UNIVERSIDADE FEDERAL DO PAMPA

Adriano Tolfo Dotta

Uso de um chatbot generativo como intermediário para potencial melhoria na experiência do usuário em jogos de Ficção Interativa

Alegrete
2024

Adriano Tolfo Dotta

Uso de um chatbot generativo como intermediário para potencial melhoria na experiência do usuário em jogos de Ficção Interativa

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Doutor Marcelo Resende Thielo

Coorientador: Prof. Mestre Jean Felipe Patikowski Cheiran

Alegrete
2024

ADRIANO TOLFO DOTTA

USO DE UM CHATBOT GENERATIVO COMO INTERMEDIÁRIO PARA POTENCIAL MELHORIA NA EXPERIÊNCIA DO USUÁRIO EM JOGOS DE FICÇÃO INTERATIVA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Com- putação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Dissertação defendida e aprovada em: 05 de julho de 2024.

Banca examinadora:

_____

Prof. Dr. Marcelo Resende Thielo

Orientador

UNIPAMPA

_____

Prof. Me. Jean Felipe Patikowski Cheiran

Co-orientador

UNIPAMPA

_____

Prof. Dra. Amanda Meincke Melo

UNIPAMPA

Prof. Dra. Brenda Salenave Santana

UFPEL

Assinado eletronicamente por **MARCELO RESENDE THIELO**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 09/07/2024, às 18:37, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **Brenda Salenave Santana**, **Usuário Externo**, em 10/07/2024, às 09:39, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **JEAN FELIPE PATIKOWSKI CHEIRAN**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 10/07/2024, às 15:06, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **AMANDA MEINCKE MELO**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 10/07/2024, às 20:20, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

A autenticidade deste documento pode ser conferida no site https://sei.unipampa.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1475175** e o código CRC **D027674B**.

Este trabalho é dedicado aos apaixonados por jogos que,
independente da situação, espalham sua paixão em toda
e qualquer oportunidade com suas palavras e ações.

AGRADECIMENTOS

Por mais que este trabalho seja de minha autoria, ele jamais poderia ser concluído sem o apoio de inúmeras pessoas que não permitiram que me abalasse, desistisse ou pensasse que não era capaz.

Agradeço aos meus orientadores, os quais fizeram grande parte deste trabalho de acreditar no meu potencial e corrigir e lapidar minhas imperfeições.

Agradeço à minha mãe e grande pilar Denise, que desde o início sempre me apoiou em todos os momentos e esteve presente nas minhas conquistas.

Agradeço à minha irmã Liandra, que sempre me conduziu e deu suporte referente ao meio acadêmico com sua experiência de estudante.

Agradeço à minha companheira de vida Cláudia, a qual foi e ainda é um enorme pilar para meu fortalecimento pessoal e mental, felicitando-me pelos acertos e indicando os meus erros para que me torne alguém ainda melhor.

Por fim, agradeço ao meu pai João que independente de onde estiver, que esteja olhando por mim em meu caminhar. Obrigado por ter sido o responsável por grande parte de quem sou hoje.

RESUMO

Nos anos 70 e 80 surgiram os jogos de ficção interativa, que revolucionaram o mundo dos *games* ao permitirem que os jogadores conversassem com a máquina e tomassem suas próprias decisões, escolhendo seus caminhos e decidindo suas ações. Essa liberdade cativou os jogadores, porém, na época, estes jogos eram limitados por *scripts* programados, os quais aceitavam apenas palavras presentes no dicionário do código do jogo. No entanto, a evolução da tecnologia abriu espaço para melhorias nesse sentido, sendo possível substituir tais *scripts* por meio da comunicação com uma *API* de inteligência artificial, como o *ChatGPT*, por exemplo. Com isso, visualiza-se uma possibilidade de oferecer aos jogadores uma experiência ainda mais imersiva e personalizada, em que cada decisão tomada estará mais flexível com a escrita e o modo de escrever do jogador. O objetivo deste trabalho é implementar uma interface que faz uso do *ChatGPT* como um interpretador de comandos para o jogo tornando a jogabilidade mais agradável e menos restrita. Para isso, serão utilizadas duas ferramentas principais: o interpretador de jogos *Z-Machine* Frotz, o qual permite que seja possível rodar os jogos, e também a *API* do *ChatGPT*, sendo uma ferramenta de fácil acesso para desenvolvimento. Ambas as ferramentas são *open-source*. A solução desenvolvida foi avaliada por meio de testes locais em uma perspectiva qualitativa para identificar problemas e para verificar se melhorias implementadas melhoraram o uso dos comandos na jogabilidade e não descaracterizaram o gênero de ficção interativa, trazendo à tona o mecanismo de interpretação de termos realizando comparações entre comandos inseridos no jogo original com os mesmos comandos sendo inseridos na integração, sendo interpretados e retornados como comandos válidos.

Palavras-chave: Jogos. Inteligência Artificial. ChatGPT. Experiência de Usuário. Ficção Interativa.

ABSTRACT

In the 70s and 80s, interactive fiction games emerged, revolutionizing the world of *games* by allowing players to converse with the machine and make their own decisions, choosing their paths and actions. This freedom captivated players, but at the time, these games were limited by programmed *scripts*, which only accepted words present in the game's code dictionary. However, the evolution of technology has opened the door for improvements in this regard, making it possible to replace such *scripts* through communication with an artificial intelligence *API*, such as *ChatGPT*, for example. This presents the possibility of offering players an even more immersive and personalized experience, where each decision taken is more flexible with the player's writing and writing style. The aim of this work is to implement an interface that uses *ChatGPT* as a command interpreter for the game, making gameplay more enjoyable and less restrictive. For this, two main tools will be used: the *Z-Machine* Frotz game interpreter, which allows the games to be run, and the *ChatGPT API*, an easily accessible development tool. Both tools are *open-source*. The developed solution was evaluated through local tests from a qualitative perspective to identify problems and verify if implemented improvements enhanced the use of commands in gameplay without mischaracterizing the interactive fiction genre. This involved comparing the interpretation mechanism of terms by inserting commands in the original game with the same commands being inserted in the integration, interpreted, and returned as valid commands.

Key-words: Games. AI. ChatGPT. Flexibility.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

Nos últimos anos, o surgimento do *ChatGPT* (do inglês *Chat Generative Pre-training Transformer*) como uma ferramenta inteligente tem aberto grandes possibilidades em diversas áreas de conhecimento. Uma delas é a dos jogos digitais, onde essa tecnologia tem se mostrado uma grande aliada na busca por uma experiência cada vez mais imersiva e personalizada para os jogadores (BISWAS, 2023). Isto deve-se ao fato de que jogos digitais utilizam mecanismos nos quais os *chatbots* cognitivos e generativos mostram-se bastante apropriados, como em diálogos, sistema de jogos textuais, geração da própria inteligência de Personagens Não Jogáveis (*NPC*, do inglês *Non-Playable Character*), entre outras características (ESO, 2023). Nesse contexto, identificamos um gênero conhecido como ficção interativa, que, por conta de seus *scripts* e dicionários pré-definidos, muitas vezes limitam a interação do jogador com o ambiente virtual, pois ele acaba dependendo do conhecimento de termos exatos sobre como deve-se jogar o jogo em questão. Para superar essa limitação, desenvolvemos uma abordagem que permite ao jogador digitar livremente, sem estar restrito a regras pré-estabelecidas, e que o auxilia em situações mais complexas por meio do uso de uma ferramenta: o *ChatGPT*. Esta solução se dá pelo fato de o *ChatGPT* poder ser utilizado como um *parser* e também como montador de expressões de acordo com o que lhe for solicitado (ORTIZ, 2023), sendo capaz de realizar comparações e análises de uma frase complexa fornecida pelo jogador e filtrar apenas a informação exigida pelo jogo como instrução.

O *ChatGPT* é rico em funcionalidades e diferentes implementações, como indica a documentação disponibilizada pela OpenAI (2020). Conforme Limna et al. (2023), a ferramenta desenvolvida pela *OpenAI* traz um histórico de estudo de pelo menos 50 anos quando o primeiro programa de inteligência artificial ELIZA foi criado (LIMNA et al., 2023). Com o passar do tempo, o desenvolvimento e o estudo relacionados à inteligência artificial evoluíram de tal forma que atualmente temos algo como o *ChatGPT*, o qual usa processamento de linguagem natural e aprendizado de máquina em seu assistente virtual de interação com usuários. Um trecho de Limna et al. (2023) define o que é o *ChatGPT*:

> O desenvolvimento do ChatGPT envolveu um processo de ajuste fino que combinou técnicas de aprendizado supervisionado e de reforço. O ChatGPT foi projetado para ser altamente inteligente, intuitivo e capaz de responder a solicitações complexas de maneira semelhante a um ser humano. Com suas capacidades avançadas, o ChatGPT está mudando a forma como interagimos com a tecnologia e abrindo caminho para uma nova era de IA inteligente e conversacional.

Já os jogos de ficção interativa são meios de apreciar uma obra, na qual a história é guiada pelo próprio jogador que realiza as ações do personagem: para onde ir, o que fazer, entre outros. Por mais que não seja um gênero de jogos digitais em proeminência atualmente, utilizar este gênero como ponto de partida deste trabalho permite explorar o uso do *ChatGPT* dentro do mercado de jogos digitais e também resgatar esse tipo de

jogo para a comunidade, abrindo a possibilidade de se interessarem pelas obras de ficção interativa, buscando por mais conteúdo, e consequentemente aumentando o apreço pela leitura.

Nesse sentido, este trabalho de conclusão de curso tem como objetivo explorar as possibilidades do uso do *ChatGPT* como interpretador intermediário das entradas do usuário em jogos de ficção interativa, os quais são executados pelo interpretador de jogos *Z-Machine*, Frotz. Visamos tratar a entrada dos dados do jogo *Zork I* de ficção interativa. Utilizando da Interface de Programação de Aplicações (*API*, do inglês *Application Programming Interface*) do *ChatGPT*, junto de uma comunicação cliente-servidor desta *API* com o código que recebe as entradas de texto, vislumbra-se a possibilidade de otimizar e adaptar o jogo de forma mais flexível, ou seja, expandindo os dados de entrada para além do dicionário presente no código do jogo e fazendo com que os comandos do jogador sejam mais aceitos do que o normal.

2 TRABALHOS RELACIONADOS

Foram realizadas algumas pesquisas com o passar do tempo a respeito de trabalhos que utilizem do *ChatGPT* como um interpretador na indústria dos jogos, porém foram encontrados poucos casos de implementação até o presente momento. Em 2024, foram publicados dois trabalhos com o uso do *ChatGPT* como um interpretador: um deles a respeito da influência da inteligência artificial no diálogo de *NPCs* (Non-Plater Character) e na experiência de jogabilidade (IAROVOI; HEBBLEWHITE; TEH, 2024). Este trabalho representa um conceito diferente do trabalho, pois adequa-se às escolhas de diálogo e ações do usuário, sendo o NPC um elemento que utilizando o GPT entende estas decisões do jogador e realiza ações adequadas a situação. Enquanto isso, o outro trabalho encontrado também aborda a questão de interação com NPCs, porém com relação a pesquisa do potencial das interações com os modelos de linguagem (ANAND; POLYAK, 2024), sendo esta pesquisa baseada em como é possível criar interações NPC-Jogador utilizando de modelos de linguagem interpretativos.

Existem diversos trabalhos que utilizam os recursos do *ChatGPT* em outros contextos, como na construção de jogos de tabuleiro (JUNIOR et al., 2023), no desenvolvimento de roteiros de *light novels* (GROW; KHOSMOOD, 2023), entre várias outras aplicações que fogem do tema principal deste trabalho. O foco deste estudo define-se a respeito da ação do jogador ser interpretada, e não programada.

Por ser ainda um estudo emergente, provavelmente com o passar do tempo surgirão novos trabalhos abordando o tema e utilizando do *ChatGPT* de diversas maneiras dentro da questão de interpretação de termos e contextos, assim como na questão da interpretações de diálogos ou texto, mas atualmente apresenta-se como algo escasso e com poucos trabalhos ao redor do tema.

Entretanto, existem alguns trabalhos que abordam outras questões do uso do *ChatGPT* dentro do tema de Ficção Interativa, geralmente envolvendo a roteirização de histórias. Nesses trabalhos, um deles aborda o ato de ensinar o *ChatGPT* a desenvolver histórias complexas (CHEN et al., 2023) visando a capacidade de geração de histórias de modelos de linguagem para obtenção de histórias com prompts escritos por humanos. Este trabalho utiliza o *ChatGPT* para criar e conduzir os próprios jogos de ficção interativa, onde o mesmo, por sua capacidade interpretativa, consegue criar e entender as ações do próprio jogo com seus próprios cenários e objetos, enviando ao GPT amostras de treinamento para facilitar o entendimento da ferramenta e também para conseguir entender os comandos de ação posteriormente. O intuito do trabalho é fazer com que o *ChatGPT* entenda a ação final do jogador, como por exemplo, abrir um baú para que após isto seja possível revelar um outro objeto, tudo escrito em um mesmo comando de ação. A ideia é fazer com que o GPT entenda o que o usuário deseja e interpreta esta ação, sem que seja necessário uma série de comandos em sequência para que sejam executadas, além de também avaliar qual a melhor ação disponível para um determinado objeto baseado no

que lhe foi dado como instrução. A base deste trabalho consiste em simplificar e mapear as sequências de ação, onde, para avaliar se o agente pode concluir a experiência, os autores elaboraram um jogo de aventura em texto para simular uma casa para o jogador interagir.

Além disso, há uma pesquisa questionando se os LLMs (do inglês, *Large Language Models*) conseguem jogar jogos textuais de maneira aceitável (TSAI et al., 2023). Nesta pesquisa, investigou-se a capacidade do *ChatGPT* de jogar jogos de texto, onde é necessário entender o ambiente e responder a situações dialogando com o mundo do jogo. Um estudo de caso foi realizado com o jogo *Zork I*, o mesmo utilizado em nosso trabalho. O cenário atual do jogo e uma série de ações possíveis foram enviados manualmente ao *ChatGPT*, permitindo que ele jogasse o jogo. No entanto, foram constatadas diversas inconsistências nas respostas, seja por não entender as limitações dos comandos, por alucinações aleatórias (invenção de locais ou objetos), ou por incoerências ao ser questionado sobre eventos passados já enviados anteriormente. Em resposta a isso, os autores incluíram o passo-a-passo do jogo como parte do treinamento do *ChatGPT*, proporcionando-lhe uma boa noção do mapa e das ações do jogo, embora suas jogadas ainda fossem um pouco imprecisas. Conforme será observado neste trabalho, quanto mais informações são inseridas no GPT, mais frequentes são as confusões e as respostas imprecisas. Embora o *ChatGPT* tenha um conhecimento completo sobre os locais do mapa e as ações necessárias para chegar a determinados lugares, ele não demonstrou uma boa inteligência ao efetivamente jogar o jogo.

Por fim, outro trabalho interessante de ser observado é sobre uma investigação sobre como um jogador se comporta baseado em certos cenários climáticos utilizando de um jogo textual pelo *ChatGPT* (ZHOU et al., 2024). O trabalho investiga a eficácia de avaliações baseadas em jogos para sondar atitudes em relação à mudança climática. Utilizando o *ChatGPT*, é simulado um cenário climático futurista em forma de jogo e o estudo analisa como a interação dos jogadores com o jogo pode influenciar sua conscientização climática, identificando a preferência por valores democráticos do jogador. Para isto, foi criado um sistema de *chat*, simulando uma conversa entre a máquina e o usuário onde são abordados assuntos climáticos e visualizada a abordagem do jogador baseada em suas respostas. Como resultado positivo dessa pesquisa, os participantes no geral saíram com uma maior conscientização e com melhores atitudes referentes a cenários climáticos quando comparados ao seu contato inicial ao jogo.

