicado por Giusto Bellavits (matemático italiano, 1803-1880) sobre Geometria, em 1832, onde ele abordou a noção de vetor de maneira explícita. Os elementos básicos da sua obra são os segmentos de reta. Dados dois pontos A e B do plano, os segmentos AB e BA, de extremidades A e B, foram considerados por Bellavits elementos distintos. Essa convenção foi admitida por ele devido ao fato de que o segmento de reta delimitado pelos pontos A e B pode ser encontrado de duas maneiras diferentes: partindo de A para B, ou partindo de B para A. A classificação dos segmentos feita por Giusto, deu-se através da relação de equipolência que, por sua vez, originou a noção de vetor.

2.2.2 Segmentos orientados

Um segmento orientado é determinado por um par de pontos ordenados, em que o primeiro representa a origem do segmento, e o segundo representa a

extremidade. O segmento orientado de origem no ponto A e extremidade no ponto B será representado por AB.

Geometricamente, um segmento orientado AB é representado por uma flecha de A a B.

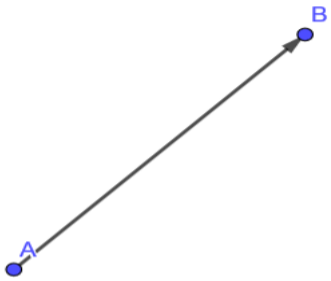


Figura 1. Segmento orientado. Fonte: autor da pesquisa.

Um segmento orientado é dito nulo quando sua origem coincide com sua extremidade, ou seja, quando esse segmento é determinado por um par de pontos coincidentes.

Se dois segmentos orientados não nulos AB e CD tem mesma direção, então as retas AB e CD são paralelas ou coincidentes. Os sentidos de dois segmentos orientados só podem ser comparados se eles tem a mesma direção. A figura 2 representa dois segmentos com mesma direção e sentidos contrários; portanto, opostos.

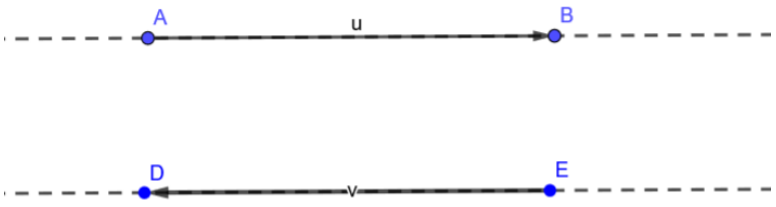


Figura 2. Segmentos orientados AB e ED opostos. Fonte: autor da pesquisa

Dois segmentos orientados AB e CD são equipolentes quando tem mesmo comprimento, direção e sentido. Indica-se por AB ≡ CD.

2.2.3 Vetores no plano

Sejam A e B pontos no plano. O vetor $\vec{u} = \overrightarrow{AB}$ ou $\vec{u}$ = B – A é o conjunto de todos os segmentos orientados que tem mesmo comprimento, direção, sentido de $\vec{u}$, ou seja, o conjunto de todos os vetores que são equipolentes a $\vec{u}$. Cada segmento equipolente a AB representa o vetor $\overrightarrow{AB}$. (Veja a figura 3).

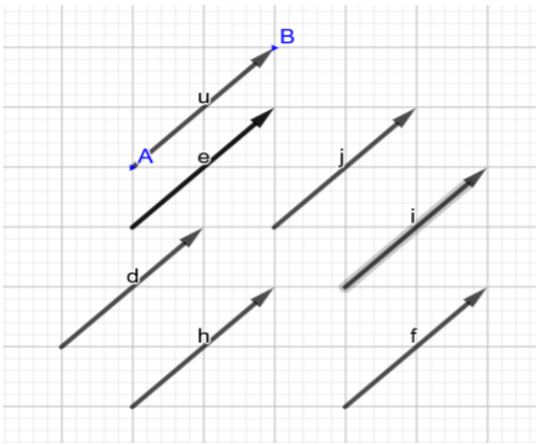


Figura 3. Vetores representantes de $\overrightarrow{AB}$. Fonte: autor da pesquisa.

Os vetores são manipulados através de suas representações em relação a um sistema de eixos ortogonais dado.

Dados um ponto A = (a_1, a_2) e B = (b_1, b_2), os números $b_1 - a_1$ e $b_2 - a_2$ são as coordenadas do vetor $\vec{u} = \overrightarrow{AB}$ e escreve-se $\vec{u} = (b_1 - a_1, b_2 - a_2)$.

Exemplo 1: sejam A = (1,3) e B = (3,1). As coordenadas do vetor $\vec{u} = \overrightarrow{AB}$ são dadas por $\vec{u}$ = (3 – 1,1 – 3) = (2,-2).

Seja OXY um sistema de eixos ortogonais no plano. Para todo vetor $\vec{u}$ existe um único ponto P tal que $\vec{u} = \overrightarrow{OP}$. Além disso, as coordenadas do ponto P coincidem com as coordenadas do vetor $\vec{u}$.

Exemplo 2: dados os pontos E = (-1,3) e F = (3,2), o ponto P tal que $\overrightarrow{OP} = \overrightarrow{EF}$ é dado por P = (3-(-1),2-3) = (4,-1). Os vetores $\overrightarrow{OP}$ e $\overrightarrow{EF}$ estão ilustrados na figura 4.

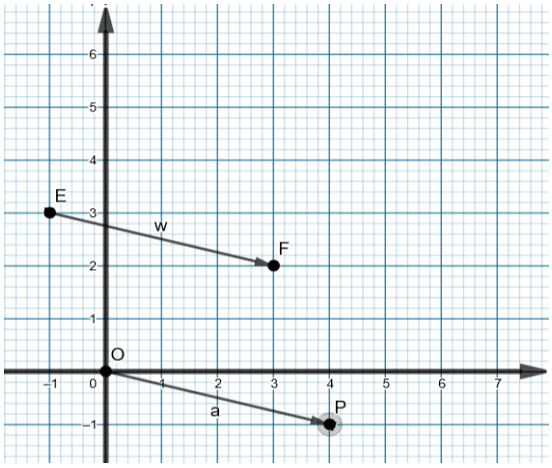


Figura 4. EF ≡ OP, Exemplo 2. Fonte: autor da pesquisa.

Geometricamente, os vetores são representados por segmentos de retas orientados no plano ou espaço. Com isto podemos associar a um par *(a,b)* módulo, direção e sentido. A direção e o sentido do segmento identificam a direção e o sentido do vetor. E o módulo é o comprimento do vetor, representado por:

$$\sqrt{a^2 + b^2}$$

Que também é chamado de norma do vetor, representado por

$$|| \vec{u} || = \sqrt{a^2 + b^2}.$$

O módulo do vetor *u* está representado na figura 5.

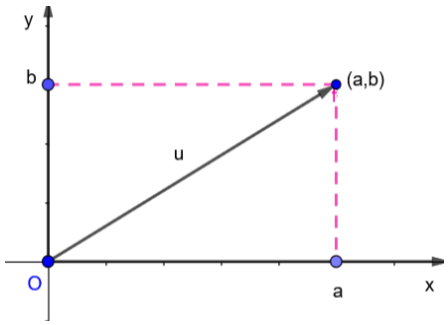


Figura 5. Representação gráfica de um vetor no plano. Fonte: autor da pesquisa

2.3 OPERAÇÕES COM VETORES

Neste tópico, serão estudadas as operações com vetores que serão usadas para dar ou manipular o movimento de objetos em um ambiente virtual. A saber:

2.3.1 Soma de vetores

A operação de adição de vetores que a cada par de vetores $\vec{u}$ e $\vec{w}$ associa um novo vetor, é o vetor soma de $\vec{u}$ e $\vec{w}$, designado por $\vec{u} + \vec{w}$, e é definido da seguinte maneira:

Sejam os vetores $\vec{u}$ e $\vec{w}$. Se os vetores $\vec{u}$ e $\vec{w}$ são tais que a extremidade do vetor $\vec{u}$ coincide com a origem do vetor $\vec{w}$, então o vetor soma de $\vec{u}$ com $\vec{w}$ é o vetor $v = \vec{u} + \vec{w}$ que origem na origem de $\vec{u}$ e extremidade na extremidade de $\vec{w}$. (ver figura 6).

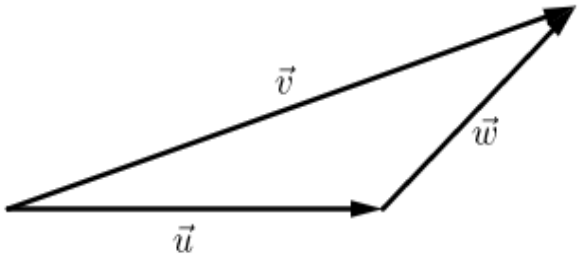


Figura 6: Soma de vetores. Fonte: autor da pesquisa

Este modo de somar dois vetores é conhecido como regra do triângulo. Outra maneira de se representar geometricamente a soma de vetores é a regra do polígono, nesse caso, teremos a soma entre mais de dois vetores. Usando esta regra podemos somar qualquer quantidade de vetores, desde que a origem de um vetor coincida com a extremidade do anterior. O vetor soma (ou vetor resultante) será o vetor que tem a origem na origem do primeiro vetor e a extremidade na extremidade do último, formando assim um polígono. (Veja as figuras 6 e 7).

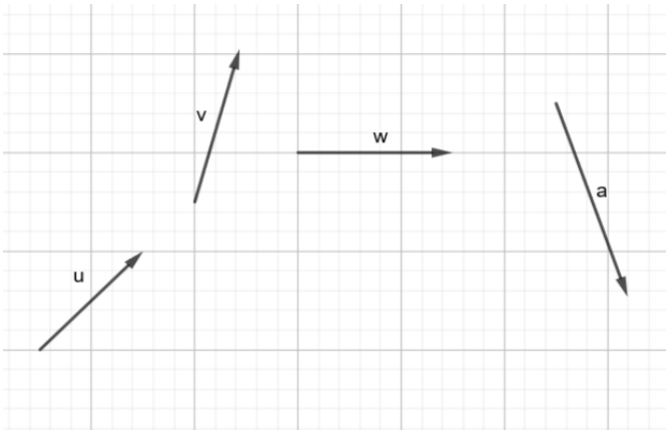


Figura 7. Soma de vetores no plano. Fonte: autor da pesquisa

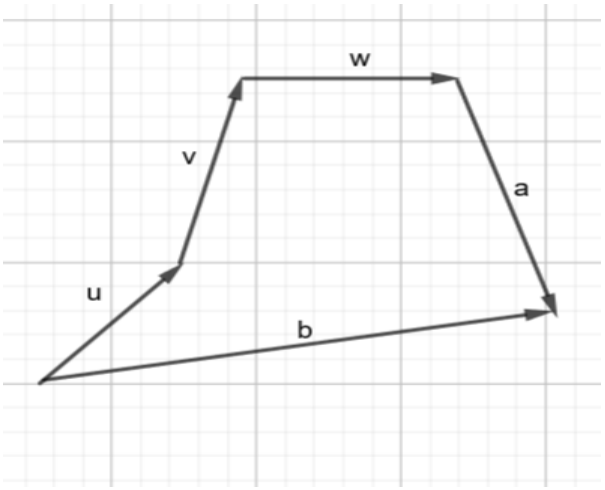


Figura 8. Soma de vetores no plano: regra do polígono. Fonte: autor da pesquisa

Na figura 8, o vetor $\vec{b}$ é o vetor soma dos vetores $\vec{u}$, $\vec{v}$, $\vec{w}$ e $\vec{a}$ representados na figura 7.

Outra forma geométrica de visualizar a soma de dois vetores no plano é feita pela regra do paralelogramo. Conforme representado na figura 9.

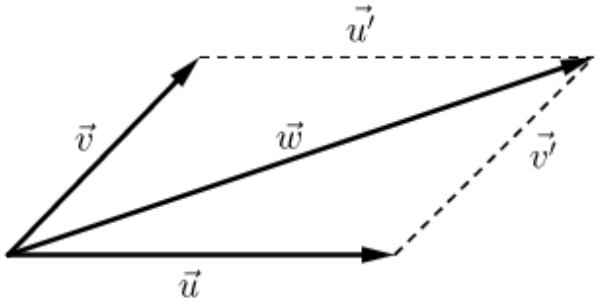


Figura 9. Soma de vetores: regra do paralelogramo. Fonte: autor da pesquisa

Na regra do paralelogramo os vetores $\vec{u}$ e $\vec{v}$ tem a mesma origem. Na extremidade do vetor $\vec{u}$ traçamos o vetor $\overrightarrow{v'}$ e paralelo ao vetor $\vec{v}$ e na extremidade do vetor $\vec{v}$ traçamos o vetor vetores $\overrightarrow{u'}$, paralelo ao vetor $\vec{u}$, formando assim um paralelogramo. O vetor soma, representado por $\vec{w}$, é a diagonal do paralelogramo com origem no encontro das origens dos vetores $\vec{u}$ e $\vec{v}$ e extremidade no encontro das extremidades dos vetores $\overrightarrow{u'}$ e $\overrightarrow{v'}$.

Quando se somam dois vetores $\vec{u} = (a_1, a_2)$ e $\vec{v} = (b_1, b_2)$, somam-se as coordenadas desses vetores. A soma de dois vetores *u* e *v* é definida da seguinte maneira:

$$\vec{u} + \vec{v} = (a_1+b_1, a_2+b_2).$$

Por exemplo, se forem somados os vetores $\vec{u}$ = (2,-3) e $\vec{v}$ = (1,4), tem-se:

$$\vec{u} + \vec{v} = (2+1,-3+4)$$
$$\vec{u} + \vec{v} = (3,1)$$

A adição de vetores satisfaz às seguintes propriedades:

Sejam $\vec{u}$, $\vec{v}$ e $\vec{w}$ vetores quaisquer, então:

(A1) comutativa: $\vec{u} + \vec{v} = \vec{v} + \vec{u}$

(A2) associativa: $\vec{u} + (\vec{v} + \vec{w}) = (\vec{u} + \vec{v}) + \vec{w}$

(A3) existência de elemento neutro aditivo: $\vec{u} + \vec{0} = \vec{u}$

(A4) existência do inverso aditivo: $\vec{u} + (-\vec{u}) = \vec{0}$.

Demonstrações:

Propriedade A1:

Sejam os vetores $\vec{u} = (x_u, y_u)$ e $\vec{v} = (x_v, y_v)$, temos que:

$$\vec{u} + \vec{v} = (x_u+x_v, y_u+y_v) = (x_v+x_u, y_v+y_u) = \vec{v} + \vec{u}$$

De onde concluímos que

$$\vec{u} + \vec{v} = \vec{v} + \vec{u}$$

Propriedade A2:

Sejam os vetores $\vec{u} = (x_u,y_u)$, $\vec{v} = (x_v,y_v)$ e $\vec{w} = (x_w,y_w)$temos que:

$$(\vec{u} + \vec{v}) + \vec{w} = (x_u+x_v,y_u+y_v) + (x_w,y_w) = (x_u+x_v+x_w,\ y_u+y_v+y_w) = (x_u,y_u) + (x_v+x_w,y_v+y_w) = \vec{u} + (\vec{v} + \vec{w})$$

De onde concluímos que $(\vec{u} + \vec{v}) + \vec{w} = \vec{u} + (\vec{v} + \vec{w})$.

As demonstrações das propriedades A3 e A4 ficam a cargo do leitor.

2.3.2 Diferença de vetores

A subtração de dois vetores $\vec{a}$ e $\vec{b}$, é definida como sendo a soma do vetor $\vec{a}$ como o oposto de $\vec{b}$. Assim:

$\vec{a}$ - $\vec{b}$ = $\vec{a}$ + (-$\vec{b}$) = $\vec{c}$

A representação gráfica de $\vec{a}$ - $\vec{b}$ é dada por:

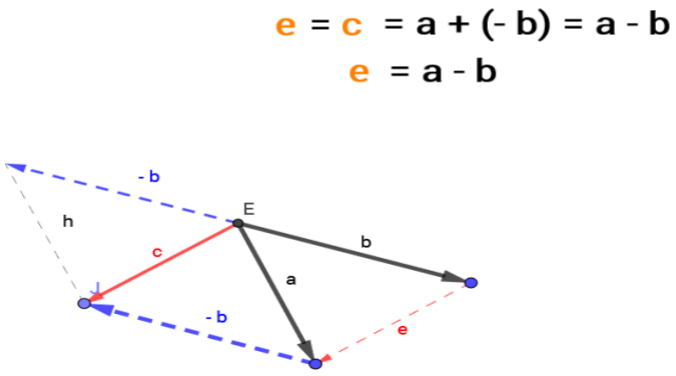


Figura 10: Diferença de vetores. Fonte: autor da pesquisa.

Quando se efetua a subtração entre dois vetores, subtraem-se as coordenadas desses vetores. Por exemplo, dados os vetores $\vec{u}$ = (5,6) e $\vec{v}$ = (-2,4), se quisermos encontrar $\vec{u}$ - $\vec{v}$, teremos:

$$\vec{u} - \vec{v} = (5,6) +(– (-2,4))$$
$$\vec{u} - \vec{v} = (5,6) + (2,-4)$$
$$\vec{u} - \vec{v} = (5+2,6-4)$$
$$\vec{u} - \vec{v} = (7,2)$$

2.3.3 Multiplicação de um vetor por um escalar

Na multiplicação de vetores por um escalar, é necessário se pensar um pouco diferente. Quando se fala em multiplicar um vetor por um escalar, estará se falando em redimensionando esse vetor, ou seja, aumentando ou diminuindo seu comprimento de acordo o com o valor escalar pelo qual o vetor está sendo multiplicado.

Sejam um vetor $\vec{u}$ e um número real *k,* o produto de *k* por $\vec{u}$ é um vetor $\vec{v}$ que é representado por:

$$\vec{v} = k \, . \, \vec{u}$$

O vetor $\vec{v}$ terá a mesma direção do vetor $\vec{u}$ e seu sentido será definido pelo sinal de *k* (se $k > 0$, os dois vetores terão o mesmo sentido; se $k < 0$, os dois vetores terão sentidos contrários). Se $k = 0$ ou $\vec{u} = 0$, então $\vec{v} = 0$.

Na figura abaixo temos $\vec{v} = 3 \, . \, \vec{u}$

Figura 11. Multiplicação de um vetor por um escalar. Fonte: autor da pesquisa.

