Astronomia interativa: uma abordagem de software educativo para o ensino-aprendizagem

João Guilherme Cavalcante Alves Dias de Araújo, Vitor de Oliveira Silva

Universidade Cruzeiro do Sul (UNICSUL) – São Paulo – SP – Brasil

jgcadda@gmail.com, vitor.osilva02@gmail.com

Resumo: Neste trabalho, exploraremos o uso de representações digitais interativas no ensino de astronomia e física, com foco na simulação do princípio da gravidade e da energia escura em ação. O trabalho foi iniciado com uma análise da literatura acerca de simulações digitais de fenômenos físicos em portais acadêmicos. A partir desses achados e da mobilização do repertório adquirido no curso de Ciência da Computação, foi criado um software de uso didático em JavaScript a ser usado por professores de física e ciências e seus alunos em idade escolar para a apreensão de fenômenos astronômicos, buscando tornar acessível a visualização e a internalização de eventos e regras do mundo físico que podem ser, por vezes, abstratos e de difícil acesso.

Palavras-chave: Software. Simulação. Educação. Física. Astronomia. Gravidade.

Interactive astronomy: an approach to teaching and learning using educational software

Abstract: In this paper, we will investigate the use of interactive digital representations when teaching astronomy and physics, with a focus on simulating the principle of gravity and dark matter in action. The work began with an investigation of literature on digital simulations of physical phenomena that can be found in academic web portals. Using these findings and mobilizing the contents learned in the Computer Science undergraduate program, we created a didactic software in JavaScript to be used by physics and science teachers and their school students in order to grasp astronomic phenomena, seeking to increase the accessibility of the visualization and internalization of real-world events and rules, which may be abstract and difficult to access.

Keywords: Software. Simulation. Education. Physics. Astronomy. Gravity.

1. Introdução

Nossa intenção com o presente trabalho é entender o estado atual do ensino da ciência astronômica e, possivelmente, contribuir para sua longa trajetória. A astronomia é uma das ciências mais antigas de que se tem notícia – conforme demonstram indícios como o círculo de rochas de Nabta Playa, descoberto no Egito e feito há mais de sete mil anos (Betz, 2020) – e sua origem é a necessidade humana de entender a realidade em que vivemos e suas transformações. Um exemplo dessa demanda é a motivação para compreender as estações do ano e as suas mudanças.

Ao longo das eras, esse conhecimento passou por diversas transformações. As descobertas realizadas na ciência aumentaram nosso conhecimento a respeito da imensidão do universo, ao passo que o ensino a seu respeito tem, lamentavelmente, perdido espaço nas escolas. Essa perda de espaço tem como possíveis causas a falta de tempo para a abordagem adequada da disciplina, a escassez de materiais didáticos e a falta de formação adequada dos professores.

Há diversas ferramentas de representação que podem ser utilizadas no ensino; exemplos disso incluem imagens, maquetes e, mais recentemente, simulações baseadas em software. Tais simulações, sobretudo quando comportam recursos interativos, trazem diversas vantagens para o aprendizado científico, incluindo a possibilidade de visualizar fenômenos físicos aos quais não temos acesso a olho nu – como a relação entre a ocorrência de estações do ano e a angulação da terra em relação ao plano do sistema solar (Beserra, 2016) – e a comparação entre cenários partindo de um mesmo princípio físico – como a ação da gravidade no movimento de corpos com diferentes massas. Como nos ensina Neres (2017), a utilização de simulações pode ser de grande valia ao permitir a visualização de fenômenos astronômicos complexos, o desenvolvimento de hipóteses e a realização de experimentos por parte dos próprios alunos.

Softwares de simulação proporcionam uma incrível liberdade por criarem um mundo virtual no qual parâmetros, ângulos e escalas podem ser manuseados livremente. Esses programas podem auxiliar os discentes em assuntos que exigem visualizações tridimensionais ou progressões temporais, por vezes difíceis de representar por meios tradicionais e cruciais para o aprendizado da astronomia.