3 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, são detalhados conceitos importantes para compreensão da proposta, como o que são jogos de ficção interativa, sua evolução, gêneros de jogos presentes no trabalho, tecnologias, recursos de inteligência artificial, entre outros. É importante ressaltar que os gêneros que originaram-se dos jogos de ficção interativa não serão tratados da perspectiva de uso em conjunto com o *ChatGPT*, apenas como ilustração da importância do gênero como um fator de evolução no cenário de jogos, além de trazer o conhecimento necessário para a definição de ficção interativa.

3.1 Jogos de Ficção Interativa

Os jogos de ficção interativa surgiram em meados de 1975 com o jogo *Colossal Cave Adventure* (Figura 1), criado por Will Crowther, como um novo gênero de jogos para computador (ZURAWEL, 2017). Esse gênero que ficou conhecido como *Adventure* e baseia-se na ideia do jogo *Colossal Cave Adventure* como exemplo, no qual um computador apresenta ao jogador um texto com a descrição do contexto (lugar, objetos, personagens ao redor) onde o jogador está, seguido de um *prompt*[1]. O jogador responde com uma frase de uma ou mais palavras indicando a ação que deseja realizar, como por exemplo pegar um objeto, ir a algum lugar, entre outros. O interpretador do jogo realiza então o reconhecimento do comando e atualiza o *status* do jogo, mostrado na tela do jogador um novo trecho de texto que representa a consequência daquela ação.

Figura 1 – Colossal Cave em um terminal

Fonte: OSTechNix (2017).

[1] Campo de inserção de comandos para o jogo

3.1.1 Adventures Gráficos

Como um ciclo natural do mundo dos jogos, gêneros foram evoluindo e novas maneiras de se jogar foram desenvolvidas, e claramente os jogos de ficção interativa não seriam diferentes disso. Com isso, a evolução dos *Adventures* veio com a implementação de visualização gráfica dos cenários e objetos na tela junto dos *prompts* de comando, sendo conhecidos por *Adventures* Gráficos tendo *Wizard and the Princess* (Figura 2), criado por Ken e Roberta Williams, como um dos pioneiros deste novo mecanismo, podendo ser visto em MOSS (2011).

Figura 2 – Wizard and the Princess

Fonte: MOSS (2011).

Esta pode ser considerada uma inovação relevante, já que o fato de se poder visualizar algo que anteriormente apresentava-se apenas como uma sequência de textos acaba por aprimorar a imersão dentro destes jogos. Neste sentido, em 1983, a empresa *Sierra On-Line* criou *King's Quest: Quest for the Crown* (Figura 3), que foi responsável por definir e consolidar o gênero de ficção interativa, por ser o primeiro jogo de computador a suportar 16 cores no padrão Adaptador Gráfico Aprimorado (EGA, do inglês *Enhanced Graphics Adapter*) e também o primeiro a oferecer um mundo pseudo-3D com os jogadores controlando um personagem (MOSS, 2011).

Figura 3 – King's Quest: Quest for the Crown

Fonte: MOSS (2011).

Porém é neste momento que se pode observar um dos maiores problemas com jogos de ficção interativa: a limitação da interpretação dos comandos do jogador. Os jogos contavam com *parsers* de texto[2] que identificavam os comandos inseridos pelo jogador. Esses comandos fazem parte do código do jogo, devendo-se inserir um comando que siga estritamente a lógica por traz da ação, como por exemplo, um caso onde seja necessário pegar uma chave em certos jogos o comando seria "PEGAR CHAVE", não aceitando sinônimos como coletar, agarrar, etc. Dessa forma, o jogo iria rejeitar a ação e o jogador acabava confuso com um fator que deveria ser um mecanismo simples e acessível de jogabilidade. O quebra-cabeça dos jogos acabava se expandindo para além do contexto do jogo, onde o próprio usuário geralmente tinha dificuldades em entender como deve escrever o comando que deseja.

Este problema se destacou com a implementação de visualização gráfica dos jogos devido ao jogador não conseguir identificar com precisão os elementos gráficos de jogos com baixa resolução. O jogador então poderia ficar pensando: "Aquilo é uma pedra? Um tronco? Seria isso o lagarto dito no texto? Afinal é uma rocha ou uma pedra? Qual a forma que deveria ser escrito?" Com isso, não era mais apenas a questão de se desvendar os comandos de ação ao jogar, mas também de identificação dos objetos presentes no aspecto visual, já que os elementos gráficos da época não eram muito detalhados. Além disso, devido a ser apenas uma visualização do que está sendo descrito no texto do jogo, não havia elementos de jogabilidade envolvendo a imagem. Foi então que surgiu uma nova vertente do gênero: o *Point and Click*.

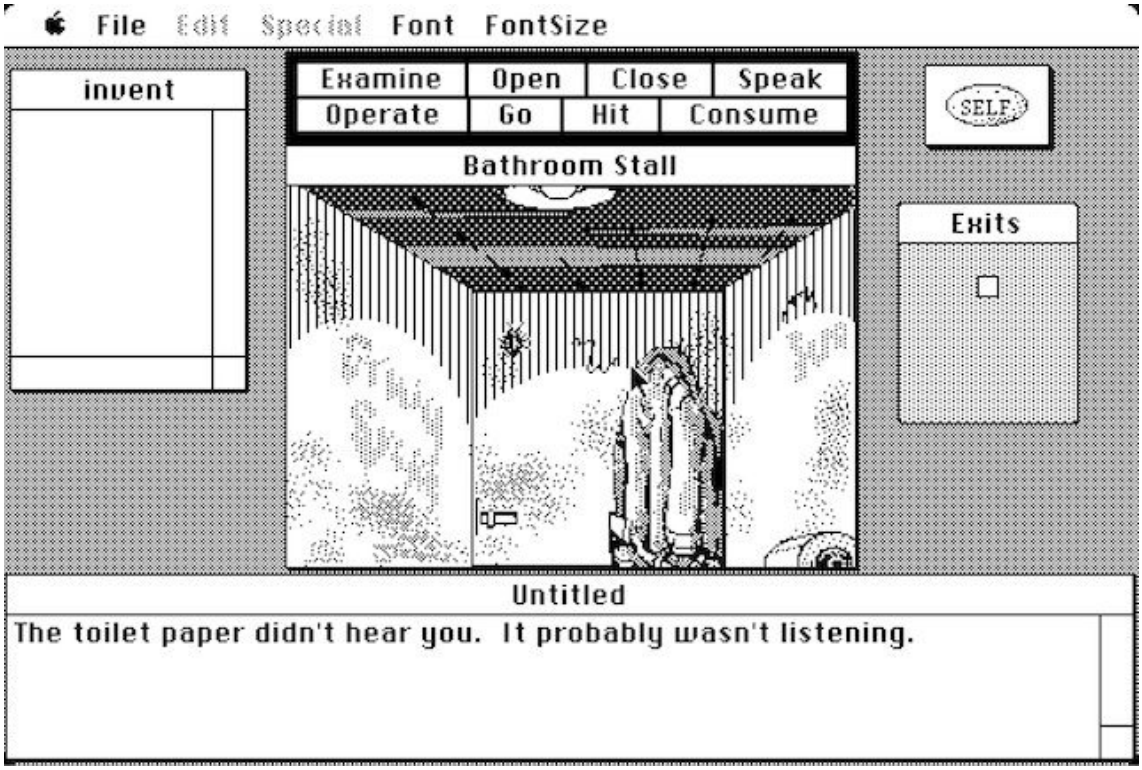
3.1.2 Point and Click

No ano de 1984 foi lançado o *MacIntosh*, produzido pela *Apple*, que popularizava o uso do *mouse* introduzido inicialmente pela Xerox Corporation, onde algumas desenvolvedoras de jogos viram uma oportunidade de criação de jogos que aproveitassem esse dispositivo (MOSS, 2011). Foi então que a *Silicon Beach Software* lançou o jogo *Enchanted Scepters*, o primeiro jogo *Point and Click* com menus para seleção de ações e uma janela de texto separada da tela de visualização gráfica do jogo. Porém, o jogo que realmente inovou o gênero de ficção interativa e deu renome ao *Point and Click* foi *Déjà Vu*, desenvolvido pela *ICOM Simulations* no ano seguinte.

Déjà Vu não contava somente com uma tela gráfica e menus de seleção, mas sim com um sistema completo de jogabilidade com o *mouse*. Esse sistema e jogabilidade incluída, por exemplo, cenário visual, possíveis saídas, inventário com a lista de objetos obtidos, possibilidade de arrastar objetos do cenário para o inventário, e botões para os comandos de ações, funcionando da seguinte forma: ao clicar em uma ação e posteriormente em uma parte do cenário, será realizada tal ação neste ponto clicado. Estes novos recursos, visuais e modo de jogar podem ser melhor visualizados na Figura 4.

[2] Mecanismo de identificação de palavras dentro de uma frase

Figura 4 – Déjà Vu



Fonte: MOSS (2011).

Porém, como abordado em MOSS (2011), por mais que tenha sido algo inovador, levou um tempo até que este padrão de jogabilidade se tornasse popular, já que o público ainda preferia o *Adventure* clássico com a principal forma de interação sendo via textos.

Atualmente, com a constante evolução na área de desenvolvimento de jogos e experiências interativas, o modo de jogar evoluiu consideravelmente com elementos visuais, ações manuais (utilização do *mouse* para desencadear eventos) e desenvolvimento de novas histórias, mas ainda assim mantendo a jogabilidade mais clássica com relançamentos de franquias ou novos lançamentos nesta linha de jogos interativos. Isto pode ser visto em exemplos como jogos interativos da empresa *Telltale*, os quais são uma mistura de ficção interativa de diálogos, uma *novel*[3] com o fator de jogabilidade utilizando o personagem e de renascimentos de clássicos como *Return to Monkey Island*, lançado no ano de 2022 - o título anterior original da franquia foi lançado no ano de 2009 segundo a Wikipedia (2023) com um hiato de 13 anos. Foram lançados dois jogos nesse intervalo de tempo, porém eram apenas republicações dos jogos antigos para sistemas atuais.

Assim como qualquer gênero ou estilo de jogos, há um nicho e comunidade específica. O fato de jogos de ficção interativa estarem sendo lançados até os dias atuais torna a comunidade viva e ativa dentro de seu público (CHINCHILLA, 2023). Contudo,

[3] Jogos baseados no diálogo com respostas de múltipla escolha para determinação de ações e eventos que definem o rumo da história

por mais que inovações na área e adaptações para a atualidade tragam novos jogadores para o gênero, como a evolução para *Point and Click* e *novels*, a parte mais saudosista e antiga da comunidade, a qual tem por preferência algo mais literário com textos sem visualização para exercício de criatividade e imaginação, acaba se afastando consideravelmente pela preferência desta forma antiga de jogar, terminando por não receber atenção das desenvolvedoras de jogos.

3.2 Inteligência Artificial

O conceito amplo de Inteligência Artificial (IA) varia conforme os autores e suas correntes filosóficas. Como exemplo, Cossetti (2018) define a Inteligência Artificial como "um avanço tecnológico que permite que sistemas simulem uma inteligência similar à humana" ou também como "a capacidade das máquinas de pensarem como seres humanos: aprender, perceber e decidir quais caminhos seguir, de forma racional, diante de determinadas situações". Ainda de acordo com Cossetti (2018), há décadas iniciou-se um estudo a respeito de agentes inteligentes, os quais percebem e realizam por conta própria a melhor forma de tratamento da situação. Tais agentes encontram-se totalmente presentes nos dias de hoje, seja em painéis de carros, eletrodomésticos, sistemas de atendimento, redes sociais, e até mesmo em ferramentas que foram criadas com esta tecnologia justamente como intuito de serem inteligentes e gerarem informações por conta própria, como é o caso do *ChatGPT* (OPENAI, 2022).

Além disso, de acordo com Russell (2020), a Inteligência Artificial é definida como "o estudo de como fazer computadores realizarem tarefas que, no momento, os humanos fazem melhor". Ainda de acordo com Russell (2020), a IA envolve a criação de agentes inteligentes que podem perceber seu ambiente, raciocinar sobre o que percebem e tomar ações para atingir seus objetivos, além das diferentes abordagens para criação de agentes inteligentes, incluindo sistemas baseados em regras, sistemas baseados em conhecimento, sistemas baseados em aprendizado e sistemas baseados em evolução.

As aplicações da inteligência artificial são tantas que Longo et al. (2020) afirmam que "O campo [da Inteligência Artificial] se tornou tão diversificado em termos de métodos, frequentemente determinados por questões e atributos específicos do domínio, que é praticamente impossível obter um conhecimento aprofundado sobre tudo."

3.2.1 Processamento de Linguagem Natural

Neste contexto uma das vertentes da inteligência artificial é o Processamento de Linguagem Natural (PLN), a qual define-se como uma ciência que auxilia na interpretação e manipulação de linguagem humana por parte dos computadores. De acordo com SAS (2023), "o processamento de linguagem natural incorpora técnicas diversas para interpretar a linguagem humana, desde métodos estatísticos e de *machine learning* a abordagens

algorítmicas e baseadas em regras".

De acordo com Gonçalves (2021), o PLN tem como objetivo a recuperação de informações a partir de textos, a tradução automática, a interpretação de textos e a realização de inferências a partir de textos. O PLN pode ser dividido em três tópicos macro: Entendimento da linguagem humana, Geração de linguagem humana e Processamento de linguagem humana. Do ponto de vista linguístico, o foco das pesquisas em PLN podem variar em diferentes níveis de análise, podendo ser: fonético ou fonológico, morfológico, sintático, semântico ou pragmático. Por exemplo, a análise sintática, a estrutura gramatical é utilizada para entender o relacionamento entre as palavras e a organização das frases. Já na análise semântica, a ideia é tentar compreender o significado das palavras e frases, além do contexto em que elas são utilizadas (GONçALVES, 2021).

Com isso, ainda conforme Gonçalves (2021), o PLN permite que as máquinas entendam a linguagem humana e, portanto, possam realizar tarefas como a tradução automática, a análise de sentimentos, a identificação de tópicos e a extração de informações.

Como exemplo, o próprio *ChatGPT* é definido como uma ferramenta que utiliza do processamento de linguagem natural, como pode ser observado em Ortiz (2023). O *ChatGPT* utiliza uma arquitetura de modelo de linguagem chamada *Generative Pre-trained Transformer* (GPT), que significa, de maneira resumida, um transformador generativo pré-treinado com uma vasta quantidade de informações advinda da Internet, como livros, artigos e inúmeras outras fontes de informação.

Sendo assim, ainda de acordo com Ortiz (2023), o processamento da linguagem natural do *ChatGPT* faz com que os questionamentos ou informações desejadas digitadas pelo usuário consigam ser interpretadas pelo computador e retorne os dados corretos obtidos do banco de dados da ferramenta.

3.2.1.1 Parsers de texto

Um dos pontos chave referentes ao PLN, além de permitir os computadores a entenderem e processarem linguagem humana, de acordo com Singh (2019), é a questão dos dados de texto não estruturado. Em tarefas tradicionais de aprendizado de máquina são tratados dados tabulares estruturados, ou seja, os dados são definidos dentro de uma categoria/definição baseado na linha ou coluna em que se encontram. Então, para que possam se aplicar técnicas de aprendizado de máquina ao PLN, é necessário saber como tratar a questão do texto. Sendo assim, os *parsers* de texto surgem como uma solução para este fator.