Quando se multiplica um vetor por um escalar, estará se multiplicando suas coordenadas por esse escalar. Por exemplo, multiplicando o vetor $\vec{u}$ = (4,7) por k = 2, teremos:

$$\vec{v} = k \, . \, \vec{u}$$
$$\vec{v} = 2 \, . \, (4 \, , \, 7)$$
$$\vec{v} = (2.4 \, , \, 2.7)$$
$$\vec{v} = (8 \, , \, 14)$$

A multiplicação por escalar satisfaz as seguintes propriedades:

Sejam $\vec{u}$ e $\vec{v}$ vetores quaisquer não nulos e k_1, k_2 números reais, então:

(M1) $k_1 \, . \, (\vec{u} + \vec{v}) = k_1 \, . \, \vec{u} + k_1 \, . \, \vec{v}$

(M2) $(k_1 + k_2) \cdot \vec{u} = k_1 \cdot \vec{u} + k_2 \cdot \vec{u}$

(M3) $k_1 \cdot (k_2 \cdot \vec{u}) = (k_1 \cdot k_2) \cdot \vec{u}$

(M4) $1 \cdot \vec{u} = \vec{u}$ e $0 \cdot \vec{u} = 0$.

Demonstrações:

Propriedade M1:

Sejam os vetores $\vec{u} = (u_1, u_2)$ e $\vec{v} = (v_1, v_2)$ e k_1 um número real. Temos que:

$$K_1 \cdot (\vec{u} + \vec{v}) = k_1 \cdot [(u_1, u_2) + (v_1, v_2)] = k_1 \cdot (u_1 + v_1, u_2 + v_2)$$

$$= (k_1 \cdot u_1 + k_1 \cdot v_1, k_1 \cdot u_2 + k_1 \cdot v_2)$$

$$= (k_1 \cdot u_1, k_1 \cdot u_2) + (k_1 \cdot v_1, k_1 \cdot v_2)$$

$$= k_1 \cdot (u_1, u_2) + k_1 \cdot (v_1, v_2)$$

$$= k_1 \cdot \vec{u} + k_1 \cdot \vec{v}$$

As demais propriedades se demonstram de maneira análoga.

2.3.4 Módulo de um vetor

Matematicamente, módulo é um grandeza que representa o comprimento de um vetor. O módulo (também chamado de norma) do vetor $\vec{v} = (a, b)$ é encontrado pela equação matemática:

$$||\vec{u}|| = \sqrt{a^2 + b^2} \qquad (1.3.4.1)$$

A figura 12, mostra o módulo do vetor $\vec{u} = (a,b) = (6,8)$.

$$||\vec{u}|| = \sqrt{6^2 + 8^2} = \sqrt{36 + 64} = \sqrt{100} = 10$$

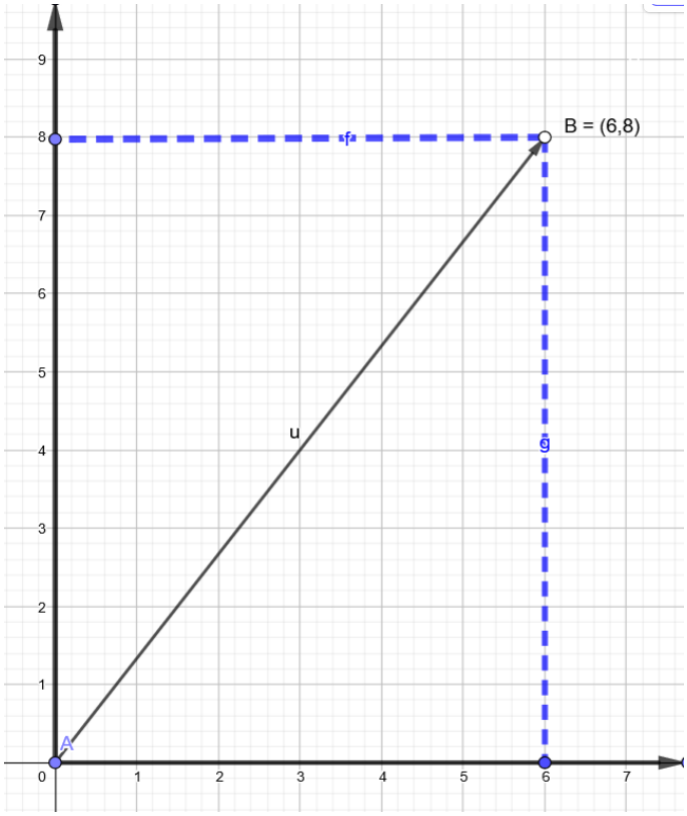


Figura 12: Módulo ou comprimento de um vetor. Fonte: autor da pesquisa.

As operações descritas anteriormente serão necessárias ao uso de alguns comandos que serão usados para dar movimentos ordenados a objetos na tela do computador e, em alguns casos, para calcular a magnitude de um vetor.

2.4 VETORES NO ESPAÇO

Definiremos vetores no espaço de maneira semelhante ao que foi definido em vetores no plano, tomaremos apenas alguns cuidados nos casos em que for discutido sobre retas paralelas, pois no espaço nem sempre retas que não se intersectam, são paralelas, visto que elas podem não estar no mesmo plano, por exemplo. A representação de um vetor no espaço tridimensional tem suas coordenadas nos eixos ortogonais XYZ. (Ver figura 13).

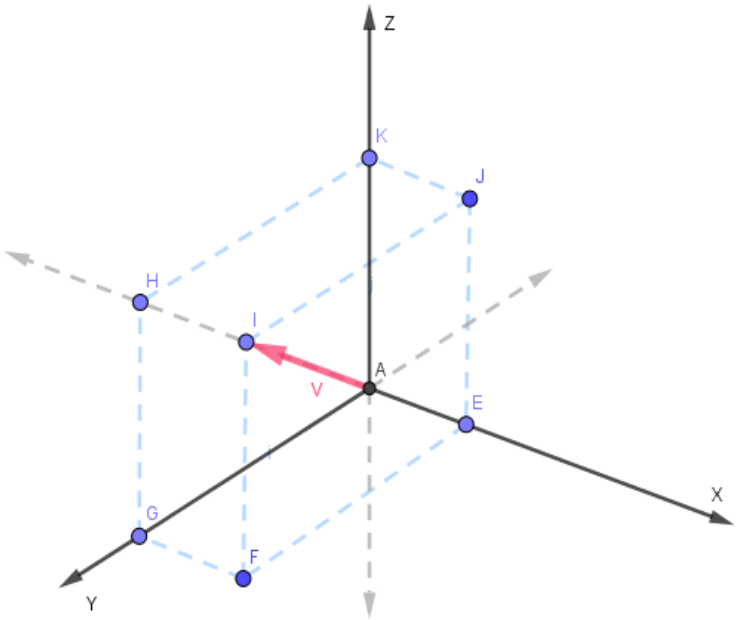


Figura 13. Representação de um vetor no espaço. Fonte: autor da pesquisa.

Sejam A = *(a,b,c)* e B = *(a',b',c')* pontos no espaço. Os números reais *a' – a, b' – b e c' – c* representam as coordenadas do vetor $\overrightarrow{AB}$ no sistema de eixos ortogonais XYZ. Indica-se por:

$$\overrightarrow{AB} = (a' - a,\ b' - b,\ c' - c)$$

Pelo exposto acima, as coordenadas de um vetor podem ser calculadas usando qualquer segmento orientado que o represente. De maneira particular, dado um vetor $\vec{v} = (a, b, c)$, o ponto P = (*a, b, c*) satisfaz $\vec{v} = \overrightarrow{OP}$, em que o vetor $\overrightarrow{OP}$ é o representante na origem do vetor $\vec{v}$, que tem origem na origem do sistema XYZ.

2.4.1 Soma de vetores e multiplicação de vetores por um escalar

Sejam os vetores $\vec{u} = (u_1, u_2, u_3)$ e $\vec{v} = (v_1, v_2, v_3)$. Definimos a soma dos vetores $\vec{u}$ e $\vec{v}$ como $\vec{u} + \vec{v} = (u_1 + v_1, u_2 + v_2, u_3 + v_3)$, e a multiplicação de $\vec{u}$ por um escalar $k \in R$ é dada por $k.\vec{u} = (ku_1, ku_2, ku_3)$.

A adição de vetores e multiplicação de um escalar por um vetor no espaço tridimensional, satisfazem as mesmas propriedades que a adição de vetores e multiplicação de um vetor por um escalar no plano.

2.5 COMO A APLICAÇÃO DOS VETORES PODE DAR MOVIMENTOS AOS SERES EM AMBIENTE VIRTUAL.

Nessa seção, será tratada a aplicação dos vetores geométricos na computação gráfica, usados para darem movimentos aos seres em um ambiente virtual e fazer algumas simulações. A computação gráfica é a área da computação destinada à geração de imagens em geral, em forma de representação de dados e informações, ou de fenômenos do mundo real.

A aplicação dos vetores para tal finalidade, se dá por meio de comandos específicos em linguagens de programação Java, utilizada na ferramenta *processing* 3.3.5 (gratuito). O *processing* pode ser utilizado na computação gráfica, na programação usada para dar movimentos a objetos no ambiente virtual e também pode ser utilizada para o ensino de algoritmos.

Inicialmente, é necessário entender o que significam alguns comandos básicos que serão usados para nosso propósito.

Quando se trabalha com comandos em linguagem de programação usa-se, inicialmente, variáveis que são declaradas por meio de comandos específicos. Esses comandos significam que é reservada parte da memória do computador para armazenar informações que são representados por um conjunto de dados específicos, os quais são chamados por Tipos de Dados. Os Tipos de Dados definem os valores que uma variável pode assumir, bem como quais operações podem ser efetuadas sobre esse dado. A seguir são apresentados os principais tipos de dados utilizados na programação da qual faremos uso nesse trabalho.

- ***Float* (ponto flutuante)**: Significa a inserção de valores reais. Com essa variável reservamos parte da memória do computador para armazenar valores reais, que mais tarde podem ser usadas em operações matemáticas que serão chamadas por meio de funções que definem essas operações.
- ***Int* (números inteiros):** Significa a inserção de valores inteiros. Nesse caso, separamos uma parte da memória do computador para armazenar os valores inteiros.

A sintaxe para a definição de uma variável na ferramenta *Processing 3.3.5* começa com a definição do Tipo de dado, em seguida o identificador da variável, e por fim, a atribuição de um valor inicial à variável.

A estrutura do código de processamento é dividida em duas seções ou funções principais: *setup e Draw*. A seção *Setup* é usada para definir propriedades iniciais de ambiente (por exemplo, tamanho da tela, cor de fundo, carregamento de imagens ou fontes, etc.) e a seção *Draw* para executar os comandos de desenho (por exemplo, ponto, linha, elipse, imagem, etc.) em um ciclo que pode ser usado para Animação.

O estudo sobre aplicação dos vetores nesse ramo da computação gráfica, será introduzido por meio de alguns exemplos, a saber:

2.5.1 Bola de salto quicando sem vetores.

Nesse exemplo tem-se, no ambiente virtual, o movimento de uma bola programada usando apenas as variáveis x e y (que nos dão a localização do objeto na tela projetada num plano cartesiano) sem o uso de vetores, e as velocidades do objeto em relação aos eixos x e y. Abaixo, o código que gera esse movimento na bola. Esse é apenas um exemplo. Necessário é lembrar que a posição da bola varia de acordo com os valores que são acrescidos à posição em relação a esses eixos. (Ver figura 14).

```
// inicialmente cria-se as variáveis que serão usadas nas operações
float x = 10;
float y = 10;
float velocidadex = 2;
float velocidadey = 2;
// setup () é executado uma vez quando o sketch começa e draw () faz
um loop para todo o sempre (até você sair).
void setup() {
  size(800, 600);
  smooth();
}
void draw() {
  background(255);
  //Mova a bola de acordo com sua velocidade.
x = x + velocidadex;
  y = y + velocidadey;
```

```
        // cria-se a função que fará o objeto se manter na tela de execução
          if ((x >=width) || (x <= 0)) {
            velocidadex = velocidadex * -1;
          }
          if ((y >= height) || (y <= 0)) {
            velocidadey = velocidadey * -1;
          }
        // dever ser definidas as cores da borda externa da bola, a espessura
        dessa borda, a cor do preenchimento interno da bola e suas
        características.
          stroke(0);
          strokeWeight(2);
          fill(127);
          ellipse(x, y, 48, 48);
        }
```

Código 1: bola se movimentando sem o uso de vetores

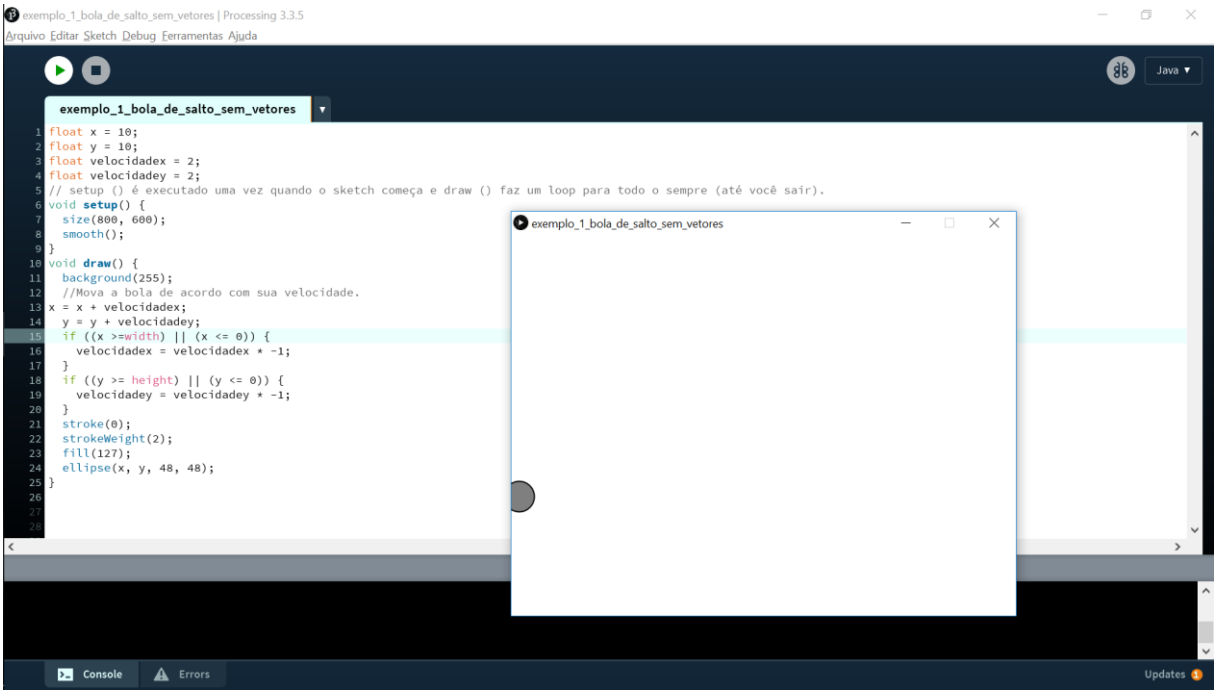


Figura 14. Objeto se movimentando na tela segundo uma velocidade atribuída a ele. Fonte: autor da pesquisa.

A Figura 14 mostra o ambiente do *processign* que contém os comandos, e a tela de execução dos mesmos (a tela branca onde o objeto executa seu movimento)

Seguem as descrições de alguns comandos:

float: como citado anteriormente, cria as variáveis que serão usadas nas operações chamadas pelas funções que se seguem;

velocidadex *e* velocidadey: são as velocidades do objeto em relação aos eixos x e y (plano cartesiano);

***ellipse*(x, y, 48, 48):** Nesse exe]]mplo, o objeto apresentado na tela é uma elipse. As coordenadas (x, y) representam a localização do objeto na tela e as coordenadas (48, 48) são os comprimentos (em pixels) dos eixos da elipse.

As equações abaixo descritas, nos remetem à equação matemática $S=s_0 + v.t$ (equação do movimento retilíneo uniforme). Isso quer dizer que o objeto virtual iniciará seu movimento, a partir de uma posição inicial (x,y) e continuará se movimentando em relação aos eixos x e y, segundo os valores acrescidos às variáveis x e y (posição do objeto).

x = x + velocidadex*;*

y = y + velocidadey*;*

O comando "*if*" significa "se". Esse comando introduz uma condição lógica para realização de uma atividade. No exemplo sobre a bola de salto, o comando if ((x >= *width)* || (x <=0)) significa que o a bola deve se movimentar dentro do intervalo em que x >=*width (width*=800, valor atribuído em *size*(800,600)), ou x <=0.

Para que o objeto permaneça dentro desse intervalo, ou seja, para que essa condição seja verdadeira, é multiplicada a velocidade em relação a x por -1 (velocidadex = velocidadex * -1;); assim, sempre que o objeto que se movimenta no ambiente virtual quiser ultrapassar esses limites, ele refletirá o sentido do seu movimento quando colidir com as bordas horizontais da tela, permanecendo assim, dentro do limite lhe imposto, em relação ao eixo x. De maneira semelhante, o comando if ((y >=*height*) || (y <=0)) impõe condições para que o objeto em movimento não ultrapasse os limites em relação ao eixo y (y >=600 = *heigth* ou y <=0) para isso a velocidade em relação a vertical é multiplicada por -1 (velocidadey = velocidadey * -1;) para que o objeto reflita o sentido do seu movimento, sempre que colidir com as bordas verticais.

O comando *stroke*(0) se refere às cores da borda do objeto. A cor zero deixa a borda preta, e a cor 255 deixa a borda branca. O comando *strokeWeight(2)* determina a largura da borda externa do objeto. O comando *fill*(127) trata do preenchimento interno da bola que se movimenta (variando de 0 a 255). Uma observação interessante

quanto a isso é que quando atribuímos, por exemplo, três valores ao *fill()*, ao *strok()* ou ao *strokeWeight(),* fazendo-os variar de 0 a 255, estamos fazendo a combinação das três cores primárias (vermelho, verde e azul) gerando uma nova cor para preenchimento interno e para as bordas do objeto. Se atribuirmos dois valores, estaremos combinando duas dessas cores. Se atribuirmos apenas um valor, estamos fazendo a cor variar entre preto e branco.

Na Figura 14 temos algo muito simples - uma tela em branco com uma forma circular (uma "Bola") viajando por aí. Esta bola tem algumas propriedades, que são representadas no código como variáveis.