Considerando a presença cada vez maior da tecnologia na vida de todos, incluindo professores e alunos, essa ferramenta se tornou um recurso fundamental para a pedagogia, o que nos convida enquanto acadêmicos a entender sua utilização. Para Brianeza e Malacarne (2013), o uso da tecnologia se encaixa muito no terceiro momento pedagógico descrito por Angotti e Delizoicov (1994, apud Brianeza; Malacarne, 2013), que se refere à sistematização do conhecimento do aluno e é a etapa na qual a aprendizagem se consolida. Para analisar essa hipótese, as autoras realizaram atividades envolvendo astronomia com alunos do 8º ano do ensino fundamental, constatando que exercícios interativos e práticos com simulações de software, experimentos e jogos obtiveram um maior engajamento dos alunos em relação a outras atividades realizadas.

Vemos assim que as metodologias alternativas para o ensino de física vêm ganhando cada vez mais destaque. Porém, vale ressaltar que o interesse em formas mais

eficientes de se ensinar ciências não vem de hoje. Como afirma Ribeiro (2020), desde os anos 1970, com o lançamento do Sputnik I – o primeiro satélite artificial do planeta –, o interesse na matéria de física aumentou globalmente e foi impulsionado pela corrida espacial. Houve assim um grande foco no desenvolvimento dessa disciplina, que na época já possuía uma aproximação com os computadores, mas também foram identificados desafios no seu ensino, muitos dos quais se perpetuam até hoje: pouco conhecimento matemático dos estudantes e concepções de mundo alternativas que não seguem a lógica científica, além de métodos tradicionais de ensino que se concentram excessivamente na utilização de cálculos em vez de uma abordagem mais conceitual. Em resumo, há indícios de que o ensino de ciências é prejudicado quando se afasta das atividades práticas e quando há exagero em aulas expositivas sem uma descrição contextual que se aproxime mais dos alunos.

Nesse contexto, portanto, uma possível solução para esses problemas é a conciliação de metodologias mais tradicionais com metodologias complementares, como o ensino de ciências a partir de simulações por software, sendo esse o objetivo do presente trabalho.

No entanto, antes abordar a criação do software, é necessário definir os fundamentos e delimitações do projeto. No caso, optamos por focar na simulação da gravidade e da energia escura, visando tanto a educação quanto a divulgação científica e buscando esclarecer a complexidade da dinâmica entre esses dois fenômenos que, como apontado por Menezes (2009), desempenham papéis cruciais na cosmologia moderna. Nela, entende-se que o universo está se expandindo aceleradamente – conforme mostra a lei de Hubble –, desafiando assim a compreensão tradicional da gravidade e sugerindo a presença da enigmática energia escura, que parece se opor à atração gravitacional entre massas e desafia o conceito de um universo estático, além de pôr em questão as previsões das equações de campo de Einstein, que previam o colapso do universo sobre si próprio por conta da gravidade.

2. Revisão da literatura

Citaremos agora alguns trabalhos acadêmicos que investigam o ensino de física e ciências no âmbito escolar mediado por simulações geradas por software.

Em “A astronomia no ensino fundamental e o uso do software JClic” (Brianeza; Malacarne, 2013), as autoras descrevem os resultados do uso de diversos recursos pedagógicos para transmitir conceitos da astronomia para alunos do 8º ano do ensino fundamental, dentre eles o software JClic – utilizado para que os alunos elaborassem um jogo de memória com os assuntos em pauta. Os diversos jogos, experimentos e simulações empregados, por fomentarem a participação dos alunos, favoreceram a absorção de conceitos científicos, tornando o processo de aprendizagem mais lúdico e próximo dos alunos.

O artigo “O uso de simuladores no ensino de astronomia”, de Becker e Strieder (2011), trata das possibilidades pedagógicas do uso do software Stellarium, capaz de emular a observação de corpos celestes e fenômenos da astronomia. Nesse trabalho, há uma descrição de oficinas de educação continuada para professores do ensino

fundamental e médio com o propósito de capacitar os frequentadores das oficinas no uso do Stellarium para fins didáticos.

O trabalho “Ensino de astronomia com os softwares Stellarium e Celestia”, de Beserra e colaboradores (2016), oferece uma investigação dos fenômenos astronômicos capazes de serem simulados utilizando os softwares mencionados no título do artigo, considerando inclusive as concepções alternativas ou errôneas entre alunos – e até mesmo docentes – que podem ser evitadas com o estudo das emulações produzidas pelos programas. Este artigo é de interesse específico para a área da computação por incluir indicações a respeito da construção de scripts para o programa Celestia.

