Universidade do Minho
Escola de Engenharia

Leonel Domingues Deusdado

Ambientes Virtuais Povoados com Simulação Eficiente de Detecção de Colisões e Planeamento de Trajectos em Navegação Realmente 3D

Tese de Doutoramento
Área de Informática

Trabalho efectuado sob a orientação de
Professor Doutor António José Borba Ramires Fernandes
Professor Doutor Orlando Manuel de Oliveira Belo

Dezembro 2008

A o s m e u s p a i s
L á z a r o e C e l i n a

A ciência (Do latim *scientia* - conhecimento) é um sistema de saberes obtidos através de leis gerais, testado através de métodos científicos, dependentes da lógica. A liga de materiais de que é composta incorpora valores, conceitos e sentimentos tais como: o estudo da natureza, a descoberta da verdade, a racionalidade, a meditação, a organização, a observação, a experiência, o tempo, o espaço, o movimento, a causa, o número, a prova, o sonho e o amor, entre tantas outras. Mas a peça central desta cadeia são… **as pessoas**. E é para todas as pessoas que directa ou indirectamente contribuíram e contribuem para a ciência o meu primeiro agradecimento.

Em relação às investigações que deram origem a este trabalho científico, foi fundamental o apoio e contributo de diversas pessoas, que sem elas não teria sido possível, as quais passo a referir:

Ao professor António Ramires Fernandes, meu orientador, pelo tema de investigação, e pela disponibilidade e confiança que tem depositado em mim. Ao longo destes anos, foram muitas as conversas que mantivemos e não posso deixar de mostrar o apreço pela possibilidade de expôr e confrontar novas ideias. À pessoa que sempre esteve disposta a ajudar, e nos momentos certos soube efectuar uma análise crítica e mostrar que eu estava errado.

Ao professor Orlando Belo, meu co-orientador, pela sua disponibilidade e ajuda no debate de ideias em temas complementares, fornecendo uma outra perspectiva dos problemas e soluções.

À minha família, ao meu pai, à minha mãe, irmãos Sérgio e Paulo, "Tiaguito" e "Martita", que sempre me souberam compreender e me suportaram durante os tempos mais difíceis.

A todos os amigos e colegas que também ansiaram por este momento. Em especial à Nelma, ao "Chico" e "Lúi", que foram e são verdadeiros amigos.

A todos, os meus sinceros agradecimentos.

Agradeço também ao Instituto Politécnico de Bragança, particularmente à Escola Superior de Tecnologia e Gestão, onde me foram dadas as condições para desenvolver a minha actividade académica e científica, onde ensinar e investigar constituem um motivo de orgulho.

A tecnologia de produção de ambientes virtuais tem vindo cada vez mais a ser utilizada em projectos de animação, desenho e avaliação em diversas áreas. Mundos virtuais com níveis de detalhe consideráveis estão a começar a emergir em toda a parte, desde largas áreas das actuais cidades até ambientes virtuais interiores mais específicos e detalhados (edifícios habitacionais, estádios, estruturas industriais, reconstruções arqueológicas, etc). No entanto, melhorar a aparência visual destes edifícios virtuais já não é suficiente. Com o propósito de proporcionar novas condições de simulação a aplicações tais como o planeamento urbano, simulações comportamentais e de fluxo de pedestres, entretenimento, etc, é necessário o povoamento destes ambientes. Povoar estes mundos simulando a presença de vida, adiciona um toque extra à visualização e realismo, mas infelizmente traz também carga adicional ao sistema. Uma das lacunas da pesquisa nesta área é a representação eficiente de ambientes densamente povoados, com simulação de navegação autónoma realmente tridimensional das personagens, enquadradas em modelos ou cenários arbitrários. Diversas condições e áreas de actuação são necessárias quando pretendemos simular a presença humana (através de personagens sintéticas animadas) nestas circunstâncias, tais como a detecção de colisões, planeamento de trajectos, algoritmos comportamentais, *rendering* dinâmico da geometria, entre outros.

Nesta tese, é publicado um método transversal de modo a exibir e consolidar comportamentos autónomos de multidões virtuais em ambientes reais de animação. O sistema tem a capacidade de incluir um grande número de personagens lidando com mundos 3D arbitrariamente complexos, não requerendo qualquer conhecimento prévio da geometria, e proporcionando navegação em tempo real, autónoma, e tridimensional.

Inicialmente, é apresentado um método de detecção de colisões usando técnicas conservadoras, capaz de comportar milhares de avatares e lidar com cenas 3D de grandes dimensões e complexidade, não necessitando de qualquer informação ou conhecimento prévio do modelo. Este método demonstrou ser um mecanismo eficiente e escalável de detecção de colisões entre os agentes e o ambiente 3D. Recorrendo a um mapeamento e extracção de dados automático a partir do modelo inicial, fornece a detecção de colisões e a interacção entre os próprios agentes virtuais, e os agentes virtuais e o ambiente que os rodeia. Este método mostrou-se apropriado como base de implementação posterior de algoritmos de planeamento de percursos e outros algoritmos comportamentais, onde o avatar incorporará procedimentos de mais alto-nível.
Para projectos de desenho, simulação e testes de facilidades de navegação em locais públicos, é importante prever as principais rotas ou fluxos a serem usados. Uma segunda aproximação apresentada, consiste em decompor a cena 3D em partições multi-nível (para navegação em ambientes 3D, principalmente em interiores de edifícios) criando um sistema que possa usar este tipo de catalogação como informação relevante de modo a planear rotas de acordo com as deslocações em várias alturas.
A outro nível, o objectivo foi também testar a base de navegação criada, desenvolvendo mecanismos de implementação de novos e naturais comportamentos associados à navegação das personagens virtuais, lidando com várias variáveis de interacção, permitindo um comportamento mais realista e de reacção entre estes e o ambiente virtual. Em resumo, foram definidos sistemas de condições, regras e propriedades capazes de produzir comportamentos mais naturais e autónomos em personagens virtuais representativos da conduta humana.

Virtual environment technology has been increasingly used for animation projects, design and evaluation in several areas. Virtual worlds, with considerable levels of detail, are starting to emerge everywhere, from large areas of actual cities to detailed and complex virtual indoor environments (buildings, stadiums, industrials structures, archaeological reconstructions, etc). However, improving the visual appearance of these virtual buildings is not enough anymore. In order to provide applications with new simulation conditions such as urban planning, behaviour and flow of pedestrian's simulation, entertainment, etc, requires the populating of these virtual environments. Populating these worlds to simulate the presence of life, adds an extra touch to the visualization and credibility, but unfortunately it also brings an extra burden to the system. One of the issues of the research in this area is the representation of a densely crowded environment, simulating autonomous and real three-dimensional navigation to the virtual characters in arbitrary three-dimensional models or scenarios. Several steps are required when we need simulate the human presence (by synthetic animated characters) in these circumstances, such as collision detection, path planning/finding, behavioural algorithms, dynamic rendering of geometry, amongst others.

In this thesis, a transversal approach is presented to demonstrate and consolidate autonomous virtual crowd behaviours in realistic animation environments. The system is able to include a large number of characters dealing with arbitrarily complex 3D worlds, not requiring any prior knowledge of the geometry, and providing real-time navigation, autonomous, and really three-dimensional.

Initially, a method for efficient and scalable conservative collision detection is presented, that is able to deal with large and complex 3D scenes with thousands of avatars, not requiring any prior knowledge of model. This method demonstrated to be a fast, efficient and scalable collision detection process between virtual agents and the 3D environment. Using an automatic data extraction and mapping process from the initial graphical model, it provides collisions detection and interaction between virtual agents, as well as virtual agents and the environment that encircles them. This method proved to be appropriate as a basis for further implementation of path planning/finding algorithms and other behaviours algorithms.
For design projects, simulation and the study of crowd behaviour facilities in public places, it is however important to be able to predict heavily used routes or peak flows. The second approach presented, consists in decomposing the 3D scene in multi-level sub-divisions (for navigation in 3D environments such as indoor building) creating a system that can use this type of cataloguing as relevant information to planning and finding routes, according to the movements at the various levels of heights.
At another level, the goal was testing the base of navigation, developing mechanisms for new and natural behaviour implementations associated with virtual characters navigation, dealing with some interaction variables, representing a more realistic react/interact behaviour. In summary, autonomous conditions systems, rules and properties were defined, that are capable to produce behaviours representative of human condition.

A*	A Star
ACL	**A**gent **C**ommunication **L**anguage
ACS	**A**nt **C**olony **S**ystem
AFD	**A**utómatos **F**initos **D**eterministas
API	**A**pplication **P**rogramming **I**nterface
CAD	**C**omputer **A**ided **D**esign
CG	**C**omputação **G**ráfica
DevIL	**Dev**eloper's **I**mage **L**ibrary
FIPA	**F**oundation for **I**ntelligent **P**hysical **A**gents
FSM	**F**inite **S**tate **M**achine
GNU LGPL	**GNU L**esser **G**eneral **P**ublic **L**icense
GPU	**G**raphics **P**rocessing **U**nit
IA	**I**nteligência **A**rtificial
ISO	**I**nternational **S**tandards **O**rganization
JPEG	**J**oint **P**hotographic **E**xperts **G**roup
LOD	**L**evel **o**f **D**etail
MANG	**M**ulti-**A**gent **N**avigation **G**raph
MASSIVE	**M**ultiple **A**gent **S**imulation **S**ystem **i**n **V**irtual **E**nvironment
MIT	**M**assachusetts **I**nstitute of **T**echnology
OPENGL	**Open G**raphics **L**ibrary
PDI	**P**rocessamento **D**igital de **I**magens
RV	**R**ealidade **V**irtual
SAGE	**S**emi-**A**utomatic **G**round **E**nviroment
SMA	**S**istemas **M**ulti-**A**gentes
VUL	**V**irtual **U**rb**a**n **L**ife
VRML	**V**irtual **R**eality **M**odelating **L**anguage
WIMP	**W**indow-**I**con-**M**enu-**P**ointer
X3D	3D ISO Standard XML
2D	Duas Dimensões
2,5D	Duas Dimensões e meia
3D	Três Dimensões

“ A mente que se abre a novas ideias, jamais voltará ao seu tamanho original ”

Albert Einstein

CAPÍTULO 1

Introdução

O Homem desde sempre tentou inventar dispositivos que lhe permitissem reproduzir os mecanismos da natureza nas suas mais variadas e diversas formas. Para os organismos vivos, a situação não foi divergente. As primeiras tentativas que se realizaram para criar modelos e dispositivos artificiais capazes de exibir propriedades semelhantes às que encontramos nos seres vivos datam já do início da década de 50 do século passado, coincidindo com a própria origem dos computadores.

No entanto, a fundação do conceito de «vida artificial», emergindo de um interesse comum de várias áreas de investigação, foi proposto há apenas 21 anos pela mão de Christopher Langton que, em 1987, em Los Alamos, nos Estados Unidos da América, reuniu pela primeira vez no 1º Workshop em Vida Artificial, grande parte da comunidade científica interessada nesta temática. De acordo com a definição dada pelo fundador, a vida artificial trata de todos os assuntos relacionados com formas de vida fabricadas tecnologicamente pelo homem com inspiração clara nos sistemas naturais que nos rodeiam. Desde a síntese da imagem do passado e visionando o futuro, de acordo com um estudo actual, levado a cabo pela prestigiada empresa de consultadoria Gartner[1], os próximos dez a vinte anos vão ser marcados pela emergência de novas indústrias ligadas ao "fabrico" de sistemas de organismos sintéticos e digitais com capacidade adaptativa e vida autónoma, admitindo-se que, nos países desenvolvidos, já nos próximos 5 anos se passe a gastar mais tempo com processos de interacção realizados em mundos virtuais do que com outros no mundo real.

Presentemente, é normal associarmos a expressão mundo virtual à ideia de um ambiente tridimensional (3D) que tipicamente se assemelha ao mundo real, com

[1] http://www.gartner.com/

regras mundanas, tais como a gravidade, topografia, locomoção, acções em tempo real e comunicação. É idealizado como uma espécie de "mundo alternativo" que muitas das vezes, é concebido para dar alguma noção de entendimento sobre o mundo real. Uma área emergente com interesse crescente e directamente associada à criação de vida artificial nestes mundos virtuais é a computação gráfica 3D em tempo real, em parte devido à sua potencial aplicação ao desenvolvimento de jogos de computador, e mais recentemente aos chamados Massive Multiplayer Online Role-Playing Game (MMORPG´s), ou seja, aplicações que permitem a milhares de utilizadores criarem e identificarem-se com personagens num mundo virtual dinâmico, no qual interagem através destas em tempo real, com outros utilizadores usando a Internet. Este tipo de mundos virtuais, atraem cada vez mais subscritores e empresas que encontram neles novas oportunidades de investigação e negócio.

Os modelos tridimensionais virtualizando edifícios ou outro qualquer tipo de construção urbanística são também cada vez mais utilizados para simular aspectos ou conceitos de construção antes destes estarem totalmente edificados, de modo a poder planear por exemplo áreas e fluxos de circulação de pedestres, bem como visualizar padrões comportamentais associados a situações específicas (como por exemplo situações de grande densidade populacional ou situações de perigo e pânico). Assim, os arquitectos e engenheiros, poderão avaliar várias opções durante a etapa de desenho, assistidas por simulações ou animações computacionais. A área da simulação circunscrita à animação comportamental numa ambiência sintética, em tempo real ou não, abre a possibilidade para os investigadores perceberem e depreenderem computacionalmente comportamentos humanos mais ou menos complexos, quando inseridos em determinados ambientes, ou perante situações específicas.

Com o incremento e desenvolvimento das pesquisas na área de animação comportamental, também as áreas de desenvolvimento do entretenimento digital, cinema, produções multimédia, simulação, treino, ensino, visitas guiadas, etc, têm explorado recorrentemente a utilização de personagens virtuais. Quanto mais dotadas forem as “habilidades” destas entidades simuladoras, mais realistas e valiosos poderão ser os resultados das suas acções. Percebem-se hoje alguns aspectos cruciais associados à simulação da “verdade” da vida real, como por exemplo, quão mais perceptivas forem as entidades de uma simulação, mais inteligentes também o poderão ser, considerando para isso, o uso de técnicas de inteligência artificial (IA) associadas à utilização de uma série de algoritmos apropriados, preenchendo assim a

demanda actual e previsivelmente futura por alcançar capacidades de percepção virtuais aproximadas da realidade. São já comuns algumas personagens virtuais (actuando individualmente ou em grupo) possuindo certos tipo de inteligência, com o seu comportamento modelado e dedicado à apreciação de uma história (a história da existência dessa personagem), levando em linha de conta a sua missão, objectivos, estabelecimento de rotas de navegação, previsão de colisões e correcção de trajectos, adaptação ao ambiente, cooperação com outros elementos autónomos, etc…

No entanto, povoar ambientes virtuais com agentes inteligentes animados, dotados de comportamento autónomo e credível, capazes de interagir entre si e com os utilizadores do mundo real é um objectivo em direcção ao qual tem havido progressos notáveis, mas que depende ainda de muito trabalho de investigação e desenvolvimento.

1.1 | Identificação do Problema

Desde o famoso artigo de Reynolds em 1987 [1], houve um número impressionante de registos de pesquisa e investigação no uso de modelos virtuais caracterizando os seres vivos, habilitados com determinados comportamentos para a geração de animações e simulações por computador. A motivação por detrás de todo este trabalho, reside no facto já referenciado e subjacente à condição humana de tentar criar seres virtuais à sua imagem e à imagem (física e comportamental) daqueles que o rodeiam, e paralelamente, ao enorme avanço a nível do hardware gráfico e desenvolvimentos na programação específica associada, que têm permitido superações constantes neste campo de actuação.

Durante estes últimos anos a modelação virtual de edifícios, bairros e até cidades tornou-se assim comum. Mas a construção destes modelos (com a excepção dos jogos de computador) ainda é raramente pensada e orientada para a integração de personagens virtuais que os habitem. Existem portanto, imensos destes modelos tridimensionais que quando apresentados, nas suas mais variadas aplicações, carecem de "ilusão de vida" e consequentemente de realismo e até de noção de escala. Incluir simulação de personagens virtuais nestes modelos (arbitrários e complexos, estáticos ou dinâmicos), projecta um custo muito elevado, não só a nível de renderização, mas também ao nível da detecção de colisões e planeamento de trajectos. Neste campo, e após a análise de várias implementações presentes no estado da arte percebe-se a

necessidade da incorporação de novos métodos mais eficientes e melhor adaptados às condições de simulação específicas, no que diz respeito à incorporação e gestão da conjuntura ambiental necessária a uma correcta movimentação de um número elevado de personagens, tentando contrariar os elevados recursos consumidos, provavelmente desnecessários e não compensatórios em relação à qualidade gerada.

A simulação de multidões humanas envolve diferentes aspectos da computação gráfica, tais como por exemplo, a representação visual, técnicas de detecção de colisão, construção de algoritmos de orientação sobre o modelo a habitar, planeamento e estabelecimento de rotas de navegação, entre outros. Os requisitos actuais nesta área exigem que o mecanismo por detrás da implementação destas técnicas proporcione simulação em tempo real com capacidades de apresentação de um grande número de entidades em movimento. Mais ainda, exigem flexibilidade, eficiência e escalabilidade em simulações complexas e desempenhos não-lineares para uma vasta área povoada em ambientes virtuais. Assim, a simulação e *rendering* em tempo real de cenas muito populosas, com milhares de diferentes entidades, capazes de lidar com as "pretensas" colisões com o ambiente e as próprias personagens virtuais, continua a afirmar-se ainda hoje como um desafio.

Se a esta aspiração, adicionarmos ainda a capacidade destas personagens poderem navegar pelo modelo de uma forma livre, e verdadeiramente tridimensional[2], com movimentação a vários níveis de altura e em sobreposição, como por exemplo nos casos de navegação em pontes, na transição em altura entre os vários pisos em edifícios, estruturas industriais e arquitectónicas, ou túneis, mantendo um sistema automático, capaz de habilitar um qualquer modelo 3D a este tipo de navegação, garantindo ainda o aspecto realista desejado de comportamento e detecção de colisão, então, encontramos aqui uma área de investigação com capacidade de desenvolvimento de mais valias, recorrendo a métodos mais eficientes e mais abrangentes, onde somente é referida como possível trabalho futuro, em artigos de referência neste ramo de investigação (tais como: [2-6] entre outros). O termo tridimensional aparece aqui reforçado com a expressão "verdadeiramente", devido a ser muitas das vezes aplicado e confundido com situações 2,5D, onde unicamente se utilizam os vectores X,Y para a geração de uma base planar com a adição extra de um valor de altura por cada ponto do plano, o que, de modo nenhum permite uma correspondência aos mundos 3D reais, com possibilidade de incorporar várias alturas

[2] 3D Real – Incorporação de várias alturas independentes (plano Z) por cada localização do plano XY.

independentes para um mesmo ponto XY. Esta diferenciação é fundamental para se entender a abrangência e verdadeira contribuição científica desta tese.

Segundo Desney Tan et all [7], o abundante trabalho já realizado em navegação 3D, tem falhado em duas vertentes: a investigação para a criação de técnicas de navegação para tarefas e aplicações específicas, e a investigação de modo a poder perceber os princípios cognitivos subjacentes à própria simulação. Uma das funções específicas (circunscrita à primeira vertente referida por Desney Tan et all) muito utilizada em simulação, fazendo uso de personagens virtuais, é o planeamento de rotas de navegação (*path planning/finding*). Existem vários problemas em desenvolvimentos emergentes associados ao ramo da animação comportamental em computação gráfica e inteligência artificial, sendo, o planeamento e a procura de rotas de navegabilidade provavelmente os mais populares entre eles.

De modo a poderem ser identificados potenciais constrangimentos aquando da simulação, as cenas deverão ser produzidas contendo modelos virtuais acurados e representativos do enquadramento do ambiente a simular. Conhecer quais os princípios que regem a navegação no interior ou exterior de modelos em ambientes 3D, é falar de analisar e conseguir decompor de forma organizada o próprio ambiente virtual, e obter a partir daí, métodos capazes de simular navegação autónoma com determinados objectivos de localização espacial. O desenvolvimento neste campo aborda principalmente uma procura mais inteligente entre os vários pontos de navegação espacial, a sua usabilidade, e aplicação a qualquer ambiente no qual o computador tenha necessidade de movimentar objectos pelo modelo virtual. No entanto, os desenvolvimentos nesta área concentram-se sobretudo na utilização de modelos planares. Quando confrontados com ambientes 3D, os algoritmos actuais utilizam normalmente modelos conhecidos e preparados previamente. Se o ambiente for desconhecido à partida, então os algoritmos existentes não são aplicáveis de todo, ou o seu desempenho é deficiente ou não eficiente, já que foram projectados para trabalhar particularmente em duas dimensões.

Não menos importante também, é o cuidado a ter na representação e simulação realista dos humanos virtuais que irão povoar estes ambientes. Tendo presente, que a movimentação natural dos seres humanos é bastante complexa e "provavelmente" não determinista, com comportamentos de alto-nível, relevando algum tipo de inteligência superior, em consequência, é particularmente difícil também de emular utilizando personagens virtuais.

Idealmente, estes seres virtuais deveriam ser conscienciosos e imprevisíveis, reunindo condições para poder actuar de forma livre (também emocionalmente). Mas o quanto ainda estamos longe desta situação ideal? A sua "inteligência" é limitada aos resultados obtidos no desenvolvimento de novos métodos no campo da inteligência artificial. E infelizmente também, a grande maioria dos investigadores desta área, que desejam animar figuras sintéticas interagindo em mundos virtuais, por norma não são peritos em programação e animação por computador, sendo o contrário também verdade. No entanto é reconhecido por todos, as enormes potencialidades da representação sob a forma de personagens virtuais constituir um meio natural para poder avaliar visualmente estes progressos.

1.2 | Formulação do Problema

Uma vez identificados os principais constrangimentos, responsáveis pela ineficiência ou inexistência da resolução total do problema que se pretende estudar, o passo seguinte centra-se na sua formulação. A formulação do problema marca o início de uma série de acções, de índole prática, que deverão conduzir à sua resolução, na melhor das hipóteses, ou, tão simplesmente, à obtenção de um conhecimento mais aprofundado da realidade, que permita antever e atenuar alguns dos seus efeitos.

O intento de povoar e simular determinados comportamentos de um grande número de humanos virtuais autónomos num ambiente 3D inicialmente desconhecido, de uma forma inovadora, era o objectivo proposto à partida. Pretendeu-se evoluir este sistema de forma a permitir incorporar e visualizar o cenário traçado, sem perda de eficiência e realismo. O sistema deveria possibilitar a definição de características e regras de comportamento que permitam que as personagens virtuais adquiram "alguma" autonomia no mundo virtual, e que fossem detentoras de identidade singular, sem no entanto ignorar formas de comportamento colectivo e de interacção entre as individualidades ou grupos destas. Na vertente gráfica a preocupação incidiu sobre a detecção de colisões, quer entre cada agente e o ambiente, quer entre os próprios agentes. Neste âmbito o cerne do trabalho centrou-se na extracção de informação de forma completamente automática de modelos gráficos tridimensionais representativos do ambiente de simulação com complexidade arbitrária, capaz de servir de base a um mecanismo eficiente de detecção de colisões (estático para o

ambiente virtual, e dinâmico entre as personagens que o habitem), e planeamento de rotas de navegação.

O processo de preparação e construção deste sistema para a inclusão de personagens virtuais proporcionando vida aparente em modelos 3D arbitrários potencialmente habitáveis, seria tornado transparente, para que de forma clara e "mecânica", utilizadores normais, programadores ou investigadores nesta área de actuação, consigam aproveitar ou mesmo melhorar as potencialidades facultadas por este trabalho de investigação.

Sendo a navegação e orientação num mundo virtual uma das principais e primeiras "habilidades" a incorporar numa personagem autónoma, com o apoio do sistema anteriormente definido, foi desenvolvido um mecanismo que permite dividir e catalogar as várias localizações 3D aptas à locomoção de acordo com o seu posicionamento nos diversos níveis sobrepostos em altura aptos à navegação, bem como os espaços de acesso e transição entre estes, tentando promover assim um grau superior de organização, gestão e realismo na navegação a diferentes alturas, que tenha em linha de conta as potenciais vantagens e constrangimentos provocados por esta.

1.3 | Contributos Científicos

A evolução e propagação das tecnologias que lidam com informação codificada digitalmente, aplicadas ao âmbito da simulação de comportamentos humanos, percorrendo as áreas da computação gráfica e inteligência artificial, tem vindo a provocar profundas mudanças em diversas esferas das nossas vidas. Estas mudanças constituem-se como fascinantes desafios à comunidade académica em geral, e aos profissionais, cujas áreas de actuação estão relacionadas com a produção de trabalho científico em particular baseado nestas tecnologias, sobretudo, quando potenciam formas inovadoras de análise de processos, e de adopção e exploração desse conhecimento em contextos de interacção social.

A inovação é um dos processos mais críticos em qualquer investigação científica, e nesta não foi excepção. Esta, pode não se caracterizar directamente por um processo ligado à invenção, mas também, por mecanismos contínuos de reengenharia, com associação de ideias e conceitos já existentes a novos processos,

ou mesmo unicamente à redefinição de regras capazes de aumentar a eficiência e eficácia, obtendo vantagens competitivas. A mais valia desta inovação reside entre outras, no repensar dos problemas e avaliação dos resultados conseguidos, de modo a alcançar novas conquistas no âmbito do domínio científico respectivo. A inovação que se pretende, no âmbito deste trabalho de investigação, pode resumir-se em três vectores distintos de actuação, os dois primeiros na área da computação gráfica e o terceiro na área da inteligência artificial:

1 Partindo de uma base adaptada do conceito da discretização de um ambiente ou espaço 2D, evoluir e inovar para a preparação do mapeamento de um qualquer ambiente realmente tridimensional com alguma dimensão e complexidade, permitindo incluir e simular navegação a um número elevado de personagens virtuais, de uma forma completamente automática e transparente, com identificação eficientemente e em tempo real de zonas navegáveis neste tipo de cenas 3D, ou seja, aquelas que eventualmente produzam a possibilidade ou não de locomoção às personagens, derivando no final, num sistema de detecção de colisões eficaz e eficiente direccionado para este tipo de ambientes específicos.

2 Progredir desde a aquisição e representação espacial do ambiente representado pelo modelo anteriormente descrito, para a possibilidade de localização e catalogação de zonas espaciais específicas nestes mesmos ambientes, suportando o que por norma é conotado como planeamento de rotas de navegação em tempo real. Esta capacidade de planeamento ou enquadramento direccionado para posterior navegação e orientação num mundo 3D com as características antes descritas, permitirá alcançar desempenhos mais apropriados e realistas, e heurísticas mais eficientes para uma navegação orientada das personagens virtuais em simulação.

3 A um nível mais detalhado de caracterização do estado activo e reactivo de cada personagem em simulação, irão ser implementados modelos geradores de alguma autonomia às personagens envolvidas, resultando também comportamentos colaterais quando lidam com as várias variáveis de interacção destes com o ambiente ou outras personagens virtuais em actividades específicas. Este processo será capaz de gerar comportamentos mais realistas e adaptados a situações concretas, para que durante a simulação se possam

estimar procedimentos associados a uma conduta de actuação a nível individual ou em grupo.