Localização	x e y
Rapidez	velocidadex e velocidadey

O uso de vetores não nos permitirá fazer nada de novo. Apenas adicionar vetores não fará seus esboços de processamento simularem a física. No entanto, eles simplificarão seu código e fornecerão um conjunto de funções para operações matemáticas comuns que acontecem repetidamente ao programar o movimento. Como uma introdução aos vetores, serão trabalhadas operações em duas dimensões, pois em três dimensões, seria necessário usar operações mais complexas.

Uma maneira de descrever um local é o caminho retirado da origem para alcançar essa localização, ou seja, é o caminho traçado desde a origem do movimento até o ponto de chegada. Portanto, uma localização pode ser o vetor que representa a diferença entre localização e origem. Estamos armazenando os dados para dois números de ponto flutuante, um x e um y.

Existem muitas operações matemáticas que são comumente usadas com vetores. Abaixo está uma lista abrangente das operações disponíveis e como funcionam na classe PVector (um vetor no ambiente virtual na ferramenta *processing* 3.3.5). Passaremos por alguns dos principais agora.

Quadro 1

- Add ( ) - adicionar vetores
- sub ( ) - subtrair vetores

- mult ( ) multiplicação de um vetor por um númerop real
- normalize ( ) - normalize o vetor para um comprimento unitário de 1
- limit ( ) - limite a magnitude de um vetor
- rand( ) - dimensiona o vetor com a multiplicação
- div ( ) - escala o vetor com divisão
- mag ( ) - calcula a magnitude

No quadro 1 são citadas algumas operações, entre as tais, adição de vetores. Elas nos darão suporte para introduzirmos nosso próximo exemplo, que se sergue. veja:

2.5.2 Bola de saltar com vetores (Pvectors)

Esse exemplo mostra agora como um objeto se move no ambiente virtual com o uso de vetores, que definirão, por exemplo, localização, velocidade e aceleração. Observe os comandos abaixo. (Ver figura 15).

```
       //inicialmente são criadas as variáveis que serão usadas nas
operações
       PVector posicao;
       PVector velocidade;
       // a função setup guarda as configurações iniciais
       void setup() {
         size(800,600);
         smooth();
         background(255);
         posicao = new PVector(10,100);
         velocidade = new PVector(2.3,5);
       }
       // a função Draw guarda os comandos que permanecerão em constante
execução
       void draw() {
           fill(255,10);
         rect(0,0,width,height);
         posicao.add(velocidade);
       // a função if introduz a condição para que o objeto permaneça na
tela de execução.
         if ((posicao.x >= width) || (posicao.x <= 0)) {
           velocidade.x = velocidade.x * -1;
         }
```

```
        if ((posicao.y >=height) || (posicao.y <= 0)) {
            velocidade.y = velocidade.y * -1;
          }
          stroke(0);
          fill(175);
        ellipse(posicao.x,posicao.y,16,16);
        }
```

Código 2: bola se movimentando com o uso de vetores

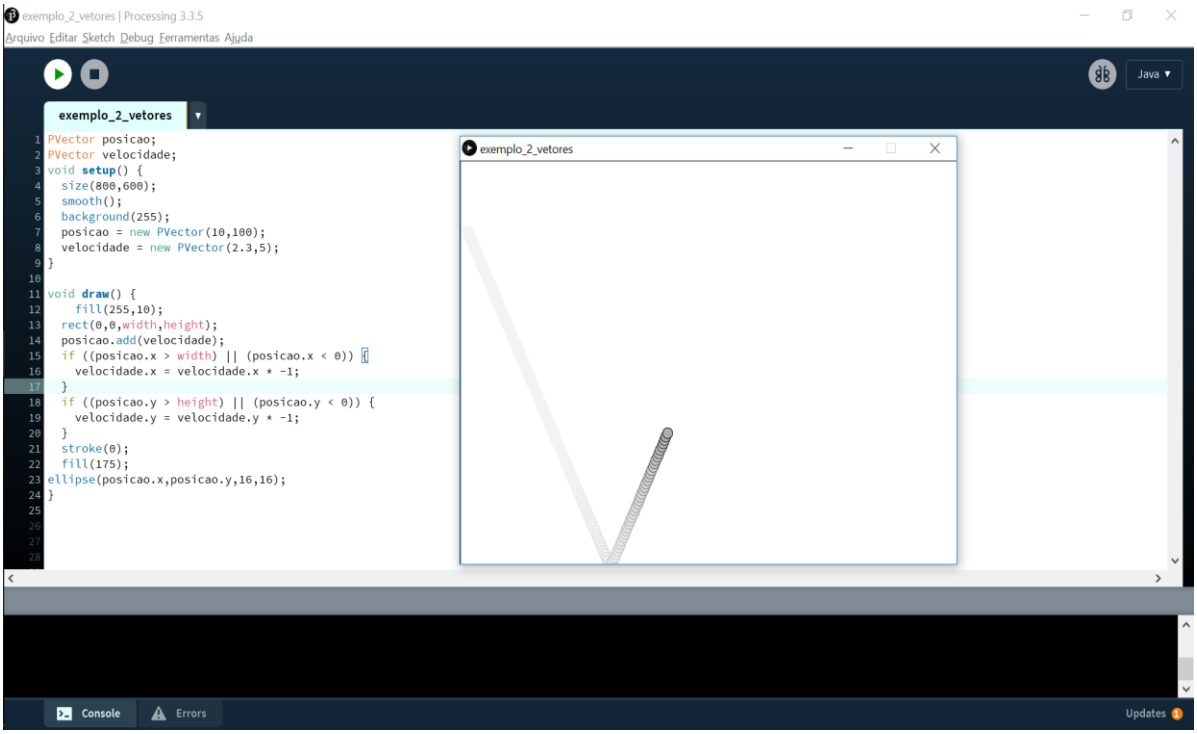


Figura 15: bola de salto com vetores. Fonte: autor da pesquisa.

Entenda os comandos acima: Inicialmente são criadas as variáveis que, nesse caso, tem características de vetores (Pvector posicao *e* Pvector velocidade). Criadas as variáveis, as operações são iniciadas por meio da função setup *(void setup*()). Define-se que a variável posicao é um vetor de coordenadas (10,100) (*posicao = new PVector(10,100);)*, e que a variável velocidade é um vetor de coordenadas (2.3,5) (velocidade = new PVector(2.3,5);). Em seguida, na função *void draw* (para que o comando permaneça em execução até você fechar o programa) dar-se ordem à bola em movimento que ela não pode ultrapassar os limites x >= 800 ou x <= 0 e y >= 600 ou y <= 0, refletindo o sentido de seu movimento sempre que colidir com as bordas

da tela, permanecendo assim, sempre dentro da tela de execução do comando *(if ((*posicao*.x >= width)* || *(location.x <= 0)) { velocidade.x = velocidade.x *-1; } if ((*posicao*.y >= height)* || *(*posicao*.y <= 0)) { velocidade.y = velocidade.y *-1; }.*

Define-se a cor das bordas do objeto e a cor de preenchimento interno do mesmo (*stroke*(0)) e *fill*(175)). Por fim, declara-se que o objeto é um círculo (elipse com eixos focal e não focal com comprimentos iguais) e sua localização ao longo do movimento (*ellipse(*posicao*.x,* posicao*.y*,16,16);).

O movimento na tela é gerada por que os vetores são adicionados várias vezes (quando na função *void draw*), gerando assim sempre um novo vetor de comprimento maior por meio do comando posicao*.add(*velocidade*)*; e dando ao objeto uma nova posição a cada execução desse comando. Isso faz com que ele permaneça sempre em movimento.

2.5.3 Magnitude de um vetor

De maneira não menos importante à introdução ao estudo dos vetores para dar movimento a objetos em ambientes virtuais, será visto agora, um exemplo que envolva a subtração de vetores, programando os mesmos para encontrar a magnitude de um outro vetor. Segue:

```
      void setup() {
        size(400,320);
      }
      void draw() {
        background(255);
      //Dois PVectors, um para a localização do mouse e outro para o centro
da janela
        PVector mouse  = new PVector(mouseX,mouseY);
        PVector center = new PVector(width/2,height/2);
      //Subtração de PVector!
        mouse.sub(center);
      //Desenhe uma linha para representar o vetor.
        translate(width/2,height/2);
        line(0,0,mouse.x,mouse.y);
      }
 (SHIFFMAN 2012, p. 56)
```

Código 3: comprimento de um vetor

O exemplo acima descreve, na tela de execução, uma linha que, partindo do centro (*PVector(width/2,heigth/2))* segue o mouse *(PVector(mouseX,mouseY))* por meio da subtração de vetores. O vetor que vai da origem até o mouse é chamado de vetor mouse. Veja o que acontece por trás do que está sendo executado na tela do programa nas figuras 16 e 17.

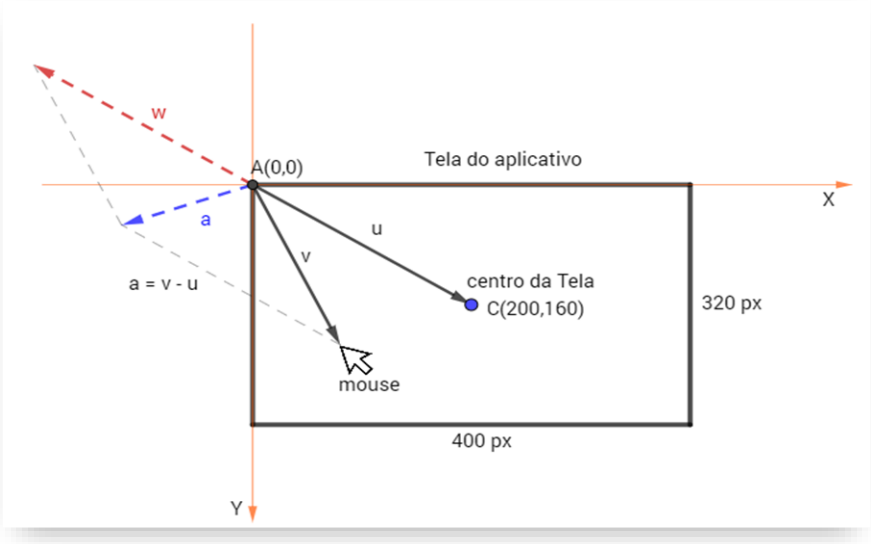


Figura 16. Cálculo da magnitude de um vetor. Fonte: autor da pesquisa.

A referência (ou origem) do plano cartesiano XY sempre fica na posição *top* e *left* da Tela de execução do aplicativo, Na figura 16, fazendo ***a = v – u***, tem-se que a origem do vetor (**a**) sempre estará na origem (0,0) do plano cartesiano XY.

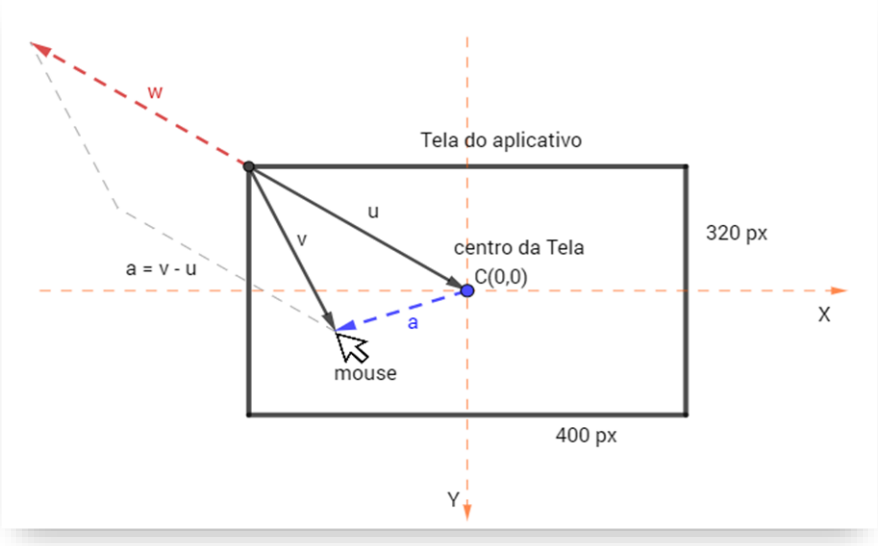


Figura 17. Cálculo da magnitude de um vetor 2. Fonte: autor da pesquisa.

Ao ser aplicado o método *translate*(x_1,y_1) faz-se com que a origem do plano cartesiano mude para a posição (x_1,y_1). Na Figura 17, temos que a origem do plano XY foi transladado para a posição (200,160) ou seja, fazendo *translate*(200,160) o vetor **a**, ou qualquer outro vetor que for criado abaixo da linha que contém o método *translate***,** terá origem no centro da tela do aplicativo. (Ver figura 18).

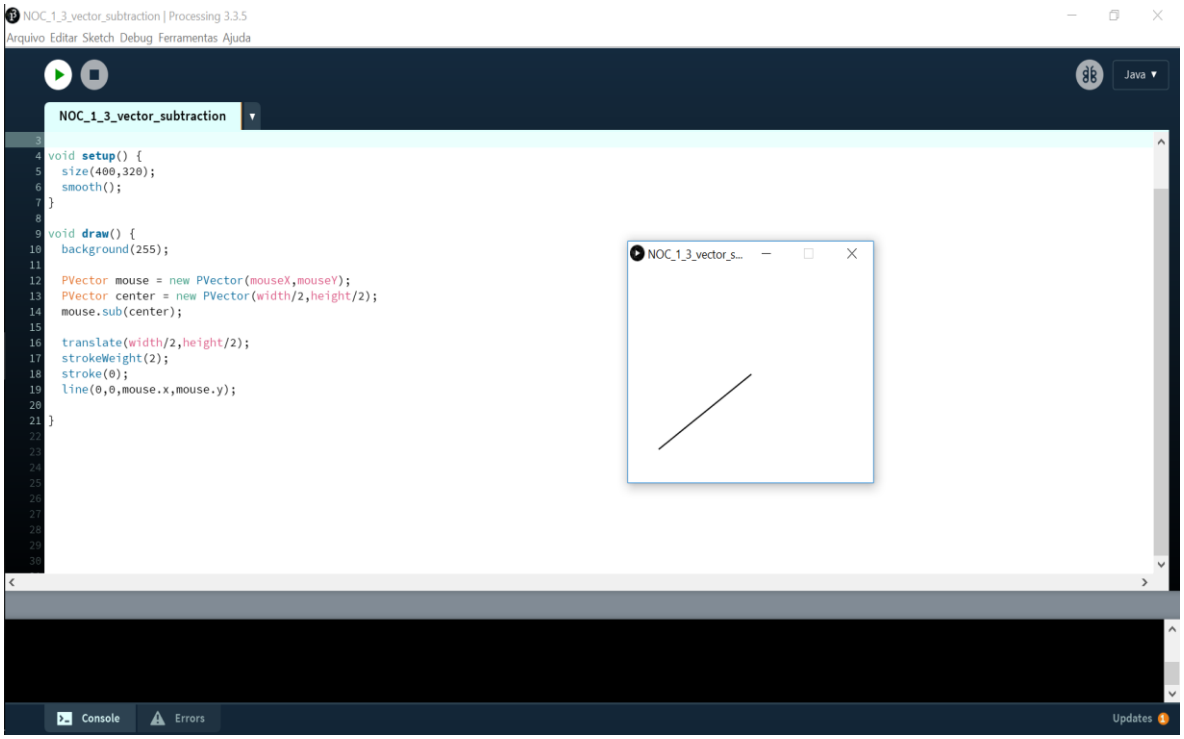


Figura 18: Magnitude de um vetor. Fonte: autor da pesquisa.

Observe agora o exemplo abaixo:

```
      void setup() {
        size(800,600);
      }
      void draw() {
        background(255);
        PVector mouse = new PVector(mouseX,mouseY);
        PVector center = new PVector(width/2,height/2);
        mouse.sub(center);
        mouse.mult(0.5);
        translate(width/2,height/2);
        line(0,0,mouse.x,mouse.y);
      }
(SHIFFMAN 2012, p. 57)
```

Código 4: comprimento fixo de um vetor

Observe que o exemplo dado é o mesmo exemplo citado anteriormente, mas acrescido do comando *(mouse.mult*(0.5);), isso significa que o vetor que será visualizado na tela tem a metade do vetor mouse.

Veja agora como calcular a magnitude (módulo, comprimento) de um vetor, e a representá-lo por meio de um retângulo de comprimento m igual ao módulo do dado vetor, e largura 10 (essa largura varia de acordo com o valor atribuído a ela). Observe o exemplo a seguir:

```
    void setup() {
      size(800,600);
      }
    void draw() {
      background(255);
      PVector mouse = new PVector(mouseX,mouseY);
      PVector center = new PVector(width/2,height/2);
      mouse.sub(center);
    //A magnitude (comprimento) de um vetor pode ser acessada através da
    função mag (). Aqui é usado como a largura de um retângulo desenhado
    no topo da janela.
      float m = mouse.mag();
      fill(0);
      rect(0,0,m,10);
      translate(width/2,height/2);
    // o vetor é representado por uma linha que parte do centro da tela
    para a localização atual do mouse
      line(0,0,mouse.x,mouse.y);
          }
SHIFFMAN (2012, pág. 41)
```

Código 5: comprimento de um retângulo representado por um retângulo

Esse exemplo dá, por meio de comandos, a magnitude (ou comprimento) de um vetor. Cabe aqui observar que o comando *rect*(0, 0, m, 10) se refere à localização inicial do retângulo que descreve o comprimento do vetor mouse, na barra superior da tela. O retângulo de base m e altura 10 que aparece na parte superior esquerda da tela de execução, representa o comprimento do vetor mouse em questão. (Ver figura 19).