Ainda conforme Singh (2019), *parsers* de texto são tarefas comuns de programação que separam uma série de texto dada em componentes menores com base em algumas regras. Uma de suas formas mais conhecidas são as expressões regulares (Regex) as quais definem-se como expressões que identificam padrões em outros textos. Um exemplo disso pode ser dado da seguinte forma: na frase "O homem, cabeludo, foi encontrado", as

expressões regulares podem, por exemplo, dividir o texto conforme encontram-se vírgulas, ficando “O homem,”, “ cabeludo,”, “ foi encontrado” ou até mesmo identificar palavras em específico. Tal funcionalidade é de suma importância para o presente trabalho, pois com ela poderemos coletar e fazer uso das informações necessárias dentro do texto de entrada inserido pelo usuário nos jogos de ficção interativa de forma prática. Além disso, como um exemplo de aplicação na indústria, este fator de identificação é muito utilizado em aplicações que recebem arquivos XML para organização e atribuição dos dados do arquivo.

Por fim, conforme Singh (2019), Regex também são capazes de realizarem o processo de tokenização, o qual define-se como a conversão de um texto em *tokens* (um *token* pode ser definido como a divisão de algo em uma pequena parte) com base em regras pré-determinadas. Este processo de tokenização auxilia nas tarefas de pré-processamento de texto, como mapeamento ou localização de palavras (SINGH, 2019).

Esta questão de determinar regras para localização em texto será de grande valia para o trabalho, visto que estaremos visando obter um comando específico do jogo de ficção interativa de dentro de uma frase estruturada pelo próprio jogador.

3.2.2 Chatbots

Um *chatbot* é a união entre aprendizado de máquina com interação humano-computador onde uma inteligência artificial faz uso do processamento de linguagem natural para que o computador entenda a linguagem humana e tenha comportamento próprio (IBM, 2023b). Inicialmente eram utilizados como *chats* restritos a uma programação, ou seja, um *chat* em que não necessariamente há uma conversa, mas sim um menu interativo com escolhas de atendimento no suporte ao cliente de sites, por exemplo. Contudo, nos últimos anos este tipo de mecanismo evoluiu e hoje temos ferramentas como o *ChatGPT* que são *chatbots* generativos. De acordo com Codecademy (2023), “Um *chatbot* generativo é um programa de *chat* em domínio aberto que gera combinações originais de linguagem em vez de selecionar respostas pré-definidas”, ou seja, nada mais é do que um gerador de combinações da linguagem ao invés de respostas pré-prontas dentro de um *script*.

Um questionamento feito por Shridhar (2017) chama a atenção: Por mais que os *chatbots* generativos sejam os mais inteligentes, eles seriam a solução para todos os problemas?

Na fala de Kumar, a conclusão de pensamento é uma reflexão crítica sobre as limitações dos chatbots generativos. Embora reconheça a inteligência desses *chatbots*, é questionado se são realmente capazes de resolver todos os problemas. Isso sugere que, apesar de suas capacidades avançadas, existem limitações e talvez seja necessário cautela e discernimento ao considerar seu uso como solução universal.

Nesta mesma publicação, Kumar Shridhar define uma implementação de um *chatbot* generativo por meio de Rede Neural Recorrente (RNN, do inglês *Recurrent Neural*

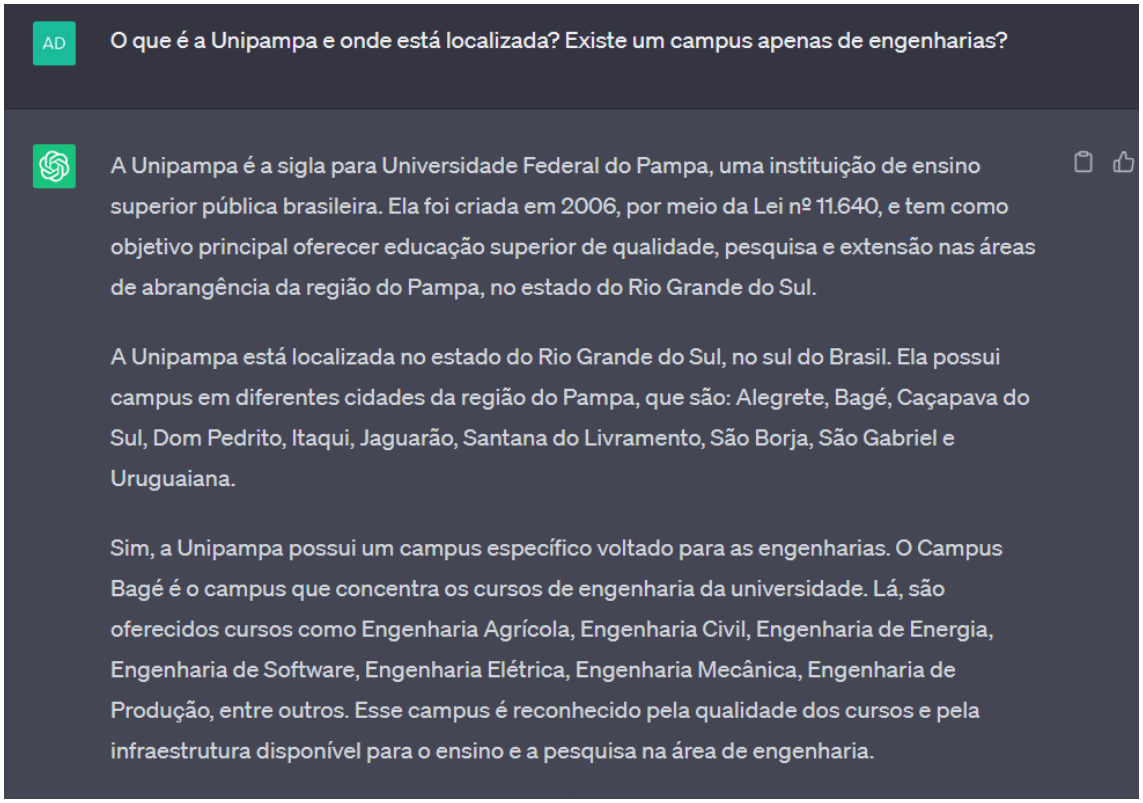
Network), que define-se como uma topologia de rede onde a rede reinjeta resultados parciais nela mesma de modo a resolver o problema a que se propõe. Por conta disso, a RNN torna mais eficaz o Processamento de Linguagem Natural, uma vez que o contexto da próxima palavra não depende apenas da palavra anterior, mas de uma série de palavras. Com isso, os *chatbots* generativos tornam-se interpretativos e não restritos ao *script* ou a um dado específico inserido anteriormente.

3.2.3 ChatGPT

Inserido ao contexto de *chatbots* generativos, foi desenvolvida uma nova ferramenta para o mercado: o *ChatGPT*. Sua implementação se diferencia da RNN por mais que tenham algumas similaridades podendo ser observado em Agrover112 (2023), o qual define a mecânica da seguinte forma: em cada passo de tempo, a entrada de dados contém algum contexto em relação à saída, a qual cada entrada geralmente refere-se a algum contexto da saída. Portanto, mesmo sem treinamento, as entradas atuam como "verdades fundamentais" do que realmente se é desejado. Em outras palavras, a abordagem do *ChatGPT* é como um assistente inteligente que, a cada momento, recebe informações sobre o que está acontecendo e usa essas informações para entender o que deveria fazer em seguida. Mesmo sem treinamento específico, ele usa essas "verdades fundamentais" das entradas para gerar saídas que fazem sentido, lidando com o contexto da informação, enquanto uma *RNN* depende de um tipo de memória interna para ter essas informações passadas, podendo ter problemas com informações muito antigas. Ainda em sua conclusão é citado o seguinte trecho: "Isso é bastante teórico, sem nenhuma evidência empírica ou até mesmo provas concretas de que os *ChatGPTs* são *RNNs*. A principal ideia foi destacar as similaridades entre os mesmos.", então por mais que não se tenha uma definição exata de que o *ChatGPT* seja uma *RNN*, entender essa similaridade ajuda a compreender seu funcionamento. Além disso, no site oficial e documentação (OPENAI, 2022) podem ser encontradas informações adicionais e mais detalhadas.

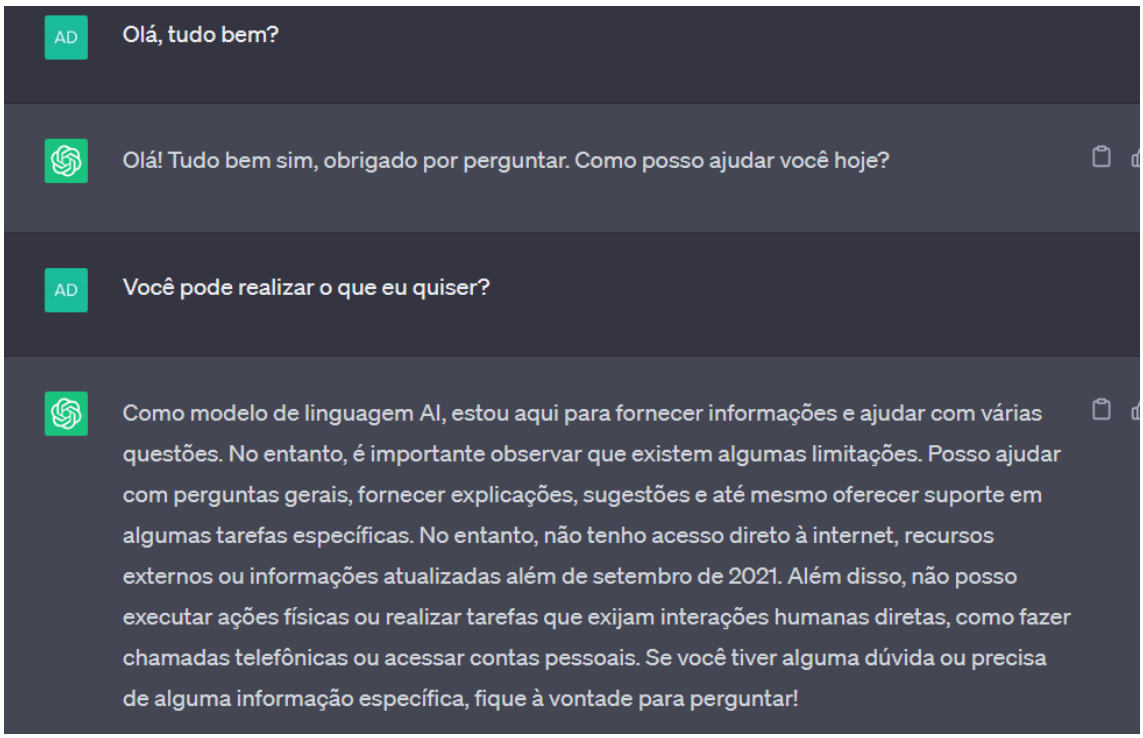
O *ChatGPT* foi treinado utilizando do método *Reinforcement Learning from Human Feedback* (RLHF) com treinamento supervisionado mesclando os dados obtidos do treinamento com alguns testes da inteligência artificial junto de alguns usuários, transformando isto em um formato de diálogo como pode ser observado ainda em OpenAI (2022). Em linhas gerais, o *ChatGPT* é um campo de conversa entre o usuário e a IA com um rico banco de dados de informações. Para fins de visualização, duas imagens, Figura 5 e Figura 6, apresentam exemplos de utilização dessa ferramenta e seu funcionamento pode ser visto com melhores detalhes na Figura 7. Como pode ser observado, na Figura 5 encontra-se uma informação incorreta dada a respeito da Unipampa e o campus que concentra os cursos de engenharia mostrando que por mais que a ferramenta interprete o que foi perguntado, em muitas situações ocorre de a resposta não ser a esperada, já que o campus das engenharias encontra-se em Alegrete e não em Bagé.

Figura 5 – Teste com ChatGPT - 1



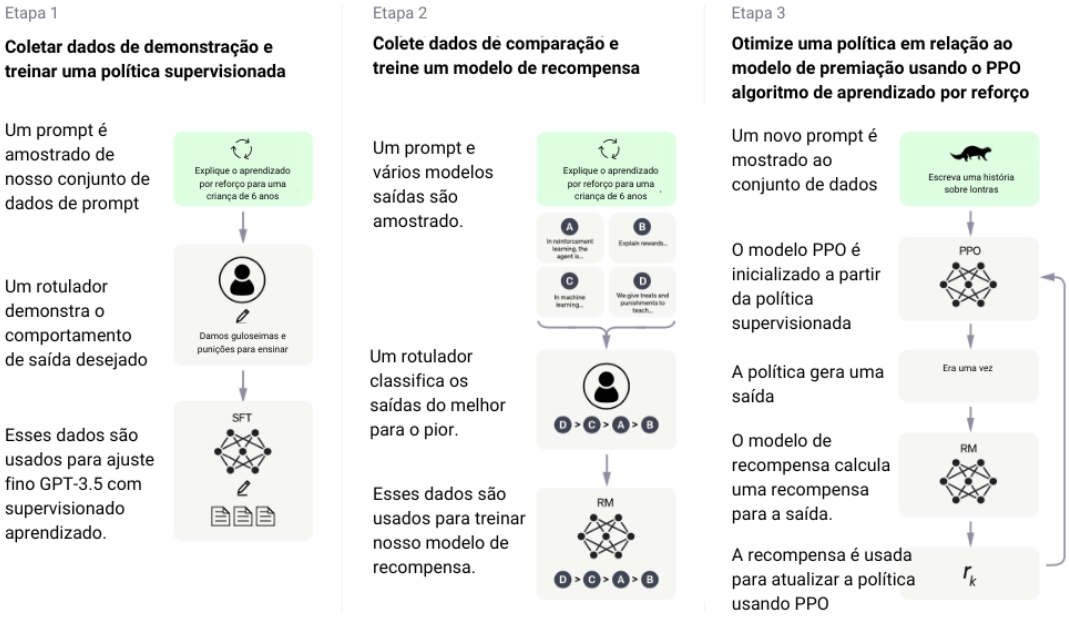
Fonte: O autor.

Figura 6 – Teste com ChatGPT - 2



Fonte: O autor.

Figura 7 – Etapas de Funcionamento do ChatGPT



Fonte: NEWRIZON, 2023.

3.2.3.1 ChatGPT API

RedHat (2022) define uma API como uma série de definições e protocolos para construção e integração de softwares. De forma resumida, a função de uma *API* é basicamente permitir que um produto ou serviço se comunique com outros produtos ou serviços sem saber de que forma estes foram implementados, ou seja, permite a comunicação entre as duas partes de forma simplificada.

Sabendo destes conceitos, de que a *API* é uma interface facilitadora de comunicação entre serviços e produtos, a utilização da *ChatGPT API* não é diferente do citado. Podendo ser implementada tanto em Python quanto JavaScript, esta *API* contém todas as funcionalidades do ChatGPT além de melhorias de precisão nas respostas de acordo com a documentação OpenAI (2020). Para este trabalho, a *API* tem um papel fundamental que será justamente do uso do chat e da precisão de respostas para que seja feita a análise correta dos comandos de acordo com a entrada de dados do usuário na aplicação. Isto será mais detalhado posteriormente.

Além disso, alguns usos dessa *API* já estão sendo empregados em outros jogos, como *The Elder Scrolls V: Skyrim* em ESO (2023), e também como chatbots de *WhatsApp* em Goulart (2023).

