BIBLIOTECA PARA CRIAÇÃO DE JOGOS UTILIZANDO GERAÇÃO PSEUDO-RANDÔMICA PARAMÉTRICA, REALIDADE VIRTUAL, INTELIGÊNCIA ARTIFICIAL E REDES DE COMPUTADORES

Universidade Federal de Santa Catarina
Programa de Pós-Graduação em
Engenharia de Produção

BIBLIOTECA PARA CRIAÇÃO DE JOGOS UTILIZANDO GERAÇÃO PSEUDO-RANDÔMICA PARAMÉTRICA, REALIDADE VIRTUAL, INTELIGÊNCIA ARTIFICIAL E REDES DE COMPUTADORES

César Ramirez Kejelin Stradiotto

Dissertação apresentada ao
Programa de Pós-Graduação em
Engenharia de Produção da
Universidade Federal de Santa Catarina
como requisito parcial para obtenção
do título de Mestre em
Engenharia de Produção

Florianópolis

2002

César Ramirez Kejelin Stradiotto

BIBLIOTECA PARA CRIAÇÃO DE JOGOS UTILIZANDO GERAÇÃO PSEUDO-RANDÔMICA PARAMÉTRICA, REALIDADE VIRTUAL, INTELIGÊNCIA ARTIFICIAL E REDES DE COMPUTADORES

Esta dissertação foi julgada e aprovada para a obtenção do título de **Mestre em Engenharia de Produção** no **Programa de Pós-Graduação em Engenharia de Produção** da Universidade Federal de Santa Catarina

Florianópolis, 25 de junho de 2002.

Prof. Edson Pacheco Paladini, Dr.
Coordenador do Curso

BANCA EXAMINADORA

Prof. Rogério Cid Bastos, Dr.
Orientador

Prof. João Bosco da Mota Alves, Dr.

Prof. Luiz Fernando Jacintho Maia, Dr.

Prof. Rodolfo Pinto da Luz, Dr.

Profa. Rosane Porto Seleme Heinzen, Msc.

Aos meus pais, Nídia e Alonso,
que sempre apoiaram meus estudos,
e vão estar no meu coração
para o resto da minha vida

Agradecimentos

À Universidade Federal de Santa Catarina,
ao meu orientador Prof. Dr. Rogério Cid Bastos
aos professores do Curso de Pós-Graduação
e aos colegas do Laboratório de Realidade Virtual,
que me ajudaram, sugeriram e incentivaram

Sumário

Lista de Figuras

Lista de Equações

Lista de Tabelas

Lista de Reduções

2D – Bidimensional
3D – Tridimensional
3I – Imersão, Interação, Imaginação
A* – A-Estrela
ADN – Ácido Desoxirribonucléico
AGP – Accelerated Graphics Port
API – Application Program Interface – Interface de Programação de Aplicações
AVC – Ambientes Virtuais Compartilhados
BFS – Best-First Search
CAD – Computer Aided Design – Projeto Auxiliado por Computador
def. vert. – Deflexão Vertical
DOS – Disk Operation System
HMD – Head Mounted Display – Mostrador Montado na Cabeça
IA – Inteligência Artificial
IBM – International Business Machines
IC-CAVE – International Centre for Computer Games and Virtual Entertainment
inc. pulso – Inclinação do Pulso
inc. vert. – Inclinação Vertical
LAN – Local Area Network – Rede de Área Local
LVQ – Learning Vector Quantization
MLP – Multi-Layer Perceptron
NVE – Networked Virtual Environment
RAM – Random Access Memory – Memória de Acesso Aleatório
RFC – Request For Comments – Requisição de Comentários
RNA – Redes Neurais Artificiais
ROM – Read Only Memory – Memória Somente para Leitura
RV – Realidade Virtual
TCP/IP - Transmission Control Protocol / Internet Protocol – Protocolo de Controle de Transmissão / Protocolo de Internet
UDP – User Datagram Protocol – Protocolo de Datagrama de Usuário
VRML – Virtual Reality Markup Language

Resumo

Atualmente os jogos digitais (*videogames*) envolvem uma fatia considerável de mercado, sendo um negócio de bilhões de dólares (BUSHNELL *apud* RADINSKY, 1997 www.sjmercury.com/revolutionaries/bushnell.htm, junho 2001).

Navegando a rede mundial de computadores, pode-se observar a enorme quantidade de jogos disponíveis, para demonstração e potencial compra pelos usuários cibernéticos. Tais demonstrações são tanto para computadores pessoais como emuladores de sistemas proprietários de *videogames*.

A demanda crescente, junto com o avanço da tecnologia e a concorrência por uma fatia deste mercado, forçam os produtores de jogos a fornecerem produtos e serviços inovadores aos seus clientes, que poderão interagir cada vez mais profundamente com os ambientes virtuais dos jogos digitais, e também com outros clientes, espalhados ao redor do mundo.

Sistemas de jogos e periféricos que permitem a interação mais natural com o ser humano, que dão a possibilidade de o usuário interagir com os personagens virtuais através da fala e gestos do corpo, e deixam interagir com outros usuários em outros locais do planeta, ou ainda jogos que são capazes de simular situações e comportamentos parecidos com os personagens da vida real, e produzam ambientes tão imprevisíveis quanto a própria natureza, fazem com que o usuário cada vez mais se convença de que está imerso neste novo mundo virtual.

O trabalho a seguir apresenta uma pesquisa feita sobre jogos de computadores, cujo resultado final foi a criação de uma biblioteca de funções para criação de jogos digitais. Tal biblioteca é capaz de conduzir a máquina na geração, através de número randômicos, de mundos imprevisíveis e situações de clima, assim como são no mundo real.

A biblioteca criada é capaz de permitir a interação de um usuário com a máquina através de várias interfaces, fazendo com que a comunicação entre usuário e computador seja mais humana. Ela ainda possibilita a interação entre vários usuários através de uma rede local de computadores.

Finalmente, esta biblioteca possui algoritmos que gerenciam o comportamento de agentes inteligentes, para que também interajam entre si e com os vários usuários do sistema.

Palavras-Chave: Jogos Digitais, Geração Randômica de Ambientes, Agentes Inteligentes, Ambientes Virtuais Compartilhados, Interação.

Abstract

At present videogames take a considerable piece of market, being a multi-billion dollar business (BUSHNELL apud RADINSKY , 1997 www.sjmercury.com/revolutionaries/bushnell.htm, June 2001).

By navigating through the world wide web, a great number of videogame titles can be seen for demonstration and for sale by cybernetic users. Such demonstrations are for both personal computers and proprietary systems of videogames.

The growing demand, with the technological advance and the competition for this market, forces the game producers to give innovative products and services to their clients. Thus, clients will interact more and more deeply with virtual environments from digital games, and with other clients all over the world.

Game systems and peripherals which allow more natural interactions with humans, enable interactions with virtual characters through body gestures and speech. Moreover, they enable interaction with other players in several places throughout the planet. Also, games that simulate situations or behavior similar to the real ones, and to produce environments as unpredictable as its own nature. As a result, the user feels more and more convinced that he/she is in this new virtual world.

The presented work shows a research about personal computer games, and its final result was the creation of a library of functions for the creation of digital games with virtual worlds. Such library is able to conduct the machine in the generation, through random numbers, of unpredictable worlds and weather situations similar to the real world.

The given library is still capable to allow the interaction between user and machine through many interfaces, making the communication between computer and user more human-like. It enables many users to interact each other through a local network.

Finally, this library has algorithms which manages the intelligent agents behavior, for interaction between themselves and many users of the system.

Key-Words: Digital Games, Random Generated Environments, Intelligent Agents, Networked Virtual Environments (NVE's), Interaction.

1- INTRODUÇÃO

1.1 Origem do Trabalho

Quando se olham jogos de ação em computadores, nota-se na maioria dos casos uma deficiência: toda vez que o usuário inicia um jogo, no mesmo nível de dificuldade, o jogo está sempre com seus mesmos personagens, mesmos cenários, portas ocultas no mesmo lugar, armadilhas no mesmo lugar. Todo um ambiente estático é feito, sem a possibilidade de criar uma nova situação para o jogador ou jogadores.

Como conseqüência, os usuários decoram as principais posições do jogo, onde existem os principais objetos e portas secretas, e assim, sua execução se torna obsoleta e sem graça.

O estudo de tecnologias disponíveis de *Hardware* e *Software*, Inteligência Artificial, Realidade Virtual e Redes de Computadores deram origem ao presente trabalho.

Estes estudos serviram para a implementação de uma biblioteca, capaz de gerar ambientes tridimensionais pseudo-randômicos.

Tal biblioteca também serve para implementar elementos inteligentes que pudessem manter uma comunicação e interação com o usuário, e gerar desafios diferentes a cada jogo iniciado.

Junto a estas características, adicionou-se à biblioteca a capacidade de ser programada para permitir a comunicação entre vários computadores, para que vários usuários pudessem interagir de forma cooperativa ou concorrente.

Por último, uma pesquisa de campo foi feita em dois fliperamas da cidade de Florianópolis, com 80 (oitenta) indivíduos, revelando que mais de um terço destes tinham preferência por jogos de corrida de veículos. Esta observação serviu de estimulo para a criação de um protótipo de um jogo de corrida de veículos que funcionasse em redes de computadores.

1.2 Importância / Justificativa

Jogos e diversão fazem parte do crescimento humano, (PIAGET, 1988):

> "... a criança que joga desenvolve a percepção, inteligência, sua tendência à experimentação e seus instintos sociais. Os jogos computadorizados apenas substituem os jogos coletivos ou individuais de antigamente, a grande desvantagem é que normalmente se joga sozinho, mesmo que se tenha adversários on-line, incentivando o individualismo e a solidão."

Para (ANTUNES, 1998), os jogos servem de iniciação à vida, e para o acúmulo natural de novas experiências. Estas experiências são procuradas naturalmente devido ao processo de crescimento do ser humano:

> "... alguns dos mais destacados pensadores de nosso tempo demonstraram vivo interesse pela questão lúdica e pelo lugar dos jogos e das metáforas no fenômeno humano e na concepção de mundo: hoje a maioria dos filósofos, sociólogos, etólogos e antropólogos concordam em compreender o jogo como uma atividade que contém em si mesma o objetivo de decifrar os enigmas da vida e de construir um momento de entusiasmo e alegria na aridez da caminhada humana. Assim, brincar significa extrair da vida nenhuma outra finalidade que não seja ela mesma. Em síntese, o jogo é o melhor caminho de iniciação ao prazer estético, à descoberta da individualidade e à meditação individual.
>
> O que leva uma criança a brincar?
>
> Toda criança vive agitada e em intenso processo de desenvolvimento corporal e mental. Nesse desenvolvimento se expressa a própria natureza da evolução e esta exige a cada

instante uma nova função e a exploração de nova habilidade. Essas funções e essas novas habilidades, ao entrarem em ação, impelem a criança a buscar um tipo de atividade que lhe permita manifestar-se de forma mais completa. A imprescindível "linguagem" dessa atividade é o brincar, é o jogar. Portanto, a brincadeira infantil está muito mais relacionada a estímulos internos que a contingências exteriores. A criança não é atraída por algum jogo por forças externas inerentes ao jogo e sim por uma força interna, pela chama acesa de sua evolução. É por essa chama que busca no meio exterior os jogos que lhe permitem satisfazer a necessidade imperiosa posta por seu crescimento".

Do ponto de vista acadêmico-científico, o estudo de técnicas em jogos computadorizados fornece fontes de pesquisa em áreas diversas, tais como Realidade Virtual, Inteligência Artificial e Redes de Computadores.

O estudo de técnicas de programação de algoritmos utilizando Realidade Virtual tem grande destaque no desenvolvimento de jogos virtuais e simuladores.

(HUONG *et al.*, 1998) afirma que, em mundos virtuais, quanto maior o nível de detalhe, maior o senso de presença:

"Em anos recentes, pesquisadores têm investigado primeiramente os parâmetros de visualização sobre a sensação de presença do indivíduo em mundos virtuais. Alguns destes parâmetros são: o nível de detalhe visual (textura e número de polígonos), campo de visão, visão estereoscópica *versus* biótica, uso de sensores de rotação na cabeça ou não, e taxa de amostragem no vídeo (*framerate*). Em geral, os pesquisadores descobriram que quanto maior o nível de realismo visual, maior a sensação de presença (BARFIELD, 1995 e HENDRIX, 1996)"

E coloca à tona o dilema detalhe visual *versus* sensação de presença (HUONG *et al.* 1998):

"Quanto mais os projetistas aumentam o nível de detalhe visual, mais lenta se torna a resposta do sistema, reduzindo a sensação de presença do usuário no mundo virtual."

Ainda em (HUONG *et al.*, 1998), faz-se menção de vários trabalhos feitos sobre o estímulo das sensações através da audição, olfato, paladar e tato, com a intenção de aumentar o realismo em mundos virtuais, e também sobre a necessidade destes trabalhos para aumentar o realismo em algumas aplicações ou simulações em Realidade Virtual.

Dado que existe uma necessidade de estímulo de várias sensações para aumentar o realismo de mundos virtuais, conclui-se daí que há a necessidade pela programação de algoritmos que permitam à máquina fazer o processamento dos dados provenientes de sensores do corpo humano, e transmitir as informações de volta a esse mesmo corpo, estimulando suas sensações.

LUBAN (2001), afirma em seu artigo que a indústria de jogos está muito próxima de recriar a experiência de assistir a um jogo como se assiste a um filme, e próxima de oferecer alguns recursos para produção de jogos que sejam os mesmos usados na produção de filmes.

LUBAN (2001) recomenda alguns ítens de projeto para uma boa apresentação, imersão e credibilidade sobre o ambiente no desenrolar do jogo.

Entre estes ítens destacam-se o som, o comportamento dos personagens, as aparências do jogo, a origem randômica e os movimentos independentes da natureza, e recursos simulando a vida dentro dela.

Quanto ao som, o autor aponta a função deste como sendo a de um poderoso descritor de situações, e sobre futuras pesquisas que se basearão em suas propriedades:

> "... Certamente, a edição (de jogos) envolve não somente a imagem como também a dimensão do som. Nunca subestime o verdadeiro poder descritivo do som. O som é um dos

aspectos dentro do projeto de jogos que irá basear os mais sérios desenvolvimentos nos próximos anos... "

"... Pense no som primeiro. Antes de vermos um local, nós o percebemos através do som. Em um jogo, o som é colocado – com freqüência – como um ingrediente adicionado, mesmo sendo aquele que dá mais "gosto" ao jogo. Quando um personagem atravessa uma estrada comprida, por exemplo, o jogador não pode escutar o mesmo som sempre. É a natureza aleatória do som que permite um universo mais vivo."

Quanto ao comportamento dos personagens, o autor leva em conta o realismo e a consistência de seu comportamento com o ambiente:

"A animação de personagens deve ser consistente com o ambiente. Quando um personagem é dirigido a um muro, seria inútil atravessar esse muro. Faria mais sentido se este personagem simplesmente parasse. Quando um obstáculo fica no caminho, o personagem deve entender que precisa pular por cima ou dar a volta... ...o personagem deve ter inteligência o suficiente para adaptar seus movimentos ao ambiente."

"... Personagens controlados pelo computador devem possuir os comportamentos mais realísticos possíveis. Mais do que se basear em técnicas falhas de Inteligência Artificial, o plano do jogo deve prover uma introdução própria e regras robustas de comportamento para estes personagens. Em ... os inimigos movem-se pelas redondezas, esconde-se em abrigos e enfrenta combates em estilo quase real. ... guardas são animados de um modo que lhes dá extraordinária presença: eles expandem suas formações, param para olhar em volta, e geralmente se comportam de maneira realista. Em ... personagens falam uns com os outros e mudam o timbre da voz quando percebem atividade suspeita. Todos estes detalhes de comportamento nos encorajam a acreditar nos personagens encontrados durante o jogo..."

Continuando o depoimento sobre os personagens, o autor refere-se ao fato de que estes servem de motivação ao jogador e têm função vital no desenrolar da história, executando funções que ajudam o jogador:

"Preste bastante atenção nos personagens secundários. Eles têm uma função crítica na história. Eles dão motivação ao herói, trazendo personalidade e vida ao mundo criado pelo autor e são freqüentemente o melhor caminho para introduzir novas pesquisas.

Em termos de jogabilidade, estes personagens se prestam para múltiplos usos: eles devem ajudar o jogador guiando-o através de um labirinto, ou lutando pelo seu time, ele pode estar temporariamente encarnado pelo jogador, ou deve se sacrificar pelo jogador, para que este esteja vivo e bem até cumprir seu objetivo."

Ainda em LUBAN (2001), recomenda-se o uso de detalhes nas aparências dos ambientes e concordância na montagem de cenários virtuais:

"As aparências devem ser detalhadas e devem oferecer crédito ao jogo. Em termos visuais, a aparência deve incluir todos os detalhes que fazem as coisas parecerem reais. ...existe a preocupação de criar todos os objetos que os usuários desejam ver, tais como ferramentas de jardinagem próximas ao jardim, roupas espalhadas pela casa, etc. ..."

Finalmente o autor dá seu depoimento quanto à origem randômica dos fenômenos e objetos da natureza:

"A natureza odeia uniformidades. Outro caminho usado para criar um universo mais "aceitável" é evitar a repetição e a uniformidade. Dê uma olhada através de sua janela e você verá que as construções possuem diferentes estilos. Isto é porque elas não foram construídas ao mesmo tempo. As pessoas na rua não se parecem umas com as outras. Elas se vestem de todas as maneiras e com as mais variadas roupas, com diferentes formas e tamanhos. Elas não andam no mesmo ritmo, esta é a vida real... ...fundos de ambientes decorados com particularidades. Nada dá mais personalidade a uma sala ou a uma construção do que um detalhe que você nunca viu antes. Pode ser tão simples quanto um cinzeiro com uma chepa de cigarro acesa dentro, uma peça de um equipamento que você espera encontrar dentro do tipo de sala que você visita, música vinda de um rádio, etc. ..."

"Coisas simples podem ser animadas para dar uma satisfatória sensação de vida ao ambiente. Pense nos poderosos efeitos de cortinas que esvoaçam... ...ou de folhas secas

sendo sopradas pelo vento... . Em uma rua, carros passando , ventiladores circulando em um armazém, um bando de pássaros voando no horizonte, etc."

(ÇAPIN *et al.*, 1999), falando sobre as possibilidades da telepresença, comenta sobre a comunicação via redes de computadores como uma tecnologia chave na troca de experiências por pessoas, incluindo aí os jogos de computadores:

"Telepresença é o futuro de sistemas multimídia: permitirá seus participantes de compartilhar experiências profissionais e privadas, encontros, **jogos** e festas. Ambientes Virtuais Compartilhados (AVC's) (em inglês Networked Virtual Environments – NVE's) são a tecnologia-chave para implementar esta telepresença. AVC's são sistemas que permitem muitos usuários, distantes geograficamente, interagirem em um ambiente virtual comum".

ÇAPIN ainda comenta sobre a necessidade de criação de personagens que representem os participantes deste mundo virtual, os avatares, sejam eles autônomos ou dirigidos por seres humanos:

"Particularmente, um dos mais importantes desafios sobre pesquisa em AVC's é a representação do usuário através de avatares... no AVC. Esta representação pode variar de um simples cubo com vida até humanóides altamente realistas, com articulações de corpo e face. Nós acreditamos que avatares tridimensionais em tempo real serão essenciais no futuro, porque quanto melhor a representação do personagem, maior o senso de presença no ambiente, e maior a sua habilidade de comunicação com outros usuários. Nós precisamos de humanóides virtuais autônomos para povoar estes mundos virtuais".

Observando os fatos relatados por (HUONG *et al.*, 1998), nota-se a necessidade de estudos na área de interação homem-máquina através de vários meios, tais como microfone, luvas digitais, *HMD*, *mouse*, placas aceleradoras gráficas, placas de som, óculos tridimensionais, dispositivos simuladores de força, torque e aceleração, e dispositivos de retorno de sensações táteis.

Também se justifica a necessidade de iniciar pesquisas na área de efeitos visuais e de texturas e métodos de amostragem de imagens em tempo real (*framerate*).

Das palavras de LUBAN, que afirmam que os personagens devem ser realistas e com comportamento consistente com o ambiente, percebe-se a necessidade de métodos de atuação de Agentes Inteligentes, Vida Artificial, métodos de previsão de colisão em tempo real, movimentos articulados de humanóides, que por sua vez podem ser extendidos para outros tipos de movimentos: amebóide ou articulada com mais extremidades.

Tais linhas de pesquisa têm importância para que o comportamento dos personagens se assemelhe ao comportamento natural dos seres vivos que encontramos na natureza.

Assim o jogo torna-se mais realista, fazendo com que os usuários passem a acreditar nos personagens que vêem. O mesmo é afirmado por ÇAPIN, que defende o fato de que, quanto mais realista o comportamento de um avatar, maior a crença do usuário na existência desse avatar, e maior a imersão dentro do mundo virtual.

Como LUBAN também afirma que os personagens têm função vital no jogo, funcionando como motivadores, sente-se a necessidade de criação de avatares que ajudam na redução da quantidade de trabalho executada pelo ser humano, dando-lhe a liberdade de se dedicar inteiramente ao que lhe interessa durante um jogo.

O autor ainda faz notar o fato de a natureza possuir seus fenômenos e objetos randômicos, justificando aí a necessidade de linhas de pesquisa para criar

métodos de geração pseudo-randômica – pela máquina – de objetos de formatos variados, tanto dinâmicos quanto estáticos, tais que sejam tão imprevisíveis quanto a própria natureza.

Há também a necessidade de métodos para comunicação via redes de computadores e criação de protocolos de comunicação, para que as experiências ganhas no decorrer de um jogo possam ser compartilhadas por vários participantes (ÇAPIN).

Baseada nestas justificativas, uma pesquisa de técnicas em programação de algoritmos em Realidade Virtual, Inteligência Artificial, Vida Artificial e Protocolos de Redes de Computadores foi desenvolvida, tal que possibilite, em modo de protótipo, a geração de um mundo virtual novo a cada vez que o jogador inicia o jogo, permitindo a participação de outros usuários através de uma rede de computadores.

1.3 Objetivos Geral e Específico do Trabalho

O objetivo geral do trabalho é criar, através do uso de geradores pseudo-randômicos de objetos virtuais, situações pseudo-aleatórias para estágios de um jogo qualquer, tais que o ambiente virtual criado seja de certa forma imprevisível, e que este possua as mesmas leis físicas que a natureza.

Estes geradores pseudo-randômicos serão implementados através de uma biblioteca de funções e procedimentos.

Além da criação do ambiente pseudo-aleatório, a biblioteca deve ser capaz de possibilitar ao profissional programar e gerenciar o comportamento de agentes inteligentes dentro destes novos mundos.

São objetivos específicos:

- Proporcionar níveis satisfatórios de interação homem-máquina através de *hardware* específico;
- Estudar uso de técnicas de IA, como Reconhecimento de Padrões, Mineração de Dados, Sistemas Baseados em Regras, Agentes Inteligentes e Vida Artificial;
- Implementar ferramentas de comunicação via redes de computadores;
- Analisar e implementar algoritmos para geração pseudo-randômica de cenários (terrenos, mapas, personagens, etc.);
- Aumentar os níveis de imersão no mundo virtual gerado utilizando-se de *hardware* de RV;
- Estudar técnicas de detecção rápida de colisão;
- Implementar um protótipo de jogo.

1.4 Modelo Teórico do Trabalho

O trabalho proposto foi dividido em 4 fases:

- Pesquisa bibliográfica;
- Pesquisa sobre funcionamento de *software;*
- Pesquisa sobre funcionamento de *hardware;*
- Criação da biblioteca e implementação do protótipo;

1.5 Estrutura

Este trabalho está dividido em dez capítulos. O primeiro capítulo dá uma introdução ao trabalho, o que foi feito, mostra os motivos de onde se originou esta pesquisa e seus objetivos gerais e específicos.

No segundo capítulo o termo JOGO é definido, assim como seu histórico, focando os jogos de *videogame*, jogos de computadores, e uma referência a algumas tecnologias para produzi-los.

No terceiro capítulo são citadas e explicadas resumidamente as tecnologias que foram estudadas para o desenvolvimento do protótipo.

No quarto capítulo são apresentados o modelo conceituai do jogo e suas justificativas, são apresentadas as técnicas aplicadas no protótipo, incluindo suas limitações e relação entre elas.

No quinto capítulo são apresentados os resultados obtidos do jogo, e uma tabela comparativa indicando qual das tecnologias estudadas no terceiro capítulo foram empregadas e/ou refutadas no protótipo.

No sexto capítulo uma série de conclusões tiradas a respeito do modelo e sua implementação são mostradas.

No sétimo capítulo são dadas sugestões de melhoras e alterações no protótipo implementado.

No capítulo oito é descrita a utilidade das tecnologias pesquisadas para áreas do conhecimento tais como farmácia, bioquímica, medicina, robótica, administração, educação, entretenimento, computação, física, entre outras.

No capítulo nove são mostradas as utilidades deste trabalho.

No décimo capítulo são descritas possíveis áreas de pesquisa para futuros trabalhos.

Junto ao trabalho seguem-se quatro anexos: O Anexo I descreve a pesquisa de campo que forma uma das bases para este trabalho. O Anexo II mostra um algoritmo de *pathfinder* para encontrar uma trajetória entre dois pontos num labirinto composto de quadriláteros. O Anexo III mostra exemplos de texturas otimizadas para jogos. O Anexo IV contém um glossário dos termos técnicos utilizados nesta dissertação.

2 – JOGOS

2.1 Definições

Segundo (ANTUNES, 1998) 'jogo' deriva do substantivo masculino latino *jocu*, que significa gracejo. Incluindo o uso das regras no ato de jogar, comparando o jogo à metáfora da vida, e cuidadosamente restringindo sua abrangência como um estímulo ao crescimento e desenvolvimento cognitivo:

> "Em seu sentido etimológico, portanto, expressa um divertimento, brincadeira, passatempo sujeito a regras que devem ser observadas quando se joga. Significa também balanço, oscilação, astúcia, ardil, manobra. Não parece ser difícil concluir que todo jogo verdadeiro é uma metáfora da vida".

Em (HOUAISS et. al., 2001) a palavra JOGO é composta de um elemento compositivo *jog*, do verbo latim *joco*, entre outros verbos:

> JOG – el. comp. antepositivo, do v. lat. *joco, as, āvi, ātum, āre*, já em Plauto (254-184 a.C.), contra o verbo lat. clássico *jocor, āris, ātus, sum, āri* (depoente) 'gracejar, apodar, mofar, zombar, simular brinquedos, brincar' – com representação em român.: romn. *juca*, it.. *giocare*, friul. *dzuyá*, fr. *jouer*, provç. cat. português *jogar*, esp. *jugar* – a que se acresceram noções conexas com 'azar, sorte, o aleatório' e noções conexas com 'desempenho lúdico'; a v. em causa, aliás desbanca, no vulg., o v. *ludo, is, si, sum, ēre* 'jogar, dar-se a um exercício, recrear(-se), folgar, morar, zombar, compor, tanger, etc.', ver LUD(I)-; o v. *jocor* é derivado de *jocus, i* 'gracejo; graça; divertimento; brincadeira; galhofa' (romn. provç. *joc*, it. *giuoco*, friul. *dzug*, fr. *jeu*, cat. *joch*, esp. *juego*, port. *jogo*), de que derivam tb. lat. *jocōsus, a, um*, 'que gosta de gracejar, folgazão, alegre', *jocundus, a, um* 'agradável, deleitoso' e *joculāris, e* 'divertido, faceto, risível' (it. *giullare*, prov. cat. *joglar*, esp. *juglar*, port. *jogral*);

> LUD(I) – el. comp. antepositivo, do lat. *ludus, ī* 'jogo, divertimento, recreação', der. e comp. latinos: *ludibrĭum, ĭi* 'joguete, zombaria; insulto, ultraje; ', *ludĭus, ĭi* 'pantomimo, comediante', *ludicer* e *ludicrus, a, um* 'de jogo, de divertimento', *ludĭcrum, ĭ* 'divertimento, recreio, folga', *allūdo* ou *adlūdo, is, si, ūsum, ĕre* 'brincar, divertir-se, gracejar', *alludĭo* ou *adludĭo, as, āvi, ātum, āre* 'brincar com, fazer festa a, gracejar', *allusĭo, ōnis* 'ação de brincar com, brinquedo, afago', *colludo, is* 'jogar com, fazer conluio', *collusĭo, ōnis* 'conluio, fraude'...

2.2 Histórico dos Jogos

2.2.1 Origem de Alguns Jogos

Jogos são meios de entretenimento muito antigos. Inclusive há referências de que nem sempre os jogos antigos foram usados somente para esse fim. Em (MACEDO et. al., 2000) Faz-se uma referência ao jogo chamado *MANCALA*, que era associado a rituais mágicos e/ou sagrados, que surgiu a cerca de 7000 (sete mil anos) provavelmente na região onde hoje é o Egito.

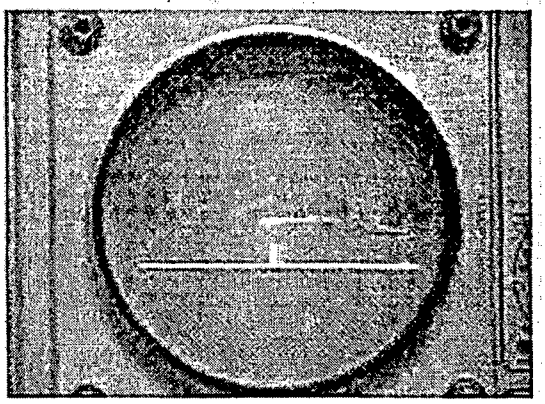
Em (FARUM-BADLEY, 2000) faz-se referência ao jogo WARRI, no atual país de Barbados, que se originou de um modelo de *mancala* que existia no Alto Nilo (atual Sudão) a 3600 (três mil e seiscentos) anos atrás.

Também em (MACEDO et. al., 2000), um jogo semelhante ao RESTA UM já era mencionado pelo poeta grego Ovídio (43 a.C. – 16 d.C.).

Em (IBMHF, 2002) é mencionada uma referência de que o jogo de boliche se originou provavelmente do Antigo Egito, baseado na descoberta de uma tumba egípcia de uma criança, datada de 3200 a.C. com um brinquedo com peças semelhantes às peças do boliche atual.

Em (GAMESPOT, 2002) fala-se sobre a criação da empresa **Marukufu Company**, em 1889, por Fusajiro Yamauchi, para manufaturar e distribuir cartas de um jogo japonês chamado Hanafuda. Em 1907 a empresa começa a produzir cartas de jogo comuns. A empresa muda seu nome para **Empresa de Cartas de Jogos Nintendo**, em 1951. Nintendo significa "deixar a sorte para o paraíso".

Fig. 1: Tela do osciloscópio do jogo criado por Willy Higinbothan como o precursor do *Pong*.



Em (THE DOT EATERS, 2002), é mencionado a criação do primeiro protótipo de *videogame* por Willy Higinbotham em 1958. Sua tentativa de tornar a visita ao seu laboratório (**Brookhaven National Laboratory**) um pouco mais interessante se tornou um precursor do *videogame* atual. Com a ajuda de um osciloscópio com uma tela de diâmetro de cinco polegadas (Fig. 1), e duas caixas com diais para controlar as raquetes (pequenos traços, Fig. 2), criou uma versão rudimentar de um jogo de *ping-pong*.

Fig. 2: Controle criado por Willy Higinbothan para o precursor do *Pong*.

2.2.2 *Arcades*

Antes de descrever resumidamente a história dos videogames a partir de algumas fontes, define-se a palavra *fliperama* como uma composição de duas palavras inglesas: *flipper* + *panorama* (HOUAISS, 2001):

"1 – jogo que consiste em fazer pontos cada vez que uma bilha é acionada por mecanismos elétricos no interior de uma prancha inclinada. 2 – casa comercial de recreação que oferece jogos elétricos e eletrônicos operados por ficha ou moeda. ETIM do inglês *flipper* (1822) 'o que se move com movimentos curtos e rápidos' p. ext. 'jogo eletrônico acionado por movimentos bruscos dos

dedos, principalmente os polegares' + *ama* (contração de *panorama*), (proveniente do modelo de *cinerama*)"

Em (THE DOT EATERS, 2002) afirma-se que o primeiro *arcade* (no Brasil também chamado de *fliperama*), foi criado por Nolan Bushnell, um empregado da empresa **Ampex**, situada em Sunnyvale, Califórnia.

Em 1971 Bushnell sai da **Ampex** e cria o primeiro jogo para máquinas operadas por fichas, chamado COMPUTER SPACE, e convence a empresa **Nutting Associates** a produzir o *arcade*.

1500 unidades deste *fliperama* foram produzidas, mas suas vendas não foram tão boas quanto o esperado, fazendo Bushnell concluir que os controles usados no jogo deveriam ser mais simples e em poucas peças, para permitir ao usuário jogar mesmo sem um conhecimento inicial dos controles.

Em 1972, Bushnell funda a **Atari**, que significa "pôr em cheque" num jogo (traduzido do inglês: "**Atari** is the equivalent of "check" in the game"), e cria seu primeiro jogo para *arcades*, chamado PONG, uma representação simplificada do jogo de tênis, junto com outro colega de trabalho na antiga empresa **Ampex**: Al Alcorn.

No mesmo ano Bushnell e Alcorn começam a vender suas máquinas de fliperama, vendendo mais de 8500 unidades em um ano.

Em 1973, outras empresas começam a vender suas versões *arcades* do mesmo jogo. Estas empresas são: **Taito**, **Sega**, **Allied**, **PMC**, **Chicago Coin**, **Williams** e **Midway**. Inclusive a própria Atari fazia clones diferenciados de seu jogo PONG.

Neste mesmo ano a **Atari** é a líder no mercado de fliperamas, alavancando US$ 3,2 milhões, mas uma competição acirrada pelo mercado de *fliperamas* inicia com a empresa **Kee Games**, de Joe Keenan.

Muitos empregados evadiram da Atari para a Kee Games, e em 1974 a Kee Games cria o jogo TANK, cuja novidade no campo de *arcades* era a inclusão de *chips* de memória somente de leitura (Read-Only-Memory), para sustentar a memória de gráficos.

Em 1974, dois empregados da Atari, Bob Brown e Harold Lee, ficaram encarregados de criar uma versão doméstica do jogo PONG, capaz de se acoplar a qualquer circuito de televisão. O objetivo da empresa era vender 150000 (cento e cinqüenta mil) unidades naquele ano. Cada unidade custaria US$100.

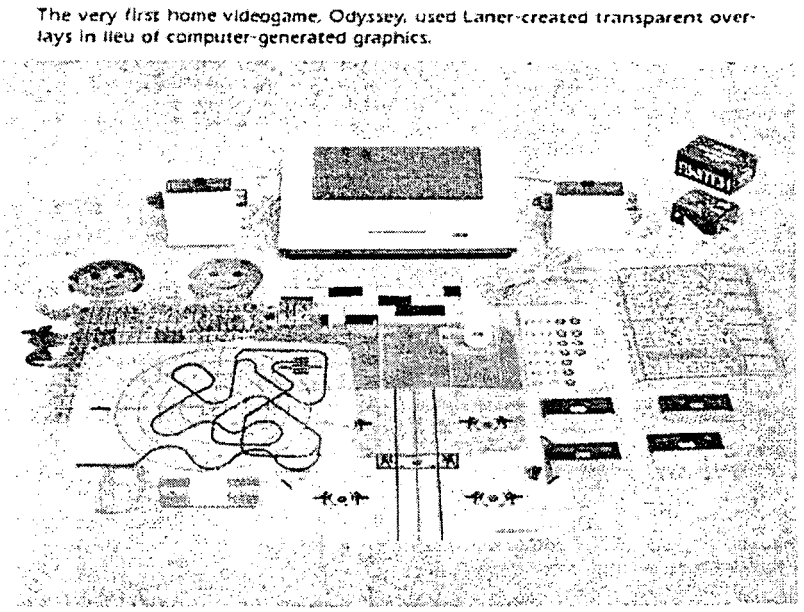
A partir de 1975 a **Atari** faria 40 milhões de dólares por ano, chamando a atenção do conglomerado **Warner Communications**, e em 1976, Nolan Bushnell venderia a **Atari** para a **Warner** por US$ 28 milhões.

A partir desta inclusão de capital, a **Atari** se tornaria uma marca tão famosa quanto a **Coca-Cola** ou a **Kleenex**.

2.2.3 *Videogames*

Segundo (HART, maio de 2001) os *videogames* iniciaram em 1966 com a tentativa de Ralph Baer de criar um dispositivo de treinamento militar que melhorasse os reflexos dos soldados e estratégias. Deveria ser um dispositivo portável.

Fig. 3: O primeiro *video-game*, *Odissey*, da empresa *Magnavox*, com máscaras para a tela da televisão. (Impresso com permissão de [HART – 41]).



Entre 1972 e 1977, considerada a era dos *videogames* de primeira geração (exemplo na Fig. 3), houveram várias empresas produzindo *videogames*: a ***Magnavox*** de Ralph Baer, com o *seu videogame* Odissey, contendo vinte jogos, a **Atari**, com o jogo PONG e suas variações (tênis, hóquei, entre outros). A ***Fairchild Camera and Instrument***, de Robert Noyce, e a empresa ***Baily***. Na época, poucos usuários estavam dispostos a pagar mais caro pela diversidade de jogos oferecidos pelo ***Odissey*** ou ***Fairchild***, onde muitos alegavam que somente um ou dois jogos agradavam. Assim, o mercado foi dominado pela ***Atari*** até 1977.

A Segunda geração de *videogames* foi de 1977 a 1981, com os produtos caracterizados por efeitos visuais e auditivos mais convincentes, sendo seu mercado dominado pelo modelo VCS/2600 da empresa ***Atari***.

Entre 1982 e 1984 perdurou a terceira geração de *videogames*, auxiliada pelo uso de disquetes com grande capacidade de memória(180 *kilobytes* contra os 16 *kilobytes* de cartuchos ROM), mas sofrendo muitas perdas devido à concorrência com os fliperamas de rua, que possuíam de dez a vinte e cinco vezes mais memória do que os *videogames* caseiros, permitindo jogos mais realistas.

A Quarta geração de *videogames* inicia em 1985 com a redução dos custos de fabricação de *chips* de Memória Dinâmica de Acesso Aleatório e o início da produção de processadores de oito bits, baixando o preço dos processadores antigos. A partir de 1985 os *videogames* da empresa **Nintendo** passaram a dominar o mercado. Em 1989 as revendedoras de seus produtos não estavam mais de acordo com a política de vendas da empresa, terminando neste ano a quarta geração de *videogames*.

A partir de 1989, o único concorrente para a empresa **Nintendo**, que oferecia *videogames* de 8 *bits*, era a **Sega (Service Games)**, com o seu *videogame* Sega Master System, sendo o primeiro *videogame* a conter óculos 3D, enquanto a **Nintendo** oferecia em seus *videogames* as pistolas eletrônicas.

2.2.4 Jogos de Computadores

De acordo com (THE DOT EATERS, 2002), em 1972 foi criado o jogo Hunt The Wumpus (Caça ao Wumpus), por Gregory Yob, na Universidade de Massachussets, em Dartmouth. Sendo um jogo que usava somente texto, onde o usuário se move através de um sistema de cavernas conectadas, armado de somente cinco flechas, procurando pelo monstro de nome Wumpus, que também

está se movendo através das cavernas. O objetivo do jogo é aniquilar o monstro, com uma das flechas.

Neste mesmo ano Willie Crowther cria ADVENTURE, um *adventure* baseado no jogo de dados e de papel DUNGEONS AND DRAGONS. O objetivo do jogo é passar por uma caverna cheia de labirintos e voltar ao ponto de partida com o maior número de tesouros possível. O processo de diálogo do usuário com o jogo é estruturado na forma "verbo-nome".

Em 1977, Dave Lebling, Tim Anderson e Bruce Daniels terminam um protótipo do jogo ZORK, criado num computador PDP 10 do Massachussets Institute of Technology, no mesmo estilo de ADVENTURE, e posteriormente se torna uma grande sensação para os estudantes, sendo disseminado através da ARPAnet (antes de se tornar a Internet em 1984).

Até 1981, ao jogo ZORK foram se acrescentando dados até que o jogo alcançasse o tamanho de 1 megabyte. Neste momento, o jogo já estava sob lei contra cópias pertencendo à empresa recém-criada **Infocom**.

Entre 1979 e 1981, surgiam os computadores pessoais modelos **Tandy TRS-80** e **Apple**. Em 1979 a **Infocom** faz seu primeiro *royalty* sobre o jogo ZORK I para o modelo TRS-80, e em 1980, Bruce Daniels cria o mesmo jogo, para o modelo Apple II, vendendo 6000 cópias em oito meses.

Contando todas as vendas para as plataformas de computadores pessoais ao redor do mundo, ZORK I vendeu mais de um milhão de cópias.

Em 1979, o primeiro jogo em redes de computadores foi criado por Roy Trubshaw, um pesquisador da universidade de Essex – Inglaterra, para o modelo

de computadores DEC-10. Seu jogo se consistia num MUD ("Multi-User Dangeon" ou Calabouço Multi-Usuário), ou seja, um mundo multi-usuário baseado em comunicação de texto em que os usuários podem conectar e interagir com uma base de dados de ítens e usar um tipo rudimentar de conversa através da rede. Neste jogo estavam disponíveis vinte salas, e as atividades do jogo eram controladas com um vocabulário com dez comandos.

Em 1980, Ken e Roberta Willians criaram MISTERY HOUSE (Casa do Mistério) através de sua empresa doméstica **On-Line Systems**, o primeiro jogo para computadores pessoais combinando letras e gráficos. Apesar de seus gráficos grosseiros (Fig. 4), somente feito com linhas, o jogo chegou a vender 11000 (onze mil) cópias no primeiro ano, a US$24,95 a unidade.