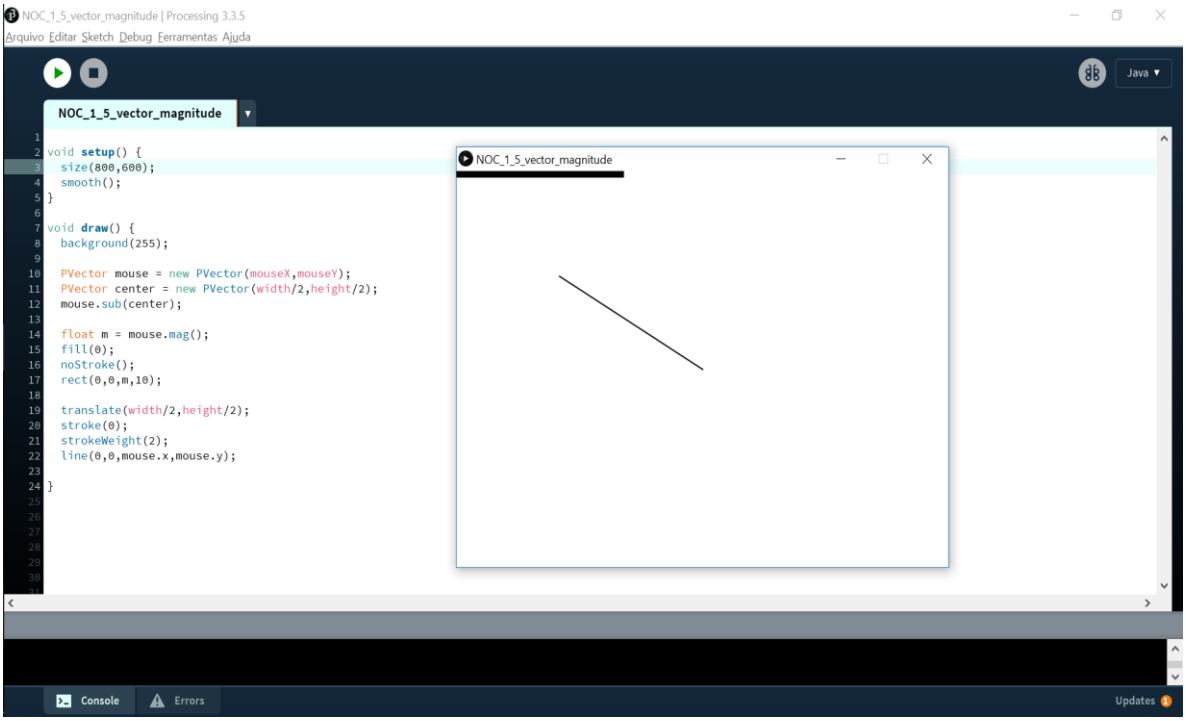


Figura 19. Magnitude de um vetor representado por retângulo de altura fixa. Fonte: autor da pesquisa.

2.6 VETORES NORMALIZADOS

Normalizar um vetor, é encontrar um vetor de magnitude 1 que tenha a mesma direção e sentido desse vetor. Em matemática esse processo é feito dividindo as coordenadas do vetor por seu módulo, dado na equação (1.3.4.1). De maneira semelhante, faremos isso programando por meio de comandos básicos. Veja o exemplo a seguir:

```
void setup(){
size (800,600);
smooth();
}
void draw() {
  background(255);
  PVector mouse = new PVector(mouseX,mouseY);
  PVector center = new PVector(width/2,height/2);
  mouse.sub(center);
      //Neste exemplo, depois que o vetor é normalizado, ele é
multiplicado por 50 para que seu comprimento na tela não varie. Note
que não importa onde o mouse esteja, o vetor terá o mesmo comprimento
(50) devido ao processo de normalização.
```

```
        mouse.normalize();
          mouse.mult(50);
          translate(width/2,height/2);
          line(0,0,mouse.x,mouse.y);    }
```

Código 6: normalização de um vetor

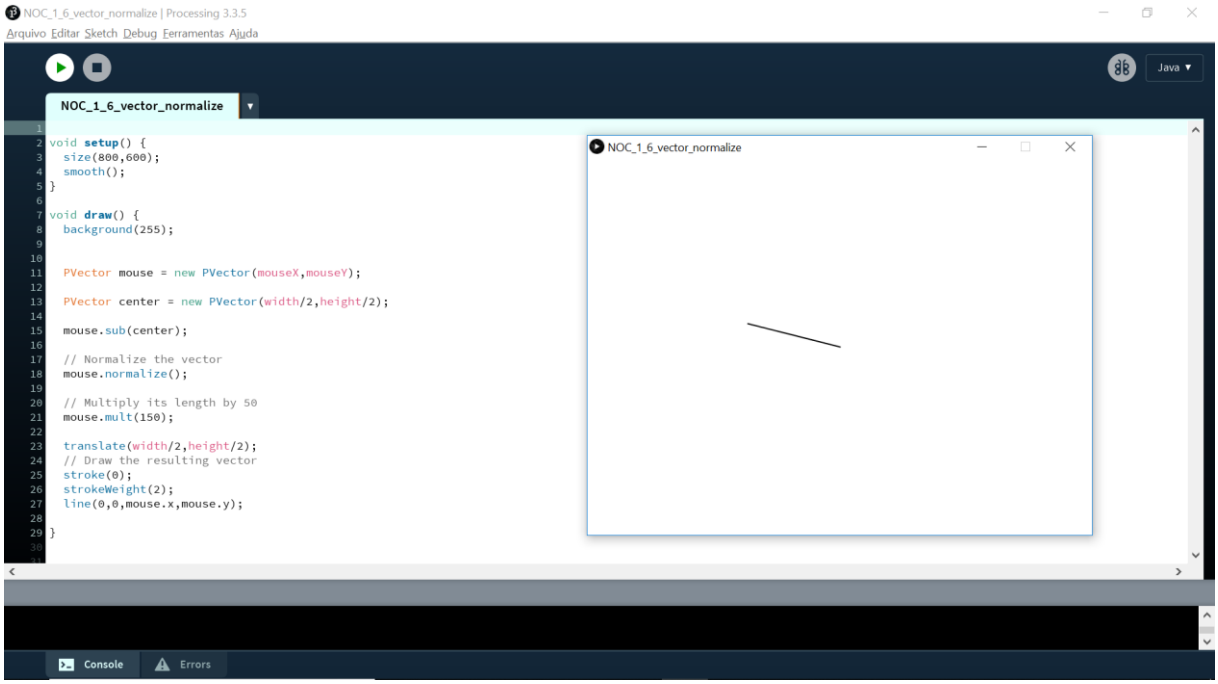


Figura 20. Vetor normalizado. Fonte: autor da pesquisa.

Nesse exemplo, os comandos usados são os mesmos usados nos dois exemplos anteriores, mas com uma pequena diferença, acrescentados dos comando normalizar e *smooth*() (esse comando suaviza o movimento do objeto). Quando se normaliza um vetor o deixa-se esse vetor com comprimento 1; assim, pode-se fazer com que esse vetor passe a assumir qualquer comprimento, bastando para isso acrescentar o comando *mouse.mult*(valor pelo qual será multiplicado o vetor normalizado). Nesse comando, o vetor assumirá comprimento igual ao valor que estiver entre os parênteses.

2.6.1 Aplicação da velocidade ao objeto

A partir de agora, serão tratados os comandos que introduzam na bola velocidade e aceleração. O que queremos fazer agora é avançar para encapsular toda a lógica do movimento dentro de uma classe. Nesse caso, cria-se uma classe genérica que descreverá um objeto se movimentando pela tela.

Um objeto possui dois dados: posição (posicao) e velocidade, ambos objetos *PVectors*. O objeto precisa se mover e precisa ser visto. Vamos implementar essas necessidades com as funções denominadas atualizar () e desenhar (). A função atualizar() atualiza instantaneamente a posição da bola por meio dos comando que introduzem os cálculos da adição de vetores, fazendo com que ela (a bola) permaneça em movimento, e a função desenhar() faz com que você visualize a bola em movimento na tela.

Todo o código de lógica de movimento será colocado em atualizar () e desenharemos o objeto em desenhar (). Vamos, arbitrariamente, decidir inicializar nosso objeto, dando-lhe uma localização aleatória e uma velocidade aleatória por meio do comando *random*. Concluindo, deve-se terminar a classe genérica, que será chamada de classe Movimento, incorporando uma função para determinar o que o objeto deve fazer quando atingir a borda da janela. Por enquanto, será feito algo bem simples, apenas envolvê-lo nas bordas. Observe o exemplo a seguir:

```
      // cria-se inicialmente a variável do tipo movimento
       Movimento variavel;
       void setup() {
         size(800,600);
         smooth();
       // usando a variável, cria-se o objeto da classe movimento
         variavel = new Movimento();
       }
       void draw() {
         background(255);
       // aplica-se ao objeto as funções constantes nas funções
atualizar, condição e desenhar
         variavel.atualizar();
         variavel.condicao();
         variavel.desenhar();
       }
       // a classe movimento encapsula toda a lógica da programação usada
       para dar movimento ao objeto
       class Movimento {
         PVector posicao;
         PVector velocidade;
```

```
       // a função movimento é a função construtora do objeto e guarda
seus estados  iniciais
         Movimento() {
           posicao = new PVector(random(width), random(height));
           velocidade = new PVector(random(-2, 3), random(-3, 2));
         }
       // a função atualizar guarda os comandos que permanecerão em
constante execução
         void atualizar() {
           posicao.add(velocidade);
         }
       // a função desenhar guarda os comandos que fazem com o objeto
seja desenhado na tela
         void desenhar() {
           stroke(0);
           strokeWeight(2);
           fill(127);
           ellipse(posicao.x, posicao.y, 48, 48);
         }
       // a função condicao guarda os comandos que fazem o objeto
       permanecer dentro da tela de execução
         void condicao() {
           if (posicao.x >= width) {
             posicao.x = 0;
           }
           else if (posicao.x <= 0) {
             posicao.x = width;
           }
           if (posicao.y >= height) {
             posicao.y = 0;
           }
           else if (posicao.y <= 0) {
             posicao.y = height;
           }
         }
       }
```

Código 7: movimento com velocidade

Alguns dos comandos acima já foram explicados em exemplos anteriores, dispensando comentá-los novamente. Nos limitaremos a comentar os comandos novos, ainda não vistos anteriormente.

O comando variavel = *new* Movimento() cria um objeto da classe Movimento. Uma classe descreve o comportamento, as ações e o estado (valores guardados) desse objeto. A variável criada nos dará acesso a todos os comportamentos inerentes ao objeto pertence à classe Movimento, nesse caso, temos uma programação orientada a objeto. Os comandos variavel.atualizar(), variavel.desenhar() e variavel.condicao(); significam que os comandos pertencentes a essas três funções serão aplicados ao objeto.

O comando posicao = *newPVector(random(width),random(heigth)*); significa que o vetor localização é um vetor que, por meio do comando random, assume posições aleatórias (não definidas) instantâneas dentro da tela de execução, e o comando velocidade = *newPVector(random(-2,3),random(-3,2))* significa que a velocidade com que a bola se movimenta na tela se alterna entre valores aleatórias que variam entre -2 e 3 em relação à horizontal e entre -3 e 2 em relação à vertical.

Cabe ressaltar, que todos os comandos podem estar dentro de uma mesma pasta, ou pode ser criada uma pasta adicional que conterão a classe movimento com seus respectivos comando. (ver figura 21).

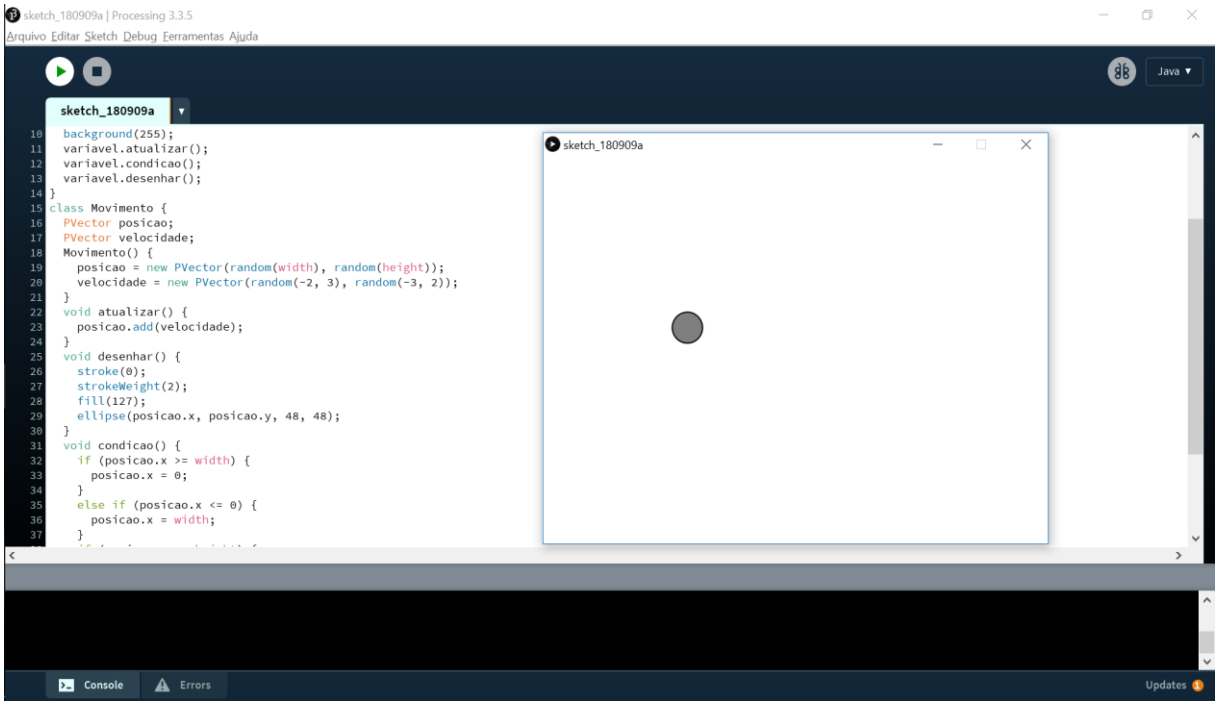


Figura 21. Bola em movimento segundo uma velocidade atribuída a ela. Fonte: autor da pesquisa.

2.6.2 Aplicação da aceleração ao objeto

O exemplo anterior será, agora, tratado de maneira a ser introduzida no objeto (bola) uma aceleração fazendo com que seu movimento se aproxime do mundo real ao nosso redor. A definição de aceleração que será usada aqui é a taxa de variação da velocidade, que por sua vez é a taxa de mudança de localização da bola. Faremos

com que a aceleração afete a velocidade e essa por sua vez, afete a localização da bola. Observe:

```
      Movimento o;
      void setup() {
        size(800,600);
        smooth();
        o = new Movimento();
      }
      void draw() {
        background(255);
        o.atualizar();
        o.condicao();
        o.desenhar();
      }
      // na classe movimento será encapsulada toda a lógica da
      programação usada para dar movimento ao objeto
      class Movimento {
        PVector posicao;
        PVector velocidade;
      // cria-se um novo vetor que determinará a taxa de variação da
velocidade, a aceleração
        PVector aceleracao;
        float limitevelocidade;
        Movimento() {
          posicao = new PVector(width/2, height/2);
          velocidade = new PVector(0, 0);
          aceleracao = new PVector(0.1, -0.01);
          limitevelocidade = 10;
        }
        void atualizar() {
          velocidade.add(aceleracao);
          velocidade.limit(limitevelocidade);
          posicao.add(velocidade);
        }
        void desenhar() {
          stroke(0);
          strokeWeight(2);
          fill(127);
          ellipse(posicao.x, posicao.y, 48, 48);
        }
        void condicao() {
          if (posicao.x > width) {
            posicao.x = 0;
          }
          else if (posicao.x < 0) {
            posicao.x = width;
          }
          if (posicao.y > height) {
            posicao.y = 0;
          }
          else if (posicao.y < 0) {
            posicao.y = height;
          }
        }
      }
```

Código 8: movimento com aceleração

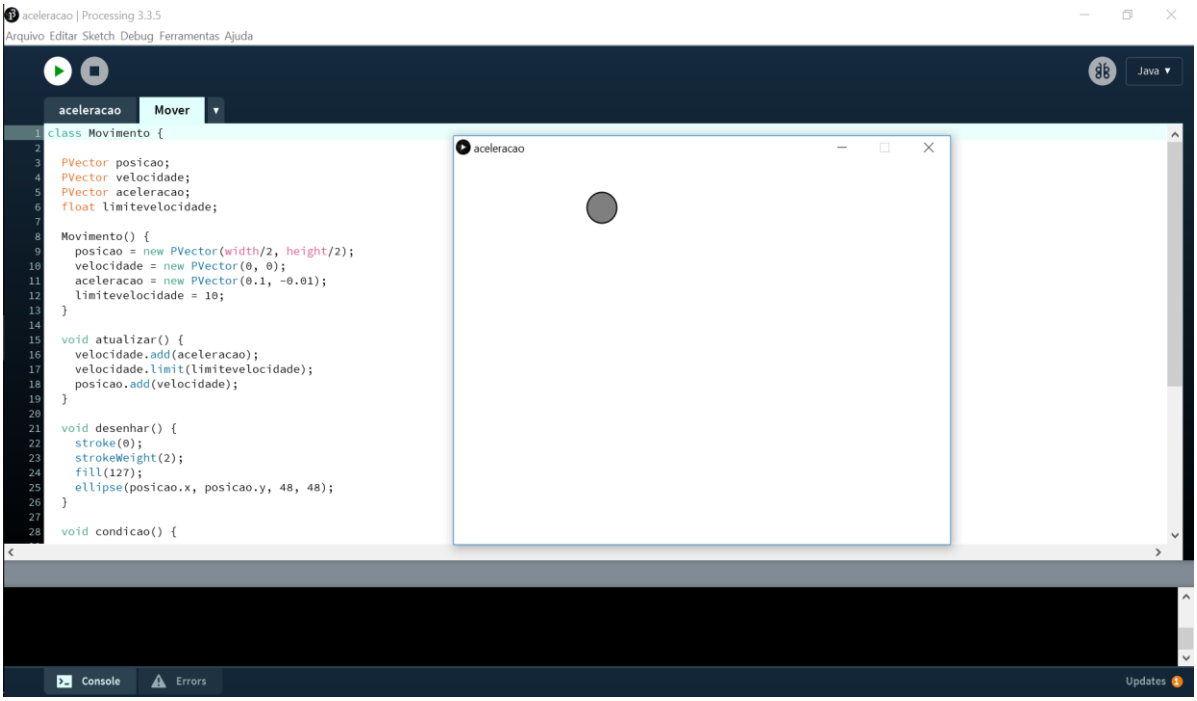


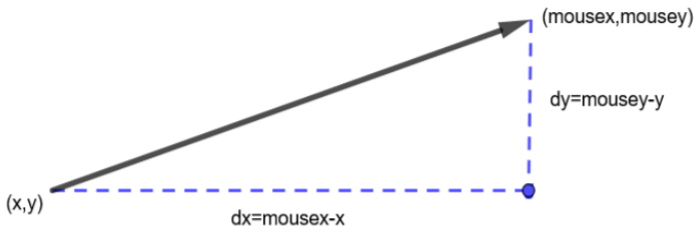
Figura 22. Bola se movimentando sob efeito de uma aceleração. Fonte: autor da pesquisa.