3.3 Máquinas Virtuais (VM)

De acordo com IBM (2023a), uma Máquina Virtual (VM, do inglês *Virtual Machine*), em resumo, é a representação virtual ou emulação de um computador físico. Esta representação consegue ser realizada devido a virtualização[4], onde cada *VM* apresenta seu próprio Sistema Operacional (SO) e seus próprios aplicativos em uma única máquina física. Para isso acontecer, é necessário um mecanismo conhecido como *hypervisor*, o qual é responsável por alocar recursos como processadores, memória e armazenamento, para cada *VM*. Sendo assim, ao utilizar um *hypervisor*, o computador físico poderá separar o sistema operacional em várias "máquinas virtuais" independentes.

Existem dois tipos primários de *hypervisors*: *hypervisors* do tipo 1 e *hypervisors* do tipo 2 definidos por IBM (2023a).

Hypervisors do tipo 1 têm execução direta no hardware físico, ocupando o lugar do SO, sendo geralmente utilizado um software distinto para criação e manipulação de *VMs*. Este tipo também permite a utilização de uma *VM* como modelo, viabilizando a criação de diversos modelos de *VM* para diferentes propósitos, como testes de software, bancos de dados de produção e ambientes de desenvolvimento (IBM, 2023a).

Hypervisors do tipo 2 possibilitam a criação manual de uma *VM*, sendo possível configurar manualmente a quantia de núcleos do processador e a memória a ser utilizada, sendo possível utilizar de opções de aceleração 3D para gráficos dependendo dos recursos do *hypervisor* utilizados (IBM, 2023a).

Este conceito é utilizado no trabalho em dois pontos principais: para executar o Frotz, foi utilizada uma *VM* com sistema operacional Linux. Além disso, o próprio Frotz é considerado, em sua definição, como uma máquina virtual.

3.3.1 Frotz

Por terem seus códigos escritos para máquinas com arquiteturas muito diferentes das atuais, jogos muito antigos de ficção interativa como *Colossal Cave Adventure* e *Zork* tornam-se de difícil acesso para serem jogados atualmente, pois *Z-Machine* não é mais suportado em sistemas atuais. Vendo este problema, David Griffith começou a implementação de um interpretador de jogos de ficção interativa para Unix, o Frotz (GRIFFITH; JOKISCH, 2016b).

Griffith e Jokisch (2016b) indicam que o Frotz foi escrito inicialmente por Stefan Jokisch no período entre 1995 e 1997 e que Galen Hazelwood iniciou o processo de portabilidade para Unix, onde posteriormente David Griffith ficou como responsável pela finalização do procedimento e de manter as atualizações do interpretador até os dias de hoje. Ainda sim, inúmeras versões para diversas plataformas foram desenvolvidas com o

[4] Camada de abstração que permite os elementos de hardware de um único computador serem divididos em várias *VMs*

passar dos anos por autores diferentes (GRIFFITH; JOKISCH, 2016a).

O Frotz é definido como "um interpretador para jogos da *Infocom* e outros jogos *Z-Machine*" (GRIFFITH; JOKISCH, 2016b), compilando a *Z-Machine Standard* na versão 1.1. Além disso, consta com recursos como compilar e rodar se complicações em sistemas Unix, portabilidade para diferentes sistemas e hardwares (GRIFFITH; JOKISCH, 2016a), possibilidade de uso de interfaces gráficas e sons (GRIFFITH; JOKISCH, 2016b).

Além de seu uso básico de possibilitar o acesso a jogos antigos de ficção interativa, o Frotz possui códigos *open-source* de todas as suas versões, ou seja, é possível obter os códigos do interpretador de uma versão específica e alterá-lo conforme desejado, seja para testes, integrações ou melhorias (GRIFFITH; JOKISCH, 2016a). Com isso, o Frotz torna-se nosso motor de funcionamento para que sejam possíveis rodar estes jogos e testar as modificações e ajustes desse trabalho.

4 DESENVOLVIMENTO

Após ter conhecimento das limitações do jogos de ficção interativa, do funcionamento do interpretador Frotz e das capacidades da *API* do *ChatGPT*, pensamos na ideia de unir estes fatores para justamente contornar o problema de limitação de dicionário dos *Adventures* utilizando como base o jogo *Zork I*. Como o Frotz é *open-source*, alteramos seu código fonte para adaptar seu sistema para a *API* do *ChatGPT* enviando uma frase escrita pelo usuário, sendo inserida no *ChatGPT* com um *script* elaborado neste trabalho que acabe por retornar o comando correto presente no dicionário do jogo original. Com isso, foi possível realizar uma grande melhoria no comando de entrada inserido pelo jogador, tornando a jogabilidade dinâmica e com um fator interpretativo da sentença sem estar restrito aos comandos específicos do dicionário do jogo em si. Além disso, não foram realizadas muitas modificações no código original do jogo mantendo assim sua essência.

4.1 Proposta

Utilizamos o interpretador de jogos de ficção interativa Frotz alterando seu mecanismo de *input* para receber e enviar o comando via uma arquitetura cliente-servidor entre o código fonte em C e um código Python contendo a integração da *API* do *ChatGPT*, que realizará a conversão do texto no código relacionado.

4.2 Visão Técnica

Inicialmente, avaliamos a disponibilidade e o processo de implementação da *API*, juntamente com os custos associados. Entre as opções, nosso foco principal para este estudo é a funcionalidade exclusiva de "Chat" (conhecida como gpt-3.5-turbo) a ser usada em uma implementação cliente-servidor entre as linguagens de programação C e Python. Em relação à disponibilidade e ao custo, essa escolha se mostra viável com despesas mínimas, totalizando U$0.0015 por cada 1000 *tokens* de entrada e U$0.002 por cada 1000 *tokens* de saída. Para este caso, *tokens* são as palavras nas utilizações da *API*, ou seja, para cada 1000 palavras inseridas será cobrado um valor, assim como para o retorno de cada 1000 palavras será cobrado um outro valor.

Além disso, para interpretar esses jogos e integrar a *API ChatGPT* ao código do interpretador, descobrimos o interpretador Frotz, que é compatível com várias plataformas, incluindo Windows, Linux, DOS e até mesmo Nintendo DS. Foi realizada uma tentativa de implementação com a versão de Windows, porém por mais que tenhamos seguido o passo-a-passo à risca do guia disponibilizado pelo autor, ainda sim ocorreram diversos erros de compilação da versão *open-source*, onde descobrir e resolver os problemas relacionados a essa versão acabariam custando tempo de desenvolvimento do trabalho e foi optado seguir por outro caminho. Como resultado, optamos por usar a plataforma Linux e instalamos o mecanismo "Dumb Frotz", uma variante simplificada do interpreta-

dor Frotz que preserva a experiência do usuário original sem interfaces, cores e sons. Essa interface de linha de comando provou ser a opção mais adequada para conduzir nossos testes justamente por sua simplicidade e menor custo em desempenho, já que seria necessário encerrar a execução e reiniciá-la para toda vez que fossem realizadas alterações no código.

4.3 Implementação

Para facilitar essa integração, optamos por uma arquitetura cliente-servidor, onde o código do Frotz (escrito na linguagem de programação C) se comunica com uma base de código em Python contendo a integração do *ChatGPT*. A linguagem de programação Python foi escolhida pela sua simplicidade de implementação e melhor compatibilidade com chamadas de *API* em comparação aos esforços de integração e invocação de *APIs* na linguagem de programação C.

Ao iniciar o jogo selecionado, é mostrado o texto referente ao enredo do jogo e abaixo liberado o campo de inserção de entradas de comando para jogar. Neste momento, um *socket* de comunicação é estabelecido para transmitir as informações do campo de entrada de texto para o código em Python que contém a integração com o *ChatGPT*, onde o texto recebido é analisado conforme a relação "verbo + objeto" (o padrão do jogo selecionado para os testes). Por mais que este seja o comando habitual de jogabilidade, o jogo também aceita comandos com uma única palavra mas geralmente com comandos únicos (como abrir o inventário, salvar, olhar em volta) ou em casos de movimentação (como norte, sul, leste ou oeste), sendo assim, a integração do *ChatGPT* tentará contornar a parte de casos envolvendo a relação "verbo + objeto" por depender da interpretação de termos dentro do contexto das palavras. Será devolvido apenas o comando no padrão desta relação. É importante reconhecer que esse procedimento pode enfrentar desafios devido ao padrão de resposta do *ChatGPT*, que muitas vezes inclui informações contextuais ou explicações adicionais. Para mitigar esse problema, um formato de mensagem padronizado foi implementado no código para garantir que a resposta retornada siga consistentemente um padrão de comando. O padrão inicialmente fora estruturado da seguinte forma:

"Se a seguinte frase contiver um verbo de ação em inglês, e também uma palavra que remeta a um objeto ou lugar em algum momento, retorne para mim diretamente apenas a frase 'verbo + objeto', sem mais nenhuma palavra a mais que isso nem explicação nem nada, apenas o que foi pedido entre aspas porém sem as aspas. A frase é: I want to climb the rocky mountain"

É importante notar que existem jogos que utilizam padrões diferentes de "verbo + objeto", às vezes aceitando combinações de três ou mais palavras. Embora existam alguns casos de uso de três ou mais palavras no jogo escolhido para este trabalho, estes são apenas exceções à regra e não o padrão do jogo. Para este trabalho, os testes e experimentos foram realizados sempre com base no jogo *Zork I* seguindo o padrão mencionado.

Porém tal *script* acabou se tornando inconsistente, uma vez que quanto mais informações e condições fossem adicionadas ao tratamento, mais o *ChatGPT* se confundia e retornava comandos incorretos. A imprecisão e inconsistência foi um fator crucial para que o *script* devesse ser repensado, sendo necessária uma nova abordagem.

Com isso, foi elaborado um novo padrão: dividir o *script* em partes, tornando cada iteração sobre a chamada do *ChatGPT* mais específica, fazendo com que assim, acabe tornando a resposta mais precisa. Os pontos avaliados na criação e ajuste do *script* foram:

- Qual o contexto que o GPT deve interpretar?
- Qual o dado desejado?
- Como o GPT deve retornar esse dado?
- Quais condições devem ser atendidas?
- Existe algum padrão de resposta do GPT que atrapalhe o retorno do dado?
- Ao final, existe alguma exceção à regra que faça com que ocorram problemas e inconsistências no retorno do script?

Esta linha de raciocínio deve-se ao fato de que quanto melhor for definido o intuito da mensagem passada para o *ChatGPT*, mais preciso será o retorno. Em um primeiro momento pensa-se em qual é o contexto da mensagem que deve ser interpretador. Após definido, deve-se pensar qual dado dessa mensagem é relevante para o contexto, para assim ter uma base do que o GPT deve retornar como resposta. Porém apenas isto torna a interpretação muito vaga, então outras questões devem ser avaliadas como um todo. Quais condições devem ser atendidas? Elas existem no contexto? Perguntas como estas tornam-se essenciais para o entendimento por parte do GPT para que ele acabe não se confundindo ou retornando algo incorreto.

Algo muito comum na utilização do *ChatGPT* é o seu padrão de resposta. Como por exemplo, se for perguntado a ele o resultado de “2 + 2” ele retornará uma resposta como “O resultado de 2 + 2 é 4”, quando eu apenas quero que seja respondido para mim “4”, sendo assim, necessário elaborar uma solução para esta questão nos *scripts*.

Ao final de tudo, após toda a análise e avaliação do *script* criado seguindo estes critérios, pode acabar ocorrendo questões que fujam da regra original ou conflitem entre si, sendo necessário assim reavaliar ou pensar em uma forma de contornar o problema.

Ao seguir esta série de critérios, as seguintes partes do *script* foram criadas:

Primeira Parte: “Se a frase contiver alguma palavra do seguinte conjunto de palavras: ("odysseus", "inventory", "look", "quit", "restore", "restart", "save", "time", "wait"), retorne escrito para mim apenas a primeira palavra deste conjunto que aparecer, sem explicações ou comentários. Caso a frase não contenha nenhuma das palavras do conjunto, retorne para mim apenas a palavra äction:. A frase é: "+ command + ".”

Este é o primeiro caso de análise da entrada inserida pelo usuário. “command” refere-se a variável do código *Python* que contém o texto de entrada do usuário. Para este caso, será detectado se o jogador deseja realizar algum desses comandos únicos que costumam ser padrões de jogos de ficção interativa. Eles estão em um escopo separado porque não seguem o padrão de entrada "verbo + objeto". Esses comandos consistem em apenas uma palavra, mas ainda permitem que o jogador use uma frase ou sentença para ativá-los.

Para evitar problemas de retorno, foi restringido para este e outros casos que seja retornada apenas a primeira instância que estiver presente na sentença inserida, para que assim o *ChatGPT* não acabe por pegar um termo incorreto do *input*.

A parte do “sem explicações ou comentários” segue a mesma linha da primeira versão do *script*, pois o GPT em muitos momentos acaba explicando a resposta dada para aquela entrada de texto. Isto o impede de enviar tal mensagem.

Ao final, a parte do retorno da palavra “action” ocorre para controle de condicional dentro do código para justamente verificar se é um comando de jogo único ou se acabará sendo uma ação dentro do jogo como pegar objetos, abrir portas, entre outros.

Segunda Parte: “Tenho a seguinte frase: "+ command + ". Identifique o contexto desta frase. Caso remeta a "ir a algum lugar", me retorne apenas a palavra "go" sem nenhuma explicação ou comentário a respeito. Do contrário, caso remeta a "ler alguma coisa", me retorne apenas a palavra "read" sem nenhuma explicação ou comentário. Do contrário, caso remeta a "largar algo ou alguma coisa", me retorne apenas a palavra "drop". Do contrário, caso remeta a "pressionar algo ou alguma coisa, como press ou push", me retorne apenas a palavra "push" sem nenhuma explicação ou comentário. Do contrário, caso remeta a "mover algo ou alguma coisa", me retorne apenas a palavra "move". Do contrário, caso remeta a "entrar em algum lugar", me retorne apenas a palavra "enter" sem nenhuma explicação ou comentário. Do contrário, caso remeta a "pegar alguma coisa", me retorne apenas a palavra "get" sem nenhuma explicação ou comentário. Do contrário, caso remeta a escalar ou subir, me retorne apenas a palavra "climb" sem nenhuma explicação ou comentário. Do contrário, caso remeta a "atacar" algo ou alguém, me retorne apenas a palavra "attack" sem nenhuma explicação ou comentário. Do contrário, caso remeta a "matar" algo ou alguém, me retorne apenas a palavra "kill" sem nenhuma explicação ou comentário. Do contrário, caso remeta a "abrir algo ou alguma coisa" me retorne apenas a palavra "open".”

Esta segunda parte é semelhante a primeira versão mencionada do *script* original, onde ele obtém o verbo de ação. Porém, aqui o GPT passa a realizar a função de justamente interpretar o contexto e o significado da sentença, então situações como “take me to” e “get me to” serão interpretadas como o comando “go” presente no dicionário do jogo, “throw away” e “get rid of” serão interpretados como o comando “drop”, “get into” e “adentrate” serão interpretados como o comando “enter” e também “punch”, “kick” e derivados de golpes ou briga serão interpretados como o comando “attack”. Estes são apenas alguns exemplos do funcionamento para demonstração da funcionalidade. Nos resultados obtidos são apresentadas comparações do jogo rodando sem a integração e com a integração tratando alguns dos casos citados.

Na maioria dos casos, o jogo nem ao menos reconhece certas palavras por não estarem cadastradas em seu dicionário de jogo. Mesmo o jogo “Zork I” tendo um dicionário consideravelmente amplo, não consegue cobrir todos os casos possíveis. É neste momento que a integração mostra a sua utilidade e melhoria referente à jogabilidade, deixando o jogador menos restrito ao dicionário do jogo e tornando-o mais dinâmico, amigável e possivelmente até mais divertido por conta de ser possível brincar mais com as sentenças para realizar determinadas ações dentro do jogo.