Para além dos contributos científicos, a realização deste trabalho detêm outras motivações, que se podem traduzir em atributos genéricos, tais como: permitir facilmente reutilizar e incluir em modelos 3D simulação de seres vivos, suportando uma utilização e controlo acessível; gerar conhecimento automático dos percursos existentes entre cada ponto de navegação 3D; simular situações específicas, em locais específicos, utilizando um grande número e variedade de diferentes espécies e tamanhos de representações de personagens virtuais, etc.

1.4 | Modelo de Investigação

É desejável, que no contexto de um processo de investigação científica na área da computação gráfica e inteligência artificial, se fomente a integração da ciência com a tecnologia. Enquanto que a ciência tem por objectivo aumentar o nível de conhecimento sobre o meio, a tecnologia pretende proporcionar a criação de cada vez mais e melhores instrumentos que facilitem a interacção com esse mesmo ambiente [8].

O conhecimento científico resulta assim de um processo de pesquisa, sistemático e organizado. É clara a distinção entre o trabalho de investigação e o trabalho de recolha de informação: o primeiro é uma actividade heurística, de descoberta de informação, que envolve uma componente de análise e interpretação dos dados encontrados — trata-se de responder à questão "porquê?"; o segundo é um trabalho passivo que não envolve ainda a interpretação e o tratamento de dados — trata-se de responder à questão "o quê?". É inerente a esta coligação uma revisão constante da informação recolhida e das correlações que se estabelecem entre todos os dados. Sente-se aqui perfeitamente identificada, a famosa frase de Abraham Lincoln : "Se eu tivesse oito horas para derrubar uma árvore, passaria seis a afiar o meu machado".

Qualquer que seja o método utilizado, o modelo de investigação terá sempre presente, que uma correcta investigação obedece a princípios gerais comuns em todas as áreas do conhecimento: 1) a análise da informação faz-se em sistema aberto, nunca pretendendo ter encontrado a última verdade sobre um assunto; 2) o investigador não assume ter encontrado a resposta certa para um problema, mas

reconhece que descobriu um caminho para uma resposta certa e que esse caminho pode ser sondado por outros investigadores; 3) não se assume, por outro lado, que é impossível chegar a qualquer resultado correcto; 4) os dados são analisados com espírito crítico; 5) os problemas a investigar devem estar claramente enunciados; 6) as generalizações devem ser validadas por vários testes e por uma experimentação fundamentada e sistemática; 7) a resolução de problemas deve ser realizada com uma metodologia pré-definida.

Pegando neste último ponto, todos estes conceitos encontram-se organizados e projectados nesta tese seguindo o modelo *Action Research* de Stephen Kemmis [9], desdobrando-se e desenvolvendo-se a partir de uma série de iterações cíclicas ou interactivas fundamentais, resumidas a cinco etapas. São elas:

- O Plano – Nesta fase da investigação, o importante é identificar, formular e contextualizar o problema, para que se possam traçar os objectivos e as linhas de orientação para as acções a desenvolver. O plano identifica-se com os capítulos 1,2 e 3 desta tese.

- Acção – As acções são desenvolvidas ao nível dos capítulos 4, 5 e 6 e requerem a concepção de um referencial que deverá ter atenção aos contornos e variantes do problema, por forma a facilitar a realização de um protótipo adequado e integrado à realidade do estudo.

- Observação – Nesta fase de intervenção, o investigador centra-se sobretudo na monitorização do protótipo que deverá estar apto e sujeito a um processo de simulação. Por definição, as simulações e realização de estudos de caso devem desenvolver-se a partir de experiências conduzidas em ambientes especificamente criados para o efeito, e que permitam representar de forma fiel, as condições naturais em que os eventos irão decorrer.

- Reflexão – Uma vez implementada e testado o comportamento funcional do protótipo, surge a necessidade de proceder à sua avaliação. Este momento surge referenciado no final dos capítulos identificativos das acções desenvolvidas (capítulos 4, 5 e 6), de modo a enquadrar e perceber a avaliação dos resultados experimentais.

- Revisão – Esta identifica-se com o início de um novo ciclo. O saber adquirido no âmbito do estudo de *Action Research*, em que a acção gera conhecimento e o conhecimento origina novas interrogações, que por sua vez poderão induzir em novas pesquisas. A revisão e conclusões dos contributos científicos obtidos encontram-se situadas no capítulo 7, juntamente com uma reflexão, acerca de novas questões e exigências que possam servir de base a futuros trabalhos de investigação.

1.5 Estrutura da Tese

A estrutura da tese segue uma sequência lógica baseada na linha de acção do modelo de desenvolvimento *Action Research*, característica de resto, da generalidade dos trabalhos científicos. A tese inclui sete capítulos, sendo o primeiro introdutório, o segundo e terceiro servirão de base e explanarão alguns fundamentos subjacentes à problemática abordada, os seguintes dedicam-se aos desenvolvimentos efectuados em enquadramentos específicos na resolução do problema pretendido, seguindo as fases de acção, observação e reflexão, sendo que o último capítulo se dedica à fase de revisão com a apresentação de conclusões e possível trabalho futuro. Esta organização poderá ser observada através da Figura 1.

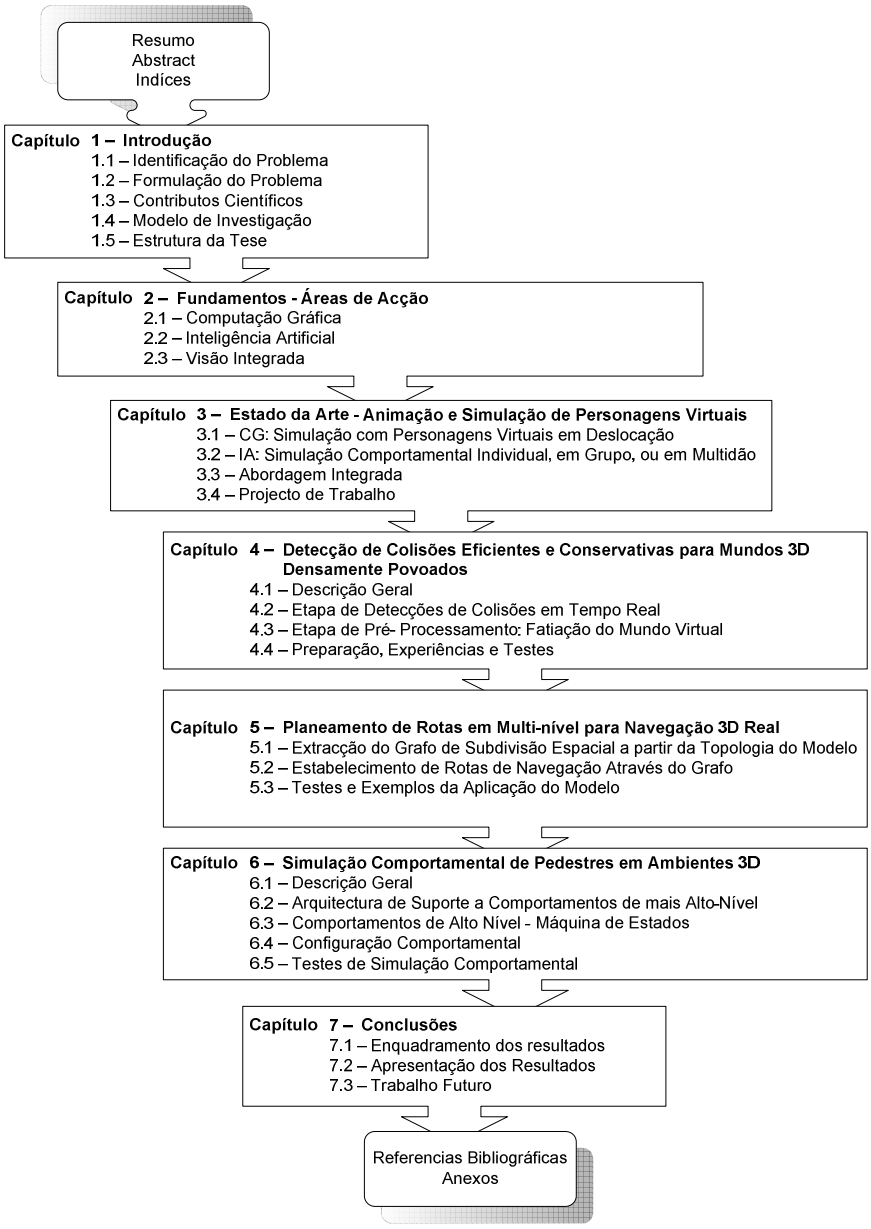


Figura 1 – Estrutura da tese

Em resumo, e aplicando aqui maior detalhe e estruturação, os capítulos encontram-se organizados da seguinte forma:

O primeiro capítulo destina-se à apresentação do "plano" de trabalhos, com realce para as questões da motivação, objectivos e interesses da pesquisa. Neste capítulo procede-se ainda, à caracterização científica do projecto.

No segundo capítulo são fornecidas as bases para a "acção". São apresentados os fundamentos estabelecidos nas áreas de computação gráfica e inteligência artificial para formalizar conceitos que suportem um entendimento de base para o trabalho

científico aqui apresentado. Estes conceitos envolvem noções de entendimento sobre ambientes virtuais, personagens virtuais, animação e simulação associadas à problemática em estudo.

O terceiro capítulo aborda o estado da arte genérico e específico nas respectivas áreas de afectação do trabalho científico, que servirá de suporte aos trabalhos desenvolvidos nos capítulos seguintes, geradores do principal contributo científico aportado por esta tese. É analisado em paralelo nesta fase, quais as lacunas científicas existentes nas áreas específicas de actuação, e quais delas este trabalho pretenderá ajudar a preencher.

O quarto capítulo expõe o desenvolvimento que permite detectar colisões em ambientes tridimensionais desconhecidos à partida para mundos densamente povoados. É apresentado o desenvolvimento do modelo que servirá de base aos objectivos almejados (acção), uma reflexão ao seu posicionamento e enquadramento face à formulação do problema (observação), e por fim testado, para se proceder à sua avaliação em modelos exemplificativos (reflexão).

No quinto capítulo, seguem-se as mesmas 3 etapas (acção, observação e reflexão) do modelo *Action Research* referido anteriormente, mas para a problemática do planeamento e identificação de rotas de navegação para o tipo de ambientes que pretendemos simular (ambientes desconhecidos e navegáveis verdadeiramente em três dimensões). Assim, é feita uma reflexão e discutido um plano de execução, que se traduz num modelo capaz de efectuar o mapeamento do ambiente, catalogando as diferentes bases de navegação a alturas distintas e os acessos respectivos, de modo a poder criar condições de orientação mais eficientes, e o estabelecimento de heurísticas apropriadas ao planeamento de rotas e direcções associadas às condições de navegabilidade em cenas 3D. O modelo encontrado é também testado de modo a proceder à sua validação e avaliação.

O tema de animação comportamental será desenvolvido no capítulo sexto. Aqui, e seguindo as iterações utilizadas nos dois capítulos anteriores, são apresentados modelos de comportamentos associados à caracterização da credibilidade e autonomia em simulações de populações de entidades virtuais caracterizando comportamentos humanos. São apresentados ainda, alguns exemplos de experiências submetidas ao protótipo funcional proveniente das investigações dos capítulos anteriores, conjugando agora comportamentos de baixo e alto-nível.

Por fim, são apresentados os resultados obtidos, enquadrados no contributo científico traçado e esperado inicialmente, são discutidas e comparadas as soluções alcançadas, traduzidas em conclusões e avaliadas segundo o contributo científico previsto. Após uma última revisão crítica do funcionamento integrado do sistema desenvolvido, é apresentado um plano de sugestões para um possível trabalho futuro.

CAPÍTULO 2

Fundamentos: Áreas de Acção

2.1 | Computação Gráfica

A computação gráfica é uma área que abrange um espectro largo de aplicações, desde os populares jogos electrónicos até ao projecto dos mais modernos equipamentos para viagens espaciais, passando também pelo cinema e publicidade, com a geração dos mais incríveis "efeitos especiais" e filmes de animação efectuados por computador, e também pela medicina, onde a criação e visualização de imagens de órgãos internos do corpo humano, e até fetos com poucos meses de vida, possibilita o diagnóstico electrónico de males que noutros tempos somente seria possível com intervenções cirúrgicas complicadas e comprometedoras. Segundo a ISO ("International Standards Organization") a computação gráfica pode ser definida como um conjunto de métodos e técnicas utilizados para converter dados num dispositivo gráfico, via computador.

Se tomarmos como base a definição da ISO, duas áreas têm uma estreita relação com a computação gráfica, são elas:

a) Processamento de Imagem: envolve técnicas de transformação de imagens. As transformações visam, em geral, melhorar características visuais da imagem como por exemplo aumentar o contraste, alterar a focagem ou ainda aumentar/reduzir cores, e reparar eventuais distorções.

b) Reconhecimento de Padrões: também conhecida como análise de imagens, busca isolar e identificar determinados componentes de uma imagem a partir da sua representação visual.

A Figura 2 ilustra o relacionamento entre a Computação Gráfica, o Processamento de Imagem, o Reconhecimento de Padrões e o Processamento de Dados convencional, segundo a visão da ISO.

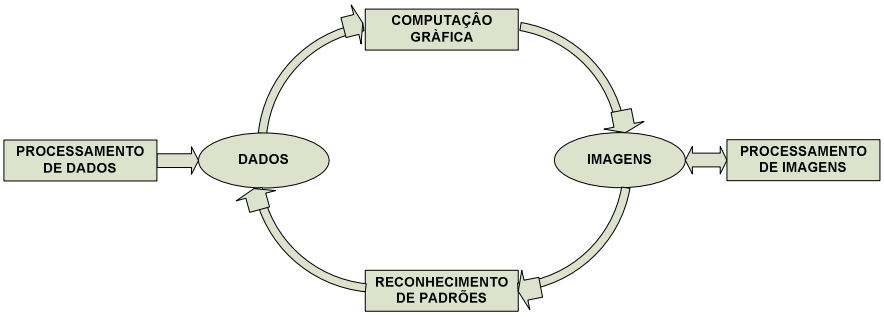


Figura 2 – Relacionamentos da computação gráfica segundo a visão da ISO

Parece existir consenso entre os pesquisadores da história da computação gráfica de que o primeiro computador a possuir recursos gráficos de visualização de dados numéricos foi o "Whirlwind I" (furacão), desenvolvido pelo MIT[3]. Este equipamento foi desenvolvido em 1950, com finalidades académicas e também possivelmente militares pois, logo de seguida, o comando de defesa aérea dos EUA desenvolveu um sistema de monitorização e controlo de voos (SAGE - Semi-Automatic Ground Enviroment) que convertia as informações capturadas pelo radar em imagens num tubo de raios catódicos (na época uma invenção recente).

Acontece que nesta época os computadores eram orientados para fazer cálculos pesados para físicos e projectistas de mísseis não sendo próprios para o desenvolvimento da computação gráfica.

Em 1963, surgiu uma das mais importantes publicações de computação gráfica de todos os tempos, a tese de doutoramento do Dr. Ivan Sutherland [10] ("Sketchpad - A Man-Machine Graphical Communication System"), propunha uma forma de interacção muito semelhante ao que hoje chamamos interfaces WIMP – Window-Icon-Menu-Pointer.

Esta publicação chamou a atenção das indústrias automobilísticas e aeroespaciais americanas. Os conceitos de estruturação de dados bem como o núcleo da noção de

[3] Massachusetts Institute of Technology

computação gráfica interactiva levaram a General Motors (GM) a desenvolver o percursor dos primeiros programas de CAD[4]. Logo de seguida outras grandes corporações americanas seguiram este exemplo, sendo que no final da década de 60 praticamente toda a indústria automobilística e aeroespacial já utilizava software CAD, para projectar e simular novos produtos.

Dois factores, entretanto, foram fundamentais para o desenvolvimento da computação gráfica tal como hoje a conhecemos:

a) O desenvolvimento da tecnologia de circuitos integrados durante a década de 70, que permitiu a redução de preços e a consequente popularização das máquinas.

b) O fim da ideia de que os fabricantes de computadores devem fornecer apenas a máquina e o sistema operativo e que os utilizadores devem escrever os seus próprios programas de aplicação. A popularização dos programas de aplicação prontos e integrados (editores de texto, editores gráficos, processadores de imagem, bases de dados, etc) permitiram a popularização da computação gráfica na medida em que possibilitaram que o utilizador comum sem conhecimento ou tempo para desenvolver aplicações gráficas pudessem utilizar as facilidades das mesmas.

Em 1983, o computador pessoal é eleito «homem do ano» pela revista Time. Um investigador, tido como louco, lança um novo conceito – a multimédia. O "louco" era Nicholas Negroponte, dez anos depois adorado como o herói digital. Paralelamente em 1987, Chris Langton promove amplamente a ideia de «vida artificial» e altera o paradigma da investigação em torno da inteligência artificial – ficava mais perto a possibilidade de sintetizar o humano.

Em 1990 Rogers [11] classifica a computação gráfica em passiva e interactiva. Refere que, por computação gráfica passiva entende-se o uso do computador para definir, armazenar, manipular e apresentar imagens gráficas. O computador prepara e apresenta dados armazenados sob a forma de figuras e o observador/utilizador não interfere nesse processo. Exemplos deste tipo de actividade podem ser a simples geração automática de um gráfico de barras a partir de uma tabela, bem como a

[4] Computer Aided Design

complexa simulação do movimento de um veículo espacial a partir dos dados recolhidos em campo. Computação gráfica interactiva também utiliza o computador para preparar e apresentar imagens. Neste caso no entanto, o observador/utilizador já poderá interagir (em tempo real ou não) com a imagem.

Na actualidade, a visualização de imagens sintéticas fotorrealísticas, "fisicamente correctas", juntamente com a interactividade e animação realista, têm sido metas importantes na área da computação gráfica. Porém, este objectivo atinge-se à custa de cálculos complexos que consomem grandes quantidades de tempo e recursos de cálculo. Daí que, durante algum tempo, se tenha considerado que a interactividade era um objectivo mutuamente exclusivo da produção de imagens sintéticas de qualidade elevada. Este conflito de objectivos traduz-se principalmente pelo tempo que é necessário despender na produção de cada imagem, quando se contemplam por exemplo os fenómenos da iluminação global fisicamente correcta. Desta forma, o enorme esforço computacional envolvido sempre foi considerado um elemento comprometedor da interactividade e restringiu esta à utilização de imagens de baixa qualidade.

A recente banalização dos recursos computacionais, em especial devido aos enormes avanços no hardware gráfico, em simultâneo com o desenvolvimento na generalidade de novas técnicas e algoritmos, permitiu a criação de algumas soluções mais eficientes do que a simples utilização massiva de hardware em estações gráficas especializadas. De acordo com Aylett et all [12, 13], diversos factores têm motivado esta expansão. Primeiro, o aumento do poder computacional tem permitido não apenas a exploração de um alto grau de realismo visual, mas a adição de uma camada de inteligência aos ambientes. Segundo, a disponibilidade de bibliotecas e padrões gráficos 3D, tais como VRML[5], OpenGL[6], Java3D[7], X3D[8], etc; tem promovido o desenvolvimento de ambientes 3D. Terceiro, as técnicas de IA (inteligência artificial), tais como as implementadas em agentes inteligentes, e o processamento de linguagem natural, têm amadurecido em paralelo, sendo utilizadas e exploradas nas interacções entre os utilizadores e o ambiente.

Apesar de actualmente a evolução das técnicas de software e hardware gráfico permitirem a visualização de ambientes e objectos com um nível de realismo visual

[5] Virtual Reality Modelating Language
[6] Open Graphics Library
[7] API desenvolvida em Java comportando programação para 3D
[8] XML ISO standard para representações em 3D

muito alto, a capacidade de produzir animações realistas de forma automática ainda está muito longe do desejável. Se por um lado é possível produzir imagens de humanos virtuais que podem facilmente ser confundidas com imagens de humanos reais, por outro, basta colocar um humano virtual na realização de tarefas com alguma autonomia e sua natureza virtual vai ser facilmente identificável.

Deste modo, actualmente a área específica da computação gráfica de Animação e Simulação Gerada por Computador, abrange diferentes grupos de pesquisa concentrados em diferentes aspectos gráficos dos ambientes e da inteligência embutida nestes. Os investigadores que trabalham nesta área simulam ambientes virtuais semelhantes ao mundo real, habitados por entidades autónomas "inteligentes" exibindo já uma grande variedade de comportamentos. Estas entidades podem ser objectos estáticos ou dinâmicos, mais simples ou mais complexos, personalizando representações virtuais de formas de vida (humanos ou animais), entre outros.

2.1.1 Humanos Virtuais – Agentes e Avatares

Muitas são as razões que conduzem ao desenvolvimento de modelos virtuais humanos especializados e muitas são as áreas de aplicação destes modelos. Entre elas:

- Engenharia: análise e simulação de protótipos virtuais;
- Conferência-virtual: teleconferências usando representações convincentes e virtuais dos participantes, visando por exemplo reduzir a taxa de transmissão de dados;
- Interacção: representações de humanos virtuais guiados por dispositivos de realidade virtual em mundos virtuais;
- Ambientes virtuais: vida e trabalho em ambientes virtuais dedicados à visualização, análise, treino ou apenas experiência;
- Jogos: personagens em tempo real com acções e personalidade dedicados ao entretenimento;

- Treino: desenvolvimento de habilidades, coordenação de equipas e tomada de decisões;

- Educação: ensino à distância, atendimento interactivo e instruções personalizadas;

- Militar: simulação de campos de batalhas com participantes individuais, treino de equipas em operações militares e de paz.

- Comunicação: aprendizagem de formas de comunicação alternativas, por exemplo a Língua Gestual (tema do trabalho apresentado pelo autor desta tese nas suas investigações de Mestrado [14]).

- Etc…

Na concepção de humanos virtuais, existem várias noções para o que chamamos de fidelidade virtual, dependendo da aplicação. Por exemplo, fidelidade quanto a altura, tamanho, capacidades comportamentais, número de personagens, limites de elasticidade das articulações, etc. Estas são essenciais para certos tipos de aplicação, enquanto que para outros, o mais importante é a fidelidade em termos de tempo, função, animação em tempo real, etc.

Existem portanto graduações de fidelidade que são inerentes aos modelos e às intenções. Alguns modelos são bastante avançados em determinados aspectos, mas ineficientes em outros. De uma forma bastante genérica, os modelos de representação de humanos virtuais podem ser caracterizados em pelo menos, cinco dimensões [15]:

- Aparência: Modelos fisiológicos
- Função: Limitações humanas
- Tempo: Produção em tempo real
- Autonomia: Inteligência
- Individualidade: Personalidades variadas

Os humanos virtuais como modelos computacionais avançados de representação de pessoas e comportamentos, podem ser usados para avaliação ergonómica e simulação

de eventos, incluindo assim a nossa própria representação física e comportamental em ambientes virtuais.

Uma classificação em relação à presença e características para actores sintéticos é proposta por Thalmann [16], considerando quatro tipos básicos de actores:

1) Participantes: o actor virtual tem uma aparência física natural e é animado de acordo com as características de um corpo real. Esta técnica pode ser chamada de método de rotóscopia em tempo-real [17] e consiste na gravação de dados de entrada gerados por um dispositivo de realidade virtual em tempo real;

2) Actores guiados: actores virtuais guiados são actores cujos movimentos são dirigidos pelo utilizador, ainda que estes não correspondam directamente aos movimentos do utilizador. São movimentos baseados na mesma técnica usada no controlo das marionetas;

3) Actores autónomos: são actores capazes de possuir um comportamento próprio. Por comportamento entende-se não apenas a reacção às acções do ambiente, mas também a maneira como a personagem codifica e usa esta informação. O actor pode perceber objectos ou outros actores no ambiente através de sensores;

4) Actores perceptivos interactivos: trata-se de um actor consciente não somente da existência de outros actores como também de pessoas reais. Este actor pode também ser considerado autónomo.

O que faz um humano virtual aproximar-se do humano não é principalmente a sua aparência exterior, mas os seus movimentos, reacções e tomadas de decisões naturais, apropriadas e sensíveis ao contexto. Estes humanos virtuais são vistos de formas diferenciadas como agentes ou avatares. O agente é gerido por programas computacionais, e o avatar é controlado pelo utilizador/programador [18]. O avatar corresponde assim, tão-somente a uma representação gráfica completamente controlada, enquanto que um agente possui já um certo grau de autonomia e "inteligência".

O termo avatar é oriundo do sânscrito (avatãr) que na teogonia indiana (hinduísmo) significa cada uma das encarnações de um deus, especialmente da divindade Vishnu. A utilização deste termo não é pura coincidência. Os avatares são no léxico computacional novas identidades ou personagens lúdicas que nos substituem nos espaços virtuais, comandados ou controlados pelos humanos de uma forma directa e interactiva.

Já o conceito de agentes é normalmente conotado também com o conceito de autonomia derivando muitas das vezes para o termo de agentes autónomos que é relativamente recente e surgiu dos estudos na área de inteligência artificial [19]. Existem diferentes e até conflituantes definições para o termo, conforme pode ser conferido em [20]. Segundo Maes [21], agentes são sistemas computacionais inseridos em ambientes complexos e/ou dinâmicos, que têm a capacidade de perceber e agir de modo autónomo, para atingir objectivos ou executar tarefas para os quais tenham sido modelados.

Um modelo baseado em agentes consiste na criação de uma população destas entidades, com capacidade de percepção e acção similar aos componentes simulados. Para que os agentes possam agir, como se fossem os próprios componentes de um sistema, devem ser dotados de comportamentos (regras que definem suas possíveis acções). Isto é conseguido através da modelação de comportamentos, que consiste na análise dos componentes de um sistema e extracção das suas principais características para que possam ser incorporadas nos respectivos agentes. Desta forma, a modelação de comportamentos é uma etapa fundamental na criação de um modelo baseado em agentes (com detalhe no capítulo 6).

Nos dias de hoje, cada vez são mais os casos implementados no cinema ou televisão, onde se podem ver personagens sintéticas bastante realistas em actuação, tais como alienígenas, guerreiros, brinquedos, dinossauros, etc. Aqui as personagens virtuais poderão actuar de uma forma diferente ao de um simples desenho animado. Personagens criadas tipicamente para as cenas de um filme poderão incorporar pretensões de reutilização, ou já algum tipo de autonomia. Por norma é utilizado software dedicado (detalhado mais adiante), ou construído especificamente para poder prover alguma forma de independência às personagens virtuais. Talvez o mais conceituado destes programas seja o software "Massive", foi um dos primeiros sistemas 3D de animação para a geração de efeitos visuais em filmes e televisão, reconstruindo grandes grupos de personagens virtuais com alguma autonomia e

reutilização, ficando famoso pela recreação virtual da trilogia do Senhor dos Anéis. As principais emoções simuladas e recreadas basearam-se em qualidades como o caminhar detectando colisões, bravura, fadiga, alegria ou tristeza. Este tipo de personagens já é transportado e reutilizado muitas das vezes para jogos de Pc´s ou na própria Internet, como é o caso também do Senhor dos Anéis numa reprodução virtual na rede mundial [22] que faz justiça ao universo criado pelo seu autor. Agentes e avatares convivem num mundo virtual, onde pessoas de todo o mundo em tempo real poderão interagir e viver as batalhas da terra média.