Em “Cosmologia: de Einstein à energia escura” (Alcaniz, 2007), discute-se as crescentes evidências da expansão acelerada do universo. Segundo o artigo, a compreensão clássica da gravidade vem sendo desafiada pela presença de uma energia que contraria a atração gravitacional. Essa energia, conhecida como energia escura, constitui aproximadamente 3/4 da densidade total de energia do universo e é a força motriz por trás de sua expansão acelerada. Este fenômeno enigmático, que se alinha com as observações de supernovas do tipo Ia, ressalta a importância de revisitar os princípios cosmológicos estabelecidos e considerar novas teorias que possam explicar a aceleração cósmica observada, potencialmente reformulando nosso entendimento sobre a estrutura e o destino do cosmos.

Encerrando esta breve seleção de artigos relevantes, mas sem esgotar o material acadêmico escrito a respeito do assunto, temos a dissertação de mestrado “O Stellarium como estratégia para o ensino de astronomia” (Neres, 2017). Partindo da teoria da aprendizagem significativa de David Ausubel, Neres (2017) descreve uma sequência de ensino e aprendizagem feita para professores de ensino médio que desejam utilizar o software Stellarium para o ensino da astronomia em aulas de física. Além de uma sugestão pedagógica embasada, o trabalho se destaca por incluir um manual para o uso do software e explicações de seus principais comandos para gerar simulações astronômicas.

3. Metodologia

Para o presente projeto, tem-se como metodologia a pesquisa bibliográfica e a construção de um software de simulação com finalidade educativa. A pesquisa bibliográfica será realizada nas plataformas Google Acadêmico e Portal CAPES a partir das palavras-chave “simulação”, “física”, “astronomia”, “educação”, “gravidade” e termos correlatos. Os artigos serão selecionados por critério de relevância, além de serem priorizadas publicações mais recentes.

O software, por sua vez, é desenvolvido em linguagem JavaScript, linguagem de programação para plataformas web, de modo que funcione em uma ampla gama de dispositivos (como computadores, notebooks e celulares tradicionais). Sua confecção obedece às seguintes etapas: a) desenvolvimento do software em HTML e CSS com as bases para a simulação, b) adição de uma interface de usuário com navegação fluida, c) criação de textos para elucidar os tópicos abordados, d) construção de um ambiente 2D com funções que se aproximam dos cálculos de física e e) inserção de funcionalidades interativas que engajam o usuário na exploração dos conceitos científicos apresentados.

4. Resultados

4.1 Primeiros passos

Seguindo o ciclo de vida de desenvolvimento de software (SDLC, Software Development Life Cycle), já cumprimos a primeira fase, a concepção, em que, como discutido anteriormente neste artigo, exploramos as potenciais melhorias no ensino de conceitos físicos complexos. Esta fase inicial envolveu uma análise das necessidades educacionais e a identificação de áreas nas quais uma ferramenta interativa poderia oferecer ajuda.

Prosseguindo para a segunda fase, a definição de requisitos, temos a descrição das especificações técnicas e pedagógicas que o software deve atender. Os requisitos funcionais estabelecidos foram a capacidade de simular a gravidade e a energia escura em 2D e a capacidade de ajustar parâmetros como massa, distância e constantes cosmológicas. Já os requisitos não funcionais são usabilidade, compatibilidade com diversas plataformas e tempo de resposta otimizado na simulação.

A terceira fase é o projeto, no qual traduzimos os requisitos coletados em um plano para o desenvolvimento do software. Aqui temos o projeto da arquitetura do software centralizada em tecnologias web front-end, usando HTML5 para estruturação do conteúdo, CSS para estilização e JavaScript como linguagem de programação. A modelagem dos dados foi projetada para representar corpos celestes, enquanto a interface do usuário foi modelada para permitir aos usuários manipular variáveis e visualizar os resultados da simulação em tempo real. O núcleo do software, implementado em JavaScript, gerencia a lógica das fórmulas de física da simulação, incluindo cálculos detalhados de gravidade e energia escura. Essas abordagens têm como foco a escalabilidade, a manutenção, a facilidade de execução e a capacidade de funcionar em uma ampla quantidade de dispositivos, como computadores desktop, notebooks e celulares.

A quarta fase, a de codificação, contempla a efetiva implementação do projeto. Essa etapa é crucial, pois é onde as ideias e planos estabelecidos nas fases anteriores são transformados em um código funcional.