Terceira Parte: “Tenho a seguinte frase: "+ command + ". Dentro do contexto da mensagem, caso esta contenha uma palavra (substantivo) que remeta a um objeto ou lugar ou direção ou personagem como por exemplo "spider", "troll", entre outros, retorne para mim apenas a primeira dessas palavras que aparecer na ordem da sentença, sem realizar nenhuma explicação ou comentário sobre, apenas a palavra.”

Esta terceira parte é o reconhecimento do alvo da ação. Na segunda parte obtemos o retorno da ação, porém esta ação deve estar direcionada a algo ou alguém. Neste contexto, o *script* realiza a análise de qual é justamente o alvo desta ação com alguns ajustes finos na tentativa de tornar o retorno mais preciso e também com que o GPT se confunda menos. Como o jogador pode inserir adjetivos ao objeto ou utilizar palavras compostas, o *script* foi ajustado para que ele pegue apenas justamente o alvo genérico da ação.

No geral, estas três partes da interpretação resolvem a maioria das questões do jogo, porém existem alguns casos isolados e específicos que acabaram necessitando um tratamento isolado por conta do seu funcionamento e também da sintaxe dentro do jogo. Por exemplo, em certo momento do jogo será possível obter uma lanterna, e esta seguirá por um comando de três palavras como por exemplo “turn lamp on” para poder ligá-la.

Como pode ser observado, é necessário um terceiro termo para que o jogo entenda a ação do jogador de ligar a lanterna, fugindo do padrão “verbo + objeto”. Cada jogo terá comandos como este, muitos divergindo entre si e tendo sintaxes completamente diferentes. Por conta disso, este trabalho voltou-se para implementar a ferramenta em cima do jogo *Zork I* para demonstração do funcionamento e utilidade da integração.

Para este caso, foi realizada a seguinte adaptação:

Tratamento da Lanterna: “Tenho a seguinte frase: "+ command + ". Identifique se a frase tem o contexto de ligar ou desligar uma "lamp"e que não tenha em sua estrutura as palavras "grab", "get", "take" ou "push". Caso remeta a ação de ligar, me retorne apenas a palavra "on" sem nenhuma explicação ou comentário. Do contrário, caso remeta a ação de desligar, me retorne apenas a palavra "off" sem nenhuma explicação ou comentário. Do contrário, retorne para mim N̈aN̈.”

Pegando o contexto de ligar ou desligar a lanterna, viu-se a necessidade de excluir alguns verbos que poderiam também remeter a ação de ligar a lanterna que influenciassem na ação de obter a lanterna dentro do jogo, fazendo com que fosse frustrante o ato de constantemente tentar pegar o objeto até que o GPT interpretasse corretamente. Interpretando corretamente, o GPT deve retornar a ação exata referente a lanterna (ligar ou desligar) para unir os termos no tratamento dentro do código. Ao final, o retorno de “NaN” é apenas um termo/palavra qualquer para o GPT retornar ao código por questões

de tratamento condicional, para evitar que houvesse um retorno errôneo ou aleatório. Ou seja, caso não ocorra a situação de retorno de algum comando, é retornado apenas a palavra “NaN”.

Além da lanterna, há o caso do combate que utiliza a sintaxe de quatro termos como por exemplo “kill troll with sword”, o qual também acabou necessitando uma atenção exclusiva para este caso. Esta situação foi adaptada para o seguinte:

> **Tratamento do Combate:** “Se a frase conter um inimigo como "spider", "troll", entre outros, junto também de uma arma de combate, retorne escrito para mim apenas a arma de combate somente se todos os requisitos forem cumpridos (uma arma junto de um inimigo na mesma frase), sem comentários ou explicações. Caso a frase não cumpra o conjunto de requisitos, apenas retorne me avisando o não cumprimento. A frase é: "+ command + "."”

Esta interação foi tratada de forma que o comando fosse ativado apenas na situação de haver uma arma junto de um alvo (inimigo) para que não conflitasse com questões como jogar a arma fora ou obtê-la. Alguns exemplos de inimigos foram dados para o GPT entender eles como personagens e evitar que interprete o contexto errado.

Após a análise, o comando devidamente configurado será enviado de volta para o interpretador Frotz, alinhando-se com a estrutura do seu código como na Figura 8. Como resultado, mesmo que várias palavras sejam inseridas no campo de entrada, apenas as palavras-chave essenciais para o comando serão consideradas. Esse comando específico será então retornado ao interpretador, facilitando uma entrada de texto dinâmica e adaptável sem a necessidade de extensas modificações no código-fonte. Uma visão mais simples da implementação pode ser vista em Figura 9.

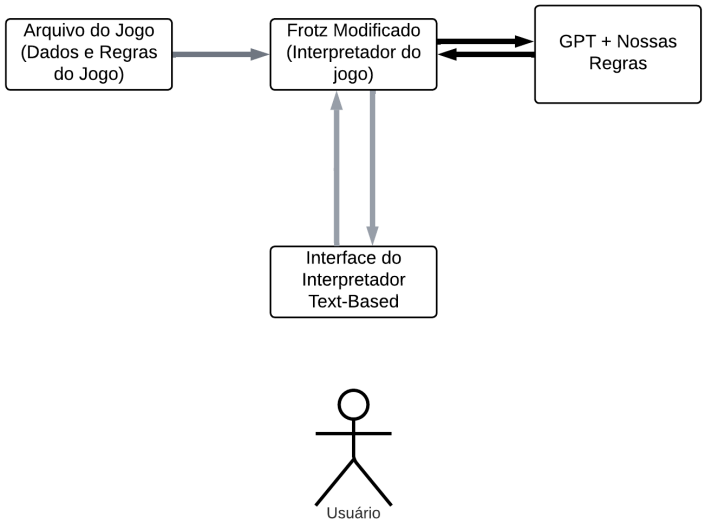
4.4 Avaliação da solução

Nosso trabalho priorizou testes locais realizados pelo estudante e pelos orientadores desta pesquisa utilizando a integração desenvolvida do *ChatGPT* no jogo *Zork I*. Foram realizadas chamadas de voz virtuais para discussões e sugestões de melhorias, refino de métodos já existentes e testes da integração conforme eram realizadas alterações. Ao final do trabalho, foi possível realizar uma avaliação referente a efetividade e viabilidade da solução.

Para avaliar a efetividade e viabilidade de nossa solução, foram levados em conta a complexidade de integração da *API* do *ChatGPT* com o código dos jogos *Z-Machine* por meio do *open-source* do Frotz e também a aceitação do comando retornado ao jogo após inserção do dado de entrada no *script*.

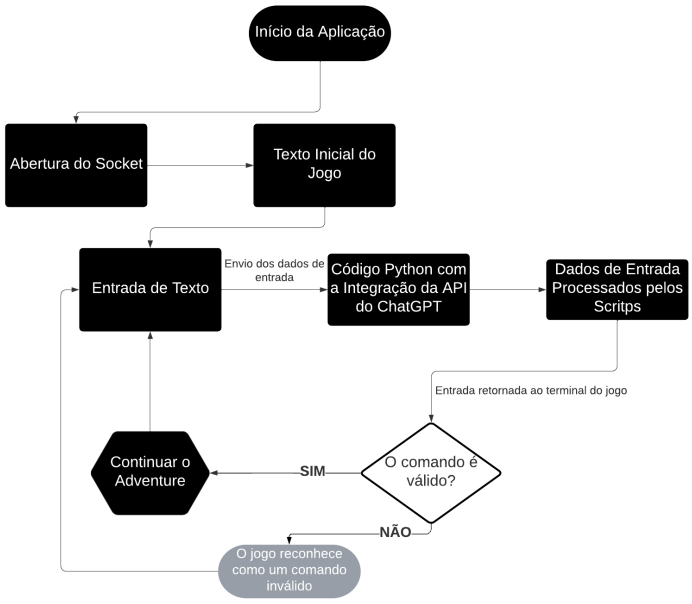
Em relação a complexidade, avaliou-se ser de um nível moderado de dificuldade a realização dos ajustes no código do Frotz por não haver uma documentação a respeito

Figura 8 – Arquitetura preliminar do sistema (flechas mais escuras indicam troca de dados via conexão do *socket*; flechas mais claras consistem na relação original do sistema não modificada.)



Fonte: O autor.

Figura 9 – Fluxograma de execução.



Fonte: O autor.

das funções, chamadas e métodos criados pelo autor, porém a comunicação de um cliente em C com um servidor criado em Python e o desenvolvimento de uma integração da *API* do *ChatGPT* em Python apresentou-se como algo de baixa complexidade.

A avaliação da qualidade da solução foi realizada por meio de testes locais em

uma perspectiva qualitativa, para identificar problemas durante o uso e para verificar se as melhorias implementadas não descaracterizaram a ficção interativa e se houve melhora efetiva da experiência dos jogadores. Essa avaliação foi feita a partir da observação dos dados obtidos em uma comparação em jogar a versão original do jogo *Zork I* e a versão com interface do *ChatGPT*.

Os testes foram realizados da seguinte forma: o estudante, que possuía a integração já configurada e instalada em sua máquina, compartilhava sua tela com os orientadores e amigos presentes, que forneciam instruções sobre o que deveria ser inserido no jogo com base no texto apresentado. Cada pessoa utilizou abordagens e palavras diferentes, e alguns casos foram incluídos como apêndices neste trabalho para melhor visualização e comparação dos comandos no jogo original com a integração do *ChatGPT*. Os problemas e inconsistências identificados nessa experiência foram cruciais para o desenvolvimento e ajuste do trabalho, pois pessoas diferentes pensam de maneira distinta e trazem interações e retornos variados, tornando a análise e validação da consistência da ferramenta mais precisa, refinando ainda mais a precisão e a consistência do uso da ferramenta. Alguns jogadores avançaram mais na aventura, enquanto outros realizaram apenas alguns testes de termos conforme o jogo prosseguia, resultando na criação de uma lista de melhorias e correções de problemas, tanto de forma geral no jogo quanto em situações mais específicas. Um exemplo foi o uso da lanterna, pois o GPT frequentemente se confundia ao interpretar se o usuário estava tentando pegar a lanterna ou ligá-la, fazendo com que a maioria das pessoas ficasse travada nesse trecho. Isso exigiu um ajuste específico, criando-se um método de tratamento apenas para a lanterna, conforme citado anteriormente no desenvolvimento.

Além disso, o estudante realizou individualmente várias melhorias ao longo do tempo para aprimorar o trabalho, como ajustar alguns padrões de retorno do GPT que precisavam ser tratados (explicações de suas respostas, pontuações, termos incorretos), realizar novos testes com novas palavras, avançar um pouco mais no jogo e refinar o código e os *scripts* conforme necessário. Este período de desenvolvimento (de março a junho de 2024) foi extremamente focado em aprimorar a ferramenta, corrigir problemas ao máximo, refinar e torná-la utilizável de maneira adequada, para que fosse interessante tanto para quem já conhece o gênero de jogos quanto para aqueles que estão tendo seu primeiro contato com a experiência.

Um ponto importante a ser observado é a questão das atribuições específicas dentro do *script*. Poderão existir casos que fogem à regra do padrão de jogabilidade que não serão tratadas corretamente por meio de um script muito genérico enviado ao GPT, fazendo com que o retorno se torne inconsistente e retornem comandos errôneos como já vistos anteriormente. Por haverem diferentes regras e dicionários entre os diferentes jogos, para este trabalho em questão mostrou-se mais eficiente em questões de resultado e precisão tratar certos casos específicos separadamente, enquanto a maior parte do jogo consegue

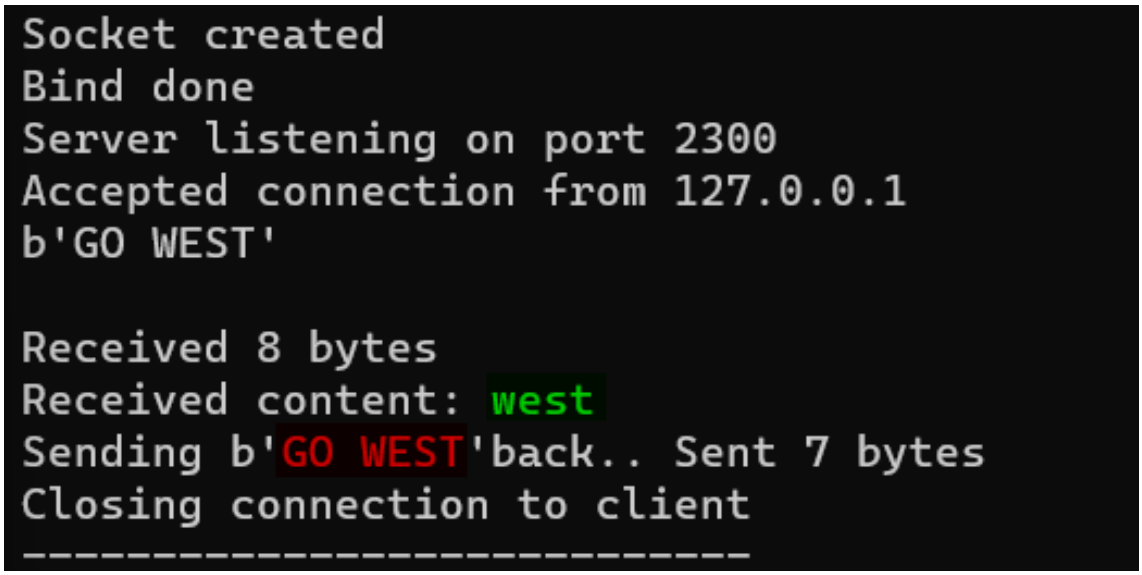
ser tratada com os comandos mais genéricos.

5 RESULTADOS

Nas Figuras 10 e 11, mostramos algumas capturas de tela dos resultados obtidos inicialmente neste trabalho, com base em nossos experimentos, da comunicação em andamento.

Na Figura 10, está o recebimento no código Python, com a integração do *ChatGPT*, de um texto de entrada vindo do comando de entrada dado pelo jogador diretamente do campo do jogo. No campo "Received content" observa-se o texto digitado pelo jogador, o qual foi "west". Este é um comando válido de movimentação como já mencionado anteriormente, onde questões de movimentação com pontos cardeais são aceitos com um único termo. Esta captura de tela foi obtida inicialmente neste trabalho como forma de testar a comunicação do campo de entrada do interpretador FrotZ com o *ChatGPT* por meio da integração em Python. Sendo assim, ao inserir este comando do jogador no *script* inicial citado neste trabalho, podemos observar que na linha de "Sending" já temos o comando escrito corretamente e sendo enviado de volta para a tela do jogo.

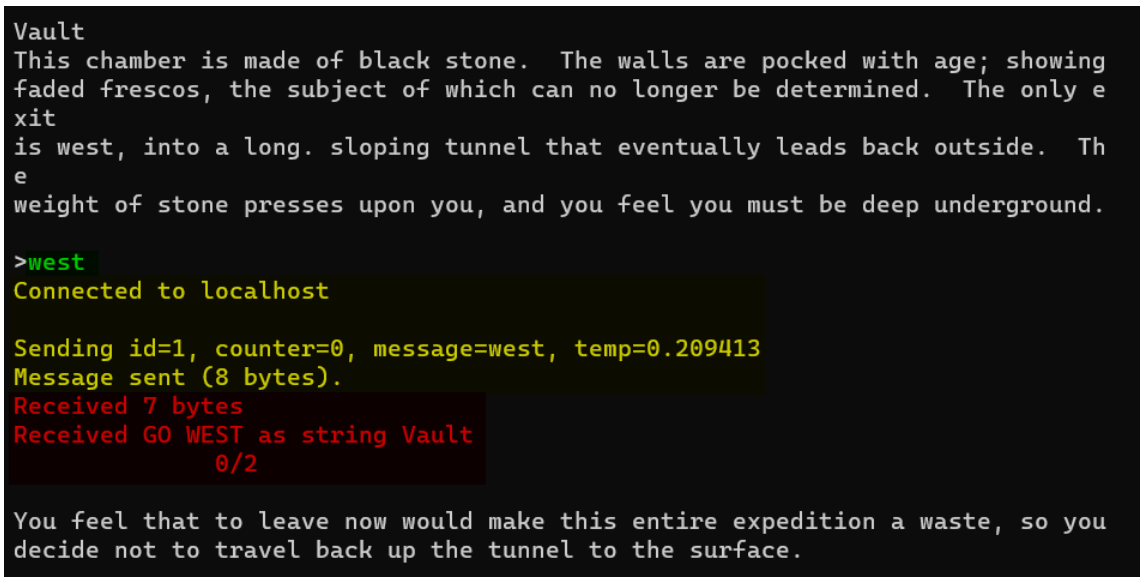
Figura 10 – Servidor em Python.