Estas personagens como elementos gráficos representativos de características físicas e dinâmicas associados aos humanos são decompostos e articulados à imagem das fisionomias reais. As animações de figuras articuladas tornaram-se populares devido ao antigo desejo de animar e simular seres humanos em todas as suas valências através de actores sintéticos em ambientes de animação por computador Numa animação por computador, uma figura articulada representando uma personagem virtual com capacidade de movimentação é frequentemente modelada por um conjunto de segmentos rígidos conectados por junções. Abstractamente, uma junção é um ponto de contracção na relação geométrica entre dois segmentos adjacentes, onde o movimento relativo de um ao outro está restringido de alguma forma. Este assunto está relacionado directamente com os tópicos de cinemática e cinemática inversa, frequentemente usados em testes de animação e simulação envolvendo quase todas as áreas da engenharia.

2.1.2 | Animação e Simulação

2.1.2.1 | Animação

Animar, significa literalmente, dar vida a algo, mover algo que não pode mover-se por si só. A animação adiciona aos gráficos a dimensão do tempo, o que significa um aumento considerável na quantidade de informação transmitida [23].

Os primeiros projectos em animação (dignos deste nome) tiveram início por volta do século XIX, com a descoberta da persistência da visão. Esta descoberta levou à criação de um instrumento chamado Zoetrope [23], que consistia num cilindro oco, com rotação no seu eixo de simetria, onde estavam pintadas à volta do seu interior

uma sequência de diversas imagens ligeiramente modificadas. Ao rodar o cilindro o espectador tinha a ilusão de que as imagens se moviam. Porém, a ideia de usar uma câmara para fazer com que objectos ou imagens sem vida projectassem a ilusão de se moverem surgiu apenas em 1890 [23].

Actualmente a animação é recorrentemente ligada à era digital, caracterizada pela arte de criar imagens em movimento utilizando mecanismos específicos, com base em computadores. A história da animação digital está directamente relacionada com a história da computação gráfica. Desde os primeiros dispositivos disponíveis foram percebidas as possibilidades do seu uso para a geração da ilusão de movimento. Subjacente à esta evolução é a utilização da técnica *stop-motion* [24], onde modelos articulados e modelos de cenários são utilizados em conjunto com uma câmara. Em cada quadro gravado os modelos são ligeiramente modificados criando ilusão de movimento e correspondente animação.

Na sua forma mais primitiva, a animação por computador significa usar um renderizador padrão para produzir e reproduzir quadros consecutivos [25], porém, o termo animação, num contexto alargado e também de computação gráfica e dos jogos por computador, refere-se a qualquer alteração numa cena em função do tempo. As características passíveis de modificação, neste caso, são por exemplo: posição, forma, cor, iluminação, material, transparência, entre outros. [26], [27].

De acordo com Parent, R. [23], a animação por computador classifica-se, basicamente, em dois tipos:

a) **Animação assistida por computador** (*Computer Aided Animation*), onde a máquina é responsável apenas por automatizar (ou assitir) algumas etapas envolvidas na produção da animação. Geralmente refere-se a animações 2D aproximando-se muito das animações tradicionais, desenhadas à mão, onde o único método algorítmico realizado por este tipo de animação é a modificação quadro a quadro assistida por computador, onde o objecto e os quadros que formarão a ilusão de movimento estão já definidos, geralmente por meio de uma estrutura de *sprites*[9].

[9] Sprites – Imagens ou animações 2D integradas numa série de cenas animadas

b) **Animação gerada por computador** (*Computer Generated Animation*), diz respeito a animações totalmente geradas por computador. Ainda de acordo com Parent [23], as técnicas para produção desse tipo de animação classificam-se como:
   - **Técnicas de baixo nível**: são, geralmente, algoritmos de interpolação de quadros chave (*in betweening*) ou de formas chave (*morphing*) e
   - **Técnicas de alto nível**: são técnicas obtidas através de algoritmos ou modelos usados para gerar movimento através de um conjunto de regras ou restrições. O movimento é gerado através da especificação das regras ou restrições no modelo ou algoritmo desejado e de um conjunto de parâmetros iniciais. O sistema é, então, executado e o movimento dos objectos é calculado através dessas regras e restrições. Um modelo que se encaixa perfeitamente neste parâmetro é por exemplo a animação baseada em conceitos obedecendo a leis da física.

Animações tridimensionais por computador possuem vantagens únicas não disponíveis nas animações tradicionais. Por exemplo, as animações podem ser produzidas directamente de modelos ou de um conjunto de equações especificando o comportamento dinâmico de estruturas ou máquinas no seu espaço cartesiano natural, e isso possui implicações maiores no campo da visualização científica. Esse tipo de animação é, em algumas situações, qualificado como simulação, e é por definição, capaz de produzir animações extremamente realistas, ou informação fidedigna de um qualquer comportamento em teste.

2.1.2.2 | Simulação

Simulação é a imitação de operações de processos reais ou sistemas sobre o tempo. Simular envolve a geração de uma história artificial do sistema e a partir da observação desse histórico é possível inferir características do sistema real em representação [28]. Os sistemas reais ou conceptuais podem ser modelados usando simulação.

Apud Law, Schmidt, Taylor e Kilton [29] definem um sistema como sendo uma colecção de entidades que agem e interagem em função de algum objectivo lógico. Já um modelo, é uma representação do sistema em estudo. Em geral, sistemas simples, são representados por modelos matemáticos. Esta abordagem é denominada de

solução analítica. No entanto, a maioria dos sistemas reais são bastante complexos para serem representados analiticamente. Por esta razão são utilizados computadores no processo de simulação para avaliar modelos de forma numérica e gerar dados para estimar valores reais das características que o sistema possui [29].

De acordo com Rickel et al. [30], Gratch et al. [31] e Anastassakis et al. [32], as aplicações potenciais destes ambientes são consideráveis, podendo ser empregues numa vasta variedade de áreas. Estas áreas estão principalmente relacionadas com a simulação na indústria, envolvendo a simulação de colisões de automóveis, a robótica, a dinâmica do movimento de pedestres e robôs, a física, as pesquisas espaciais, entretenimento e educação, etc [33, 34].

Em computação, simular consiste em empregar técnicas matemáticas via computador com o propósito de imitar um processo ou operação do mundo real. Desta forma, para ser realizada uma simulação, é necessário construir um modelo computacional que corresponda à situação real que se deseja simular. Do ponto de vista do software, um jogo de computador por exemplo, poderá ser bastante semelhante a um processo de simulação de sistemas em tempo real. Um processo de simulação deve acompanhar a geração e execução de eventos no tempo, testando condições segundo as regras do domínio em questão. Um sistema de tempo real tem como principal característica o facto de poder atender a determinadas restrições de temporização externa em qualquer instante.

A simulação virtual, como reprodução artificial de um fenómeno natural, facilita a exploração de hipóteses através de uma experimentação em condições que podem ser controladas, para isso terá de haver também um controlo do ambiente onde a simulação se desenrola, de modo a não dissociar o ambiente de simulação com os objectos nele simulados, constituindo um todo indissociável.

2.1.3 Ambientes Virtuais de Animação e Simulação

O ambiente virtual (2D ou 3D) é um cenário, estático ou dinâmico, armazenado no computador e exibido, em tempo real, através de técnicas de computação gráfica. O termo ambiente virtual refere-se a uma representação de um sistema, apto para observação, estudo e navegação, onde personagens virtuais (avatares ou agentes) poderão ser simuladas. O aumento da popularidade computacional nos últimos anos

evidenciou também o esforço para o desenvolvimento de ferramentas que permitam a criação de ambientes virtuais com maior grau de realismo e interacção.

Construir ou utilizar um bom ambiente virtual é bastante útil porque confere a um agente simulado um espaço artificial para testar o seu comportamento, podendo-se variar arbitrariamente os parâmetros ambientais. Por exemplo, a bem conhecida simulação de um evento discreto gera sequências de eventos, que são distribuídas no tempo pela mesma lei, como seriam no mundo real. Num ambiente virtual, isto poderá também acontecer, onde os sentidos de um agente são estimulados por sentidos virtuais e os estímulos produzidos por estes, devem produzir o mesmo comportamento e respostas por parte do agente, tal como se proviessem do mundo real.

Assim, o ambiente de navegação virtual, pode ser compreendido como uma representação digital realista da totalidade ou parte de uma qualquer superfície a povoar. Os cenários tridimensionais são, por excelência, os que mais desafios colocam aos sistemas de visualização (interactiva ou não). São muitas vezes, a base de trabalho para panóplias de aplicações, desde o planeamento urbanístico, sistemas de navegação automóvel, simuladores de catástrofes, de impacto ambiental, meteorologia, turismo, educação, entre tantas outras. Todavia, todas estas aplicações têm um factor em comum: a diversidade e a quantidade de dados geométricos e imaginológicos.

Uma forma tradicional de gerir a volumetria e posterior correspondência de movimentação numa cena tridimensional é imaginá-la como um sólido composto por elementos volumétricos elementares (voxels), os quais se erguem a partir de uma base horizontal. Existe uma razão forte para a não utilização explícita de voxels em modelos de ambientes de simulação: o espaço para armazenamento. Uma cena armazenada através de voxels pode ter tamanho excessivamente grande, mesmo em baixa resolução espacial, além de que fornece um grau de liberdade acima do necessário para a sua modelação.

Uma forma mais apropriada de armazenamento e gestão pode ser utilizada se atentarmos para o facto de que ambientes de navegação são estruturas tipicamente com características bidimensionais (com possibilidade de incorporar diferentes alturas aos vários pontos do plano, o que na literatura sobre modelação geométrica se designa de 2,5D) do que realmente tridimensionais. Mas abstractamente, se

assumirmos que cada ponto existente em superfícies num plano X,Y possuir um único valor em altura, são descartados por exemplo a presença de áreas de simulação em túneis, pontes, interiores de edifícios, e qualquer outra estrutura de maior complexidade que não satisfaça a restrição acima descrita. Ou seja, a palavra simulação ficaria confinada a modelos limitados na sua área de aplicação, suportando superfícies 2D ou 2,5D e não 3D como existem de facto na realidade. A construção destes modelos com características de algo mais que as simples duas dimensões (sem no entanto abranger realmente a terceira dimensão) constitui um caso específico de trabalho de computação gráfica, designado de objectos de terrenos (*Terrain Objects*) ou mais precisamente de superfícies de elevação através de campos de alturas (*Heightfields* ou *Heightmaps*).

Um campo de alturas é definido por um conjunto bidimensional de amostras de altura de uma superfície. Matematicamente, podemos representar um campo de alturas por uma função analítica de elevação em função de X,Y.

O tratamento de dados de um ambiente de simulação realmente 3D, modelado não usando simples valores de altura representando uma estrutura 2,5D, é actualmente requerido para simulações com pretensões mais ambiciosas no que diz respeito à capacidade de lidar com o ambiente virtual em toda a sua aptidão espacial representando de forma mais realista as várias possibilidades de deslocação presentes em condições reais. O planeamento e a especificação de estruturas de representação e formas de processamento de dados representando estas condições de desempenho é bastante complexo de gerir, e consistem num problema em aberto na literatura específica, principalmente ao nível da sua aplicação de modo mais eficaz e eficiente. É este processo que se pretende discutir e resolver de uma forma inovadora e mais produtiva no capítulo 4 desta tese.

Para esta especificação, vão ser usadas técnicas de planeamento de navegação e detecção colisões, a partir de vistas ou projecções sobre o ambiente de simulação 3D, recorrendo à utilização do buffer de profundidades, de modo a permitir obter um mapa discreto do mundo, e poder aplicar sobre este, as técnicas anteriormente referidas. Mecanismos de projecção recorrendo à utilização do buffer de profundidades são amplamente conhecidos da literatura nesta área de actuação, os quais são optimizados no *pipeline* gráfico mais comummente utilizado em computação gráfica.

2.1.3.1 Processos de Gestão e Visualização Gráfica

O *pipeline* gráfico compreende todo o processo desde a especificação de uma cena tridimensional até à sua visualização num ecrã bidimensional ou em qualquer outro periférico de saída. Devido à natureza essencialmente sequencial da geração de imagens tridimensionais num computador, todo este processo é dividido em várias fases, que no seu conjunto formam uma estrutura de sequenciação gráfica (*pipeline* gráfico). A ordem segundo a qual estas etapas são executadas é habitualmente a que se apresenta em [33, 35, 36].

Estruturalmente, o *pipeline* gráfico é composto por 3 fases conceptuais: Aplicação, Geometria e Rasterização. Algumas destas fases são ainda constituídas por várias sub-fases que formam, também elas, um novo *pipeline*. Num pipeline gráfico, algumas fases são executadas exclusivamente em software, outras em hardware, e outras ainda dependem da arquitectura em que se está a trabalhar, sendo que, há actualmente uma tendência em deslocar a maior quantidade possível destas fases para o suporte via hardware dedicado, por forma a poder acelerar essas mesmas etapas. As técnicas e fases utilizadas que fazem uso da optimização do hardware gráfico para a recolha de dados sobre os objectos envolvidos na cena, e consequentemente poder estabelecer técnicas para detecção de colisões, e planeamento de rotas de navegação, enquadram-se nas sub-fases da Geometria e Rasterização, ambas presentes no *pipeline* gráfico standard da geração de gráficos por computador, as quais se sintetizam de seguida.

Fase da Geometria

Projecção

A sub-fase da Projecção é uma fase intermédia da fase da Geometria, onde se processa principalmente a transformação num volume de visualização através de um cubo unitário chamado de volume de visualização canónico. Os dois tipos de projecção mais comuns são a projecção ortogonal ou ortográfica e a projecção em perspectiva.

A projecção perspectiva identifica-se pelo facto de os objectos parecerem diminuir de tamanho à medida que se afastam. Isto acontece porque o volume de visão é uma parte de uma pirâmide invertida, onde os objectos mais próximos parecem maiores porque ocupam uma maior porção do volume de visão do que aqueles que estão mais afastados.

Numa projecção ortográfica é estabelecido um volume de visualização que corresponde à representação de um paralelepípedo rectângulo, Figura 3. O objecto ou objectos, só serão visualizados se estiverem contidos neste volume, limitados por um valor máximo ou mais distante do observador em Z (Z-Far), e um valor mínimo ou mais próximo do observador (Z-Near).

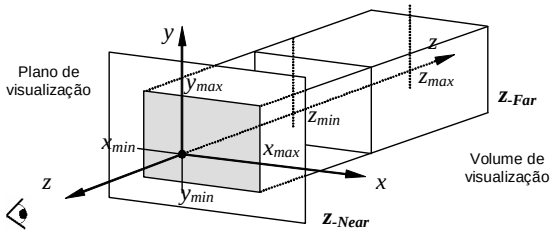


Figura 3 – Volume de visualização na projecção paralela ortográfica

A projecção ortográfica caracteriza-se por preservar o paralelismo entre rectas paralelas, porém não fornece a sensação de profundidade ao contrário do que acontece na vista ou projecção em perspectiva. Ao volume de visualização, antes da projecção, (área de base rectangular) dá-se o nome de *frustum.*

A informação relativa à profundidade (coordenada Z) passa a ser guardada num buffer especial chamado z-Buffer ou buffer de profundidade, que é utilizado posteriormente (fase de Rasterização) para, entre outras coisas, solucionar problemas de visibilidade.

Fase de Rasterização

No fim do *pipeline* gráfico, é realizada a fase de Rasterização que é quase sempre implementada em hardware, ainda que existam algumas implementações em software [35, 36]. Nesta fase todas as primitivas são desenhadas, isto é, os vértices, representados em coordenadas no ecrã, e toda a informação a eles associada (como o valor de profundidade, a cor, e a coordenada da textura, quando existe) são convertidos para pixeis do ecrã.

Esta fase é também responsável pela resolução das questões de visibilidade. Com recurso ao z-Buffer, onde (como vimos) é armazenada toda a informação de profundidade, é determinado se um pixel de uma dada primitiva deve ou não

aparecer na imagem final. Durante o desenho de uma primitiva o valor de profundidade de cada pixel que a compõe é comparado com o valor de profundidade existente no z-Buffer (e que corresponde a um pixel de uma primitiva previamente desenhada na mesma posição). Se o valor for menor ao existente no z-Buffer, então o pixel da nova primitiva está mais perto da câmara e o pixel resultante deve ser actualizado.

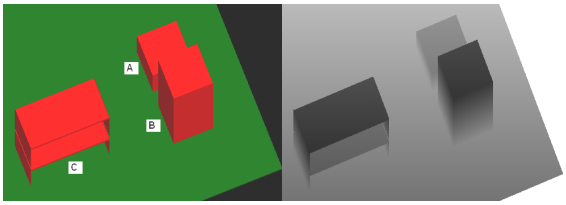
Exemplo de teste na Utilização do Z-Buffer

Para que possamos realizar uma representação acurada e realista de um modelo tridimensional complexo, numa cena contendo vários objectos diferentes, é necessário que superfícies normalmente invisíveis de um determinado ângulo ou ponto no mundo sejam também renderizadas invisíveis no computador. Este foi durante muito tempo, um problema fundamental abordado pela pesquisa em computação gráfica. De todos os algoritmos para determinação de superfícies visíveis (*visible-surface*) ou determinação de superfícies ocultas (*hidden-surface*), o buffer de profundidade ou z-Buffer é talvez o método mais simples e com certeza o mais amplamente utilizado.

O princípio de funcionamento do algoritmo é muito simples: Para todo pixel na *viewport*, além de um registro da intensidade, cor, transparência, brilho…etc, que deverá ser utilizado ao se apresentar este ponto em particular no ecrã do computador, é mantido também um registro da profundidade (em termos de coordenada Z) do objecto na cena que estiver mais próximo. Cada vez que um novo polígono é processado, um valor de Z e de intensidade são calculados para cada pixel que estiver dentro dos limites do polígono. Se o valor de coordenada Z obtido para aquele polígono for inferior ao valor de Z armazenado para aquele pixel no buffer, então este objecto está mais próximo do que algum objecto anteriormente renderizado naquela posição, logo, oculta-se o objecto anteriormente renderizado, e é substituído o valor armazenado naquela posição do buffer pelo novo valor.

Num primeiro exemplo de teste de uso e aproveitamento do z-Buffer, recorreu-se ao modelo simplista 3D da Figura 4-I) construído com cubos sólidos em OpenGL[10] simulando dois prédios de alturas diferentes (A e B), e outro com um nível intermédio de altura, navegável no interior (C).

[10] OpenGL (Open Graphics Library) é uma especificação de definição de uma API multiplataforma e multi-linguagem para a escrita de aplicações capazes de produzir gráficos computacionais 2D e 3D.



I II

Figura 4 – Modelo 3D com o respectivo mapa de alturas representado numa escala de cinzentos

A imagem resultante, extraída do z-Buffer é apresentada na Figura 4-II), é uma imagem em tons de cinza, onde cada um destes tons, representa a profundidade (em valor decimal entre 0 e 1) guardada no mapa de profundidades. Tons mais escuros representam valores mais próximos do Z-Near (valor com menos profundidade em relação à câmara), consequentemente, valores de cinza mais claros representam maior profundidade (valores mais próximos do Z-Far). Ao inverso, no ambiente 3D, e em relação à base de simulação, locais com os valores mais escuros são estruturas com altura superior, tal como se pode verificar na Figura 4-II).

2.2 Inteligência Artificial

O ser humano procurou sempre compensar suas fragilidades físicas com a construção de artefactos. Com o passar do tempo estes dispositivos foram evoluindo e o homem foi criando máquinas cada vez mais revolucionárias. A grande utopia foi sempre dotar estes mecanismos com a capacidade cognitiva dos seres humanos.

> «*Encontramo-nos no limiar de um esquema inteiramente novo. Até agora fomos moldados pela mão invisível da evolução darwiniana, um processo poderoso que aprende com o passado, mas é cego quanto ao futuro. Provavelmente por acidente, colocou-nos numa posição a partir da qual podemos fornecer-lhe alguma da visão que lhe falta. Podemos estabelecer objectivos para nós próprios e segui-los resolutamente, aceitando as perdas a curto prazo em favor de maiores benefícios mais à frente*».

Este extracto retirado do Livro "Homens e Robots" [37], de Hans Moravec, dá-nos o mote para poder perceber melhor o contexto onde se situa actualmente a inteligência artificial (IA).

A inteligência artificial é por um lado uma ciência, que procura estudar e compreender o fenómeno da inteligência, e por outro, um ramo da engenharia, na medida em que procura construir instrumentos para apoiar a inteligência humana. A IA é inteligência como computação, tentando simular o pensamento dos humanos e os fenómenos cognitivos associados. No entanto, a IA continua a ser a procura do modo como os seres humanos pensam, com o objectivo de modelar esse pensamento em processos computacionais, tentando construir um corpo de explicações algorítmicas dos processos mentais humanos. É isto o que distingue a IA dos outros campos de saber, colocando o ênfase na elaboração de teorias e modelos de inteligência com base em programas de computador.

Os estudos em IA dividem-se em quatro ramos fundamentais. Distinga-se pois, uma área ligada ao estudo das redes neuronais e ao conexionismo que se relaciona também com a capacidade dos computadores aprenderem e reconhecerem padrões. Um outro ramo ligado à biologia molecular na tentativa de construir vida artificial. Um terceiro relacionado com a robótica, ligada à biologia e procurando construir máquinas que alojem vida artificial. E finalmente o ramo clássico da IA com fortes

ligações desde o início à psicologia, desde os anos '70 à epistemologia e desde os anos '80 à sociologia, e que tenta representar na máquina mecanismos de raciocínio.

Mais recentemente surgiu um novo paradigma conhecido como "paradigma de agentes", o qual conquista nos dias de hoje um elevado nível de interesse entre os investigadores. Este novo paradigma incorporando o ramo clássico de IA, aborda o desenvolvimento de entidades que podem actuar de forma autónoma e racional. Se retomarmos a definição anterior onde se considera a IA como um meio para o desenvolvimento de sistemas que pensem e actuem racionalmente, podemos pensar que a IA, no seu conjunto, trata realmente de construir precisamente ditas entidades autónomas e inteligentes.

Resultado deste interesse, surge em 1996 a *Foundation for Intelligent Physical Agents* (FIPA) de modo a implementar especificações standards de software para sistemas baseados em agentes interactivos e heterogéneos. Desde a sua fundação até aos dias de hoje, esta instituição tem jogado um papel crucial na estandardização do desenvolvimento de agentes, promovendo um grande número de iniciativas e eventos que muito têm contribuído para o desenvolvimento e massificação da tecnologia de agentes.

Mas onde está a IA? Certamente "dentro dos agentes que são capazes de representar as situações que enfrentam e de realizar acções possuindo processos para manipular essas representações". Mas estará ela no algoritmo, ou pelo contrário na arquitectura de estados mentais?

De acordo com o apresentado anteriormente, a IA pode ser vista desde a perspectiva do desenvolvimento de agentes inteligentes. Esta ideia, considerada como um novo desafio a curto prazo, está a ser defendida e investigada por numerosos cientistas nesta área, sirva a modo de exemplo a seguinte frase: «Os agentes constituem o próximo avanço mais significativo no desenvolvimento de sistemas e podem ser considerados como a nova revolução ao nível do software». Esta frase foi já pronunciada pelo Dr. Nicholas Jennings no seu discurso ao receber o prémio ao melhor investigador novel do congresso internacional de inteligência artificial celebrado em Estocolmo (IJCAI'99) em 1999. Resulta ainda mais impactante quando tal afirmação se vê confirmada por numerosos indicadores actuais, como por exemplo o grande interesse despertado tanto a nível académico como industrial. Frente a uma afirmação deste tipo, surgem obrigatoriamente algumas perguntas de

carácter geral: em que consiste este novo paradigma? O que é um agente? Como se caracteriza um agente? O que estes nos oferecem de novo?

Da mesma forma que acontece com a própria definição de IA, podem encontrar-se propostas na literatura com um grande número de definições para o conceito de agente, sem que nenhuma delas tenha sido plenamente aceite pela comunidade científica, sendo talvez a mais simples [38], que considera um agente como uma entidade que percebe e actua sobre um entorno ou ambiente, aquela que melhor define este conceito.

Uma explicação para esta divergência de opiniões surge do facto de existirem vários tipos de agentes, usados em diferentes domínios de aplicações. Os agentes são criados para atenderem propósitos específicos, ou seja, são entidades que encapsulam conhecimentos sobre algum domínio [39]. Como observado em [40], não há definição universalmente aceita do termo agente, mas há um consenso geral de que a autonomia é a ideia central. Em termos de software, a autonomia está relacionada com processos que operam sem a intervenção directa do homem, e que podem atingir seus próprios objectivos.

A construção de agentes inteligentes pressupõe a existência de estruturas simbólicas (representação), a capacidade de elas poderem raciocinar (procura) e a existência de conhecimentos (matéria prima). Assim o campo mais popular da IA é sem dúvida o da engenharia do conhecimento, pois é aí que se concebem os sistemas periciais capazes de representar conhecimentos e raciocínios. A resposta à pergunta anterior (onde se localiza a I.A? no algoritmo, ou na arquitectura de estados mentais?), é: naturalmente em ambos, porque para a IA, representa os dois lados da mesma moeda. Pesquisas sobre realidade virtual produziram recentemente novos meios de simulação, proporcionando ambientes mais flexíveis para a experimentação. Neste contexto, compreender e perceber a autonomia artificial torna-se uma questão central. As ciências cognitivas tradicionalmente estudam a natureza do conhecimento e as suas funções. Actualmente, a comunidade científica que trabalha em inteligência artificial (IA) colabora activamente quer estimulando os sistemas biológicos quer modelando e construindo sistemas artificiais (*animats*), tendo sempre presente o conceito de autonomia.

2.2.1 | O Conceito de Autonomia

Hoje em dia quem é que não compreende o termo autonomia? Mesmo assim, possui vários significados de acordo com o contexto: autonomia de um carro (alcance da mobilidade), região autónoma (independência política), autonomia biológica (vida), etc. Em geral, um sistema (indivíduo social, organismo biológico, artefacto, mercado comum,...) é autónomo apenas se for capaz de proporcionar a si próprio as suas leis ou conduta, opondo-se aos sistemas heterónomos que são movidos pelo exterior. Este ponto de vista cresceu tornando-se num desafio ambicioso e extremamente interessante para as diversas disciplinas das ciências cognitivas.

De entre os trabalhos extraordinários sobre este aspecto das ciências cognitivas, sobressai o conceito de autopoiese, aplicado por Maturana e Varela em 1980 [41], identificado como a tentativa de extrair as características fundamentais da vida. Este, abrange certamente o conceito de autonomia. Apesar da riqueza deste modelo, é de difícil interpretação e uso nas ciências da computação, porque desenvolve noções como identidade ou emergência que são ainda demasiadamente abstractas para as aplicações computacionais.