Fig. 4: Primeiro jogo para computadores pessoais (1980) combinando gráficos e palavras.

Posteriormente o casal muda o nome de sua empresa para **Sierra On-Line**, e em 1983 a empresa é encarregada pela IBM para fazer um jogo mostrando as capacidades gráficas do seu novo modelo de computador PCJr.

Nasce assim o *adventure* KING'S QUEST, o primeiro jogo feito para computadores pessoais mostrando gráficos pseudo-tridimensionais, com dezesseis cores em vídeo CGA (Color Graphics Adapter).

Com um time de seis programadores e a um custo de desenvolvimento de US$700000 (setecentos mil dólares), KING'S QUEST foi lançado em 1984. Se

contados todos os sistemas mais populares de computadores pessoais para os quais este jogo foi produzido, KING'S QUEST vendeu mais de 2,7 milhões de cópias, rendendo ainda oito seqüências.

Em 1986, Scott Murphy, e Mark Crowe lançam o *adventure* SPACE QUEST, sob a propriedade da empresa **Sierra On-Line**. O jogo foi um sucesso com sete seqüências, mas SPACE QUEST 7 parece ter sido cancelado pela empresa em 1998.

A empresa **Sierra On-Line** manteve sua liderança de mercado em jogos tipo *adventure* com interface gráfica até 1987, quando desafiada pela **Lucasfilm Computer Games Division** (depois renomeada para **LucasArts**), com seu jogo MANIAC MANSION, do mesmo estilo.

Em 1979, Richard Garriot termina um CRPG ("Computer Role Playing Game" ou Jogo de Atuação em Teatro feito por Computador, ver as definições em ROLE MAKER, 2002) na linguagem AppleSoft Basic chamado AKALABETH.

No jogo o usuário navega por um mapa repleto de símbolos ASCII, completando as tarefas dadas pelo seu tutor eletrônico, e batalhando criaturas em gráficos pseudotridimensionais.

Sob a propriedade da California Pacific Software Company, o jogo era distribuído em disquetes 5 ¼, vendendo 30000 (trinta mil) cópias.

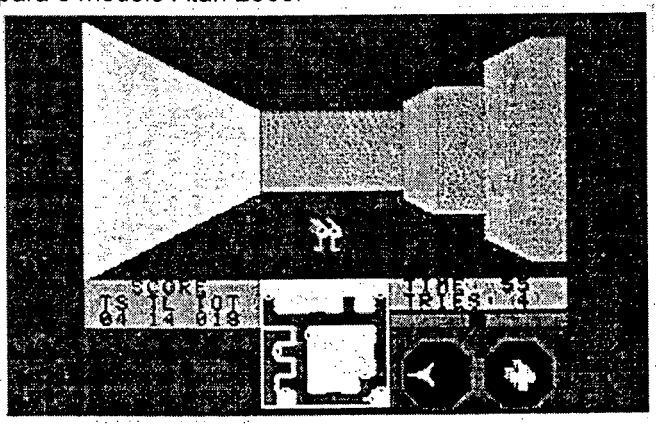
Em 1978, Jim Connelley e Jon Freeman terminam Starfleet Orion, programado em BASIC para o computador pessoal Commodore PET, fazendo o primeiro jogo de combate tático no espaço, feito para microcomputadores. Também criaram a primeira editora sobre jogos de computadores, a **Automated Simulations**.

Ainda em 1979, a dupla criou o primeiro jogo CRPG cujas características incluíam a possibilidade de o jogador poder criar as propriedades e personalidades de seus personagens. O jogo ganhou o prêmio de Jogo de Computador do Ano, da ***Academy of Adventure Gaming Arts & Design's Origin***, em 1980 e dele partiram mais duas seqüências, que foram programadas em outras plataformas de computadores pessoais.

Posteriormente em 1981, Jon Freeman e sua esposa Anne Westfall abandonam a **Automated Simulations** para formar a **Free Fall Associates**, Posteriormente mudando seu nome para **Epyx**. Em 1983 a empresa cria o jogo JUMPMAN, que contém três níveis de dificuldade com trinta labirintos cada nível. O jogo rendeu 40000 (quarenta mil) cópias.

Em 1982, a empresa **Starpath** (que tem Jim Connelley como um de seus integrantes) inventa o Supercharger, um expansor de memória para o computador pessoal modelo Atari 2600. Provavelmente o jogo mais famoso deste modelo de computador pessoal seja ESCAPE FROM THE MINDMASTER, um jogo tridimensional em primeira pessoa (Fig. 5). Em 1983 foi lançado o jogo de corridas PITSTOP, e em 1984 sua seqüência PITSTOP II, que seria o primeiro jogo contendo tela separada em duas partes, permitindo dois jogadores participarem do mesmo evento ao mesmo tempo, na mesma máquina.

Fig. 5: Tela do jogo *Escape From The MindMaster*, para o modelo Atari 2600.



Em 1984 a empresa **Epyx** lança SUMMER GAMES, um jogo de esportes olímpicos, feito em seis meses, totalmente em linguagem de máquina (também chamada código *assembly* ou do inglês *"assembly code"*) para o modelo de computador pessoal **Atari 2600**, vendendo 100000 (cem mil) cópias.

2.3 Sistema Atual de Produção e Consumo de Jogos para Computadores Pessoais

2.3.1 Uso da *Internet* para Consumo de Jogos

Navegando pela *Internet*, o usuário se depara com várias páginas que disponibilizam demonstração ou versões inteiras de jogos para computadores pessoais.

Tais servidores podem ser genéricos, com vários tipos de jogos, classificados de acordo com o tema e contendo *links* de fabricantes ou de títulos de jogos, ou ainda podem ser comunidades ou *sites* comerciais específicos para apenas um título, contendo todas as informações, estatísticas, dados relevantes, arquivos compartilhados, truques, correções de *software* (*patches*) e *links* para outros sites correlatos ao mesmo jogo.

Dentro de tais *sites* existem também as listas de discussão sobre dicas que podem ser usadas para correção de *software* ou para facilitar o alcance do objetivo do jogo – estas dicas também são chamadas de *cheats* ou *cheatcodes*.

2.3.2 Bibliotecas de Criação de Jogos

Bibliotecas de Criação de Jogos são ferramentas de *software* que facilitam a produção de um jogo ou os dados que o compõem. Podem ser somente com

recursos gráficos (*OpenGL* da Silicon Graphics© para placas aceleradoras de vídeo), recursos de modelagem física em tempo real (*HAVOK, 2002*) ou ainda bibliotecas genéricas, que envolvem desde a modelagem matemática gráfica até sons, música e periféricos (*DirectX* da Microsoft©).

Existem ainda bibliotecas específicas totalmente recriadas ou derivadas de bibliotecas básicas ou de bibliotecas genéricas (3D Engines List, 2000).

2.3.3 Emuladores para Jogos

Em (WHATIS?COM, 2002) define-se um emulador de computadores como sendo "um componente de *hardware* / programa, que é feito com a finalidade de ser um outro componente de *hardware* / programa..." , ou seja, um emulador é um programa que pode fazer o papel de um computador, um sistema operacional ou um software, dentro de um determinado computador.

A partir de um emulador é possível executar, por exemplo, em um computador pessoal tipo IBM©, um programa que inicialmente funcione somente para computadores pessoais da Apple Computers©.

Usando emuladores, é possível executar jogos de uma marca específica de *videogame* em um computador pessoal. A *Internet* é uma fonte ampla de páginas que fornecem emuladores experimentais (exemplo: EMUMANÍACOS, 2002), capazes de executar jogos de sistemas de *videogame* em computadores pessoais.

3 – CONSIDERAÇÕES E TECNOLOGIAS ABORDADAS NESTA DISSERTAÇÃO

3.1 Motivos

Desenvolver um jogo usando Realidade Virtual Compartilhada e Inteligência Artificial requer uma série de variáveis a serem determinadas, e um planejamento detalhado das necessidades e procedimentos a serem tomados para construí-lo.

Antes de chegar ao jogo inteiramente funcional, deve-se implementar um protótipo, para conferir seu correto funcionamento, verificar passo a passo seus algoritmos e interfaces necessários para que funcione, e sugerir possíveis mudanças para o produto final.

Como trabalho experimental, uma biblioteca de funções e um protótipo do jogo usando Realidade Virtual Compartilhada e Inteligência Artificial foi construído para testar algumas possibilidades de *software*, interface e *hardware*, e posteriormente implementar sua otimização.

Já de início este protótipo tem o objetivo de fazer com que o usuário não saiba a diferença entre o mundo virtual e o mundo real.

O protótipo do jogo deve possuir as seguintes características:

- O jogo se constitui numa corrida de veículos numa pista cercada;
- O objetivo do jogo é ser o primeiro a completar um determinado número de voltas numa pista aleatória;

- O mundo criado possui dois tipos de obstáculos: a cerca em volta da pista e os obstáculos isolados. Os obstáculos isolados são constituídos por árvores, pedras e pequenas construções;
- Cada usuário pode colidir com outros usuários, ou com os obstáculos do mundo;
- Para aumentar o realismo do jogo, um fenômeno climático de chuvas foi adicionado, com comportamento imprevisível;
- O jogo também deve possuir agentes inteligentes, representados por bandos de animais virtuais, ou agentes capazes de executar um diálogo com o usuário;
- Os agentes inteligentes devem evoluir segundo as leis da genética, e através de regras simples chegar a sistemas complexos de organização;
- Estes agentes devem usar técnicas de previsão de movimentos do usuário;
- Alguns recursos de Inteligência Artificial devem ser usados para permitir a cada máquina executar tarefas de previsão de trajetórias, encontrar caminhos entre dois pontos através de obstáculos (*pathfinder*), aprendizagem de comportamento humano e métodos de perseguição e fuga;
- Pode ser jogado por vários usuários, por intermédio de uma rede local de computadores;
- Os veículos de cada usuário devem ser capazes de reconhecer os comandos ou atividades do usuário através de algumas interfaces: teclado,

joysticks, *cyberpucks*, *HMD's* e luvas digitais, com modelos opcionais de navegação;

- Cada veículo deve ser capaz de entender um número mínimo de palavras, ditas pelo usuário através do microfone, possibilitando um diálogo;
- Possibilidade de visão em primeira e terceira pessoas;
- O jogo deve possuir um recurso de navegação entre ambientes, tal que não prejudique a imersão do usuário no mundo virtual;
- Recursos de visualização geral do ambiente serão usados, para que os usuários possam planejar suas ações isoladas ou em conjunto antes do início do jogo;

Devido a estas justificativas, as tecnologias estudadas foram as seguintes:

- Programação Orientada a Objeto, para um fácil gerenciamento dos módulos de programação criados;
- Realidade Virtual;
- Vida Artificial para simular comportamento de bandos de seres vivos;
- Algoritmos Genéticos como ferramenta de otimização e aprendizado de máquina;
- *PathFinding* (Procura por Trajetórias);
- *DataMining* (Mineração de Dados) para previsão de trajetórias feitas pelo usuário;
- Redes Neurais (especificamente as *Multi-Layer Perceptron* e *Learning Vector Quantization*);

- *VRML*;
- Transferência de dados via redes de computadores, para permitir a comunicação de vários usuários no mesmo ambiente (Realidade Virtual Compartilhada);
- Um Modelo de Colisão, usando mapas *raster*;
- Métodos de pré-processamento de elementos geométricos para geração de topologias aleatórias (Paredes e Terrenos) e de texturas;
- Reconhecimento de Palavras Faladas, incluindo um Modelo de Diálogo Homem-Máquina;
- Dois modelos físicos de navegação de ambientes;
- *Hardware* para interface homem-máquina, com seus modelos de funcionamento e programação.

3.2 Modelagem de Projetos Orientada a Objetos

Segundo (RUMBAUGH *et al*, 1997) a modelagem orientada a objetos se caracteriza por possuir unidades básicas chamadas de objetos, que combinam a estrutura e o comportamento dos dados em uma única entidade. A modelagem baseada em objetos inclui quatro aspectos: identidade, classificação, polimorfismo e herança.

Identidade significa que os dados são separados em unidades discretas e distintas, chamadas de objetos, conseqüentemente cada objeto possui suas próprias características individuais.

Classificação significa que objetos que possuem a mesma estrutura de dados e o mesmo comportamento são agrupados em uma classe, e cada classe descreve possivelmente um número infinito de objetos individuais.

Polimorfismo significa que uma mesma operação pode atuar de modos diversos em classes diferentes.

Herança é o compartilhamento de atributos e operações entre classes com base em um relacionamento hierárquico. Uma classe pode ser definida de forma abrangente e depois refinada em sucessivas subclasses. Cada subclasse incorpora ou herda as características de sua classe mãe e acrescenta suas próprias e exclusivas características.

Em (TAFNER, 1996) também são dadas explicações sobre os ítens de programação orientada a objetos.

3.3 Conceitos de Realidade Virtual: Imaginação, Interação e Imersão

Segundo (BURDEA, 1996-b):

> "Realidade Virtual pode ser definida também como "Ambientes Virtuais" (SHERIDAN, 1992 *apud* BURDEA, 1996-b), "Ciberespaço" (ELMER-DEWITT, 1993 *apud* BURDEA, 1996-b), "Ambientes Verídicos" (CODELLA, JALILI, KOVED & LEWIS, 1993 *apud* BURDEA, 1996-b), ou "Realidade Artificial" (KRUEGER, 1991 *apud* BURDEA, 1996-b), representando uma interface gráfica de alta qualidade que imerge o usuário em um mundo simulado. No caso da RV ou Ambientes Verídicos, o mundo simulado consiste em replicar a realidade física existente. Por outro lado, para Realidade Artificial e Ambientes Virtuais, a simulação pode partir da realidade conhecida, permitindo experiências que não podem ser reproduzidas no mundo real."

Do fato de que Realidade Virtual pode ser definida como Realidade Artificial e Ambientes Virtuais, e que estes dois últimos podem partir da realidade conhecida, permitindo experiências que não podem ser reproduzidas no mundo real, a diferença entre o mundo real e estas experiências só pode ser uma conseqüência da **imaginação**, da **criação** ou da **percepção** – dentro da realidade artificial ou ambiente virtual existente – de situações ou fatos que o ser humano não percebe e não pode criar no mundo real.

A interação consiste em explorar todos os meios sensoriais do ser humano, em tempo real para haver uma comunicação entre este e o mundo virtual no qual está inserido. Segundo (BURDEA, 1996-b):

> "A peça-chave para a experiência de usuários de Realidade Virtual é a interação multimodal em tempo real envolvendo todos os sentidos, desde a visão, audição, tato cheiro e até gosto. Esta rica interação sensorial (às vezes chamada de sobrecarga sensorial), acoplada com simulações de respostas em tempo real produz uma compelidora e cativante sensação de imersão".

A imersão, também segundo (BURDEA, 1996-a) é uma "sensação de estar rodeado por um mundo sintético com o qual ele pode interagir e modificar".

Segundo (SHERIDAN, 1992-b *apud* BURDEA, 1996-a) a imersão é uma "sensação ideal, na qual, com uma tecnologia boa o suficiente, uma pessoa não seria capaz de distinguir entre presença física (*actual presence*), telepresença e presença virtual".

A imersão ocorre depois da interação, quando, depois de o ser humano modificar o mundo virtual à sua volta, se torna convencido de que está no mundo virtual, e passa a confundi-lo com o mundo real.

3.4 Vida Artificial

Segundo (ESTEVAM, 1997) a VIDA ARTIFICIAL foi iniciada nos fins da década de setenta, se preocupando em reproduzir *in silico* o comportamento de seres vivos observado na natureza. Ela veio para complementar a teoria biológica sobre a vida, que afirma que um ser vivo é assim considerado por obedecer os seguintes critérios: é composto de células, é baseado na química do carbono, ocorrendo numa solução aquosa, e seus processos vitais são controlados por uma célula chamada ADN.

A teoria da vida artificial se baseia na teoria da informação, publicada por SHANNON e WEAVER (*apud* ESTEVAM, 1997), e a partir da derivação da segunda se afirma o seguinte:

- A vida é informação que governa a forma e função, e é passada adiante;
- Apresenta certas leis de auto-organização e
- A vida evolui.

Tem-se aí a teoria da Vida Artificial. Pode-se concluir daí que a vida artificial possui algumas propriedades:

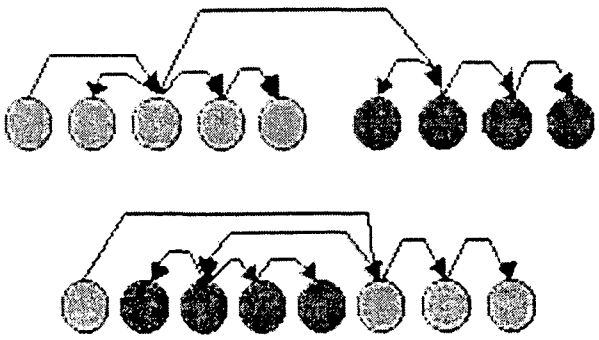
- Apesar de cada indivíduo dentro do sistema possuir uma regra simples para gerenciar seu comportamento, o comportamento global é muito organizado;
- A abordagem do programa não é *TOP-DOWN* (ESTEVAM, 1997), mas sim dependente do comportamento individual programado em cada elemento. Deixa-se então que o próprio sistema encontre suas soluções;
- Os indivíduos possuem comportamento emergente, ou seja, a partir de um sistema inicialmente desorganizado, chega-se a um sistema organizado.

Em (GRAND, 1997) existe uma breve explicação de um programa chamado *Creatures*, que utiliza Redes Neurais (FAUSSET, 1994), Algoritmos Genéticos (GOLDBERG, 1998) e Vida Artificial para simular a vida em colônia de pequenos personagens chamados *Norns* e *Grendels*.

Segundo (WOODCOCK, 1998) os comportamentos de cada espécie simulada em um jogo podem ser parametrizados, fazendo com que estas se comportem de forma diferente.

Em (SIMS, 1994) é descrita uma experiência de geração de criaturas pelo computador usando membros articulados em forma de paralelepípedos. Tais criaturas possuíam um código genético que permitia movimentos tais como nadar, andar, saltar e seguir a luz. A sobrevivência era garantida pela posse de um cubo no centro do ambiente. As criaturas que ficavam menos tempo possuindo o cubo eram eliminadas.

Fig. 6: Exemplo de *grafting* dos códigos genético do pai e da mãe para formar o código do filho.



Estas criaturas podiam capturar dados do meio onde navegavam, sendo que tais dados eram transformados através de funções simples, como função onda senoidal, interpolação linear, valor absoluto, onda de dente de serra, função memória, maior, menor, soma, entre outras, e serviriam para tomar decisões sobre como atuar no ambiente tridimensional para permanecerem vivas.

Elas possuíam seu código genético modificado por mutação, *crossover* (GOLDBERG, 1998) e *grafting* (SIMS, 1994). Esta última (Fig. 6) se

Fig. 7: Duas criaturas lutando por um mesmo objeto.

Fig. 8: Criatura nadando.

caracterizando por ser feita com dois genótipos: O pai é copiado, e uma das suas conexões é escolhida randomicamente e é ajustada para apontar para uma conexão qualquer no genótipo mãe. Os nós não conectados no genótipo pai são removidos, e os nós conectados no genótipo mãe e todos os seus descendentes são anexados para formar o filho.

Ao usuário / expectador era permitido eliminar criaturas que não tivessem uma aparência agradável. Cada criatura passava por um teste de colisão entre os seus membros articulados, e caso houvesse sobreposição dos membros, esta criatura era eliminada. As Figuras 7 e 8 mostram alguns tipos de movimentos feitos por estas criaturas, tais como nadar e competir entre si por um cubo.

Na página dos laboratórios do IC-CAVE (*International Centre for Computer Games and Virtual Entertainment*, 2001) faz-se uma nota sobre o fato da facilidade de detectar colisão através de membros com formato cilíndrico com extremidades esféricas (chamadas *sphyls*), se comparada à detecção de colisão de membros com formato paralelepipédico.

3.5 Algoritmos Genéticos

Segundo (GOLDBERG, 1998) Algoritmos Genéticos são algoritmos de procura baseados no mecanismo de seleção natural e de genética. Assim como na vida biológica, todos os seres vivos possuem suas informações (fenótipo) armazenadas em suas moléculas de ADN (genótipo), e estes tentam se adequar ao meio onde vivem.

Quanto mais adequado ao meio estiver o genótipo do ser vivo, mais chance de sobrevivência tem sua espécie. A vida biológica dentro do meio depende da capacidade do ser vivo de adequar-se às suas mudanças, fazendo-o através de:

- Multiplicação dos seus genes;
- Mutação dos seus genes;
- Cruzamento de informações dos seus genes ou *crossover.*

Fazendo um comparativo do meio biológico com programas de computadores, dentro de um dado programa, existem vários elementos que possuem informações sobre o seu comportamento ou forma (o fenótipo) armazenadas em cordões de caracteres (o genótipo). Dado um problema modelado (o meio onde vivem), são programados vários objetos que fazem a procura por uma solução deste problema (tentam sobreviver ao meio).

A sobrevivência de um conjunto de objetos é dada pelo resultado que cada elemento fornecer como solução do modelo: Quanto mais ajustado à solução do problema for o valor fornecido pelo elemento, mais chances a sua espécie terá para sobreviver. Tais elementos, assim como na vida biológica, têm capacidade de copiar seus cordões de caracteres (multiplicação dos genes), de alterar os

valores dos seus cordões de caracteres (mutação dos genes), e de cruzar os seus cordões de caracteres (*crossover*).

Em (WATSON, 1996) é mostrado um código de uso de algoritmos genéticos para atuação em jogos de tempo real, onde num dado mundo virtual, naves possuem suas estratégias de jogo modificadas com base nestes algoritmos.

3.6 *Pathfinder* (Procura por Trajetórias)

Procura por trajetórias (*pathfinder*) é uma técnica usada em Inteligência Artificial para fornecer um possível caminho entre dois pontos, dado o mapa bidimensional ou tridimensional do ambiente navegado. Pode ser executado por somente um agente ou um conjunto de agentes que têm um objetivo em comum (um exército que deve se locomover de um ponto a outro, por exemplo).

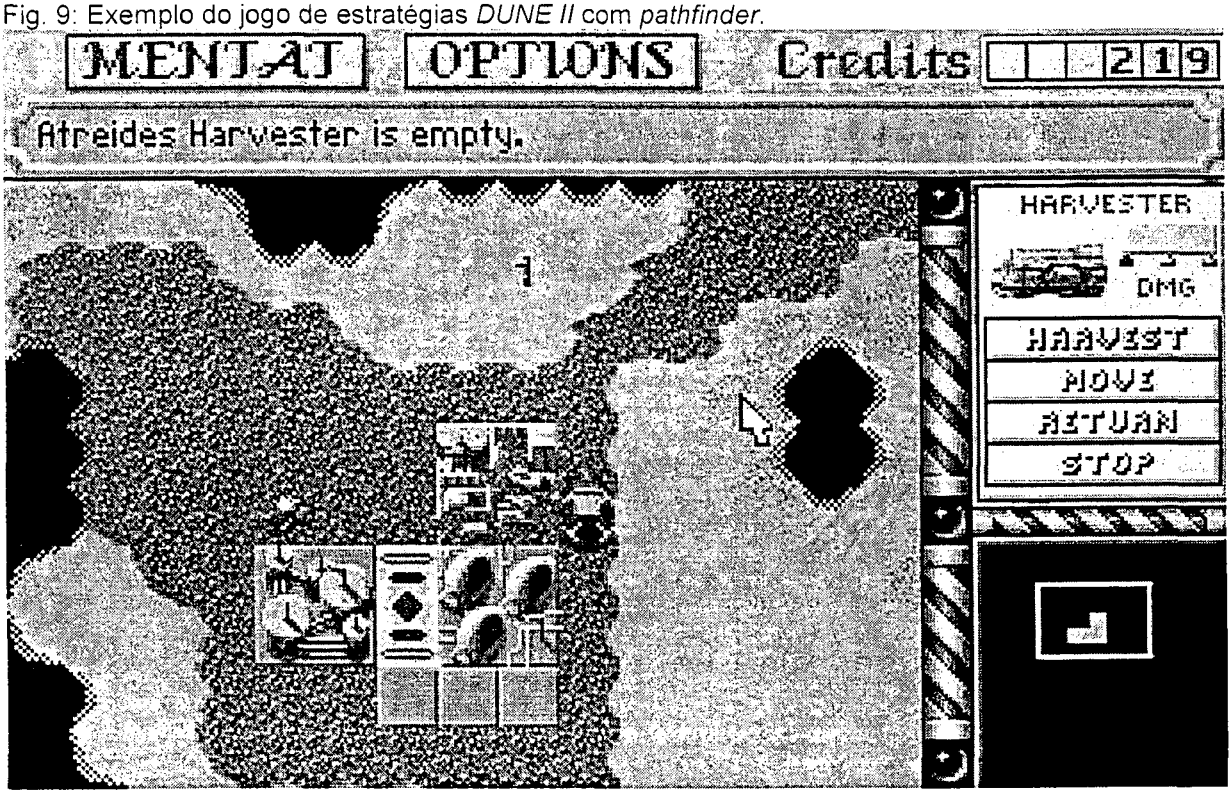
O caminho fornecido por esta heurística nem sempre é o melhor caminho, pois a resposta para o problema, no decorrer de um jogo de ação ou estratégia, deve ser fornecida rapidamente, mas o caminho fornecido como resposta já é uma possível solução para a locomoção do agente entre os dois pontos.

Em (PATEL, 2000) são descritas algumas características de programação sobre o algoritmo A* (A-Estrela) para encontrar um caminho entre dois pontos estáticos, que se baseia no custo de locomoção do agente através de um território determinado. Também são citados outros dois métodos de procura de caminhos: o algoritmo de *Dijkstra*, e o algoritmo da procura pelo primeiro melhor caminho encontrado (em inglês: *Best-First Search* ou *BFS*).

Em (BAERT, julho de 2000) são descritas quatro heurísticas usadas para procura de caminho entre dois pontos utilizando energia potencial de campo (em inglês: *Potential Fields*).

A técnica da energia potencial de campo se consiste em que, dados os pontos inicial e final da locomoção de um agente, este se deve deslocar do ponto em que está para um ponto já determinado de energia mais baixa. O ambiente por onde o

Fig. 9: Exemplo do jogo de estratégias *DUNE II* com *pathfinder*.



agente deve andar é mapeado de forma *raster*, de tal maneira que o ponto de energia mais alta é o ponto onde o agente se encontra, e o ponto de energia mais baixa é o ponto para onde o agente deve ir.

Os obstáculos são marcados como tendo energia potencial muito alta, garantindo que o agente nunca irá passar por cima de um obstáculo.

A Figura 9 mostra uma tela do jogo de estratégias *DUNE II* (*WestWood Studios*, 2001) onde o veículo indicado pela cruz vai até o ponto indicado pelo *mouse*. A técnica usada para fazer o movimento é de *pathfinder*.

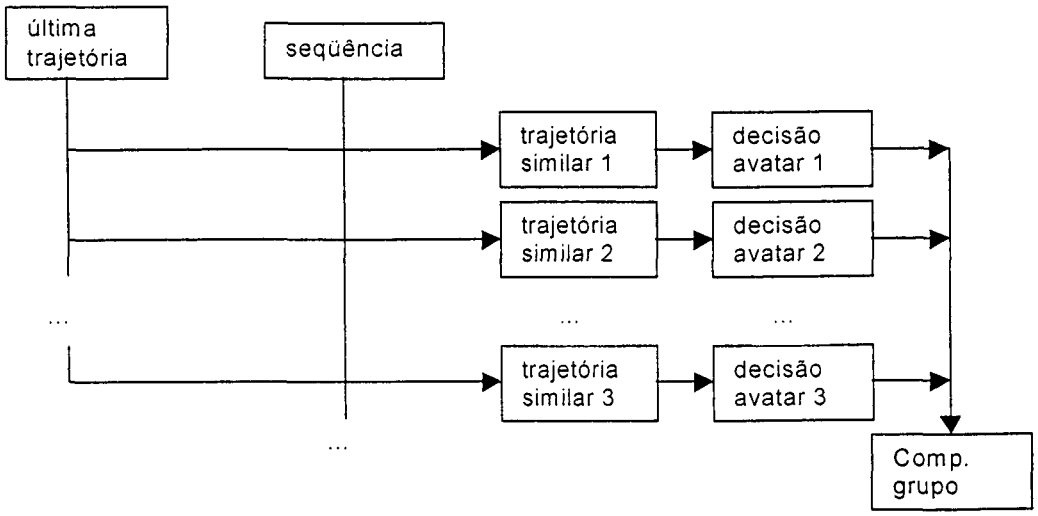
3.7 Previsão de Trajetórias com Mineração de Dados

A aplicação de técnicas de mineração de dados de Séries Temporais em detecção de trajetórias similares e sua previsão se justifica por três motivos:

- A função de encontrar trajetórias similares gera, para uma única entrada, várias respostas, ou seja, é uma função bijetora;
- Não existe uma distribuição estatística conhecida para o comportamento da trajetória de um usuário durante um jogo de ação, sendo necessária alguma técnica de mineração de dados.
- Os dados são dispostos em forma seqüencial.

Do ponto de vista da Vida Artificial, a partir de uma seqüência de dados, pode-se gerar várias respostas, onde uma resposta serve como entrada para a tomada

Fig. 10: Fluxo de informações e comportamento do grupo de agentes inteligentes para *pathfinder*



de decisão de um agente inteligente, pertencente a um mesmo time de agentes dentro de sua classe, e cada agente se comporta de acordo com aquela resposta. O comportamento isolado de cada elemento resulta no comportamento final do grupo (Fig. 10), que é a abordagem *down-top* vista na seção 3.4 - VIDA ARTIFICIAL.

3.8 Redes Neurais Artificiais (RNA)

Segundo (FAUSSET, 1994) uma Rede Neural Artificial (RNA) é:

"Um sistema de processamento de informações que tem certas características de performance em comum com redes neurais biológicas. RNA's têm sido desenvolvidas como generalizações de modelos matemáticos da cognição humana ou da biologia neural, baseadas nos seguintes princípios:

- O processamento das informações ocorre em muitos elementos simples chamados neurônios;
- Os sinais são passados entre os neurônios através de conexões;
- Cada conexão tem um peso associado, que, numa rede neural típica, multiplica o sinal transmitido;
- Cada neurônio aplica uma função de ativação (usualmente não-linear) sobre a próxima entrada da rede (que é a soma da multiplicação dos sinais pelos pesos) para determinar o sinal de saída.

Uma rede neural é caracterizada por possuir:

- Um padrão de conexões entre os neurônios (chamada de arquitetura da rede);
- Um método próprio para determinar os pesos das conexões (chamado de algoritmo de treinamento ou aprendizagem); e
- Uma função de ativação".

Mais detalhes podem ser vistos também em (HAYKIN, 1998).

RNA's se aplicam a problemas onde não existe um conhecimento prévio da distribuição estatística dos dados estudados. São tolerantes a erros durante o seu treinamento, podem generalizar resultados e também continuam a fornecer resultados corretos mesmo se houver uma perda de parte da rede (redundância de dados).

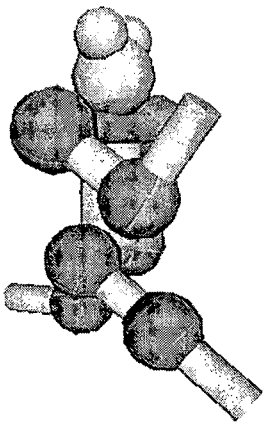
Redes Neurais devem ser treinadas para que aprendam a classificar os padrões fornecidos, e seu treino é limitado e pode ser refeito em épocas regulares, caso os dados também mudarem no correr do tempo.

Em (BASTOS, 1998) existem algumas regras sobre determinação de parâmetros de treino para redes neurais artificiais. Para maiores detalhes, ver (FAUSSET, 1994) e (HAYKIN, 1998).

(WATSON, 1996), em seu jogo exemplo, utiliza RNA's para que o agente inteligente do jogo (uma nave-tanque fornecedora de combustível) saiba navegar entre os meteoros no espaço.

(GRAND, 1997) explica alguns exemplos do comportamento das criaturas do *software Creatures*, utilizando RNA.

Fig. 11: Avatar visto em VRML.



3.9 *VRML*

Surgindo da necessidade de demonstrações de gráficos tridimensionais na rede mundial de computadores, a *Virtual Reality Markup Language* (Linguagem Incrementada para Realidade Virtual – Fig.11), iniciou em janeiro de 1995 (HARTMAN, 1996) com

com uma primeira versão que possibilitava somente visualização e navegação através de elementos tridimensionais estáticos.

Em agosto de 1996 foi feita a segunda versão desta linguagem, possibilitando não só a navegação através de mundos tridimensionais, mas também a interação do usuário com os modelos através do *mouse*.

Atualmente, empresas como a *Parallel Graphics* (maio de 2000) e *Blaxxun* (julho de 2000) possibilitam a interface do usuário com o mundo tridimensional através do teclado, também permitem a utilização multi-usuário de mundos *VRML* através de *software* específico. Em (MARRIN, 1997) é mostrado passo a passo a programação e o funcionamento da *VRML* 2.0.

A navegação de mundos *VRML* é permitida por *softwares* chamados *plug-ins*, que podem ser instalados em navegadores da rede mundial de computadores. Os navegadores mais usados atualmente são o *Netscape Communicator* (junho de 2000) e o *Microsoft Internet Explorer* (junho de 2000). Uma lista de *plug-ins* para *VRML* é dada na Bibliografia.

3.10 Redes de Computadores

3.10.1 Conceitos de Redes de Computadores

Segundo a (*T1 Glossary 2000*, 2002) e a (*ATIS*, 2002) uma rede de computadores é:

> "1 – Uma rede de nós que executam processamento de dados, que são interconectados com a finalidade de comunicação de dados. 2 – Uma rede de comunicação em que os instrumentos finais são computadores".

3.10.2 Conceito de Protocolo

Também segundo a (*T1 Glossary 2000*, 2002) e a (*ATIS*, 2002) um protocolo é:

> "1 – Um conjunto formal de convenções governando o formato e controle de interações entre unidades funcionais que se comunicam. Nota: Os protocolos governam porções de uma rede, tipos de serviço... 2 – Em arquiteturas de sistemas de comunicação em camadas, é um conjunto formal de procedimentos que são adotados para facilitar uma interoperação funcional dentro da hierarquia em camadas."

Dentro do escopo que fala sobre Redes de Computadores, neste trabalho, uma pesquisa foi feita sobre somente dois protocolos de comunicação de dados, o TCP/IP e o UDP.

3.10.3 TCP/IP

O TCP/IP (*Transmission Control Protocol / Internet Protocol*) se caracteriza por fazer a comunicação entre várias arquiteturas de redes de computadores diferentes (*X25*, *Token Ring*, entre outras), e possibilitar o acesso à *Internet* (HUNT, 1998).

A comunicação com TCP/IP se dá por pacotes de informação com tamanho dependente da rede de menor capacidade, e do fluxo de informações entre as redes. O protocolo também se certifica de que o endereço objetivo da transmissão está ativo, e então faz a transferência das informações.

3.10.4 UDP

Segundo (POSTEL e a RFC768, 1980) o UDP (*User Datagram Protocol*) se caracteriza por ser um protocolo de comunicação de dados, também através de

várias redes de computadores interconectadas, que utiliza um mínimo de mecanismos de protocolos. A chegada da mensagem depois de enviada, e a proteção duplicada não são garantidas.

3.10.5 Comparação entre Arquiteturas de Redes e Protocolos de Comunicação

Segundo (PAUSCH *et al.*, 1994) existem dois tipos básicos de arquiteturas de redes de computadores levando em conta sua comunicação: as arquiteturas "multiponto" e as arquiteturas "ponto a ponto" (*multicasting*).

As arquiteturas multiponto se caracterizam por possuírem um computador servidor, que se conecta com vários computadores clientes, e que cada cliente é conectado somente com o servidor.

As arquiteturas ponto a ponto se caracterizam pelo fato de que cada computador pode se comunicar com os demais computadores que participam da rede, sem a necessidade de uma máquina intermediária.

O autor também compara o modelo multiponto e ponto a ponto, apontando uma desvantagem da primeira: quando o número de clientes, e conseqüentemente, o fluxo de informações cresce muito, o servidor fica sobrecarregado, fazendo com que os pacotes de informação se acumulem em uma fila, ocasionando atrasos na resposta às ações destes clientes.

Quanto à atualização das propriedades, o autor afirma que esta pode ser feita de duas formas: A atualização por *frame* (tela) e a atualização pelo tempo.

A atualização por tela é feita dentro de um ciclo simples tal como um ciclo **para ... faça** ou **enquanto ... faça**, ou após um evento, tal como a pressão de uma

tecla. Desta formá, computadores com maiores velocidades movem seus objetos mais rápido do que computadores menos poderosos.

A atualização pelo tempo é feita utilizando o tempo como referência, ou seja, a atualização dos mundos de todos os computadores é feita a intervalos regulares de tempo. Isto faz com que todos os objetos atuem com a mesma velocidade em todos os computadores.

3.11 Detecção de Colisão

O estudo de técnicas de Detecção de Colisão em Realidade Virtual considera qualquer método de prevenção de interpenetração entre dois poliedros, que se movimentam e giram com o tempo.

Estas técnicas são aplicadas para simular, em Ambientes Virtuais, o fenômeno físico que rege que dois corpos não podem ocupar o mesmo lugar no espaço.

Segundo (GUIBAS et. al., 1999) a detecção de colisão entre dois poliedros é importante para preservar o realismo de ambientes virtuais:

"...detecção de colisão, que é crucial em se tratando de criar ambientes virtuais realísticos, pois a colisão induz a comportamentos incorretos ou ao visual prejudicado."

Em (JIMÉNEZ *et. al.*, 2000) as técnicas de computação gráfica estão intimamente ligadas com detecção de colisão:

"A Computação Gráfica engloba um vasto conjunto de aplicações relacionadas ao CAD, Realidade Virtual e Simulações Físicas, que requerem rápidas técnicas de detecção de colisão".

3.12 Geração Pseudo-Randômica Paramétrica de Ambientes Virtuais

Segundo LUBAN (2001), a grande maioria dos jogadores nunca chegam ao fim de seus jogos, não cumprindo seus objetivos, e coloca os motivos para tal fato:

> "Em realidade, uma grande maioria de jogadores nunca termina seus jogos. Um número de fatores explica este fenômeno. Primeiro, produtos que fornecem um estilo rígido de jogo terminam por cansar os usuários. Além disso, uma vez que os jogadores dominam os controles de um jogo em particular, o desafio termina, e este é o interesse. Muitos jogadores também abandonam o jogo quando estão em um labirinto ou em um ponto que eles acham impossível de sobrepujar."

Aqui se nota dois princípios que devem ser seguidos por projetistas de jogos: os jogos de computadores devem ser flexíveis, para que não fatiguem os jogadores, e deve existir um modo de criar novas possibilidades para um mesmo tema e ambiente de jogo, tal que não termine o desafio oferecido pelo jogo.

O método descrito aqui é chamado de Geração Pseudo-Randômica Paramétrica de Ambientes Virtuais pelos seguintes motivos:

Em (BURDEA, 1996-b, também citado em 3.3), define-se Ambientes Virtuais como sendo um ambiente simulado em que repetem os fenômenos físicos da realidade.

Em (HUONG et. al., 1998, também citado em 1.2), afirma-se que os parâmetros de interface homem-máquina que influenciam muito a sensação de realismo são o detalhe visual, a resposta do sistema e o fato de que o ambiente muda de acordo com a direção à qual o usuário está olhando.

Assim, conclui-se que, se o sistema é capaz de representar o Ambiente Virtual através de elementos geométricos tridimensionais com uma boa resposta à ação

do usuário, ele é capaz de oferecer ao usuário uma sensação muito realista deste Ambiente.

Por outro lado, se analisando o mundo real, observamos que este mundo é composto por vários elementos separáveis, possuindo suas propriedades independentes. Entre estas propriedades, pode-se destacar as propriedades visuais de cada elemento, tais como forma geométrica, textura e/ou cor, absorção, reflexão e emissão de luz.

Para cada elemento em separado, suas propriedades podem ser separadas em propriedades mais simples, numéricas, contínuas ou discretas, que uma vez todas determinadas, determina-se também a propriedade em questão deste elemento.

Estas propriedades simples são os **parâmetros** do elemento estudado, e uma vez determinados estes parâmetros, determina-se a propriedade total do elemento.

Se o mundo real pode ser separado em elementos discretos, com uma geometria definida, propriedades de textura, cor e luz, e, dado um sistema onde se consegue reproduzir um Ambiente Virtual em tempo real, então este Ambiente Virtual também pode ser composto de elementos discretos, cujos parâmetros são suas formas geométricas, cor, texturas e comportamentos de luz.

Uma forma de gerar – ou numa definição mais forte e arriscada – criar um Ambiente Virtual através de um meio digital é o objetivo deste trabalho, onde para isto, um dos requisitos é tornar este Ambiente tão imprevisível quanto a natureza real, utilizando-se então dos números pseudo-randômicos digitais.

Em (RIPLEY, 1987) define-se os números **randômicos** em termos de previsibilidade:

"...Na linguagem comum atual, chamamos os fatos de 'randômicos' quando não podemos prevê-los. ..."

Em (NEVEU *apud* RIPLEY, 1987) afirma-se que a provável existência dos números randômicos vem dos axiomas de Kolmogorov. Em muitos casos envolvendo simulação, a geração destes números provém da observação de fenômenos físicos.

Ainda em (RIPLEY, 1987), faz-se uma menção sobre o uso de recursos mecânicos para geração de imprevisibilidades em jogos de risco, tais como a roleta e as loterias (WEST *apud* RIPLEY e INOUE *apud* RIPLEY).