Fonte: O autor.

Já na Figura 11 temos o texto referente ao jogo na parte superior em letras brancas, logo abaixo o campo de inserção de comandos pelo usuário contendo seu comando em letras verdes e em letras amarelas a conexão sendo estabelecida entre este terminal de execução do jogo com o terminal em Python, enviando este dado de entrada do jogador para a integração do *ChatGPT*. Após isso, em letras vermelhas, observa-se o retorno do comando após tratamento com o *ChatGPT* via *script* inicial deste trabalho, onde a mensagem abaixo é a continuação do jogo descrevendo o cenário após execução do comando "GO WEST", demonstrando que a execução foi um sucesso.

Figura 11 – Cliente integrado ao código fonte do Frotz.



Fonte: O autor.

Alguns poucos testes foram realizados utilizando a integração separadamente para otimização do tempo. Isso foi realizado para que não fosse necessário executar o interpretador em toda situação que fosse realizado algum ajuste ou teste. Nestes testes foi utilizada a sentença "I want to go and play in west or north and play with kids" para forçar o funcionamento de interpretação da integração, por mais que seja um caso irreal de jogabilidade dentro do jogo.

Após algumas iterações com o *script* inicial, notam-se já problemas em relação a estruturação do comando junto da interpretação das informações necessárias podendo ser observado na Figura 12.

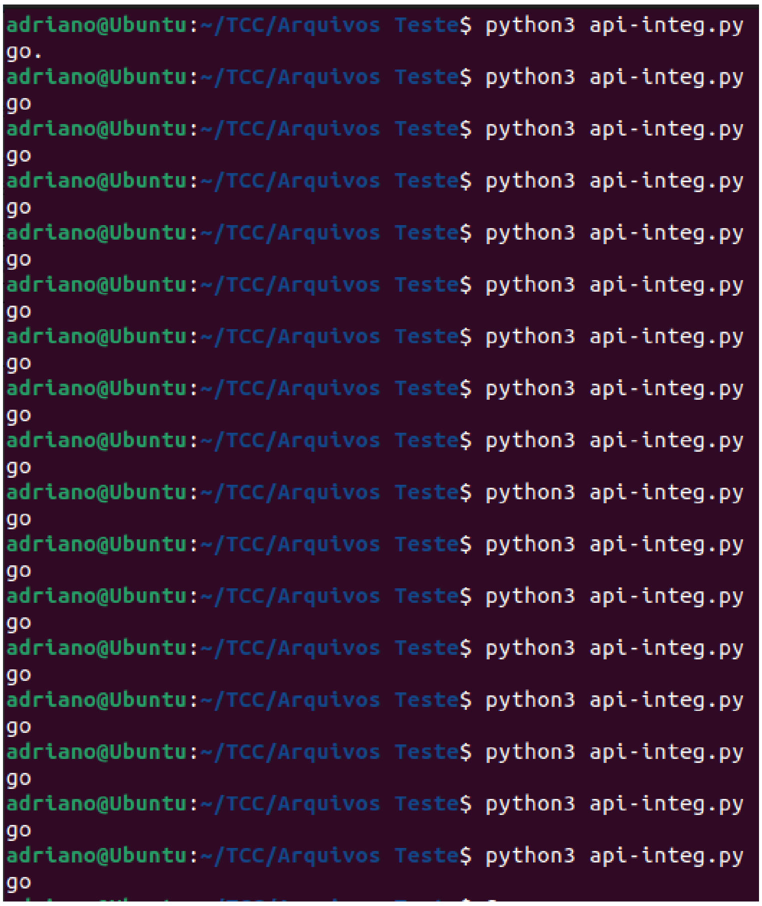
Figura 12 – Teste do script inicial com a frase de entrada “I want to go and play in west or north and play with kids” e com respectivos resultados de várias execuções.

```
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
go west
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
go west
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
play kids
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
want to play, west
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
want go + kids
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
want to play, west
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
go and play, west
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
want to play, west
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
go play
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
go play
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
go west
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
go west
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
go west
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
go west
adriano@Ubuntu:~/TCC/Arquivos Teste$ python3 api-integ.py
want to go + west
```

Fonte: O autor.

Houve tentativas de consertar este *script*, porém todas sem sucesso. Com isso, foi realizada a divisão em partes da interpretação para que se tornasse mais específica e aumentando a precisão do retorno dos comandos. Testes iniciais foram realizados com a obtenção do comando de ação, o qual se mostrou muito mais consistente como visto na Figura 13.

Figura 13 – Teste isolado da integração com parte do script referente ao comando de ação.

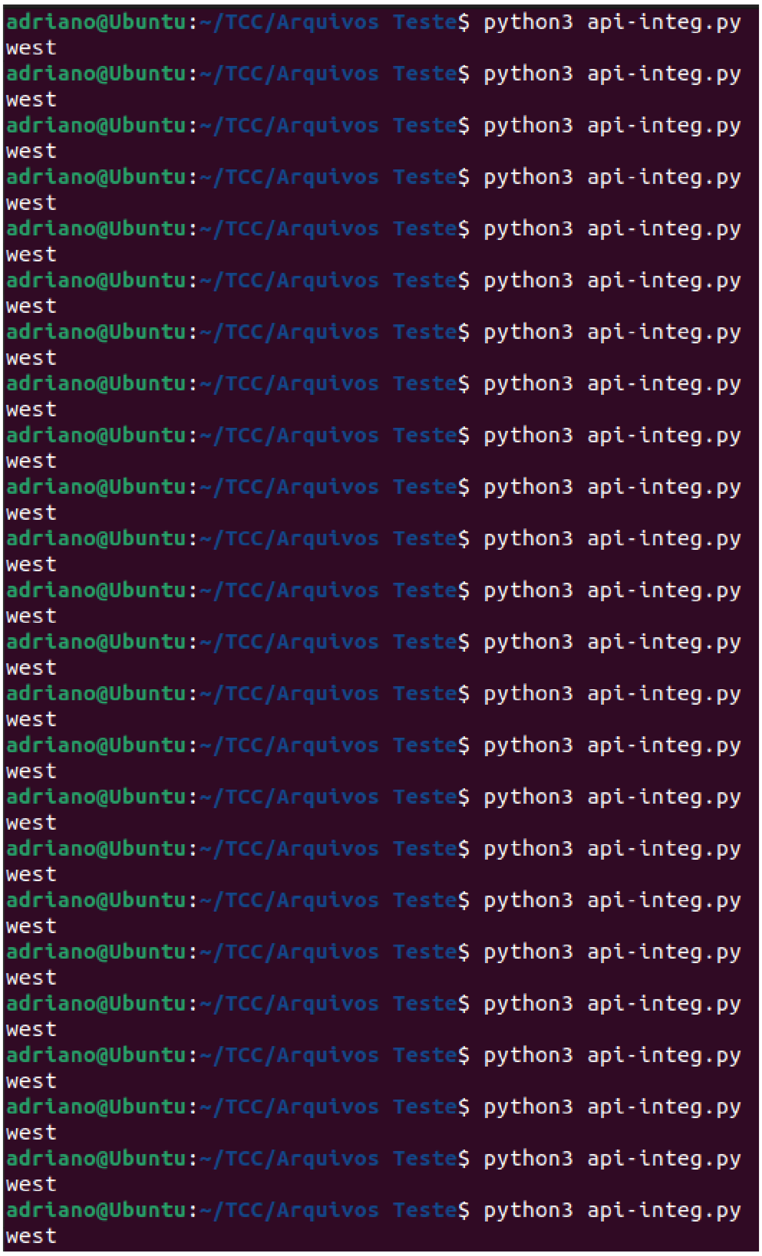


Fonte: O autor.

Porém o GPT não é completamente preciso e pode acabar retornando alguma instância incorreta sem qualquer relação com o contexto dado no *script* ou aleatoriamente com um ponto final, como verificado na captura de tela.

Após isto, foram realizados testes com o alvo desta ação, o qual se mostrou muito mais consistente e em testes isolados retornou sempre o comando desejado como observado na Figura 14.

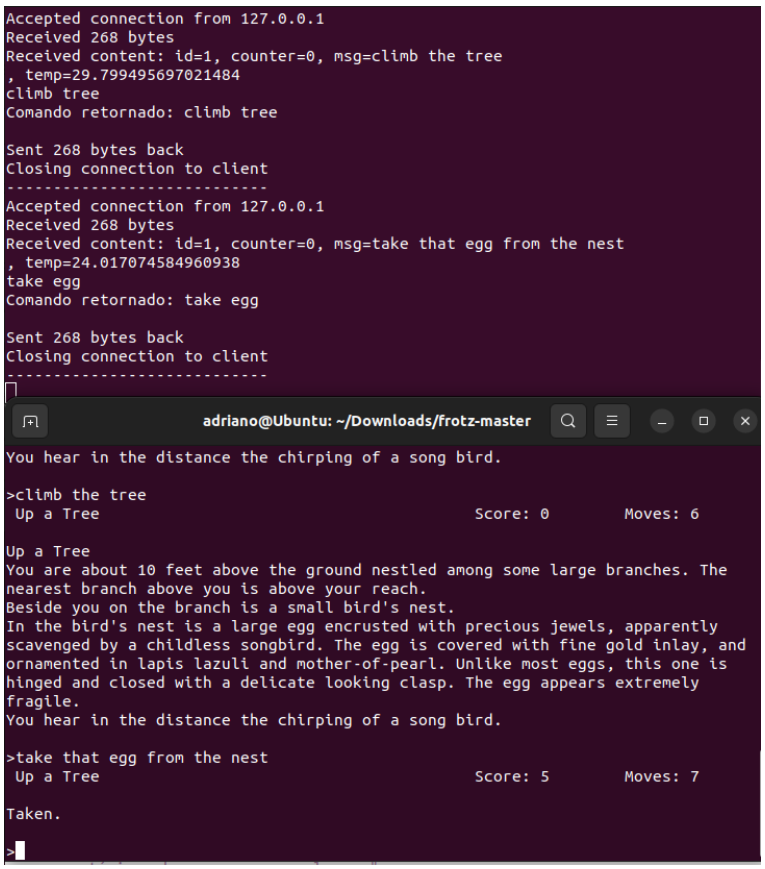
Figura 14 – Teste isolado da integração com parte do script referente ao alvo da ação.



Fonte: O autor.

Foram realizados ajustes conforme os testes continuavam e no momento em que foi identificado que a integração estava consistentemente funcionando, foram iniciados os testes com os inputs dados dentro do jogo. Em um primeiro momento, foi verificado o funcionamento apenas com frases complexas que remetam ao comando diretamente como na Figura 15, onde “take” e “egg” são um comando do dicionário do próprio jogo.

Figura 15 – Teste da integração com o jogo Zork I com uma frase.

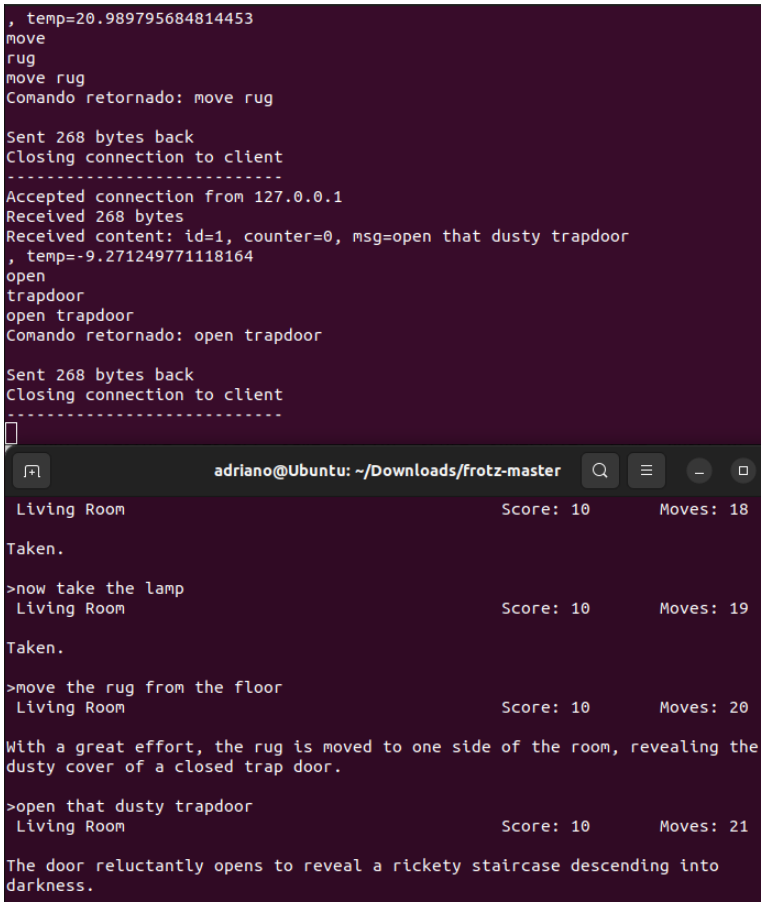


Fonte: O autor.

Estes exemplos de imagens (Figura 15) estão definidos com a parte superior da captura sendo o servidor em Python com algumas informações para observação do funcionamento do recebimento, interpretação e envio do comando. Pode ser observado que a mensagem recebida foi “take that egg from the nest”, sendo interpretado e retornado apenas o comando “take egg” para o interpretador do jogo.

Na Figura 16 foi identificado outro caso de frases com comandos já existentes no dicionário, porém desta vez com o uso de adjetivos junto do objeto ou local alvo, onde o GPT interpretou corretamente qual era o objeto genérico alvo da ação mencionada no texto de entrada do jogador.

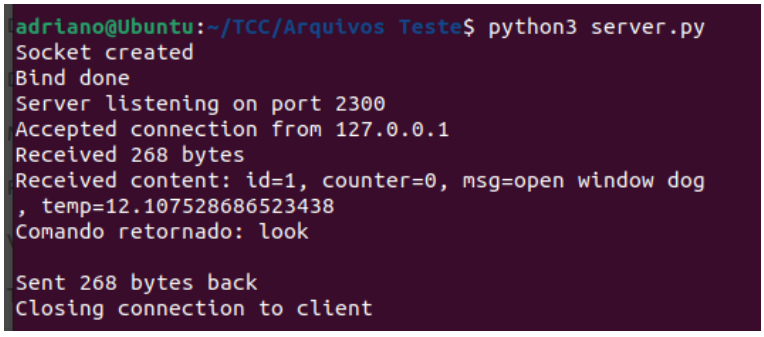
Figura 16 – Teste da integração com o jogo Zork I com a frase contendo um adjetivo.



Fonte: O autor.