2.2.1.1 | Agentes Autónomos

Actualmente existem inúmeras interpretações sobre agentes autónomos e muitas discordâncias entre estas. Muitas vezes um agente autónomo é confundido com uma simples aplicação que recebe dados, executa um processo e retorna algum resultado.

Por estes motivos Franklin e Graesser [42] procuraram elaborar um conceito mais abrangente e detectar aspectos que diferenciam uma aplicação computacional qualquer de um agente. Baseado nestes autores, é possível afirmar que todo agente é um programa, mas nem todos os programas são agentes. Um agente autónomo está situado num ambiente, sente e age sob este durante algum tempo, sendo que as suas acções são determinadas por um controlo próprio, onde o agente sentirá os efeitos das suas acções presentes, no futuro. Na Figura 5 é possível observar uma visão abstracta da relação de um agente com o ambiente [43].

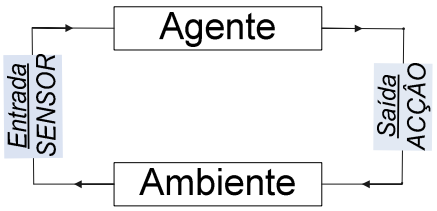


Figura 5 – Representação abstracta de um agente

Para Franklin e Graesser [42], sempre que nos referimos a agentes autónomos estamos a considerar questões de raciocínio orientadas para algum domínio, como por exemplo: persistência, autonomia, orientação, noções de ambiente, percepção, acção, etc.

2.2.1.2 Autonomia Artificial

De que propriedades necessita um agente para ser autónomo? Responder a esta pergunta através de dois pontos de vista diferentes (do observador e do programador) revelar-nos-á uma nova dimensão da questão.

O Ponto de Vista Exterior (Observador)

A apreciação da autonomia de um agente vista sobre a perspectiva de um observador particular: depende do seu passado, do seu conhecimento abstracto e intenções, entre outras, sendo o ponto de vista neste caso, externo ao agente.

Como observadores, nós também tiramos partido deste ponto de vista externo quando tentamos analisar as capacidades de um agente.

De modo a facilitar a avaliação, são introduzidos dois critérios em apreciação:

a. Preservar a sua integridade física e energética (sobrevivência).
b. Satisfazer as tarefas atribuídas pela sociedade (papel social).

O primeiro critério é substancial para o agente. O segundo é um critério de utilidade para o programador. Estes critérios não pretendem caracterizar exaustivamente a noção de autonomia, apenas se limitam a ser ferramentas práticas para avaliar o grau de autonomia de um agente. Do ponto de vista exterior, considera-se que um agente

capaz de exibir comportamentos que satisfaçam os critérios desta avaliação, é um agente autónomo.

O Ponto de Vista Interior (Programador)

Por outro lado, o fenómeno da autonomia baseia-se na estrutura interna e organização do agente. A arquitectura e a dinâmica interna determinam o agente, independentemente do observador.

A IA considerou sucessivamente a autonomia como uma capacidade de racionalização (no paradigma cognitivo) depois como capacidade de acção (no paradigma comportamental). Actualmente, uma nova linha de pensamento emerge destes dois paradigmas.

Racionalização:

Raciocínio (e representação) é um tópico central no paradigma cognitivo. A resolução de problemas e o planeamento em particular, podem ser apresentados superficialmente como um processo de procura num espaço de soluções. Invariavelmente, o tamanho do espaço de procura cresce exponencialmente com o tamanho do problema. Assim, programas cognitivos usam heurísticas específicas e mecanismos de controlo para lidar com esta explosão de combinações. Mais ainda, estes programas de racionalização são maioritariamente inspirados no senso comum da compreensão do pensamento humano.

Actividade (Acção):

A acção é a natureza intrínseca de qualquer agente, estar imerso num ambiente é não poder evitar interagir com ele.

O paradigma comportamental, através de uma parte da comunidade de IA (Braitenberg 1984; Brooks 1986; Anderson e Donath 1990) [41] tomou parte num ambiente real («actividade» de aqui para a frente) como um tópico central, inspirado nas ciências da vida e aplicado aos robots móveis. Estes trabalhos sublinharam o valor da "acção situada" ("Acções tomadas no contexto de circunstâncias concretas e particulares" – Suchman, 1987) como uma boa solução do problema da actividade. Eles tratam as acções como ciclos de resposta a estímulos, espacial e temporalmente localizados: apenas o ambiente

imediato no instante presente é considerado. Em ambientes reais, este método assegura a reactividade frente a eventos imprevisíveis.

Estes trabalhos conduziram ao desenvolvimento de sistemas sensitivo-motores, produzindo agentes situados que são inspirados no mundo animal (particularmente em insectos).

Actividade e Racionalização:

Resultados práticos dos dois paradigmas aparecem em lados opostos do esquema da Figura 6:

- Agentes situados são capazes de lidar com ambientes reais mas não possuem capacidade (ou apenas possuem pouca) de racionalizar. Devido a isto, o seu comportamento é limitado a reacções simples, e são essencialmente movidos pelo ambiente.

- Os programas planeadores possuem elevada capacidade de racionalização (ao nível simbólico) mas são geralmente aplicados a domínios de estimulação devido à sua incapacidade de satisfazer contingências de resposta temporal.

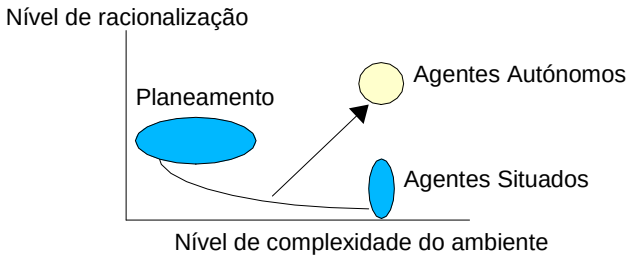


Figura 6 – Actividade, Racionalização e Actividade

Os primeiros sistemas são situados (comportamentos reactivos), os segundos são orientados por objectivos (comportamentos com finalidade). Existe uma forte convicção de que a situabilidade e orientabilidade, aparentemente opostas, formam, de um ponto de vista interior, uma base propícia para a autonomia artificial. A arquitectura desenvolvida no capítulo 6 integrará ambas as dimensões.

Alguns autores continuam a tentar reduzir a discrepância entre a racionalização e a actividade: [44], [45], descritos em [41], outros ainda, trabalham no planeamento

reactivo. Tanto quanto sabemos, o crescimento numa dimensão resulta na perda da outra dimensão. De um ponto de vista interno, consideramos que um agente capaz de ser autónomo deve ser capaz de orientar a sua actividade situada.

2.3 Visão Integrada

Desde uma visão inicial, geral e abrangente da construção deste projecto, podem identificar-se sucessivos níveis de abstracção na chamada pirâmide da computação gráfica, que introduzirão crescentes faculdades às entidades virtuais populáveis:

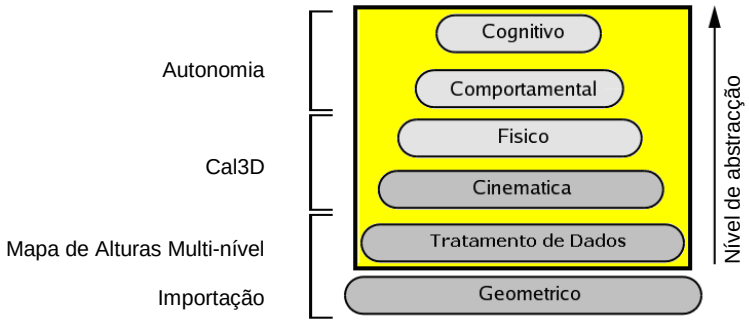


Figura 7 – Pirâmide de computação gráfica, níveis de abstracção utilizados

O nível geométrico (de mais baixo nível) é onde ocorrem as primitivas básicas de animação e a manipulação dos elementos fundamentais 3D, vértices e triângulos. Esta camada é incorporada através de mundos 3D virtuais já desenvolvidos e assim, só aqui especificada através de métodos e algoritmos para a sua importação e adaptação.

O nível de tratamento de dados é o mais importante e inovador em todo o processo, é onde se definem mecanismos de especificação, tratamento e guarda de informações extraídas a partir do nível Geométrico, que servirão de suporte à navegação, testes de colisões e definição de rotas de navegação como base de actuação e definição do comportamento dos agentes especificados nos níveis superiores.

No nível da cinemática manipulam-se os triângulos que formam o corpo dos agentes. Estes triângulos são normalmente associados a um esqueleto permitindo assim o controlo da gama de movimentos válidos para cada "osso" desse esqueleto, dos ângulos que formam, das suas dependências e na distorção que infligem aos

triângulos que compõem o corpo. Resulta assim uma representação 3D de um ser humano virtual com pretensões de movimentos credíveis.

No nível físico são representados os movimentos possibilitados pela camada cinemática, as animações criadas são disponibilizadas às camadas superiores numa forma elementar, isto é, a deslocação bípede, o caminhar. Estes dois níveis fazem uso da biblioteca de animação Cal3D apresentada mais adiante.

A nível comportamental tomam-se as decisões, definem-se objectivos e executam-se acções de reacção.

Nível cognitivo: definem-se métodos de apreensão de conhecimento do mundo onde o agente está inserido. Esse conhecimento deverá ser tido em conta pelo nível comportamental apartir do nível de tratamento de dados onde se procede à leitura e recolha de informação do mundo geométrico pré-carregado.

CAPÍTULO 3

Estado da Arte: Animação e Simulação de Personagens Virtuais

3.1 | CG: Simulação com Personagens Virtuais em Deslocação

A criação de ambientes de simulação através do fabrico de sistemas artificiais (com agentes sintéticos animados em 2 ou 3 dimensões), revelando comportamentos realistas, simulando procedimentos humanos de movimentação e interacção, corresponde a criar sistemas de agentes operando em conjunto. Efectuar simulações nestes sistemas multi-agentes é um campo de investigação actual, proeminente e muito interessante. Podemos adquirir conhecimento através do entendimento das causas, problemas e restrições do sistema, além de permitir desenvolver melhores soluções com baixo custo.

É fácil perceber, que em função das aplicações, existem no geral, dois caminhos que tem vindo a serem seguidos nas pesquisas em animação no ramo da computação gráfica: o caminho artístico, onde os esforços são dirigidos com o fim de se obterem resultados interessantes, no que respeita ao efeito visual, e o caminho da exploração científica onde são geradas animações através da simulação de fenómenos científicos [33]. È nesta última vertente que se pretende simular a movimentação e actuação de pedestres virtuais, principalmente ao nível do planeamento de trajectos, isto no que respeita à CG, envolvendo a análise da cena 3D para numa primeira fase conseguir detecção de colisões usando métodos mais apropriados, que capacitam uma extracção eficaz de informação espacial neste tipo de ambientes, para posteriormente, utilizando técnicas de IA, definir trajectos, e animar individualidades de personagens

virtuais personalizando representações de certos comportamentos inerentes aos humanos, ou grupos destes, em ambientes virtuais representativos das condições físicas associadas ao mundo real.

As primeiras simulações com personagens virtuais criadas para os primeiros jogos de computador, utilizavam apenas modelos bidimensionais ou quase unidimensionais, onde o protagonista apenas podia mover-se com um grau de liberdade (para a direita ou esquerda). Com a evolução tecnológica, muitos dos jogos actuais empregam o que há de mais moderno em hardware e software para apresentar universos virtuais em três dimensões com uma infinidade de recursos de interacção. Os processadores gráficos de placas de vídeo para computadores domésticos possuem hoje a capacidade de desenho de milhões de faces em 3D (triângulos) por segundo, usando efeitos de textura, iluminação, atmosfera, transformações, cálculo de visibilidade, transparência, cor, etc; automaticamente e em tempo real.

Em simulação, ambientes de diferentes tipos (espaços urbanos abertos ou fechados, exteriores ou interiores), habitados por humanos virtuais, podem ser aplicados, por exemplo, em projectos arquitectónicos [46], arqueológicos [47], ou no controlo de tráfego de pessoas ou veículos [48]. Além destas, também encontramos simulações de humanos virtuais em situações de emergência [49], onde a simulação do comportamento individual ou em grupo de pessoas e animais [1] se revela de grande utilidade. Na área de entretenimento os jogos são a eleição, citados como aplicações de grande potencial [50], com cenários que podem ser adaptados conforme o curso do jogo, mas também teatros [51], museus [52] e lojas virtuais [53], onde o utilizador pode navegar e interagir com outros utilizadores ou assistentes virtuais. Além destes, aplicações tais como as histórias interactivas [54], onde o utilizador é um participante activo podendo interferir no curso das mesmas, têm surgido também como uma nova forma de entretenimento. Na área educacional, a incorporação de personagens tutores [55] e a exploração de interacções multi-modais, juntamente com sofisticadas técnicas de representações da informação podem prover experiências de aprendizagem agradáveis e efectivas.

Os investigadores e programadores têm continuamente buscado caminhos para simular o mundo real com precisão. E no que respeita à simulação com personagens virtuais, um dos primeiros e principais problemas a resolver são os relacionados com a movimentação dos objectos representando as personagens no ambiente virtual. Dessa forma, a necessidade de obter mecanismos avançados para a detecção de

colisões e a definição de rotas de navegação é imprescindível para dar ao utilizador uma maior sensação de autonomia e interacção. Estas técnicas contribuem para a obtenção de um maior realismo de simulação num mundo virtual, determinando a capacidade de interacção dos objectos num dado espaço de tempo. O desafio é definir quais serão as técnicas ideais ou a combinação destas, a aplicar em ambientes de simulação específicos, levando em consideração as pretensões e complexidade destes ambientes.

3.1.1 | Detecção de Colisões

A colisão entre objectos é ubíqua na dinâmica do mundo físico, desde as moléculas até às galáxias. Consequentemente, a detecção de colisões é uma tarefa fundamental na computação gráfica aquando da simulação do mundo real. Por exemplo, em simulações de corridas de automóveis, são implementados algoritmos de detecção de colisões de modo detectar colisões entre os veículos em movimento e também com as obstruções estáticas; em simulações e treino de tiro, rápidos relatórios de colisão são requeridos para julgar se as balas virtuais atingiram os alvos pretendidos; no treino de avançadas e complexas operações médicas virtuais são simuladas detecções de colisão, onde os órgãos internos deverão ocupar um espaço único reagindo ao toque da utilização de artefactos cirúrgicos. Muitos outros exemplos de aplicações e consequentes técnicas capazes de efectuar esta detecção, são intensamente aplicadas nas mais variadas áreas, tais como a realidade virtual, robótica, desenho por computador, fabrico por computador (CAD/CAM), animação e simulação [56],[57], etc.

A detecção de colisões para múltiplos objectos em movimento sempre foi um tópico de grande interesse por parte dos investigadores, como se pode comprovar pela vasta literatura existente nesta área [56],[57],[58],[59]. Desde uma abordagem mais imediata, este tipo de detecção é analisado principalmente por uma interacção geométrica entre os objectos, e consequentemente um problema de intersecção [60]. Assim, uma forma pouco escalável e pouco eficiente de determinar quais os objectos que se intersectam (colidem) é testar todos os objectos entre si a fim de determinar quais estão em contacto. Obviamente esta técnica torna-se inviável quando se pensa em simulações mais complexas e em tempo real. Assumindo uma implementação naïve, este tipo de algoritmos assume uma complexidade do tipo $O(N^2)$, onde N é o número de objectos considerados. A situação agrava-se ainda mais, se ao invés de

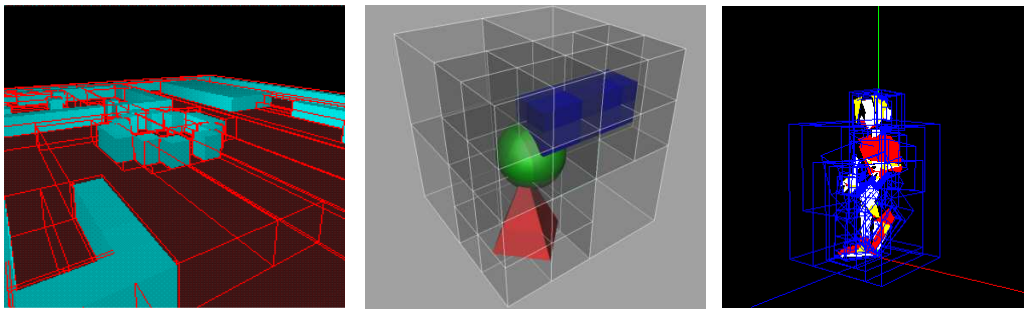
considerar N como o número de objectos, considerarmos N como o número de primitivas ou faces que descrevem e compõem os objectos (triângulos na maioria das vezes). Podemos ter assim uma cena com milhares ou milhões de triângulos, tornando inviável determinar o conjunto de objectos colidindo em tempo real.

Um outro ponto importante na detecção de colisões é o grau de precisão da colisão que se deseja obter. Diferentes simulações requerem diferentes necessidades ou níveis de precisão. Para simular navegação num ambiente virtual, pode ser suficiente a detecção de colisão entre a câmara (representando por exemplo uma personagem virtual na primeira pessoa) e os volumes envolventes associados aos objectos. Por outro lado, e tomando como exemplo uma simulação na área da física, para obter os pontos de contacto entre dois objectos colidindo, é essencial uma maior exactidão, pois, provavelmente necessitaríamos de conhecer os locais específicos de contacto entre os objectos para poder inferir resultados ou conclusões, exigindo a detecção de colisão entre os elementos "primitivos" (triângulos) que compõem a malha poligonal da geometria que representa os objectos.

Tomando em consideração as necessidades requeridas, a detecção de colisão em mundos virtuais pode se avaliada em duas diferentes perspectivas: a detecção exacta e a detecção conservativa (também denominada por alguns autores de estreita e larga respectivamente). Num primeiro exemplo, e em jeito de introdução para a distinção entre estas duas perspectivas imagine um avatar caminhando por um jardim de uma cidade virtual. Detecção de colisão conservativa pode ser usada enquanto o avatar caminha no jardim. Quando o avatar encontra e alcança por exemplo um jornal ou outro qualquer item ou objecto mais específico, então aqui seria requerida a detecção de colisões do tipo exacta.

Embora a detecção de colisões seja tradicionalmente uma solução de software, o hardware gráfico é cada vez mais usado para este propósito (na obtenção de dados para suportar a posterior detecção de colisões propriamente dita) devido aos avanços já anteriormente referidos a este nível [61],[62],[63],[64]. Basicamente, métodos de detecção de colisão assistidos por hardware têm a vantagem de tirar partido, das cada vez mais poderosas soluções de hardware dirigido especificamente para o processamento gráfico. A implementação complementar destes métodos é também bastante mais simples quando comparados com algoritmos baseados na inspecção da geometria da cena.

De um modo geral, e em jeito de representatividade de utilização tradicional, existem variadas técnicas de modo a detectar a "interferência" entre objectos geométricos, usando estas duas perspectivas de detecção de colisão, muitas destas, são relatadas no s*urvey* apresentado por Lin e Gottschalk [57]. Algumas usam estruturas de dados ordenadas hierarquicamente, provenientes de cenas em animações espacialmente subdivididas, para detecção de colisões mais ou menos precisas, e outras mais conservativas, através do uso de algumas técnicas bem conhecidas de partição de espaços, como por exemplo: *spheres trees* [65], *BSP trees* [66] e *Octrees,* também com ou sem *View Frustum* [67], e utilização de volumes envolventes: *axis-aligned bounding boxes* (AABBs) ou *oriented bounding boxes* (OBBs) *bounding boxes* [58, 59], (Figura 8).



A B C

Figura 8 – Detecção de colisões usando: Bsp, Octree e Bounding Boxes

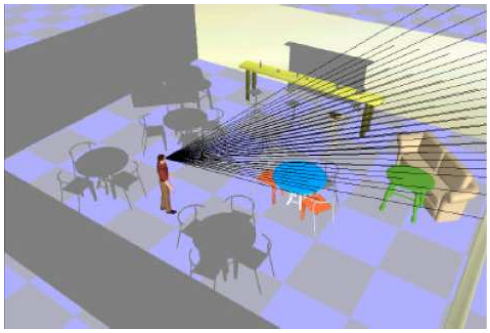
A maioria das técnicas descritas no *survey* de Lin e Gottschalk, tentam resolver o problema de uma forma mais árdua, obtendo interferências de colisão precisas entre objectos complexos. Existem no entanto imensas situações onde este tipo de análise é dispensável e até inconveniente, devido a não necessidade de incorporar uma grande precisão à detecção, e à consequente enorme carga computacional que poderá ser libertada para promover a aplicação com novas técnicas mais abrangentes e enquadramentos ambientais mais complexos para um número elevado de objectos em movimentos que os povoem. A razão pela qual este tema de investigação envolve várias páginas e até capítulos de livros da área da computação gráfica [36], deriva e envolve principalmente a necessária procura incessante na demanda da eficiência resultante das várias técnicas de detecção de colisões. Em seguida, serão analisadas com maior detalhe algumas destas técnicas abrangidas nestas duas vertentes ou perspectivas, em relação à sua aplicação, eficácia, eficiência, escalabilidade, entre outros aspectos de interesse associados à necessidade e realismo requerido para as implementações criadas neste trabalho de investigação.

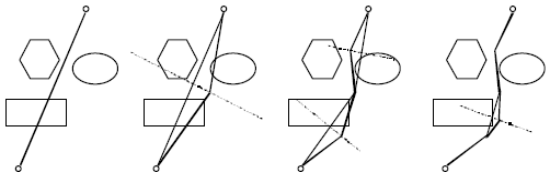
3.1.1.1 | Detecção de Colisões Exactas

O objectivo da perspectiva de colisão exacta, é calcular e devolver a especificação detalhada de todos os pontos nos objectos virtuais em eventual colisão no espaço 2D ou 3D. Este tipo de detecção é testado principalmente através de cálculos de intersecção de triângulos utilizando a geometria da cena. O custo deste género de cálculo de intersecções é directamente dependente da complexidade do modelo ou ambiente virtual em representação. O que, como é de fácil dedução, requer uma disponibilidade computacional muito elevada para cenas que envolvam uma extensa e detalhada componente geométrica principalmente tridimensional, com possibilidade de milhares de testes de colisão em simultâneo, tornando este tipo de detecção impraticável quando lidamos com cenas mais complexas.

Exemplos de trabalhos que incorporem este tipo de detecção de colisões utilizando volumes envolventes hierárquicos podem ser vistos por exemplo em [58],[68]. Em [69] são determinados conjuntos de colisões potenciais (PCS – *Potentially Colliding Set*) através de consultas à informação guardada após o uso do GPU para detecções de objectos com imagens sobrepostas, e posterior uso de técnicas com base no CPU, de modo a estabelecer detecção de colisões exactas. "*Interference Detection*" é o termo usado em [70], onde se descreve e apresenta um método para detectar e evitar colisões inspirado em volumes de sombras. Outro método relacionado nesta área de pesquisa de colisões entre o avatar e as próprias roupas é descrito em [71]. Marco Winter e Marc Stamminger [72] apresentaram mais recentemente um método de detecção de colisão preciso e portador de alguma antecipação, com base num câmara de *frustum* variável, onde são detectadas interferências a partir de um mapa de alturas tomado desde a "vista natural" do avatar, capaz de encapsular com precisão as distâncias até aos objectos, evitando os obstáculos e movimentando o avatar sem colisões.

Uma abordagem neste estilo, utilizando colisão exacta é também usada por Vosinakis e Panayiotopoulos em [73], onde o avatar "envia" raios de modo a detectar caminhos obstruídos. O método utilizado nesta abordagem implementa uma componente de detecção de colisões com o ambiente com base em *Ray Casting*. Durante o ângulo de visão de um agente são "disparados" raios para o ambiente sintético, e é feita a leitura das detecções de intersecção entre os raios e os objectos que compõem o ambiente num determinado frustum (Figura 9).





PROCURA DE TRAJECTOS COM ALGORITMO DE SUBDIVISÃO

Figura 9 – Mecanismo de detecção com base em Ray Casting e procura de trajectos

Imagem Original em [73]

Através deste método, conseguem-se identificar posições, tamanhos, e o tipo de objectos de forma precisa dispostos no campo de visão, e usar esta informação para construir um mapa da cena, permitindo a navegação evitando os objectos. São ainda usadas técnicas de procura de trajectos recorrendo a algoritmos de subdivisão em segmentos a partir de linhas perpendiculares ao trajecto (Figura 9), este algoritmo funciona de forma recursiva em relação aos novos segmentos criados, até se conseguir identificar um percurso sem colisões. Este trabalho reflecte também o muito esforço na construção de ferramentas de apoio a programadores no desenho, síntese e comportamento de imagens e animações 3D de ambientes virtuais com agentes ou avatares, fornecendo características como: cinemática inversa, modelação baseada em elementos físicos (*bones* e *joints*), detecção de colisões, respostas a estímulos e visão 3D.

Em [74], Redon et all, apresentam um algoritmo para detecção de colisões entre o movimento de um avatar e o ambiente virtual circunvizinho. Esta implementação, utiliza a posição e orientação do avatar em etapas de tempo discretas, para, usando um movimento de interpolação entre uma posição inicial e uma final, conseguir inferir com exactidão possíveis interferências de objectos neste trajecto (Figura 10).

Figura 10 – Mecanismo de detecção de colisões com base em interpolação

Imagem Original em [74]

È assim utilizada uma espécie de varrimento de volume no espaço a analisar (*swept volume*) entre um determinado posicionamento inicial e final. Uma nova interpolação de retorno à posição original (movimento inverso) permite obter os tempos de computação e posicionamento espacial da possível colisão bem como a posição do avatar no momento que ocorre a colisão. Deste modo, é conseguida uma representação precisa, detalhada e organizada de forma hierárquica, do conhecimento do espaço ocupado por objectos presentes em determinados locais da cena. No entanto, e como seria previsível para esta detecção de colisão do tipo "*in-between motion*" durante o percurso do movimento do avatar, apresenta um consumo de recursos computacionais elevado devido à análise sucessiva e em tempo real da informação resultante da interpolação para todos os movimentos das componentes articuladas do avatar, limitando o número de avatares nestas condições, bem como animações em ambientes mais complexos.

Algumas conclusões menos positivas associadas à pretensão de simular multidões humanas em movimento generalizado após rever a utilização de técnicas abrangendo a detecção de colisões sob a perspectiva exacta, resultam de incorporarem uma precisão muito detalhada, por vezes desnecessária e restritiva. Devido à grande quantidade de objectos em movimento e ao tempo inerente necessário para apresentar em tempo real a simulação da movimentação de acordo com a detecção de colisões de forma realista de todos eles, terão de ser encaradas outras soluções, capazes de negociar e lidar com menor precisão na detecção de colisões, sem uma necessidade explícita na exactidão, mas, provocando como contrapartida, um potencial muito superior de eficiência, possibilitando representações muito mais escaláveis.