O autor então define o que são números pseudo-randômicos (também chamados de quase-randômicos):

> "Uma seqüência de números pseudo-randômicos (Ui) é uma seqüência de números determinísticos no intervalo [0,1], tendo as mesmas propriedades estatísticas relevantes de uma seqüência de números randômicos".

Por último, se um sistema pode gerar um Ambiente Virtual com bom realismo para o usuário, sendo que este Ambiente possa ser sintetizado a partir de elementos geométricos básicos, que são determinados a partir de parâmetros numéricos, e estes parâmetros numéricos podem ser produzidos pseudo-randomicamente por este mesmo sistema, então um Ambiente Virtual pode ser gerado a partir de números pseudo-randômicos, por meio de um conveniente algoritmo, disso se constituindo a Geração Pseudo-Randômica Paramétrica de Ambientes Virtuais.

3.13 Reconhecimento de Palavras e Sistema de Diálogos tipo Questão-Resposta para Interação Falada com *Bots*

A ciência do Reconhecimento de Palavras consiste no estudo de um conjunto de algoritmos e de interfaces que tem a finalidade de reconhecer uma palavra isolada, ou um conjunto de palavras, ditas por um humano, separadamente com espaços de tempo entre elas, ou continuamente, com a mínima margem de erro possível.

Em (TAFNER, 1996) é descrito a pesquisa com um reconhecedor de palavras isoladas implementando um sistema contendo Redes Neurais Artificiais (ver seção 3.8 Redes Neurais Artificiais).

Por sua vez, um Sistema de Diálogos é – segundo (ECKERT, 1996) – um sistema de algoritmos que tem a finalidade de iniciar e/ou continuar o diálogo com o usuário, tendo inclusive alguma iniciativa.

O modelo de ECKERT é composto de duas etapas com um complicante:

- Reconhecimento de Palavras, com o problema da
- Fala Espontânea, e o
- Processador do Diálogo.

A partir da pesquisa e classificação do autor, o problema de Reconhecimento de Palavras (o entendimento da expressão do usuário) pode ser separado em quatro classes:

- Reconhecer significa somente reconhecer a seqüência falada;
- Reconhecer significa construir uma estrutura interna;

- Significa deduzir todas as conseqüências da expressão falada;
- Significa reagir de forma apropriada e inteligente.

O problema da Fala Espontânea engloba todo tipo de fala não ideal, ou seja, todas as redundâncias, retornos, correções, pausas, palavras fora de ordem e reinícios, feitas num diálogo normal entre dois seres humanos.

Também das suas pesquisas a partir de Sistemas de Diálogos já feitos, os Processadores de Diálogos se constituem dos seguintes processos, baseados nas características de iniciativa do diálogo e no conjunto de respostas esperadas do sistema:

- Sistemas de Menus, que oferecem as alternativas com uma pergunta, com exemplo de serviços telefônicos de empresas com ajuda *on-line*, onde os tons das teclas são a resposta do usuário, em vez de palavras faladas;
- Sistemas de Pergunta-Resposta (**PR**), onde o usuário requisita um conjunto de ítens, e o sistema oferece como resposta um conjunto que preencha totalmente o pedido do usuário. Para exemplificar, o autor refere-se a um sistema chamado ATIS;
- Sistemas Conversacionais, que possuem a habilidade de moverem iniciativa de um lado a outro da conversa, e vice-versa. Os diálogos contêm muitas voltas, com requisições, respostas, elucidações, confirmações, e outros. Em Sistemas Conversacionais, o alcance de um objetivo complexo é separado em vários passos que são executados em seqüência, enquanto no **PR** estes passos são combinados num único sentido;

Finalmente, segundo (ROEHL *et al.* 1997) e (DAMER, 2000), *bots* são objetos controlados por processos autônomos, que participam de um mundo virtual.

3.14 Navegação em Ambientes Virtuais

A Navegação em Ambientes Virtuais consiste de todos os tipos de deslocamento que podem ser feitos dentro de um Ambiente Virtual. Segundo (McKINLAY *apud* STEED, 1993), a navegação em Realidade Virtual engloba quatro tipos:

- Movimento Geral, ou seja, a exploração de um modelo de construção;
- Movimento por Objetivos, ou seja, movimento a um ponto específico de interesse, tal como para examinar um detalhe;
- Movimento por Coordenadas Específicas, que é um movimento para um ponto específico dentro de um sistema de coordenadas;
- Movimento por uma Trajetória Especificada.

Existem duas técnicas básicas para estes tipos de navegação: ou o ponto de vista se move em relação ao ambiente, ou o ambiente se move em relação ao ponto de vista do usuário, sendo uma questão de diferentes referências.

Em (WARE e OSBORNE *apud* STEED, 1993) são dadas metáforas que exemplificam estas diferenças:

- Movimento do periférico de entrada de sinais corresponde exatamente ao olho do observador;
- Cenário do Ambiente Virtual está na mão, ou seja, uma rotação e/ou translação estão ligadas ao movimento do periférico de entrada sob controle do usuário;

- Ponto de vista funciona como um veículo flutuante, tendo sua rotação e velocidade controladas pelo usuário, através do periférico;

Partindo das classes de navegação vistas acima, pode-se notar a existência de um problema técnico relativo a um Ambiente Virtual: a dimensão física que este Ambiente pode alcançar.

Para tal tipo de problema, algumas bibliografias oferecem duas soluções diferentes: o uso de locais (BARRUS *et al.* 1996), e o uso de âncoras (HARTMAN, 1996, ROEHL, 1997, MARRIN, 1997).

Segundo (BARRUS *et al.*, 1996) há a necessidade de criação de mundos virtuais compartilhados de larga extensão, mas acarretando uma série de problemas:

> "Existe um desejo natural de criar ambientes multi-usuário de larga extensão, com grande quantidade de objetos, e larga quantidade de usuários interagindo com o ambiente. Mas isto acarreta em uma série de problemas: o gerenciamento eficiente do fluxo de grandes quantidades de dados entre uma grande quantidade de usuários; o posicionamento preciso e a informação sobre a velocidade de objetos dispostos em um grande volume de espaço; e a possibilidade de os projetistas criarem partes deste ambiente separadamente, e combiná-los juntos depois".

Para solucionar tais problemas, (BARRUS *et al.*, 1996) propõe o novo conceito de LOCAIS, que consiste no fato de que, apesar de um mundo virtual ser muito grande, o usuário vê somente a parte do mundo que está navegando:

> "O conceito de LOCAIS é baseado na idéia de que, apesar de um mundo virtual ser muito grande, muitas partes deste mundo, que podem ser observadas por um simples usuário em um dado momento, não são necessariamente locais em sua natureza. Isto quer dizer que um usuário poderia esperar que um grande mundo virtual seria grande primariamente porque,

como numa cidade, este combina um grande número de pequenas atividades localizadas (por exemplo, uma conversação envolvendo algumas pessoas aqui e uma outra simulação envolvendo vários outros objetos em outro lugar), muito mais do que uma atividade individual muito grande e complexa."

Por outro lado, a tecnologia de âncoras consiste de uma substituição de um Ambiente Virtual por outro, a partir de uma ativação de um elemento deste Ambiente Virtual anterior (The Virtual Reality Modeling Language, janeiro de 2002).

Um ambiente virtual compartilhado criado randomicamente também pode alcançar grandes dimensões, o que sugere que as tecnologias de locais e âncoras podem ser usadas.

Em alguns jogos pode-se notar o uso de âncoras, sempre que o usuário passa de um nível de dificuldade para o seguinte. Jogos como *Doom* I e II, *Wolfenstein*, *Quake* e *Duke Nuken 3D* (ID Software, junho de 2000) utilizam esta técnica. *Half-Life* (Sierra Studios, maio de 2000) utiliza vários arquivos em disco para descrever seus ambientes.

3.15 *Hardware* para Interface Homem-Máquina

No Dicionário Houaiss da Língua Portuguesa (HOUAISS, 2001) encontra-se, para a palavra emprestada do inglês *hardware*, no ramo da Informática, a seguinte definição:

"O conjunto dos componentes físicos (material eletrônico, placas, monitor, equipamentos periféricos, etc.) de um computador".

Ainda no mesmo dicionário, encontra-se o seguinte significado para a palavra interface:

"1 – Elemento que proporciona uma ligação física ou lógica entre dois sistemas, ou partes de um sistema que não poderiam ser conectados diretamente; 2 – INFORMÁTICA: Fronteira compartilhada por dois dispositivos, sistemas ou programas que trocam dados e sinais; 3 – INFORMÁTICA: Meio pelo qual o usuário interage com um programa ou sistema operacional (por exemplo: DOS, Windows)."

Um sistema de Realidade Virtual deve dar ao usuário a sensação de presença no Ambiente Virtual, e também permitir a interação deste usuário com o ambiente. Para dar estas sensações, é necessária uma interface entre o usuário e este Ambiente Sintético.

Tal interface deve ser capaz de, no mínimo, compreender os estímulos deste usuário, processá-los para a forma de sinais digitais para o computador, e fornecer de volta uma resposta perceptível para aquele usuário. Esta transformação de sinais humanos para sinais digitais e vice-versa é feita pelo *hardware* periférico conectado à máquina.

Dependendo do tipo e formato de sinal, e que tipo de transdução um *hardware* específico fará durante a transferência de dados homem-máquina na sua interação virtual, uma pré-programação deve ser feita na máquina, indicando como serão processados os dados analógico-digitais provenientes do usuário, e como retornarão estes dados.

(STEED, 1993) afirma que o equipamento de Realidade Virtual tem sido popularizado pela mídia como um sistema computadorizado que possui os seguintes componentes:

- Um *display* montado na cabeça (traduzido do inglês *Head Mounted Display*), que possui dois projetores de imagem, um para cada olho, e que é capaz de

mostrar uma imagem tridimensional do ambiente ou objeto sintético para o usuário.

- Um sistema de rastreamento que pode identificar a posição e a orientação da cabeça do usuário.
- Um periférico de entrada de dados digitais, que pode ser segurado ou vestido na mão do usuário, que também é rastreado, e que serve para localização e/ou captura de objetos virtuais.
- Alguma forma de saída de sinais digitais de áudio.
- Um banco de dados que contém informações sobre o Ambiente Virtual, seus objetos componentes e suas formas de interação.

4 APLICAÇÃO E ANÁLISE

4.1 Observações e Considerações

No Anexo I – PESQUISA DE CAMPO E RESULTADOS – nota-se a grande preferência dos usuários por jogos envolvendo corridas de veículos (35% de uma amostra de 80 indivíduos), e a idade dos jogadores abrange uma faixa de 7 (sete) a 28 (vinte e oito) anos, totalizando 92% (noventa e dois por cento) do público.

No mesmo anexo é mostrado também que uma boa parte dos entrevistados (setenta e nove por cento) prefere tipos de jogos que envolvam o raciocínio, e que 99% (noventa e nove por cento) do público que jogava no momento da entrevista era do sexo masculino.

Finalmente, a pesquisa revela que setenta e sete por cento dos entrevistados gostariam de sistemas de jogos que geram ambientes virtuais aleatórios, e oitenta

e seis por cento destes preferem a existência de personagens que possibilitem um diálogo com o jogador.

Em (TODESCHINI, 1999) é relatada uma observação feita em uma máquina de *fliperama* de jogos de luta, de que aproximadamente metade das vezes em que um personagem era escolhido, a escolha era feita para um personagem que lutasse capoeira, com traços culturais semelhantes aos lutadores brasileiros.

A partir destes elementos e das observações feitas neste item, o tema do jogo deve considerar os seguintes fatores:

- Adequação do jogo à cultura e idioma do indivíduo, ou grupo, que está jogando;
- Personagens semelhantes aos tipos físicos do país natural dos jogadores, e em boa parte do sexo feminino e/ou de raça negra ou índia;
- Criação de situações que exijam o raciocínio;
- Possibilidade de jogo permitindo algum tipo de interação social (jogo via redes de computadores);
- Criação de situações não violentas, e que despertem interesse em jovens.

4.2 Discussão

Segundo (HEETER, 1994), ao analisar os motivos pelos quais as pessoas jogavam *BattleTech*, concluiu que as mulheres estão menos interessadas em violência e competição:

> "Diferenças significantes na idade aparecem nos ítens **Divertimento e Excitação** e **Violência e Competição**, com os adultos reportando um pouco menos de prazer em *BattleTech* do que os jovens. Diferenças significantes na idade aparecem nos motivos **Violência** e **Competição**

> e também no motivo **Realidade Virtual**. **Mulheres estão menos interessadas em violência e competição**. Elas estão mais inclinadas a dizer que vieram jogar *BattleTech* para experimentar Realidade Virtual."

Ainda em HEETER, existe uma suspeita de que muitos usuários do jogo *BattleTech* vão ao jogo para ter alguma interação social:

> "... De fato, os jovens desistem depois de jogarem menos de 10 vezes, **e a jogabilidade** e **as interações sociais se tornam as razões para continuar jogando**. Os motivos para interação social são significativamente altos para mais de cinqüenta fanáticos em jogos do que para outro grupo de jogadores".

PROVENZO (*apud* HART, 2001) descobriu que as mulheres estavam mais interessadas em jogos de raciocínio:

> "... No período em que a pesquisa foi conduzida, o gênero mais popular de jogos, entre os jogadores do sexo masculino, era o de luta com visão lateral. Os gêneros populares entre as mulheres eram os jogos de planejamento (em inglês: *Role Playing Games*) e os jogos de quebra-cabeça".

4.3 Tema e Modelo do Jogo

O modelo escolhido foi um jogo de corrida de veículos, apresentando poucas situações de violência (classificando-se em choques entre veículos, choques de veículos com obstáculos e choques de veículos com elementos simulando vida artificial).

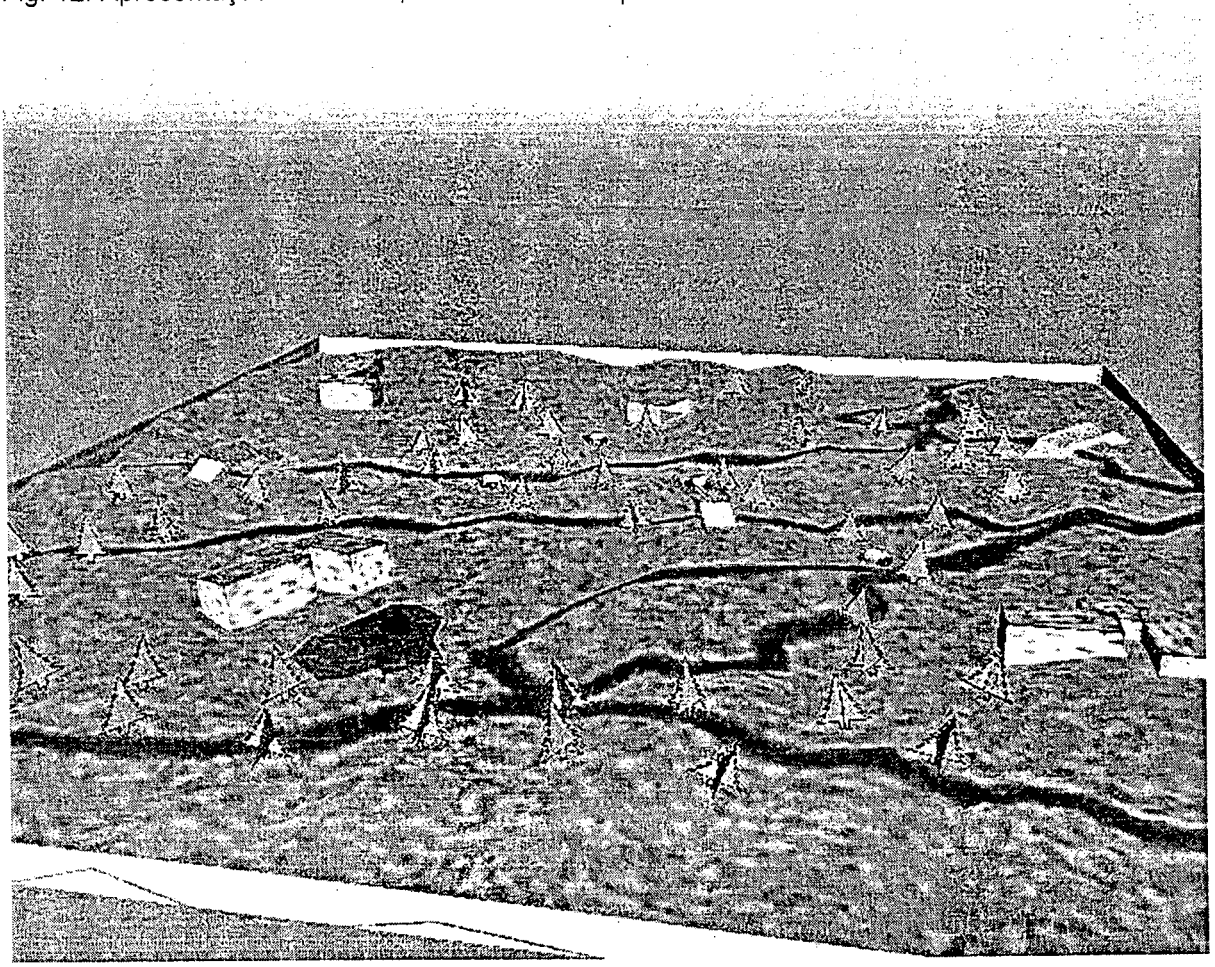
O modelo cria um ambiente pseudo-aleatório, a partir de algoritmos de pré-processamento, e este ambiente gerado pode ser compartilhado por vários jogadores, permitindo uma interação social limitada.

Este modelo usa expressões do idioma português do Brasil, o que mostra uma adequação do jogo à cultura brasileira.

4.4 Componentes Principais do Jogo

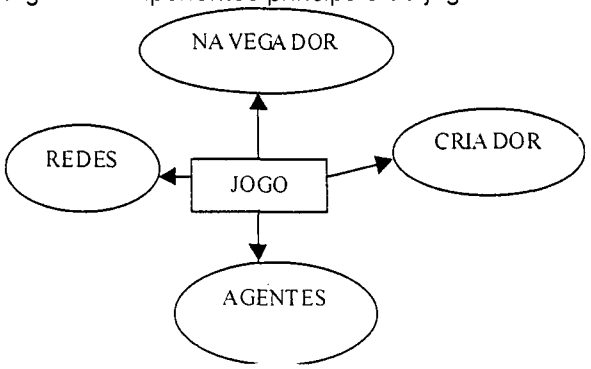
Na Figura 12 é mostrado um esboço tridimensional – feito em VRML – de como ficaria um mundo depois de gerado. Nota-se o terreno acidentado, com cor predominante vermelha, três veículos, um lago, algumas ramificações de rios, três pontes, três paredes e várias árvores.

Fig. 12: Apresentação do mundo, visto em terceira pessoa.



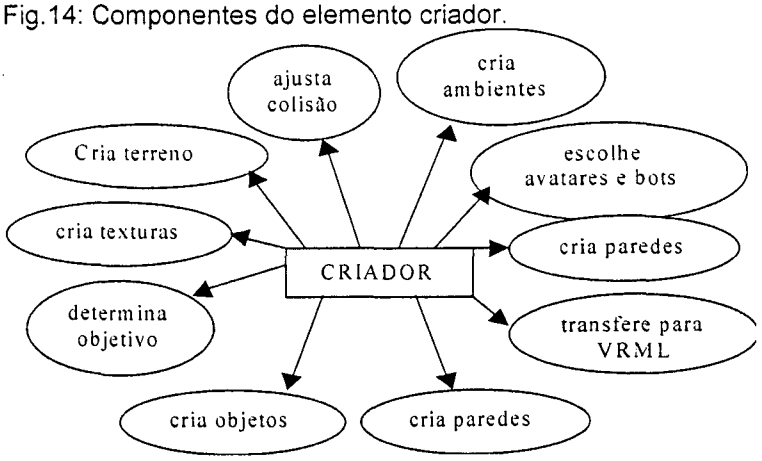
O modelo do jogo se consiste de quatro componentes principais que farão o gerenciamento dos objetos, personagens e comportamentos durante o decorrer do evento: CRIADOR, REDES, NAVEGADOR E AGENTES (Fig. 13).

Fig.13: Componentes principais do jogo.



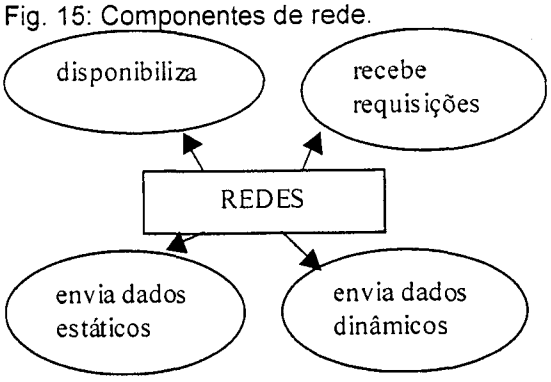
- Componente criador (Fig. 14): responsável pela criação de objetos estáticos e dinâmicos com base em número pseudo-randômicos, faz o mapa do terreno onde o jogo se realizará, calcula os pontos de colisão, cria os ambientes ligados ao jogo, passa o mundo criado para arquivos de formato *VRML* e determina o objetivo do jogo a ser alcançado.

Fig.14: Componentes do elemento criador.



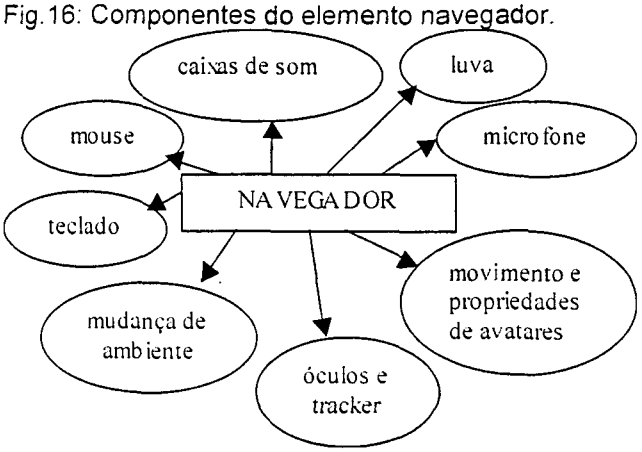
- Componente de rede (Fig. 15): responsável por fazer a comunicação entre os computadores, disponibilizando conexões para possíveis requisições de clientes, gerencia também a troca de informações de dados dinâmicos sobre os elementos do jogo.

Fig. 15: Componentes de rede.



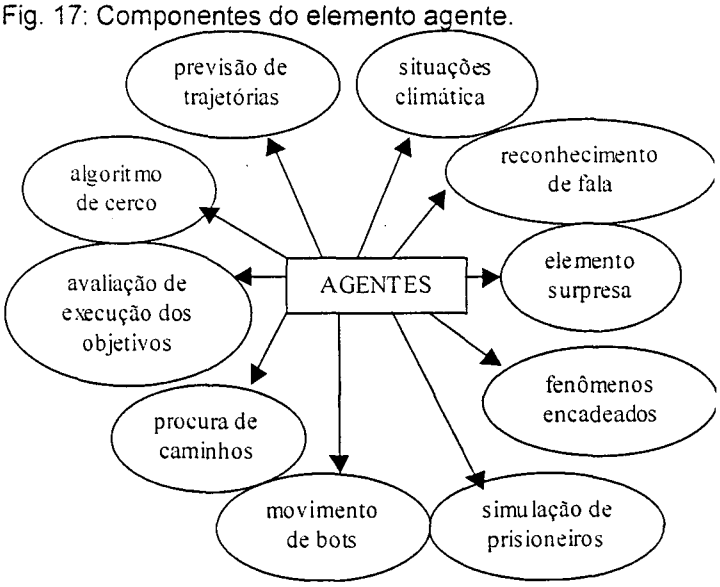
- Componente navegador (Fig.16): gerencia a interface dos usuários com o computador, dando a resposta programada a cada ação do usuário. Responsável pelo posicionamento e propriedade dos jogadores dentro do mundo.

Fig.16: Componentes do elemento navegador.



- Componente de agentes (Fig. 17): responsável pela inteligência dos personagens presentes, e pela avaliação das tarefas executadas pelos jogadores para o término do jogo.

Fig. 17: Componentes do elemento agente.



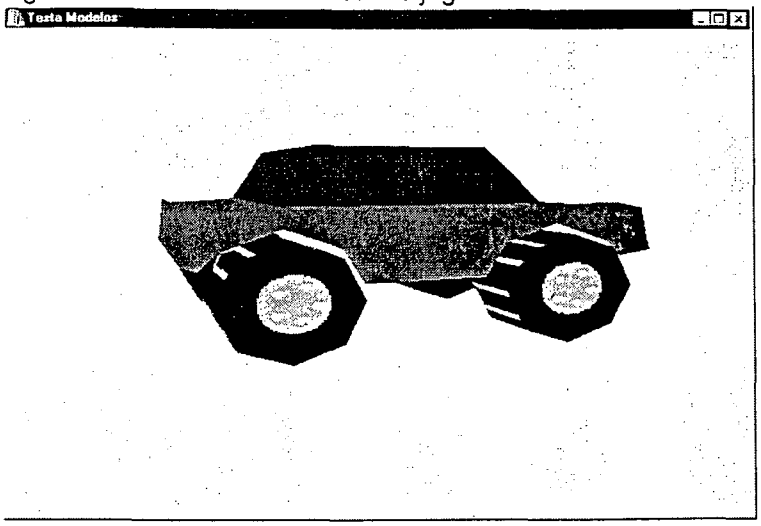
4.5 Utilização das Tecnologias Apresentadas para a Criação do Protótipo

Das tecnologias apresentadas aqui, e que foram estudadas superficialmente ou com certa profundidade, quase todas foram aplicadas em algoritmos com seus respectivos protótipos (Técnicas de Algoritmos Genéticos para solução de problemas sendo a única exceção).

Um conjunto um pouco mais restrito foi utilizado para implementar o protótipo do jogo de corridas de automóveis, incluindo Geração Pseudo-Randômica de

Cenários Tridimensionais e Texturas, Apresentação Gráfica (*rendering*), Produção de Mensagens Sonoras, Interação e Imersão utilizando *Hardware* de Realidade Virtual, Comunicação em Redes de Computadores, Técnicas de Previsão de Colisão e Previsão de Trajetórias, e Vida Artificial.

Fig. 18: Modelo de veículo visto no jogo.



Alguns ítens adicionais foram criados, tais como um visualizador de modelos 3D, onde o modelo é girado com o uso das teclas ou das luvas digitais, como se o usuário estivesse segurando um objeto (Fig. 18), um simulador de comportamento em grupo para Vida Artificial, com avatares, e um sistema de treinamento de

Fig. 19: Tela principal do *software* de treinamento de rede neural multi-camadas para reconhecimento de palavras faladas.

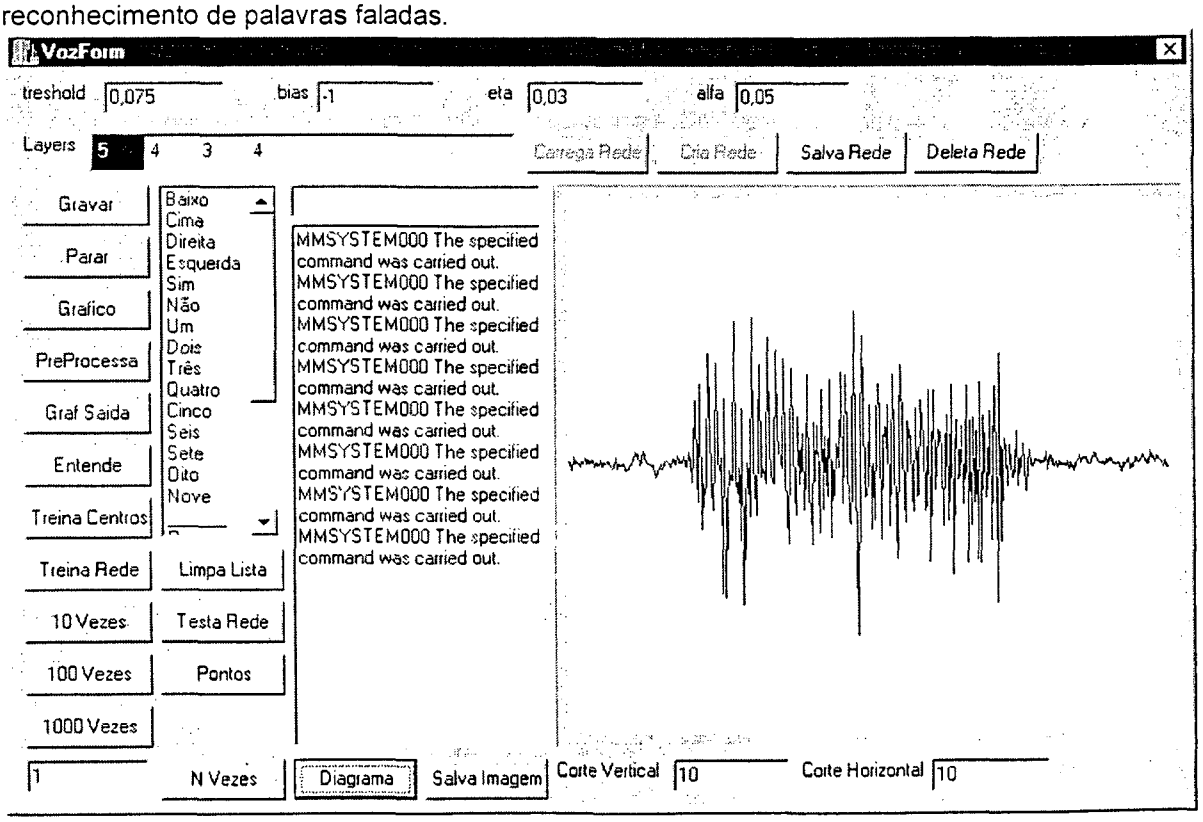
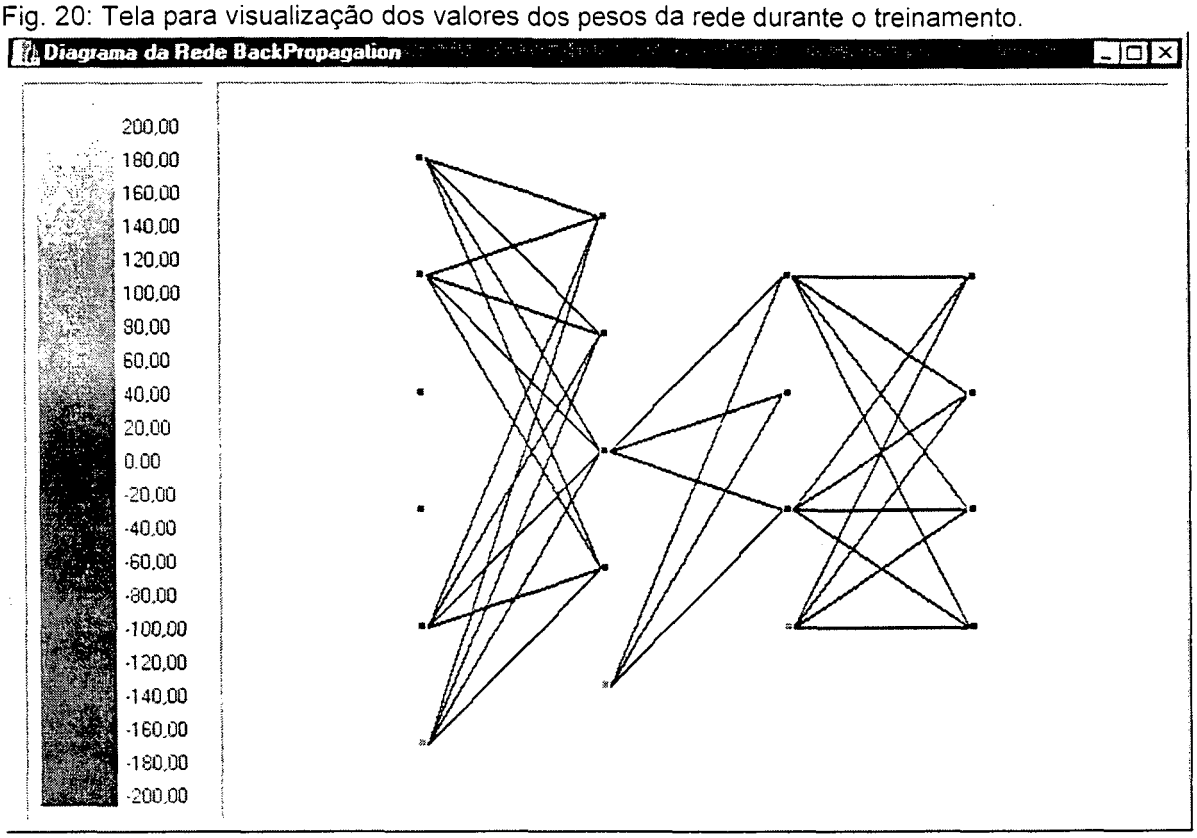


Fig. 20: Tela para visualização dos valores dos pesos da rede durante o treinamento.



Redes Neurais para reconhecimento de palavras isoladas, como ferramentas temporárias para chegar ao fim do protótipo (Fig. 19 e 20).

O ítem Diálogo tipo Questão-Resposta foi implementado temporariamente, junto com um *bot* humanóide, e foram posteriormente retirados devido ao insucesso em seus resultados.

4.5.1 Escolha das Ferramentas de Trabalho

Para que tais tecnologias pudessem ser implementadas, algumas necessidades foram detectadas:

- Deve haver um meio de comunicação imediata entre vários computadores, e um protocolo de comunicação deve existir para transferir os dados estáticos e dinâmicos do jogo em tempo real.

- Deve existir uma interface para que o computador entenda os movimentos e informações vindas do ser humano, e repassá-las novamente para ele, assim como um protocolo de comunicação entre o computador e esta interface.
- O desenvolvimento do protótipo deve levar pouco tempo, ter custo reduzido, ser simples, reutilizável e ter grande quantidades de fontes de informação e consulta.

Por estas razões a linguagem escolhida para o trabalho foi a linguagem C orientada a objetos, com biblioteca gráfica *OpenGL* (JR. WRIGHT, 1996), rodando num sistema operacional *Windows 98©*, com sua Interface de Programação de Aplicações (*Appllcation Program Interface - API*).

Para visualização rápida dos elementos tridimensionais do mundo antes do jogo, a linguagem escolhida foi a *VRML*, por ser uma linguagem compilada e compreensível em modo texto, podendo sua reprodução ser rapidamente automatizada.

Todos estes ítens possuem várias fontes de informação disponíveis na rede mundial de computadores (*Internet*).

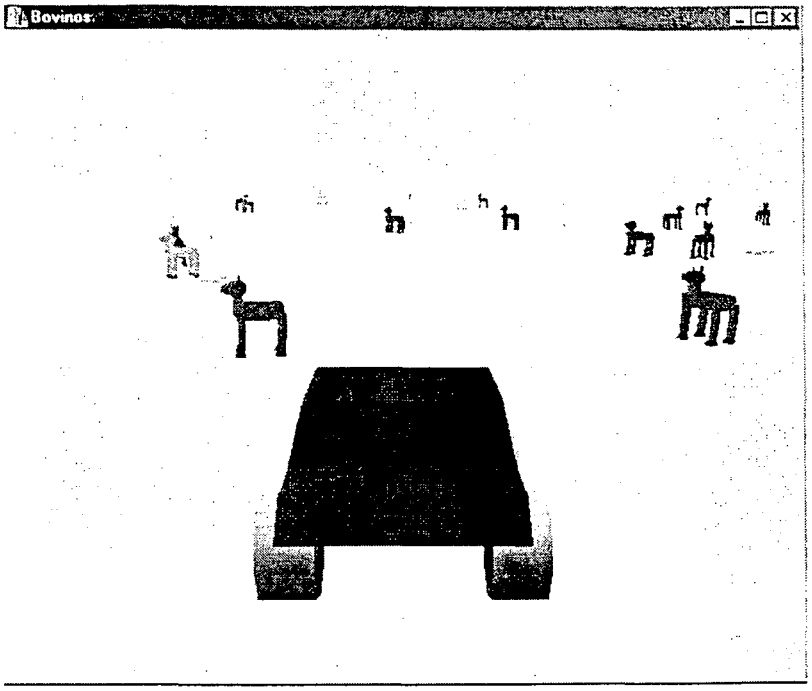
4.5.2 Utilização de Programação Orientada a Objetos

A programação orientada a objetos foi usada em toda a implementação, desde a parte de algoritmos mais básicos até a leitura de dados a partir dos periféricos de *hardware*, redes de computadores, programação de sons e a biblioteca gráfica *OpenGL*.

4.5.3 Vida Artificial

Um modelo simples de *bots* com vida artificial foi proposto e implementado. O modelo se consiste em um grupo de elementos que possuem movimentação suave e aleatória. Se o usuário tenta se aproximar dos componentes do grupo, estes fogem em linha reta. O resultado é mostrado na Figura 21.

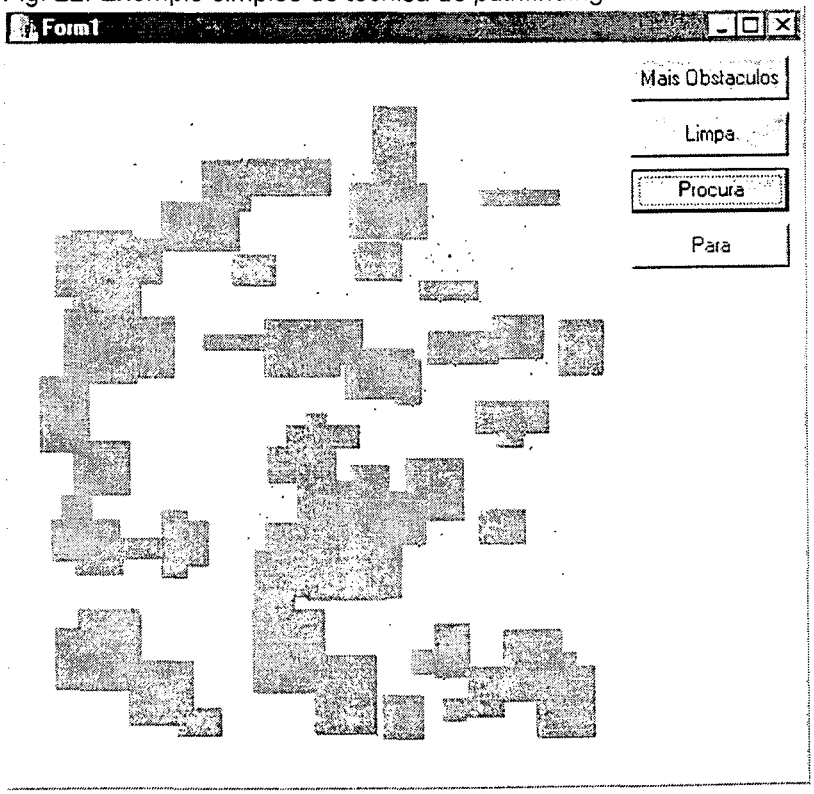
Fig. 21: Figura de programa que simula o comportamento de uma manada de bovinos.



4.5.4 *Pathfinder*

Um método vetorial de procura por trajetórias foi programado para funcionar com mapas em imagens tipo *raster*. Este método foi baseado num algoritmo de pré-processamento para identificação de

Fig. 22: Exemplo simples de técnica de *pathfinding*.



contornos para reconhecimento de imagens do mesmo tipo, descrito em (DUDA, 1973). Desta forma, o agente inteligente é programado para andar pelas bordas dos obstáculos.

No Anexo II é mostrado o fluxograma para a locomoção de um ponto entre a sua posição inicial e seu destino.

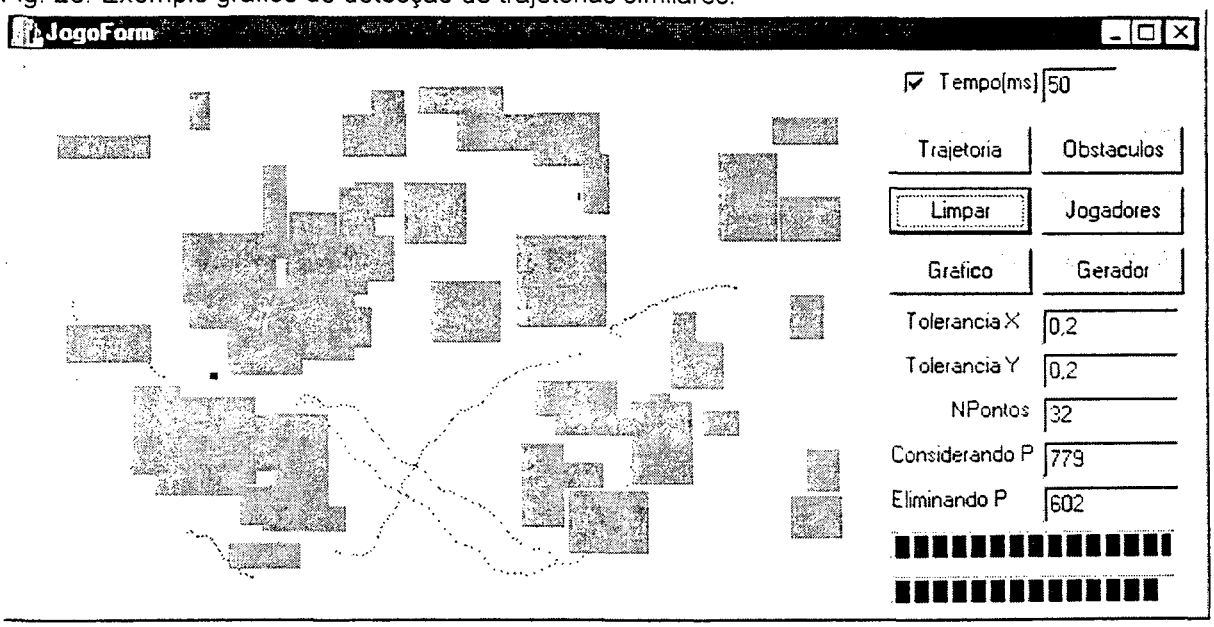
Na Figura 22, um conjunto de duzentos personagens, representados por minúsculos pontos pretos, foi programado para partir de uma posição aleatória inicial e se locomover até o ponto em vermelho, evitando os obstáculos em cinza, criados pseudo-randomicamente.