Veja às descrições dos comandos (novos). Inicialmente são acrescentadas as variáveis aceleração (Pvector aceleracao;) e limite do aumento da velocidade (*float* limitevelocidade;), além das já existentes. Quando se adiciona a velocidade à aceleração por meio do comando (velocidade.*add(*aceleracao*)*) colocando-o na função atualizar(), faz-se com que a velocidade da bola fique variando. Para cada nova atualização da soma do vetor velocidade com o vetor aceleração, temos uma nova velocidade para a bola em movimento, que por sua vez será adicionada novamente à aceleração, e obtida uma nova velocidade, e assim por diante; assim, a velocidade tende a aumentar a cada vez que esse comando é executado, e como a soma tende a aumentar, o movimento da bola também aumentará até atingir o limite imposto pelo comando limilevelocidade;.

De maneira análoga aos exemplos anteriores, os comandos constantes na função *void* condicao*()* fazem com que o movimento da bola se limite à tela, não ultrapassando as bordas da mesma.

2.6.3 Objeto seguindo o mouse

Por último, será trabalhada a interatividade com aceleração. Tentaremos mostrar algo mais complexo, calculando dinamicamente a aceleração de um objeto por meio de regras declaradas por meio do algoritmo acelere para o mouse. Quando se quer calcular um vetor com base em regras ou fórmulas, deve-se conhecer duas grandezas: magnitude e direção. Em se tratando de direção, o vetor aceleração deve apontar da localização do objeto em direção ao local do mouse. Veja a ilustração abaixo:



A ilustração mostra um *PVector* apontando da localização do movimentador até mouse. O comando que devemos usar deverá decidir com que rapidez o objeto deve acelerar em direção ao mouse. Se no exemplo anterior tomarmos a função *void* atualizar*()* como se segue abaixo, tem-se a bola seguindo o mouse. Lembrando que comando “limitevelocidade” diz com que velocidade a bola deve seguir o mouse. Veja:

```
Movimento o;
void setup() {
  size(800,600);
  smooth();
  o = new Movimento();
}
void draw() {
  background(255);
   o.atualizar();
   o.desenhar();
}
// Na classe movimento, teremos os comandos abaixo:
class Movimento {
  PVector posicao;
  PVector velocidade;
  PVector aceleracao;
   float limitevelocidade;
  Movimento() {
   posicao = new PVector(width/2,height/2);
    velocidade = new PVector(0,0);
// limita-se a velocidade com o objeto deve seguir o mouse por meio
de uma variável criada para esse fim
    limitevelocidade = 5;
  }
  void atualizar() {
    PVector mouse = new PVector(mouseX,mouseY);
    PVector aceleracao = PVector.sub(mouse,posicao);
```

```
        velocidade.add(aceleracao);
        velocidade.limit(limitevelocidade);
           posicao.add(velocidade);
         }
         void desenhar() {
           stroke(0);
           strokeWeight(2);
           fill(127);
           ellipse(posicao.x,posicao.y,48,48);
         }
       }
```

Código 9: objeto seguindo o mose

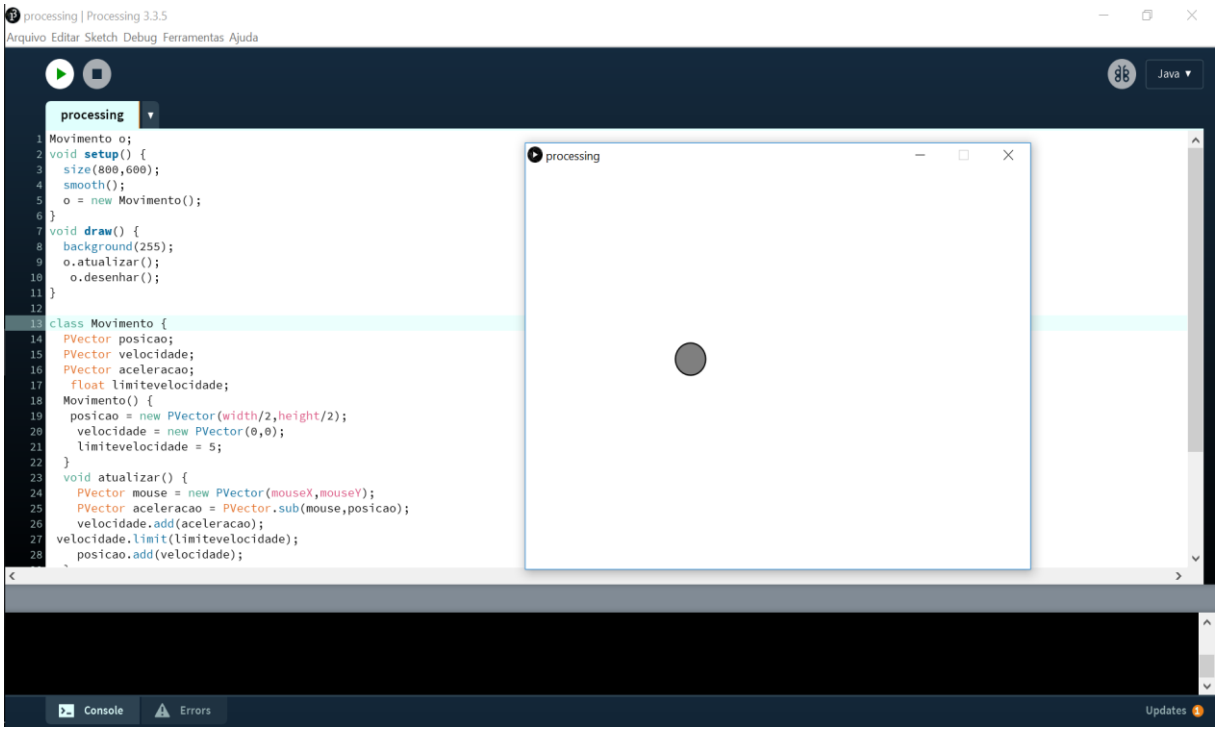


Figura 23. Seguindo o mouse. Fonte: autor da pesquisa.

No exemplo anterior, foi visto como calcular uma aceleração dinâmica por meio de um vetor apontando de um ponto na tela para o local do mouse. A partir de agora será formalizado o conceito de uma força e sua relação com aceleração.

2.7 FORÇAS E LEIS DE NEWTON

A definição de força que usaremos é mais formal que a usualmente usada no mundo real, e vem das leis do movimento de Isaac Newton. Uma força é um vetor que faz com que um objeto provido de massa se movimente.

A primeira lei de Newton, ou princípio da inércia, diz que um objeto em repouso permanece em repouso, e um objeto em movimento permanece em movimento uniforme se nenhuma força o afetar. No mundo de processamentos pode-se afirmar a primeira lei de Newton da seguinte forma: a velocidade de PVector de um objeto permanecerá constante se estiver em um estado de equilíbrio. A terceira lei de Newton, ou princípio da ação e reação, diz que para cada ação existe uma reação, ou seja, forças sempre ocorrem em pares, as duas forças são de igual intensidade, mas em sentidos opostos.

Em processamento, na terceira lei de Newton, se for calculado um PVector f que é uma força de um objeto A em um objeto B, deve ser também aplicada a força – Pvector.mult(f,-1) - que B exerce no objeto A. Nesse caso, está sendo feito com que a mesma força que o corpo A aplica em B, seja aplicada por B em A, na mesma direção, mas em sentidos opostos, causando assim, uma reação.

A segunda lei de Newton declara que força é igual ao produto da massa pela aceleração. Essa é a lei mais importante para essa nossa etapa da aplicação dos vetores. A aceleração é diretamente proporcional à força e inversamente proporcional à massa. Isso significa que se você empurrar um objeto, quanto mais forte você empurrar, mais rápido ele se moverá (acelerará). Quanto maior for o objeto, mais devagar ele se moverá. Vale lembrar que, como visto anteriormente, a posição é ajustada por velocidade e a velocidade por aceleração.

Peso vs. Massa

- A ***massa*** de um objeto é uma medida da quantidade de matéria no objeto (medida em quilogramas). Quanto maior a massa de um objeto, maior será a dificuldade de fazê-lo permanecer em repouso ou em movimento.

- ***O peso***, embora muitas vezes confundido com massa, é tecnicamente a força de atração entre esse objeto e o planeta. O peso varia de acorda com a massa do corpo e da gravidade local. Da segunda lei de Newton, podemos calculá-lo como tempos de massa da aceleração da gravidade (p = m * g). O peso é medido em newtons.

2.7.1 Gravidade aplicada a um objeto

O exposto acima será tratado por meio de exemplos que nos permitam compreender melhor:

```
// como visto anteriormente cria-se variável que será usada para cria o
objeto
      Movimento o;
      void setup() {
        size(800,600);
        smooth();
      // use a variável para criar o objeto
        o = new Movimento();
      }
      void draw() {
        background(255);
      //Faça duas forças.
         PVector vento = new PVector(0.01,0);
        PVector gravidade = new PVector(0,0.1);
      // aplique as forças ao objeto
        o.applyForce(vento);
        o.applyForce(gravidade);
      // aplique os commandos das funções atualizar, desenhar e condicao
ao objeto
        o.atualizar();
        o.desenhar();
        o.condicao();
      }
      //Nessa exemplo temos dois objetos novos (até aqui ainda não vistos),
      um vetor vento (vento) que está atuando na horizontal, e um vetor
      gravidade (gravidade) que nesse caso, atua na vertical. Na classe
      movimento seguem os comandos:
      class Movimento {
        PVector posicao;
        PVector velocidade;
        PVector aceleracao;
      //O objeto agora tem massa!
        float massa;
        Movimento() {
```

```
      //A massa dos objetos pode assumir qualquer valor, você é quem decide
      isso. Se chamarmos a massa apenas de m, as massas dos objetos
      assumirão valores diferentes, portanto, terão //dimensões
      diferentes.
          massa = 1;
      //  defina-se as coordenadas dos vetores posição, velocidade e
aceleracao
          posicao = new PVector(30,30);
          velocidade = new PVector(0,0);
          aceleracao = new PVector(0,0);
        }
      //Segunda lei de Newton.
        void applyForce(PVector forca) {
      //Receba uma força, divida em massa e adicione a aceleração.
          PVector f = PVector.div(forca,massa);
          aceleracao.add(f);
        }
        void atualizar() {
      //Movimentos que serão atualizados instantaneamente.
          velocidade.add(aceleracao);
          posicao.add(velocidade);
      //Agora adicione limpando a aceleração de cada vez! Esse comando faz-
      se necessário para que o vento e a gravidade não se alterem no decorre
      do movimento em execução.
          aceleracao.mult(0);
        }
        void desenhar() {
          stroke(0);
          fill(175);
      //Escalando o tamanho de acordo com a massa.
          ellipse(posicao.x,posicao.y,massa*16,massa*16);
        }
      //Um pouco arbitrariamente, estamos decidindo que um objeto é
      refletido quando atinge as bordas de uma janela.
        void condicao() {
          if (posicao.x > width) {
            posicao.x = width;
            velocidade.x *= -1;
          }
      else if (posicao.x < 0) {
            velocidade.x *= -1;
            posicao.x = 0;
          }
          if (posicao.y > height) {
      //Mesmo que você diga que não deve tocar diretamente na localização
e velocidade, há algumas exceções. Aqui está sendo feito isso como uma
maneira rápida e fácil de reverter a direção do nosso objeto quando ele
atinge a borda.
            velocidade.y *= -1;
            posicao.y = height;
          }
      }
```

Código 10: vento e gravidade aplicados ao objeto

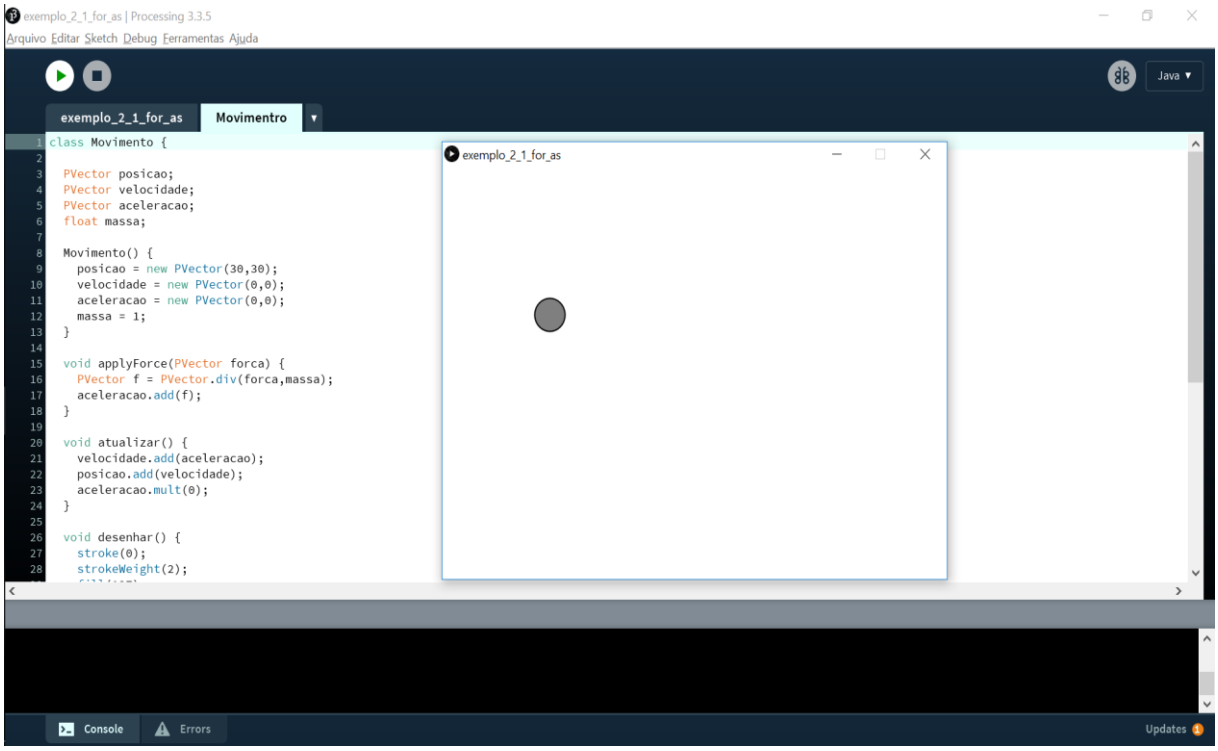


Figura 24. Objeto em movimento sob efeitos do vento e da gravidade. Fonte: autor da pesquisa.

Para dar continuidade à execução dos comandos, cria-se outra pasta (mas pode-se trabalhar com os comandos numa só pasta) clicando na seta que aponta para baixo, em seguida em nova aba, e nomeia-se a pasta que, nesse caso, será chamada de movimento. Nessa pasta deve ser acrescentada toda a classe movimento (acima). Serão descritos agora algumas funções e comandos novos. Dentro da classe Movimento acrescentou-se a variável massa (*float* massa). O valor dessa variável irá interferir no movimento da bola, pois quanto maior for a massa, mais lento será o movimento da bola. Isso decorre do fato de se estar trabalhando, agora, com forças, e para isso está sendo usado a segunda lei de Newton que diz que força é o produto da massa pela aceleração, no que resulta que a aceleração é o resultado da divisão da força pela massa. Por esse motivo, quando se aumenta o valora da massa, se diminui o valor da aceleração (movimento).

Em movimento(), deve ser definido o valor dessa massa. Uma função de papel importante é *void applyForce(PVector forca).* Essa função recebe e aplica ao objeto as forças vento e gravidade (*PVector forca*). Em seguida criamos o vetor f que é o resultado da divisão da força pela massa (f/m), isso também decorre da segunda lei de Newton (a=f/m).

Para perfeita execução do comando, essa força f deve ser adicionada à aceleração, quando feito isso, estará se adicionando à aceleração uma força na horizontal (vento) e uma força na vertical (gravidade). Esse comando faz com que a bola execute movimentos na horizontal e na vertical (movimento sinuoso, parabólico). A aceleração deve ser multiplicada por zero para que tanto a gravidade como o vento permaneçam constantes, pois como a velocidade é adicionada à aceleração e a posição à velocidade, a cada instante a aceleração deve ser zerada para não implicar na variação do vento e da gravidade, para que a velocidade permaneça constante.

2.7.2 Gravidade aplicada a mais de um objeto

No exemplo anterior, foram vistas forças atuando em um único objeto, mas pode-se expandi-lo adicionando mais objetos (quantos você quiser), por meio do comando *arrays*, que estarão sob a ação dessas forças. Veja:

```
      // usa-se o comando array, abaixo, para adicionar quantos objetos se
deseja
      Movimento[] variavel = new Movimento[20];
      void setup() {
        size(800,600);
        smooth();
      // os objetos são inseridos no ambiente virtual por índices i, que
      variam de 0 ao número máximo de objetos criados
        for (int i = 0; i < variavel.length; i++) {
      // os objetos são criados contend massa e localização inicial
          variavel[i] = new Movimento(random(0.1,4),0,0);
        }
      }

      void draw() {
        background(255);
        for (int i = 0; i < variavel.length; i++) {
      // é necessário definir vento e gravidade como vetores e criar as
      coordenadas dos vetores vento e gravidade
          PVector vento = new PVector(0.01,0);
          PVector gravidade = new PVector(0,0.1);
```

```
      // aplica-se as forças vento e gravidade aos objetos e os comandos
      que farão com que os objetos se movimentem, sejam desenhados na tela
      de execução e permaneçam dentro da mesma
          variavel[i].applyForce(vento);
          variavel[i].applyForce(gravidade);
          variavel[i].atualizar();
          variavel[i].desenhar();
          variavel[i].condicao();
        }
      }
```

Código 11: gravidades iguais aplicadas a vários objetos

Para trabalhar com dois ou mais objetos, devem-se iniciar os comandos com Movimento variaveis = *new* Movimento[quantidade de objetos], isso significa que está sendo criada uma variável, e essa variável representa os objetos da classe Movimento . A função *for (int i = 0; i* < variaveis*.length; i++)* que está dentro da função *void setup* diz que os objetos são inseridos por índices i, que variam de 0 à quantidade máxima de objetos inseridos na classe Movimento. No comando variaveis*[i]* = *new* Movimento*(random(0.1,4),0,0)*; limita-se a massa do objeto aleatoriamente entre 0,1 e 4 (qualquer valor nesse intervalo), e sua localização inicial é a posição (0,0). Vale observar que em variaveis[i] = *new* Movimento*(random(0.1,4),0,0)* as coordenadas são (massa=*random*(0.1,4), posição.x = 0, posição.y = 0)

A classe Movimento do exemplo de aplicação de forças com mais de um objeto, tem apenas um item a mais que o de com apenas um objeto, é o comando Movimento*(float m, float x , float y)*. Nele temos as variáveis massa *(float m)* e localização em relação aos eixos x e y *(float x, float y)*.