3.1.1.2 | Detecção de Colisões Conservativas

Outra grande parte dos desenvolvimentos nesta área tenta resolver este problema de uma forma menos precisa, mas com uma envolvência superior adaptada às necessidades específicas, muito mais eficiente no consumo dos recursos de hardware e software, e escalável de modo a poder fazer face a um número elevado de objectos em movimento. As várias técnicas de detecção de colisões via perspectiva conservativa (ou de banda larga), são comummente utilizadas quando um grande número de personagens virtuais se encontra em movimento num mundo virtual, e à necessidade de apresentá-los com certo grau de realismo em tempo real. Nestas

condições, basicamente são necessárias duas peças de informação fundamentais: a posição onde se encontra o avatar num determinado momento, e a informação se pode avançar sem colidir. A utilização de técnicas de colisão baseadas neste tipo de análise com base no terreno, podem fornecer em alguns casos também a altura a que cada avatar será posicionado, assumindo que as primitivas gráficas que constroem o terreno sejam claramente identificáveis. Quando utilizadas estas técnicas são evidenciados os primeiros passos para avançar para a navegação verdadeiramente tridimensional. Se esta análise abranger também ambientes desconhecidos à partida, então, é por norma requerido e reservado, um tempo inicial para o pré-processamento (conhecimento e análise) do novo mapa de navegação.

Conseguir eficiência na simulação baseado no conceito de seguir um mapa de navegação capaz de representar um terreno navegável, surgiu inicialmente como um problema ligado à robótica. Onde, por exemplo Lengyel et all [75] exploraram mecanismos de rasterização do espaço via hardware de modo a gerar células configuradoras e representativas do ambiente de simulação, usadas para determinar trajectos livres de obstáculos para a navegação de robôs.

Uma pretensão mais realista para a representação de humanos virtuais navegando em tempo real em ambientes desconhecidos à partida, com capacidade de detectar e evitar colisões neste enquadramento, deveria ter em linha de conta, a capacidade do avatar caminhar por cima ou por debaixo de obstáculos existentes no mundo virtual, a que aqui é chamada de navegação multi-nível ou verdadeiramente 3D. Mas em quase todos os casos da investigação neste campo, é traduzido o mapeamento do ambiente 3D, numa simples grelha de navegação planar (2D), ou em outros casos, incorporando a esta, também a informação da elevação do terreno para cada ponto de navegação (2,5D). O que não traduz muitas das vezes uma amplitude alargada de movimentação e liberdade aos agentes povoadores de diversos modelos virtuais com capacidades de representar ambientes reais em todos os domínios físicos.

É neste quadro de aparência de navegação tridimensional que investigadores como Chrysanthou, Tecchia, Loscos e Conroy têm dado ênfase à componente de sistemas para visualização de personagens virtuais autónomas em ambientes virtuais urbanos. Todos estes investigadores, basearam os seus trabalhos em pesquisas de investigadores anteriores que tinham por objectivo, reduzir o processo computacional da detecção de colisões, através de uma discretização do espaço a navegar. São os casos das pesquisas apresentadas por Rossignac et all [76] , e

Myszkowski et all [77], nos meados da década de 90. Nestas pesquisas, era já dada prioridade à utilização do hardware gráfico para proceder à rasterização necessária do espaço, de modo a encontrar interferências entre modelos em movimento. No entanto os seus trabalhos ficaram confinados a simulações utilizando um pequeno número de objectos CAD.

Retomando as pesquisas e desenvolvimentos mais actuais, é apresentada seguidamente de forma mais detalhada, dois dos trabalhos desenvolvidos por Tecchia et all, na área e componente específica aqui abordada:
Em [5] é apresentada uma proposta de um sistema que integra componentes para detecção de colisões com regras de comportamento de alto nível, e ainda a definição de acções específicas para determinadas áreas, como por exemplo esperar em paragens de autocarro. No entanto este sistema não especifica a interacção com objectos, nem a possibilidade de trabalhar em dois ou mais níveis de altura, como por exemplo um cenário de um parque de estacionamento com vários pisos. Assim, foi construída uma plataforma que permite ao utilizador desenvolver e visualizar o comportamento de um número elevado de agentes. Adoptou-se uma grelha 2D sobre a qual os agentes navegariam (Figura 11). Combinando o efeito de vários níveis, cada agente individual reagiria dependendo do nível testado, da área por ele ocupada e a sua posição relativa em relação a outros agentes. De modo a simular diferentes tipos de comportamento, a plataforma foi constituída por 4 camadas ou níveis de teste:

1ª - Camada de detecção colisões com outros agentes.
2ª - Camada de detenção de colisões com o ambiente.
3ª - Camada de comportamentos básicos (diferenciados por zonas).
4ª - Camada comportamentos mais complexos (funções de retorno por célula ocupada).

Figura 11 – Simulação de comportamento e teste de colisão utilizando 4 camadas de teste

Imagem Original em [5]

Usando um cenário simples como exemplo, podemos exemplificar a utilização destas 4 camadas: " Um agente passeando pelo pavimento com destino a uma paragem de autocarro, enquanto caminha, ele desvia-se de cabines de telefone, quiosques e de outros agentes, até parar e aguardar a chegada do autocarro, quando este chega a função chamada de retorno é activada, causando o saltar do agente para dentro do autocarro."

Em [2], os mesmos autores apresentam também um mecanismo interessante que determina automaticamente a altura a que se devem posicionar e movimentar personagens virtuais para simular o movimento de andar ou correr num determinado modelo com diferentes alturas, recorrendo a um mapa de profundidades guardado a partir de uma vista superior sobre o ambiente a navegar. Foi construído um sistema, em que partículas (representando agentes em movimentação) detectam colisões entre elas e o ambiente envolvente. A ideia passa pela representação discreta e estática do mapa de alturas 2D a partir de uma vista superior de um modelo 3D de um ambiente a simular, via rasterização (Figura 12). O mapa resultante mantém em memória (recorrendo ao *z-Buffer*), e para cada *pixel,* o valor da altura que será testado em cada *frame* da simulação, antes do movimento de cada partícula. Se um dado valor do mapa de alturas for igual ou dentro de um valor estabelecido como suportável de ser ultrapassável em relação à capacidade individual de deslocação guardado por cada partícula, então estas, poderão avançar, senão é detectada uma possível colisão e, testado e iniciado o movimento noutra direcção.

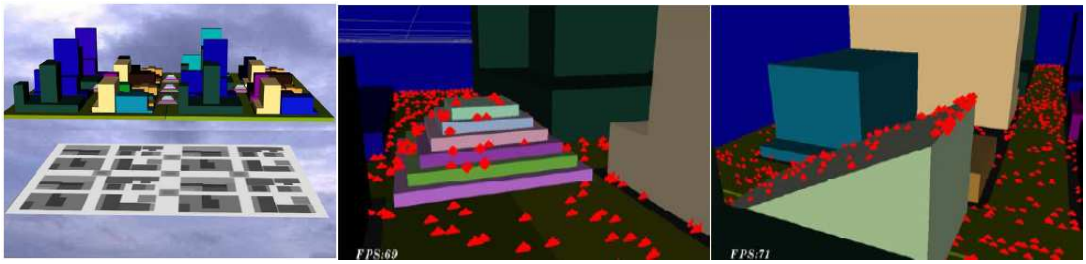


Figura 12 – Mapa de alturas a partir de um modelo 3D e comportamento de partículas na presença de desníveis graduais e evitando obstáculos

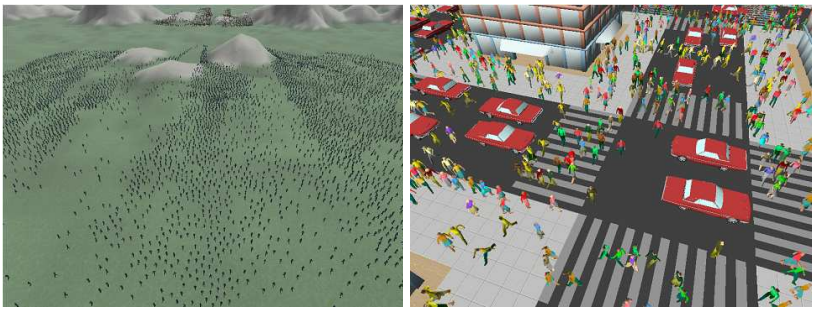
Imagem Original em [2]

Nos casos citados anteriormente, mesmo que a geometria se mantenha em 3D, o aspecto da navegação de simples agentes ou multidões é restrita a uma superfície 2D ou 2,5D. Um dos principais problemas destes sistemas é a incapacidade do

mapeamento automático poder detectar vários níveis de navegação em altura (tais como as existente em pontes, passagens subterrâneas ou simples níveis de acesso interiores a casas ou prédios) e as consequentes limitações com os mais variados modelos que reflectem o mundo real, gerando consequentes perdas na capacidade de provocar maior abrangência de movimentação e consequente maior capacidade de um realismo mais alargado à animação.

Técnicas semelhantes com contributos e aplicações de modelos para simulação de multidões em terrenos 2,5D, podem ser encontradas em [4, 48, 78-81] entre outros muitos. São destacadas em continuação, duas destas, apresentadas recentemente. A primeira referenciada pela dinâmica de movimentação e detecção de colisões apresentada, no segundo caso pela extracção automática do mapa de navegação e consequente modelo integrado e sequencial, por forma a acautelar colisões e promover rotas de navegação em direcção aos objectivos de deslocação.

Assim, Treuille et all [81] apresentam um modelo que integra navegação global com movimentação de objectos representando obstáculos, interagindo com a movimentação de personagens virtuais, sem a necessidade explícita da detecção colisões exacta. Este modelo demonstra a fluência da movimentação e interacção entre as personagens e o dinamismo de uma cena específica sob uma grande variedade de situações, permitindo exibir e analisar fenómenos emergentes da dinâmica de multidões, também observadas no mundo real.



A B

Figura 13 – Variedade de cenários simulados com recurso ao modelo de Treuille

Imagem Original em [81]

Os mundos a simular, não são no entanto desconhecidos à partida, não permitindo uma utilização e escalabilidade de modelos mais alargada. A superfície de movimentação é subdividida em pequenas regiões ou células (por norma do tamanho

dos avatares em ambientes detalhados), onde são impostas (para a detecção de possíveis colisões) as localizações estáticas de objectos imóveis (como casas, ou prédios), e localizações dinâmicas de objectos em movimento (avatares, automóveis, etc). O modelo permitiu simular a dinâmica de um exército de perto de 2 mil guerreiros (simulados por partículas) numa grelha de 60x60 (Figura 13-A), bem como a fluência das movimentações de personagens virtuais numa movimentada praça de uma grande cidade (Figura 13-B), com utilização de uma grelha de 120x120 células, com cerca de mil avatares em movimentação.

O segundo caso pode ser visto em [4], onde Gonzaga da Silveira e Musse apresentam uma estrutura genérica, capaz de suportar a criação semi-automática, de gestão e visualização de complexos modelos urbanos com simulação virtual humana, à qual dão o nome de "Virtual Urban Life" (VUL). Para a criação desta estrutura são necessários certos *inputs* apropriados, tais como: a geometria do modelo de dados ou cena a povoar, edifícios e artefactos presentes na cidade, informação suplementar da especificidade de algumas áreas do modelo, bem como o tipo e características associadas às personagens virtuais a incluir. A caracterização do terreno é descrita através da função $f : R^2 \rightarrow R$, $z= f(x,y)$, onde x, y são as coordenadas no plano e z representa os correspondentes valores de elevação.

O modelo de importação de dados e o seu enquadramento posterior é abordado em repositórios de níveis de abstracção distintos. A relação entre estes diferentes níveis permite um uso e transição de informação mais eficiente adaptada às necessidades do nível em questão e as suas interacções com os níveis contíguos. A partir da geometria importada (definida como nível 0), e em fase de pré-processamento, é deduzida uma nova rede poligonal mais abrangente do terreno a povoar, de modo a eliminar a complexidade relativa inicial, e conseguir uma malha poligonal mais linear, e de tamanho adaptado às necessidades específicas. Esta nova representação do espaço (definida como nível 1), pode servir como um re-arranjo das futuras localizações dos objectos a associar, e, poderá incorporar subdivisões tão abrangentes como ruas, passagens específicas, espaço físico para colocar edifícios, artefactos, etc.

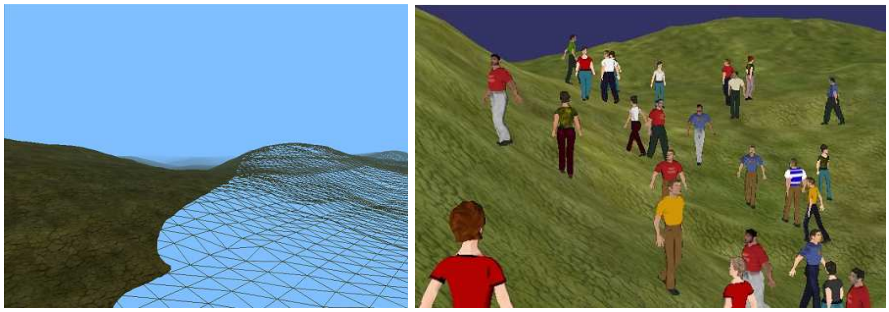


Figura 14 – Importação da superfície geométrica a povoar e a sua adaptação à simulação
Imagem Original em [4]

A detecção de colisões é então realizada como um tipo de interruptor entre a análise dos níveis 0 e 1, para detecção mais ou menos precisa entre objectos em movimento e objectos estáticos. Foram conseguidas simulações incorporando até 1250 pedestres virtuais, caminhando em superfícies abertas, ou em cenas simulando ruas de uma qualquer cidade formadas por blocos de edifícios.

Uma outra solução que se concentra no caso das navegações 2,5D, mas, onde já se consideram alguns elementos possibilitando navegação a diferentes níveis de altura como é o caso de pontes ou alguns desníveis é apresentada em [82]. Este artigo está direccionado para as investigações da chamada navegação geral [83], utilizando técnicas de exploração livre sobre um determinado modelo e não sobre caminhos ou destinos predeterminados. Como foi apresentado atrás, existem várias técnicas para moldar o movimento sobre superfícies planares com base em grelhas sobrepostas em diversos terrenos, no entanto estas técnicas não tratam colisões com superfícies verticais de uma forma adequada, subestimando objectos e zonas do ambiente que se deseja realmente 3D.

Assim, a investigação em [82] faz uso do método BSP, onde o mundo virtual é decomposto em células que se vão interligando ao longo dos seus limites. Este método foca a exploração de complexos modelos e cenas 3D, onde existe uma livre exploração do ambiente multi-nível. A técnica descrita neste artigo assenta sobretudo na criação de uma estrutura de células no espaço x,y e z, criada pela decomposição dos objectos presentes numa cena, em regiões demarcadas pelas suas faces e vértices. Quando um agente se localiza numa determinada célula, conhece as células seguintes de possível navegação dependendo da direcção a tomar, e cada célula mantém propriedades da face do polígono associado, como a altura, dimensão e ligação a outras células (Figura 15).

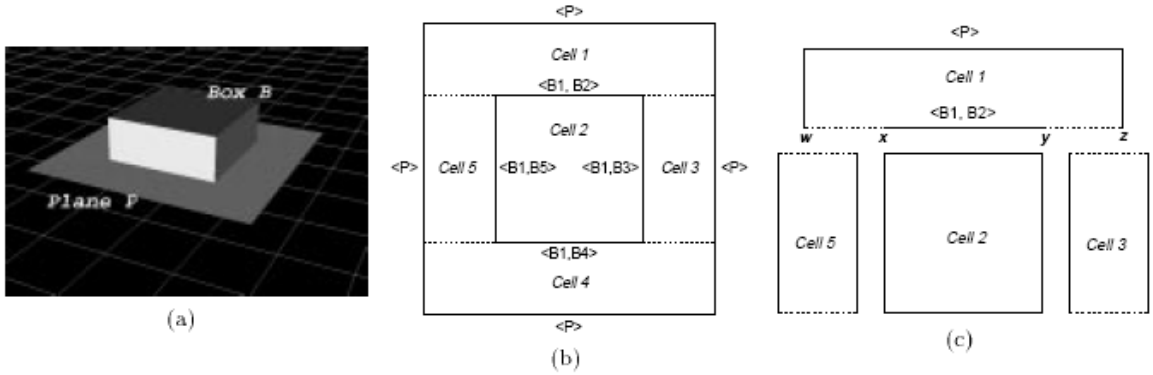


Figura 15 – Exemplo de uma cena e a respectiva decomposição em células de objectos

Imagem Original em [82]

Esta estrutura é extremamente eficiente para testar elementos estáticos. Podem ainda ser acrescentados ou eliminados novos elementos uma vez que a estrutura de células esteja construída. Este método simplifica muito a técnica de determinar o tipo de superfície a considerar em qualquer ponto da cena, podendo distinguir diferentes níveis de navegação. O problema reside no facto de termos de lidar com modelos com um número de faces elevado (+ de 10.000 faces), já que o tempo de construção desta estrutura de células é bastante significativo. Se por um lado podemos lidar com a exploração verdadeiramente 3D do ambiente, por outro, o problema coloca-se na mesma grandeza da análise da perspectiva das colisões precisas, ou seja, na impossibilidade de lidar com cenas mais complexas, devido também à imensa carga computacional requerida na decomposição do ambiente, o que inviabiliza uma simulação em tempo real ou uma fase de estudo em tempo comportável para a análise da geometria para ambientes desconhecidos.

Um grande volume de trabalho dedicado ainda a esta área de estudos científicos é desenvolvido no laboratório de realidade virtual (VRlab) do *Swiss Federal Institute of Technology*, liderado por Daniel Thalmann, um dos pioneiros das investigações utilizando humanos virtuais. A investigação encontra-se repartida entre a definição de sistemas para atribuir comportamento autónomo e a qualidade de visualização/modelação de personagens virtuais, detalhando o comportamento destas personagens individualmente ou em grupo, animação de movimentos, e aspectos cognitivos e emocionais de interacção social. Este conjunto de trabalhos representou e representa ainda um grande avanço científico no desenvolvimento de humanos virtuais. Nesta linha, em [84], é apresentado um método automático de movimentação global, através da geração de caminhos "óptimos" entre células num

espaço discretizado, baseado numa grelha 3D composta de divisórias uniformes. Cada célula mantém atributos para expressar as condições físicas (ex: parede) ou um estado (ex: ocupada). Os testes de colisão com base nesta subdivisão do espaço e teste de células habilitadas à navegação ou não, são apresentados em investigação anterior por parte dos mesmos autores, onde são colocadas diferentes etiquetas a cada célula, de modo a conhecer a sua ocupação a várias alturas [85]. Áreas não navegáveis (consideradas obstruídas aquando da sectorização horizontal e vertical do espaço 3D) são apontadas como acessíveis ou inacessíveis, com o recurso a um algoritmo de procura de caminhos alternativos para os circundar (Figura 16).

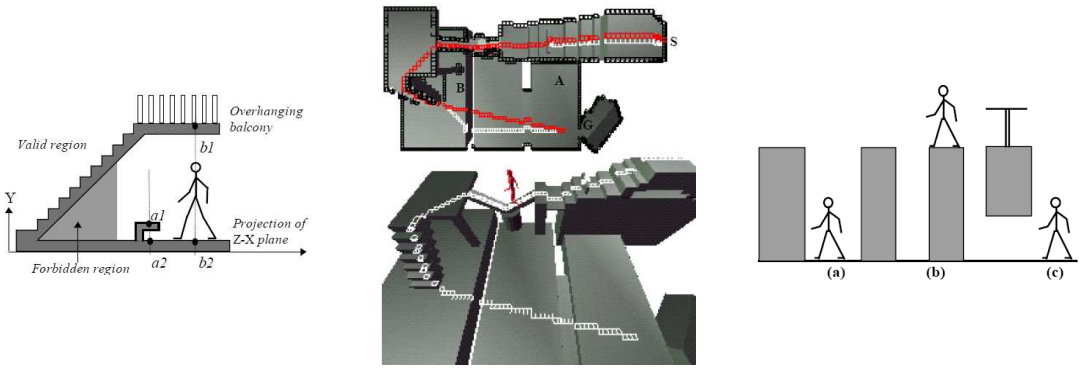


Figura 16 – Teste de caminhar sobre escadas

Imagem original em [84]

Em ambientes 2D, a localização dos avatares é dada por uma função que tem em conta unicamente o plano horizontal, isto significa que só poderá existir uma única localização válida para o posicionamento horizontal de um dado avatar. Num espaço verdadeiramente 3D isto já não acontece, podendo esta função comportar múltiplos valores. A investigação do trabalho científico agora analisada levada a cabo por Bandi e Thalmann, reflecte já esta preocupação, a de uma navegação livre num ambiente onde são analisadas algumas situações de deslocação e colisão tendo em conta não só uma única localização horizontal no mapa planar. Para isso, a discretização envolvendo o espaço tridimensional, foi implementada através de uma análise e divisão sectorial do espaço em Y (estipulada inicialmente) a partir de um enquadramento e limitação do espaço da cena caracterizada agora por um *bounding box*. Cada camada resultante foi analisada e computada como um novo *clipping volume*, de forma a serem tratadas sucessivamente através da utilização do *framebuffer* que permitiu projectar esta nova informação no mesmo *bounding box*, e alcançar indicações de sobreposição de espaços e a resultante informação da ocupação ou não das células presentes em cada camada de teste com base na altura do

avatar. No entanto, não é considerada neste trabalho a altura/profundidade da cena ou dos obstáculos (terceira dimensão) de uma forma contínua, impossibilitando o tratamento da altura dos objectos ou espaços da cena de uma forma mais abrangente e não contida à simples condição de passagem ou não passagem. Esta simplificação dificulta a sua aplicação a mundos desconhecidos mais complexos e com grande variedade de navegação e acessos a vários níveis de altura (por exemplo estruturas representando vários níveis interiores navegáveis de edifícios) e a aplicação de métodos de navegação que levem em linha de conta o peso dos virtuais esforços das subidas ou descidas. A discretização baseada neste tipo de processamento de uma matriz de células 3D de todo o ambiente, revelou-se também pouco eficiente de implementar para detecções de sobreposições em cenas mais complexas, ou com a incorporação de uma variedade populacional de varias dimensões e tamanhos. Outra desvantagem é a impossibilidade de prolongar e reutilizar este método para a detecção de colisões entre as próprias personagens virtuais.

Mais recentemente em 2006, as investigações neste laboratório (VRlab), geraram um estudo comandado por Julien Pettré [3], onde se faz referência explícita à necessidade de construção de plataformas, capazes de forma automática, decompor e preparar um qualquer modelo tridimensional, de modo a poder animar e planear em tempo real, rotas ou percursos de navegação para grandes quantidades de humanos virtuais, incluindo detecção de colisões com objectos estáticos presentes no ambiente, de uma forma verdadeiramente tridimensional, considerando para isso uma análise sectorial de todo o volume da cena.

Neste mesmo artigo [3], e com base em estudos anteriores dos mesmos investigadores [86], é apresentada uma solução para a especificação de um grafo possibilitando navegação tridimensional, extraído em período de pré-processamento, de um ambiente de navegação realmente 3D, usado posteriormente para tarefas de navegação e simulações mais realistas, neste tipo de envolvência. Estes autores fazem uso de uma sectorização do volume da área do ambiente navegável, utilizando uma vista superior do modelo, e localizando para os mesmos pontos do plano x,y, diferentes níveis de altura onde um determinado humano virtual com uma altura específica poderia navegar. Esta fatiação é realizada a alturas iguais às da altura definida para as personagens estipuladas para povoar a cena. Assim, recolhendo e testando as várias alturas de pontos da geometria do modelo, recorrendo ao valor destes facultados pelo z-Buffer, são interligados (prevalecendo o valor mais elevado), aqueles que sendo adjacentes, a sua equidistância não supere a altura das personagens

virtuais a incluir na animação. Uma das principais dificuldades sentidas pelos autores, nesta etapa (fase 0 - identificada assim por ser a primeira actuação em pré-processamento), foi a de conseguir extrair a altura e lidar com obstáculos verticais, a partir de uma vista superior e perfeitamente perpendicular a estes objectos, que apareciam invisíveis no mapa de profundidades. A solução de recurso utilizada (e quase que podemos considerar como de força bruta), foi a de complementar comparações morosas de mais duas vistas ortográficas (esquerda e frontal), para testar a presença deste tipo de obstáculos em todos os vértices da estrutura geométrica do ambiente virtual. É de notar assim a ausência deste tipo de estruturas nos casos exemplificativos da utilização deste método. Após esta primeira preparação da cena, surgia o problema de estipular a que níveis de guarda de informação de acessos em altura se situariam os avatares aquando a sua navegação em espaços sobrepostos e multi-níveis em altura. Assim, foi construído um grafo de navegação tridimensional dado pelo volume de cilindros consecutivos (com a altura dos avatares) deduzidos a partir da grelha de navegação em altura, conseguida anteriormente (Figura 17). O diâmetro destes cilindros corresponderia às áreas navegáveis num determinado espaço cilíndrico x,y,z, geradores assim dos nodos do grafo de navegação, onde as interligações do mesmo se dariam nas intersecções dos pontos entre cilindros adjacentes.

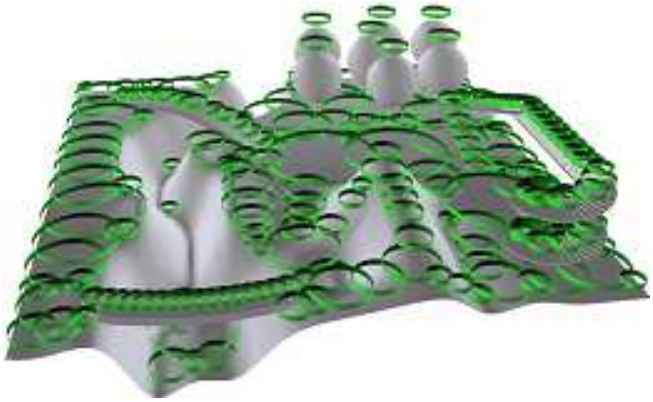


Figura 17 – Exemplos da composição do grafo de navegação em multi-nível a partir de áreas cilíndricas adjacentes

Imagem original em [86]

Este método apresenta no entanto algumas limitações, em termos de eficiência e escalabilidade de navegação, mas principalmente um elevado consumo de tempo e recursos, na fase de pré-processamento e preparação da cena virtual a povoar. Refira-se em jeito de exemplo, que a cena pouco complexa da Figura 17, demorou cerca de 1 minuto a ser computada para este tipo de método (num Pentium IV, com 1Mb de

Ram). Isto por várias razões. Uma delas é a dificuldade sentida pelos autores de lidar com obstáculos verticais, já que apresentados desde uma vista superior aparecem invisíveis no mapa de profundidades. Tiveram de recorrer para estes casos, a comparações morosas de mais duas vistas ortográficas (esquerda e frontal), para testar a presença deste tipo de obstáculos. Um outro problema que pode influenciar a dificuldade de composição do grafo de navegação, é a presença de muitos e pequenos objectos na cena (principalmente localizados em multí-nivel e zonas de navegação mais exíguas e complexas), capazes de gerar inúmeros nodos (cilindros) e respectivas interligações que irão ser testadas entre cilindros adjacentes no plano x,y, bem como o teste de interligação de cilindros desnivelados e intersectados em níveis de altura também em intersecção (por exemplo aqueles presentes nas escadas ou rampas de acesso a diferentes níveis de navegação em altura). Também não estão garantidas as chamadas detecções de colisões locais, ou seja, entre as próprias personagens que compõem a animação, já que o método global de pesquisa não fomenta esta especificidade.