4.5.5 Previsão de Trajetórias com Mineração de Dados

A implementação de um algoritmo de previsão de trajetórias foi feita somente para um ambiente bidimensional discreto (uma imagem *raster*) onde o labirinto gerado era composto de vários blocos de formato quadrado, posicionados de forma pseudo-randômica.

Este algoritmo não foi implementado no protótipo final do jogo.O modelo criado possui três agentes inteligentes funcionando em programação concorrente através de *threads*. O primeiro deles é responsável por capturar as posições consecutivas do jogador, e os outros dois agentes se encarregam de prever a possível e futura trajetória do jogador a partir de seu histórico de trajetórias.

Fig. 23: Exemplo gráfico de detecção de trajetórias similares.



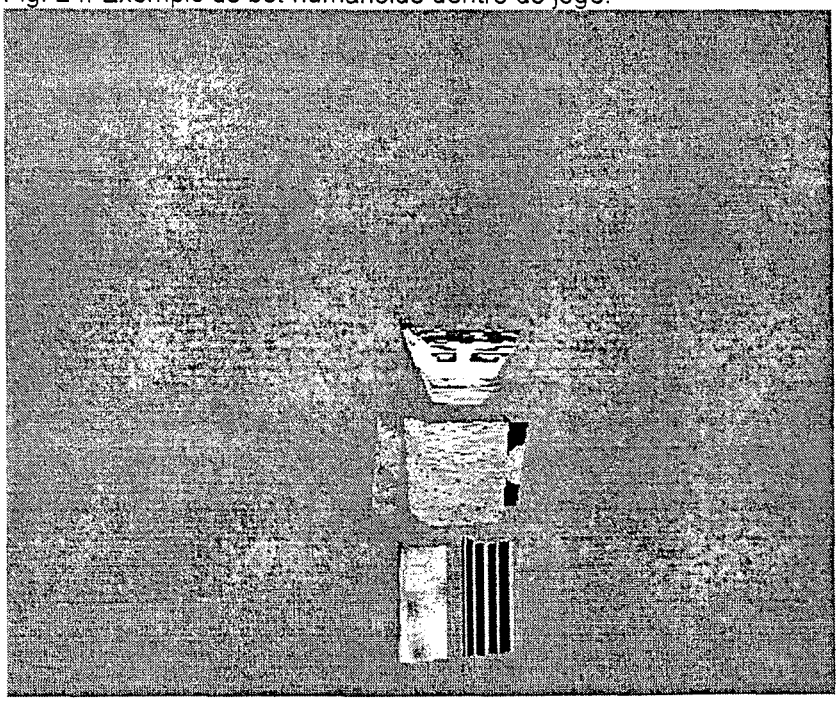
Na Figura 23 é mostrada uma tela deste caso prático em que a partir da última posição do jogador (ponto vermelho) o método estima uma possível trajetória que aquele pode tomar (em vermelho). O método identifica as sub-trajetórias semelhantes encontradas (marcadas em azul ou verde), com as trajetórias estimadas a partir da última posição mostradas em vermelho.

4.5.6 Sistema de Diálogo tipo Questão-Resposta utilizando Reconhecimento de Palavras Isoladas com ajuda de RNA

4.5.6.1 Diálogos Parametrizados

Um modelo de diálogo é proposto para o comportamento de *bots* (ver seção 3.13), que podem representar personagens com personalidades diferentes, evidenciados durante um eventual diálogo falado com o jogador.

Fig. 24: Exemplo de *bot* humanóide dentro do jogo.



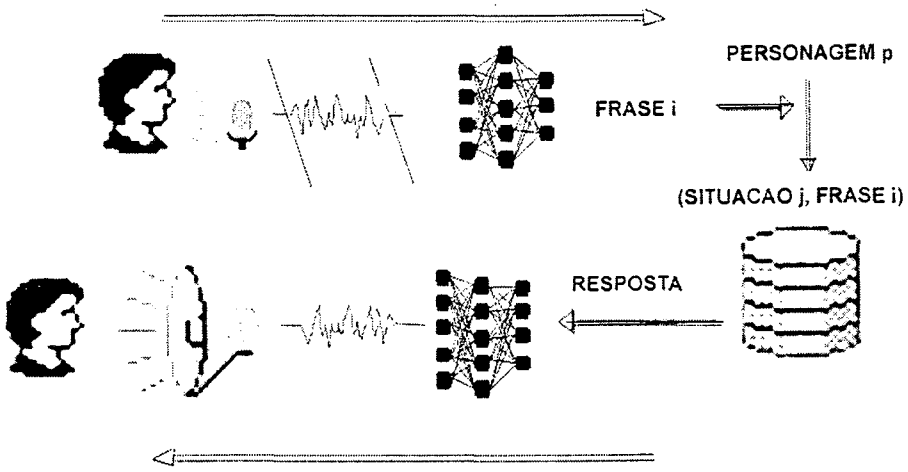
Neste modelo experimental, ao se aproximar do personagem autônomo (*bot*, Fig. 24), o usuário tem a disponibilidade de falar no momento em que desejar, enquanto estiver nas proximidades.

A máquina, munida de um sistema de disparo e reconhecimento de sinais de fala, processa os sinais digitais provenientes do microfone e, com a ajuda de uma função matemática, encontra o resultado final em forma de vetores. Este resultado final é um vetor correspondendo à palavra dita pelo usuário.

O personagem, que já tem suas características escolhidas aleatoriamente pela máquina, responde á pergunta do usuário usando um sistema baseado em regras, consulta a sua base de dados e respondendo com algumas frases já pré-gravadas. A Figura 25 possui um fluxograma do processo descrito acima.

Este algoritmo foi implementado temporariamente no protótipo do jogo e depois retirado, pois fornecia muitos resultados errados.

Fig. 25: Diagrama esquemático do sistema de diálogos parametrizados.



Abaixo é mostrada uma tabela com exemplos de personagens que simulam as personalidades infantil, formal e agressiva, sendo reproduzidas através de gravações sonoras faladas.

No protótipo programado, em determinado momento, o jogador se aproxima de qualquer um dos *bots* dentro do mundo virtual, que, dependendo de sua personalidade lhe faz uma pergunta oferecendo alguma coisa.

O jogador então responde "Sim" ou "Não". Dada a resposta do jogador, o *bot* entende a resposta, com o uso de uma RNA, e finalmente responde, também dependendo de sua personalidade (Tabela 1).

Tabela 1: Respostas dependentes da personalidade.

Personalidade	**Pergunta**	**Resposta do Jogador**	**Resposta do *Bot***
Infantil (fala mansa, fina e intimidada).	"O senhor quer alguma coisa?".	"Sim"	"O que o senhor quer comprar?".
		"Não"	"Tá bom".
Formal (fala educada e clara).	"Deseja alguma coisa, senhor?".	"Sim"	"O que gostaria de comprar, senhor?".
		"Não"	"Fique à vontade, senhor".
Agressivo (fala rápida, alta e ríspida).	"Quer comprar alguma coisa aí o maluco?".	"Sim"	"O que é que tu quer comprar?".
		"Não"	"Tá limpo!".

4.5.6.2 Reconhecimento de Fala

Para que o sistema de diálogo funcione no protótipo, a máquina deve ser capaz de processar e entender os dados da palavra falada pelo usuário. Uma escolha inicial de como os dados digitais de fala são capturados e armazenados (dados de entrada), como os dados de saída para o significado das palavras serão armazenados, e como os dados de entrada serão processados para chegar aos dados de saída.

O processamento feito pela máquina é composto de sete fases:

- Captura e conversão dos sinais de fala para sinais digitais por um microfone;
- Pré-processamento do sinal capturado para produzir o vetor de dados de entrada;
- Geração dos dados de saída com o uso de uma rede neural tipo *LVQ2* (FAUSSET, 1994) de treinamento não supervisionado;
- Treinamento supervisionado e teste de uma rede neural *MLP* com *backpropagation* (FAUSSET, 1994 e HAYKIN, 1998);

- Processamento do sinal de fala pela rede neural *MLP*, fornecendo vetores de saída;
- Interpretação destes vetores de saída convertendo-os em palavras pré-armazenadas.

Em (TAFNER, 1996) é feita uma extensa descrição de como funciona o mecanismo de captura de som pelo microfone, o pré-processamento dos sinais digitais de entrada e o processamento dos sinais resultantes com uso de RNA.

Abaixo são dadas explicações de cada um destes itens, com uma breve revisão sobre algumas propriedades e características de RNA.

4.5.6.2.1 Captura de Sinais via Microfone

A comunicação via microfone foi programada usando a Interface de Programação de Aplicações (*API - Application Program Interface*) do Sistema Operacional *Windows 98 ©*. Os sinais analógicos capturados são armazenados na memória volátil (RAM) do micro na forma de vetores.

4.5.6.2.2 Pré-Processamento da Fala

Após a captura do sinal de fala pelo microfone, um pré-processamento é feito para eliminar dados redundantes ou sem valor para a rede neural, reduzindo a quantidade de sinais e aumentando a sua representatividade.

No algoritmo de pré-processamento implementado, foram utilizados quatro procedimentos para aumentar a chance do reconhecimento do sinal pela rede:

- Corte dos sinais de silêncio nas extremidades do sinal principal;
- Tomada do tempo resultante do sinal, e normalização deste tempo em relação a uma base;
- Normalização do sinal de voz no eixo do tempo;
- Normalização do sinal de voz no eixo da intensidade de som.

As Figuras 26 a 28 mostram o sinal original e os sinais resultantes depois de cada processo.

Fig. 26: Sinal de voz original para a palavra "Sim".

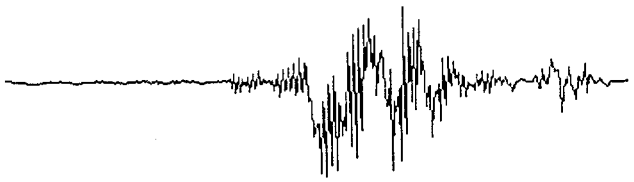


Fig. 27: O mesmo sinal pré-processado com os cantos cortados.

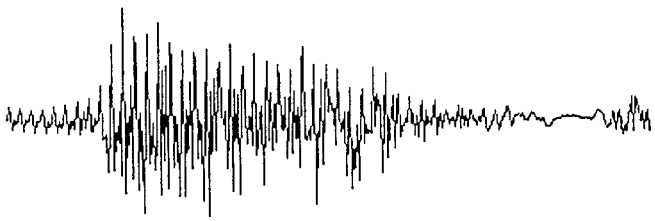
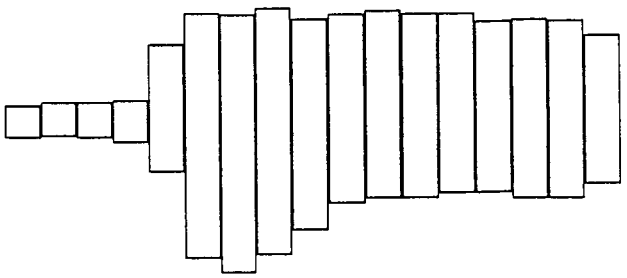


Fig. 28: Sinal normalizado nos dois eixos.



4.5.6.2.3 Geração dos Dados de Saída com Uso de RNA tipo *Multi-Layer Perceptron* e *Learning Vector Quantization*

Para a função de reconhecimento de fala, dois tipos de redes foram usados: Rede *Perceptron* Multi-camada (ou *Multi-Layer Perceptron* – FAUSSET, 1994 e HAYKIN, 1998) e uma rede derivada da rede de *Kohonen* (FAUSSET, 1994 e HAYKIN, 1998): rede *Learning Vector Quantization* tipo 2 (*LVQ2*) (FAUSSET, 1994).

A rede *LVQ2* foi usada para encontrar o centro ótimo representativo das palavras "Sim" e "Não", e a rede *MLP* foi usada para o treinamento propriamente dito.

Os dados de saída são representados por vetores de duas dimensões, bipolares (FAUSSET, 1994), de valores constantes, no intervalo [-1,1].

Inicialmente não se sabe qual a melhor representação vetorial para as duas palavras que servem como resposta para a rede ("Sim" e "Não").

Devido a este fato, uma rede neural de treinamento não supervisionado, tipo *LVQ2* (FAUSSET, 1994) foi usada para estimar os valores de cada vetor representativo (centros de resposta) de cada uma das duas palavras.

4.5.6.2.4 Treinamento, Teste de uma Rede Neural *MLP* e Conversão dos Vetores de Saída em Palavras Pré-Armazenadas

A função que recebe os vetores digitais pré-processados de entrada e fornece os vetores de saída representativos das palavras é uma rede neural *Perceptron* Multi-Camadas (*Multi-Layer-Perceptron*).

Para que a rede possa fornecer os valores esperados com os dados de entrada correspondentes, um treinamento tipo *backpropagation* é feito, tal que agilize o aprendizado da rede.

Logo após o treinamento da rede, um teste inicial é feito para estimar a porcentagem de acerto da rede a partir dos dados de entrada dos sinais de som, e dos dados de saída para as respostas.

Caso o índice de acertos estiver acima de um limite mínimo, a rede já estará pronta para processar sinais de fala capturados pelo microfone, e fornecer os vetores de saída corretos para cada palavra.

Dado que os vetores de saída nunca fornecem exatamente o mesmo valor dos centros de resposta, a comparação com o valor correto é feita através da distância euclidiana: dada um vetor de resposta gerado, este será representativo da palavra cujo centro de resposta estiver mais próximo.

4.5.7 Transferência de Dados via Redes de Computadores

Para que o jogo fosse compartilhado por vários usuários em tempo real, a partir de diferentes máquinas, e que permitisse a participação cooperativa de cada jogador para atingir o objetivo oferecido pelo sistema, um modelo de comunicação de computadores ligados em rede foi feito.

O modelo leva em consideração os seguintes fatores: O protocolo de comunicação utilizado, a quantidade de computadores que poderiam se comunicar durante o jogo, como estão ligados estes computadores e como se dará a transferência de dados estáticos e a atualização das propriedades dinâmicas do mundo virtual.

Os protocolos de comunicação utilizados são o *TCP/IP* (*Transmission Control Protocol / Internet Protocol*) (HUNT, 1998), e o *UDP* (*User Datagram Protocol*) (POSTEL, 1980).

A quantidade de computadores testados para a comunicação foi de somente cinco. As arquiteturas de comunicação utilizadas são a multiponto e a ponto a ponto, ligados a uma rede local, não sendo necessário portais, e a atualização de propriedades utilizada neste trabalho foi a atualização por tempo (HUNT, 1998).

No protótipo implementado, após a criação do Ambiente Virtual por um determinado servidor, este fica disponível para os clientes, iniciando o processo de transferência de dados estáticos (geometria, cores e textura do Ambiente Virtual) para cada máquina simultaneamente. Durante este processo, a confiabilidade dos dados transmitidos é indispensável, necessitando o protocolo *TCP/IP* com comunicação multiponto.

Após a transferência dos dados estáticos, inicia-se o jogo de corridas propriamente dito. Aqui os dados são efêmeros, não causando grande prejuízo se uma parte deles se perder durante a sua transferência, mas a velocidade de transferência é muito importante, para que os usuários participantes percebam as mudanças dos elementos dinâmicos (caracteres de vida artificial e outros adversários da rede) em tempo real. Aqui o protocolo de redes *UDP* foi usado.

4.5.7.1 Transferência de Dados Estáticos

Para que os vários clientes participem do mesmo ambiente em tempo real, este deve ser transferido a partir do servidor antes de começar o jogo. Durante o período de implementação, três métodos diferentes de transferência de dados

estáticos foram experimentados: 1 – Transferência de dados com base em ciclos. 2 – Transferência através de blocos de pedido de tamanho constante. 3 – Transferência através de blocos de pedidos de diferentes tamanhos e tipos.

1 – Transferência com base em ciclos: Inicialmente os dados são agrupados em um bloco único e contínuo com um determinado tamanho. Após a conexão *TCP/IP* entre um cliente e o servidor, ambos possuem contadores que são incrementados até que o tamanho total do bloco único seja alcançado.

Após cada incremento uma cópia do pedaço do bloco é transferida do servidor para somente um cliente, que captura o bloco enviado correspondente ao mesmo valor do contador.

Este método é sujeito a erros, pois o ciclo pode variar no tempo de execução, dependendo da velocidade de processamento da máquina. Cada máquina cliente deve esperar que os dados sejam transmitidos em sua totalidade para o cliente que estiver sendo atendido antes.

O bloco único onde são copiados os dados ocupa uma memória adicional que deve ser liberada após todas as transferências.

2 – Transferência através de blocos de pedidos de tamanho constante: No servidor, todos os dados são agrupados em um bloco único. Após o agrupamento, permite-se a cada cliente iniciar a seqüência de pedidos de cada pedaço de bloco de tamanho constante.

O servidor identifica o autor do pedido e envia a este o bloco da posição requisitada. Este método permite que qualquer cliente peça a seqüência de

pedidos a qualquer momento antes de iniciar o jogo, sem causar envios errados de dados. O bloco único de memória deve ser liberado antes de iniciar o jogo.

3 – Transferência através de pedidos de tipo e tamanho diferentes: Aqui a formação de um bloco único não é mais usada, economizando memória. Cada cliente faz o pedido do tipo que deseja transferir, e o servidor devolve como resposta o tamanho total do tipo ou sub-tipo de dado pedido.

Cada cliente pode pedir, então, o bloco da posição de memória especificado, até que o tamanho total deste tipo seja alcançado, e a partir daí iniciar o pedido de informações sobre o próximo tipo, até que a transferência de dados estáticos esteja completa.

Este método é o melhor dos três, pois poupa memória, e permite a qualquer cliente pedir qualquer tipo de bloco de dados a qualquer momento antes do jogo. É um método flexível, servindo para qualquer tipo de dado.

4.5.7.2 Transferência de Dados Dinâmicos

Dados dinâmicos se caracterizam por se alterarem com o tempo.

No protótipo, existem três tipos de dados dinâmicos: dados de clima, dados de posição de veículos e dados de posição de indivíduos simulando vida artificial.

Um dado de clima é uma variável de um *byte* indicando um número inteiro. Dados de posição são compostos de quatro variáveis: uma variável de um *byte* indicando a direção para onde o veículo está apontando, e três variáveis de dois *bytes* indicando as posições horizontal e vertical e a profundidade do elemento. Os dados de posição dos indivíduos de vida artificial são os mesmos dados de posição do veículo.

Para a transferência de dados dinâmicos, dois requisitos foram necessários: que os dados ocupem o menor espaço possível, e que possam ser identificados.

Apesar de os dados a serem transferidos terem tamanhos diferentes, eles são alocados em blocos de mesmo tamanho (tipos *union*, de programação em C), para dispensar protocolos complexos de comunicação.

A transferência de dados dinâmicos é feita ponto a ponto usando protocolo UDP.

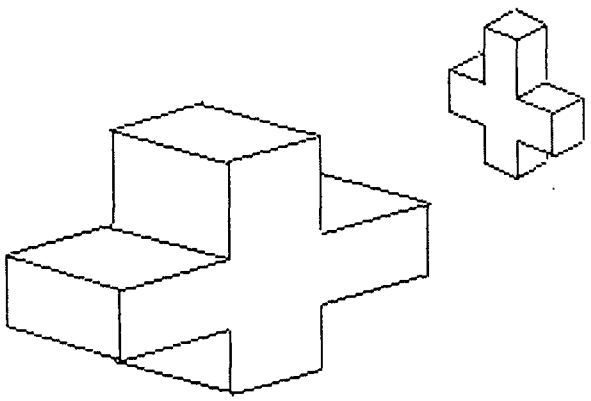
4.5.8 Geração Pseudo-Randômica Paramétrica de Ambientes Virtuais

Foram criados alguns métodos randômicos para criar modelos tridimensionais de objetos, ambientes e texturas, que tentam simular as situações visuais e físicas oferecidas pela natureza.

4.5.8.1 Modelos de Objetos

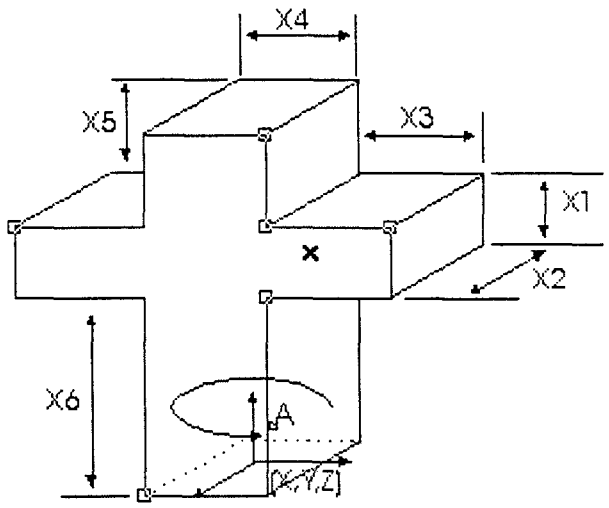
A geração aleatória de mundos tridimensionais se consiste em criar objetos parametrizados cujas coordenadas são geradas a partir de números pseudo-aleatórios produzidos pela própria máquina. Alguns exemplos de objetos parametrizados são dados nas Figuras 29 e 30.

Fig. 29: Exemplos de objetos com diferentes valores de parâmetros.



Cada objeto – inclusive mapas *raster* – dentro do ambiente pode então ser representado através de

Fig. 30: Exemplos de parâmetros geométricos para um objeto.



um vetor indicando o tipo de objeto que representa e as coordenadas correspondentes, entre elas a posição e orientação no espaço, dimensões características, cores e texturas de superfície.

A geração de números aleatórios é limitada por valores máximos e mínimos nas dimensões correspondentes de cada objeto. As limitações impostas são cumulativas.

Os objetos implementados foram: um terreno de malha quadrada com alturas pseudo-randômicas, árvores, pedras, paredes (posteriormente retirados), uma pista de corrida de veículos e um rio (também retirado). Seus modelos serão descritos nos ítens seguintes.

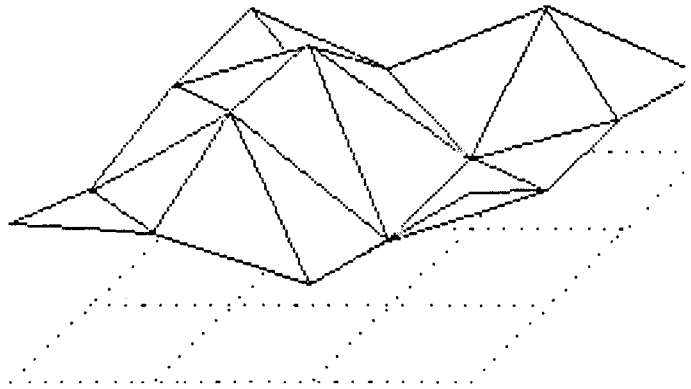
4.5.8.2 Modelos de Terreno

4.5.8.2.1 Modelo Genérico e o Muro

Especial atenção é dada a um objeto que serve como referência para todos os outros objetos do jogo: o terreno. É sobre ele que todos os objetos e jogadores vão se posicionar.

O protótipo projetado para o terreno é uma malha quadrada de pontos cotados, onde um conjunto de três pontos não colineares e adjacentes irá formar um plano inclinado (Fig. 31).

Fig. 31: Superfície representativa do terreno gerado.



Somente a existência do terreno não garante uma imersão total dentro do mundo virtual. Ao se aproximar dos extremos deste mundo, o jogador nota o imenso vazio que o cerca, percebendo que está num mundo virtual.

Para solucionar em parte esta imperfeição, um muro foi criado, tal que não permita ao usuário ver o vazio.

4.5.8.2.2 O Platô

Um modelo derivado da malha quadrada é o terreno composto de planos e platôs. Segundo o dicionário Aurélio da Língua Portuguesa um platô significa planalto. Planalto é uma "grande extensão de terreno plano ou pouco ondulado, elevado, cortado por vales nele encaixados".

Para visualizar o platô em três dimensões, um modelo matemático com simetria radial foi criado, utilizando uma função sigmoidal (FAUSSET, 1994 e HAYKIN, 1998) como função de perfil para o platô, por ser uma função que possui duas propriedades que a fazem adequada para este perfil:

Dada a função sigmoidal

$$f(x) = \frac{1}{1+e^{-\sigma x}} \quad (1)$$

Tem-se que:

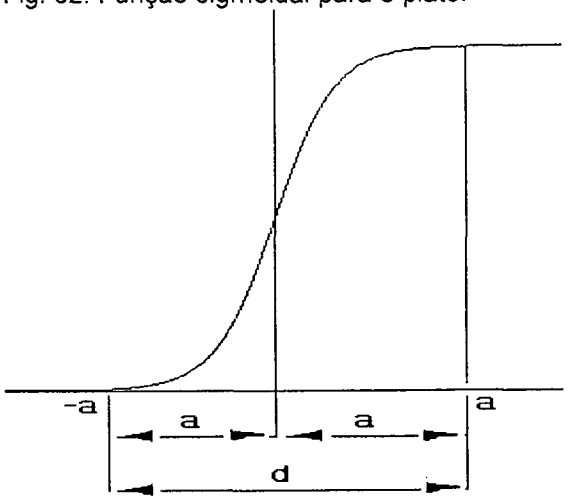
$$a) \lim_{x \to \infty} f(x) = 1$$

Ou seja, quanto maior o valor de x, mais próxima da unidade a função estará. Este valor pode ser considerado constante para um alto valor de x.

$$b) \lim_{x \to -\infty} f(x) = 0$$

Ou seja, quanto mais baixo for o valor de x, mais próxima de zero a função será.

Fig. 32: Função sigmoidal para o platô.



A dedução da equação matemática que dá formas a este acidente geográfico segue-se abaixo:

O domínio da função sigmoidal é infinito positivo e negativo. Para ajustar esta função para mundos virtuais, este domínio é limitado simetricamente, ou seja, os dois limites no domínio são **-a** e **a**, onde **a** é um valor escolhido tal que os valores da função **f(-a)** e **f(a)** sejam 0.01 e 1 - 0.01 = 0.99, respectivamente. A distância **d = 2a** é a extensão do declive da sigmoidal (Fig. 32).

$$f(-d/2) = 0.01$$

Tem-se então que

$$\frac{1}{1+e^{-\sigma(-\frac{d}{2})}} = 0.01 \qquad (2)$$

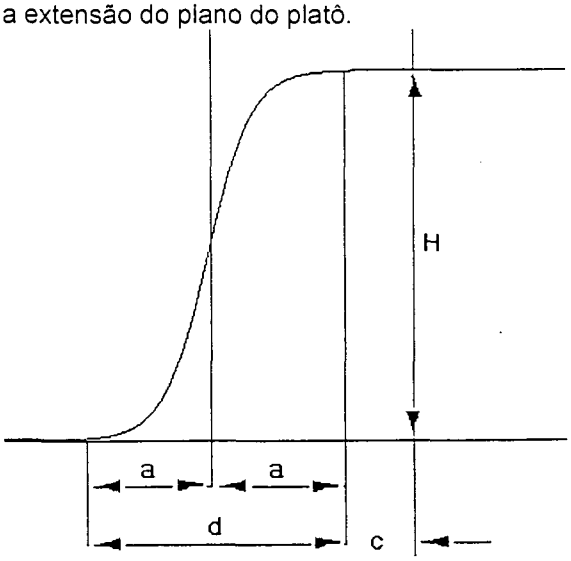
e $\sigma d = 2\ln 99 = k = 9.1902$

Chegando ao mesmo resultado se iniciarmos em:

$$f(\frac{d}{2}) = 0.99$$

Fazendo-se um deslocamento de eixos para que o topo da função possua uma extensão determinada **c** (Fig. 33):

Fig. 33: Função sigmoidal deslocada para dar a extensão do plano do platô.



$$f(x) = \frac{1}{1+e^{-\sigma(x+\frac{d}{2}+c)}} \qquad (3)$$

Para eliminar a constante desconhecida σ, multiplica-se e divide-se o expoente logaritmo por d:

$$f(x) = \frac{1}{1+e^{-\sigma d(\frac{1}{2}+\frac{(c+x)}{d})}} \qquad (4)$$

Sabe-se que **σd = K = 9.1902**, e considerando a função **f(x)** como a função que delimita o perfil do platô, e ainda que este é radialmente simétrico, substitui-se **x** por um raio **r**.

A função é multiplicada por uma altura constante **H**. Assim sendo, tem-se a equação do platô, que é assim definida:

Dado o platô de altura máxima **H**, raio do plano superior **c**, e extensão de declive **d**, a função de superfície com simetria radial **P(r)** que rege a sua forma física é dada por:

$$P(r) = \frac{H}{1 + e^{-K(\frac{1}{2} + \frac{(c+r)}{d})}}$$ Se **c** ≤ **r** ≤ **d** + **c**; (5)

$$P(r) = H$$ Se **r** < **c**; (6)

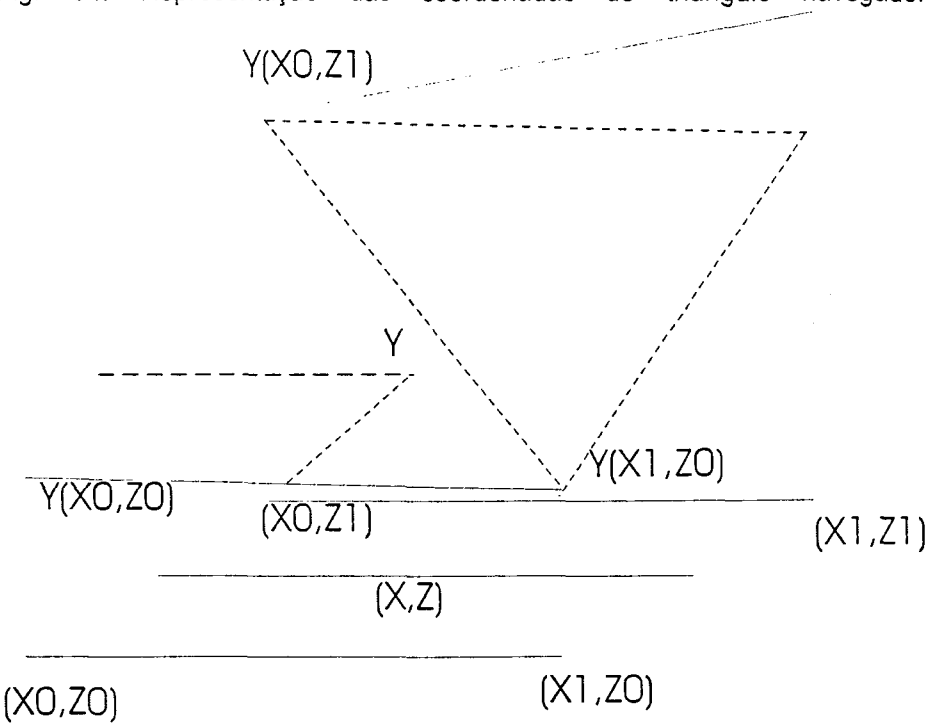
$$P(r) = 0$$ Se **r** > **d** + **c**; (7)

Onde **K** é uma constante valendo **9.1902**.

4.5.8.2.3 Posicionamento e Orientação Geométrica de Objetos Baseados na Topologia do Terreno

Para garantir um posicionamento dos objetos e jogadores sobre a superfície do terreno, o algoritmo seguinte é usado: dependendo do triângulo onde o objeto se encontra, a equação do plano que contém este triângulo é encontrada em função de seus três pontos determinantes (Fig. 34), sendo a altura uma função planar das posições horizontal e longitudinal (***y=ax+bz+c***).

Fig. 34: Representação das coordenadas do triângulo navegado.

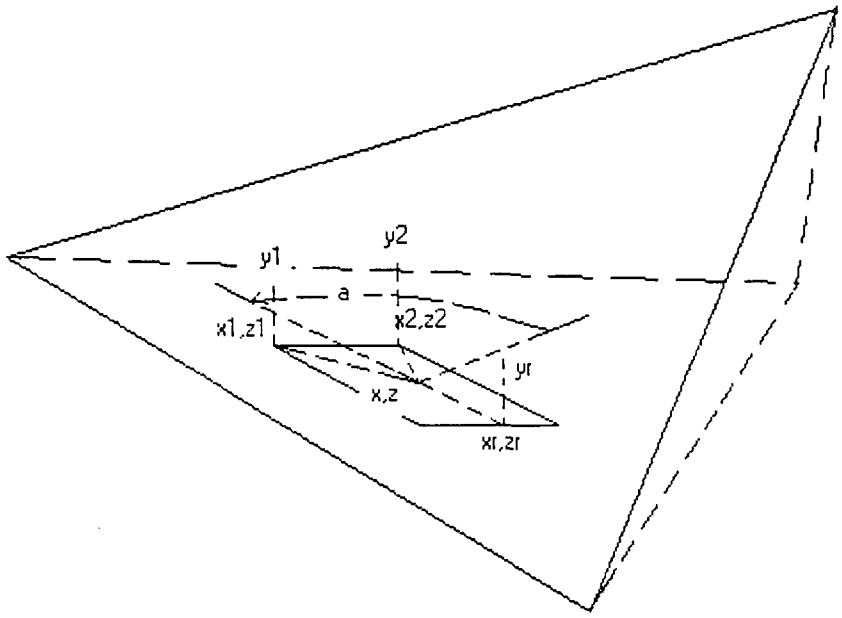


Sabe-se a posição dos objetos e dos usuários, tem-se então a altura onde estes se encontram após a aplicação da equação acima.

Para que fique orientado de acordo com a inclinação do plano em que se encontra, o avatar sofre dois giros. O primeiro ao redor do eixo vertical, definido pelo usuário através da direção que este toma durante a navegação pelo mundo. O segundo giro é feito ao redor do vetor que indica a frontal do *avatar.*

No modelo o *avatar* é considerado um triângulo cuja posição de seus vértices é calculada com o método de posicionamento descrito acima. Seu ângulo de giro ao redor do eixo vertical é **a**, e tem-se também a posição **p(x,z)** do centro do *avatar*.

Fig. 35: Geração das alturas correspondentes do triângulo representante do avatar.



Através de parâmetros fixos, tem-se as posições **p1(x1,z1)** e **pr(xr,zr)** (conjunto de equações 8 ao lado), e para cada par de posições horizontais, calcula-se suas alturas correspondentes (Fig. 35).

$$x1 = x - dx.\cos(a) - dz.\mathrm{sen}(a)$$
$$z1 = z + dx.\mathrm{sen}(a) - dz.\cos(a)$$
$$y1 = y(x1, z1)$$

$$x2 = x + dx.\cos(a) - dz.\mathrm{sen}(a)$$
$$z2 = z - dx.\mathrm{sen}(a) - dz.\cos(a)$$
$$y2 = y(x2, z2)$$

$$xr = x + dx.\mathrm{sen}(a)$$
$$zr = z + dz.\cos(a)$$
$$yr = y(xr, zr)$$

Daí tem-se os pontos **p1**,**p2** e **pr** não colineares e conseqüentemente o vetor **Nf** normal ao plano que passa por estes pontos (Fig. 36).

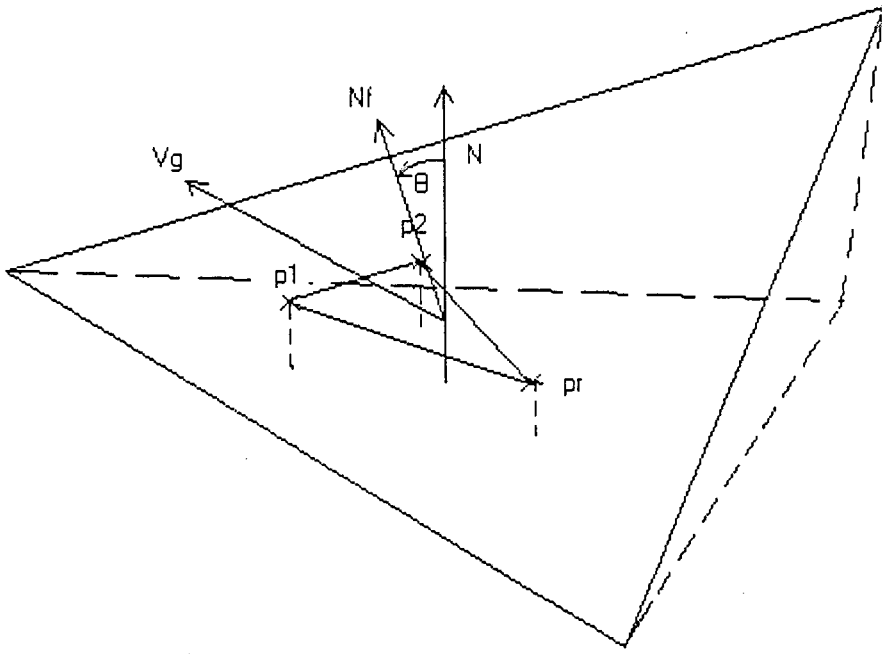
Define-se o vetor **N** como sendo (0,1,0), ou seja, um vetor vertical apontando para cima.

Os vetores **p1**, **p2** e **pr** definem o vetor normal **Nf** :

Nf = (**pr** - **p2**) **X** (**p2** - **p1**); (9) onde **X** é o produto vetorial.

Calcula-se então o vetor **Vg** frontal ao redor do qual o avatar gira pela segunda vez, e o ângulo de giro θ. A seqüência de cálculo é mostrada abaixo.

Fig. 36: Cálculo do vetor frontal **Vg** do avatar e do ângulo θ de giro.



O vetor frontal **Vg** para o segundo giro é calculado:

$$\mathbf{Vg} = \mathbf{N} \times \mathbf{Nf} = (0,1,0) \times \mathbf{Nf} = (Nfz,0,-Nfx); \qquad (10)$$

O ângulo de giro θ é calculado como abaixo:

$$\theta = \cos^{-1}\left(\frac{N.N_f}{N.N_f}\right) = \cos^{-1}\left(\frac{N_{fy}}{1.N_f}\right) \qquad (11)$$

Onde o operador (.) (ponto) é o produto escalar de dois vetores e o operador |.| é o operador módulo de um vetor.

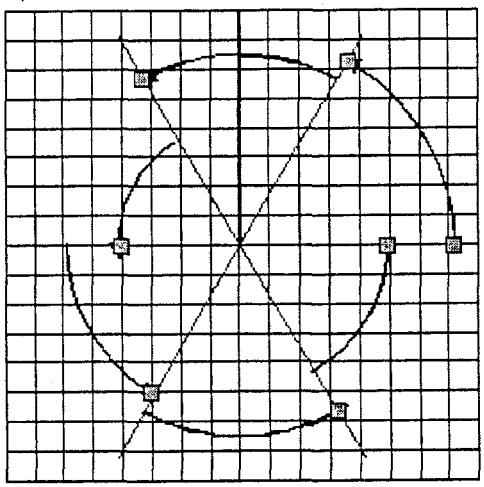
Após esta seqüência de cálculo o avatar é girado de um ângulo θ ao redor do vetor **Vg**. Ajustando-se à inclinação do terreno.

4.5.8.2.4 Geração Pseudo-Randômica de Paredes

A maioria dos objetos tem seu posicionamento baseado em somente um ponto no espaço. O uso de matrizes de valores de ponto flutuante permitiu a criação de técnicas de geração aleatória de paredes, ou seja, uma parede é armazenada em uma matriz de dimensões **2 X n**, que representa as dimensões **x** e **z** dos vértices da parede.

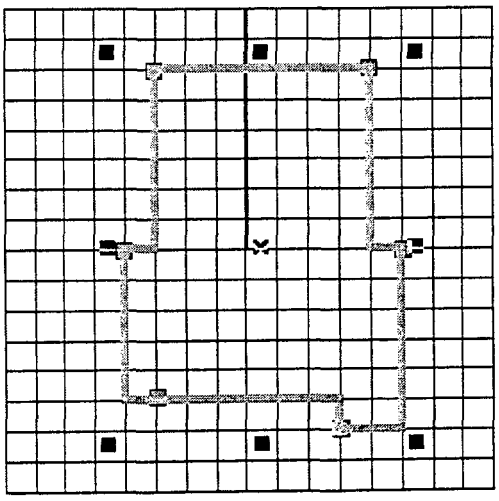
Paredes aleatórias podem ser usadas para uso em construções virtuais, tais como casas, prédios, depósitos, hangares, labirintos, acidentes geográficos, entre outros.

Fig. 37: Início do processo de geração de paredes.



Dentro do contexto do trabalho, uma parede é criada a partir de um sistema de coordenadas polares bidimensionais. Uma variável aleatória e um contador angular são usados para gerar os pontos no plano. O contador faz a volta completa de um círculo a intervalos constantes enquanto a variável aleatória recebe valores de raio dentro de um intervalo específico (Fig. 37).