4.2. Desenvolvimento HTML

Começamos pelo cabeçalho HTML, que contém as configurações mais gerais e aspectos fundamentais da página, como metadados, vinculação do CSS e JavaScript ao HTML, importação da fonte “Exo” do Google Fonts para aprimorar a estética e o título da página.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Cosmos em Equilíbrio: Gravidade e Energia Escura Desvendadas</title>
    <link href="https://fonts.googleapis.com/css2?family=Exo:wght@400;700&display=swap" rel="stylesheet">
    <link rel="stylesheet" href="styles.css">
    <script src="main.js" defer> </script>
</head>
```

Figura 1. HTML parte 01

Após a seção de cabeçalho, elaboramos o corpo do documento, que contém elementos essenciais para a estrutura e o conteúdo da página. Entre eles, <Canvas>, estabelecendo uma área destinada a gráficos criados via JavaScript que será importante posteriormente. A tag <img src...> insere uma imagem no documento (neste caso, com propósitos estéticos). As tags <h1> e <h2> definem títulos de texto, estabelecendo uma hierarquia clara e organizada, enquanto <p> é usada para parágrafos de texto, fornecendo explicações e informações. As tags e <section> são usadas para organizar e estruturar o conteúdo de uma página da web. Na imagem a seguir, são exibidas as primeiras <divs> como exemplos, enquanto as demais estão minimizadas para simplificar a visualização.

```
<body>

<section class="cabecalho">
    <img src="img/universo.png" alt="Imagem de fundo representando o espaço">
    <div class="Texto">
        <div>
            <h1>TGI</h1>
            <h2>Forças em Conflito: Uma Jornada Interativa pela Gravidade e Energia Escura</h2>
            <h3> &amp;amp; xxx</h3>
        </div>

        <div id="sobre-projeto">
            <p>O universo é um vasto e intrigante espaço de mistérios. Neste projeto, exploramos duas forças fundamentais que o moldam: a gravidade e a energia escura. Através de simulações
            interativas, você terá uma visão única de como elas atuam e influenciam a dança das galáxias e estrelas.</p>
        </div>
        <div id="sobre-gravidade">...
        <div id="simulacao-gravida...
        <div id="sobre-energia-esc...
        <div id="simulacao-energia...
        <div id="dinamica-forças">...
        <div id="simulacao-complet...
        <div id="consideracoes"> <...
        <div id="referencias"> <h2...
    </div>
</section>
</body>
</html>
```

Figura 2. HTML parte 02

4.3. Desenvolvimento CSS

Na folha de estilo CSS, os elementos “font-family: ‘Exo’, sans-serif”, “color: white” e “background-color: #000” são utilizados para estabelecer, respectivamente, a fonte “Exo” como padrão para todo o texto da página, a cor do texto como branca e a cor de fundo da página como preta. Essa combinação de estilos contribui para criar um design que remete ao tema espacial da página. Os outros elementos de estilo são utilizados para assegurar um posicionamento adequado e a clara visibilidade dos componentes da página, reforçando a coesão e a estética do layout.

```
body {
    font-family: 'Exo', sans-serif;
    color: white;
    margin: 0;
    padding: 0;
    background-color: #000;
}

.cabecalho {
    position: relative;
    text-align: center;
    padding: 5% 10%;
}

.cabecalho img {
    width: 100%;
    height: auto;
}

.Texto {
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    max-width: 800px;
}

.Texto h1, .Texto h2, .Texto h3 {
    background-color: rgba(0, 0, 0, 0.5);
    padding: 0.5em;
    border-radius: 5px;
}

.Texto h1 {
    font-size: 2.5em;
}
.Texto h2 {
    font-size: 1.5em;
}
.Texto h3 {
    font-size: 1em;
}
```

Figura 3. CSS parte 01

Outra parte importante do CSS é a consulta de mídia, “@media only screen”, utilizada para aplicar estilos específicos a uma página web com base nas características do dispositivo em que ela está sendo visualizada. Por fim, temos uma breve utilização do Canvas antes de desenvolvimentos posteriores em JavaScript.