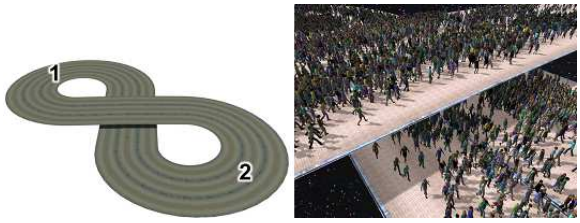


Figura 18 – Exemplos de simulação em cenas menos complexas de navegação realmente 3D
Imagem original em [3]

De referir que este tipo de abordagem ao problema e resolução do mesmo utilizada por este autores, foi já referenciada em alguns trabalhos anteriores ligados à robótica [87],[88], na tentativa de construção de uma plataforma capaz de lidar com terrenos "rugosos" e desnivelados para o planeamento de movimentação de robôs neste tipo de condições.

Resumindo, os sistemas de navegação em ambientes virtuais mais utilizados para um número elevado de personagens em movimento, empregam técnicas de navegação via perspectiva, para por exemplo, caminhar sobre um plano, ou mesmo voar no espaço. Tipicamente são usadas técnicas de mapeamento dos objectos do ambiente 3D para 2D ou 2,5D. Posteriormente é efectuada uma análise numa dimensão reduzida do espaço utilizando mapas subdivididos em células, onde cada uma destas contém informação sobre as condições de navegação nessa posição específica. Não é

prestado neste tipo de trabalhos, especial atenção à navegação livre de personagens virtuais em modelos arbitrários mais complexos como factor de realismo e pluralidade. As metodologias encontradas também não se provaram escaláveis e reutilizáveis no sentido da utilização do mesmo método base para o suporte das necessidades básicas a uma navegação tridimensional realista (detecção de colisões e constrangimentos de deslocação com o ambiente e os próprios objectos em deslocação, pluralidade na utilização de entidades virtuais em deslocação, a preparação da informação como suporte ao planeamento de percursos aproveitando a terceira dimensão do ambiente como informação contínua e relevante para implementar mecanismos mais realistas de interacção com este; tudo isto envolvendo uma eficiente utilização dos recursos computacionais).

O estudo de fundo efectuado sobre esta actualidade investigacional, deixa transparecer alguns problemas ou necessidades evidentes por resolver nesta área de actuação. Por um lado a quase totalidade das técnicas bem conhecidas e largamente utilizadas para a simulação de movimentações de personagens virtuais em terrenos 2D ou 2,5D desconhecidos à partida, não se adequam a ambientes com as três dimensões como liberdade de actuação, perdendo claramente no realismo e flexibilidade durante a simulação. Este tipo de técnicas consegue mapear apenas a área superior do plano vertical e expandir o nível de actuação deste, à movimentação horizontal utilizando os valores finais de *rendering*, considerando unicamente a altura superior navegável da estrutura que existe neste tipo de ambiente, tomando o exemplo de uma casa com vários andares, unicamente o telhado seria assim habilitado à movimentação. Por outro lado, algumas técnicas que teoricamente proporcionariam esta liberdade de movimentação, são altamente limitativas, como se comprovou e concluiu anteriormente pela análise dos trabalhos em investigações anteriores em relação à complexidade do ambiente a povoar, como por exemplo a utilização da malha poligonal da geometria da cena usada directamente nas pesquisada para o posicionamento e direccionamento das personagens em movimento, ou mesmo uma decomposição da estrutura geométrica e a construção de uma estrutura tridimensional mais abrangente, (como por exemplo BSP com octree´s) projectando desta forma, um tempo de construção bastante significativo com limitações ao nível da complexidade do ambiente e consequente gestão do espaço tridimensional para projectar simulações em tempo real de qualidade em condições mais exigentes.

A discretizaçao do espaço em malhas regulares ou não, produzindo grelhas de células espaciais acopladas e identificativas de localizações específicas das zonas potencialmente acessíveis durante a movimentação dos pedestres virtuais, revela-se um processo eficiente e escalável, mas até agora muito limitado à navegação bidimensional. Poderá questionar-se o porquê desta limitação. A resposta terá obrigatoriamente duas vertentes. Se por um lado, e até há bem pouco tempo não se sentiu a necessidade de lidar com a distinção de independência entre a construção de mundos virtuais e a inclusão de seres que os habitassem, onde estes eram arquitectados à medida das movimentações específicas e programadas das personagens a incluir. Deste modo eram conseguidas movimentações com todos os graus de liberdade e realmente 3D, mas limitadas a locais previamente conhecidos e estipulados, chamados por vezes também de caixas de "Petri"[11] controladas e experimentais. Este tipo de adequação é realizada recorrendo a técnicas custosas, e em período de programação, envolvendo um conceito de não transparência e pouca ou nenhuma flexibilidade ou re-utilização. Através de uma análise mais detalhada, podemos perceber que isto acontece comummente nos casos de filmes, spots publicitários e, as bem conhecidas animações e simulações em diversos jogos de computador envolvendo personagens virtuais em movimento e interacção. Quando se passa a ponderarem-se plataformas usando independência e autonomia entre a geometria e a simulação a incluir, os trabalhos são direccionados automaticamente para o tipo de navegação mais recorrente entre os humanos (navegação bidimensional), muito condicionados também pela disponibilização de modelos globais e pouco detalhados, simulando para o exemplo das cidade virtuais unicamente a navegação pelas ruas e espaços abertos das mesmas. Nesta vertente, temos que a utilização dos métodos conhecidos de discretrização de espaços virtuais desconhecidos à partida, não se adaptam totalmente em múltiplas condições, à nova pretensão mais realista e evoluída de mapear de forma automática geo-virtualmente em 3D, as zonas de navegação destes novos ambientes.

Terão de ser repensadas novas plataformas, que mantendo todas as qualidades de independência, eficiência e escalabilidade das simulações, proporcionassem agora uma abertura superior ao realismo para uma nova etapa de navegação verdadeiramente tridimensional em qualquer espaço apto para esta condição, sem necessidade de uma preparação específica e prévia do modelo 3D para o efeito, abrindo-se assim, novas avenidas e factores de realismo para que qualquer mundo

[11] Por alusão ao controlo do cultivo de alguns micróbios em ambientes fechados (recipientes cilíndricos de vidro ou plástico) usados na Microbiologia.

virtual simule um ambiente real de modo mais transparente e abrangente para os utilizadores finais.

Quando necessitamos explorar virtualmente ambientes mais complexos com multiplicidade de espaços de navegação que incluam passagens à vários níveis de altura, como em situações interiores de edifícios em altura utilizando espaços inclinados ou desnivelados, é desejável que as aplicações mantenham os participantes (agentes ou avatares) em movimento numa posição e condição de altura realista em relação ao terreno e espaço envolvente. Para este tipo de terrenos ou cenas, com análise do espaço navegável a diferentes níveis de altura, as técnicas comummente utilizadas não se aplicam, ou apresentam ainda alguns constrangimentos.

Simular neste tipo de ambientes ou estruturas arquitectónicas de edifícios virtuais mais ou menos complexas, onde rampas, escadas ou outras passagens desniveladas entre os vários níveis de navegação à mesma altura são recorrentes, bem como a presença de múltiplos objectos capazes de ser transponíveis ou não nos vários níveis de altura, transportando este tipo de simulação para qualquer ambiente realmente 3D, sem necessidade de qualquer conhecimento à partida, e gerando respostas de colisão em tempo real, comportando uma envolvência de navegação realmente 3D, são abordagens complexas onde terão de ser utilizadas especificações de detecção de colisões e planeamento de percursos mais eficientes, tendo em conta o espaço tridimensional circundante.

3.1.2 Planeamento de Percursos – Áreas de Desenvolvimento e Métodos Utilizados

Existe um extenso esforço reflectido em trabalho científico relacionado com a navegação em ambientes virtuais. Segundo Brian Salomon [89], estes trabalhos podem ser classificados em duas grandes categorias: os que procuram entender os princípios cognitivos que estão por detrás da navegação, e os que actuam no desenvolvimento de técnicas de navegação em aplicações e situações específicas. Exemplo dos primeiros podem ser percebidos em [90], onde são desenhados e explorados princípios cognitivos associados e aplicados à navegação em grandes ambientes virtuais, outro exemplo de trabalho considerável nesta categoria é o exposto por Li e Ting [91] no desenho de interfaces mais "amigáveis" com o utilizador de forma a melhorar a performance de navegação. Na segunda categoria

encontramos diversos trabalhos que enumeram, expõem, classificam, aplicam e testam diferentes técnicas usadas na navegação de ambientes virtuais. É esta categoria que vai servir de enquadramento ao trabalho apresentado no capítulo 5.

A maioria destes trabalhos de investigação nesta área usa uma etapa de pré processamento para o cálculo de mapas de percursos através da procura em grafos hierárquicos. Mas será que estes abordam a procura de percursos em cenas realmente 3D? Estes algoritmos terão realmente em linha de conta por exemplo os vários níveis sobrepostos e navegáveis em altura na procura de destinos, e as necessidades específicas na tomada de decisões aquando as mudanças de alturas (subidas e descidas de escadas ou rampas para alcançar mais eficientemente os objectivos)?

Conhecer quais os princípios que regem a navegação no interior ou exterior de modelos e ambientes 3D é falar de analisar o próprio ambiente virtual e obter a partir daí, métodos capazes de detectar colisões e simular também navegação autónoma com determinados objectivos. Quando consideramos a possibilidade de abordar mundos virtuais com vários níveis (sobrepostos ou não) de navegação em altura, a maior parte das técnicas utilizadas na procura de rotas não incorporam um planeamento adaptado e específico para estas condições, ou então não são eficientes para projectar deslocação assistida a pedestres em ambientes arbitrários e complexos.

De seguida são introduzidas algumas aplicações neste âmbito em áreas de desenvolvimento paralelas, formalizando um ponto de partida para o estado da arte neste campo específico.

3.1.2.1 | Áreas de Desenvolvimento

Robótica: A representação de ambientes navegáveis, onde se inclui a procura de rotas e percursos de navegação tem sido extremamente estudado no campo da robótica onde a navegação é reconhecida como indispensável [92]. Alguns dos algoritmos colocados em prática para um planeamento de movimentação global são baseados tomando exemplos alternados da configuração de espaços na movimentação de robôs apresentados por Kavraki e Overmars [93],[94]. Em contraste com a área da robótica, os algoritmos de *path planning* para a navegação de humanos virtuais em ambientes virtuais não utilizam sensores de captação de dados, mas concentram esforços na análise da estrutura e composição do ambiente virtual. O planeamento de percursos

ou simplesmente de movimentações imediatas é nesta área um campo de investigação sempre em aberto.

Jogos: A procura e definição de percursos são dos mais visíveis tipos de inteligência artificial (IA) encontrados no desenvolvimento de jogos para computador. O seu desenvolvimento mais avançado ganha cada vez mais importância e normalmente afecta o sucesso ou o insucesso de um jogo. O progresso neste campo aborda principalmente uma procura inteligente de rotas ou caminhos de navegação, a sua usabilidade e aplicação a qualquer jogo (na construção de "motores" como base de sustentação de vários jogos) na qual a componente lógica computacional tenha necessidade de movimentar objectos pelo ambiente virtual, por exemplo pessoas, veículos ou unidades de combate de uma localização inicial para um destino a designar. Na indústria dos jogos existem vários problemas e desenvolvimentos emergentes associados à CG e IA, o *path planning/finding* é provavelmente o mais popular entre estes, e potencialmente o mais frustrante [95].

Simulação em projectos arquitectónicos e industriais: O termo "*Walkthrough*" (caminhar através de…) é usualmente empregue na indústria de software e animação virtual, de modo a descrever a "inspecção" de algoritmos e código fonte no processo de simulação e avaliação de percursos utilizados por humanos virtuais, determinados por condições de inputs e escolhas realizadas ao longo de um trajecto. O propósito será o de assegurar a capacidade e competência de movimentação autónoma ou não, de personagens virtuais individuais ou em grupo. O termo "*architectural walkthrough*" utiliza software para promover visitas virtuais a estruturas arquitectónicas de modo a simular navegação numa construção real. Também por vezes referido como "*Flythrough*" é apontado como um importante contributo e uma ferramenta de grande valia na demonstração de como um determinado número de pedestres se poderá comportar numa determinada e futura estrutura real, permitindo a possibilidade de uma avaliação através de várias vistas e ângulos em edifícios ou estruturas industriais. As primeiras implementações comerciais apareceram durante os anos oitenta, época em que os avanços na computação suportaram software com processamento capaz de simular objectos virtuais em espaços também virtuais. Hoje encontramos imensas referências e exemplos que aplicam estas técnicas, tais como [96],[89],[97],[46], entre outros.

Uma aproximação inteligente antes de seguir alguma estratégia de modo a poder ultrapassar os obstáculos quando encontrados, será o planeamento e definição

anteriormente de toda ou parte da rota ou caminho de acordo com determinados algoritmos. Em alguns casos, o objectivo será encontrar o percurso em função do custo (mais rápido, de menor distância ou menor esforço, etc.) entre várias alternativas, enquanto noutros o propósito poderá simplesmente encontrar e fazer uso da primeira solução (percurso) válida. De seguida, serão descritos alguns modelos ou algoritmos mais convencionais utilizados que lidam com o problema de *path planning/finding*, integrando conhecimentos da computação gráfica e de inteligência artificial.

3.1.2.2 | Mapas de Percursos Pré Definidos

Mapas pré definidos poderão servir como estruturas de grafos usados em *path finding*. Assim, o criador do mundo virtual pode anteriormente definir uma rede de localizações interligadas e navegáveis gerando sub-caminhos, que permitam uma mais rápida pesquisa por parte do algoritmo. Sendo fácil de implementar enquanto procedimento, algumas lacunas podem ser deduzidas: o mapa terá de ser criado manualmente, e nem todas as localizações poderão ser alcançáveis e navegadas (unicamente aquela parte do mapa pré computada). O mais importante e frustrante de tudo é que o mapa terá de ser conhecido. Enquanto que, este tipo de instruções de navegação pré definidas apresentam um máximo de controlo para o programador, não são flexíveis e aplicáveis em tempo real, e não escaláveis para grandes ambientes virtuais densamente povoados recreando situações mais complexas.

3.1.2.3 | Procura Directa nos Polígonos

Talvez o método mais directo de atacar o problema de *path finding* é considerar a estrutura geométrica do ambiente como uma estrutura gráfica e realizar a procura de caminhos directamente na representação poligonal da cena, este método é conhecido como "*NavMeshs*". Os trabalhos mais visíveis nesta área são baseados na procura com base em triângulos, ou decompondo os polígonos presentes na cena num conjunto de polígonos convexos (a convexidade assegura que cada ponto dentro de um polígono pode ser alcançado ou alcançar qualquer outro ponto do mesmo polígonos através de uma linha recta, sem tocar ou cruzar a borda desse mesmo polígono) e realizar a procura de cada um destes com os seus adjacentes. Trabalhos como [98] abordam este assunto. O problema é que o tamanho dos polígonos pode variar. Isto significa que o

algoritmo de procura não poderá usar os polígonos como nodos de um grafo como acontece numa aproximação utilizando grelhas de células. Por isso, a maioria das aplicações produzem um pré processamento do ambiente de modo a poder representa-lo de uma forma mais simples, conveniente e utilizável.

3.1.2.4 | Grelhas de Células

A tecnologia baseada na decomposição celular reestrutura espaços de navegação (o conjunto de todas as posições acessíveis no ambiente) em regiões adjacentes cuja vizinhança é conhecida e explorada de modo a ser pesquisável numa sequência ordenada, facultando a possibilidade de descobrir espaços navegáveis através dela. A partir desta figuração do ambiente, é então construída uma representação geométrica do espaço livre de navegação, usado posteriormente na construção de um grafo hierárquico de interligações, cujo nodos são as células e os limites destas traduzem a adjacência. Geralmente podem ser distinguidos dois métodos: a decomposição exacta, e uma decomposição aproximada das células. A primeira consiste na definição e composição das células como uma representação exacta do espaço navegável [99] (são exemplos: *constrained delaunay tringulation, convex poygons and trapeziodal,* etc). O segundo consiste no uso de formas já predefinidas de células, acopladas e incluídas ao espaço livre para navegação (põe exemplo: *uniform grids, quadtress,etc*) [84],[100].

O *path finding* é um problema comummente ligado a ideia de pesquisa em grafos. O mundo é decomposto, convertido e abstraído num modelo em forma de grafo, e efectivados algoritmos e técnicas de procura conhecidos, como por exemplo o A-estrela (A-star ou simplesmente A*) ou outra sua variante.

3.1.2.5 | Algoritmo de Pesquisa A*

Este algoritmo foi descrito pela primeira vez em 1968 por Peter Hart, Nils Nilsson, e Bertram Raphael [101]. Nesta publicação, foi inicialmente chamado de algoritmo A, passando mais tarde a ser conhecido por A*. Usando este algoritmo com uma heurística apropriada poderá atingir-se um percurso próximo do óptimo tendo em conta a eficácia.
Este resulta da combinação do algoritmo de *Dijkstra* e do algoritmo *Best-first*:

- *Algoritmo de "Dijkstra"*: Edsger W. Dijkstra desenvolveu este algoritmo clássico de modo a poder percorrer grafos com ligações de pesos diferentes. A cada passo, a procura é focada nos nodos vizinhos não processados de forma a avaliar as respectivas distâncias desde o início.

- *Algoritmo "Best-first"*: este algoritmo usa a função heurística F(n)=h(n) de procura ao nodo de destino. Esta procura tenta expandir o nodo que é mais próximo ao objectivo, acreditando numa condução rápida ao alvo.

O A* é um algoritmo que apresenta soluções para um problema mais abrangente, que um simples algoritmo não resolveria. Utiliza o tipo de busca por amplitude onde o objectivo é verificar primeiro os nós mais próximos do nó inicial de um grafo, até chegar ao destino da busca, e também busca por profundidade que ao contrário da anterior consiste em procurar prioritariamente os nós sucessores mais distantes do nó inicial de um grafo, até encontrar o destino. A combinação dos dois, agregados a métodos de heurísticas, resulta num algoritmo de elevado desempenho para uma margem de erro mínima.

No algoritmo A* é também adicionada a função heurística de proximidade à solução. Como resultado é alcançada a função F(n), composta pela soma de duas outras funções g(n) e h(n). É apresentado a seguir a representação clássica do algoritmo A*:

F(n)=g(n) + h(n)

g(n) - A função g é determinística, indicando para um dado nó o custo do percurso do nó inicial até esse nó, ou seja, é uma medida do custo de chegar de um nó inicial até ao nó corrente. Note-se que g não estima nada, sendo o somatório dos custos do percurso do nó inicial até ao nó corrente.

h(n) - A função h será uma estimativa do custo do percurso desde o nó corrente até um nó terminal (ou estado final). Esta função h será uma forma de embeber conhecimento sobre o domínio do problema na pesquisa de soluções.

F(n) - A função F representa o somatório de g(n) e h(n), será assim o caminho estimado mais curto e não será descrito até o algoritmo não terminar. É uma estimativa da chegada do nó inicial até um nó final de acordo com um percurso que gerou o nó corrente.

Uma das componentes mais difíceis de resolver no algoritmo A* é a criação de uma boa função de heurística para determinar o h(n). A função heurística difere de um algoritmo devido a esta ser mais uma estimativa e não necessariamente uma solução correcta. A performance resultante do algoritmo A* é muito dependente da qualidade da função heurística. Pinter e Stout [95, 102] descrevem possíveis optimizações e soluções para os principais problemas de eficiência que o A* é afectado.

Nos dias de hoje, *path finding* hierárquico incorpora múltiplos grafos e múltiplas decomposições de procura de espaços com granularidade diferente como um meio de reduzir o tempo de procura, mas por vezes com algumas perdas de fidedignidade [103]. De modo a poder melhorar a prestação destes comportamentos podemos incorporar e optimizar métodos similares e complementares aos da utilização de grelhas de células, tais como, mapas de percursos e campo potenciais [104].

3.1.2.6 | Mapas de Percursos Gerados Automaticamente

O processo de construção de mapas de percursos resulta na computação de uma rede estandardizada de percursos ou caminhos (linhas rectas ou curvas) através dos espaços livres e navegáveis. Um exemplo de destaque deste sistema é o estudo apresentado em [105] onde se estabelece a conexão entre os vértices do ambiente que se "vêem" uns aos outros. O principal exemplo desta técnica de computação são os diagramas de *Voronoi* dentro dos espaços livres, que permitem a geração de vértices e consequentemente de percursos [106, 107].

A utilização de algoritmos para a construção automática de exploração de caminhos com base em grafos hierárquicos é um desenvolvimento bastante recorrente na navegação de terrenos desconhecidos, usando um período de adaptação do grafo hierárquico ao ambiente de navegação (pré-processamento), este método pode ser usado nas mais diversas situações de navegação a partir de um mundo dividido em secções.

Desde uma descrição geométrica do ambiente, Lamarche [99] propõe um método de extracção automática da topologia do ambiente recorrendo a triangulações *Delaunay*, cálculo de distâncias mais curtas entre os cantos e paredes dos edifícios, e a

conjugação de ambos. Este tipo de subdivisão de espaços é guardado de forma catalogada num grafo hierárquico, o qual irá responder posteriormente a consultas de navegação (Figura 19). Este método no entanto é específico para ambientes bem delineados e limitado para terrenos planos (2D).

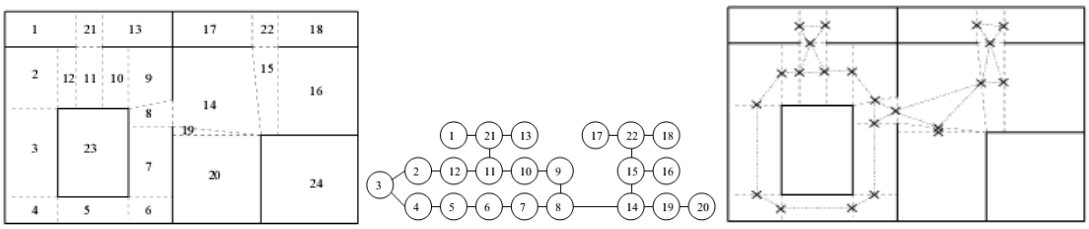


Figura 19 – Grafo de navegação de subdivisões convexas com base em triangulação Delaunay

Imagem original em [99]

Outro exemplo interessante baseado neste tipo de implementação pode ser observado em [108], o qual, serve de guia virtual de visitas a edifícios como museus (com obras de arte como objectivos direccionais, localizadas nas várias secções do edifício). Assim, neste trabalho, dado um modelo arbitrário 2D e uma posição de início, o algoritmo traça um caminho livre de colisões até aos locais pretendidos representados por uma sequência de nodos. Em cada um deles há um ou mais pontos de interesse, valores de tempo de permanência, e posições da câmara associados a estes (Figura 20). O processo pode ser dividido em 3 fase gerais:

1 – Detecção da estrutura geral, e organização tipo (cell + portal);
2 – Determinação de quais as células mais importantes a visitar, com base em medidas de avaliação sobre a quantidade e importância de informação catalogada nessas células;
3 – Construção do caminho que percorre as células mais relevantes.

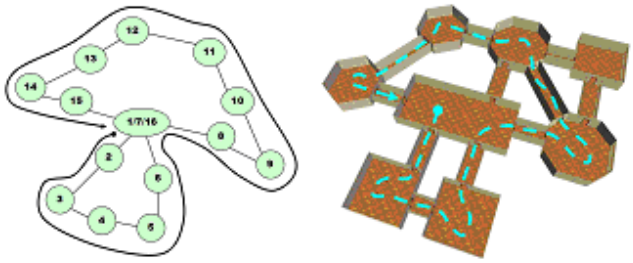


Figura 20 – Organização do grafo correspondente aos pontos de interesse a visitar.

Imagem original em [108]

Um outro trabalho de referência nesta área de navegação, num mundo desconhecido à partida desde uma análise do ambiente e formulação do grafo representativo da sua navegabilidade é apresentado em [89], onde se permite a navegação em ambientes complexos 3D com base em algoritmos para o planeamento de caminhos e decomposição de espaços. Existe a pré-computação destes, obtendo um roadmap de navegação livre de colisões, o qual é utilizado em tempo real para a animação do avatar ao longo do caminho traçado (Figura 21). Esta aproximação combina métodos determinísticos com estratégias aleatórias de modo a conseguir retirar os benefícios de ambos. São usadas estratégias de amostragem combinada, construindo grafos de visibilidade; dividindo recursivamente o espaço em quadrantes (uma estratégia determinística) com escolha de uma amostra aleatória dentro de cada quadrante de modo a testar a localização no espaço.

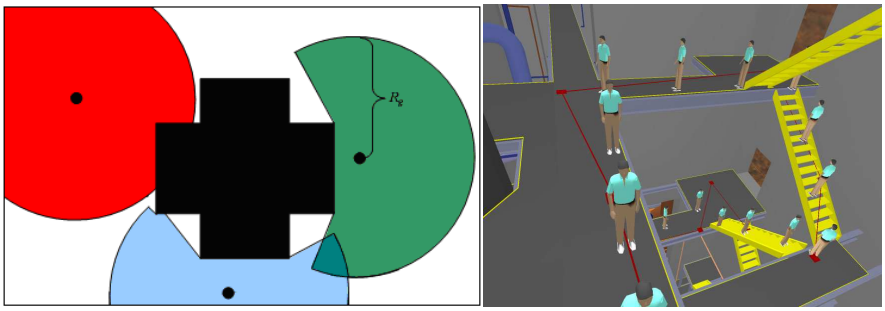


Figura 21 – Divisão recursiva do espaço em quadrantes utilizados na navegação.
Imagem original em [89]

Como já referenciado anteriormente, investigações recentes [3, 86] provenientes do laboratório VRlab, e encabeçadas por Pettré, deram origem a uma plataforma de navegação em tempo real com base no planeamento de rotas de navegação a partir de um mapa tridimensional baseado num grafo, deduzido de forma automática, a partir de cilindros interligados representando áreas navegáveis em ambientes tridimensionais (Figura 22).