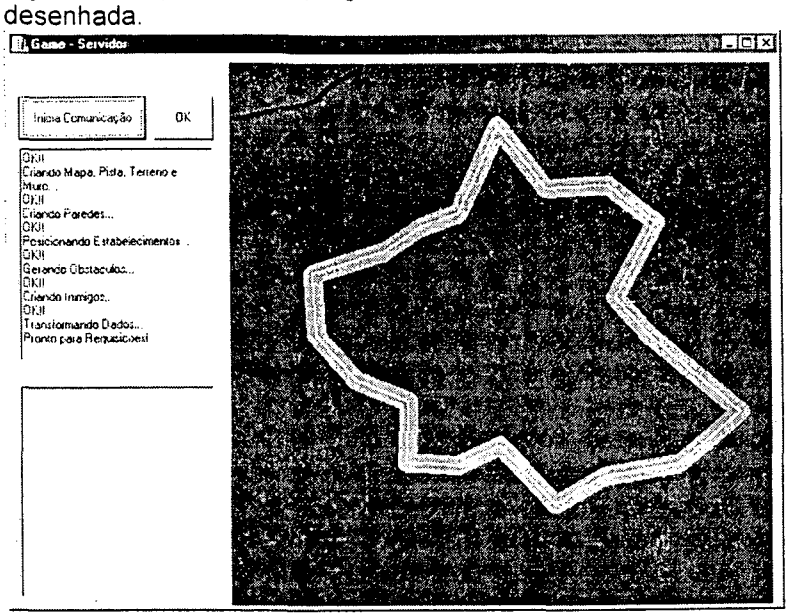
Fig. 38: Vista superior da parede aleatória após o pré-processamento.



Após a geração pseudo-randômica, os pontos passam por um pré-processamento para que as paredes fiquem em ângulo reto, sem repetições de pontos, extensões ou ciclos (Fig. 38).

4.5.8.2.5 Geração Pseudo-Randômica da Pista de Corrida

Fig. 39: Tela principal do programa servidor criador, com a pista desenhada.



A pista de corrida é gerada pelo mesmo método usado na geração das paredes, com a diferença de que não é usado pré-processamento para tornar os ângulos dos vértices retos. Os vértices são unidos através de linhas com grande espessura para dar suavidade nestes pontos (Fig. 39).

4.5.8.2.6 Geração Pseudo-Randômica de Rios

Para gerar os rios com curvaturas suaves, o posicionamento de seus *pixels* foi feito através de uma função composta de duas sub-funções. Dado um contador **i** no sentido vertical do mapa, existe uma variável **x(i)=*rand*[-x1;x1]**, onde **x1** é um número pequeno, e ***rand*[]** é uma função que gera um número pseudo-aleatório entre o intervalo dado, uma segunda variável **sx(i)** é o somatório dos valores de x, e uma terceira variável **ssx(i)** que é o somatório de **sx(i)**, ou seja:

$$sx(i) = \sum_{0}^{i} x_i \qquad (12)$$

$$ssx(i) = \sum_{0}^{i} sx(i) \qquad (13)$$

$$ssx(i) = \sum_{0}^{i} \sum_{0}^{i} x(i) \qquad (14)$$

4.5.9 Uso de Texturas

Segundo (HUONG *et al*, 1998) as texturas são uma das ferramentas usadas para dar a imersão no mundo virtual criado. LUBAN (2001) ao fornecer algumas dicas para a produção de jogos, propõe a criação de ambientes naturais que sejam os mais imprevisíveis que se possa criar.

Com base nestas duas observações, um tratamento especial foi dado ao trabalho de criação e tratamento de texturas e sobre o seu gerenciamento pela placa aceleradora gráfica.

O Anexo III mostra alguns exemplos de como otimizar o uso de texturas.

Uma segunda utilidade foi dada às texturas: o de conter um mapa *raster* do ambiente gerado, possibilitando a prevenção de colisão de corpos com rápido processamento. Tais ítens são descritos logo a seguir.

4.5.9.1 Criação de Texturas Usando *Hardware e Software* Específicos

Texturas são indispensáveis no efeito estético de jogos tridimensionais. Elas são responsáveis por dar aparência mais realística aos objetos que recobrem, aumentando a imersividade do ambiente (HUONG *et al*, 1998).

Graças ao recurso de texturas, detalhes tridimensionais de rugosidade podem ser eliminados havendo um ganho na velocidade de processamento gráfico no jogo.

A atividade de criação e modelagem de texturas é facilitada em todo o seu processo graças à informática. Texturas podem ser criadas de duas formas: ou desenhadas, ou capturadas de fontes naturais e transferidas para meio digital.

Dois meios digitais de captura são utilizados, de forma barata e rápida: com um *scanner* de mesa, onde os desenhos ou fotos devem estar disponíveis no local onde o *scanner* se encontra, ou com uma câmera fotográfica digital, com a qual o profissional captura as texturas a partir de fotos tiradas *in natura*, e pode passá-las para o microcomputador em arquivos de formato conhecido (*BITMAP* ou *JPEG*) (PANDYA & MACY, 1996).

Após a captura, as texturas podem ser editadas com inúmeros efeitos disponíveis em *software* para edição de imagens, tais como efeitos *neon*, transparência, riscos de grafite, granulosidade, envelhecimento, descoloração, mancha, distorção por lente, entre outros. A *Internet* é um meio riquíssimo em exemplos de texturas.

4.5.9.2 Detalhe da Textura *versus* Quantidade de Memória Usada

Placas aceleradoras gráficas são usadas para gerenciar as texturas no lugar do computador hospedeiro, poupando-lhe tempo de processamento e memória.

Fig.40: Exemplo de textura otimizada.

Fig.41: Efeito final da textura otimizada.

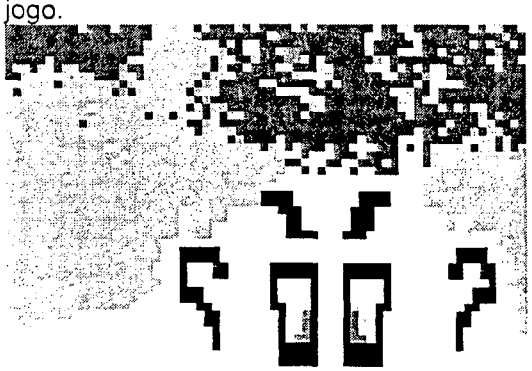
Para que estas placas possam fazê-lo, o computador deve passar a textura – já lida de um arquivo ou criada em tempo real – da sua memória para a memória da placa aceleradora.

A placa aceleradora também possui um limite de memória para texturas e objetos tridimensionais, que não deve ser ultrapassado.

Jogos possuem uma imersividade maior quando oferecem texturas mais detalhadas e, conseqüentemente de maior tamanho, o que leva a ocupar mais memória da placa aceleradora gráfica.

Para evitar tal problema, um método bastante simples é muito usado para reduzir tamanho de memória usada: as texturas de várias partes de um mesmo objeto são colocadas todas juntas em um mesmo arquivo *raster*. Um exemplo de arquivo *bitmap* com texturas comprimidas, com uma foto do humanóide onde é usada, são dados nas Figuras 40 e 41.

Fig.42: Textura otimizada para um *bot* do jogo.



Uma tentativa de otimização de texturas foi usado para dar expressão à face de um personagem que poderia ser usado durante o jogo. A textura original é dada na Figura 42 ao lado.

4.5.9.3 Textura Usada no Terreno

A textura colocada no terreno é um *bitmap* de extensões moderadas (128 X 128 pontos ou menos), também gerado pseudo-aleatoriamente através de coloração RGB como solo básico.

4.5.10 Detecção de Colisão Feita por Mapas *Raster*

Um método de prevenção de colisões foi implementado usando texturas: A função de colisão é feita com um mapa *raster* mostrando a planta baixa reduzida do mundo virtual.

Durante o processo de posicionamento aleatório dos objetos, as posições tridimensionais respectivas são transformadas em coordenadas bidimensionais inteiras, que são indicadas por um ponto (indicador de colisão) de cor específica dentro deste mapa *raster*.

Para o caso de corpos de grandes dimensões (paredes, por exemplo) a projeção do obstáculo é preenchida com pontos com a cor específica, indicando que é uma região de colisão.

A existência deste ponto no mapa *raster* comunica ao algoritmo que, naquela posição equivalente no mundo tridimensional, existe um objeto (árvores ou pedras, por exemplo).

Após esta transformação, o algoritmo verifica se, na posição para onde o usuário deseja ir, existe um ponto indicador de colisão que representa um dos objetos do ambiente (uma pedra, árvore ou um limitador da pista de corrida). Se houver, o jogador não executa o movimento.

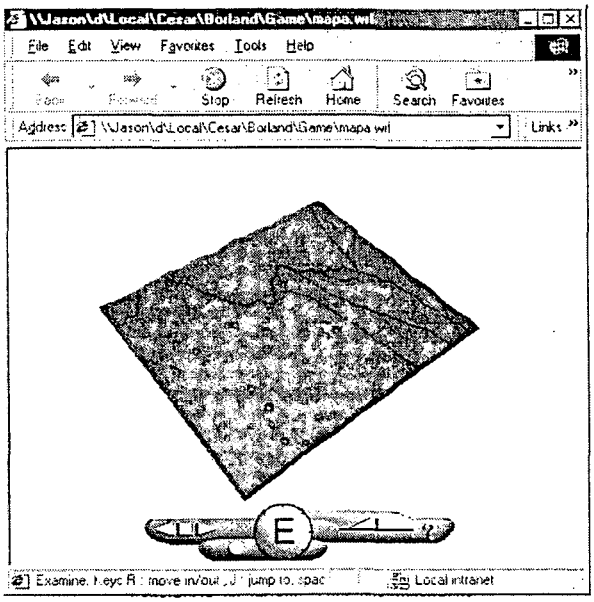
Citando outros métodos, a linguagem *VRML* usa um método de detecção de colisão considerando o usuário no mundo navegado como um cilindro com altura e diâmetro definidos, e demais objetos sendo revoltos por cônicas, tais como cilindros ou paralelepípedos.

Em (REBELO, 1998) a detecção de colisão é feita considerando um setor de cilindro com sessenta graus à frente do usuário, com um diâmetro e altura determinados.

Uma função de transformação de coordenadas e verificação de pontos através de cores em mapas *raster* funciona mais rápido do que as duas anteriores, que fazem intersecção de superfícies, embora seja menos preciso e ocupe maior quantidade de memória.

4.5.11 Reprodução de Ambiente Gerado em VRML

Fig. 43: Mapa tridimensional do mundo criado, visto em *VRML*.



No protótipo, após a criação do ambiente, com seu terreno e objetos, um arquivo *VRML* é criado (Fig. 43), com o intuito de permitir aos usuários fazerem uma verificação do ambiente gerado, e um planejamento de trajetórias a serem seguidas durante a corrida.

4.5.12 Aplicação de Modelos de Navegação em Ambientes Virtuais com *Hardware* Periférico para Interface Homem-Máquina

Locomoção do Usuário: Modelos Físicos Discretizados

Na implementação do protótipo do jogo de corridas, algumas características físicas do modelo real de um automóvel guiado por um volante foram adotadas:

- Caso o veículo estiver em movimento, com o volante girado de um ângulo qualquer, se o usuário solta o volante, este retorna à posição de equilíbrio. O carro então volta a andar em linha reta;
- Aperto do acelerador faz com que o veículo ande a uma velocidade crescente até um limite máximo;
- Relaxamento do acelerador faz com que o veículo permaneça à velocidade constante;

- A influência do atrito sobre as rodas do veículo faz com que este abaixe a sua velocidade até zero, ou que sofra uma redução em sua velocidade final;
- Quanto maior o giro do volante, medido a partir do centro, menor o raio da curva feita pelo veículo, quando em movimento;
- Se o volante for girado quando o carro estiver parado, o carro não gira.

Assim, para a inclusão destas características físicas utilizando uma interface digital específica, parte-se de um modelo físico sobre o movimento retilíneo uniformemente acelerado, ajustando-o para um sistema de coordenadas polares, de forma discretizada. O modelo físico é explicado abaixo. Finalmente uma tabela comparativa da implementação do modelo dependendo da interface digital utilizada é mostrada.

4.5.12.1 Modelo Físico Discretizado do Movimento Retilíneo uniformemente Acelerado

Para o computador, o tempo transcorre de modo discreto através de pulsos, onde cada pulso é emitido depois de um determinado intervalo de tempo.

Em termos de programação algorítmica, consegue-se fazer passar o tempo na forma de contadores, que executam funções pré-programadas em períodos de tempo determinados e constantes.

Assim, os modelos físicos contínuos que representam a realidade são adaptados ao funcionamento da máquina, ou seja, de forma discretizada.

4.5.12.2 O Modelo do Movimento Não-Retilíneo e Não-Uniforme

No jogo, tal modelo é usado para possibilitar a navegação do jogador no ambiente virtual criado. A partir deste modelo simples, se faz variar o vetor velocidade em módulo e direção através das interfaces com o usuário.

Sendo o modelo físico contínuo para o cálculo da posição em um movimento não retilíneo e não uniforme, dada a velocidade de um ponto em função do tempo:

$$\vec{s} = \vec{s}_0 + \int_0^{t_{max}} \vec{V}(t)dt \qquad (15)$$

Discretizando a equação:

$$\vec{s} = \vec{s}_0 + \sum_{i=1}^{N} \vec{V}(i).\Delta t \qquad (16)$$

$$N = \frac{t_{max} - 0}{\Delta t} \qquad (17)$$

Onde Δt tem um valor constante por conveniência, e ***i*** indica a situação atual depois de ***i*** períodos de tempo Δt.

Então:

$$\vec{s} = \vec{s}_0 + \vec{V}(1).\Delta t + \vec{V}(2).\Delta t + ... + \vec{V}(N).\Delta t \qquad (18)$$

Se enumerarmos as posições ***s***, temos:

$$\vec{s}_1 = \vec{s}_0 + \vec{V}(0).\Delta t$$
$$\vec{s}_2 = \vec{s}_1 + \vec{V}(1).\Delta t$$
$$\ldots$$
$$\vec{s}_{i+1} = \vec{s}_i + \vec{V}(i).\Delta t \quad (19)$$

A última equação é a equação discretizada para o movimento não retilíneo e não uniforme. s_i é a posição do ponto na situação atual, s_{i+1} é a posição na situação imediatamente posterior à situação atual, Δt é o intervalo de tempo constante entre o tempo atual e o imediatamente posterior, e **V(i)** é a velocidade do ponto, que pode variar tanto em função dos valores lidos pela interface do usuário quanto em função de fenômenos físicos programados, como será visto mais adiante.

4.5.12.3 O Modelo da Velocidade Não Uniforme

Assim como a variação da posição de um ponto é computada de forma discreta, a partir de um modelo contínuo, a variação da velocidade também é tratada desta mesma forma.

Dos pontos de vista algorítmico e físico, a variação da velocidade de um ponto deve-se a dois motivos: um é inerente aos fenômenos físicos simulados no ambiente virtual, tais como aceleração da gravidade, atrito, resistência do ar,

resistência oferecida por fenômenos climáticos, tais como chuva, granizo, vendavais, buracos no solo, elevações ou declives, entre outros.

O outro motivo é o controle que o usuário possui sobre esse ponto, ou do objeto que é representado por esse ponto, através da interface com a máquina, simulando o volante e o motor de um automóvel, o manche e as turbinas de um avião ou o timão e o motor de um barco ou lancha.

Estes exemplos mostram que o usuário pode controlar tanto a direção para onde o ponto está indo quanto a velocidade com que este ponto vai nesta direção.

Matematicamente falando, usuário e natureza (ambiente virtual) – representada de início pelas suas ações atenuadoras – podem controlar o vetor velocidade, tanto em direção quanto em módulo.

Assume-se então que a velocidade, da forma contínua, varia a partir de uma velocidade inicial incluindo a ação da soma de todas as acelerações num dado instante:

$$\vec{V} = \vec{V}_0 + \int_0^{t_{\max}} \vec{\alpha}(t)dt \qquad (20)$$

Onde a aceleração resultante $\alpha(t)$ é a soma de todas as acelerações atuantes no tempo t:

$$\vec{\alpha}(t) = \vec{\alpha}_{usuario}(t) + \vec{\alpha}_{atrito}(t) + \vec{\alpha}_{ar}(t) + \vec{\alpha}_{terreno}(t) + ... + \vec{\alpha}_n(t) \qquad (21)$$

Discretizando a equação da velocidade:

$$\vec{V} = \vec{V}_0 + \sum_{i=1}^{N} \vec{\alpha}(i)\Delta t \qquad (22)$$

Onde N, Δt e ***i*** são os mesmos usados nas equações 15 a 19 do modelo para Movimento Não-Retilíneo e Não-Uniforme. Tem-se então que:

$$\vec{V} = \vec{V}_0 + \vec{\alpha}(1)\Delta t + \ldots + \vec{\alpha}(N)\Delta t \quad (23)$$

Enumerando as velocidades ***V(i)*** tem-se:

$$\vec{V}_1 = \vec{V}_0 + \vec{\alpha}(0).\Delta t$$
$$\vec{V}_2 = \vec{V}_1 + \vec{\alpha}(1).\Delta t$$
$$\ldots \quad (24)$$
$$\vec{V}_{i+1} = \vec{V}_i + \vec{\alpha}(i).\Delta t$$

A última equação é a equação discretizada para a velocidade não uniforme.

Finalmente, colocando as duas equações discretizadas, velocidade e posição, tem-se a forma algorítmica para o cálculo atualizado da posição a intervalos de tempos constantes:

Para o tempo de índice ***i***, faz-se:

$$\vec{\alpha}(i) = \vec{\alpha}_{usuario}(i) + \vec{\alpha}_{atrito}(i) + \vec{\alpha}_{ar}(i) + \vec{\alpha}_{terreno}(i) + \ldots + \vec{\alpha}_n(i) \quad (25)$$

$$\vec{V}_{i+1} = \vec{V}_i + \vec{\alpha}(i).\Delta t \quad (26)$$

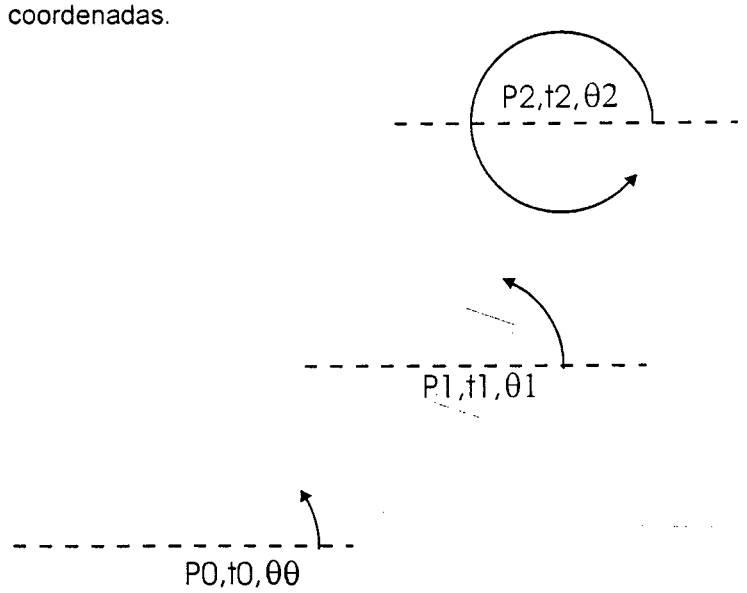
$$\vec{s}_{i+2} = \vec{s}_{i+1} + \vec{V}(i+1).\Delta t \quad (27)$$

4.5.12.4 O Modelo da Curva do Veículo no Plano de Coordenadas Polares

Considerando o movimento do veículo no sistema plano de coordenadas polares, observa-se que seu movimento, para cada intervalo de tempo Δt, depende de dois vetores e um valor em módulo:

A posição atual do veículo p(θ,r), do módulo da velocidade atual V(t), e do vetor da direção atual do veículo d(θ,r).

Fig. 44: Posição e direção do veículo descrito através de três coordenadas.



Simplificando o vetor direção, onde o raio tem módulo unitário, este pode ser descrito apenas pelo seu ângulo em relação ao eixo de referência do plano. Assim, pode-se descrever a posição e a direção do veículo no tempo t em termos de p, θ e t (Fig. 44).

Por conveniência, transforma-se a posição do veículo para o sistema cartesiano plano de coordenadas:

$$\vec{s}_i(x,y) \equiv \vec{p}(\theta,r)$$

Porém, permanece o sistema polar de coordenadas para a direção que a frente do veículo está apontando, e calcula-se o incremento da posição do veículo a partir do módulo da sua velocidade atual, do intervalo de tempo Δt e das funções trigonométricas básicas de transformação de coordenadas:

$$\vec{d}_i(x,y) = V.\Delta t.(\cos\theta, \operatorname{sen}\theta) \quad (28)$$

Assim, se **Si** for incrementado de **di** para cada intervalo de tempo Δt tem-se que:

$$\vec{s}_i = \vec{s}_{i-1} + \vec{d}_{i-1} \quad (29)$$

$$\vec{s}_i = \vec{s}_{i-1} + V_{i-1}.\Delta t.(\cos\theta_{i-1}, \operatorname{sen}\theta_{i-1})$$

que é semelhante à equação (27), sendo a equação para atualizar a posição **Si** do veículo, no tempo de índice **i**.

Daqui observa-se que o ângulo θ indica a direção para o qual o veículo se desloca e também a direção para onde o veículo está apontando.

4.5.12.5 Inclusão de Fenômenos Físicos

Tal como em (23) e (24), o vetor velocidade da equação (26) também varia com os vetores de aceleração provocados pelos fenômenos naturais. Para simplificar o modelo, seus valores são tomados somente em módulos, ou seja:

$$\alpha(i) = \alpha_{usuario}(i) + \alpha_{atrito}(i) + ... + \alpha_n(i)$$

$$V_{i+1} = V_i + \alpha(i).\Delta t$$

$$\vec{s}_{i+2} = \vec{s}_{i+1} + V_{i+1}.\Delta t.(\cos\theta_{i+1}, \operatorname{sen}\theta_{i+1})$$

4.5.12.6 Modelos Matemáticos de Navegação com Hardware de Realidade Virtual: Algoritmos

Quatro tipos de *hardware* para RV foram usados, com implementação de algoritmos para os seus respectivos modelos.

Para modelar as equações foi necessária uma análise inicial de como cada *hardware* específico forneceria seus valores durante a navegação, e posteriormente qual a variável dentro do modelo da curva do veiculo seria modificada pela atuação deste hardware.

Inicialmente, para o modelo foram consideradas oito variáveis:

- A posição do automóvel;
- Ângulo de direção do veículo em relação a um eixo de referência;
- A velocidade do veículo;
- A aceleração do veículo;
- A força de atrito convertida para aceleração;
- A aceleração angular controlada do volante do veículo;
- A velocidade de retorno do volante e
- A aceleração de retorno do volante.

A Tabela 2 mostra uma comparação entre o hardware utilizado, seu modo de controle, limite de valores fornecidos pelo *hardware*, combinações utilizadas ou não, as variáveis controladas e a função de transferência.

Tabela 2: Comparação de *Hardware* e Algoritmos correspondentes.

HARDWARE	MODO DE CONTROLE	LIMITES	DETALHES DE CONTROLE	VARIÁVEIS CONTROLADAS E CONSTANTES	ALGORÍTMO DA FUNÇÃO DE TRANSFERÊNCIA
TECLADO	Pressionado / solto = binário	0 e 1	As teclas cima/baixo servem de acelerador e de freio, e as teclas esquerda direita controlam o ângulo do volante do veículo	Aceleração do veiculo constante: α = cte; Aceleração do atrito constante: αat = cte < α; Velocidade variável: v(i); Ângulo de direção variável: θ(i); Aceleração angular do volante: αv = cte; Velocidade angular do volante variável: ω = ω(i); Aceleração angular de retorno do volante: αr = cte; Posição variável: p(x,z) = p(x(i),z(i));	se(cima) {v(i) = v(i-1) + α;} se(baixo) {v(i) = v(i-1) - k.α; k<1} v(i) = v(i-1) - αat; se(v(i) < 0) v(i) = 0; se(v(i) > vmax) { v(i) = vmax; } se(esquerda) {ω(i) = ω(i-1) + αv; } se(direita) {ω(i) = ω(i-1) - αv; } ω(i) = ω(i-1) – αr * signal(ω(i-1)); θ(i) = θ(i-1) + ω(i); x(i)=x(i)+ v(i)*cos(θ(i)); y(i)=y(i)+ v(i)*sen(θ(i));

HARDWARE	MODO DE CONTROLE	LIMITES	DETALHES DE CONTROLE	VARIÁVEIS CONTROLADAS E CONSTANTES	ALGORITMO DA FUNÇÃO DE TRANSFERÊNCIA
JOYSTICK	Contínuo horizontal / vertical	0 a 65535 vertical e horizontal	Controle vertical: aceleração / frenagem Controle horizontal: esquerda / direita Não utiliza botões	Aceleração do veículo variável: $\alpha(i) = k * \text{def. vert.(joystick)};$ Aceleração do atrito constante: $\alpha at = cte;$ Velocidade angular variável: $\omega = \omega(i);$ Velocidade variável: $v = v(i);$ Posição variável: $p(x,z) = p(x(i),z(i));$	$\alpha(i) = k *$ deflexão vertical (joystick); $v(i) = v(i-1) + \alpha(i) - \alpha at;$ se$(v(i) < 0)$ $v(i) = 0;$ se$(v(i) > vmax)$ $\{v(i)=vmax;\}$ $\omega(i) = m *$ deflexão horizontal (joystick); $\theta(i) = \theta(i-1) + \omega(i);$ $x(i)=x(i)+ v(i)*\cos(\theta(i));$ $y(i)=y(i)+ v(i)*\text{sen}(\theta(i));$

HARDWARE	MODO DE CONTROLE	LIMITES	DETALHES DE CONTROLE	VARIÁVEIS CONTROLADAS E CONSTANTES	ALGORÍTMO DA FUNÇÃO DE TRANSFERÊNCIA
CYBERPUCK	Contínuo horizontal / vertical	0 a 65535 vertical e horizontal	Controle vertical: aceleração / frenagem Controle horizontal: esquerda / direita Movimenta-se somente com um dos botões pressionado	Aceleração do veículo variável: $\alpha(i) = k * inc.\ vert.(cyber);$ Aceleração do atrito constante: $\alpha at = cte;$ Velocidade angular variável: $\omega(i) = m * inc.\ horiz.\ (cyber);$ Velocidade variável: $v = v(i);$ Posição variável: $p(x,z) = p(x(i),z(i));$	se(botão pressionado) { $\alpha(i) = k *$ inclinação vertical (cyber); $v(i) = v(i-1) + \alpha(i) - \alpha at;$ se(v(i) < 0) v(i) = 0; se(v(i) > vmax) {v(i)=vmax;} $\omega(i) = m *$ inclinação horizontal (cyber); $\theta(i) = \theta(i-1) + \omega(i);$ $x(i)=x(i)+ v(i)*\cos(\theta(i));$ $y(i)=y(i)+ v(i)*\text{sen}(\theta(i));$ }

HARDWARE	MODO DE CONTROLE	LIMITES	DETALHES DE CONTROLE	VARIÁVEIS CONTROLADAS E CONSTANTES	ALGORÍTMO DA FUNÇÃO DE TRANSFERÊNCIA
LUVA + TECLADO	Luva: Continuo com giro ao redor do pulso e giro ao redor do eixo horizontal Abertura e fechamento de um dedo somente Teclado: Pressionado / solto = binário	0 a 255 girando no eixo horizontal 0 a 128 girando ao redor do pulso 0 para o dedo aberto e 255 para o dedo fechado 0 e 1 para o teclado	A luva é usada somente para girar o volante do veículo, quando o usuário fecha os dedos As teclas cima/baixo servem de acelerador e de freio	Aceleração do veiculo constante: α = cte; Aceleração do atrito constante: αat = cte < α; Velocidade variável: v(i); Velocidade angular variável: $\omega(i) = k * inc.\ pulso.\ (luva)$; Posição variável: p(x,z) = p(x(i),z(i));	se(cima) {v(i) = v(i-1) + α;} se(baixo) {v(i) = v(i-1) - k.α; k<1} v(i) = v(i-1) - αat; se(v(i) < θ) v(i) = 0; se(v(i) > vmax) { v(i) = vmax; } se(dedos fechados) { $\omega(i) = k * inclinação\ pulso\ (luva)$; $\theta(i) = \theta(i-1) + \omega(i)$; } x(i)=x(i)+ v(i)*cos($\theta$(i)); y(i)=y(i)+ v(i)*sen(θ(i));

HARDWARE	MODO DE CONTROLE	LIMITES	DETALHES DE CONTROLE	VARIÁVEIS CONTROLADAS E CONSTANTES	ALGORITMO DA FUNÇÃO DE TRANSFERÊNCIA
HMD	A direção da visão utiliza somente dois eixos: Ao redor do eixo vertical e ao redor do eixo horizontal	0 a 65535 ao redor do eixo vertical, para 360° -5500 a 5500 aprox. para giro ao redor do eixo horizontal (50° para cima e 50° para baixo)	Para determinar qual a direção do olho do observador, deve-se definir o ponto de origem do olho do observador O(x,y,z) e o ponto destino para onde se está olhando D(x,y,z)	Ângulo ao redor do eixo vertical: v Ângulo ao redor do eixo horizontal: h	D(x,y,z) = O(x,y,z) + + P(cos v, -sen h, sen v)

4.5.13 Simulação de Fenômenos Climáticos

O tempo é um elemento único, e deve ter a mesma referência e o mesmo valor para qualquer jogador. Dada essa condição, somente uma das máquinas pode fornecer esta referência. Para o caso de um servidor e vários clientes ligados em rede, o elemento único dentro da rede de computadores é o servidor, sendo assim, deve-se dar a este a função de controlador do tempo, passando seus valores atualizados periodicamente para os clientes.

O tempo é um elemento de contagem, fazendo-se notar pelos seus efeitos cíclicos ou de disparo de fenômenos. Este contador serve, por exemplo, para fazer o movimento de rotação de astros no céu, ou para indicar fenômenos sazonais de seres vivos (épocas de caça, acasalamento, hibernação ou migração).

Quanto ao efeito de disparo, fenômenos naturais aleatórios tais como tempo agradável, chuva, ventos, granizo, calor excessivo e queimadas, podem ser gerados e seqüenciados com base no tempo.

Foi implementado no jogo o fenômeno de passagem do tempo variando a claridade do ambiente (noite e dia) e chuvas sazonais, onde o servidor decide aleatoriamente quando ocorre o início das chuvas e quanto tempo elas durarão, enviando a informação para os clientes.

Deve-se considerar também o efeito que o som faz sobre o usuário. Após o sinal indicando o início da chuva, um som característico é gerado. O computador cliente também decide aleatoriamente se sinais sonoros de trovões devem ser reproduzidos. As Figuras 45 e 46 mostram as telas renderizadas do programa, com os avatares e a chuva.

Fig. 45: *Sedan* visto por um segundo usuário conectado por rede, com chuva.

Fig. 46: Caminhão dentro do mesmo mundo aleatório gerado, com chuva.

4.6 Limitações

Os algoritmos de criação pseudo-randômica de mundos não foram estudados em sua totalidade, assim como todas as técnicas possíveis não foram estudadas.

Os modelos de geração pseudo-randômica de cenários não se baseiam em teoria de jogos ou de otimização, apenas na observação das aparências do mundo real.

Este modelo de geração de mundos é aplicado somente em mundos virtuais fictícios, e não para modelar coisas que já existem, embora a biblioteca de funções possa importar objetos tridimensionais já modelados.

Todos os métodos foram criados em somente um tipo de linguagem de programação, com somente um tipo de biblioteca gráfica, para somente um sistema operacional em computadores pessoais, mas podem ser implementados em outros sistemas operacionais, com outras linguagens e outras bibliotecas gráficas.

Apesar da existência de várias bibliotecas gráficas e vários sistemas operacionais, somente duas linguagens de programação, uma biblioteca gráfica e um sistema operacional específico foram escolhidos pelos seguintes motivos:

- Facilidade de compreensão das linguagens e de implementação destas;
- Quantidade abundante e disponibilidade de informações sobre estas linguagens na *Internet*;
- Disponibilidade de informações sobre o sistema operacional usado na rede mundial de computadores;

- Preço de aquisição baixo, com resultado de alto desempenho da biblioteca gráfica (relação benefício / custo muito alta);
- Informações atualizadas – sobre estas ferramentas – disponíveis no local de desenvolvimento do protótipo;
- Informações sobre programação de *hardware* disponíveis na rede mundial de computadores;

A implementação do jogo foi feita sem consulta a algum algoritmo específico já pronto para jogos em tempo real.

São usados modelos geométricos simples para o posicionamento e orientação dos objetos do mundo virtual.

Os ítens virtuais são muito grosseiros e simplificados, e o ambiente não se altera de forma dinâmica.

A implementação do jogo funciona somente em plataforma *Windows 98©* ou superior, para computadores pessoais com velocidade acima de 333 MHz, com resposta lenta para computadores abaixo desse limite.

Para o jogo funcionar, o computador deve ter no mínimo uma placa aceleradora de vídeo, um kit multimídia e uma placa de rede. Dispositivos opcionais tais como *joystick, cyberpuck*, capacete com rastreador de ângulo (*HMD*) e luvas digitais podem ser usados, mas não há formas de fazer comunicação com outros tipos de dispositivos ou interfaces.

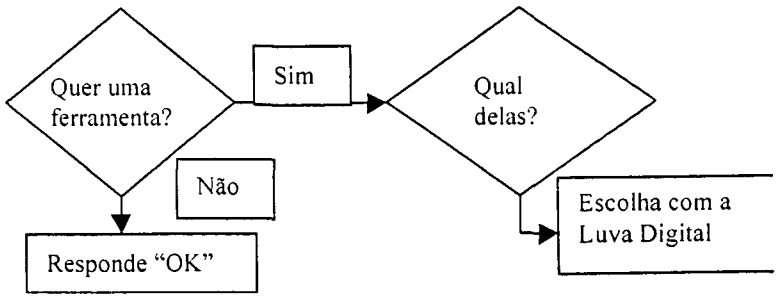
O algoritmo de transferência dos dados estáticos do ambiente virtual não é a prova de desconexões, conseqüentemente este algoritmo não é eficaz na rede

mundial de computadores, servindo apenas para redes locais (*LAN-Local Area Network*).

Para que o modelo de diálogo funcionasse (modelo tipo Questão-Resposta com *bots*) no protótipo, algumas simplificações foram utilizadas:

- Foram projetados somente oito tipos diferentes de personalidade para funcionarem no mundo virtual;
- O usuário fala somente quando está próximo do personagem autônomo, e para falar, deve ainda usar um mecanismo disparador (pressionando uma tecla), para o início e o fim da gravação;
- A rede neural utilizada foi programada para reconhecer somente duas palavras;
- Não existe um sistema de síntese de voz a partir de palavras escritas, e as respostas dadas pela máquina se restringem somente a gravações pré-produzidas;
- Para cada frase escrita pré-produzida, foram usados oito timbres de voz diferentes, representando a fala de cada personagem para aquela frase;
- O diálogo se extende até a resposta final da máquina, após o reconhecimento dos sinais da palavra dita pelo usuário;
- A partir do momento em que o diálogo exige palavras que estão fora do processamento da máquina, outros

Fig. 47: Fluxograma do detalhamento de questões após o reconhecimento de fala.



recursos de interface são utilizados, tais como a luva digital ou o teclado (Fig.47).

Devido às limitações nos algoritmos do sistema de reconhecimento de fala, nem todas as situações podem ser resolvidas através da conversação, somente as situações que requerem respostas "Sim" ou "Não".

Para contornar esta limitação, o sistema é inteligente o bastante para decidir que tipo de interface é mais apropriado para que as informações provenientes do usuário sejam processadas pela máquina.

As interfaces utilizadas pelo usuário durante o diálogo são três: o microfone, as luvas digitais e o teclado. Uma vez que o usuário esteja em situações ou diálogos que não se possa comunicar através de respostas afirmativas ou negativas (a escolha de um dos elementos de uma lista, por exemplo), o reconhecimento de palavras não pode ser mais requisitado, recorrendo-se a outras interfaces (Exemplo na Figura 47).

A rede neural foi programada para reconhecer somente duas palavras em português, e mesmo assim com índice de erro alto.

O ambiente gerado é a céu aberto, não apresentando o interior de construções virtuais.

4.7 Implementação e Análise

As Figuras 48 e 49 mostram a tela principal dos programas servidor (criador), com a textura criada para dar a aparência do terreno, e cliente. Uma caixa de texto mostra o *status* do programa antes do *rendering* do ambiente.

Fig. 48: Tela principal do programa servidor criador, com a pista desenhada.

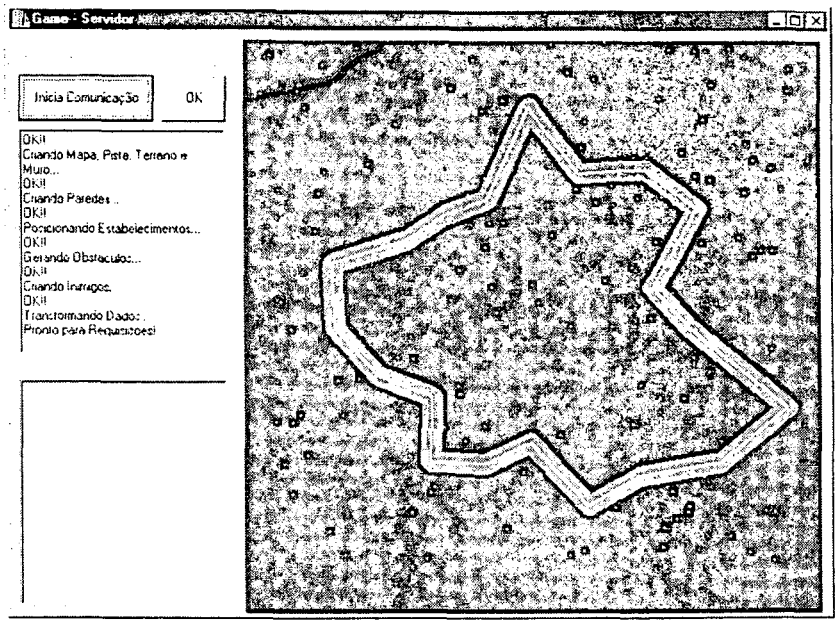
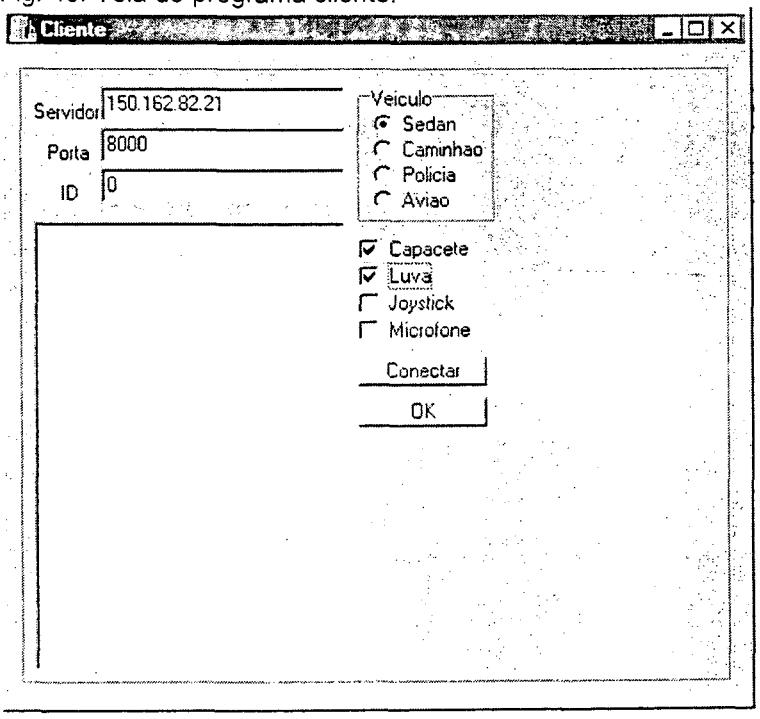


Fig. 49: Tela do programa cliente.



Dos ítens que fazem parte do modelo geral do jogo, foram implementados os seguintes ítens finais do protótipo:

- Navegação dos jogadores pelo mundo, com controle opcional de movimento pelo teclado, *joystick*, *cyberpucks*,

luvas, e controle de direção de visão por *tracker.* A Figura 47, na página 109, mostra as opções de interação usuário-computador (capacete, luva, *joystick* e microfone);

- Simulação de passagem do tempo com dia e noite;
- Escolha de avatares pelo jogador (três modelos simplificados de veículos);
- Mapeamento de colisão em texturas. Um exemplo de como os objetos podem ser mapeados através de texturas para efeitos de prever colisões é mostrado na Figura 50, onde os pontos em cinza impedem que o avatar, *bot* ou mesmo um objeto inanimado ocupem o mesmo espaço. No mundo em três dimensões, a impressão que se tem é que o usuário se choca com o objeto e não o atravessa. O tamanho do ponto no mapa *raster* corresponde proporcionalmente ao tamanho do objeto no mundo tridimensional. O mapa é apresentado sem texturas para melhor visualização e compreensão sobre o processo de colisão;

Fig. 50: Representação dos objetos fixos no mapa *raster* através de pontos.

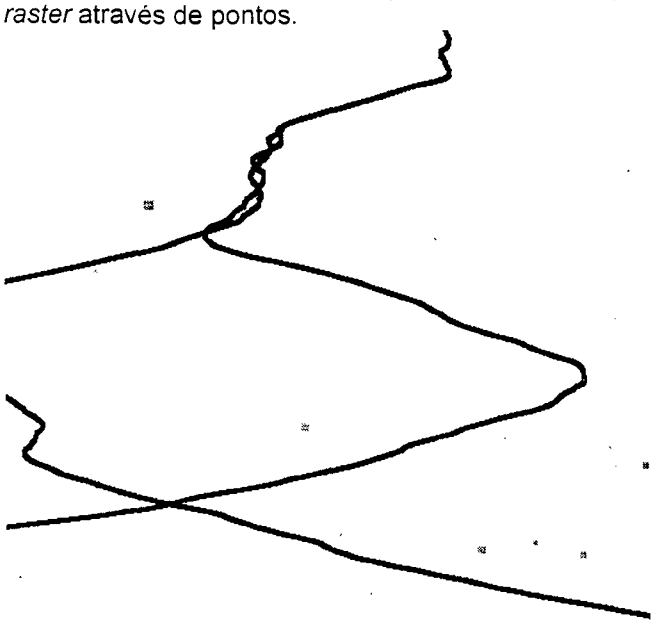


Fig. 51: Geração de pontos em volta de um centro principal.