Depois que a matriz dos objetos é declarada, criada e inicializada, os demais códigos são mais simples. São dadas ao objetos as forças do ambiente e, a partir daí, pode-se aproveitar o espetáculo dos movimentos.

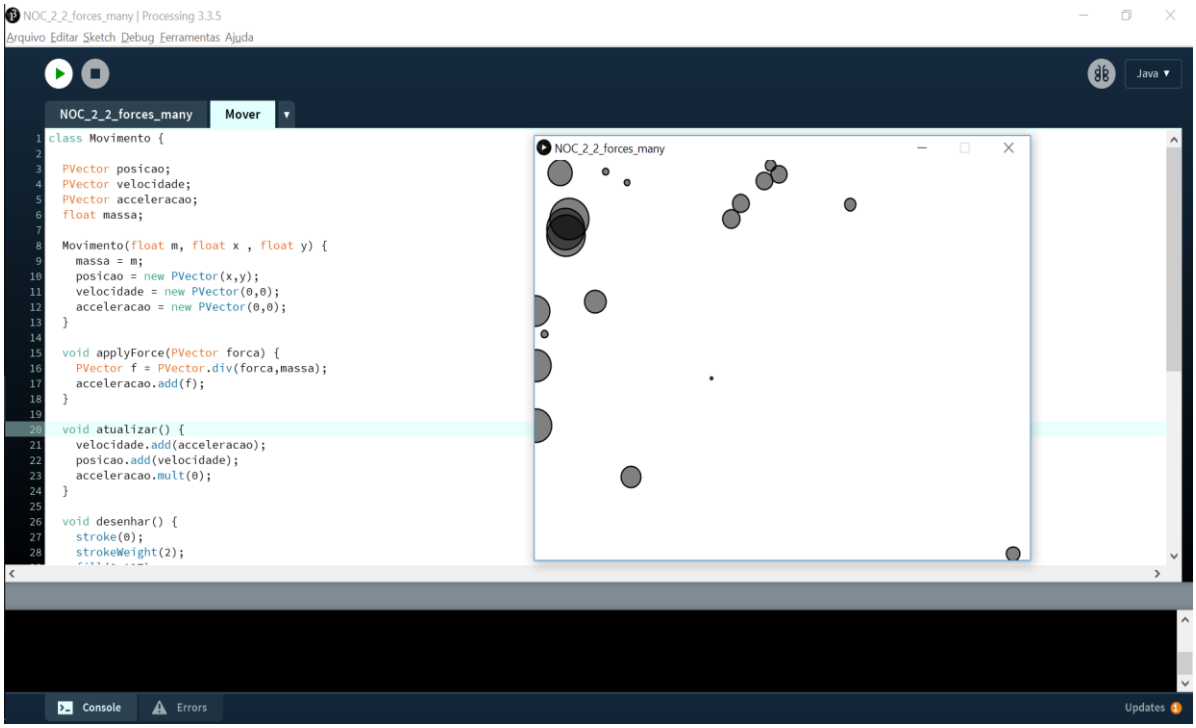


Figura 25. Gravidades iguais aplicadas a vários objetos. Fonte: autor da pesquisa.

2.7.3 Gravidade modelando uma força

Anteriormente, foi visto algo totalmente impreciso. Quanto menor a massa do objeto, mais rápido ele cai, isso de acordo com a segunda lei de Newton, pois quanto menor a massa, maior a aceleração. Mas isso não acontece no mundo real. Diz a lenda que em 1589, Galileu subiu a torre de Pisa e de lá soltou duas bolas de massas diferentes, ambas caíram com a mesma aceleração e atingiram o solo ao mesmo tempo. Por que isso?. Adiante será visto que a força da gravidade é calculada em relação à massa de um objeto. O exemplo abaixo, apresenta a gravidade escalonada por massa:

```
Movimento[] variaveis = new Movimento[20];
void setup() {
  size(800, 600);
  smooth();
  for (int i = 0; i < variaveis.length; i++) {
    variaveis[i] = new Movimento(random(1, 4), 0, 0);
  }
}
```

```
    void draw() {
      background(255);
    // São criados os vetores que representam as forças vento e gravidade,
    e a cada objeto são aplicadas essas forças vento e gravidade,
      for (int i = 0; i < variaveis.length; i++) {
        PVector vento = new PVector(0.01, 0);
        PVector gravidade = new PVector(0, 0.1*variaveis[i].massa);
        variaveis[i].applyForce(vento);
        variaveis[i].applyForce(gravidade);
    // A todos os objetos são aplicados os comandos constantes nas funções
    atualizar(), desenhar() e condicao().
    variaveis[i].atualizar();
        variaveis[i].desenhar();
        variaveis[i].condicao();
      }
    }
```

Código 12: Gravidades aplicadas a vários objetos variando de acordo com sua massa

A classe acima é idêntica à classe do exemplo anterior, exceto pelo fato de que por meio da função *PVector* gravidade = *new PVector(0, 0.1**variaveis[i].massa*);* está atribuído gravidades diferentes a cada objeto. Observe que a gravidade que atua sobre cada objeto é 0,1 vezes a massa desse objeto, e como as massas assumem valores aleatórios (variaveis[i] = *new* Movimento*(random(1, 4), 0, 0);)* a gravidade, da mesma forma, assumirá valores aleatórios, com variações proporcionais à variação da massa.

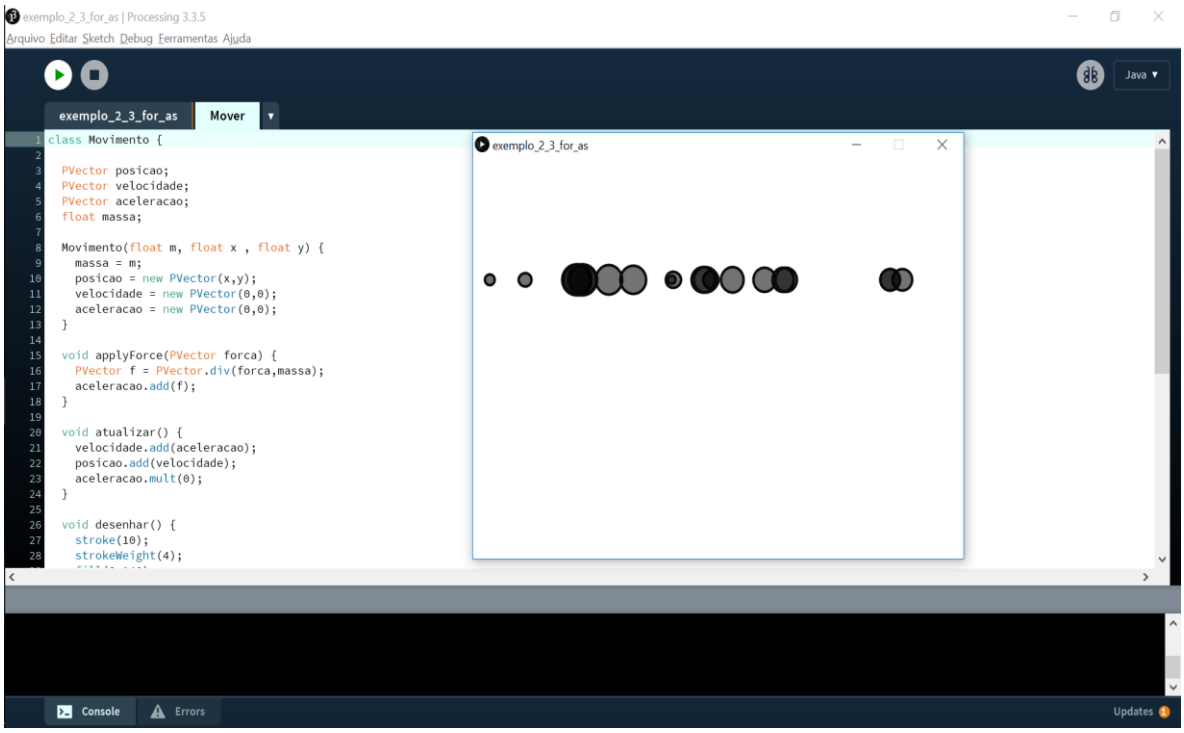


Figura 26. Gravidades diferentes aplicadas a vários objetos. Fonte: autor da pesquisa.

2.7.4 Atrito

Em física, atrito pode ser definido como sendo a resistência que os corpos opõem quando se movem uns sobre outros. O atrito é causado pelas irregularidades entre as superfícies em contato.

O atrito é uma força que faz com que a energia total de um sistema diminua quando um objeto está em movimento. Suponha que você está pilotando uma motocicleta, para pará-la você precisará acionar os freios. Os freios da moto usam o atrito para descarregar o movimento dos pneus. Em suma, sempre que duas superfícies entram em contato, há entre elas um atrito. (Ver figura 28).

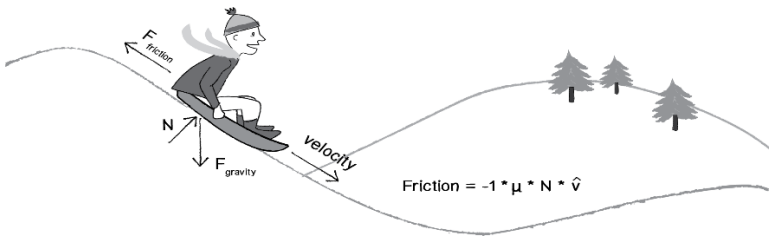


Figura 27. Atrito entre um corpo em movimento e sua superfície de apoio. Fonte: SHIFFMAN 2012.

Observe, na figura 27, que o vetor velocidade e o vetor atrito tem mesma direção e sentidos opostos. Esse fato é importante, pois será usado nos comandos para aplicação da força atrito no movimento de um objeto.

Para que se possa aplicar o atrito a um movimento, deve-se criar uma cópia do vetor velocidade por meio do comando PVector atrito = velocidade.*get()*; e multiplicá-lo por -1, isso fará com que esse vetor tenha sentido oposto ao do vetor velocidade, dificultando assim, o movimento do objeto. Para que o sentido do movimento não seja invertido, é necessário normalizar esse vetor e em seguida, aplicar o comprimento desejado a ele. Deve-se também, criar uma variável que atribua ao vetor um coeficiente de atrito, quanto maior for esse valor, maior será o atrito entre o objeto e a superfície sobre a qual está deslizando. Por ser trabalhado no mundo virtual, pode-se atribuir valores arbitrários a esse coeficiente, com base na quantidade de atrito que se quer simular. Observe o exemplo introduzido por meio dos comandos abaixo:

```
      Movimento[] variaveis = new Movimento[5];
      void setup() {
        size(800, 600);
        randomSeed(1);
        smooth();
        for (int i = 0; i < variaveis.length; i++) {
          variaveis[i] = new Movimento(random(1, 4), random(width), 0);
        }
      }
      void draw() {
        background(255);
        for (int i = 0; i < variaveis.length; i++) {
          PVector vento = new PVector(0.01, 0);
      // aqui estão sendo atribuídos valores aleatórios às massas dos
objetos
          PVector gravidade = new PVector(0, 0.1*variaveis[i].massa);
      // cria-se o coeficiente de atrito, valor que representará o atrito
      entre cada objeto e a superfície com a qual mantém contato
          float c = 0.05;
      // O vetor atrito é uma cópia do vetor velocidade, com mesma direção,
mas sentido contrário
          PVector atrito = variaveis[i].velocidade.get();
      // inverte-se o sentido do vetor atrito
          atrito.mult(-1);
      // torne a dimensão do vetor atrito igual a 1
          atrito.normalize();
      // aplique o coeficiente de atrito ao vetor atrito
          atrito.mult(c);
      // aplique as forças de atrito, vento e gravidade aos objetos, e os
      comandos contantes nas funções atualizar, desenhar e condição.
          variaveis[i].applyForce(atrito);
          variaveis[i].applyForce(vento);
          variaveis[i].applyForce(gravidade);
          variaveis[i].atualizar();
          variaveis[i].desenhar();
          variaveis[i].condicao();
        }
      }
```

Código 13: atrito

Observe que foram aplicadas, assim como no exemplo anterior, as forças vento e gravidade acrescidos da força atrito. O comando atrito*.mult(-1)*; significa que o vetor atrito dificulta o movimento do objeto, imprimindo uma força contrária a esse movimento, pois ele é uma cópia do vetor velocidade, mas com o sentido contrário.

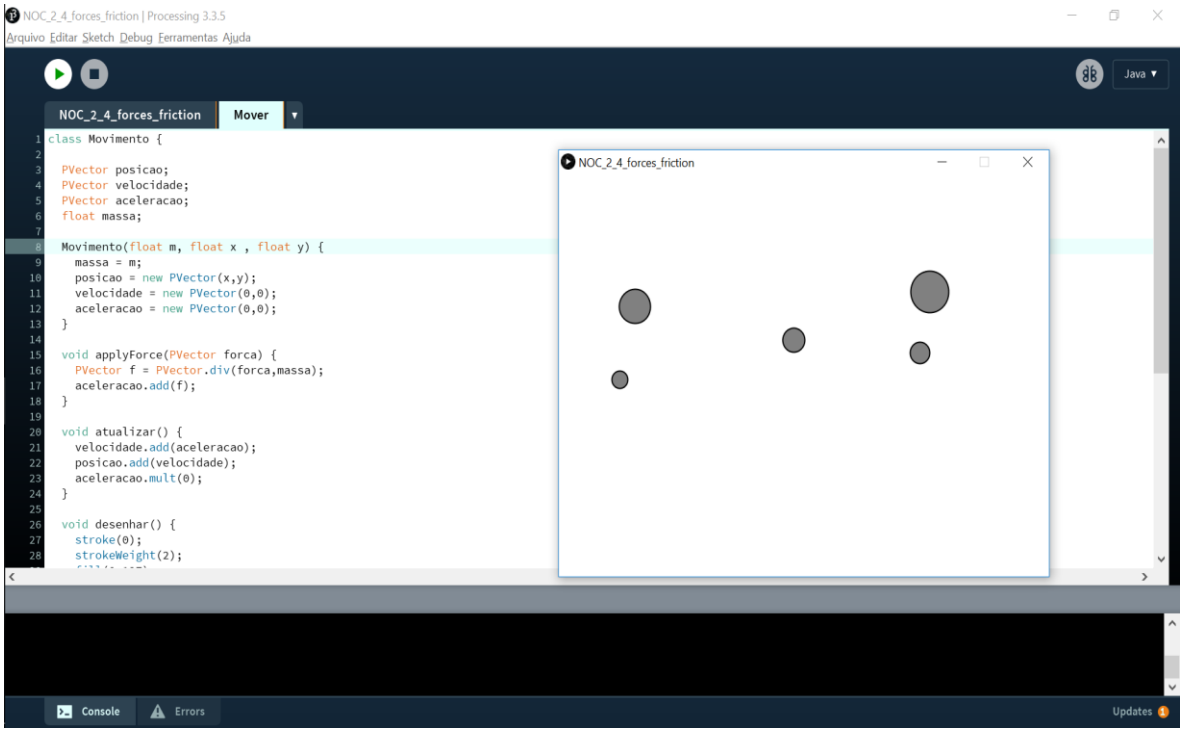


Figura 28. Atrito aplicado a objetos em movimento. Fonte: autor da pesquisa.

Com esse exemplo, será encerrada essa seção sobre aplicação de forças. Cabe aqui ressaltar que a aplicação de vetores, como forças que atuam sobre um corpo (objeto) vai além do exposto aqui. Essa aplicação foi exposta de forma sucinta por meio da ferramenta *processing* versão 3.3.5. Alguns comandos usados nesse programa podem ser escritos em português, outros, que fazem parte do pacote padrão do *processing* 3.3.5, não podem ser modificados.

A próxima etapa desse trabalho tratará da metodologia da aplicação, seguida dos resultados e discussões, e considerações finais.

3 METODOLOGIA

Os métodos de investigação utilizados na presente pesquisa foram o quantitativo e qualitativo, para Prodanov e Freitas (2013), este último tem como raízes filosóficas a fenomenologia e a interação simbólica, e o método qualitativo tem o método indutivo como método de análise, dos quais as características são descritas ou explicadas, ou seja, se foca no caráter subjetivo do objeto analisado em questão, estudando as suas particularidades e experiências individuais.

Na pesquisa qualitativa o conjunto inicial de categorias pode ser reexaminado e modificado sucessivas vezes, com vistas a obter ideias e resultados mais abrangentes e significativos. Demo acrescenta que

> Os movimentos em torno da pesquisa qualitativa buscam confrontar-se com os excessos da formalização, mostrando-nos que a qualidade é menos questão de extensão do que de intensidade. Deixá-la de fora seria deturpação da realidade.” (DEMO, 2000, p. 29).

A pesquisa quantitativa considera que tudo pode ser quantificável, o que significa traduzir em números opiniões e informações para classificá-las e analisá-las. Tem como raízes filosóficas o positivismo, o empiricismo e o lógico; e como coleta de dados, instrumentos manipulados como escala, testes, questionários e etc. Seu método de análise é o dedutivo, por meio do método estatístico. Marconi e Lakato acrescentam que

> As técnicas rigorosas de amostragem têm o objetivo de possibilitar a generalização das descobertas a que se chega pela experiência. Por sua vez, para que possam ser descritas quantitativamente, as variáveis relevantes são especificadas. Os diversos tipos de estudos experimentais podem ser desenvolvidos tanto "em campo", ou seja, no ambiente natural, quanto em laboratório, onde o ambiente é rigorosamente controlado. (MARCONI e LAKATO, 2003, p. 188)

A escolha dos métodos quantitativos e qualitativos aplicados a este trabalho, se devem; primeiro, ao fato de se necessitar quantificar (por meio dos gráficos) os

dados coletados pelos questionários fechados que tem como objetivo trabalhar as operações com vetores (muito citadas anteriormente) e; julgar a eficácia de métodos de ensino voltados a trabalhar a aplicação de conteúdos matemáticos por meio da contextualização dos mesmos, e sua relação com outras áreas.