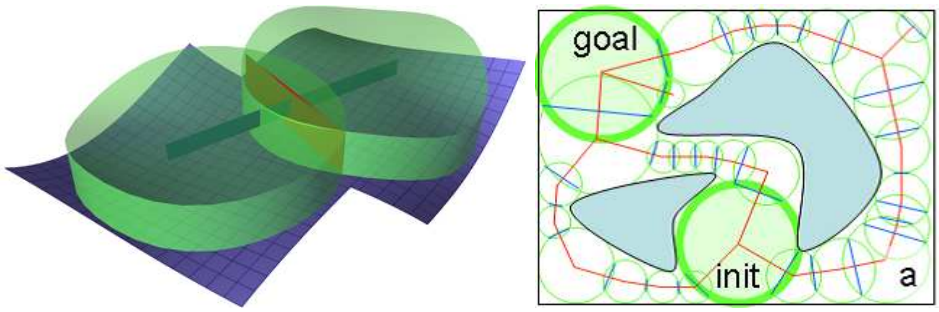


Figura 22 – Divisão do espaço em zonas cilíndricas interconectadas formando o grafo de navegação tridimensional.

Imagem original em [86]

Neste trabalho, é deduzida a localização actual dos avatares e identificado o cilindro no qual o avatar se encontra. Para movimentações no ambiente virtual, desde o ponto inicial até ao ponto final (também este enquadrado num cilindro respectivo) é utilizando o já conhecido algoritmo de pesquisa Dijkstra, onde é calculada a rota ou percurso mais curto, bem como alternativas a este, através do grafo de navegação, composto em pré-processamento, onde as zonas cilíndricas representam os nodos e as intersecções entre estas as respectivas conexões (Figura 22). O percurso é dividido entre diferentes sub-objectivos espaciais (também chamados de *waypoints*), localizados nas intersecções dos cilindros e nos respectivos eixos. Como possível plataforma de navegação livre e realmente tridimensional, possível geradora de efeitos secundários de actuação incorporando pesos e custos adaptados à deslocação neste ambiente, o grafo utilizado não incorpora a informação da diferenciação entre as localizações dos cilindros (nodos) a várias alturas, não promovendo assim uma possível navegação incorporando o tal peso multi-nível associado a um possível custo da navegação de personagens que o habitem. Este modelo comporta basicamente a gestão de cilindros ou nodos de navegação tendo em atenção as personagens aí presentes de modo a procurar rotas alternativas no caso de congestionamentos de alguns destes locais de habitual passagem. É difícil também, pensar a evolução deste sistema para a detecção de colisões entre as próprias personagens, limitando a possibilidade de gerar rotas de navegação mais dinâmicas com base neste possível constrangimento, já que a planificação dos trajectos é realizada sob uma perspectiva global e pouco adaptativa às alterações em tempo real da densidade populacional em determinados espaços mais exíguos, onde se exigiria uma planificação mais individualizada.

Mais recentemente Avneesh Sud et all [109], apresentaram um mecanismo capaz de planear e apresentar em tempo real, deslocações dedicadas a múltiplos agentes em

diversas cenas 2D. Esta representação, não incorporando a versatilidade de navegação realmente 3D, conseguida pela plataforma de Pettré et all, consegue no entanto, dedicar rotas de navegação independentes para cada avatar, mesmo para personagens com a mesma origem e destino, isto, em ambientes estáticos ou dinâmicos, onde os próprios avatares e outros elementos em movimento (ex: automóveis) vão alterando em tempo real as rotas inicialmente traçadas entre dois pontos no espaço discretizado (Figura 23).

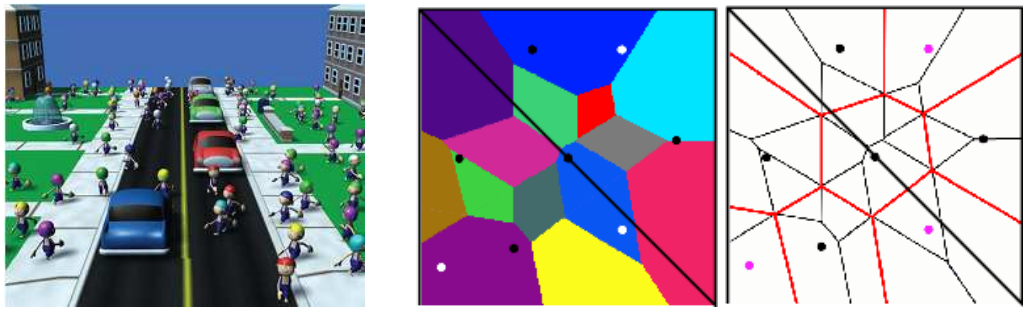


Figura 23 – Divisão do espaço em Diagramas de Voronoi dinâmicos para planeamento de percursos com cenas em movimento.

Imagem original em [109]

A construção da estrutura de navegação (grafo), aqui chamada de *Multi-Agent Navigation Graph* (MaNG), tem por base a subdivisão do espaço em diagramas de Voronoi, onde a especificação das rotas de navegação se identifica após duas pesquisas (uma local e individual e outra com uma visão mais alargada) sobre diferentes ordenações e estruturas dos diagramas de Voronoi produzidos. A computação desta estrutura é realizada em fase de pré-processamento utilizando o hardware gráfico, e técnicas de *culling* de forma a acelerar os tempos de computação. Como limitações, e pela navegação estar reduzida a espaços bidimensionais, esta navegação não contempla personagens com os vários graus de liberdade proporcionados pelos ambientes 3D aquando da escolha de caminhos alternativos à colisão. A implementação do algoritmo A* neste trabalho, não sofreu alterações no seu conceito original, o que pode provocar alguns constrangimentos endereçados à utilização deste algoritmo, como por exemplo, alguns tempos de congelamento da animação, bem como a não convergência da animação para períodos de tempo regulares durante um mesmo trajecto de várias personagens, já que a geração de um novo caminho é computado a cada frame da animação, podendo gerar alguns percursos estranhos. Outra limitação, prende-se com a impossibilidade de utilização de um grande número de personagens virtuais, já que a construção do diagrama Voronoi (para pesquisas locais e endereçadas individualmente a cada personagem)

em conjunto com a pesquisa A* computada a cada frame (como é detalhado no trabalho) resulta impossível de ser computada quando se envolve um grande número de personagens.

3.1.2.7 | Campos Potenciais

A utilização de métodos recorrendo a campos ou localizações capazes de potenciar comportamentos específicos normalmente serve-se também de ambientes discretizados e convertidos numa grelha regular de avaliação. O campo potencial é assim "ligado" a cada célula que corresponderá a um estado extra de comportamento com acções repulsivas ou atractivas geradas pelo objectivo de alcançar o destino; [110, 111] são exemplos de simulações de comportamentos de pedestres utilizando campos potenciais.

Um exemplo recente deste tipo de utilização deste método pode ser visto em [81], onde Treuille, Cooper e Popovic, implementam um mecanismo dinâmico de campos potenciais, com actualização da posição para cada agente através da passagem pelas células, onde uma escala dada por uma função de controlo potencial, estabelece o gradiente que proporcionará estados de comportamento e direcção distintos ao pedestres aí localizados. Estes campos potenciais são computados para pequenos grupos de agentes em movimento com objectivos comuns. No entanto, devido ao uso de uma função geradora de campos potenciais a actuar simultaneamente sobre a abrangência de vários agentes, poderá provocar o aumento de movimentações em direcção a locais exíguos (e menos apropriados) próximos desses campos, favorecendo agrupamentos em torno do mesmos objectivos, causando filas de espera e simulações por vezes pouco realistas.

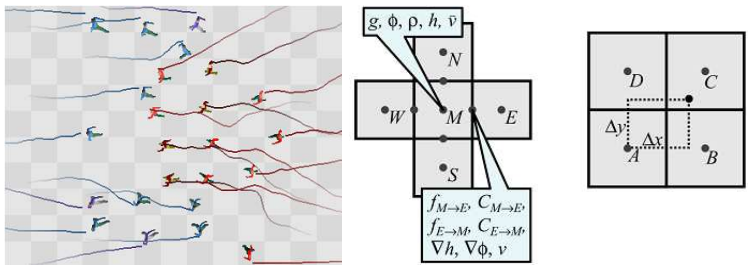


Figura 24 – Interacção entre dois grupos diferenciados de agentes com imposição de campos descriminadores às células de navegação.

Imagem original em [81]

3.2 | IA: Simulação Comportamental Individual, em Grupo, ou em Multidão

Uma simulação social é baseada num conjunto de agentes autónomos inseridos num mundo virtual que, por sua vez, condicionam as próprias personagens virtuais bem como outros objectos presentes no mundo. Estas personagens poderão apresentar-se de forma individual, em grupo, ou em multidão, com comportamentos subjacentes derivados da percepção das condições do mundo virtual, provocando assim diferentes formas de actuação nesse ambiente. Os esforços para atingir um elevado grau de realismo na simulação, concentram-se tanto nos aspectos visuais (aparência, movimento) como em comportamentos mais próximos ao de um ser humano real. Em inteligência artificial (IA), mais especificamente, em sistemas multi-agentes, sempre houve tendência em modelar o processo cognitivo inerente aos seres humanos, indo desde os modelos reactivos até cognitivos. Os modelos reactivos utilizam agentes que não possuem um modelo de raciocínio e agem com base em respostas directas a estímulos externos. Por outro lado, os agentes cognitivos, ou deliberativos, são os que possuem um modelo simbólico de raciocínio e um plano a ser realizado ou negociado com outros agentes para alcançar determinados objectivos.

Na vasta e diferenciada literatura específica da área envolvendo simulação de agentes representando comportamento individual ou em grupo, inseridos em ambientes virtuais específicos, podemos demarcar simulações de modelos reactivos e cognitivos recorrendo a: regras bem definidas "*Rule-Based*"; métodos globais na simulação de pedestres; métodos comportamentais influenciados por questões físicas e sociais; ou ainda, associados a informações embebidas no ambiente. De seguida são mencionados alguns dos mais importantes trabalhos nestas áreas de actuação.

3.2.1 | Modelos de Comportamentos – Baseado em Regras

O modelo proposto por Craig Reynolds em 1987 [1] é provavelmente o mais famoso e o mais bem sucedido modelo de movimento comportamental em grupo, concebido especialmente para simular aglomerações de animais. Pode ser usado para simular comportamentos de concentração de pássaros, cavalos, peixes, etc. Reynolds chamou-lhes de agentes "boids" (bird-oid). Este modelo é baseado numa formulação

muito simples de regras (*Rule-Based*) que cada agente no grupo deve respeitar, resultando num produto harmonioso e surpreendente. Estas regras podem ser de três tipos:

- Separação: Utilizada para a detecção de colisões entre elementos do grupo.

- Alinhamento: Usada no controlo da velocidade em relação aos elementos vizinhos no grupo.

- Coesão: Capacidade dos agentes se manterem perto de outros agentes do mesmo grupo.

Este modelo trabalhado por Reynolds encarou a premissa que o comportamento de grupo é o resultado directo da interacção entre o comportamento individual dos membros do grupo. A principal característica deste modelo é a localização: a ordem global é uma propriedade emergente de todo o sistema. A ideia não é redutora e subjacente ao simples acompanhamento de um líder, cada "boid" é identificável pelos seus elementos agrupados formando um todo, com localização específica das individualidades. As três regras expressas anteriormente caracterizam a essência do comportamento em rebanho (*flocking*), e podem facilmente ser estendidas para lidar com elementos estáticos (os boids devem ter capacidade de separar-se, contornar objectos e/ou juntar-se de novo). Quanto a evitar colisões, um boid somente considera a posição dos seus elementos próximos, determinando velocidades nas individualidades e confiando na dos seus vizinhos. Com isto, o método de colisão e de navegação livre aparece como um efeito colateral. Como Reynold refere em [1]:

> «… *if the boid does a good job of matching velocity with its neighbours, it is unlikely that it will collide with any of them any time soon. […] Static collision avoidance serves to establish the minimum required separation distance; velocity matching tends to maintain it* …»

Mais tarde, Tu e Terzopoulos [112] desenvolveram pesquisas na área da vida artificial com o objectivo de povoar um mundo marinho virtual com peixes que caçam, fogem, reproduzem-se e vagueiam. Nesse trabalho, foram modelados fisicamente as personagens a simular, o ambiente, o estilo de locomoção, a percepção (visão sintética) e alguns comportamentos de mais alto nível. Deverá ser realçado que os objectos constituintes deste mundo virtual subaquático são mais complexos que os

boids criados por Reynolds. A modelação das espécies de peixes é já considerada aqui como massas dinâmicas, caracterizando corpos com músculos capazes de se contrair criando movimentos muito naturais, como por exemplo o movimento da cauda inerente à movimentação de cada espécie de peixe. Mas as diferenças não se ficaram pelas formas, pela geometria, ou texturas utilizadas, mas também no comportamento associado a regras dependentes do tipo de personagem (predadores, presas ou pacifistas), com inclusão de variáveis capazes de alterar os estados internos das personagens, compreendendo por exemplo: a fome, a libido, o medo, e alguns parâmetros da geração de habitats dependentes de águas mais quentes ou mais frias. As intenções dos agentes são aqui confinadas a um algoritmo que usa todos estes aspectos para produzir um *output* relativamente credível a nível comportamental.

Outro exemplo que podemos encontrar na simulação comportamental entre grupos de seres vivos, é o caso da simulação do trabalho comunitário protagonizado pelas formigas. No mundo real as formigas são capazes de encontrar o menor caminho entre a fonte de comida e o formigueiro. Elas fazem isso sem se comunicarem directamente umas com as outras, mas através de uma pequena modificação no ambiente, ou seja, deixando o hormônio feromônio no caminho. Baseado nesta ideia, Dorigo [113] desenvolveu o *Ant Colony System* (ACS). O ACS é um sistema de propósito geral, que pode ser usado para resolver problemas de análise combinatória e comportamental associado também a agentes virtuais humanos.

Caminhando em direcção ao uso de modelos mais realistas em simulações de entidades caracterizando humanos com base em regras bem explícitas, Brogan e Hodgins [114] descreveram um algoritmo de controlo de movimentos para entidades com alguma "dinâmica significativa" aplicada a grupos de personagens. Por dinâmica significativa, os autores consideram aplicações onde o foco de atenção é dirigido para sistemas de simulação cuja dinâmica fosse suficientemente complexa que gerasse um grande impacto no movimento das entidades simuladas. Por exemplo, um dos casos estudados e apresentados pelos autores simula a dinâmica na movimentação de robots com uma só perna (Figura 25). Este tipo de personagens foi significativo para os testes dinâmicos já que estes não poderiam variar intencionalmente a sua velocidade durante os constantes e necessários saltos para a sua movimentação, enquanto não estivessem em contacto com o chão. Para outros exemplos de caracterização com movimentações dinâmicas, os autores escolheram simular também grupos de ciclistas em movimento (Figura 25). Este tipo de algoritmo, à imagem dos anteriores é basicamente dividido em duas partes: percepção e

posicionamento. A percepção actua permitindo simular entidades com sentidos de posicionamento e velocidade relativa em relação a entidades próximas. O posicionamento funciona através do retorno da posição desejada e natural de uma dada entidade, dada a visibilidade de outras entidades, bem como outros obstáculos encontrados no mundo virtual a simular.

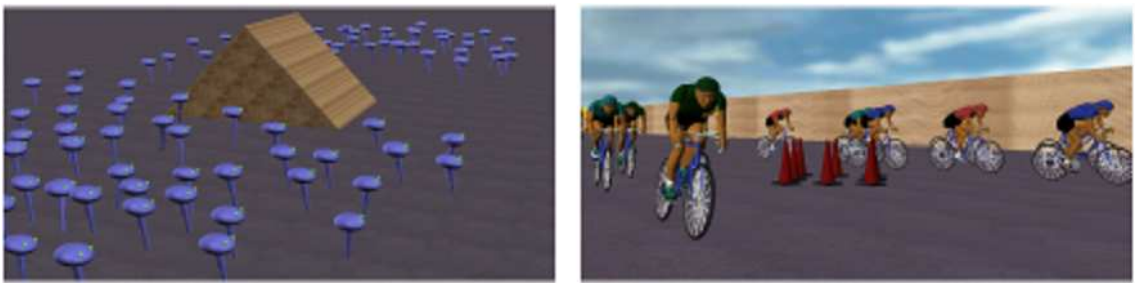


Figura 25 – Grupos de robôs e ciclistas em movimento dinâmico.
Imagem original em [114]

Este tipo de aproximações tiveram um efeito de grande êxito, mas tendem a não serem satisfatórias em pedestres humanos. Implicitamente, o estado dos boids torna-os cooperativos e com partilha geral do mesmo objectivo e preocupações associadas. Em multidões humanas os objectivos poderão ser muito mais variados, onde por exemplo todos podem tentam evitar colisões, com um resultado final bastante diferente de um cardume de peixes. Isto acontece devido a provavelmente poucos pedestres partilharem o mesmo objectivo, e por todos tenderem a agir de uma forma mais individualista, não reforçando assim as regras de alinhamento e coesão. Esta diferença é realçada e parece mais evidente quando observadas densidades populacionais mais baixas, onde por exemplo cada individualidade é livre de escolher a sua velocidade.

De modo a poder reproduzir alguns destes comportamentos de forma mais intuitiva, existem actualmente sistemas comerciais tais como: Softimage/Behavior, CrowdIT, Massive [115] analisados posteriormente no capítulo 6, baseados também em sistemas "*Rule-Based*". No entanto estes sistemas não são escaláveis, na perspectiva do autor. Simular comportamentos de humanos virtuais em tempo real e em ambientes desconhecidos é diferente à preparação necessária para animar produções cinematográficas de cenas *off-line* específicas em ambientes específicos (mesmo podendo conter certos graus de autonomia).

3.2.2 Modelos Globais na Simulação de Pedestres Humanos

Os métodos globais consideram a multidão como um todo, e caracterizam o seu estado com propriedades como a fluidez densidade, velocidade, etc. O ambiente envolvente é modelado graficamente de forma separada, com cada esquina ou rua a ser considerada um local de transito dependente do tipo de multidão de agentes nele inserido. Para o caso da simulação de humanos, a concentração de pedestres virtuais é por norma propagada ao longo de todo o ambiente gráfico, onde são obtidos tempos e percursos de navegação.

Este tipo de descrições globais são também usadas com sucesso em estimativas em larga escala, tais como em estudos dos efeitos na modificação do desenho de estruturas navegáveis (adição de novas entradas, ou saídas, etc) ou por exemplo no estimar dos tempos de evacuação de um determinado edifício. No entanto, não tomam em consideração alguns elementos de comportamento, e não podem ser usadas com precisão nas mais diversas escalas, de modo a conseguir estudar e prever como os pedestres reais agiriam verdadeiramente. São possíveis diferentes variantes deste método, especialmente nos casos de localização de congestionamentos e alocação de rotas. Um exemplo deste tipo de método é apresentado por Predtechensky e Milinsky [116](também referenciado em [117]).

Em trabalhos mais recentes, como o de Celine Loscos, et al, em 2003 [110] (Figura 26), são implementadas técnicas que permitem a simulação até 10.000 agentes pedestres em tempo-real, através de uma visão macroscópica de comportamentos numa simulação de simulação global.

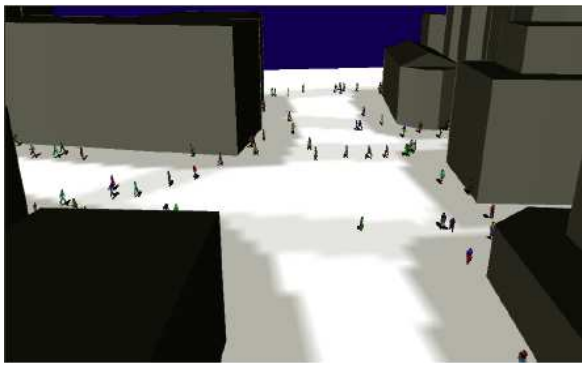


Figura 26 – Exemplo de simulação macroscópica com 7000 pedestres.

Imagem original em [110]

Por norma as colisões são diferenciadas e detectadas com base em mapas 2D discretizados (distinguindo zonas caminháveis ou não), e identificação de trajectórias de uso mais comum. São também implementados mecanismos de convergência de objectivos em agentes individuais ou de grupo. Houve cuidados com a adaptação à densidade populacional, armazenando em cada célula da grelha de navegação, as intenções de destino dos agentes que por lá passam, prevendo assim situações de densidade muito elevada. Foram ainda implementados comportamentos de mais baixo nível, como: localizar objectivos, seguir líder, detectar colisões e adaptação geral dos pedestres ao ambiente.

3.2.3 | Modelos Influenciados por Questões Físicas e Sociais

3.2.3.1 | Modelo de Forças Sociais

Muitas pessoas acreditam que o comportamento humano é caótico ou pelo menos irregular e não previsível [118]. Isso é provavelmente verdade para comportamentos encontrados em situações complexas, tal como numa situação de pânico. Um modelo simulando estas condições, foi apresentado originalmente por Helbing e Molnar [118] com o propósito de gerar algum método de auto-organização, capaz de ser observado no comportamento de multidões. O foco de atenção deste modelo, foi direccionado especialmente para as seguintes conclusões da vida real:

- Podem ser observadas rotas de direcção uniforme, quando os pedestres se movem em direcções opostas, e se a densidade exceder um determinado valor crítico.

- Existem mudanças oscilatórias na direcção quando se caminha por passagens estreitas. As frequências destas mudanças são incrementadas com a largura e diminuem com o comprimento das passagens.

Estas assumpções são relevantes para se poder demonstrar em simulação o caminhar como um acto natural e rotineiro, onde as reacções dos pedestres poderão ser mais ou menos automatizadas e os seus comportamentos previsíveis. Assim, foi composto em [118] um modelo de variação de velocidade embutida aos pedestres, tendo em conta

a força social que representa a influência exercida pelo ambiente bem como por outros pedestres, associado às flutuações de variação de comportamentos (acidentais ou deliberadas).

De acordo com Helbing, este "simples" modelo poderá oferecer simulações bastante realistas, onde poderão ser visíveis por exemplo, os seguintes padrões:

- Numa densidade crítica, há a formação de rotas com variação dinâmica, sem a intervenção de preferências de direcção por parte dos pedestres. Este padrão aparece para ser o regulador no incremento da eficiência global do movimento.

- Ao caminhar por passagens estreitas, a direcção contém mudanças aleatórias devido à pressão exercida por pedestres em espera. Quando se excede um valor crítico o grupo de pessoas em espera tenta forçar a passagem pelo caminho, e as mudanças de direcção acontecem.

- Quando um objecto é o centro da rota de muito pedestres acontecem intercepções de movimentos, gerando padrões instáveis, como o rodar sobre o próprio agente, mas, apesar disso, contribuindo para um movimento global mais eficiente.

Este modelo descreve com sucesso um fenómeno interessante e emergente em situações críticas, mas é muito simples na hora de proporcionar uma introspecção de comportamento particular a cada um dos pedestres.

Mais tarde em 2000, em Helbing et al [119] foi proposto um novo modelo de forças físicas e sócio psicológicas adaptado, com a intenção de representar o comportamento de multidões humanas em situações de pânico. Neste novo trabalho os agentes são também designados como partículas, em virtude da semelhança com simulações da dinâmica molecular. Este modelo gera resultados também observáveis em situações reais como por exemplo, a formação de semi-discos em saídas congestionadas (Figura 27).

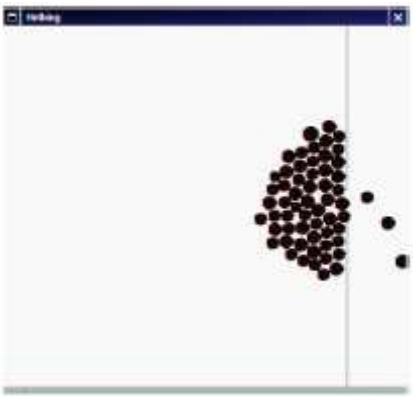


Figura 27 – Formação de um semi-disco de agentes na saída de uma sala.

Imagem original em [119]

Como característica emergente dependente de certos parâmetros individuais, alguns comportamentos de grupo podem aparecer, conforme demonstrado, em situações em que pessoas altruístas tendem a resgatar outras pessoas mais dependentes, arriscando as suas próprias vidas. Investigações mais recentes [120, 121] apresentam a generalização do modelo de Helbing mas com o objectivo de incluir mais individualidade aos agentes. Outra principal motivação destes trabalhos é também incorporar sucessivamente aspectos mais realistas ao espaço físico da simulação. Nestes trabalhos, foram simuladas diferentes configurações de parâmetros e encontradas similaridades mais convincentes em relação ao que é intuitivamente aceite.

Outro exemplo interessante nesta linha de investigação, é o sistema criado em [121] de nome PetroSim. É um modelo de evacuação em tempo real utilizado em testes pela companhia petrolífera Brasileira – Petrobrás – nas instalações desta empresa na cidade de São José, estado do Rio Grande do Norte – Brasil, que se encontram rodeadas por áreas urbanas densamente habitadas. Este sistema é utilizado de modo a prever planos de evacuação dos seus engenheiros, bem como avaliar locais e edifícios de construção nas suas instalações, potencialmente perigosos neste tipo de áreas. Aqui, o conhecimento dos agentes é composto por informações previamente computadas e também obtidas em tempo real desde o ambiente, são exemplos: informações de localização de pontos importantes, possíveis rotas entre estes, posicionamento e status pendentes de comportamentos sociais, e a percepção de regiões afectadas por acidentes. O status dos agentes contendo informações predefinidas poderão após a simulação ter inicio, variar como consequência do acontecimento de vários eventos. Alguns itens que constituem o status do agente ou

grupos destes são: diferentes *profiles* associados a individualidades ou grupos (normal, lideres ou dependentes), consequências sofridas num dado acidente num agente ou grupo, e grau ou situação de perigo durante um determinado período de movimentação dos agentes.

3.2.3.2 | Sistema de Partículas

O sistema de partículas é normalmente usado na animação por computador para a modelação e visualização de compostos físicos (nuvens, água, gás, fogo, etc). Consistem num conjunto de primitivas elementares (partículas) com atributos físicos (que obedecem a leis da física). Bouvier e Cohen [122] usaram uma proposta generalista de um sistema de partículas 3D na simulação de fluxos humanos num ambiente 2,5D. As partículas evoluem de acordos com mecanismos Newtonianos, sob a influência de várias forças podendo ser agrupadas em classes, onde é possível especificar o correspondente sexo, idade, força e estado físico, subjacentes ao comportamento humano. Este sistema assenta nas seguintes características:

- Acções de reflexo: referem a habilidade para evitar pessoas ou obstáculos instantes antes da colisão ocorrer. É definida uma área em volta da partícula, se a colisão estiver para acontecer nessa área num mesmo tempo de movimento entre vários agentes, então o valor de direcção e velocidade associado a cada uma destas entidades é modificado.

- As decisões (comportamentos de longo termo): são modeladas usando estados. Um estado define uma reacção da partícula a uma determinada percepção do entorno. Esta reacção poderá ser uma deslocação ou alteração de um outro tipo de estado. Numa simulação, cada estado é representado por um conjunto de cargas à espera da reacção das partículas a campos ou localizações de reacção específicas.

- Situações críticas: quando uma decisão de uma individualidade não pode mais influenciar o seu próprio movimento. Ocorre por exemplo em situações de grande densidade. Como as pessoas normalmente tendem a evitar situações de congestionamento, um campo repulsivo pode ser definido para as direccionar para locais de menor densidade populacional.