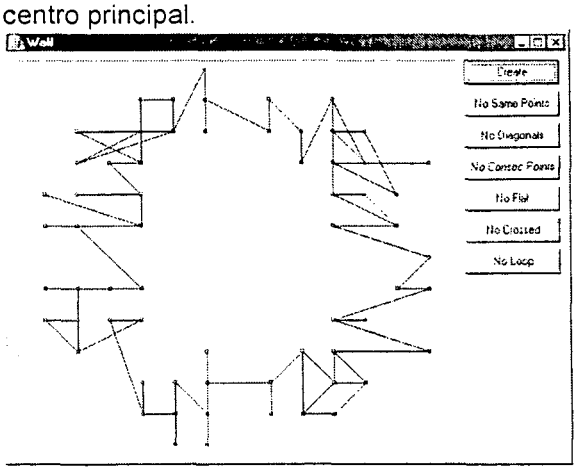


Fig. 52: Discretização dos pontos gerados.

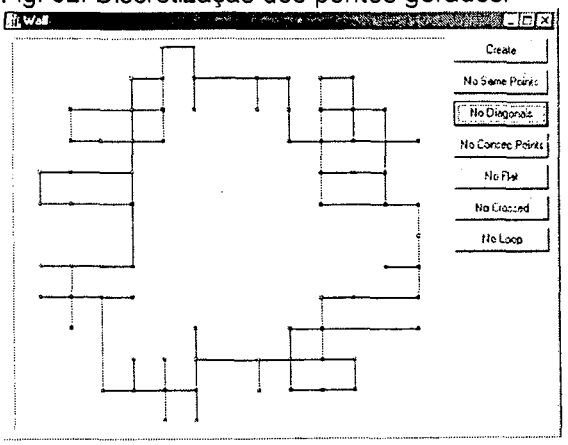


Fig. 53: Eliminação dos pontos redundantes.

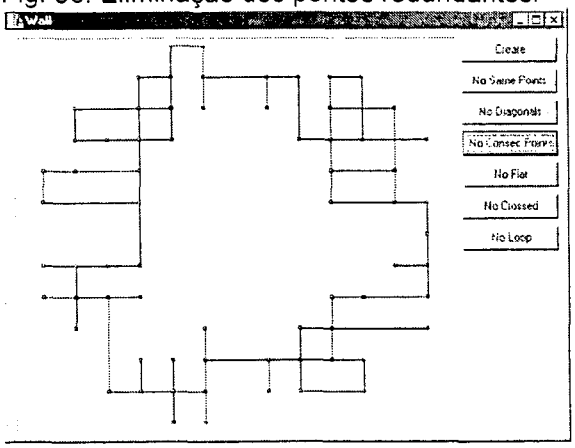


Fig. 54: Eliminação de pontos chatos.

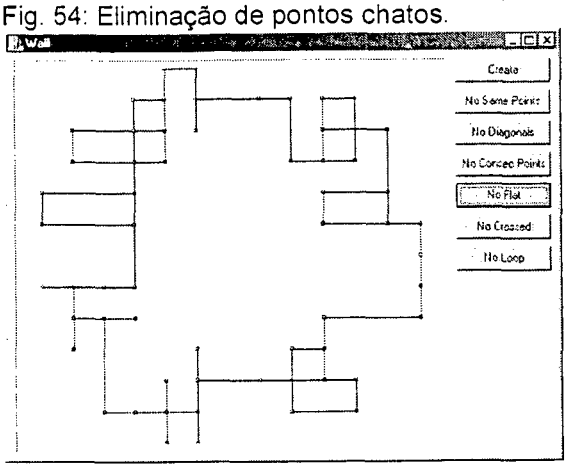


Fig. 55: Eliminação de pontos cruzados.

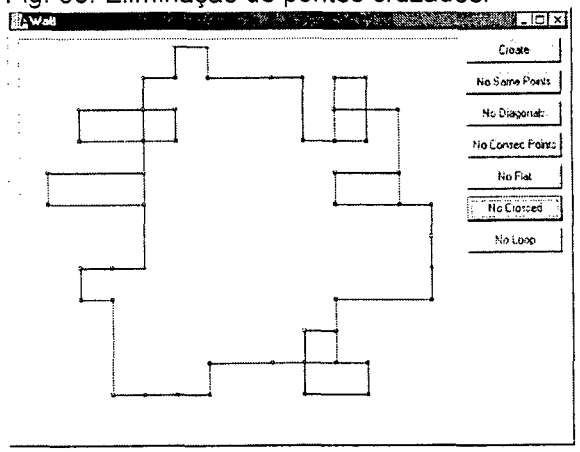


Fig. 56: Eliminação de pontos em ciclo.

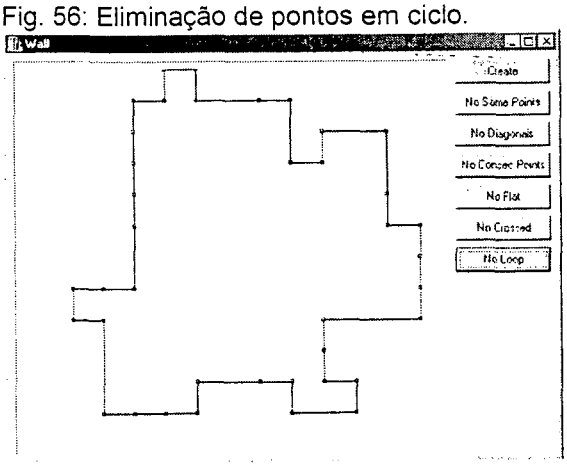


Fig. 57: Eliminação dos pontos consecutivos remanescentes.

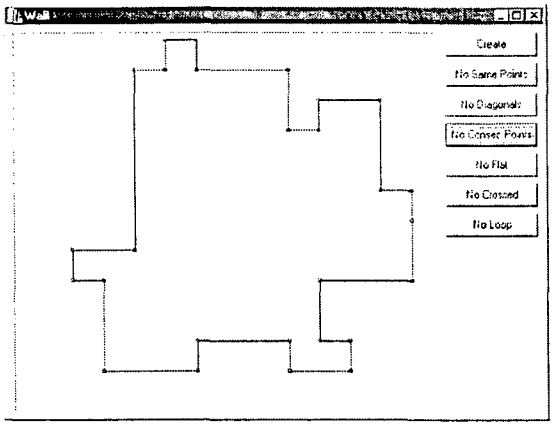
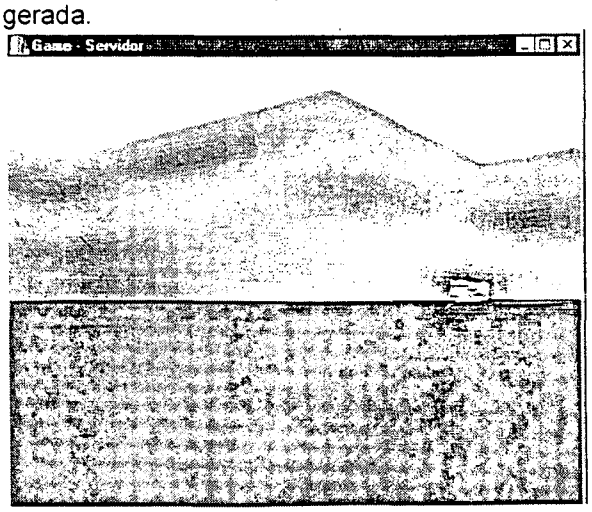


Fig. 58: Visualização 3D da parede gerada.



- Criação pseudo-randômica do terreno, a textura do terreno, paredes e objetos, e envio destes objetos criados para arquivos *VRML*. A técnica de criação de paredes aleatórias é um pré-processamento de pontos distribuídos pseudo-aleatoriamente em torno de um centro, dentro de uma região retangular. Nas Figuras 51 a 57 é descrito passo a passo o processo. A Figura 58 mostra um resultado tridimensional da parede gerada;
- Comunicação de cinco computadores em rede local, com transferência do mundo criado no servidor para os clientes no início da conexão, e envio de posição e orientação dos avatares em tempo real;
- Movimentação de *bots* com regras de vida artificial cujas características são informadas através da rede;

Além destes ítens, foram programadas ferramentas de auxílio na modelagem de objetos e sua manipulação com as luvas digitais, treinamento de RNA Multi-Camadas para reconhecimento de palavras isoladas, agentes inteligentes que tentam prever as trajetórias feitas pelo jogador em tempo real, e um algoritmo de procura por trajetórias.

5 RESULTADOS OBTIDOS

5.1 Sobre as Placas Aceleradoras, Biblioteca *OpenGL*

Algumas placas aceleradoras não aceitaram bem o algoritmo de *rendering* do terreno pseudo-aleatório (modelos *3dfx STB* (julho de 2000) e *3dfx Voodoo 2000* (julho de 2000)).

Tais modelos forneciam erros de divisão por zero, mas foram bem aceitas em duas placas: *Number Nine* modelo *Revolution IV* (julho de 2000) e *Diamond Multimedia* modelo FireGL1000 (julho de 2000).

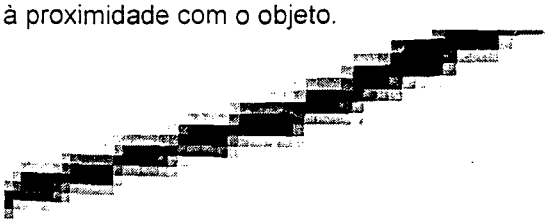
Na programação da biblioteca gráfica *OpenGl* para placas aceleradoras, os elementos *QUAD_STRIP* (WRIGHT, 1996) em cadeia são melhores de programar um terreno acidentado com elementos de resposta em tempo real do que os elementos *NURBS* (WRIGHT, 1996, NEIDER, 1997), oferecendo resultados mais precisos e rápidos.

5.2 Quanto às Texturas

A textura do terreno é muito pesada para as placas aceleradoras gráficas (512 X 512 pontos X 4 *bytes* por *pixel*), prejudicando sensivelmente a velocidade de *rendering* do ambiente virtual. A solução encontrada foi utilizar uma textura de tamanho menor (128 X 128 pontos) e eliminar a possibilidade de transparência (usando somente três bytes por *pixel*) desta textura na placa aceleradora.

Três efeitos bastante desagradáveis que freqüentemente ocorrem em jogos 3D são a pixelagem, a desuniformidade em texturas repetitivas, e o desalinhamento de extremos.

Fig. 59: Exemplo de textura pixelada devido à proximidade com o objeto.



5.2.1 Pixelagem (Fig. 59): Num mundo virtual, onde os vários objetos estão cobertos por suas respectivas texturas, à medida que o jogador se aproxima de um dos objetos, sua textura vai aumentando, até mostrar a diferença entre um pixel e outro, mostrando também seu formato retangular.

Para amenizar este tipo de problema as placas aceleradoras apresentam duas soluções diferentes:

- Gerenciamento de texturas múltiplas (*mipmapped textures*) (WRIGHT, 1996 e OpenGL®, 2001)): Uma lista de texturas é armazenada, e quanto mais próximo o usuário se encontra do objeto recoberto, a textura mais detalhada da lista é mostrada. O uso de texturas múltiplas apresenta a desvantagem de ocupar muita memória da placa.
- Uso de interpoladores de cores de uma região da textura (*blur*)) (WRIGHT, 1996 e OpenGL®, 2001)): Cada ponto da textura dada possui uma cor, e a transição de um ponto para outro possui uma suavização dada por uma interpolação. Este método ocupa menos memória, mas é mais demorado.

5.2.2 Desuniformidade de Texturas Repetitivas: Texturas podem ser repetidas várias vezes, mas seu formato retangular limita as possibilidades de texturas com bom efeito estético. As Figuras 60 e 61 mostram um caso de textura desuniforme, onde o desenho de um hexágono como textura básica resulta na formação de um

losango indesejado, e um caso de textura de tijolos que se mostra perfeita quando repetida.

Fig. 60: Textura de hexágonos desuniforme: Note a formação de pequenos losangos.

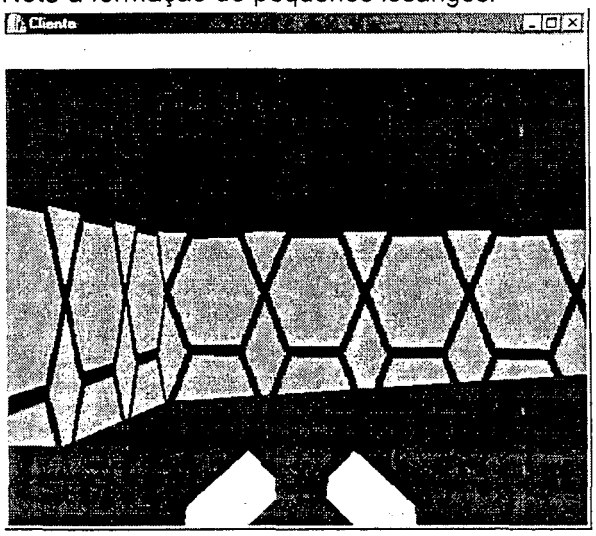
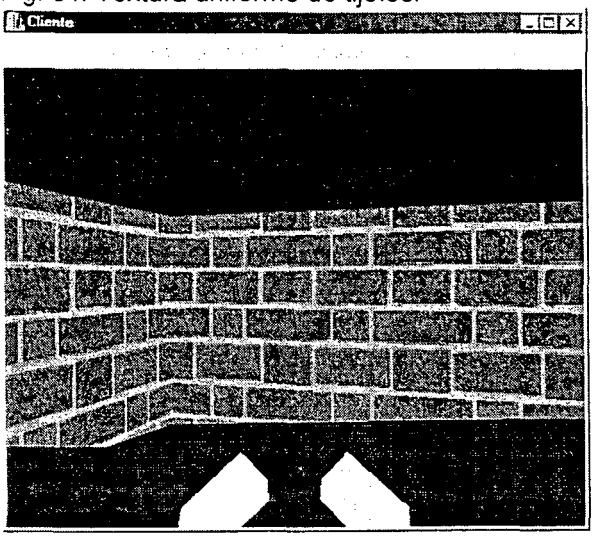


Fig. 61: Textura uniforme de tijolos.



5.2.3 Desalinhamento de Extremos: Texturas cujos extremos não coincidem, e que recobrem pequenos objetos suaves, formam uma linha perceptível e antiestética. A Figuras 62 e 63 mostram um exemplo deste efeito (TEXTURE WORLD, 2001).

Fig. 62: Textura usada para recobrir um objeto

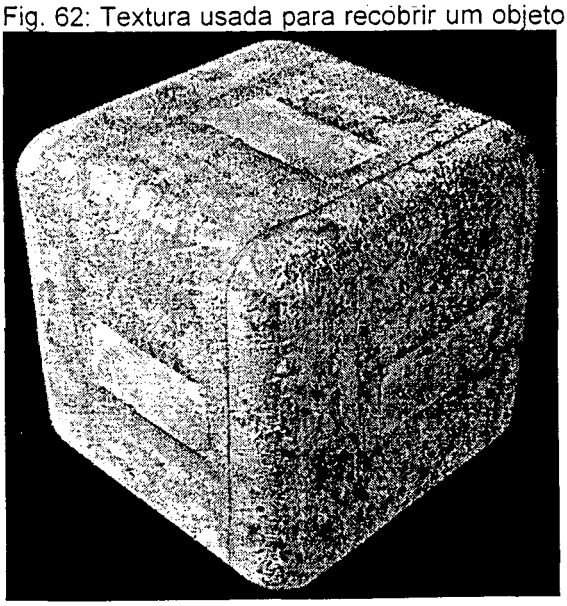
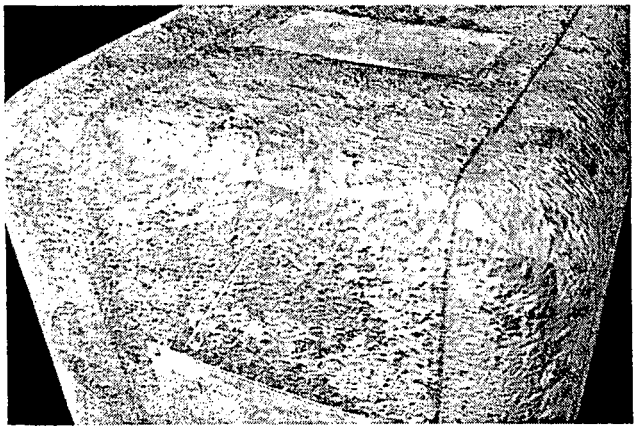


Fig. 63: Detalhe da linha indesejável (ver seta) devida ao desalinhamento dos extremos da textura.



5.3 Sobre os Protocolos de Comunicação de Redes

Nota-se a diferença de velocidade de transmissão de dados para blocos de pequeno tamanho, durante a transferência de dados dinâmicos.

Durante o funcionamento do protótipo do jogo, notou-se sensível diferença de tempo de resposta em computadores ligados em rede que tivessem diferentes velocidades de processamento.

Procurou-se então fazer a atualização de valores dinâmicos com base no tempo (PAUSCH *et al.*, 1994). E fazer a transferência do mundo virtual com base em blocos de pedidos de tipos e tamanho diferentes (ver REDES DE COMPUTADORES no capítulo 4: APLICAÇÃO E ANÁLISE).

5.4 Quanto à Ergonomia

O usuário tem a impressão de estar dentro de um *cockpit* durante a navegação sentindo o giro do volante ao dirigir o carro com a luva ou *joystick*.

O mundo criado, apesar de estar muito estilizado, já oferece imersão convincente, mesmo sem os óculos e *tracker*.

Comandos específicos de captura foram usados para aumentar a resposta do teclado, com resultados excelentes.

A manipulação de objetos feita com luvas digitais é suave, mas um pouco incômoda. Um experimento feito mostra que um jogador leva, no mundo virtual, aproximadamente meio minuto para fazer o simples posicionamento de um objeto à sua frente usando somente rotações da luva.

Os valores lidos das rotações das luvas são valores médios de três ângulos nos sentidos correspondentes, tomados em seqüência, e a rotação do objeto resulta destes valores.

Para indicar que o objeto foi pego ou solto, determinou-se um valor limite do dedo indicador acima do qual significava mão aberta e objeto solto, e abaixo deste valor, a mão estava fechada e o objeto estava seguro.

O capacete ligado ao *tracker* permite a visualização dos mundos tridimensionais somente em resolução máxima de vídeo de 640 X 480 pontos, 256 cores e freqüência de 60Hz.

Em casos onde a pessoa que joga sofre de labirintite, o uso do capacete durante a navegação pode provocar enjôo, tonturas e dor de cabeça.

O recurso de valores médios para suavização do movimento do capacete não pode ser usado, pois a própria tomada direta de leituras de ângulo do capacete é descontínua, provocando saltos nos valores das médias.

Houve sucesso na captura simultânea de valores das luvas esquerda e direita, graças aos protocolos de comunicação de portas seriais disponíveis no sistema operacional *Windows 98©*. Sendo conseguida também a leitura do *tracker* do capacete. Infelizmente, o número de portas seriais é de somente dois para cada computador, comportando somente um conjunto de dois periféricos que necessitem de comunicação serial.

Para que se consiga ler os valores das duas luvas e do *tracker* do capacete, necessita-se de mais uma placa controladora contendo uma porta serial.

5.5 Quanto ao Posicionamento de Objetos

Existe um erro mínimo de posicionamento devido ao arredondamento na passagem do mundo tridimensional para o mapa bidimensional, podendo ser ajustado com pequeno deslocamento dos objetos.

Os objetos fixos não se orientam de acordo com a inclinação do terreno, aparecendo buracos entre os planos definidores dos objetos e o plano do terreno na posição correspondente, mas se posicionam de forma convincente sobre a superfície do terreno.

Os *avatares* seguem a inclinação do plano do terreno em que se encontram.

5.6 Quanto ao Algoritmo de Reconhecimento de Fala

O algoritmo de reconhecimento de palavras isoladas foi programado para entender duas palavras inicialmente ("Sim" e "Não"), e o pré-processamento e treinamento da rede neural que o compõe ainda não fornecem resultados satisfatórios. A rede neural que é usada é programada com orientação a objetos, sendo possível utilizá-la com outras finalidades.

A freqüência de captura do microfone para o reconhecimento de fala é baixa, da ordem de 8kHz, para economia de memória e de tempo de processamento. O tempo máximo de fala é de oito segundos.

5.7 Quanto aos Agentes Inteligentes

Foram programados agentes inteligentes imitando vendedores dentro de lojas que disparam uma pergunta ao jogador caso sintam a sua proximidade, iniciando então um diálogo falado.

Para que o jogador fale durante o jogo, este deve esperar alguma pergunta sonora do agente inteligente, pressionar a barra de espaço, falar, pressionar a barra de espaço novamente, onde a partir daí o agente vai fazer o processamento da fala e fornecer a resposta, também sonora.

Na programação de agentes inteligentes que fazem a previsão de trajetória do jogador, dois objetos do compilador *Borland C++ Builder 4* (BORLAND, 1997) foram usados e comparados: um objeto *Timer* no primeiro caso, que executavam a função de procura a períodos regulares de tempo, e uma *Thread* no segundo caso, que é um objeto que funciona paralelamente ao programa principal, procurando pelas trajetórias semelhantes feitas pelo jogador.

A programação utilizando o objeto *Timer* ocasionava freqüentes travamentos na execução do processo, o que não acontecia com a *Thread*, mostrando esta última melhores resultados sobre o processamento paralelo de blocos de dados em tempo real.

5.8 Quanto ao Movimento de *Avatares* e *Bots*

Os humanóides programados podem fazer os movimentos articulados nos braços e pernas (simulando o movimento da coxa e do antebraço), cabeça (giro em dois eixos) e queixo (para simular o movimento de fala), possuindo texturas em todos estes elementos.

O movimento dos avatares não articulados é suave, mesmo quando se joga em rede.

Os *bots* simulando vida artificial não possuem movimentação articulada, mas comportam-se como vivendo em bandos.

5.9 Tabela Comparativa

A Tabela 3 tem a finalidade de descrever em resumo quais das tecnologias do capítulo 3 foram utilizadas na implementação do protótipo.

Tabela 3: Sumário do Uso das Tecnologias Estudadas.

Tecnologia	Aplicação no Protótipo
Modelagem de Projetos Orientada a Objetos	Usada em toda a programação do protótipo
Vida Artificial	Usada no regime do comportamento de *bots* (manada de bovinos no meio da pista)
Algoritmos Genéticos	Não usado
Pathfinder	Usado num protótipo derivado de uma técnica de separação de imagens em (DUDA, 1973). Ver seção 3.6 e 4.5.4. Não implementado no protótipo final.
Previsão de Trajetórias com Mineração de Dados	Aplicado num protótipo de procura por similaridades usando Séries Temporais (AGRAWAL, 1995). Não implementado no protótipo.
RNA, Reconhecimento de Palavras e Sistema de Diálogos tipo Questão-Resposta para Interação Falada	Usado num protótipo temporário de reconhecimento de palavras isoladas, para que o usuário interaja com o personagem virtual usando microfone e autofalantes.
VRML	Utilizado para uma navegação prévia no Ambiente Virtual, antes que o jogo inicie
Redes de Computadores	Usado para comunicar as informações provenientes dos vários usuários participantes e transportar o ambiente para as máquinas clientes.
Detecção de Colisão	Usado para prevenir colisão de corpos móveis no Ambiente Virtual.
Geração Pseudo-Randômica Paramétrica de Ambientes Virtuais	É um conjunto de algoritmos responsável por gerar o Ambiente Virtual onde ocorrerá o jogo. É o coração do protótipo final.
Navegação em Ambientes Virtuais	Usado para simular comportamento físico no Ambiente Virtual. Também contém previsão de colisão. Possui também modelos de navegação para automóveis e para indivíduos. Implementado no protótipo final.
Programação de *Hardware* para Interface Homem-Máquina	Conjunto de técnicas usadas no protótipo final para permitir a interface entre o usuário e o Ambiente Virtual da máquina através de *hardware* específico.

6 CONCLUSÃO

Durante o desenvolvimento do protótipo, a pesquisa mostrou a relevância ou não dos ítens pesquisados das várias tecnologias. Foram identificadas como relevantes para futuros trabalhos as seguintes:

- O modelo de navegação do automóvel;
- Os modelos de transferência de dados usando *hardware* de Realidade Virtual;
- A biblioteca gráfica utilizada;
- As técnicas de reconhecimento de fala;
- A comunicação e transferências de dados via redes de computadores;
- A filosofia de criação pseudo-randômica de elementos virtuais;
- Programação de técnicas de Algoritmos Genéticos para aprendizado de máquina e otimização de procedimentos;
- A Implementação de fenômenos climáticos pseudo-randômicos, para adicionar adversidades no jogo;
- A Implementação de técnicas de Vida Artificial para simular elementos dinâmicos aleatórios no Ambiente Virtual;

E como irrelevantes, as seguintes técnicas:

- A Implementação do modelo do terreno de pontos cotados como elemento de referência para a pista de corrida, seu uso é ineficiente e provoca morosidades no processo de *rendering*, e grande parte de sua extensão não é visível para o usuário;

- A criação pseudo-randômica de paredes fornece resultados bons dentro de determinados limites, sendo necessária ainda uma última otimização no processo de criação, mas seu uso se mostrou desnecessário para o modelo de pistas de corrida a ser desenvolvido no futuro;
- A programação de um Sistema de Diálogo tipo Questão-Resposta não é adequada para o projeto futuro, sendo suficiente um sistema que obedeça a comandos de voz do usuário, para execução de tarefas simples;
- A programação de *bots* humanóides não é necessária para o futuro projeto de corrida de veículos;
- Algoritmos de *pathfinder* e previsão de trajetórias não serão necessários para jogos envolvendo corridas de veículos;
- A linguagem VRML não é necessária para fazer uma visualização do ambiente de pista de corridas de automóveis a ser gerado. Uma navegação do ambiente dentro do próprio protótipo antes da disputa propriamente dita já é suficiente.

7 SUGESTÕES DE ALTERAÇÃO NO PROTÓTIPO

Determinação de um tema mais específico sobre o futuro trabalho a ser desenvolvido sobre jogo de corridas.

Tornar dinâmicos os elementos do Ambiente Virtual através da adição de efeitos de movimento independente em objetos, tais como portões abrindo e fechando, elevadores de grande porte e explosões.

O Sistema de Diálogos pode ser composto somente por um agente que compreende uma ordem falada do usuário e executa esta ordem. Para isso é

necessário melhorar o pré-processamento de sinais de voz e treinar a RNA para reconhecer mais palavras, e ainda possibilitar ao *software* capturar os sinais de som de forma contínua, e eliminando o disparador da leitura pelo microfone.

Implementar o surgimento de mais fenômenos naturais tais como ventos, granizo, queda de meteoros, cometas, neblina, nuvens e vulcões.

Usar técnicas de Algoritmos Genéticos e Vida Artificial para gerenciar o comportamento de colônias de agentes inteligentes.

Implementação de métodos de pré-processamento para geração pseudo-randômica de situações aleatórias para dar mais variedade ao jogo.

Otimizar funções matemáticas que regem os fenômenos físicos.

O protótipo também precisa de um método avaliador da performance ou eficácia de cada jogador.

No que concede ao assunto de redes de computadores, a transferência de dados deve ser a prova de quedas de conexão, para que funcione também na *Internet*, e não somente em Redes Locais (*LAN*'s)

Para a utilização de periféricos que transformam o movimento natural do ser humano em sinais digitais, alguns cuidados devem ser tomados, e algumas recomendações devem ser seguidas:

No caso da ausência de algum periférico que faça a transdução dos movimentos naturais, um substituto deve ser preparado. Podendo ser o teclado ou o *mouse.*

O uso de periféricos tende a crescer com o tempo, sendo necessárias mais portas de comunicação para suportar esta quantidade crescente.

Muitos periféricos já possuem protocolos de comunicação padrão, com taxas de transferência determinadas e palavras de comando já programadas. O sistema operacional *Windows 98©* já possui protocolo preparado para a comunicação serial com periféricos, e inclusive periféricos de jogos com calibração.

Os periféricos usados nesta experiência fornecem resultados absolutos de rotação, respondendo a uma velocidade limite de variação do movimento natural do corpo.

Se este valor limite for ultrapassado, valores descontínuos de rotação serão fornecidos, fazendo com que objetos visíveis – cuja orientação depende destes valores lidos – produzam saltos na tela do monitor, e fujam do controle do usuário. Para resolver este problema, médias dos valores lidos são fornecidos para a orientação destes objetos.

Tais periféricos podem ser usados por pessoas de características físicas diferentes, sendo necessário uma calibração de valores máximos e mínimos antes do uso propriamente dito.

8 UTILIDADE DAS TECNOLOGIAS PESQUISADAS

Nesta parte do capítulo serão descritas algumas aplicações das técnicas de Realidade Virtual e Inteligência e Vida Artificial estudadas. Alguns ítens foram adicionados para efeito de ilustração para interessados no assunto, incluindo também suas referências.

8.1 Uso das Tecnologias de Realidade Virtual

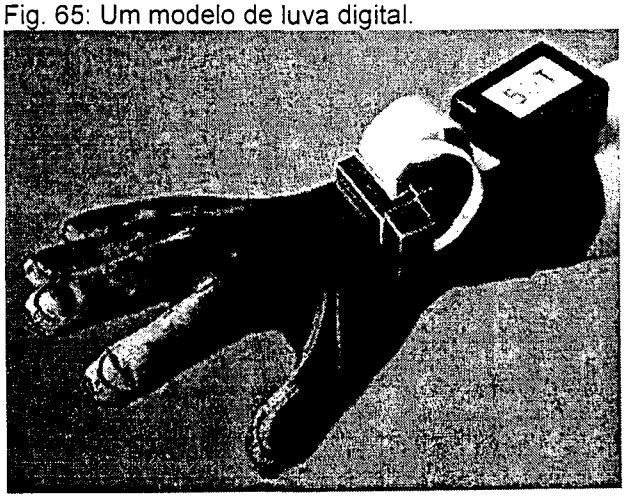
8.1.1 *Hardware*

Fig. 64: Um modelo de um HMD.

HMD ((SUTHERLAND, 1964, BURDEA, 1996-a)): Sigla em inglês de *Head Mounted Display* (Tela Montada na Cabeça), possui um sensor de ângulo de orientação (*tracker*) para enviar esses dados ao microcomputador, e o computador responde com a imagem correspondente a este ângulo. Possui visão estereoscópica.

O *HMD* imerge o usuário no Ambiente Virtual através da visão e audição, e através de microfones, dando a impressão de que este se encontra dentro do mundo onde navega. Um monitor comum e um *HMD* podem ser ligados de forma a mostrar a mesma imagem.

Fig. 65: Um modelo de luva digital.

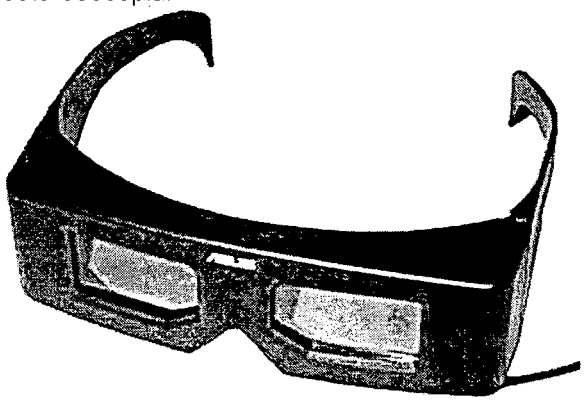


Luvas Digitais (ROEHL, 1995-a (*apud* BURDEA, 1996-b)): Servem para capturar a posição dos dedos das mãos e o ângulo de rotação do pulso, em volta do eixo do antebraço, e em volta do eixo longitudinal, fornecendo valores discretos ao computador. Usa a porta serial do microcomputador para se comunicar. Luvas mais atuais

possibilitam tomar até os valores de ângulos de cada articulação dos dedos. Suas aplicações englobam tudo que exige manipulação de objetos virtuais visíveis. Podem-se encontrar exemplos de aplicação desde jogos até medicina, física nuclear, genética, processos de fabricação, entre outros.

Óculos Estereoscópicos (AKKA, 1992 (*apud* BURDEA, 1996-b)): Usados junto com alguns modelos de placas aceleradoras gráficas para dar a impressão de estereoscopia. Não possui sensor de ângulo. Uma das tecnologias usadas para dar a impressão de profundidade é a visão estereoscópica, que é aquela formada pelos diferentes ângulos de visão dados pelos olhos ao fixarem-se em um ponto, junto com obturadores, que dão uma imagem para cada olho por vez. Estes tipos de óculos são usados em sua maioria para jogos ou apresentação de filmes.

Fig. 66: Um modelo de óculos com estereoscopia.



Joystick (GRADECKI, 1994): Semelhante ao manche de um avião, usado para capturar movimentos contínuos. Possui botões que têm função de disparadores. Faz a comunicação com o computador através de porta serial para jogos de qualquer placa de som. Necessita de aferição para poder ser usado. Usado em simuladores de vôo, jogos de rolagem de tela, corrida de veículos e jogos estilo atirador 3D.

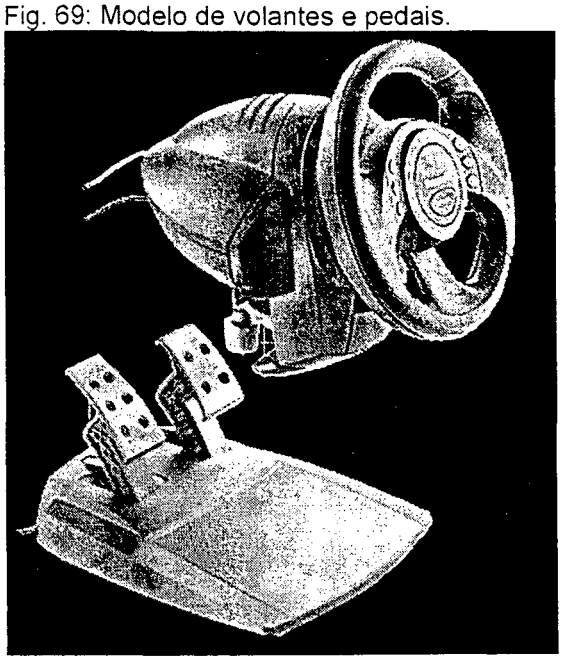
Fig. 67: Um modelo de joystick.

Cyberpuck (*Interactive Imaging Systems, Inc.*, 2002): Responsável por enviar sinais de deslocamento no plano do horizonte para o computador. Possui botões que têm a função de gatilho, disparando um evento ao serem pressionados. Aplicado em jogos de computadores.

Fig. 68: Um modelo de cyberpuck.

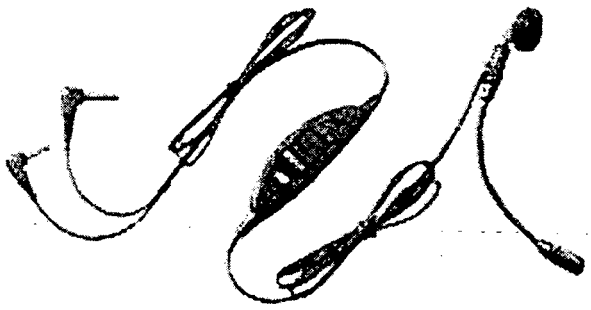
Volantes e Pedais (*Steering Wheel*): Simulam o uso de um volante e pedais de aceleração e freio de um carro. O volante fornece *feedback* de força a partir da simulação onde está sendo usado. Aplicado em jogos de computadores.

Fig. 69: Modelo de volantes e pedais.



Kit Multimídia: Microfones e Auto-Falantes (GRADECKI, 1994): A placa de som possui três finalidades: capturar e armazenar sinais digitais provenientes do microfone, produzir som a partir dos autofalantes, e comunicar o computador com o *joystick* através da porta serial para jogos. O microfone é responsável por transformar os sinais analógicos de som em sinais digitais, e o autofalante converte sinais digitais armazenados no *buffer* da placa para sinais analógicos de som. Somente o conjunto de placas de som é usado para dar efeitos sonoros a jogos de computadores ou aplicações multimídia. Quando junto a uma placa de pré-processamento de sinais de som e um microfone, sua aplicabilidade se extende

Fig. 70: Modelo de microfone e autofalantes integrados.

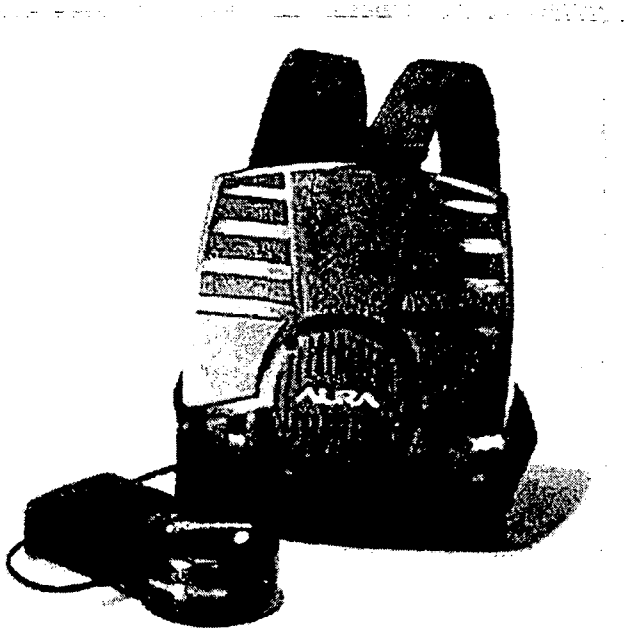


desde jogos com comunicação de som multi-usuário e de diálogo com agentes inteligentes até aplicações sérias de edição de texto ou navegação pela Internet.

Placa para Comunicação via Redes de Computadores (HUNT, 1998): *Hardware* interno responsável por fazer a comunicação entre vários computadores. A comunicação entre computadores se dá através de linguagens de comunicação próprias para placas de rede, chamadas de protocolos de comunicação. Alguns protocolos usados para comunicação de dados de jogos são os protocolos TCP/IP, IPX/SPX e UDP. Faz a comunicação de grupos de computadores, para que compartilhem a mesma informação a qualquer momento e em qualquer lugar. Servem para qualquer aplicação que necessite comunicar dados de computadores à distância. O uso mais difundido de redes é a rede mundial de computadores ou *Internet.*

Interactor (LONG e ALEXANDER, 1995 (*apud* BURDEA, 1996-a)): Colete plástico que amplifica os sinais sonoros de videogames e os transforma em vibrações mecânicas sobre o dorso do usuário. Usado em jogos.

Fig. 71: Aura Interactor.



Placa Aceleradora com Biblioteca Gráfica (Silicon Graphics, 1992 (*apud* BURDEA, 1996-a)): A placa aceleradora é um dispositivo de *hardware* que tem a finalidade de agilizar a *rendering* de Figuras tridimensionais, descarregando a memória do computador de cálculos geométricos para este fim. Algumas placas aceleradoras possibilitam visão estereoscópica. Seu campo de aplicação é vasto, extendendo-se desde o *rendering* de modelos 3D para simulações com Realidade Virtual, criação de efeitos visuais em produções cinematográficas, jogos, e aplicações em Engenharia e Medicina.

8.1.2 Simulação e Treinamento

Para simulação e treinamento, podem ser criados agentes inteligentes que auxiliem o aprendizado dos usuários, dialoguem com eles, sugiram soluções alternativas para problemas dados.

Situações que exijam raciocínio podem ser criadas, oferecendo sempre um novo problema a ser resolvido, treinando o usuário através da repetição dos mesmos fenômenos, mas com argumentos diferentes.

As tecnologias aqui pesquisadas permitem também que dados para aprendizado sejam criados, que podem ser transferidos a longas distâncias, e que se possa interagir com eles, podendo-se até modificá-los, permitindo desta forma a implementação de ferramentas de treinamento à distância.

8.1.3 Entretenimento e Jogos

A partir de algumas informações em (LARIJANI, 1993), dentro da área de jogos e entretenimento, as tecnologias pesquisadas aqui podem ser utilizadas nos seguintes ítens:

- Filmes: Computação Gráfica em Filmes, com efeitos de *morphing* (Deformação de Imagens ou Elementos Tridimensionais), animação feita por computador, simulação de fenômenos naturais e/ou efeitos especiais, ferramentas de edição, e cinematografia.
- Jogos: Atualização de títulos de jogos de 2D para 3D com aspectos de RV, passagem de títulos de vídeo para jogos ou vice-versa.
- Ferramentas de captura de imagem de objetos reais e transferência destas imagens para formato digital de dados, e recriação digital tridimensional do modelo capturado.
- *Hardware* para simulação de passeios virtuais, e que permitam interatividade, e até mesmo alterações do ambiente feitas pelos próprios usuários, em conjunto e em tempo real.

8.1.4 *VRML*

A linguagem *VRML* (*Virtual Reality Modelling Language*) é uma linguagem interpretada que reproduz Ambientes Virtuais pré-modelados, e disponibiliza estes ambientes na *Internet*. Tal disponibilidade na rede permite um acesso por um grande número de clientes, a qualquer momento. Entre as utilidades da *VRML* estão:

A criação de ferramentas para Educação à Distância, onde um grupo de experiências virtuais, já pré-programadas, está à disposição do público para ser consultado e utilizado como fonte para o aprendizado (TRAUER *et. al.*, 1997);

O uso da linguagem para modelar e analisar ambientes de trabalho quanto à ergonomia, através da observação com câmeras controladas com uma interface programada em *VRML* (MERINO *et. al.* 1998);

Simulação de passeios virtuais por locais históricos (REBELO, 1998);

Controle de braço robótico à distância (GARCIA *et. al.*, 2000);

Chat Virtual para Ensino à Distância (ROSA JÚNIOR, 2001).