Como citado anteriormente, na Figura 17 podemos observar um momento onde o GPT interpretou um contexto sem relação nenhuma com o texto de entrada enviado pelo usuário. Estes casos fogem de nosso controle, então em certos momentos isto irá ocorrer por conta de limitações da API que ainda encontra-se em versões iniciais.

Figura 17 – Retorno incorreto do ChatGPT quando o comando de entrada é “window dog”.

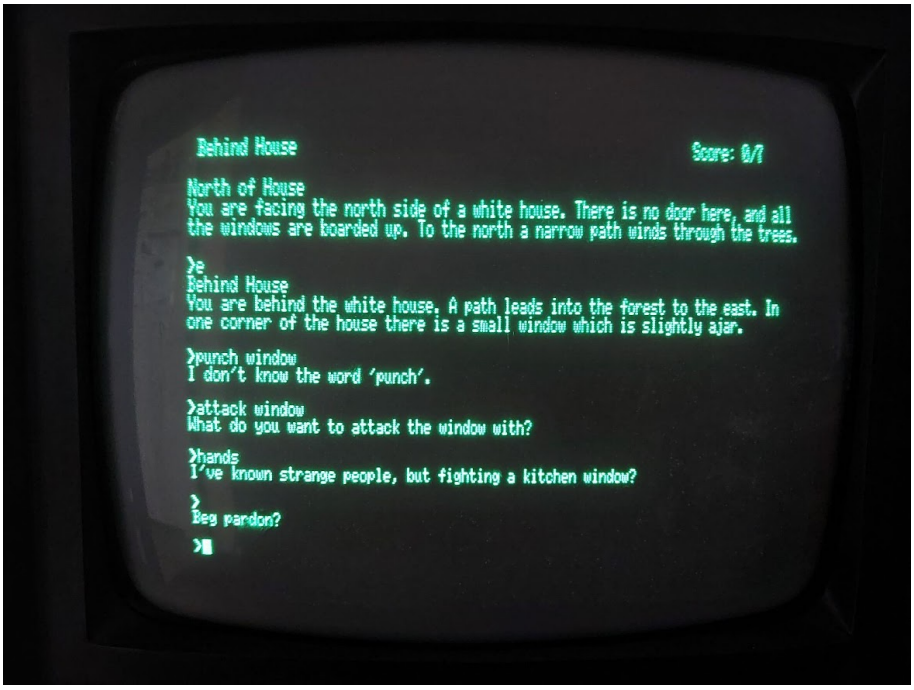


Fonte: O autor.

Por fim, nos apêndices deste trabalho estão os resultados de comparação do funcio-

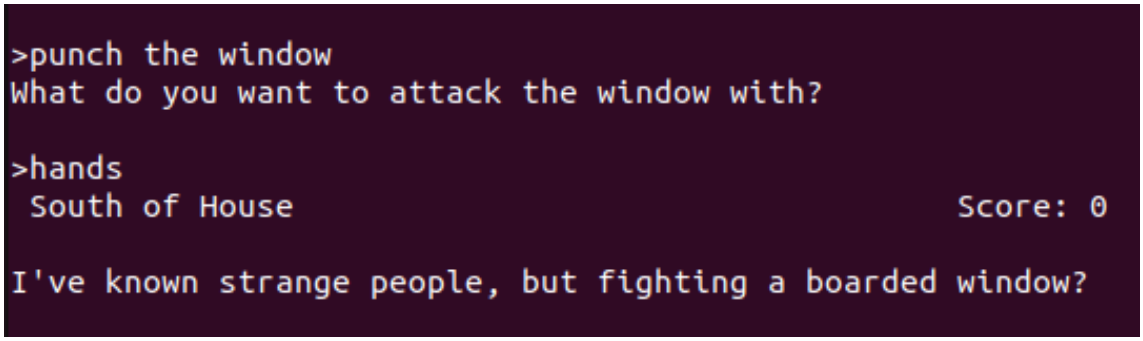
namento do jogo com e sem a integração do ChatGPT, onde está ocorrendo a interpretação de termos, retornado o comando e sendo aceitos dentro do próprio jogo. Além disso, o interpretador FrotZ modificado e a integração da API do *ChatGPT* podem ser encontrados na página do *Github* "https://github.com/AdrianoTD/FrotZ-Interpreter-Using-ChatGPT-API-for-Input-Interpretation". Na Figura 18 e Figura 19 vemos um exemplo de como a integração passa a reconhecer um comando inexistente no jogo original.

Figura 18 – Jogo original não reconhecendo o termo "punch".



Fonte: O autor.

Figura 19 – Integração do jogo com a API passando a reconhecer e validar o termo "punch".



Fonte: O autor.

6 CONSIDERAÇÕES FINAIS

Em síntese, conseguimos implementar um protótipo funcional e testar o conceito proposto, mostrando que é possível integrar tal mecanismo. Configuramos o ambiente com o interpretador Frotz compilado com a biblioteca de *socket*, assim como o lado em Python, que possui a *API GPT* em execução para possibilitar a comunicação cliente-servidor e análise de mensagens.

Como os resultados dos experimentos que realizamos até o momento sugerem, a integração de uma *engine* de ficção interativa com uma *API* de *chatbot* generativa como o *ChatGPT*, para ajudar a melhorar a experiência de fluxo do jogo para o usuário, realmente se mostrou possível e possui um grande potencial para ser uma melhoria para esse gênero de jogo. Em tempos em que os hábitos de leitura diminuíram drasticamente entre a parcela mais jovem da população, talvez o grande apelo da IA possa ajudar a aumentar novamente a popularidade dos jogos de ficção interativa e preencher essa lacuna, pelo menos parcialmente. Esperamos aprimorar o *script* de interpretação de comandos e realizar testes mais aprofundados nas etapas subsequentes, bem como investigar a possibilidade de usar o *ChatGPT* para transformar também as mensagens de saída.

Alguns desafios foram encontrados em relação ao comportamento do *ChatGPT*, pois seus retornos inconsistentes e, às vezes, aleatórios tornaram desafiadora a elaboração de modos que utilizassem os *scripts* junto com a estruturação do código de maneira precisa e consistente. Os *scripts* se tornaram o fator crucial para o desenvolvimento e obtenção de resultados neste trabalho, conforme foi elaborado, ainda que haja espaço para melhorias. Além disso, em alguns momentos ocorreram falhas na comunicação da *API*, impedindo a conexão com o servidor da OpenAI.

Para isso, devemos continuar testando a funcionalidade do *script* para que com isso sejam identificados possíveis problemas que possam gerar complicações no sistema do jogo ou até mesmo da interpretação do *ChatGPT*. Por mais que tenha sido mostrado um resultado com as capturas de tela, ainda está sendo visualizada uma melhoria no comando para justamente reduzir o retorno de comandos incompatíveis com o jogo e evitar esquecimento das regras por parte da IA. Além disso, os comandos de jogabilidade do jogo podem ser utilizados de maneira que não utilize a integração com o GPT, realizando a consulta diretamente no dicionário do próprio jogo e com isso fazendo com que o jogador tenha a opção de realizar certos comandos ou a maioria deles da maneira clássica. Esta ação de ignorar a integração serve apenas para comandos de uma única palavra ou letra, já que estas descartam interpretação de contexto.

Portanto, como trabalhos futuros, propomos revisar a estrutura de *scripts* para tornar o retorno ainda mais eficiente, seja elaborando um formato diferente ou refinando o que já foi criado. Também consideramos realizar comparações entre o uso do *ChatGPT* e outros chatbots generativos para determinar qual seria o mais adequado. Além disso, vislumbramos a possibilidade de enviar diretamente ao GPT as informações descritas pelo

jogo sobre o cenário e objetos, permitindo que ele tenha conhecimento total do ambiente ou contexto inserido, o que poderia facilitar a interpretação de termos e ações dentro do jogo.

Como este trabalho visa apresentar apenas uma potencial melhoria na experiência do usuário, também seria valioso, como trabalho futuro, realizar testes com usuários ou outros métodos que avaliem essa questão especificamente, verificando se, no geral, a experiência dos usuários foi realmente aprimorada.

Vale ainda destacar que, como resultado deste trabalho foi apresentado, no formato de *short-paper*, um artigo no XXII Simpósio Brasileiro de Jogos e Entretenimento Digital (SBGames), o qual ocorreu no período de 6 à 9 de Novembro de 2023 na cidade de Rio Grande-RS (DOTTA; THIELO; CHEIRAN, 2023).

REFERÊNCIAS

AGROVER112. **Is ChatGPT an RNN?** 2023. Acesso em: 13 jun. 2023. Disponível em: <https://medium.com/orel-group/is-chatgpt-an-rnn-e2df2d8956f9>. Citado na página 29.

ANAND, A.; POLYAK, E. **EXPLORING THE POTENTIAL OF LARGE LANGUAGE MODELS FOR ENHANCED VIRTUAL NON-PLAYER CHARACTER INTERACTIONS**. 2024. Disponível em: <https://doi.org/10.21125/inted.2024.1269>. Citado na página 19.

BISWAS, S. **Role of ChatGPT in Gaming: According to ChatGPT**. 2023. Acesso em: 12 abr. 2023. Disponível em: <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4375510>. Citado na página 17.

CHEN, Z. et al. **Ambient Adventures: Teaching ChatGPT on Developing Complex Stories**. 2023. Disponível em: <https://arxiv.org/abs/2308.01734>. Citado na página 19.

CHINCHILLA, C. **The past, present, and future of interactive fiction**. 2023. Acesso em: 5 abr. 2023. Disponível em: <https://medium.com/geekculture/the-past-present-and-future-of-interactive-fiction-51f4f589274>. Citado na página 25.

CODECADEMY. **Deep Learning and Generative Chatbots**. 2023. Acesso em: 16 abr. 2023. Disponível em: <https://www.codecademy.com/learn/deep-learning-and-generative-chatbots/modules/generative-chatbots/cheatsheet>. Citado na página 28.

COSSETTI, M. C. **O que é inteligência artificial?** 2018. Acesso em: 8 maio 2023. Disponível em: <https://tecnoblog.net/responde/o-que-e-inteligencia-artificial/>. Citado na página 26.

DOTTA, A.; THIELO, M.; CHEIRAN, J. F. **Use of a generative chatbot as a middleman to improve User Experience in Interactive Fiction games**. Porto Alegre, RS, Brasil: SBC, 2023. Disponível em: <https://sol.sbc.org.br/index.php/sbgames_estendido/article/view/27825>. Citado na página 54.

ESO. **You won't believe how ChatGPT AI changed Skyrim forever!** 2023. Acesso em: 13 jun. 2023. Disponível em: <https://www.youtube.com/watch?v=0wCjosz1vOA&ab_channel=ESO9>. Citado 2 vezes nas páginas 17 e 31.

GONçALVES, T. **PLN: o que é Processamento de Linguagem Natural?** 2021. Acesso em: 22 nov. 2023. Disponível em: <https://www.alura.com.br/artigos/o-que-e-pln>. Citado na página 27.

GOULART, B. **COMO USAR O CHATGPT NO WHATSAPP COM AUTORESPONDER WA**. 2023. Acesso em: 13 jun. 2023. Disponível em: <https://www.youtube.com/watch?v=S1r1cH2K2YE&ab_channel=BrunnoGoulart>. Citado na página 31.

GRIFFITH, D.; JOKISCH, S. **Conforms to Z-Machine Standard 1.0 and supports V1-V6 and V7/V8 games**. 2016. Acesso em: 20 mar. 2023. Disponível em: <http://www.ifarchive.org/if-archive/infocom/interpreters/frotz/>. Citado na página 33.

GRIFFITH, D.; JOKISCH, S. **Frotz - A Portable Z-Machine Interpreter**. 2016. Acesso em: 20 mar. 2023. Disponível em: <https://davidgriffith.gitlab.io/frotz/>. Citado 2 vezes nas páginas 32 e 33.

GROW, A. M.; KHOSMOOD, F. **ChatGPT GameJam: Unleashing the power of Large Language Models for Game Jams**. New York, NY, USA: Association for Computing Machinery, 2023. 51–54 p. (ICGJ '23). Disponível em: <https://doi.org/10.1145/3610602.3610605>. Citado na página 19.

IAROVOI, D.; HEBBLEWHITE, R.; TEH, P. **AI's Influence on Non-Player Character Dialogue and Gameplay Experience**. 2024. Disponível em: <https://doi.org/10.1007/978-3-031-62281-6_6>. Citado na página 19.

IBM. **O que são máquinas virtuais (VMs)?** 2023. Acesso em: 9 maio 2023. Disponível em: <https://www.ibm.com/br-pt/topics/virtual-machines>. Citado na página 32.

IBM. **What is a chatbot?** 2023. Acesso em: 15 abr. 2023. Disponível em: <https://www.ibm.com/topics/chatbots>. Citado na página 28.

JUNIOR, W. G. et al. **How ChatGPT can inspire and improve serious board game design**. 2023. Disponível em: <https://journal.seriousgamessociety.org/~serious/index.php/IJSG/article/view/645>. Citado na página 19.

LIMNA, P. et al. **The use of ChatGPT in the digital era: Perspectives on chatbot implementation**. 2023. Disponível em: <https://doi.org/10.37074/jalt.2023.6.1.32>. Citado na página 17.

LONGO, L. et al. Explainable artificial intelligence: Concepts, applications, research challenges and visions. In: HOLZINGER, A. et al. (Ed.). **Machine Learning and Knowledge Extraction**. Cham: Springer International Publishing, 2020. p. 1–16. ISBN 978-3-030-57321-8. Citado na página 26.

MOSS, R. **A truly graphic adventure: the 25-year rise and fall of a beloved genre**. 2011. Acesso em: 15 abr. 2023. Disponível em: <https://arstechnica.com/gaming/2011/01/history-of-graphic-adventures/>. Citado 4 vezes nas páginas 22, 23, 24 e 25.

OPENAI. **OpenAI API**. 2020. Acesso em: 20 mar. 2023. Disponível em: <https://platform.openai.com/docs/introduction>. Citado 2 vezes nas páginas 17 e 31.

OPENAI. **Introducing ChatGPT**. 2022. Acesso em: 16 abr. 2023. Disponível em: <https://openai.com/blog/chatgpt>. Citado 2 vezes nas páginas 26 e 29.

ORTIZ, S. **What is ChatGPT and why does it matter?** 2023. Acesso em: 8 maio 2023. Disponível em: <https://www.zdnet.com/article/what-is-chatgpt-and-why-does-it-matter-heres-everything-you-need-to-know/>. Citado 2 vezes nas páginas 17 e 27.

OSTECHNIX. **How To Play Colossal Cave Adventure Game In Linux**. 2017. Acesso em: 22 mar. 2023. Disponível em: <https://ostechnix.com/wp-content/uploads/2017/12/Colossal-Cave-Adventure-1.jpeg>. Citado na página 21.

REDHAT. **What is an API?** 2022. Acesso em: 16 abr. 2023. Disponível em: <https://www.redhat.com/en/topics/api/what-are-application-programming-interfaces>. Citado na página 31.

RUSSELL, P. N. S. J. **Inteligência Artificial: Uma Abordagem Moderna (4ª ed.)**. [S.l.]: GEN LTC., 2020. Citado na página 26.

SAS. **O que é processamento de linguagem natural? | SAS**. 2023. Acesso em: 15 abr. 2023. Disponível em: <https://www.sas.com/pt_br/insights/analytics/processamento-de-linguagem-natural.html>. Citado na página 26.

SHRIDHAR, K. **Generative Model Chatbots**. 2017. Acesso em: 10 jun. 2023. Disponível em: <https://medium.com/botsupply/generative-model-chatbots-e422ab08461e>. Citado na página 28.