A ordem de execução do presente trabalho seguiu uma sequência desenvolvida em quatro etapas, a saber:

1. Avaliação diagnóstica que permitiu fazer uma análise prévia sobre o conhecimento dos alunos acerca do conteúdo vetores;
2. Construção de exemplos que envolveram a aplicação de vetores para simular movimentos de objetos em ambientes virtuais;
3. Avaliação intermediária simulando movimentos de objetos por meio da manipulação dos comandos;
4. Avaliação final, baseada em um questionário para julgar a eficácia do desenvolvimento desse trabalho e sua importância para o aprendizado do aluno, e sua contribuição para o ensino do estudo dos vetores.

O presente trabalho teve como público 10 discentes do curso de licenciatura em matemática de uma instituição de ensino superior da cidade de São Raimundo Nonato-PI (os que já haviam cursado a disciplina geometria analítica) e foi realizado em dois encontros de 4h, cada.

O desenvolvimento da primeira etapa, deu ênfase às operações com vetores, em especial àquelas que são usadas para dar movimento a objetos em ambientes virtuais e às forças, que são grandezas vetoriais (velocidade, aceleração e gravidade). O estudo foi introduzido por meio de uma avaliação diagnóstica (Apêndice A) com os discentes que participaram do desenvolvimento desse trabalho. Os participantes foram avaliados por meio de sete questões objetivas quantitativas que versaram sobre operações de adição e subtração de vetores, módulo de um vetor, norma de um vetor, vetor unitário e multiplicação de um vetor por um escalar, e três questões subjetivas qualitativas abertas de cunho investigativo quanto ao julgamento de cada um sobre a importância do estudo e aplicação de vetores geométricos.

Essa avaliação investigativa permitiu, primeiramente, avaliar o nível de conhecimento de cada um sobre o conteúdo em questão, facilitando assim o desenvolvimento do trabalho, tendo em vista o direcionamento das dúvidas a serem

sanadas por meio de explicações básicas, mostrar aos alunos algumas áreas nas quais se podem aplicar vetores, e levá-los associar vetores com algo à sua volta. Para Yin (2005), “a investigação é um método que abrange tudo, desde o planejamento até os registros desenvolvidos pelos alunos durante a prática pedagógica fundamentada”.

Colhidos os dados, por meio da avaliação diagnóstica e investigativa, foi dado início à segunda etapa desse trabalho por meio da exposição do conteúdo aos 10 alunos que participaram do presente trabalho. Foram recapitulados o que é vetor e suas características, e as operações com vetores por meios de exemplos simples e detalhados. Feito isso, foram mostradas aos alunos algumas áreas de aplicação de vetores ou de grandezas vetoriais. Concluiu-se a segunda etapa quando os alunos passaram a trabalhar a aplicação dos vetores geométricos para simular movimentos de objetos em ambiente virtual por meio de comandos específicos usados para tal fim.

Para exposição e construção dos exemplos foi usado o programa *processing* 3.3.5. Esse programa usa a linguagem de programação Java. Quando da construção dos exemplos os alunos estudaram, de forma não aprofundada, o significado dos comandos: a criação dos objetos e das variáveis, a criação das funções e os comandos pertencentes a elas.

Como o objetivo do trabalho é mostrar a aplicação dos vetores para simular movimentos de objetos em um ambiente virtual, essa parte foi melhor exposta e detalhada por meio das descrições dos comandos que envolvem as operações com vetores, e como essas operações fazem com que o objeto se movimente.

Os discentes puderam compreender como a posição do objeto na tela do computador pode mudar várias vezes, fazendo-o se movimentar pela tela, por meio da soma de vetores (posição mais velocidade) e como sua velocidade pode aumentar ou diminuir por meio da mesma operação entre os vetores velocidade e aceleração. Foram também simulados os movimentos dos objetos sob ação de forças como vento (que atua na horizontal) e gravidade (que atua na vertical), tudo por meio de operações com vetores. Os discentes também estudaram como calcular a magnitude de um vetor por meio das operações de adição e subtração (realizados no ambiente virtual).

A terceira etapa da execução deu-se por meio de uma avaliação intermediária com cinco questões que permitiram aos alunos manipular o movimento do objeto na

tela. Os discentes substituíram comando de adição por subtração de vetores (em algumas operações), mudaram o valor de alguns comandos, como por exemplo, a velocidade do objeto quando esse colide com a borda da janela da tela de execução, e detalharam a função de alguns comandos como *velocidade.add(aceleracao), velocidade.limit(limitevelocidade) e posicao.add(velocidade).*

Por fim, os alunos foram submetidos a um questionário com perguntas abertas que lhes permitiram mostrar seu grau de compreensão sobre a função dos vetores dentro da lógica de programação usada para dar movimentos a objetos em um ambiente virtual; e, se o ensino da matemática, quando trabalhado de forma contextualizada, torna-se mais eficaz (Apêndice B).

A realização deste trabalho com alguns alunos do curso de licenciatura em matemática, permitiu observar que os discentes se envolveram mais efetivamente por aprender matemática, por conhecerem sua aplicação prática, e isso favoreceu, por contribuir positivamente, no ensino do conteúdo vetores. Alguns discentes puderam melhorar seus conhecimentos por meio de observações simples, como por exemplo, quando se somam sucessivas vezes dois vetores encontrando sempre um novo vetor, puderam notar que o conjunto de vetores resultantes dessa soma, forma o conjunto de vetores linearmente dependente, além de promover no objeto, dentro do ambiente virtual, um movimento retilíneo uniforme.

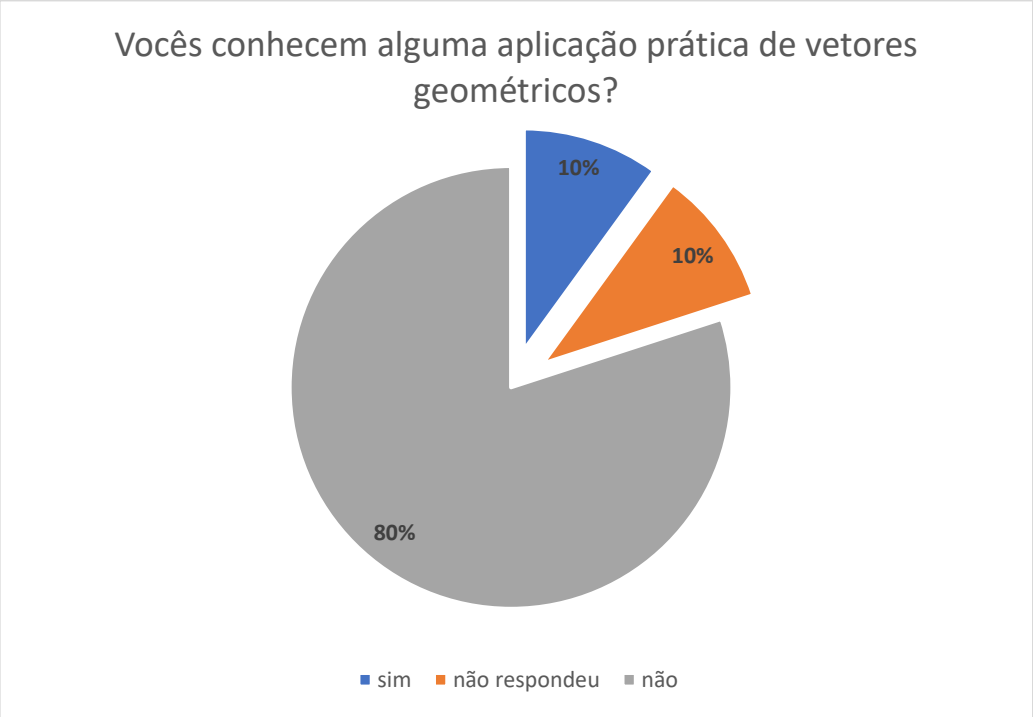
4 RESULTADOS E DISCUSSÕES

Durante o primeiro encontro foi, inicialmente, realizada uma avaliação diagnóstica com questões voltadas a operações com vetores (essas operações se fazem necessárias para que o aluno entenda como programar os vetores de modo a alcançar os objetivos deste trabalho). Essa avaliação também buscou identificar se aluno conhece alguma aplicação prática de vetores, se considera relevante o estudo dos vetores no curso de licenciatura em matemática e se durante o curso, na disciplina geometria analítica, foi exposto a ele onde seriam aplicados os conhecimentos adquiridos, assim como a relação do conteúdo vetores com outras áreas do conhecimento.

As respostas dadas pelos alunos permitiram direcionar o estudo dos vetores por meio de exemplos voltados a sanar as dúvidas iniciais que os mesmos tinham acerca de algumas operações e suas representações geométricas. Permitiu também mostrar algumas relações que esse ramo da matemática mantém com outras áreas e algumas de suas aplicações como cálculo da área de um paralelogramo e o volume de um paralelepípedo por meio do produto vetorial e do produto misto, respectivamente.

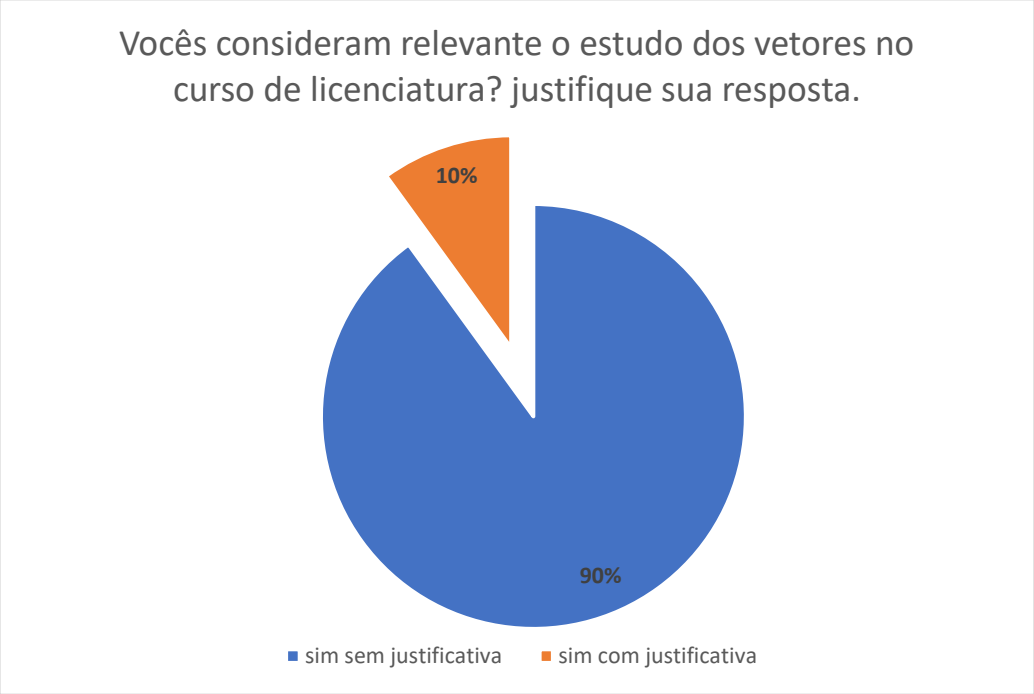
A pesquisa mostrou, inicialmente, que maioria dos discentes não conhecia nenhuma aplicação prática dos vetores, consideram relevante o estudo do conteúdo em questão, mas não souberam justificar o porquê. A maioria também relatou que durante o período em que cursaram a disciplina geometria analítica não foi mostrada nenhuma aplicação prática dos vetores ou sua relação com outras áreas. Os dados estão registrados nos gráficos a seguir:

Gráfico 1. Resposta dos discentes ao questionário sobre aplicação de vetores



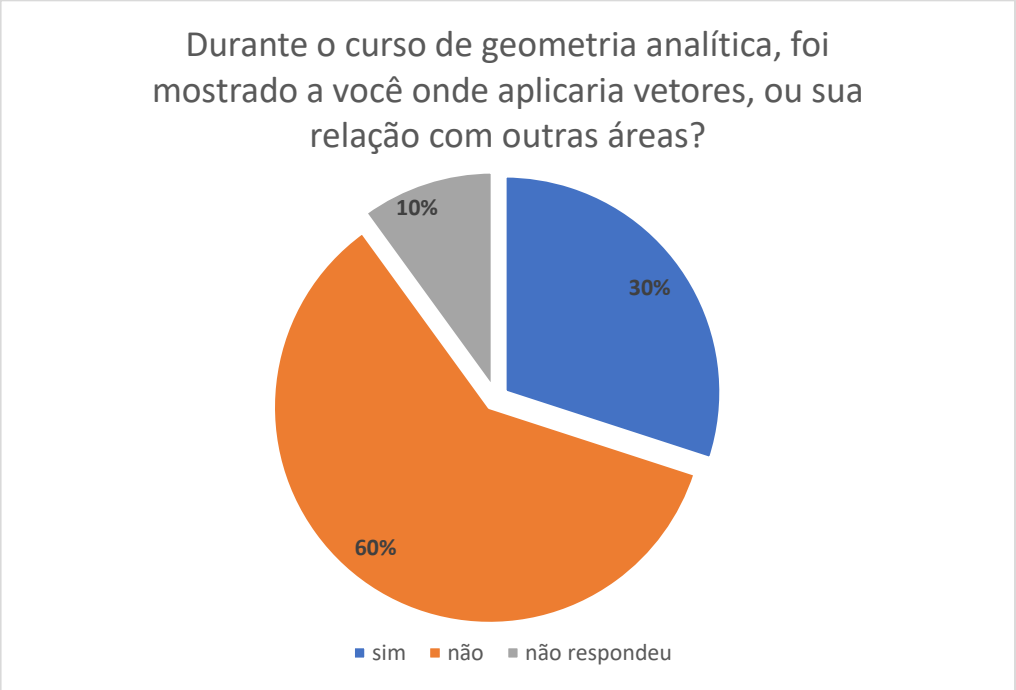
O Gráfico 1 tabula os dados da pesquisa referentes ao questionário aberto feito com os discentes. Dos dados obtidos, podemos constatar que a maioria não conhecia nenhum ramo da aplicação dos vetores geométricos, isso se deve ao fato dos mesmos terem sido submetidos a metodologias de ensino que não lhes permitiram associar a teoria com a prática. Um aluno relatou conhecer a aplicação dos vetores no campo da computação gráfica, por ter participado de um minicurso onde foi abordada tal aplicação, mas durante o curso de licenciatura em matemática, quando cursou a disciplina geometria analítica, não viu nenhuma aplicação prática do conteúdo vetores.

Gráfico 2. Reposta dos discentes ao questionário sobre a relevância do estudo dos vetores no curso de licenciatura em matemática.



Conforme o resultado apresentado no Gráfico 2, a maioria dos discentes do curso de licenciatura julgam importante o estudo dos vetores no curso de licenciatura, mesmo sem saberem justificar o porquê. Isso, certamente se deve ao fato de não conhecerem a aplicação prática dos mesmos, evidenciando que, por não conhecerem nenhuma aplicação dos vetores geométricos, não sabem relatar a importância de estuda-los, mas apenas que é importante fazê-lo.

Gráfico 3. Resposta dos discentes quando questionados se o professor da disciplina geometria analítica mostrou onde eles aplicariam vetores.



O Gráfico 3 mostra, por meio da tabulação dos dados, um dos motivos pelos quais muitos dos discentes do curso de licenciatura em matemática não conseguem associar a teoria com a prática no ensino da dessa disciplina e, consequentemente não levam essa prática para suas salas de aulas quando passam a exercer a função de docente.

As respostas ao questionário representado no gráfico 3, dadas por alguns discentes, divergem. O aluno X relatou que, durante as aulas de geometria analítica, não foi mostrada a aplicação prática de vetores, apenas a teoria. O aluno Y acrescentou que o professor da disciplina fez comentários sobre a aplicação de vetores na computação gráfica, mas não citou exemplos para mostrar tal aplicação. O aluno Z relatou em sua resposta que foi mostrada a aplicação de vetores, mas não descreveu essa aplicação.

As afirmações dos alunos diferem, pelo fato de alguns deles terem cursado a disciplina em questão, em turmas diferentes (alguns perderam a disciplina e tiveram que cursar novamente); assim, foram submetidos a metodologias de ensino que diferem uma da outra.

A presente pesquisa mostra uma das inúmeras formas de atividades alternativas para melhorar o ensino da matemática por meio da sua aplicação prática. Durante o desenvolvimento desse trabalho, ainda na primeira etapa descrita na metodologia, as questões voltadas a cálculos com vetores ajudaram o aluno a compreender o que acontece nos “bastidores” quando a bola está se movimentando pela tela.

O exemplos sobre o uso de vetores na programação para dar movimentos a objetos em ambientes virtuais, expostos aos discentes, levaram-nos a compreender qual o papel dos vetores dentro da programação Java, usados para dar movimentos aos objetos. O alunos puderam observar que através da adição dos vetores posição e velocidade (posicao.*add*(velocidade)), a bola ocupará uma nova posição sempre que esse comando for executado (várias vezes seguidas), e que ao objeto podem-se imprimir forças como a aceleração (por meio do comando velocidade.*add*(aceleracao)).

Por exemplo, quando se adicionam, na programação, o comando *(posicao.add(velocdade)),* e é dada ordem para que o programa execute esse comando várias vezes seguidas (dentro da função *draw*), até que seja fechado o programa, o que está sendo feito, é que está sendo criada uma nova posição para a bola sempre que esse comando é executado; ou seja, o vetor posição é somado ao vetor velocidade determinando assim, um novo vetor posição (onde bola estará localizada na tela); em seguida, essa nova posição é novamente adicionada à mesma velocidade gerando assim um novo vetor posição (e o objeto, no caso a bola, muda para essa nova localização); e assim por diante. Assim, o objeto se movimentará pela tela assumindo novas posições sempre que o comando acima for executado.