8.2 Uso das Tecnologias de Inteligência Artificial

8.2.1 Séries Temporais

A Análise de Séries Temporais é utilizada para identificar similaridades de seqüência de dados que variam em relação ao tempo. Muito usada para fazer previsões do comportamento financeiro em bolsas de ações, ou ainda prever a redução ou aumento da queda de consumo de energia elétrica para controle de carga em usinas.

8.2.2 Redes Neurais Artificiais

Multi-Layer-Perceptron com *Backpropagation*: Segundo (HAYKIN, 1998) este tipo de rede tem sua aplicação com sucesso nas seguintes áreas:

- Redes Neurais que aprendem a pronunciar palavras em inglês;
- Reconhecimento de palavras;

- Reconhecimento óptico de caracteres;
- Reconhecimento em tempo real de caracteres manuscritos;
- Melhora na legibilidade através de reconhecimento de sinais sonoros da fala e de imagens da face durante a pronúncia de palavras;
- Controle de *Hardware;*
- Controle de veículos;
- Detecção e classificação de alvos de radares;
- Diagnóstico médico sobre paradas cardíacas;
- Modelagem de controle do movimento ocular.

Também segundo (HAYKIN, 1998), as RNA tipo *Learning Vector Quantization* (*LVQ*) pertencem a uma classe de redes auto-organizadas, com aprendizado por competição, sua utilidade se extende, entre outros, aos seguintes temas:

- Controle de braços robóticos;
- Datilografia através de fonemas (*Phonetic typewriter*);
- Classificação de gelo no mar com uso de radar;
- Modelagem do cérebro;
- Detecção de clonagem em grandes sistemas de *software* de comunicação.

Em (WATSON, 1996) é implementado um exemplo onde se usa RNA para prover a capacidade de aprendizado de situações por personagens virtuais dentro de um jogo.

8.2.3 *Pathfinder*

Métodos de Encontros de Caminhos (*Pathfinder*) são usados em quaisquer problemas que estejam modelados da seguinte forma:

- A situação se dá numa região bi ou tridimensional, limitado por uma curva ou superfície, respectivamente;
- Esta região contém algumas sub-regiões que estão inacessíveis;
- Dentro desta região maior existe um objeto móvel, que esteja numa posição inicial acessível;
- Determina-se que este objeto móvel vá para uma posição final, a partir da posição atual, sem atravessar as regiões inacessíveis.

Dentro destes tipos de problema, encaixam-se os modelos de Sistema de Informação Geográfica ou Espacial, onde há procura por trajetórias possíveis entre dois pontos quaisquer, após um mapeamento de uma determinada região do espaço terrestre, marinho ou aéreo.

Pathfinder em Sistemas de Navegação

Em termos mais específicos, estas informações são utilizadas por Sistemas Autônomos de Navegação, junto com as tecnologias de *Pathfinder*, para que seu deslocamento através de obstáculos seja o mais independente da interferência humana durante suas operações.

(PELL et. Al., 1998) descreve ítens de projeto para a nave espacial *Cassini*, que fará a captura de dados relativos aos anéis de Saturno. Entre tais ítens estão métodos de *Pathfinder* para navegação durante a viagem.

Pathfinder em Jogos de Realidade Virtual

Quanto ao assunto de jogos de Realidade Virtual, seus ambientes são tridimensionais, contendo adversários móveis, que forçosamente devem se deslocar da melhor maneira possível entre os pontos do ambiente em questão. Para que este deslocamento se realize, algorítmos de *Pathfinder* são usados.

8.2.4 Sistemas de Diálogos com Reconhecimento de Fala

Em (EDWARDS, 1997), citam-se três tipos de sistemas de reconhecimento de fala: O sistema de reconhecimento de comandos, e o sistema de ditado por voz, este último sendo dividido em dois subsistemas independentes: o ditado por voz discreta e o ditado por voz contínua.

Tais sistemas são usados em *software* comum, tais como planilhas de cálculos, editores de texto, navegação pela Rede Mundial de Computadores, jogos dos mais variados, cujas tarefas mais corriqueiras são executados com comandos de voz simples.

Quando combinados com outros tipos de interface, suas limitações – tanto da interface de voz, quanto a interface diversa – são compensadas pelas possibilidades oferecidas pelos demais tipos.

Apesar de vários produtos já estarem prontos, para interfacear o usuário e o sistema de serviço requisitado, muitos sistemas são sujeitos a erros devido à variedade de situações em que um conjunto de palavras faladas é exposto à máquina. Mesmo assim, várias linhas de pesquisa incluindo tradutores universais em tempo real, sistemas de serviço acionado por voz ou de aquisição de

informações através de diálogo falado no comércio por telefone estão sendo desenvolvidas.

8.2.5 Algoritmos Genéticos

Em (GOLDBERG, 1998) é descrito um histórico de várias utilizações para as técnicas em AG, entre elas se pode citar:

- Na área de Biologia, existem trabalhos na simulação da evolução de populações de organismos unicelulares e a simulação da adaptação de estruturas vivas quanto à disponibilidade espaço-temporal de alimento, entre outros;
- Nas Ciências de Computação, trabalhos envolvendo resolução de situações de jogos, métodos de *clustering*, descrição de documentos e procura por funções de avaliação de jogos;
- No campo da Engenharia, pode-se citar o uso de AG no projeto de filtros adaptativos recursivos, armazenagem e colorização de gráficos, projeto de layout e compactação de circuitos integrados de larga escala de operações, projeto de configuração de teclados, etc;
- Na área da Física pode-se citar a resolução de equações não lineares para ajuste de superfícies potenciais;
- Na área das Ciências Sociais, pode-se citar a simulação de um modelo de migração e solução de problemas tipo "Dilema do Prisioneiro", entre outros.

Em (WATSON, 1996) é citado um exemplo onde se usa AG para aumentar a capacidade de personagens virtuais dentro de um jogo.

8.2.6 Vida Artificial

Em (ESTEVAM, 1997) são descritas algumas áreas de pesquisa que podem se beneficiar com a Vida Artificial:

Robótica

Observando como se processa a biologia da percepção e reação dos animais e do homem, pode-se fazer robôs que usem tais leis para capturarem e processarem as informações à sua volta, e responderem da mesma forma. O autor ainda cita o livro "O Caçador de Andróides" de Philip. K. Dick, que fornece uma idéia de como serão as mudanças sociais a partir da aplicação da Vida Artificial na Robótica.

No Instituto de Tecnologia de *Massachussets*, pesquisadores têm criado robôs baseados em insetos, gerando resultados superiores aos resultados de métodos tradicionais.

Inteligência Artificial

Pode-se tirar vantagens da abordagem *top-down* da Vida Artificial e capturar novas idéias. Entre elas as técnicas de programação com Redes Neurais. Pode-se melhorar a flexibilidade de programas de computadores, ou seja, algoritmos ou sistemas usados para um fim, serem usados com um objetivo diferente. ESTEVAM ainda afirma que, no futuro, a Inteligência Artificial e Robótica serão subgrupos da Vida Artificial.

Jogos e Realidade Virtual

Na área de jogos, é citada a criatividade dos oponentes digitais em situações cada vez mais imprevisíveis. Quanto à Realidade Virtual, a programação de Agentes Inteligentes que auxiliam uma tarefa virtual imprescindível também é auxiliada pela Vida Artificial.

Software Comercial

No artigo, arrisca-se uma previsão de que programas comerciais tais como planilhas de cálculo, editores de texto, bancos de dados, software educacional incorporarão algoritmos de Vida Artificial. O autor afirma tais coisas baseando-se no fato de que muitas tecnologias que eram consideradas descartáveis na época em que foram lançadas, se tornaram indispensáveis atualmente, assim devendo acontecer com os algoritmos de Vida Artificial.

Engenharia

Antes de escolher e implementar um projeto pode-se simulá-lo digitalmente, prevendo sua resposta às situações do possível ambiente em que será implementado. A partir destes testes e seus resultados, podem ser projetadas mudanças nas características deste projeto, observando sua melhora ou piora para o problema proposto.

Tais mudanças, na forma digital para simulação, podem ser acarretadas por Algoritmos Genéticos, um ramo da Vida Artificial, assim como a procura em profundidade por soluções, e sua admissão ou descarte a partir da observação de

seus resultados, que podem se dar a partir do cumprimento ou não de metas de projeto, armazenadas em forma também digital.

Farmácia, Bioquímica, Engenharia Genética e Medicina

O problema de testes de vacinas em seres vivos pode acarretar problemas de contaminação que podem ser catastróficos. Para evitar isso, implementam-se simulações do efeito destas vacinas em programas de Vida Artificial, o que é ecologicamente correto. Cita-se no artigo um exemplo de cientistas treinando enzimas em uma universidade americana. Se conseguirem sucesso, pode-se combinar tais algoritmos com a micro-robótica, obtendo curas para o câncer e *AIDS*.

Será também possível a engenheiros geneticistas criar modelos de vida digitais antes de as criarem com base no carbono.

9 UTILIDADE DO TRABALHO

Tais pesquisas podem ser usadas com a seguinte finalidade:

- Servir como material de consulta para futuros pesquisadores de área afins, ou ainda por entusiastas que se interessem pelo assunto de jogos e geração de Ambientes Virtuais a partir de números pseudo-randômicos, *hardware* para Realidade Virtual e redes de computadores, principalmente em âmbito nacional;
- Disponibilizar no mercado nacional um produto barato e que se destina a outros profissionais do ramo dentro do país, versátil, abrangente, expansível,

de manutenção fácil, estimulando assim a expansão da cultura nacional através de situações criadas pelos usuários desta biblioteca;

- Este produto oferece também a possibilidade de interfaceamento com periféricos de Realidade Virtual, com bons recursos gráficos, também incluindo funções de Inteligência Artificial e comunicação através de Redes de Computadores.

10 PESQUISAS FUTURAS

Apesar de uma gama enorme de assuntos ser abraçada nesta pesquisa, ela jamais estará completa, sempre abrindo pontos específicos para pesquisas mais aprofundadas. Entre os vários temas sugeridos de pesquisa derivados são:

- Geração pseudo-randômica de pistas de corrida baseada em modelos de curvas (arcos e segmentos de reta);
- Modelos otimizados de *rendering* dos Ambientes Virtuais gerados;
- Métodos de previsão de colisão utilizando análise de polígonos;
- Um método de avaliação da eficácia de cada participante;
- Um modelo robusto de transferência de dados confiáveis através da *Internet*;
- Modelos de navegação otimizados utilizando *hardware* de Realidade Virtual.

REFERÊNCIAS BIBLIOGRAFICAS

3D Engines List - <http://cg.cs.tu-berlin.de/~ki/engines.html> Acesso em julho de 2000

3D-Gamers <http://www.3dgamers.com> junho 2001

3dfx Interactive. Inc. <http://www.3dfx.com> Acesso julho 2000

3D Rad – 3D Game Creator – <http://www.3drad.com/> Acesso em julho de 2000

5DT Glove '95 USER'S MANUAL. Version 1.00 - Fifth Dimension Technologies. <www.5dt.com> Acesso em janeiro 1996.

About – 3D-Graphics / Virtual Reality < http://web3d.about.com/ > Acesso em julho de 2000

ActiveWorlds <http://objects.activeworlds.com/game/textures> fevereiro de 2001

Advanced Graphics Programming Techniques Using OpenGL – <http://www.sgi.com/software/opengl/advanced98/notes/notes.html> Acesso em julho de 2000

AGRAWAL, Rakesh; LIN, King-Ip; SAWHNEY, Harpreet S., SHIM, Kyuseok. Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-Series Databases. IBM Almadem Research Center. 650 Harry Road, San Jose, CA 95120. September 1995. Disponível em <http://www.almaden.ibm.com/u/ragrawal/pubs.html> Acesso em novembro de 2000

Al's OpenGL Page – <http://members.net-tech.com.au/alaneb/opengl.html> Acesso em julho de 2000

ANTUNES, Celso. Jogos para Estimulação das Múltiplas Inteligências. Editora Vozes – Petrópolis, RJ. 6ª Ed. 1998.

ATI Technologies Inc. <http://www.ati.com/na/pages/na_index.html> Acesso dezembro 2001 ou <http://www.firegl.com/> Acesso julho 2000

ATIS - Alliance for Telecommunications Industry Solutions < www.atis.org/tg2k/ > Acessado em fevereiro de 2002

AURA INTERACTOR – Virtual Reality Game Wear – Aura Systems, Inc. 2335 Alaska Avenue, El Segundo, CA 90245

BAERT, S. Motion Planning Using Potential Fields. Disponível em <http://www.gamedev.net/reference/programming/features/motionplanning/> Acesso julho 2000

BAIRON, Sérgio. O Discurso do Hipertexto. In: Multimídia. Editora Global - São Paulo. 1995

BARRUS, J.W. WATERS, R.C. ANDERSON, D.B. Locales and Bacons: Efficient and Precise Support for Large Multi-User Virtual Environments. MERL – A MITSUBISHI Electric Research Laboratory. TR95-16a MITSUBISHI ELECTRIC INFORMATION TECHNOLOGY CENTER AMERICA, 1996. Proceedings of VRAIS'96, Santa Clara – CA Disponível em <http://citeseer.nj.nec.com/181294.html> Acesso em abril de 2000

BASTOS, R. C. Uso de Redes Neurais Artificiais para o Estudo de Fatores de Êxito em Empreendedores de Santa Catarina. Inteligência Artificial - Departamento de Informática e Estatística – UFSC - Florianópolis 1998.

Blaxxun Interactive Disponível em <http://www.blaxxun.com> Acesso em julho de 2000

Blizzard Entertainment ®<http://www.blizzard.com> Acesso julho de 2000

Borland C++ Builder 3 for Windows 95 and Windows NT - Developer's Guide. Borland International, Inc., 100 Borland Way P.O. Box 660001, Scotts Valley, CA 95067-0001. 1997

BURDEA, G. RICHARD, P. COIFFET, P. MULTIMODAL VIRTUAL REALITY: INPUT-OUTPUT DEVICES, SYSTEM INTEGRATION, AND HUMAN FACTORS. International Journal of Human Computer Interaction, pp. 5-25. June 1996.

BURDEA, G. C. – FORCE AND TOUCH FEEDBACK FOR VIRTUAL REALITY. Ed, John Willey & Sons, Inc. New York. 1996

ÇAPIN, T. K. PANDZIC, I. S. MAGNENAT-THALMANN, N. THALMANN, D. Avatars in Networked Virtual Environments. John Wiley & Sons Ltd. Baffins Lane, Chichester, West Sussex PO19 1UD, England. 1999

CASUS Presenter - Fraunhofer Institute for Computer Graphics - Darmstadt, Germany <http://www.igd.fhg.de/CP/index.html> Acesso em julho de 2000

Communication Research Centre – Canada<http://www.crc.ca/FreeWRL/> Acesso em julho de 2000

Cosmo Software – A Computer Associates Company – <http://www.cai.com/cosmo/> Acesso em julho de 2000

Cyberlife Technology Ltd. © Creatures 3 <http://www.creatures3.com> Acesso junho de 2000

DAMER, B. Avatars! Exploring and Building Virtual World on the Internet. PeachPit Press. 1249 Eighth Street, Berkeley, CA 94710.

DUDA, R. O. & HART - PATTERN CLASSIFICATION AND SCENE ANALYSIS. Caps. 7,8 e 9. John Willey & Sons, 1973.

ECKERT, W. Understanding of Spontaneous Utterances in Human-Machine-Dialog. Lehrstuhl für Mustererkennung (Informatik 5), Universität Erlangen-Nürnberg, Martensstraβe 3, D-91058 Erlangen, Germany - 1996

EDWARDS, J. Voice-Based Interfaces Make PCs Better Listeners. Revista COMPUTER – Innovative technology for computer professionals. IEEE Computer Society. 1997. Pp14<http://computer.org>

EMUMANÍACOS < http://www.emumaniacos.com.br/ > Acesso em fevereiro de 2002

ESTEVAM, Marcos Rogério. VIDA ARTIFICIAL – A Informática Imita a Natureza. REVISTA CPUPC. Ano 03, No. 13. Bônus Rio Editora Ltda. Caixa Postal 11750 – Rio de Janeiro – RJ – CEP 22022-970 1997

ESTEVES, B. Imersão Total – Revista Superinteressante. No. 5 - ANO 14 – maio de 2000

FARUM-BADLEY, W. L. Barbados Warri: the Bajan Way <http://barbadosphotogallery.com/warri/warrigame.htm> Acessado em janeiro de 2002

FAUSETT, Laurene V. Fundamentals of Neural Networks. Architeture, Algoritms and Aplicattions. Prentice Hall International. Inc. Englewood Cliffs. New Jersey 07632 – 1994.

Fighting Bull Technologies LLC.<http://www.dpiv.com/> Acesso em julho de 2000

Game Commander – Voice Control for Games & Simulations <http://www.gamecommander.com> Acesso maio de 2000 - 224 Airport Parkway #550 San Jose, CA 95110 - U.S.A.

GameSpot - The History of VideoGames < http://www.videogames.com/features/universal/hov/index.html > Acessado em janeiro de 2002

GARCIA, F. L. S., BETTIO, R. W.
Movimentação de Braços Robóticos em VRML utilizando uma linguagem de programação X-ARM In: 4th SBC Symposium on Virtual Reality, 2001, Florianópolis. SVR 2001 - 4th Symposium on Virtual Reality. Florianópolis: UFSC, 2001. v.1. p.202 – 210

GOLDBERG, D. E. Genetic Algorithms in Search, Optimization & Machine Learning. Addison Wesley Publishing Company, Reading – Massachussets. March 1998.

GRADECKI, Joe. KIT DE MONTAGEM DA REALIDADE VIRTUAL. Tradução Josué Vieira. – São Paulo. Editora Berkeley, 1994.

GRAND, Stephen. IEEE EXPERT – Intelligent Systems & Their Applications.– IEEE Computer Society, volume 12, no. 4. pp. 15-24. JULY/AUGUST 1997

GUIBAS, L. J. HSU, D., ZHANG, Li. A Hierarchical Method for Real-Time Distance Computation among Moving Convex Bodies. Department of Computer Science – Stanford University – Stanford – CA 94305 – USA
< http://citeseer.nj.nec.com/259909.html > Acesso em janeiro de 2002

HALF-LIFE <http://www.sierra.com/games/half-life/> Acesso em maio de 2000

HARRIS, S. et al. CYBERLIFE. Ed. SAMS Publishing & Berkeley do Brasil. Av. Raimundo Pereira de Magalhães, 3305. 05145-200 São Paulo – SP 1994

HART, SAM. A Brief History of Video Games –<http://www.geekcomix.com/vgh/> Acessado em maio de 2001

HARTMAN, J. WERNECKE, J. The VRML 2.0 Handbook. Building Moving Worlds on the Web. Silicon Graphics Inc. Addison Wesley Publishing Company – Reading – Massachussets. August 1996

HAVOK Realtime Interactive Physics < www.havok.com > Acessado em fevereiro de 2002

HAYKIN, Simon. Neural Networks. A Comprehensive Foundation. pp. 138-157. IEEE Computer Society Press. McMilan Publishing Company – 113 Sylvan Avenue, Englewood Cliffs, NJ 07632. 1998

HEETER, Carrie. Why Play (VR) Games?.
http://commtechlab.msu.edu/randd/research/whyplay.html Março 1994. Acessado em Junho de 1999

HEIM, M. Virtual Realism. Oxford University Press. New York, Oxford. 1998

HOUAISS, Antônio, VILLAR, Mauro de Salles. Dicionário Houaiss da Língua

Portuguesa/ Antônio Houaiss e Mauro de Salles Villar, elaborado no Instituto Antônio Houaiss de Lexicografia e Banco de Dados da Língua Portuguesa S/C Ltda. – Rio de Janeiro, Objetiva, 2001

HUNT, C. TCP/IP Network Administration. Help for UNIX Systems Administrators. O'Reilly & Associates, Inc. 101 Morris Street, Sebastopol, CA 95472. 1998

HUONG, Q. D. WALKER, N. SONG, C. KOBAYASHI, A. HODGES, L. F. Evaluating the Importance of Multi-Sensory Input on Memory and the Sense of Presence in Virtual Environments. Graphics, Visualization and Usability Center. Georgia Institute of Technology (Georgia Tech) Atlanta, GA. 30332 USA. IEEE, 1998

IBM Voice Systems <http://www-4.ibm.com/software/speech/> maio de 2000

IBMHF - International Bowling Museum and Hall of Fame. History of Bowling. <http://www.bowlingmuseum.com/history.html> Acessado em janeiro de 2002

Id Software © 2001 <http://www.idsoftware.com/> junho de 2000

IEEE Computer Society / Intelligent Systems <http://computer.org/pubs/expert/expert.htm> Acesso junho de 2000

Infogrames <http://us.infogrames.com> Acesso dezembro 2001 ou <http://www.gtinteractive.com/> Acesso junho de 2000

Interactive Imaging Systems <http://www.iisvr.com/> Acesso setembro de 2002

International Centre for Computer Games and Virtual Entertainment <http://www.iccave.com> janeiro 2001

Jet 3D – <http://www.jet3d.com/> Acessado em julho de 2000

JIMÉNEZ, P., THOMAS, F., TORRAS, C. 3D Collision Detection. A Survey. Institut de Robótica i Informática Industrial (CSIC - UPC), Gran Capitá 2-4, (Ed. Nexus), 08034 – Barcelona – Espanha. Abril de 2000. < http://citeseer.nj.nec.com/431815.html > Acessado em janeiro de 2002

JR. WRIGHT, Richard S. & SWEET, Michael. OpenGl SuperBible. Waite Group Press - Division of Sams Publishing - Corte Madera - CA. 1996

K2I Interactive <http://www.k2i.com> Acessado em junho 2000

LARIJANI, L. C. The Virtual Reality Primer. Ed. McGraw-Hill, Inc. New York. 1993 ISBN 0-07-036417-6

Le Jargon Français. http://www.linux-france.org/prj/jargonf/ Acesso em agosto de 2002

LÉVY, Pierre. O que é o Virtual? São Paulo, Editora 34, 1996.

LÉVY, Pierre. As Tecnologias da Inteligência. Rio de Janeiro: Editora 34, 6ª edição, 1996.

LUBAN, Pascal. Turning a Linear Story into a Game: The Missing Link Between Fiction and Interactiva Entertainment. Gamasutra. Disponível em http://www.gamasutra.com/features/20010615/luban_01.htm Acesso em junho 2001

Luz, Computador, Ação! Revista Superinteressante – No. 10, Ano 14 – Outubro de 2000. Ed. Abril. Brasília-DF. Rio de Janeiro-RJ. ISSN 0104-1789. Pp72-76.

MACEDO, L. PETTY, A. L. S., PASSOS, N. C. Aprender com Jogos e Situações Problema. Porto Alegre: Artes Médicas Sul, 2000.

MARRIN, Chris & CAMPBELL, Bruce. Teach Yourself VRML 2 in 21 Days. Sams.net Publishing - 201 West 103rd Street - Indianapolis, Indiana 46290. 1997

MCREYNOLDS, T. BLYTHE, D. Advanced Graphics Programming Techniques Using OpenGL www.sgi.com/software/opengl/advanced98/notes/notes.html Acesso em julho de 2000

MERINO, E.; GONTIJO, L.; PINTO DA LUZ, R; MARTINS, A.; BARCIA, R. A New Tool to Support the Ergonomics Analysis of the Work - Virtual Observation System (V.O.S) Proceedings of the 5th Pan-Pacific Conference on Occupational Ergonomics, p.88-90, 1998, Kitakyushu - Japan.

Meta Motion, Motion Capture Distribution, Sales and Consulting - 268 Bush St. #, San Francisco, CA 94104 <www.metamotion.com/> Acesso outubro 2000

Microsoft <http://www.microsoft.com> Acesso junho 2000

Microsoft Product and Technology Catalog <http://www.microsoft.com/catalog/> - link Games / Flight Simulator 2000 Acessado em março 2000.

MITCHELL, Willian J. & McCULLOUGH, Malcolm. Digital Media Design – A Handbook for Architects & Design Professionals – Cap. 14 – pp. 313-324. Editora Van Nostrand Reinhold – 115 Fifth Avenue, New York – NY. 10003 1997

NEIDER, J. DAVIS, T. OpenGL Programming Guide. Ed. Addison-Wesley. 1997. Disponível em <http://ask.ii.uib.no/ebt-bin/nph-dweb/dynaweb/SGI_Developer/OpenGL_PG/> Acesso em agosto de 2002

Neon Helium Productions! <http://nehe.gamedev.net/> Acesso em julho de 2000

NETO, Hugo Vieira, BORGES, Díbio Leandro – Fingerprint Classification with Neural Networks. Escola de Engenharia Elétrica. Universidade Federal de Goiânia. Praça Universitária s/n – Setor Universitário, 74605-220, Goiânia – GO, Brasil.

Netscape®<http://www.netscape.com> Acesso junho 2000

Nine Revolution 4 <http://www.nine.com/> Acesso julho de 2000

OpenGL® – The Industry Foundation for High Performance Graphics <http://www.opengl.org/> Acesso em julho de 2000

OpenWorlds – Technology for Building Immersive Web3D / MultiMedia Applications <http://www.openworlds.com/> Acesso em julho de 2000

PANDYA, Abhijit S. & MACY, Robert B. - PATTERN RECOGNITION WITH NEURAL NETWORKS IN C++. Cap. 3. CRC Press & IEEE Press, 1996.

Parallel Graphics http://www.parallelgraphics.com/ Acesso em julho de 2000

PATEL, A. J. GAME PROGRAMMING INFORMATION. http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html e http://www-cs-students.stanford.edu/~amitp/gameprog.html Acesso em julho de 2000

PAUSCH, R., Gossweiler, R., Laferriere, R. J and Keller, M. L. (1994, Fall). AN INTRODUCTORY TUTORIAL FOR DEVELOPING MULTI-USER VIRTUAL ENVIRONMENTS. Presence, Teleoperators and Virtual Environments, 3(4), 255- 264.

PELL B. et. al. An Autonomous SpaceCraft Agent Prototype. Caelum Research Corporation. 1998. NASA Ames Research Center, MS 269/2 – Moffet Field, CA 94035. NEC Research Index – < http://citeseer.nj.nec.com/ > Acessado em maio de 2002

PIAGET, Jean. *Psicologia e Pedagogia.* Rio de Janeiro, Ed. Forense, 1988.

Planet Quake – The Epicenter of Everything Quake <http://www.quakeworld.com> Acesso junho de 2000

PlayStation.com Europe<http://eu.playstation.com/europe/europe_select.jhtml> Acesso em dezembro de 2001 ou <http://www.psygnosis.com> Acesso em fevereiro de 2001

POSTEL, J. Internet RFC/STD/FYI/BCP Archives – RFC768, em 28 de agosto de 1980 < http://www.faqs.org/rfcs/rfc768.html > Acessado em fevereiro de 2002

RADINSKY, Wayne (waynerad@oz.net) Interview with Nolan Bushnell <http://www.sjmercury.com/revolutionaries/bushnell.htm> Acessado em junho 2001

REBELO, I. B., PINTO DA LUZ, R. New Technologies Helps to Enhance the Knowledge. Proceedings of the 4th International Conference on Virtual Systems and Multimedia: Future Fusion - Application Realities for the Virtual Age,Vol. One - p. 286-291, Gifu, Japan, 1998

RIPLEY, B. D. (1987). Stochastic Simulation. Ed. JOHN WILLEY & SONS, New York

ROEHL, B., COUCH, J., REED-BALLREICH, C., BROWN, G. Late Night VRML 2.0 with JAVA. P90. Ziff-Davis Press. An Imprint of Macmillan Computer Publishing USA. 5903 – Christie Avenue - Emeryville, CA 94608. 1997

ROLE MAKER – A Computer Roleplaying Game Authoring System. A Computer Science Master's Thesis by Jacob Marner < http://www.rolemaker.dk > Acessado em janeiro de 2002

ROMKA DOCS – <http://romka.demonews.com/opengl/doc/index_eng.htm> Acesso em julho de 2000

ROSA JR, O. Ambientes Virtuais Cooperativos LRVCHAT3D, Um Estudo de Caso In: 4th SBC Symposium on Virtual Reality, 2001, Florianópolis. SVR 2001 - 4th Symposium on Virtual Reality. Florianópolis: UFSC, 2001. v.1. p.01 - 11

RUMBAUGH, J. et al. Modelagem e Projetos Baseados em Objetos. Tradução de Dalton Conde de Alencar. – Rio de Janeiro – Editora Campus, 1997.

SCHILDT, H. Turbo C++: Guia do Usuário. Makron Books do Brasil Ltda.– São Paulo – SP, 1992

SHINY ENTERTAINMENT ™ < http://www.shiny.com/games/mdk/> Acesso dezembro de 2001

SIMS, Karl. Evolving Virtual Creatures – Thinking Machines Corporation. Computer Graphics, Annual Conference Series, (SIGGRAPH '94 Proceedings), July 1994, pp.15-22. URL: http://genarts.com/karl/evolved-virtual-creatures.html

STAR, J. & ESTES, J. Geographic Information Systems - An Introduction. University of California - Santa Barbara. PRENTICE HALL, Englewood Cliffs, New Jersey – 07632. ISBN 0-13-351123-5 1990

STEED, A. A Survey of Virtual Reality Literature – Department of Computer Science – Queen Mary & Westfield College – Mile End Road – London – E1 4NS, 1993 – NEC Research Index – < http://citeseer.nj.nec.com/ > Acessado em fevereiro de 2002

STEIL, A. V. & BARCIA, R. M. Aspectos Estruturais das Organizações Virtuais. Submetido ao ENAMPAD99, Foz do Iguaçú, 19 a 22 de setembro, 1999.

STURMAN. David J. Computer Puppetry. IEEE Computer Graphics and Applications.. pp38-45. IEEE Computer Society 10662 Los Vaqueros Circle PO Box 3014 Los Alamitos, CA 90720 January/February 1998

Sun Microsystems - Ivan E. Sutherland - Patents and Publications <http://www.sun.com/960710/feature3/ivan-publish.html>

SUTHERLAND, I. The Ultimate Display. In: Information Processing. New York, NY, USA. Proceedings of New York, NY, USA: IFIP. p.506-508. 1965

Systems in Motion – Expertise in 3D-Graphics Visualization<http://www.sim.no/> Acesso em julho de 2000

T1 Glossary 2000 < www.atis.org/ > Acessado em fevereiro de 2002

TAFNER, Malcon Anderson – Reconhecimento de Palavras Faladas Isoladas Usando Redes Neurais Artificiais. Trabalho submetido à Universidade Federal de Santa Catarina. Departamento de Engenharia de Produção - UFSC - Florianópolis 1996 <http://www.eps.ufsc.br/disserta96/tafner/index/index.htm> Acessado em junho de 2001

TEXTURE WORLD TEXTURES <http://www.textureworld.com> Acessado em fevereiro de 2001

THE DOT EATERS – Video Game History 101 <http://www.emuunlim.com/doteaters/index.htm>Acessado em janeiro de 2002

THE VIRTUAL REALITY MODELING LANGUAGE – 6.Node Reference – ISO/IEC DIS 14772-1, 4 de abril de 1997 <http://coli.lili.uni-bielefeld.de/~milde/vrml+java/vrml2/nodesRef.html#Anchor> Acessado em janeiro de 2002

The VRwave Home Page <http://www.iicm.edu/vrwave> Acesso em julho de 2000

TODESCHINI, R. T.,STRADIOTTO, C. R. K. PEREIRA, A. T. C., Jogos e Entretenimentos em Realidade Virtual: Perspectivas e Possibilidades. 14° Simpósio Nacional de Geometria Descritiva e Desenho Técnico (Graphica 2000) – 5 a 9 de junho de 2000 - Ouro Preto – MG.

TRAUER, E.; PINTO DA LUZ, R. Virtual Lab: Ensino Através de Laboratórios Virtuais. Anais do 1° Workshop de Realidade Virtual, p.130-137, 1997, São Carlos - São Paulo, Brasil.

VASULKA, W. "Lee Harrison III", Pioneers of Electronic Art, exhibition catalog for Arts Electronica 1992, D. Dunn, ed., Ars Electronica, Linz, Austria, 1992, pp.92-95

VENKATRAMAN, N. & HENDERSON, J. Real Strategies for Virtual Organizing. Sloan Management Review, v.40, n.1, p33-48, Fall 1998

VIO <http://www.vio.com/intro.htm> Acesso julho de 2000 - 500 North Gulph Road - Suite 220 - King of Prussia - PA 19406 – USA. Tel: 610 992 5788 Fax: 610 992 5799.

WALNUM, Cleyton. Borland C++ 4.x – Dicas, Segredos e Armadilhas. Axcel Books do Brasil, 1994 – Rua da Glória, 344 – 10o andar – 20241-180 – Rio de Janeiro – RJ

WATSON, Mark. AI Agents in Virtual Reality Worlds. New York-EUA. John Wiley & Sons, Inc. ISBN 0-471-12708-6 1996

Web 3D Consortium <http://www.vrml.org> março de 2000

Webster´S Third New Dictionary Of The English Language Unabridged. Ed. Könemann Verlagsesellschaft Mbh. 1993. Bonner Straβe 126 D-50968 Cologne. ISBN 3-8290-5292-8.

WEISS, A. E. VIRTUAL REALITY – A Door to CyberSpace. Twenty First century Books. A Division of Henry Holt and Company, Inc. 115 West 18th Street – New York, NY 10011. ISBN 0-8050-3722-5 1996

Westwood Studios http://www.westwood.ea.com Acessado em janeiro 2001

WhatIs?com< http://whatis.techtarget.com/ > Acessado em fevereiro de 2002

WOODCOCK, Steve, Game AI: The State of the Industry. Gamasutra, November 20, 1998. Vol. 2, Issue 46 Published in Game Developer Magazine, October 1998<http://www.gamasutra.com/features/programming/19981120/gameai_01.htm> Acessado em outubro de 2000

ANEXO I - PESQUISA DE CAMPO E RESULTADOS

Metodologia de Pesquisa

Foram realizadas pesquisas de campo com o uso de entrevista pessoal em dois fliperamas da cidade de Florianópolis - Santa Catarina: Beira-Mar Play (localizado no Shopping Beira-Mar) e Star Games (rua Trajano, Centro).

Para realizar a entrevista foram escolhidas pessoas que estivessem em atividade em jogos de RV. A entrevista foi baseada nas seguintes questões:

- Idade:
- Sexo: M( ) F( )
- Grau De Instrução: 1° ( ) 2° ( ) 3° ( ) Grau
- Salário: Até 2 salários mínimos( ) / 3 A 5( ) / 6 A 8 ( ) / 9 A 11 ( ) / Mais De 11 ( ) /Mesada De R$_____ Por Semana
- Quanto você gastaria em jogos por semana? R$_____
- Qual jogo que você mais gosta? Por que?

- Você gostaria de um jogo em que:
  a) Você tivesse que pensar para resolvê-lo?
  b) Houvesse um amigo que falasse c/ você?
  c) O computador criasse um novo mundo cada vez que o jogo inicia?

A amostra de pessoas pesquisadas é composta de 80 (oitenta) elementos, considerados como suficientes em vista dos resultados positivos obtidos. Foram entrevistados clientes de duas lojas de fliperama na cidade de Florianópolis – Santa Catarina para tomar as amostras. A pesquisa foi feita através de um questionário direto feito pessoalmente com cada um dos entrevistados.

Dos resultados destas amostras, descobriu-se que existiam três necessidades a serem satisfeitas no ramo de jogos: A grande maioria dos clientes desejava a criação de um jogo em que, a cada início, oferecesse um novo mundo e uma nova situação a ser resolvida pelo jogador. Grande parte também gostaria muito de jogos em que se usasse o raciocínio para resolver pequenos problemas, tais como montagem de peças em automóveis em jogos de detetive, ou que se descobrisse como abrir uma porta sem as chaves, e coisas do gênero. Para a terceira questão, sobre a existência de um avatar que permitisse um diálogo com o jogador, aceitação também foi muito grande.

Mais detalhes da metodologia ver em (TODESCHINI, 2000). Os resultados encontrados foram os seguintes:

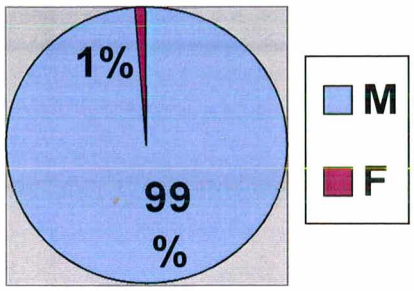


Fig. A1: Sexo dos Jogadores

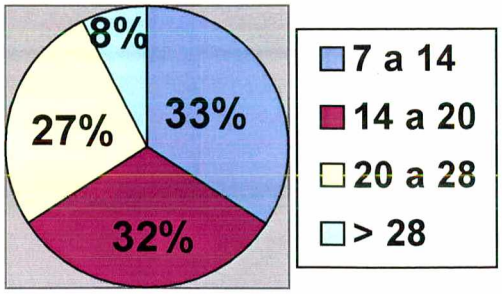


Fig. A2: Faixa Etária dos Jogadores

Através da Figura A1 observa-se que a maior parte dos clientes de jogos presentes em lojas de fliperamas é do sexo masculino. Tal observação confirma a hipótese feita por (HEETER, 1994), de que mulheres estão menos interessadas em violência e competição. Da Figura A2 observamos que a grande maioria dos usuário de jogos de fliperama tem idade entre sete e vinte anos, sendo um público ainda jovem.

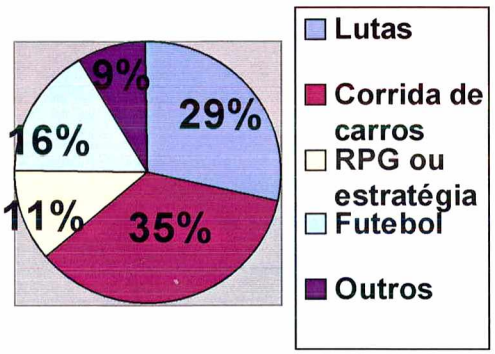


Fig. A3: Preferência por estilos de jogos

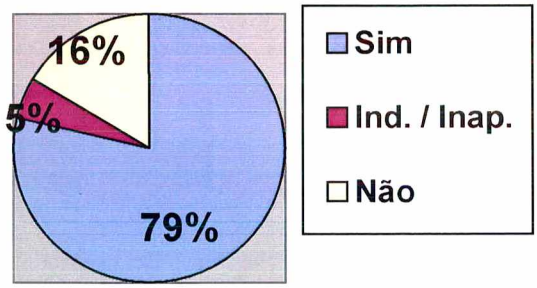


Fig. A4: Preferência por jogos de raciocínio

A Figura A3 mostra que, dos cinco tipos de jogos identificados dentro das lojas, o estilo de jogo mais procurado foi o de corrida de carros, seguido pelos jogos de luta, futebol e RPG. A Figura A4 mostra que grande parte dos entrevistados gostariam de jogos em que houvessem pequenos problemas a serem resolvidos usando o raciocínio.

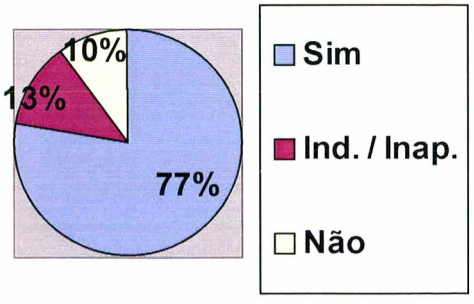


Fig. A5: Simpatizantes de jogos que geram novos mundos

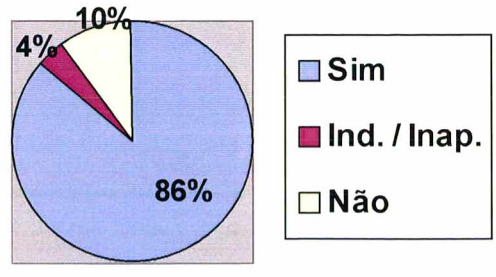


Fig. A6: Preferência por diálogos virtuais

A Figura A5 mostra que também uma grande parte dos entrevistados gostaria de jogos em que a cada início tivessem novos ambientes a oferecerem desafios aos jogadores, e a Figura A6 mostra a preferência por diálogos com personagens virtuais dentro do jogo.

Tais resultados foram julgados suficientes para que se procurasse novas tecnologias que satisfizessem as necessidades aí demonstradas, e a partir destes resultados iniciou-se a procura por dispositivos que possibilitassem a comunicação e interação com o computador, o estudo de algoritmos que pudessem criar mundos randômicos, e algoritmos que oferecessem problemas que exigissem o raciocínio para serem resolvidos. Mais detalhes sobre a escolha destas características estão descritos no capítulo **4 APLICAÇÃO E ANÁLISE,** seção **4.3 Tema e Modelo do Jogo.**

Dentro dos resultados encontrados na pesquisa, descobriu-se que os motivos que levavam as pessoas entrevistadas a jogarem jogos de computador eram o realismo do jogo, a variedade de situações e personagens, diversão, relaxamento, juntar amigos, desafio, nacionalismo, fuga da realidade, violência, uso de mágicas e ação.