SINGH, D. **Natural Language Processing – Text Parsing**. 2019. Acesso em: 9 maio 2023. Disponível em: <https://www.pluralsight.com/guides/text-parsing>. Citado 2 vezes nas páginas 27 e 28.

TSAI, C. F. et al. **Can Large Language Models Play Text Games Well? Current State-of-the-Art and Open Questions**. 2023. Disponível em: <https://arxiv.org/abs/2304.02868>. Citado na página 20.

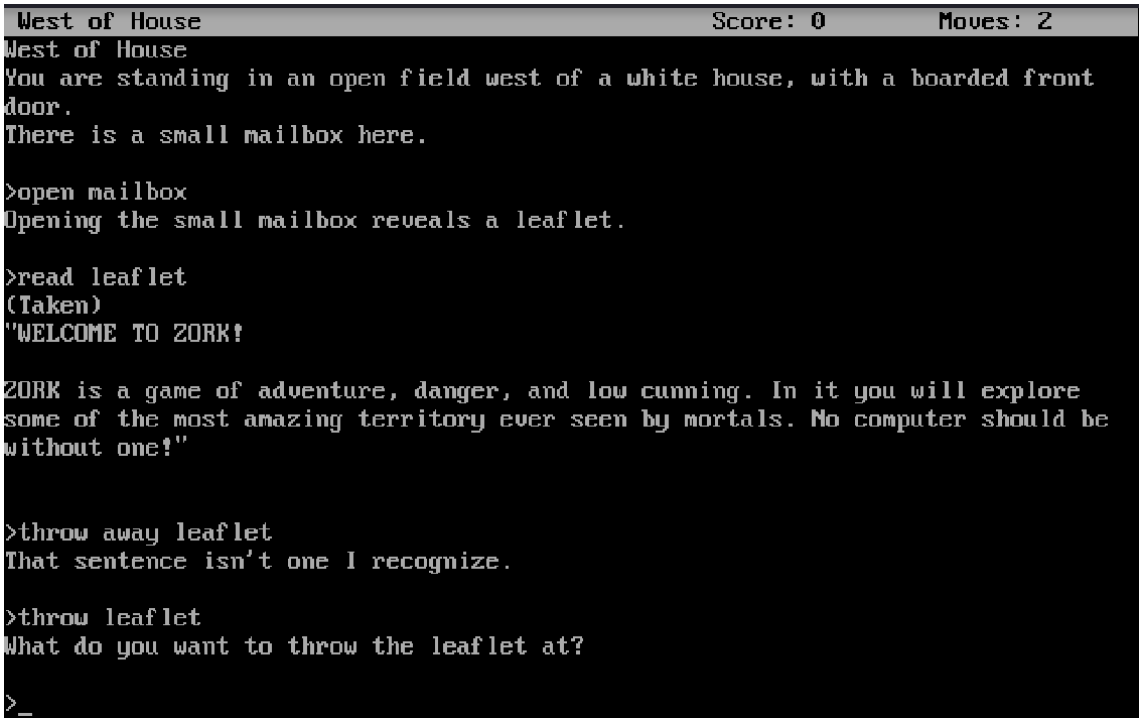
WIKIPEDIA. **Monkey Island**. 2023. Acesso em: 25 abr. 2023. Disponível em: <https://en.wikipedia.org/wiki/Monkey_Island>. Citado na página 25.

ZHOU, S. et al. **Eternagram: Probing Player Attitudes in Alternate Climate Scenarios Through a ChatGPT-Driven Text Adventure**. 2024. Disponível em: <https://arxiv.org/abs/2403.18160>. Citado na página 20.

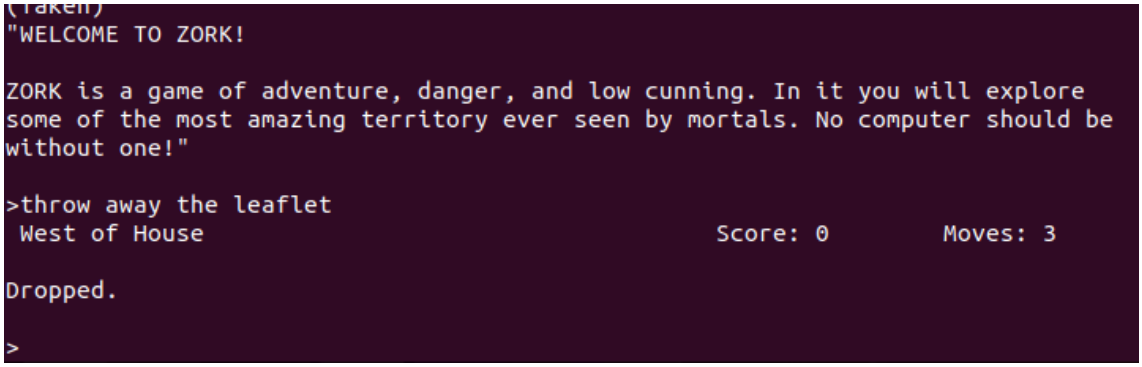
ZURAWEL, K. **Interactive fiction and the origins of the conversational interface**. 2017. Acesso em: 12 abr. 2023. Disponível em: <https://techcrunch.com/2017/03/14/interactive-fiction-and-the-origins-of-the-conversational-interface/?guccounter=1>. Citado na página 21.

7 APÊNDICES

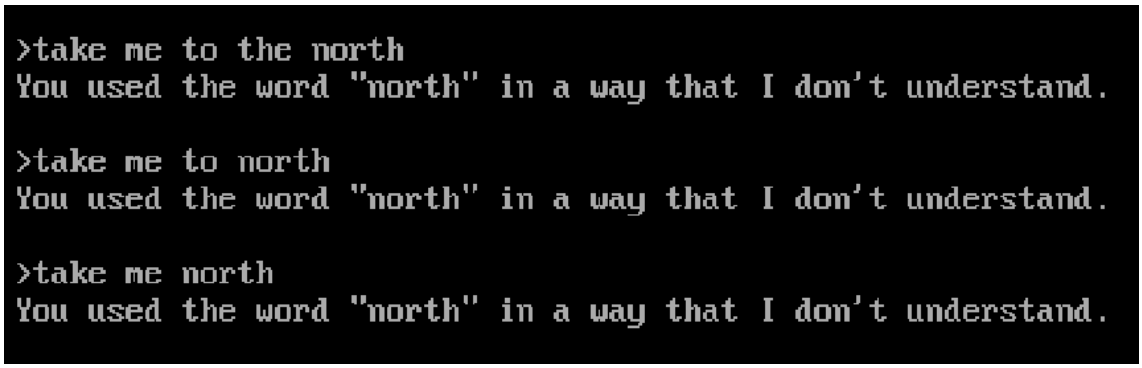
Apêndice A.1 - Jogo original não reconhecendo o termo “throw away”



Apêndice A.2 - Jogo original reconhecendo e validando o termo “throw away”



Apêndice B.1 - Jogo original não reconhecendo o termo “take me”



Apêndice B.2 - Jogo original reconhecendo e validando o termo “take me”

```
>take me to the north
 North of House                                  Score: 0        Moves: 1

North of House
You are facing the north side of a white house. There is no door here, and all
the windows are boarded up. To the north a narrow path winds through the trees.

>take me north
 Forest Path                                     Score: 0        Moves: 2

Forest Path
This is a path winding through a dimly lit forest. The path heads north-south
here. One particularly large tree with some low branches stands at the edge of
the path.

>
```

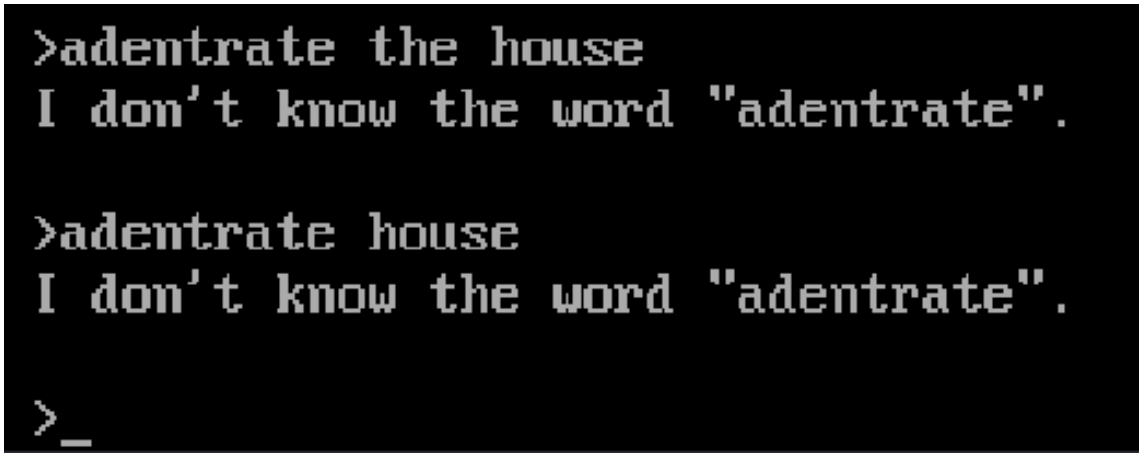
Apêndice C.1 - Jogo original não reconhecendo o termo “ascend”

```
Forest Path
This is a path winding through a dimly lit forest. The path heads north-south
here. One particularly large tree with some low branches stands at the edge of
the path.

>ascend the tree
I don't know the word "ascend".

>ascend tree
I don't know the word "ascend".

>
```

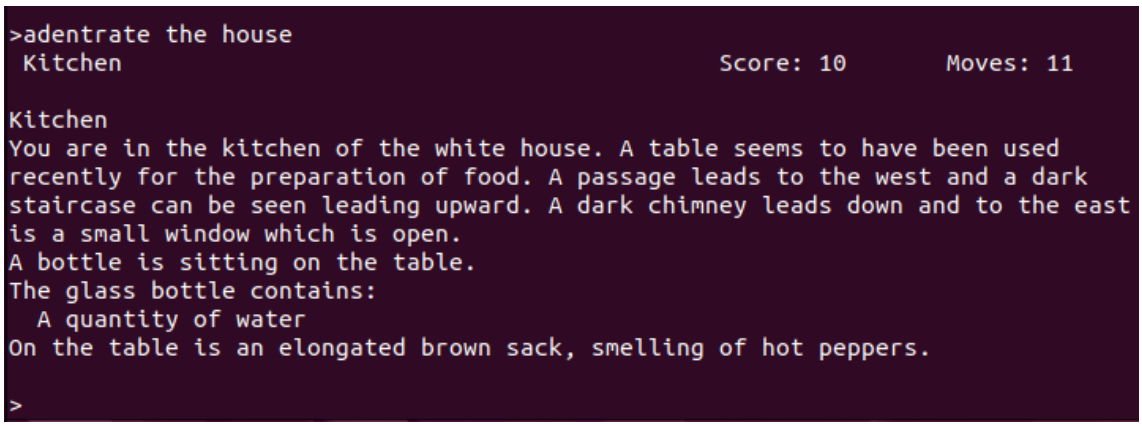
Apêndice C.2 - Jogo original reconhecendo e validando o termo “ascend”

```
Forest Path
This is a path winding through a dimly lit forest. The path heads north-south
here. One particularly large tree with some low branches stands at the edge of
the path.

>ascend tree
 Up a Tree                                       Score: 0        Moves: 6

Up a Tree
You are about 10 feet above the ground nestled among some large branches. The
nearest branch above you is above your reach.
Beside you on the branch is a small bird's nest.
In the bird's nest is a large egg encrusted with precious jewels, apparently
scavenged by a childless songbird. The egg is covered with fine gold inlay, and
ornamented in lapis lazuli and mother-of-pearl. Unlike most eggs, this one is
hinged and closed with a delicate looking clasp. The egg appears extremely
fragile.

>
```

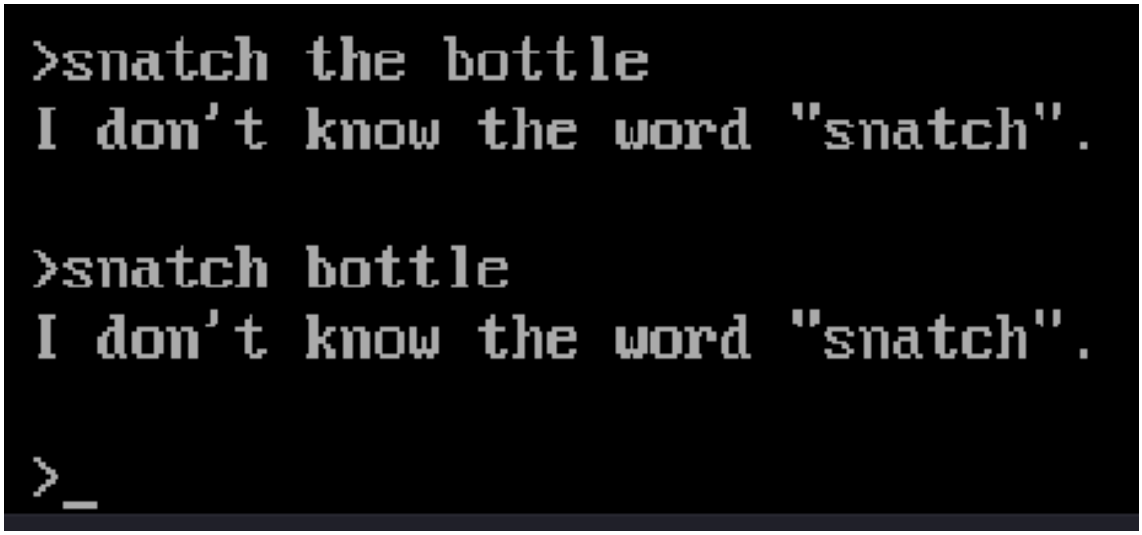
Apêndice D.1 - Jogo original não reconhecendo o termo “adentrate”



Apêndice D.2 - Jogo original reconhecendo e validando o termo “adentrate”

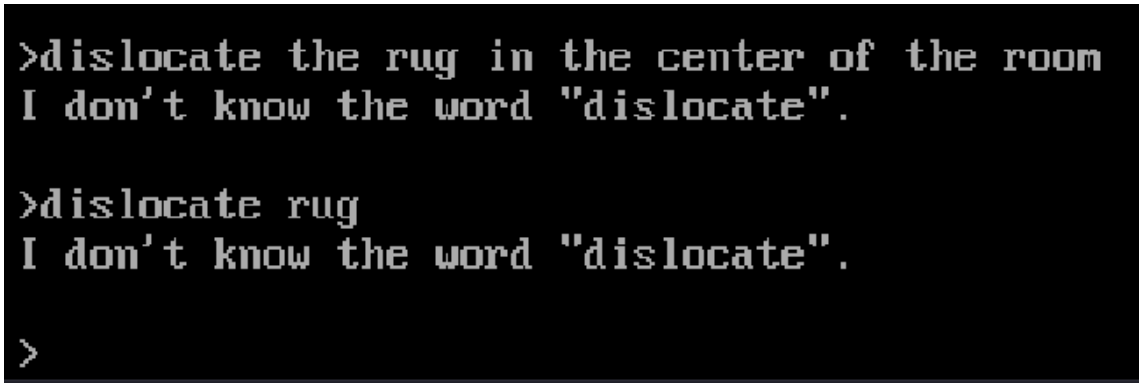


Apêndice E.1 - Jogo original não reconhecendo o termo “snatch”



Apêndice E.2 - Jogo original reconhecendo e validando o termo “snatch”

Apêndice F.1 - Jogo original não reconhecendo o termo “dislocate”



Apêndice F.2 - Jogo original reconhecendo e validando o termo “dislocate”