```css
@media only screen and (max-width: 768px) {
    /* Ajustes para telas menores (tablets, telefones) */
    .Texto {
        top: 60%; /* Ajusta a posição para telas menores */
    }
    .Texto h1, .Texto h2, .Texto h3 {
        font-size: 90%; /* Diminui a fonte para melhorar a legibilidade */
    }
}

canvas {
    background: #000;
    display: block;
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
}
```

Figura 4. CSS parte 02

4.3. Desenvolvimento Javascript

Essa é a parte mais complexa do projeto, na qual implementamos diversas funções e cálculos cruciais para a simulação. Iniciamos com a definição de constantes-chave no código: “const G = 100”, “const canvas” e “const ctx”. Essas constantes propiciam, respectivamente, uma aproximação da constante gravitacional, a obtenção do elemento “canvas”, onde os gráficos serão renderizados, e a criação de um contexto de renderização 2D para o canvas.

```javascript
const G = 100;
const canvas = document.getElementById('canvas');
const ctx = canvas.getContext('2d');
ajustarTamanhoCanvas();

```

Figura 5. JS parte 01

Após a configuração inicial, o próximo passo é definir os objetivos específicos da simulação. Para isso, inicializamos um array que servirá como repositório dos objetos da simulação. Além disso, desenvolvemos uma função de criação, responsável por configurar cada objeto com características únicas. Nessa função, atribuímos uma cor aleatória a cada objeto para facilitar a identificação visual. Também definimos seus atributos, como a posição e o raio para a circunferência do objeto. A massa de cada objeto é determinada com base em seu raio, estabelecendo uma relação direta entre o tamanho visual do objeto e sua massa física. Concluímos essa etapa ao efetivamente criar esses

objetos na simulação, onde eles começarão a interagir de acordo com as leis físicas estabelecidas.

```
let esferas = [];

function criarEsfera(x, y, vx, vy, raio) {
    const massa = Math.PI * raio * raio;
    const cor = `rgb(${Math.floor(Math.random() * 255)}, ${Math.floor(Math.random() * 255)}, ${Math.floor(Math.random() * 255)})`;
    esferas.push({ massa, x, y, vx, vy, raio, cor, fundida: false });
}

criarEsfera(50, 800, 0, 0, 15);
criarEsfera(100, 50, 0, 0, 15);
criarEsfera(900, 0, 0, 0, 5);
criarEsfera(200, 190, 0, 0, 10);
criarEsfera(350, 10, 0, 0, 10);
criarEsfera(500, 550, 0, 0, 10);
criarEsfera(650, 10, 0, 0, 10);
```

Figura 6. JS parte 02

Nesse ponto, chegamos ao cerne da nossa simulação. A função “Gravidade” é responsável por aplicar as leis fundamentais da gravitação universal aos objetos simulados. Essa função percorre todas as esferas, calculando e aplicando as forças gravitacionais entre elas de acordo com a clássica equação da gravidade:

$$F = G * (m1 * m2) / r^2$$

Para cada par de esferas, a função calcula a distância entre elas e, utilizando a constante gravitacional G, determina a força gravitacional mútua. Essa força é então decomposta em componentes ax e ay (aceleração nos eixos x e y), que são usadas para ajustar as velocidades das esferas, replicando assim o movimento realístico que esperaríamos sob a influência da gravidade. A verificação de distância assegura que as forças só sejam aplicadas quando as esferas não estiverem muito próximas, evitando cálculos desnecessários ou resultados físicos imprecisos.

```
function Gravidade() {
    esferas.forEach((esfera1, i) => {
        esferas.forEach((esfera2, j) => {
            if (i !== j && !esfera1.fundida && !esfera2.fundida) {
                const dx = esfera2.x - esfera1.x;
                const dy = esfera2.y - esfera1.y;
                const distancia = Math.sqrt(dx * dx + dy * dy);

                if (distancia < 1) return;

                const forca = G * (esfera1.massa * esfera2.massa) / (distancia * distancia);
                const ax = forca * dx / distancia;
                const ay = forca * dy / distancia;

                esfera1.vx += ax / esfera1.massa;
                esfera1.vy += ay / esfera1.massa;
                esfera2.vx -= ax / esfera2.massa;
                esfera2.vy -= ay / esfera2.massa;
            }
        });
    });
}
```

Figura 7. JS parte 03

Temos também a função de colisão, que é uma parte vital da simulação, tratando da interação física entre as esferas. Ela percorre o array de esferas, verificando cada par

para determinar se ocorreu uma colisão. Quando a distância entre duas esferas é menor que a soma de seus raios, uma colisão é detectada. Nesse caso, as esferas colididas são substituídas por uma nova esfera, cujas propriedades – massa, raio, posição e velocidade – são calculadas com base na conservação da massa e do momento. O novo raio é determinado pela soma das áreas das esferas, enquanto a nova massa é a soma das massas das esferas colididas, seguindo os princípios da física. As novas posições e velocidades são calculadas para refletir o movimento combinado das esferas após a colisão. Essa abordagem não apenas garante uma simulação realista das colisões como também preserva as leis de conservação, mantendo a integridade física do sistema simulado.