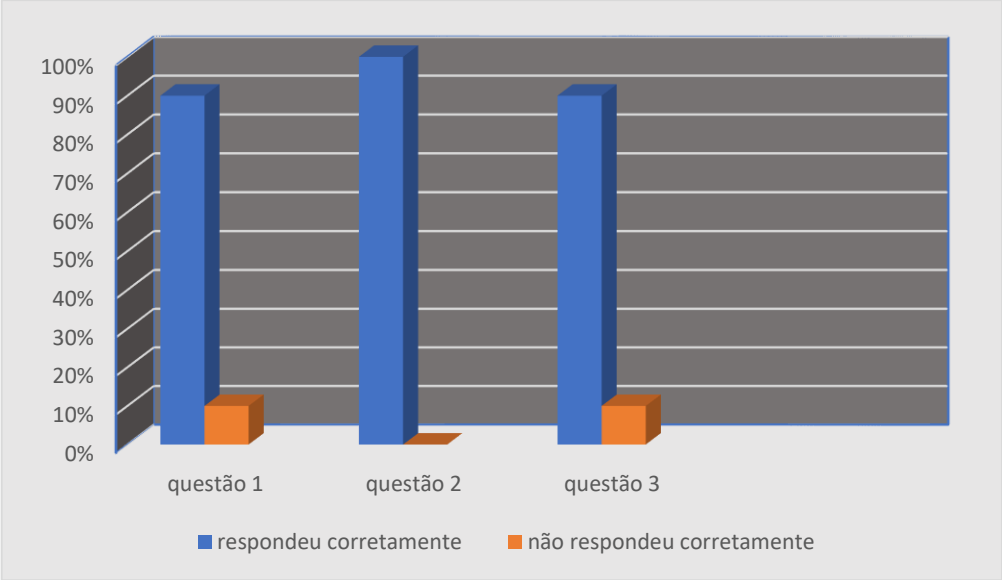
Por outro lado, quando se insere o comando *velocidade.add(aceleracao))* (também dentro da função *draw)* o que está sendo feito, de fato, é dando ordem ao programa para que faça a velocidade sofrer variações constantes seguidas, sempre que o comando for executado. Funciona da seguinte maneira: quando se soma o vetor velocidade ao vetor aceleração, obtém-se um novo vetor velocidade, quando o comando é novamente executado, obtém-se um novo vetor velocidade, e assim por diante. Dessa maneira a velocidade tende a variar (aumentar ou diminuir) infinitamente, e se essa variação pode ser limitada por meio de outro comando chamado, o *velocidade.limit(limitevelocidade).*

Os exemplos também trabalharam a criação de objetos, classes, variáveis e funções, todos os comandos necessários à perfeita execução dos movimentos.

A avaliação intermediária (Apêndice C) composta de três questões (a terceira com três itens) permitiu verificar a aprendizagem dos alunos acerca do conteúdo trabalhado. A primeira questão trata sobre o comando que determina qual a ação do objeto quando esse colidir com as bordas da tela de execução. A segunda questão, procura saber que mudança haverá com o movimento do objeto se, no comando *posicao.add(velocidade)*, o *add* for substituído por *sub.*

E a terceira, pede a descrição detalhada dos comandos *velocidade.add(aceleracao), velocidade.limit(limitevelocidade) e posicao.add(velocidade).* Os resultados estão descritos no gráfico 4.

Gráfico 4. Reposta dos discentes à pergunta voltada à manipulação dos comandos que fazem o objeto se mover pela tela.



O Gráfico 4 mostra que 100% dos alunos responderam corretamente à questão 2, e 90% responderam corretamente às questões 1 e 3. Vale salientar que os alunos que não responderam corretamente às questões 1 e 3, não o fizeram totalmente, ou seja, acertaram em parte as funções dos comandos. O aluno A relatou que, na questão 1, ao colidir com as bordas da tela de execução, o objeto reflete o sentido do seu movimento, não relatando que a velocidade irá dobrar a cada vez que acontecer essa colisão. O aluno B não descreveu corretamente, na questão 3, a função do comando velocidade. *limit*(limitevelocidade).

A execução do presente trabalho foi concluída com um questionário com duas perguntas abertas (Apêndice D). A primeira, questiona os alunos sobre a eficácia deste trabalho para compreensão da aplicação dos vetores abordada nesse projeto; e a segunda, se os mesmos julgam mais prazeroso, satisfatório e eficaz o ensino da matemática quando seguido de aplicações, aquilo que se está aprendendo em sala de aula, passando do campo teórico para o prático, não tratando a matemática como uma ciência isolada. Todos os discentes responderam ás questões contidas no referido questionário.

Levando em consideração os dados levantados durante a realização da presente pesquisa, pode-se observar que o ensino da disciplina matemática tem maior aproveitamento quando associadas a teoria com a prática, e os conteúdos são trabalhados de forma contextualizada, mostrando assim a relação entre a matemática e outros campos do conhecimento.

A contextualização e a interdisciplinaridade devem estar presentes em novas discussões e abordagens metodológicas, inseridas em novas propostas de ensino levadas para a sala de aula, que tendem a proporcionar ao aluno a oportunidade de organizar seu conhecimento, estruturar dados e informações e desenvolver seu pensamento crítico. Os PCNs, reforçam que

> Por outro lado, um conhecimento só é pleno se for mobilizado em situações diferentes daquelas que serviram para lhe dar origem. Para que sejam transferíveis a novas situações e generalizados, os conhecimentos devem ser descontextualizados, para serem contextualizados novamente em outras situações.
> (BRASIL, 1997, p. 26)

Verifica-se, pois, que o ensino da matemática, quando munido de práticas e métodos que permitam o aluno relacioná-la com outras áreas, conhecer sua aplicação, e, por meio disso, atuar como sujeito ativo nesse processo, produzem resultados significativamente positivos.

5 CONSIDERAÇÕES FINAIS

Este trabalho relata o resultado de uma pesquisa desenvolvida com alunos do curso de Licenciatura em Matemática de uma instituição da Rede Federal de ensino, da cidade de São Raimundo Nonato-PI. A pesquisa em questão, abordou o conteúdo vetores de maneira a trabalhar um dos ramos da aplicação de tal componente matemático, de forma a dinamizar o ensino. Esse tipo de abordagem é uma alternativa interessante, baseada na interdisciplinaridade para o ensino da matemática.

É relevante inovar a maneira de ensinar para que o aluno sinta-se motivado a aprender e busque construir seu conhecimento por meio de novas metodologias que conduzam a esse fim.

As etapas de desenvolvimento desse projeto, permitiram direcionar corretamente a realização das atividades propostas, e ajudaram a otimizar sua realização por meio das atividades que julgaram o conhecimento dos alunos sobre o conteúdo abordado, e aquelas que levaram os discentes a compreenderem qual a função das operações com vetores na programação usada para dar movimentos aos objetos dentro de um ambiente virtual.

Durante a execução das atividades, foi possível observar que os alunos apresentaram grande entusiasmo durante todo o desenvolvimento do projeto, visto que os mesmos sentiram-se envolvidos num processo empírico concreto, podendo assimilar melhor o conteúdo abordado no projeto em questão, associando a teoria com a prática.

Os resultados colhidos com a aplicação desse projeto nos permitem afirmar que métodos de ensinos em que o aluno atua como sujeito ativo, no processo ensino-aprendizagem, colaboram positivamente com o aprendizado do mesmo (o aluno).

Diante do exposto, levando em consideração a necessidade de se dinamizar o ensino da matemática, torna-se necessária a procura por métodos de ensino voltados ao enriquecimento das aulas por meios da aplicações práticas, para que o corpo discente entenda que a matemática está presente em tudo à sua volta, e para que possa percebê-la no seu dia-a-dia. Cabe aqui lembrar, que esse trabalho trata da

aplicação dos vetores de modo não aprofundado e traz uma introdução à lógica de programação usada em jogos de computadores.

Essa disquisição serve como base para pesquisas futuras sobre componentes matemáticos usados em programações usadas para dar ou auxiliar movimentos de seres em ambientes virtuais.

Para trabalhos futuros, a quem desejar pesquisar sobre tal tema, pode-se trabalhar o uso dos vetores em jogos de computadores com cálculos que produzam ações mais complexas, como por exemplo, a projeção ortogonal, ainda em duas dimensões, que pode ser usada em jogos de carrinhos de corrida para que, aqueles carros que não são controlados pelo jogador, não saiam da pista. Ou pode-se expandir a três dimensões com operações mais complexas, porém, com um leque maior de aplicações.

Outros componentes matemáticos, como a trigonometria, por exemplo, também tem uma vasta aplicação na computação gráfica, e são uma rica fonte nesta linha de pesquisa.

6 REFERÊNCIAS BIBLIOGRÁFICAS

ALMEIDA, Marcel Augusto Rosa de. O Tratado de Álgebra de John Wallis e suas relações com a álgebra britânica. Rio de Janeiro: UFRJ/IM, 2010.

BRASIL. Parâmetros Curriculares Nacionais. Disponível em <http://portal.mec.gov.br/seb/arquivos/pdf/ciencian.pdf>. Acesso em 03 jan. 2018.

CONCEIÇÃO, F.H.G. SANTOS, A.B. MENEZES, B.V. TORRES, N.L. A IMPORTÂNCIA DA APLICABILIDADE DA MATEMÁTICA NO COTIDIANO: Perspetiva do aluno Jovem e Adulto. Aracajú: FAMA, 2016.

DELGADO, Jorge. FRENSEL, Kátia; CRISSAFF, Lhaylla. Geometria Analítica. Coleção Profmat. 1.ed. Rio de Janeiro: Editora SBM, 2013.

FERRARO, G.N.; SOARES, P. A. T.; FOGO, R. Física básica: volume único. 4. ed. São Paulo: Atual, 2013.

GUIMARÃES, Osvaldo. PIQUEIRA, José Roberto. CARRON, Wilson. Física Mecânica 2ª ed. São Paulo: Editora ática, 2016.

HOWARD, A., RORRES C. Álgebra Linear com Aplicações. 8ª Ed. Porto Alegre: Bookman, 2001.

LAKATOS, E. M; MARCONI, M. A. Fundamentos de metodologia científica - 5. ed. - São Paulo. Atlas 2003.

LIMA, E. L.; CARVALHO, P. C. P.; WAGNER, E.; MORGADO A. C. A matemática do ensino médio. Volume 3. 6. ed. Rio de Janeiro: SBM, 2006. (Coleção Professor de Matemática)

LUÍS, Antônio Máximo Ribeiro da. ÁLVARES, Beatriz Alvarenga. GUIMARÃES, Carla da Costa. Física: contextos e aplicações, Ensino Médio. 2ª ed. São Paulo: editora Scipione. 2016.

MONTENEGRO, Fernando; PACHECO, Roberto. Orientação a Objetos. Editora Ciência Moderna, 1994.

Processing. (2013). (Versão 3.3.5) [Software]. Disponível em: <https://www.filehorse.com/download-processing-64/31136/>

PRODANOV, C. C; FREITAS, E. C. METODOLOGIA DO TRABALHO CIENTÍFICO: Métodos e Técnicas da pesquisa e do trabalho acadêmico. 2. Ed – Novo Hamburgo: Feevale, 2013.

QUEIROZ, Cecília; MOITA Filomena. As tendências pedagógicas e seus pressupostos. Campina Grande; Natal: UEPB/UFRN, 2007.

REIF, T.B. Programação de computadores: Uma proposta para o 9° ano do Ensino Fundamental, 2017. Disponível em: <https://sca.profmat-sbm.org.br/sca_v2/get_tcc3.php?id=150390755>. Acesso em 26/01/2018>.

RUMBAUGH, James; BLAHA, Michel; PREMERLANI, William; EDDY, Frederick. Modelagem e design orientados a objetos. Prentice-Hall, Inc., 1991.

SHIFFMAN, Daniel. The nature of code. Edição ilustrada. Editora D. Shiffman. 2012

STROUSTRUP, Bjarne. A linguagem de programação - Segunda edição. Addison-Wesley, 1993.

TERZIDIS, Kostas. Algorithms for Visual Design Using the Processing Language. Wiley Publishing, Inc, 2009

WIENER, Richard S.; PINSON, Lewis J. C. Programação Orientada para Objeto - Manual Prático e Profissional. Makron Books, 1991.

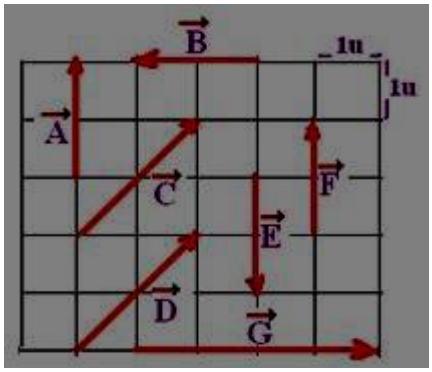
7 APÊNDICES

APÊNDICE A: Avaliação diagnóstica.

UNIVERSIDADE FEDERAL DO VALE DO SÃO FRANCISCO
MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL – PROFMAT
Mestrando: Francisco Alves dos Santos
Orientador: Dr. Alexandre Ramalho Silva
data: _____/_____/2018

AVALIAÇÃO DIAGNÓSTICA

1. Observe a figura abaixo e determine quais os vetores que:



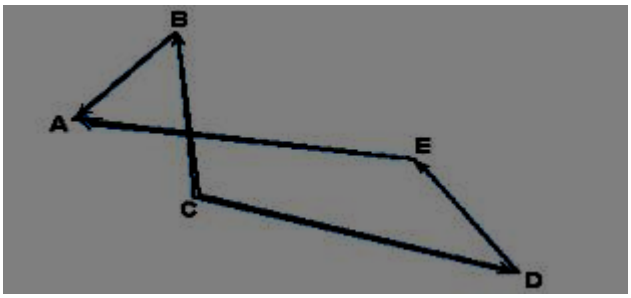
a) tem a mesma direção.
b) tem o mesmo sentido.
c) tem a mesma intensidade (módulo)
d) são iguais.

2) Quando dizemos que a velocidade de uma bola é de 30m/s, horizontal e para a esquerda,

Estamos definindo a velocidade como uma grandeza:

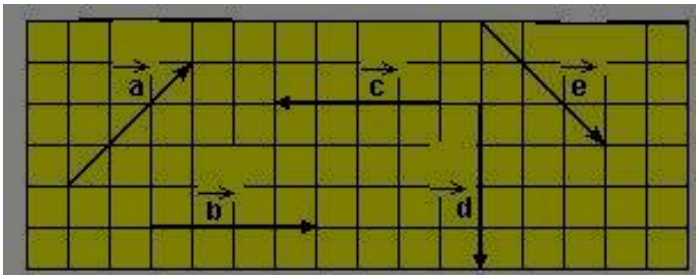
a) Escalar b) algébrica c) Linear d) vetorial

3) Analisando a disposição dos vetores BA, EA, CB, CD e DE, conforme figura a seguir, assinale a alternativa que contém a relação vetorial correta.



a) CB + CD + DE = BA + EA
b) BA + EA + CB = DE + CD
c) EA – DE + CB = BA + CD
d) EA – CB + DE = BA – CD
e) BA – DE – CB = EA + CD

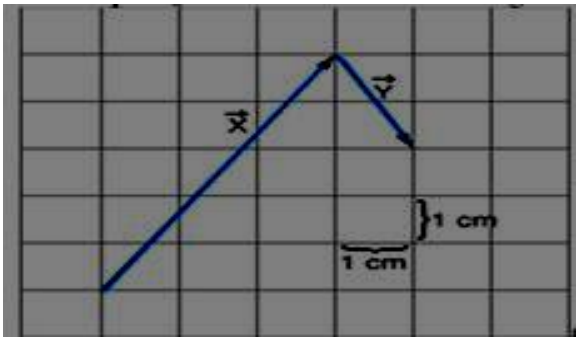
4) Dados os vetores “a”, “b”, “c”, “d” e “e” a seguir representados, obtenha o módulo do vetor soma:



$\vec{R} = \vec{a}+\vec{b}+\vec{c}+\vec{d}+\vec{e}$

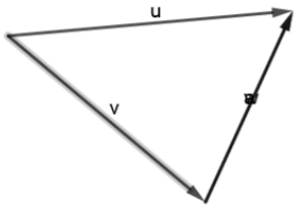
a) zero b) √20 c) 1 d) 2 e) √52

5) Na figura a seguir estão desenhados dois vetores ($\vec{x}$ e $\vec{y}$). Esses vetores representam deslocamentos sucessivos de um corpo. Qual é o módulo do vetor igual a $\vec{x}$ + $\vec{y}$?



a) 4cm b) 5cm c) 8cm d) 13cm e) 25cm

6) Para o diagrama vetorial abaixo, a única igualdade correta é:



a) u+v=w
b) v-u=w
c) u-v=w
d) v+w=u
e) w-v=u

7) Dados os vetores a = (2,-9) e b = (-1,7). Calcular:
a) a + b
b) a – b
c) 15a
d) -1a (represente geometricamente esse vetor)

8) Você conhece alguma aplicação de vetores geométricos? Em caso afirmativo, descreva-a.

9) Você considera relevante o estudo dos vetores no curso de licenciatura em matemática? Justifique sua resposta.

10) Durante o curso da disciplina geometria analítica, foi exposto a você onde aplicaria vetores, ou sua relação com outras áreas do conhecimento? Em caso afirmativo, cite-as.

APÊNDICE B: Avaliação intermediária

UNIVERSIDADE FEDERAL DO VALE DO SÃO FRANCISCO **MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL – PROFMAT** **Mestrando: Francisco Alves dos Santos** **Orientador: Dr. Alexandre Ramalho Silva** **data: _____/_____/2018** **Questionário intermediário**
1) Nos exemplo 1.2 se multiplicarmos a velocidade por -2 quando a bola colide com a borda da tela, o que acontecerá?
2) Se no exemplo 1.7, no comando *posicao*.add(velocidade) substituirmos add por sub, o que acontecerá, e que isso fará acontecer com o movimento da bola na tela?

3) No exemplo 1.8 descreva a função de cada comando a seguir:
a) velocidade.add(aceleracao);
b) velocidade.limit(limitevelocidade);
c) posicao.add(velocidade);

APÊNDICE C: Questionário final.

UNIVERSIDADE FEDERAL DO VALE DO SÃO FRANCISCO
MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL – PROFMAT
Mestrando: Francisco Alves dos Santos
Orientador: Dr. Alexandre Ramalho Silva
data: _____/_____/2018

Questionário final

1) O desenvolvimento do presente trabalho levou você a compreender qual a função dos vetores na programação usada para dar movimentos a objetos em um ambiente virtual?

2) Para você, o ensino da matemática torna-se mais prazeroso e satisfatório quando consegue relacionar o conteúdo estudado com outras áreas do conhecimento e consegue aplicar os conhecimentos adquiridos, passando do campo teórico para o prático?