O uso de equipamentos de realidade virtual se justifica pelo aumento do realismo do jogo, motivo dado por uma boa parte dos entrevistados. Pode-se deduzir desta pesquisa que um jogo se torna mais interessante à medida em que o usuário possui mais canais de interação entre ele e o mundo, tais que facilitem e permitam o detalhamento de suas atividades dentro deste mundo. Estes canais de interação seriam dados pelo *hardware* de interação com o computador, tal como o teclado, *mouse*, luva digital e microfone. No caso do uso do microfone, é necessário que o computador compreenda algumas palavras para que o diálogo se efetue. Para possibilitar esta compreensão foi implementada uma rede neural artificial.

O motivo de "juntar amigos" justifica a implementação do jogo em rede de computadores, fazendo com que um grupo de jogadores possam agir cooperativamente para resolverem um dado problema.

O motivo da "variedade de situações e personagens" justifica mais uma vez a implementação de jogos que criem novos mundos e situações.

Por último, o motivo do desafio oferecido pelo jogo, alegado também por uma boa parte dos entrevistados, exige a implementação de agentes inteligentes que tornem interessante e desafiadora a prática do jogo descrito aqui.

ANEXO II – ALGORITMO DE *PATHFINDER*

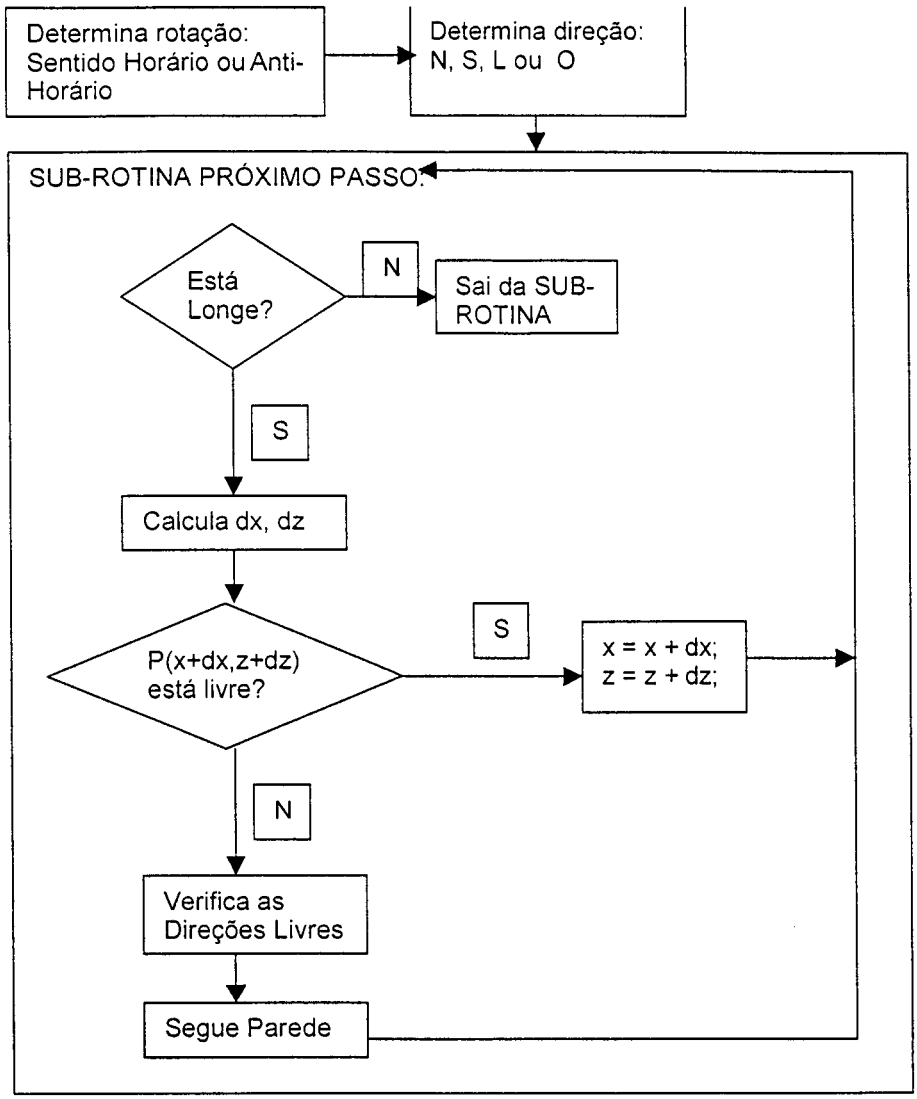


Fig. B1: Fluxograma principal de um algoritmo simples de procura por trajetórias.

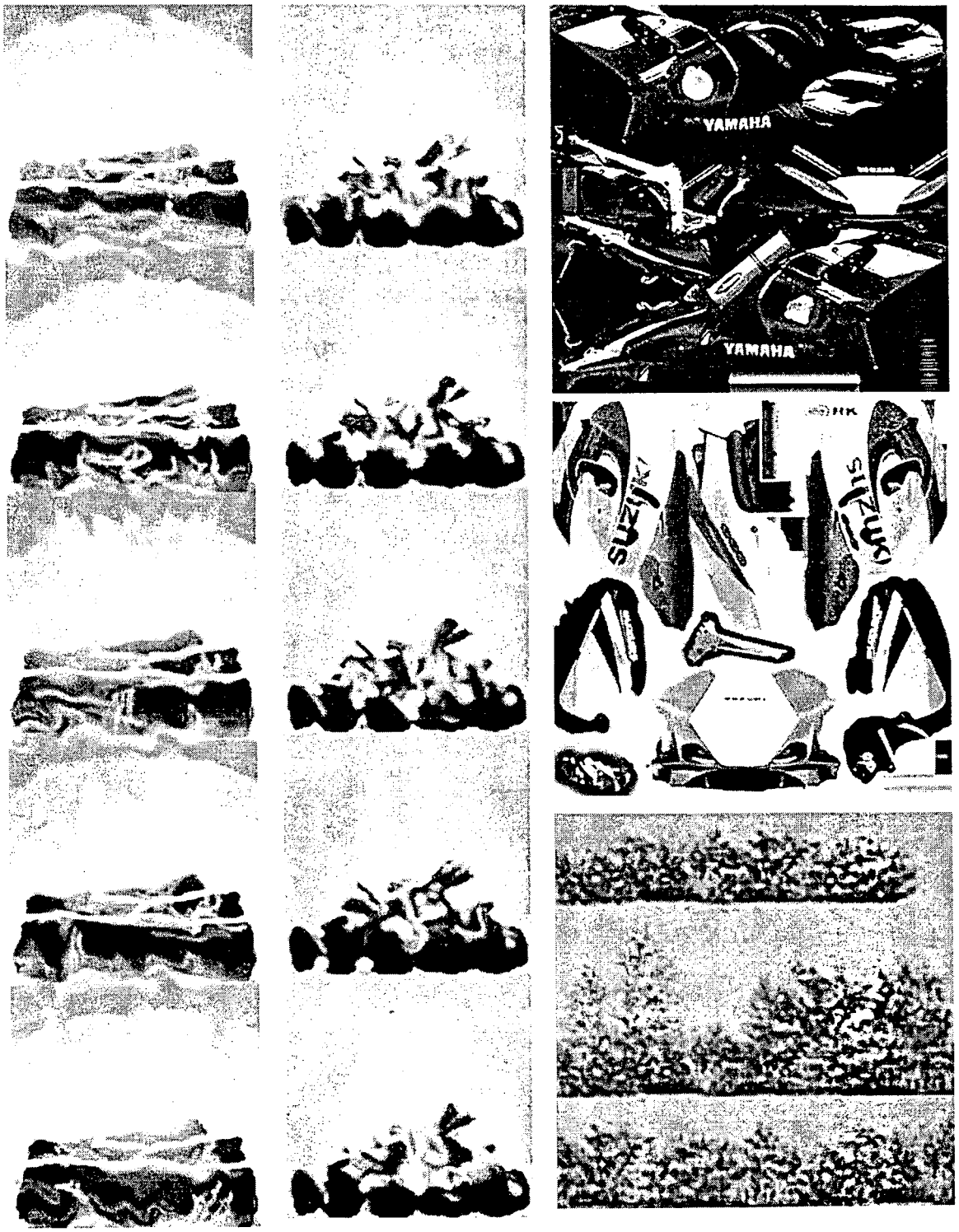
```
SUB-ROTINA DIREÇÕES LIVRES:

LIVRE[NORTE...LESTE]=FALSE;

se NORTE==LIVRE então LIVRE[NORTE]=TRUE;
se SUL==LIVRE então LIVRE[SUL]=TRUE;
se OESTE==LIVRE então LIVRE[OESTE]=TRUE;
se LESTE==LIVRE então LIVRE[LESTE]=TRUE;
```

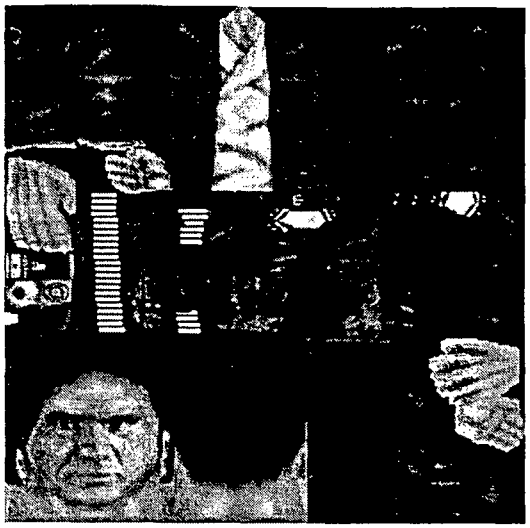
Fig. B2: Sub-rotina que indica quais as direções livres de obstáculos para o personagem.

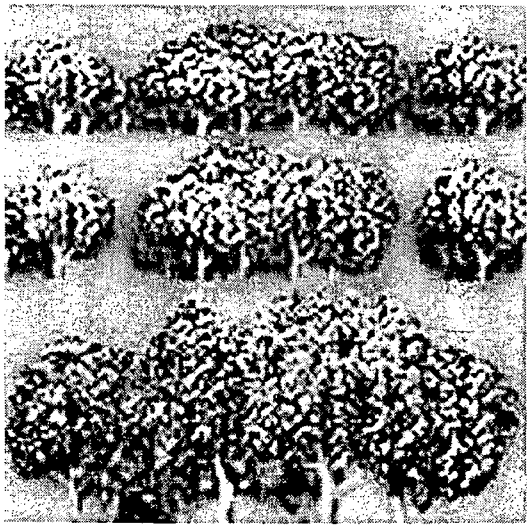
```
SEGUE PAREDE:

se SENTIDO HORARIO:
            se DIRECAO LESTE:
          se  livre[LESTE] e !livre[SUL]  vai para LESTE;
          se  livre[LESTE] e  livre[SUL]  {direcao=SUL; vai para SUL;}
          se  !livre[LESTE] e !livre[SUL]  direcao=NORTE;
          se  !livre[LESTE] e livre[SUL]  direcao=SUL;
          fim;
            se DIRECAO OESTE:
          se  livre[OESTE] e !livre[NORTE] vai para OESTE;
          se  livre[OESTE] e  livre[NORTE]  {direcao=NORTE; vai para NORTE;}
          se  !livre[OESTE] e !livre[NORTE]  direcao=SUL;
          se  !livre[OESTE] e livre[NORTE]  direcao=NORTE;
          fim;
       se DIRECAO NORTE:
          se  livre[NORTE] e !livre[LESTE] vai para NORTE;
          se  livre[NORTE] e  livre[LESTE]  {direcao=LESTE; vai para LESTE;}
          se  !livre[NORTE] e !livre[LESTE]  direcao=OESTE;
          se  !livre[NORTE] e livre[LESTE]  direcao=LESTE;
          fim;
       se DIRECAO SUL:
          se  livre[SUL] e !livre[OESTE] vai para SUL;
          se  livre[SUL] e  livre[OESTE]  {direcao=OESTE; vai para OESTE;}
          se  !livre[SUL] e !livre[OESTE]  direcao=LESTE;
          se  !livre[SUL] e livre[OESTE]  direcao=OESTE;
          fim;
fim;
se SENTIDO ANTIHORÁRIO:
       se DIRECAO LESTE:
          se  livre[LESTE] e !livre[NORTE] vai para LESTE;
          se  livre[LESTE] e  livre[NORTE]  {direcao=NORTE; vai para NORTE;}
          se  !livre[LESTE] e !livre[NORTE]  direcao=SUL;
          se  !livre[LESTE] e livre[NORTE]  direcao=NORTE;
          fim;
            se DIRECAO OESTE:
          se  livre[OESTE] e !livre[SUL] vai para OESTE;
          se  livre[OESTE] e  livre[SUL]  {direcao=SUL; vai para SUL;}
          se  !livre[OESTE] e !livre[SUL]  direcao=NORTE;
          se  !livre[OESTE] e livre[SUL]  direcao=SUL;
          fim;
       se DIRECAO NORTE:
          se  livre[NORTE] e !livre[OESTE] vai para NORTE;
          se  livre[NORTE] e  livre[OESTE]  {direcao=OESTE; vai para OESTE;}
          se  !livre[NORTE] e !livre[OESTE]  direcao=LESTE;
          se  !livre[NORTE] e livre[OESTE]  direcao=OESTE;
          fim;
            se DIRECAO SUL:
          se  livre[SUL] e !livre[LESTE] vai para SUL;
          se  livre[SUL] e  livre[LESTE]  {direcao=LESTE; vai para LESTE;}
          se  !livre[SUL] e  !livre[LESTE]  direcao=OESTE;
          se  !livre[SUL] e  livre[LESTE]  direcao=LESTE;
          fim;
fim;
```

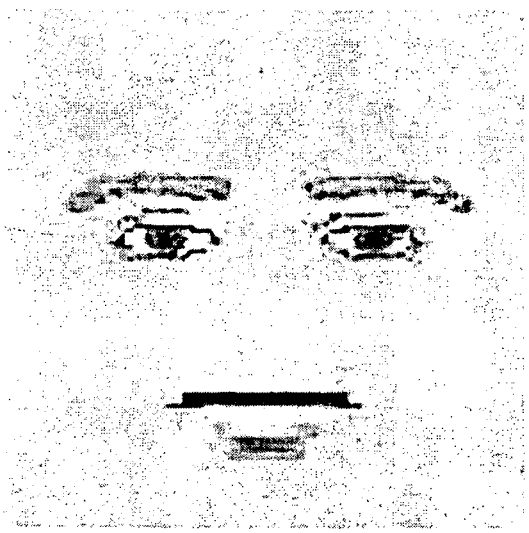
Fig. B3: Sub-rotina de decisão sobre a direção a tomar.

ANEXO III - ALGUNS EXEMPLOS DE TEXTURAS OTIMIZADAS (ACTIVEWORLDS, 2001)









ANEXO IV – GLOSSÁRIO

Nota: Devido a este glossário oferecer apenas algumas definições introdutórias, recomenda-se, para maiores detalhes, a procura pela fonte original das definições, cujas referências estão na bibliografia desta dissertação.

A

Acelerador gráfico – Um acelerador gráfico (um circuito integrado (*chipset*) acoplado a uma placa de vídeo) é um microcomponente eletrônico responsável por atualizar e enviar imagens para o monitor do computador, e também por computar efeitos especiais comuns a imagens 2D e 3D.
Aceleradores Gráficos aumentam a velocidade de apresentação de imagens no monitor, tornando possíveis efeitos visuais anteriormente inviáveis – por exemplo, a apresentação de imagens muito grandes ou em jogos interativos cujas imagens precisam mudar rapidamente em resposta às entradas do usuário. Novos computadores pessoais são vendidos com placas aceleradoras já incluídas. A capacidade de um acelerador gráfico pode ser extendida se o computador pessoal é equipado com a Porta de Gráficos Acelerados (*Accelerated Graphics Port* (*AGP*)), uma interface de barramento entre os componentes do computador envolvidos com a apresentação da imagem no monitor.
Cada acelerador gráfico provê uma interface de programação de aplicações (API). Alguns suportam mais que uma API. Entre as *API's* mais populares estão a *OpenGL* (padrão industrial) e a *DirectX* e *Direct3D* da *Microsoft*. (WhatIs?com, 2002).

Adventure – ADVENTURE – Chance, fortuna, a ocorrência de uma chance, um evento não planejado, uma chance de perda. Um empreendimento ou performance envolvendo incerteza ou mudanças desconhecidas, um encontro de riscos, um empreendimento ou experiência perigosa ou excitante. Uma experiência nova, excitante ou marcante. (WEBSTER'S THIRD NEW DICTIONARY, 1993).
Na área de jogos de computadores, um *adventure* é um jogo de computador pessoal, ou *videogame*, jogado sozinho, em que o usuário deve cumprir um objetivo, e que ocorre em um ambiente fictício produzido pela máquina. A interação com a maquina é feita com diálogos simples, feitos com palavras-chave, digitados a partir do teclado, ou ainda através de ações disparadas por objetos icônicos, apontados pelo cursor do *mouse*.

Âncoras – Na linguagem *VRML*, são *hyperlinks* para outros tipos de mídia armazenados em *URL's*. Assim como a linguagem *HTML* possui *hyperlinks* para outras páginas na *Internet*. Estas páginas ou mídias estão armazenadas em *URL's*. (MARRIN, 1997).

Avatar(es) – Em jogos 3D ou de Realidade Virtual, e em alguns fóruns de *Chat* na *Internet*, o seu avatar é o "cabeça" ou aparência que você usa para se representar. Em muitas páginas de *Chat* virtual, você pode ser um unicórnio, um pássaro, ou qualquer tipo de criatura que lhe seja interessante.
Na religião hindu, um avatar é a encarnação de uma divindade, ou ainda a manifestação de uma idéia ou de uma realidade mais avançada. (WhatIs?com, 2002).

API – Uma interface de programação de aplicações (*API*) é um método específico prescrito por um sistema operacional, ou por um programa aplicativo, pelo qual um programador que esteja criando uma outra aplicação possa fazer requisições.
Uma *API* pode ser contrastada com uma interface gráfica para o usuário, ou uma interface de comando (ambas são interfaces para usuário). Esta interface é um meio caminho entre o usuário e o programa ou sistema operacional. (WhatIs?com, 2002).

B

Bit map – Um *bitmap* (ou mapa de bits) define um espaço de tela e a cor de cada ponto (chamado de *pixel* ou *bit*) dentro deste espaço. Os formatos *GIF* (*Graphics Interchange Format*) e *JPEG* (::) são exemplos de tipos de arquivo de imagens gráficas que contêm *bitmaps*.
Um *bitmap* não precisa conter um bit de informação em formato de cor para cada *pixel* em cada linha. Ela somente precisa conter informação indicando uma nova cor à medida que o processo de exibição da imagem for rastreando ao longo da linha. Então uma imagem com uma cor muito sólida tende a requisitar um *bitmap* bem pequeno. Devido ao fato de um *bitmap* usar um método de mapa *raster* para descrever uma imagem, esta não pode ser mostrada em escala sem perder definição. Uma imagem armazenada em mapa vetorial, do contrário, é projetada para ser rapidamente redesenhada em escala. Tipicamente, uma imagem é criada usando gráficos vetoriais, e depois é convertida em formato *bitmap* (*raster*). (WhatIs?com, 2002).

Blur – Uma operação de *blur* (*blurring*) é um filtro espacial. Ela reduz ou elimina aspectos de alta freqüência de uma imagem. Normalmente usam-se matrizes de média-aritmética. (McREYNOLDS, 2001).

Bot – Contração da palavra *robot*, é um programa que opera como um agente para um usuário ou que simula uma atividade humana. (WhatIs?com, 2002).

Byte – Em muitos sistemas de computadores, um *byte* é uma unidade de dados que possui oito unidades de dígitos binários. Um *byte* é a unidade que muitos computadores usam para representar um caracter tal como uma letra, um número ou um símbolo topográfico (por exemplo , "g", "5", ou "?"). Um *byte* pode preceder um cordão de *bits* que precisam ser usados em grandes unidades para aplicações (por exemplo, um fluxo de *bits* que constituem uma imagem ou um programa de computador sendo transferidos pela *Internet*). (WhatIs?com, 2002).

C

Cheats ou ***Cheatcodes*** – Código que permite interferir nas características de um jogo, geralmente implementado pelos próprios autores. Com isso pode-se conseguir vidas ilimitadas, armas suplementares, etc. Exemplo: No jogo *Doom* são usados os *cheatcodes* IDKFQ, IDDAD, IDBEHOLD seguidos de RIVALS, IDCHOPPER, etc. Certos jogos são conhecidos por não permitirem o usuário chegar ate o fim sem a ajuda dos *cheatcodes*. (Le Jargon Français, 2002).

Crossover – Na área de Algoritmos Genéticos, dados dois elementos, onde cada um é descrito por um cordão de caracteres, ocorre *crossover* quando há o desmonte de uma parte do cordão do primeiro elemento, há o desmonte de uma parte do cordão do segundo elemento, e cada pedaço é remontado no cordão restante do outro elemento.

CRPG (Computer Role Playing Game) – Um *RPG* (*Role Playing Game*) é um jogo em que cada participante assume as características de um personagem (tal como um ogro ou um capitão de uma espaçonave futurísticas) que pode interagir com o mundo imaginário do jogo. Jogos mais populares incluem "*Dungeons and Dragons*" (Caverna do Dragão), *Battletech*, e Guerra nas Estrelas. Ambientes On-Line, conhecidos como *MUD* ou *MOO* incluem *software* para jogar e desenvolver *RPG*'s. Uma idéia diferente, mas relacionada, é o ambiente virtual em que cada participante define a sua própria aparência, ou avatar, e interage com outros dentro deste ambiente, usando o seu avatar. (WhatIs?com, 2002).

D

Datagram – Datagrama. Um datagrama, segundo o *Request for Comments* 1594, é uma "entidade, autocontida, independente de dados carregando informação suficiente para ser direcionada da máquina de origem para a máquina de destino, sem necessitar de dados sobre as comunicações anteriores entre as máquinas origem e destino, e sobre a rede de transporte".(WhatIs?com, 2002).

F

Framerate – Em animações, televisão e monitores de computadores, a taxa de amostragem (*framerate*) é o número de imagens que são projetadas na tela por segundo. *Framerates* são usadas para sincronizar áudio e imagens, seja em filme, televisão ou vídeo. Em imagens em movimento e televisão, as taxas de amostragem são padronizadas pela *Society of Motion Picture and Television Editors* (*SMPTE*). *Framerates* de 24, 25 e 30 telas por segundo são comuns. A *framerate* para animações em vídeo é de 24 por segundo, e para televisão são de 30 por segundo (Brasil).
Em fluxos de vídeo em computadores (*stream*), a *framerate* descreve taxas de reprodução para filmes tipo *AVI* e *QuickTime*. Esta taxa de reprodução para filmes *AVI* e *QuickTime* se relacionam diretamente com a continuidade desta reprodução. Quanto maior o número de imagens reproduzidas por segundo, mais

continuo o filme parece ao usuário. Baixos números de reprodução mostram imagens saltadas.Muitos fatores afetam o *framerate* conseguido no computador. Entre eles pode-se contar a placa de vídeo, uso de acelerador gráfico, uso de vários programas ao mesmo tempo, entre outros. (WhatIs?com, 2002)

Função sigmoidal – Dentro do tema de Inteligência Artificial, mais especificamente sobre Redes Neurais, uma função sigmoidal é uma função matemática de ativação dentro de uma rede neural. Ela é definida como uma função estritamente crescente, suave e com propriedades assintóticas, dois exemplos deste tipo de função são a função LOGÍSTICA e a função de TANGENTE HIPERBÓLICA (HAYKIN, 1998).

G

Geradores pseudo-randômicos de objetos virtuais – Dentro da área de Realidade Virtual feita em computadores pessoais, Geradores pseudo-randômicos de objetos virtuais são algoritmos de pré-processamento que utilizam números pseudo-randômicos para modificar parâmetros – espaciais, cores, texturas, etc. – de objetos virtuais. De uma forma mais grosseira, tais geradores servem para "criar" novos objetos virtuais.

Gráficos pseudotridimensionais (também chamados de isométricos) – ISOMÉTRICO – cujas dimensões são iguais.(HOUAISS, 2001).
Gráficos pseudotridimensionais são gráficos que dão a falsa impressão de tridimensionalidade devido a uma disposição linearmente dependente de seus eixos. Num gráfico pseudotridimensional percebe-se a formação de seus desenhos através de dois eixos x, y do plano da imagem, e mais um terceiro eixo z que simula a profundidade, e é produzido por uma combinação linear destes eixos x, y. Assim: $z = a.x + b.y$.

Grafting – Na área de Algoritmos Genéticos e Vida Artificial, dados dois elementos que têm descritos seus genótipos, ocorre *grafting* entre eles quando um nó de um dos pais se separa da cadeia de genótipo original e se reconecta com a cadeia do outro genótipo. No exemplo dado, o genótipo do pai é copiado, e uma de suas conexões é escolhida para apontar para um nó aleatório na mãe. Os nós do pai e da mãe, que ficarem desconectados, são eliminados. O filho terá seu genótipo resultante do novo conjunto de nós conectados.
Pai = ABCDE = ABC + DE, com C apontando para o gene G da mãe.
Mãe = FGHIJ.
Filho = ABC + GHIJ = ABCGHIJ. Os genes DE e F são eliminados. (SIMS, 1994).

H

Hardware – *Hardware* é o aspecto fisico de computadores, telecomunicações e outros dispositivos da tecnologia da informação. O termo serve para distinguir a "caixa" e os componentes e circuitos eletrônicos de um computador do programa que o usuário coloca nele para fazê-lo executar tarefas. O programa veio a se

chamar *software*. *Hardware* implica permanência e invariabilidade. *Software* – ou programação – pode ser facilmente variado. O usuário pode incluir um programa inteiro no *hardware* e com isso criar uma experiência inteiramente nova para o usuário. O usuário pode alterar as configurações modulares que vêm nos computadores adicionando novos adaptadores que extendem suas capacidades.
Tal como a palavra *software*, *hardware* é um termo coletivo. *Hardware* inclui não somente o console do computador, mas também os cabos, conectores, unidades de suprimento de energia, dispositivos periféricos tais como teclado, *mouse*, auto-falantes e impressoras. *Hardware* é às vezes usado como um termo coletivo descrevendo o aspecto físico de infra-estruturas de redes de telecomunicação e telefonia.

I

Imagem *Raster* – Imagens *raster* são imagens digitais criadas ou capturadas (por um *scanner* ou máquina fotográfica) como um conjunto de exemplares de um dado espaço. Um "*raster*" é uma grade com coordenadas x e y em um dado espaço (e a coordenada z para imagens tridimensionais).
Um arquivo de imagem *raster* identifica qual destas coordenadas devem ser iluminadas, em luz monocromática ou colorida. O arquivo *raster* é algumas vezes referido como um *bitmap* porque contém informações que é diretamente mapeada para a grade da tela.
Um arquivo *raster* é normalmente maior que um arquivo com uma imagem vetorial, e um arquivo *raster* é difícil de alterar sem perda de informação, apesar de já existirem *softwares* que podem converter um arquivo *raster* em arquivos vetoriais para refinamento e mudanças. Exemplos de arquivos de imagens *raster* são dos tipos *BMP*, *TIFF*, *GIF* e *JPEG*. (WhatIs?com, 2002).

Imagem vetorial – Imagens vetoriais são imagens digitais descritas através de formas geométricas que contêm parâmetros. Um vetor é uma representação de uma quantidade e uma direção ao mesmo tempo, no espaço bi ou tridimensional.
Em gráficos vetoriais, o arquivo que armazena os dados vetoriais criados contém parâmetros tais como a forma geométrica descrita e seus valores.
Por exemplo, em vez de conter um *bit* do arquivo para cada *bit* de linha desenhada, um gráfico vetorial descreve uma seqüência de pontos a ser conectada. O arquivo resultante é bem menor.
Um arquivo vetorial é também chamado de arquivo geométrico. Muitas imagens criadas com ferramentas tais como **Adobe Illustrator** e **CorelDraw** são da forma de arquivos de vetores de imagem.
Imagens para animação também são usualmente criadas como arquivos vetoriais. Por exemplo, produtos da **Shockwave Flash** permitem ao usuário criar animações bi e tridimensionais que são enviadas a um requisitante como um arquivo vetorial, e depois "rasterizadas" à medida que chegam ao computador cliente. (WhatIs?com, 2002).

Inteligência Artificial – IA é a simulação dos processos da inteligência humana pelas máquinas, especialmente sistemas de computadores. Estes processos

incluem aprendizado (aquisição de informação e regras para usar esta informação), racionalidade (usar regras para aproximar, alcançar ou definir conclusões), e auto-correção. Aplicações particulares de Inteligência Artificial incluem Sistemas Especialistas, Reconhecimento de Fala e Visão de Maquina. (WhatIs?com, 2002).

Interatividade – Em computação Interatividade é o dialogo que ocorre entre um ser humano (ou possivelmente outra criatura viva) e um programa de computador. (Programas que funcionam sem o envolvimento direto do usuário não são interativos, sendo chamados de programas *batch* ou de *background*). Jogos são usualmente considerados como grandes exemplos de interatividade. Também o são as aplicações de entrada de pedidos, mas em menor escala, pois oferecem poucas opções para a interação do usuário. Os modelos mais antigos de interação do usuário com computadores eram indiretos e consistiam em submeter comandos em cartões perfurados e deixando que o computador os lessem e executasse os comandos. Posteriormente os sistemas de computadores foram projetados de tal forma que até mesmo pessoas leigas pudessem acessá-los através de programas. As primeiras interfaces homem-computador permitiam entrada de comandos na forma de texto.

Interface – Uma interface pode ser:

1. Uma interface de usuário, consistindo de uma série de diais, botões, comandos de sistema operacional, formatos de demonstração na tela e outros dispositivos providos por um computador ou programa. Esta interface permite ao usuário comunicar-se com o computador / programa. Uma interface gráfica para usuário (*Graphical User Interface* (*GUI*)) provê para o seu usuário um modo mais ou menos orientado a imagens, para interagir com a tecnologia. Uma *GUI* é usualmente uma interface mais satisfatória ou amigável com o computador.
2. Uma interface de programação, consistindo de um conjunto de parâmetros, funções, opções e outros meios de expressar instruções de programação e dados.
3. O arranjo físico e lógico que suporta o acoplamento de qualquer dispositivo a um conector ou a outro dispositivo.

Interfacear significa comunicar-se com outra pessoa ou objeto. Com equipamento de *hardware*, interfacear significa fazer um conexão física apropriada de tal forma que duas partes deste equipamento podem se comunicar ou trabalhar juntas eficientemente. (WhatIs?com, 2002).

M

Mineração de Dados – É o processo de ordenação através de dados para identificar padrões e estabelecer relacionamentos. Parâmetros de mineração de dados incluem:

- Associação – procurar por padrões onde um evento é conectado a outro evento;
- Análise de seqüência ou caminho – procurar por padrões onde um evento conduz a um posterior evento;
- Classificação – procurar por novos padrões (talvez resultando em uma mudança no modo como os dados são organizados);
- *Clustering* – encontrar e documentar grupos de fatos que não eram previamente conhecidos;
- Previsão – Descobrir padrões de dados que podem conduzir a predições razoáveis sobre o futuro. (WhatIs?com, 2002).

Mipmapped Textures ou ***Mipmapping*** – Técnica de filtragem que seleciona uma textura a ser mostrada sobre uma primitiva, a partir de seu nível de detalhe. O programador do ambiente virtual pode escolher texturas mais detalhadas para serem mostradas quando o usuário estiver mais próximo da primitiva, e escolher texturas mais grosseiras para exibir quando o usuário estiver longe da primitiva. Desta forma, há um ganho na velocidade de *rendering* de imagens, mas se ocupa mais memória. (Advanced Graphics Programming Techniques Using OpenGL, 2000).

N

NURBS – Do inglês "*Non-Uniform Rational B-Spline*", uma superfície do tipo *NURBS* é uma superfície que é definida a partir de uma combinação de dois elementos matemáticos: Uma equação paramétrica de superfície e um conjunto de pontos no espaço, chamados pontos de controle. (NEIDER, 1997).

P

Pixel – O *pixel* (uma palavra originária da contração "*picture element*") é a unidade básica de uma cor programável em um display ou imagem do computador. É mais uma unidade lógica do que física.
O tamanho físico do *pixel* depende de qual a resolução da tela do monitor usado. Se a resolução estiver na máxima, o tamanho físico do *pixel* será igual ao tamanho do ponto (*dot pitch*) da tela. Mas se a resolução for menor do que a máxima, um *pixel* será maior do que o tamanho físico do ponto da tela (ou seja, um *pixel* usará mais que um ponto da tela).
A cor específica descrita por um *pixel* é uma mistura de três cores básicas (vermelho, verde e azul) dentro de um espectro chamado de *RGB*. Esta cor é

determinada através de três *bytes*, um *byte* para a cor vermelha básica, um *byte* para a cor verde, e um *byte* para a cor azul.
Um sistema de cores de 24 *bits* usa todos os três *bytes*, mas muitos outros sistemas (normalmente mais antigos) usam apenas um *byte*, permitindo uma gama de 256 tipos de cores somente.(WhatIs?com, 2002).

Pixelagem – Do inglês "*pixilation*".

1) É um termo às vezes usado para descrever o ato de transformar uma imagem impressa numa imagem digitalizada. Quando a imagem é capturada, ela é processada para um arquivo de formato gráfico vetorial ou *raster*.
2) Pixelagem também é a amostragem de uma imagem digitalizada onde os *pixels* individuais aparecem para o usuário. Isto pode acontecer quando uma imagem de baixa resolução é projetada numa tela muito grande, e cada *pixel* aparece visível separadamente.(WhatIs?com, 2002).

Plug-in – Aplicações *plug-in* são programas que podem ser facilmente instalados e usados como parte de *softwares* para navegação na *Internet*.
Inicialmente, o navegador da empresa *Netscape* permitia ao usuário baixar, instalar e definir programas que reproduziam sons ou animação de vídeo. Estes programas eram chamados aplicações de ajuda. Mas estes programas eram executados como aplicações separadas e exigiam uma segunda janela a ser aberta.
Uma aplicação *plug-in* atualmente é reconhecida automaticamente pelo navegador, e sua função é integrada dentro do arquivo *HTML* que esta sendo apresentado na tela.
Entre os *plug-ins* mais populares estão o Acrobat da empresa Adobe, que é um programa de apresentação e navegação de documentos, o RealNetworks, um reprodutor de vídeos tipo *streaming* (fluxo de dados), e o Shockwave, da empresa Macromedia, que é um reprodutor de sons e animação interativos.
Atualmente existem várias aplicações *plug-in*, que estarão disponíveis a qualquer momento na *Internet*, e o usuário requisita seu carregamento somente no momento em que precisar. (WhatIs?com, 2002).

Q

QUAD_STRIP – Na programação OpenGL para gráficos tridimensionais, uma *quad_strip* é uma seqüência unida de quadriláteros. Seus pontos são determinados em pares: os dois pontos iniciais são determinados, os dois pontos seguintes indicam os limites do quadrilátero formado, o terceiro par de pontos forma mais um quadrilátero, e assim por diante.(NEIDER, 1997).

R

Reconhecimento de Palavras Faladas – Reconhecimento de fala ou de voz é a habilidade de uma máquina ou programa para reconhecer e executar comandos de voz ou comandos ditados continuamente. Em geral, reconhecimento de fala

envolve a habilidade de comparar um padrão de voz a um vocabulário adquirido previamente. Usualmente, um vocabulário limitado acompanha o produto, e o usuário pode adicionar mais palavras posteriormente. *Softwares* mais sofisticados são capazes de aceitar fala natural.
Dragon Systems e IBM são duas empresas lideres na pesquisa e desenvolvimento de produtos que suportam reconhecimento de fala. (WhatIs?com, 2002).

Redes de Computadores (ou Redes de Trabalho) – Dentro dos conceitos de tecnologia da informação, uma rede de trabalho é uma serie de pontos interconectados por caminhos de comunicação. Redes podem ser interconectadas com outras redes e também podem conter sub-redes.
As topologias, ou configuração, mais comuns de redes incluem a rede tipo barramento, estrela e anel. Redes de trabalho podem ser caracterizadas em termos de distancia no espaço, como redes de área local (*LAN*) e redes de área metropolitana (*MAN*), e ainda redes de área gigante (*WAN*).
Uma dada rede de trabalho pode ser também caracterizada pelo tipo de tecnologia de transmissão de dados usada (por exemplo, *TCP/IP* ou *SNA*), pelo fato de suportar dados tipo voz e / ou dados binários.
Redes de trabalho podem ser classificadas pelo fato de serem de uso privado ou publico, ou pelos seus tipos de conexão (*dial-up* ou por interrupção) e pelos tipos de ligação física (fibra ótica, cabo coaxial, *UTP*).
Grandes redes de telefonia, e as redes que usam suas estruturas (tais como a *Internet*) compartilham e trocam estruturas e arranjos de tal forma que grandes redes de trabalho são criadas. (WhatIs?com, 2002).

Rendering – O verbo inglês "*to render*" se origina do verbo francês medieval *rendre* significando "devolver". Na tecnologia da computação gráfica, um *software* de computação gráfica pode ser usado para renderizar efeitos especiais tridimensionais, dados os parâmetros corretos. Um sistema de *display* de computador renderiza uma imagem que é recebida por este na forma de um *bitmap* ou imagem de fluxo (*streaming*).
O nome "*rendering*" é um termo às vezes usado para descrever um desenho, croqui, um plano ou outro trabalho artístico ou de engenharia.(WhatIs?com, 2002).

T

Tempo Real – É o tempo de resposta do computador que o usuário sente como suficientemente imediato, ou que permite à máquina manter algum processo externo (por exemplo, apresentar uma visualização do clima enquanto este está em constante movimento). O termo "tempo real" descreve um senso de tempo mais humano do que de máquina.(WhatIs?com, 2002).

Timer – Na linguagem C, um *timer* é um objeto que envia mensagens para a aplicação de *software* a intervalos regulares. Caso o projetista de *software* queira que uma tarefa de fundo seja executada com uma freqüência fixa no tempo, enquanto o usuário executa um outro tipo de tarefa, este tipo de objeto é usado

para indicar o momento em que esta tarefa de fundo deve se repetir.(WALNUM, 1994).

Thread – Em programação de computadores, uma *thread* é uma informação associada com o uso de um programa que pode gerenciar múltiplos usuários concorrentes. Do ponto de vista do programa, uma *thread* é uma informação necessária para servir um usuário em particular ou uma requisição de serviço particular. Se vários usuários estão usando o mesmo programa, ou ocorrerem requisições concorrentes ao mesmo programa, uma *thread* é criada e mantida para cada um deles. A *thread* permite ao programa saber qual usuário esta sendo servido, à medida em que este programa é acionado por diferentes usuários. (WhatIs?com, 2002).

Tracker – É um dispositivo de rastreamento de posição que monitora constantemente o movimento do corpo humano – cabeça, mãos ou olhos, e retorna estes valores para o computador hospedeiro, e cujas movimentações são interpretadas como mudanças no ambiente gerado no computador. (HEIM, 1998).

U

Union – Na linguagem C, uma *union* é uma localização de memória usada por muitos tipos de variáveis diferentes. Se por exemplo, uma *union* contiver um tipo de dado de ponto flutuante, que ocupa quatro *bytes*, e um tipo de dado inteiro, que ocupa dois *bytes*, estas variáveis estarão ocupando o mesmo endereço de memória, e o tamanho total da *union* será de quatro *bytes*.
Embora haja o compartilhamento de uma mesma posição de memória por duas ou mais variáveis, de formatos diferentes ou não, somente uma delas terá seu valor compreensível, dependendo do momento em que for usada. (SCHILDT, 1992).

V

Visão Biótica – BIÓTICO – adj. 1 – relativo a biota; 2 – relativo ou pertencente à vida ou aos seres vivos; 3 – criado, provocado ou induzido pela ação dos organismos vivos. (HOUAISS, 2001)
Visão biótica – Visão natural. No texto, se refere à visão que não se utiliza de recursos para simular a tridimensionalidade do mundo real.

Visão Estereoscópica – ESTEREOSCÓPICO – 1 – Processo fotográfico e de posterior projeção de imagens que dão à imagem plana a impressão de relevo, por ser o objeto fotografado ou filmado, simultaneamente, em duas perspectivas diferentes, utilizando-se de câmera com duas objetivas, uma a certa distância da outra. 2 – Visão tridimensional que se obtém de um objeto assim fotografado ou filmado, usando-se visores ou óculos que adaptam cada olho a uma das imagens. Do inglês "*stereoscopy*" (1860-1865). (HOUAISS, 2001)

Visão estereoscópica – No texto refere-se à visão que dá aos olhos a impressão de relevo, devido à projeção feita por dispositivos de emissão de imagem (*HMD*, *HUD*, Óculos 3D, Projetores) simultaneamente, em uma tela ou sobre os olhos do observador, de uma imagem vista de ângulos igualmente diferentes.

Visão em Primeira Pessoa – Na área de jogos de computadores, *videogame* ou *arcades*, a visão em primeira pessoa é aquela em que o usuário vê como se estivesse participando do jogo pessoalmente. Uma conseqüência deste tipo de visão é que o usuário não pode ver o que se localiza às suas costas. Sua vantagem é que o usuário vê o ambiente sintético da máquina tal como está acostumado a ver na realidade, o que torna o jogo do qual participa mais convincente.

Visão em Terceira Pessoa – Na área de jogos de computadores, *videogame* ou *arcades*, a visão em terceira pessoa é aquela em que o usuário controla um indivíduo personalizado por um avatar. Ao contrario da visão em primeira pessoa, o usuário pode ver o avatar completo e todo o ambiente sintético ao seu redor.

VRML – (*Virtual Reality Modeling Language*) é uma linguagem para descrever ambientes tridimensionais e as possíveis interações que podem acontecer com o usuário. Para ver um arquivo *VRML*, o usuário precisa de um navegador *VRML*, que pode ser um *plug-in* para um navegador de *Internet* já existente. (WhatIs?com, 2002).