```
function Colisao() {
    let esferasParaRemover = new Set();
    for (let i = 0; i < esferas.length; i++) {
        for (let j = i + 1; j < esferas.length; j++) {
            const esfera1 = esferas[i];
            const esfera2 = esferas[j];
            if (!esferasParaRemover.has(esfera1) && !esferasParaRemover.has(esfera2)) {
                const dx = esfera2.x - esfera1.x;
                const dy = esfera2.y - esfera1.y;
                const distancia = Math.sqrt(dx * dx + dy * dy);

                if (distancia < esfera1.raio + esfera2.raio) {
                    const novaMassa = esfera1.massa + esfera2.massa;
                    const novoRaio = Math.sqrt(esfera1.raio * esfera1.raio + esfera2.raio * esfera2.raio);
                    const novoVx = (esfera1.vx * esfera1.massa + esfera2.vx * esfera2.massa) / novaMassa;
                    const novoVy = (esfera1.vy * esfera1.massa + esfera2.vy * esfera2.massa) / novaMassa;
                    const novoX = (esfera1.x * esfera1.massa + esfera2.x * esfera2.massa) / novaMassa;
                    const novoY = (esfera1.y * esfera1.massa + esfera2.y * esfera2.massa) / novaMassa;

                    criarEsfera(novoX, novoY, novoVx, novoVy, novoRaio);

                    esferasParaRemover.add(esfera1);
                    esferasParaRemover.add(esfera2);
                }
            }
        }
    }

    esferas = esferas.filter(esfera => !esferasParaRemover.has(esfera));
}
```

Figura 8. JS parte 04

Para garantir que a nossa simulação desempenhe um bom papel na questão gráfica, foram criadas algumas funções ligadas ao canvas: a função “ajustarTamanhoCanvas” garante que a área de desenho se adapte dinamicamente ao tamanho da janela do navegador e a função “desenharEsferas” é responsável por renderizar cada esfera com um efeito visual detalhado. Utilizamos “ctx.createRadialGradient” para criar um gradiente radial que confere às esferas um aspecto luminoso e tridimensional, com cores que desaparecem suavemente até a transparência nas bordas. A função “desenharFundo” desempenha um papel crucial ao atualizar o canvas, aplicando uma camada de opacidade (definida por “ctx.fillStyle = 'rgba(0, 0, 0, 4)'”) para efetivamente limpar os rastros deixados pelas esferas em movimento. Essa técnica assegura que o canvas seja continuamente renovado, mantendo a clareza da visualização. Nossa abordagem assegura que a simulação sempre utilize o espaço disponível de forma otimizada, proporcionando uma experiência visual consistente, independentemente das mudanças no ambiente de visualização.

```
106     function desenharFundo() {
107         // Use uma opacidade que limpe os rastros de forma efetiva
108         ctx.fillStyle = 'rgba(0, 0, 0, 4)';
109         ctx.fillRect(0, 0, canvas.width, canvas.height);
110     }
111
112     function desenharEsferas() {
113     esferas.forEach(esfera => {
114
115         let gradiente = ctx.createRadialGradient(
116             esfera.x, esfera.y, 0,
117             esfera.x, esfera.y, esfera.raio
118         );
119         gradiente.addColorStop(0, esfera.cor); // Cor central brilhante e opaca
120         gradiente.addColorStop(0.2, esfera.cor); // Mantenha a cor brilhante até quase o final do raio
121         gradiente.addColorStop(1, 'rgba(0, 0, 0, 0)');
122
123         ctx.beginPath();
124         ctx.arc(esfera.x, esfera.y, esfera.raio, 0, Math.PI * 2);
125         ctx.fillStyle = gradiente;
126         ctx.fill();
127     });
128 }
129
130     function ajustarTamanhoCanvas() {
131         canvas.width = window.innerWidth;
132         canvas.height = window.innerHeight;
133     }
134
135     ajustarTamanhoCanvas();
136     window.addEventListener('resize', ajustarTamanhoCanvas);
```

Figura 9. JS parte 05

5. Considerações Finais

Como dito anteriormente neste artigo, a astronomia vem perdendo espaço nas escolas por diversas razões e o objetivo deste projeto foi criar uma solução tecnológica para o problema. Decidimos criar uma ferramenta gratuita e de fácil acesso para auxiliar o estudo da disciplina, com a esperança de ajudar professores a prender a atenção dos alunos e explicar alguns conceitos de forma visual e interativa.

Pesquisas bibliográficas foram feitas para deduzir a viabilidade e possível importância do projeto em plataformas como o Google Acadêmico. Diversos artigos encontrados e aqui citados corroboram com a noção de que métodos não-tradicionais de ensino, como o uso de softwares de simulação, jogos ou exercícios gamificados, geram mais engajamento por parte dos alunos. Com isso, concluímos que uma solução para a decadência da astronomia como área de estudo nas escolas é a mudança parcial da estratégia de ensino, normalizando o uso de ferramentas tecnológicas como softwares para complementar o ensino de astronomia. Nosso projeto, portanto, tem o potencial de ajudar a concretizar esse futuro.

Alguns obstáculos foram encontrados durante a construção do software proposto, sendo um deles a complexidade da ferramenta Unity 3D, considerada inicialmente pelos autores para o desenvolvimento do software, e a dificuldade que computadores mais antigos teriam para executar uma ferramenta criada com a Unity, o que seria contraprodutivo em relação ao objetivo deste projeto. A solução para esse obstáculo foi o abandono da ideia de usar a Unity 3D e a adoção de linguagens de programação mais simples de usar – neste caso, HTML e CSS.

Referências

Alcaniz, J. S. **Cosmologia: de Einstein à energia escura**. Com Ciência, 10 ago. 2007. Disponível em: https://www.comciencia.br/comciencia/handler.php?section=8&edicao=27&id=306. Acesso em 18 nov. 2023.

Becker, W. R.; Strieder, D. M. O uso de simuladores no ensino de astronomia. **Encontro nacional de informática e educação**, v. 2, p. 398, 2011. Disponível em: https://www.yumpu.com/pt/document/read/45328555/o-uso-de-simuladores-no- ensino-de-astronomia-inf-unioeste. Acesso em 06 jun. 2023.

Beserra, D. W. et al. Ensino de astronomia com os softwares Stellarium e Celestia. In: **Congresso Internacional de Tecnologia na Educação**, 10, 2012. Recife: Senac/DR/PR, 2012. Disponível em: https://www.researchgate.net/publication/303920019_ENSINO_DE_ASTRONOMIA_COM_OS_SOFTWARES_STELLARIUM_E_CELESTIA. Acesso em 06 jun. 2023.

Betz, E. **Nabta Playa: The world's first astronomical site was built in Africa and is older than Stonehenge**. Astronomy, 20 jun. 2020. Disponível em: https://astronomy.com/news/2020/06/nabta-playa-the-worlds-first-astronomical-site-was-built-in-africa-and-is-older-than-stonehenge. Acesso em 06 jun. 2023.

Brianeze, S. R.; Malacarne, V. A Astronomia no Ensino Fundamental e o uso do software JCLIC. In: **O professor PDE e os desafios da escola pública paranaense 2012**, v. 01, p. 30-46. Disponível em: http://www.diaadiaeducacao.pr.gov.br/portals/cadernospde/pdebusca/producoes_pde/2012/2012_unioeste_cien_artigo_silvana_regina_brianeze.pdf. Acesso em 06 jun. 2023.

Neres, L. B. **O Stellarium como estratégia para o ensino de astronomia**. Ilhéus (BA): Dissertação de Mestrado – Universidade Estadual de Santa Cruz, 2017. Disponível em: http://nbcgib.uesc.br/mnpef/images/Arquivos/Dissertao-para-divulgao-em-biblioteca-digital--leomir.pdf. Acesso em 06 jun. 2023.

Ribeiro, J. P. M. Filmes e softwares educacionais no ensino de Física: Uma análise bivariada. In: **Research, Society and Development**, v. 9, p. 36984998, 2020. Disponível em: https://www.researchgate.net/publication/342598040_Filmes_e_softwares_educacionais_no_ensino_de_Fisica_Uma_analise_bivariada. Acesso em 06 jun. 2023.