O modelo de Bouvier e Cohen mostrou-se suficientemente robusto para permitir simulações em larga escala, mas por outro lado os comportamentos individuais mantiveram-se relativamente simples. As simulações envolveram mais de 45.000 representações virtuais representativas de humanos, usadas especialmente em testes de planeamento de evacuações de emergência.

3.2.4 | Trabalhos Adaptados da Robótica para a Deslocação de Pedestres Virtuais

O problema de navegação de robôs móveis tem sido estudado ao longo de vários anos, com o objectivo de se construir um robô com elevado grau de autonomia. O aumento da autonomia de um robô móvel fica fortemente condicionado pela sua capacidade de perceber o ambiente de navegação, e está principalmente relacionado com a habilidade de aquisição de informações, com a automatização de tarefas, a construção de mapas de ambiente e o correcto planeamento de rotas, para a sua deslocação. Uma das tarefas que um robô deverá realizar autonomamente é a navegação dentro de um ambiente, sem colidir com os obstáculos. Os sistemas de visão são amplamente utilizados em tarefas de robôs "autónomos" devido à grande quantidade de informação contida numa imagem, capazes de guiar este pelo ambiente. Os sensores dos pedestres virtuais poderão ser substituídos pelo mapeamento antecipado do ambiente para a implementação de estratégias de navegações mais autónomas, e planeadas à priori. Com base neste tipo de mapas, os agentes poderão navegar livremente, resolvendo problemas de localização, planeamento de trajectórias e verificar se a posição final foi atingida. Aqui, torna-se necessário, algum tipo de descrição ou representação do espaço navegável. O problema da detecção, primeiro, e o evitar de colisões, depois, são dos mais importantes campos de investigação no caso de robôs móveis. A maior parte dos estudos nesta área resolvem esta situação com o planeamento de percursos (de forma a evitar colisões) recorrendo a informações através de sensores. As adaptações à deslocação de pedestres virtuais, tentam evitar obstáculos móveis, recorrendo ao planeamento de trajectórias. Para obstáculos móveis, o estudo de trajectórias dinâmicas apresenta-se já como uma tarefa mais complexa.

3.2.4.1 | Planeamento de Rotas em Ambientes Dinâmicos

O planeamento de uma rota livre de colisões num ambiente dinâmico é uma tarefa que obrigatoriamente envolverá um maior número de condicionantes quando comparada com planeamentos em ambientes estáticos. A partir de uma estimativa mais ou menos precisa da posição e velocidade dos obstáculos, o agente terá de encontrar uma trajectória livre de colisões.

1. Campos Potenciais derivados da Robótica

O método de campos potenciais já referido na secção 3.1.2.7 deriva de testes para o planeamento de rotas de deslocação usando robôs. Este método usando inicialmente ambientes estáticos, poderá ser adaptado a condições de ambiente distintas, bem como a adaptação dos actores envolvidos para simulações com deslocação de pedestres virtuais.

No método de utilização artificial de campos potenciais [123], um robô móvel é condicionado por "partículas" estáticas ou em movimento, utilizando um campo electromagnético, repulsivo para os obstáculos e de atracção para os objectivos. Talvez o grande problema deste método é a dificuldade em alcançar objectivos localizados em campos de valores potenciais mais diminutos. Como vantagens, temos a robustez (pode ser aplicado sem a totalidade do conhecimento da configuração do obstáculo) e a sua rapidez (aplicável em navegação real de robôs). Pode também ser adaptado para a detecção de colisões com obstáculos móveis, e aplicado em modelos virtuais com pedestres, com aplicação de "forças" como as descritas para o caso dos robôs. O único senão aqui é a impossibilidade de um controlo real de comportamentos aquando da influência destes campos.

Uma aproximação mais conveniente para o planeamento de trajectos para ambientes dinâmicos, é representar o problema num espaço que inclua a coordenada tempo. Nesse espaço, a trajectória para evitar colisões tratará com objectos estáticos em cada um dos pontos da coordenada tempo, o que facilita o estudo e a previsão do ponto de colisão a evitar. São apresentados agora dois modelos que usam o espaço *versus* tempo na representação de trajectórias mais eficientes, comportando planeamento de movimento em ambientes dinâmicos.

2. Decomposição Rota-Velocidade

A partir do modelo de gráficos visíveis (usado no planeamento de rotas em ambientes estáticos) que visa encontrar o caminho mais curto desde o ponto A ao ponto B, através da selecção de vértices dos polígonos onde se vai colidir, e da conexão dos mesmos com o propósito de traçar a rota circundante de menor trajectória, Kant e Zucker [124] propuseram uma nova extensão para tratar obstáculos móveis. Este método é processado em dois passos:

1. É encontrado o caminho mais curto até ao objectivo considerando apenas objectos estáticos através do uso de gráficos visíveis.
2. É adicionado um perfil de velocidade a este trajecto de modo a tentar evitar objectos com movimento. Para encontrar este perfil de velocidade recorreu-se a condições de continuidade, irreversibilidade, e variações previsíveis da velocidade.

Este método oferece uma solução para encontrar uma trajectória "óptima" entre dois pontos num sistema espaço/tempo. No entanto, esta aproximação não pode modelar realisticamente o comportamento humano. Um pedestre planeia normalmente a sua trajectória considerando objectos estáticos e móveis em simultâneo, e tenta encontrar a trajectória global minimizando desvios e variações excessivas de velocidade.

Trabalhos posteriores, como os apresentados por Peter Willemsem et al [125] em 2003, e por Hongling Wang et al [126] em 2004 e associados a agentes virtuais, promovem a condução destes agentes em ambientes urbanos lidando com intersecções entre os vários tipos de habitantes virtuais (automóveis, bicicletas, pedestres, etc).

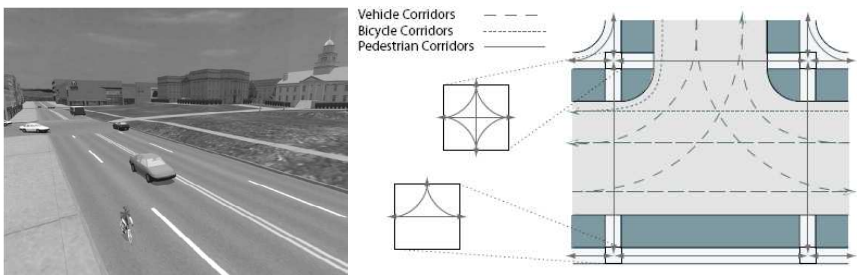


Figura 28 – Exemplo entre intersecções hierárquicas entre os vários agentes virtuais.

Imagem original em [126]

Estes trabalhos apresentam uma base de dados em tempo real da modelação de redes complexas de trajectórias em ambientes virtuais urbanos, nas quais, os agentes devem conhecer a estrutura espacial e a geometria das rotas navegáveis. Aqui a implementação dos vários tipos de intersecção, e comportamento de "escape" associado, são definidos com base em informações ambientais e informações de controlo rota/velocidade de forma a modelar o comportamento e trajectória dos agentes. São utilizados também a aceleração e desaceleração dos agentes, o controlo de movimentos (indirectamente por perseguição), e o controlo de prioridades e navegação em cruzamentos (com ou sem sinais luminosos).

3. Composição de Cilindros num Espaço/Tempo (x,y,t).

Tsubouchi e Arimoto em 1994 [127] propuseram um método que construía geometricamente uma trajectória de não colisão num espaço (x,y,t). Primeiro, foi estimada a posição e velocidade dos obstáculos móveis. Assumindo que a velocidade desses obstáculos se mantinha constante, foram deduzidos um conjunto de cilindros oblíquos no espaço (x,y,t) que proporcionariam o espaço de colisão a ser evitado. O segundo passo era encontrar a trajectória de conexão entre o ponto inicial e o objectivo a alcançar.

Assim, é traçada uma linha de ligação entre os dois pontos com um vector de velocidade constante. Se esta linha intersecta um dos cilindros definidos, é encontrada uma trajectória alternativa utilizando um algoritmo chamado de "afastamento de linhas". Este algoritmo faz uso da utilização tridimensional (aqui o tempo=3ª dimensão) de um cone com o vértice na origem e o diâmetro da circunferência variável de forma a evitar os cilindros de não navegação. É então composta uma trajectória até ao ponto de colisão, e assim sucessivamente até alcançar o objectivo. Este método descrito em [127] utiliza assim sempre o caminho mais rápido para evitar os cilindros. O ponto negativo resulta da trajectória ser conseguida através da construção de sucessivos caminhos tendo em conta os obstáculos, não havendo uma consideração e optimização global. No caso de ocorrerem múltiplos conflitos em simultâneos este método dificilmente oferece uma boa solução.

Em 2000 Frank Feurtey [128] propôs também a implementação de um modelo com base num algoritmo de movimentação de pedestres individualmente e em multidão, com base em aproximações geométricas num espaço (x,y,t). Neste modelo são

planeadas e modificadas trajectórias com base em previsões da relação velocidade e direcção dos objectos (Figura 29).

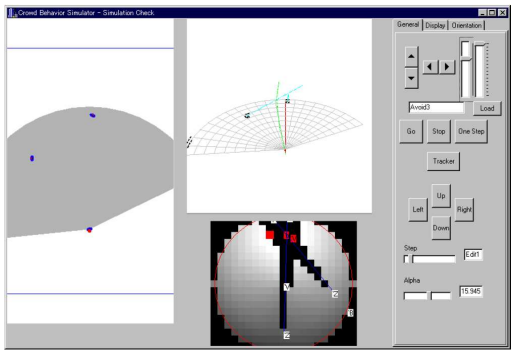


Figura 29 – Simulação e projecção de colisões no espaço/tempo.

Imagem original em [128]

Feurtey realizou, testes de simulação com uma dezena de pedestres virtuais, numa situação de projecção de situações de colisão, recorrendo a um cone de visualização geométrico (x,y,t) como meio para prever trajectórias. Neste trabalho são referidos alguns factores que determinariam a qualidade do fluxo, são eles: a liberdade de escolha na velocidade e alcance dos agentes, a habilidade de movimentação contra-corrente e a individualidade. Um dos problemas deste algoritmo era uma escalabilidade deficiente e a correspondente não aplicação a um maior número de agentes.

3.2.5 Comportamentos de Pedestres Associados à Informação Embebida no Ambiente

Trabalhos mais recentes, como por exemplo o sistema comercial de maior sucesso em jogos de computadores neste tipo de abordagem, "Os Sims" [129], um pouco à imagem da utilização dos campos potenciais no caso dos robôs, recorre a objectos ou localizações no espaço que associam aos agentes comportamentos específicos. Assim a interacção com determinados objectos provoca a advertência para determinados processos associados, tais como por exemplo "satisfazer a fome", resultando no executar de um procedimento quando a personagem virtual responde a esta advertência. Este sistema atribui um enorme realce e importância à construção (ou edição) detalhada de um ambiente rico em objectos, ou em localizações específicas, onde as personagens virtuais responderão a estas especificações inerentes ao meio.

Contém assim, na mesma linha, também e só, comportamentos específicos e direccionados para os objectos presentes numa determinada cena.

Já anteriormente, análogo ao conceito adjacente utilizado nos "Sims", Kallmann e Thalmann [130], descreveram a ideia de objectos inteligentes como: objectos que fornecem um plano para o seu uso. Aqui, uma personagem em aproximação a um objecto específico é alertada à execução de um conjunto de tarefas características do local ou do objecto. Por exemplo, um elevador informa à personagem por perto que pode pressionar botões, esperar, e entrar quando as portas se abrirem.

Outros trabalhos, como por exemplo [131], lidam com uma maior abstracção do ambiente de navegação, para ambientes estáticos e dinâmicos, representado o ambiente num mapa/grelha de alturas em 2D. Este mapa identifica os obstáculos que as personagens devem evitar, os espaços livres de navegação, e informação sobre obstáculos especiais tais como o tronco de uma árvore suspenso a poucos metros do chão onde os personagens devem rastejar ou saltar (Figura 30).

Figura 30 – Personagens virtuais lidando com informação nos obstáculos.
Imagem original em [131]

Esta informação pode ser automaticamente calculada depois de terem sido assinalados previamente, a localização e o tipo de obstáculos no ambiente. Cada comportamento especial, como o rastejar, terá também de ser pré-anotado ao obstáculo associado. O valor de alturas, dado pelo mapa de alturas (*height map*) em cada célula, é usado para poder lidar e representar terrenos com inclinações, e assim ajustar a elevação das personagens correspondente para esse espaço.

Uma plataforma de separação em diversas camadas de interacção do agente virtual com o mundo virtual a povoar proposta por exemplo por Tecchia et all [5] (já vista

anteriormente), permite a criação de comportamentos complexos e bastante reais, combinando o efeito das várias camadas onde cada agente individual reage dependendo da área que ocupa e a sua posição relativa em relação a outros agentes. A camada de comportamento referida por estes autores, comporta comportamentos a serem codificados para cada local da grelha de navegação. Usando um mapa colorido de 8 bits por cada componente RGB em cada célula, obtêm-se 2^{32} possibilidades de implementar diferentes comportamentos (Figura 31).

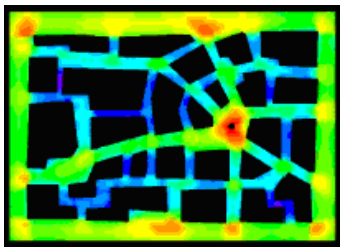


Figura 31 – Camada comportamental por correspondência a mapas coloridos.

Imagem original em [5]

Assim quando um agente alcança uma célula, o sistema testa a codificação da cor dessa célula de modo a decidir qual o comportamento a adoptar. Nesta plataforma, podem ser implementados comportamentos simples como ”esperar”, “virar à esquerda”, ou mais complexos como “calcular uma nova direcção com base no ambiente envolvente”. Através de zonas de coloração distintiva, os agentes poderão ser atraídos para pontos específicos. Alguns destes pontos específicos poderão conter informações associadas para a definição de um posterior comportamento do agente (camada de retorno).

3.2.6 Resumo dos Modelos de Simulação Comportamental

Método	Objecto	Vantagens	Desvantagens
Boids, Rule-Based	Rebanhos de animais, insectos	Aspecto visual da animação de comportamentos em rebanho (*Flocking*).	Implica um comportamento cooperativo simples.
Global	Multidão como um todo	Fornece uma estimativa do fluxo, densidade, média de velocidade e tempo de evacuação.	Não há descrição individual.
Forças Sociais	Pedestres	Revela fenómenos e comportamentos emergentes.	Descrição simplista de comportamento.
Sistemas de partículas	Pedestres	Trata com uma grande multidão.	Descrição simplista de comportamento.
Campo artificial de potências	Agentes em amb. estáticos e dinâmicos	Cálculo rápido da trajectória para evitar obstáculos.	Dificuldade em obstáculos em movimento, baixo controlo de direcção e velocidade.
Rota/Velocidade Decomposição	Agentes em ambientes dinâmicos.	Encontra um caminho óptimo.	Irrealista para os humanos.
Aproximação Geométrica com composição de cilindros	Agentes em ambientes dinâmicos.	Boa representação.	Não contém uma optimização global e escalável.
Informação embebida no ambiente	Agentes em interacção c/ objectos	Aspectos visual e comportamental da animação	Optimização dependente do ambiente. Por norma ambientes 2D ou 2,5D.

Tabela 1 – Resumo dos métodos de simulação comportamental analisados

3.3 Abordagem Integrada

A construção de cenários urbanos tridimensionais é, por excelência, actualmente uma das tarefas nas áreas da computação gráfica (CG) e processamento digital de imagens (PDI) muito requisitada. A disponibilização massiva na Internet destes modelos virtuais representativos de estruturas de edifícios ou ambientes reais existentes no passado, no presente e possivelmente no futuro, tem vindo a crescer com uma dimensão tal, que representam já hoje uma pequena indústria a nível mundial, capaz de mover ou actuar como factor de investigação e desenvolvimento, ou mesmo fazer emergir, outros sectores de actuação e expansão de actividades científicas paralelas.

Se numa primeira etapa, o objectivo principal era dar a conhecer estes mundos de uma forma mais abrangente e realista, com capacidades superiores da sua percepção, interacção e navegabilidade pelas estruturas virtuais que os compõem, tende-se agora, evoluir, para a inclusão de vida artificial nesses ambientes. Deixamos assim de ter edifícios ou cidades "frias" (designadas por muitos investigadores aos ambientes virtuais não povoados por associação a cidades desertas), para incorporar personagens que povoem de alguma forma estas construções. Esta virtualização da vida envolvendo a geometria estática, à imagem do que acontece no mundo em que vivemos, pioneira nos jogos de computador, representa um interesse crescente para vários tipos de análise em diversas áreas, ou, simplesmente para simulações ou representações públicas e pessoais de uma sociedade virtual capaz de actuar com e sem determinados constrangimentos humanos aquando da actuação no mundo real.

Quando o ambiente virtual contempla modelos que representam objectos aparentando vida própria, é fundamental que a simulação mantenha realista o seu comportamento nas mais diversas e possíveis situações que possam ocorrer durante a exploração desse ambiente virtual. Nestes cenários, segundo [132], há três características da maior importância: a qualidade do próprio ambiente virtual, a qualidade da interacção e a qualidade comportamental dos objectos que o compõem. Ora, por um lado a qualidade do ambiente virtual passa actualmente por conseguir imagens foto-realistas bidimensionais acopladas à estrutura geométrica (muitas vezes deficitária) promovendo cenas virtuais da própria virtualidade inicialmente requerida, recorrendo a texturas e impostores visuais 3D; por outro, e de uma forma mais ambiciosa, a especificação do ambiente é abordado hoje em dia com um detalhe superior a nível geométrico de completas e realistas (a todos os níveis) representações de cenários virtuais realmente tridimensionais, capazes de serem aproveitadas em todas as suas potencialidades. É neste segundo aspecto que se focará principalmente o trabalho aqui realizado. Quanto à qualidade da interacção e representação comportamental dos agentes aí inseridos, a área de recurso e abrangência contemplam desenvolvimentos recorrendo à abordagem integrada de técnicas de CG e especificações e desenvolvimentos na área da IA.

Nasce assim a possibilidade de interagir com estes modelos, elevada a uma conjuntura superior, tirando partido do maior detalhe dos modelos (a nível geométrico, texturas, efeitos gráficos, etc), com condições e especificações mais adaptadas para animações mais realistas e abrangentes. Neste tipo de construções de

cenas, o desejo estende-se em poder simular a movimentação de personagens virtuais em todas as situações possíveis proporcionadas por estas novas representações e especificações de ambientes virtuais e objectos simulando vida, para deixarmos de lado, simulações condicionadas a uma estrutura de movimentação horizontal, mesmo podendo conter já esta, variações de alturas no mesmo plano, para passarmos a poder proporcionar navegação realmente tridimensional com base numa cartografia geo-virtual 3D, utilizando a conjugação complementar também do plano vertical para deslocações com os mesmos graus de liberdade à imagem do que acontece no mundo real, capaz de proporcionar maior abrangência de interacção e comportamento mais realista das personagens habitantes.

Esta modelação de habitantes, caracterizando modelos ou personagens virtuais em deslocação, descrevendo por exemplo animais, veículos ou pedestres, simulados nestes casos por avatares ou agentes puramente virtuais, recebe atenção dos pesquisadores de animação comportamental e de realidade virtual há vários anos [1, 133, 134]. Os esforços convergem tanto nos aspectos visuais (aparência, movimento), como no caso dos pedestres em comportamentos mais próximos ao de um ser humano real.

Em computação gráfica (CG), na área da animação comportamental, verifica-se uma evolução desde os primeiros modelos comportamentais de Reynolds [1] à modelação de complexos mundos com várias personagens exercendo as mais diversas actividades. Por outro lado, observa-se na inteligência artificial (IA), mais especificamente, em sistemas multi-agentes (SMA[12]), a tendência de modelar o processo cognitivo inerente aos seres humanos, indo desde modelos reactivos [135] até cognitivos [136]. Os modelos reactivos reflectem agentes que não possuem um modelo de raciocínio e agem com base em respostas a estímulos externos. Por outro lado, os agentes cognitivos, ou deliberativos, são os que possuem um modelo simbólico de raciocínio e um plano a ser realizado e/ou negociado com outros agentes para alcançar os seus objectivos.

Outra tendência observada é traduzida em esforços em tentar integrar os resultados destas duas áreas (CG e IA) para que personagens autónomas possam ter comportamentos mais aceitáveis, ou seja, com maior credibilidade, mais próximos da realidade. Como exemplos dessa tendência podem citar-se trabalhos como: Perlin

[12] Sistema Multi-Agentes (SMA) ou Multi-agent Systems (MAS)

[137], Badler [138], Funge [139] e Torres et al. [140]. Nestes trabalhos, a preocupação foi dotar os personagens virtuais de um modelo de raciocínio cognitivo. Entretanto, além desse aspecto, esta linha de solução está relacionada a propiciar também comunicação entre as personagens autónomas, tendo em vista que um dos principais factores nas relações sociais, e portanto no comportamento dos indivíduos, é a possibilidade de comunicação entre os integrantes de uma sociedade [141], seja por interesses próprios de cada indivíduo, ou por interesses colectivos, resultando daí tarefas colaborativas e/ou cooperativas.

Um aspecto a destacar, ao modelar personagens que representam seres humanos, sendo estes seres sociais que constantemente utilizam comunicação para reafirmar suas qualidades sociais, é a naturalidade de modelação destas propriedades também nas suas representações virtuais. Segundo Badler [138], ao se tentar aumentar a credibilidade das animações de um modelo virtual, as acções e comunicações não verbais são tão ou mais importantes do que a aparência em si. Desta forma, a possibilidade de modelar os aspectos relacionados ao raciocínio com alguma forma de comunicação, acrescenta uma poderosa dimensão na representação dos aspectos sociais dos humanos virtuais, aumentando a sua credibilidade enquanto habitantes de um mundo virtual, e contribuindo em consequência para um maior nível de realismo desses mundos.

Entretanto, observa-se ainda que tais personagens (principalmente os humanos virtuais) não atingiram um nível aceitável de credibilidade, principalmente no que se refere à representação do comportamento, o que se torna mais evidente em aplicações em tempo real, complexas e dinâmicas. A dificuldade no desenvolvimento destes cenários habitados, a sua especificação, construção de sistemas e modelação de comportamentos, muito provavelmente reside no facto de que esta representação depender e requer investigação em duas linhas: a computação gráfica, para permitir a visualização em tempo real de um elevado número de personagens virtuais; e agentes inteligentes para permitir o controlo e definição de comportamento das mesmas.

3.4 | Projecto de Trabalho

Os testes de detecção de colisão empregues para manter cada agente ou humanos virtuais "ciente" da sua posição no enquadramento da envolvência ambiental, são essenciais para tarefas de planeamento de trajectos e técnicas de desvios realistas em relação aos obstáculos que poderão encontrar. Neste enquadramento irão ser utilizadas técnicas de discretização do espaço utilizando projecções e mapas de alturas recorrendo ao z-Buffer com outro nível de profundidade e eficiência, recorrendo a estruturas multi-nível simples e bem organizadas inferidas de estruturas arbitrárias tridimensionais, para conseguir manter e gerir em tempo real as condições de exigência requerida inicialmente para a navegação num determinado ambiente mais ou menos complexo (2D ou 3D) com um número significativo de agentes ou avatares. À questão da detecção de colisões nestas condições, é elaborada a investigação exposta no capítulo quarto desta tese.

As pretensões e desenvolvimentos anteriores são inerentes e complementares com a necessidade focada no planeamento de rotas de navegação para simular posicionamentos e deslocações "inteligentes" das personagens virtuais requeridas num vasto número de áreas de aplicação. A maioria dos trabalhos de investigação nesta área usa uma etapa de pré processamento para o cálculo de mapas de percursos através da procura em grafos hierárquicos. Mas será que estes abordam a procura de percursos em cenas realmente 3D, de forma integrada com as necessidades de detecção de colisões? Como actuam se considerarmos um mundo com vários níveis sobrepostos e navegáveis em altura, em interiores de edifícios. Estes algoritmos terão realmente em linha de conta a altura na procura de destinos, e as necessidades específicas na tomada de decisões aquando as mudanças de alturas (subidas e descidas de escadas ou rampas para alcançar mais eficientemente os objectivos)? A esta questão é dedicado o capitulo quinto.

A modelação e simulação baseada em agentes têm-se mostrado uma abordagem interessante onde investir na implementação de sistemas mais complexos para a simulação virtual de pedestres. Os sistemas actuais podem não resolver de forma eficiente, escalável, reutilizável, e adaptados a necessidades específicas de actuação (como o caso do tipo de ambientes virtuais que temos vindo a propor), alguns mecanismos comportamentais associados ao realismo requerido das personagens em actuação. São assim apresentadas no capítulo sexto, alguns modelos de comportamentos associados à caracterização da credibilidade e autonomia em

simulações de populações de entidades virtuais imitando comportamentos humanos, assentes numa arquitectura de actuação proveniente das investigações apresentada nos capítulos anteriores, e adaptadas às condições de simulação dos ambientes 3D mencionados anteriormente.

CAPÍTULO 4

Detecção de Colisões Eficientes e Conservativas para Mundos 3D Densamente Povoados

Com o advento de mais e maior poderio a nível de hardware gráfico (GPU[13]), bem como um maior desenvolvimento na programação específica associada (por exemplo através da utilização de *shaders*), capaz de efectuar num espaço muito curto de tempo (na casa das décimas de segundos) o *rendering* de milhões de triângulos, são proporcionadas condições excelentes para o vasto número de projectos que actualmente emergem um pouco por toda a parte, virtualizando com um considerável nível de detalhe, grandes porções das actuais cidades, importantes e destacados edifícios, construções industriais e arquitectónicas, representações virtuais de civilizações ou espaços já desaparecidos (projectos arqueológicos), etc.

Estes novos mundos 3D, convertem-se com um grau de realismo e interesse superior, quando povoados com personagens virtuais. Incorporar estas personagens sintéticas aos mundos virtuais 3D constitui uma nova fase de desenvolvimento na aprendizagem do contexto virtual, com o exemplo interessante de podermos usar os avatares para uma percepção mais realista do sentido de escala dos modelos em questão.

A visualização destes projectos em tempo real, requer o uso de um conjunto de técnicas dependentes da sua performance de modo a puder proporcionar animações fluidas, realistas e interactivas. A actual conquista de um realismo visual superior em tempo real, conseguido por exemplo pela inclusão de modelos de iluminação cada vez mais próximos dos modelos globais fisicamente correctos, ou a distribuição e especificação de tratamentos diferenciados das componentes visuais e

[13] GPU (Graphics Processing Unit, ou Unidade de Processamento Gráfico)

comportamentos físicos pelos vários núcleos dos processadores, ambos sincronizados em tempo real, significa um potencial já bastante elevado, mas ainda com a necessidade recorrente provocada pela ânsia de ter cada vez mais e melhor (associados à condição humana) em desenvolvimento e avanços ao nível tecnológico (em especial de hardware e software).

Paralelamente à necessidade da alta performance visual, o objectivo de povoar mundos virtuais congrega uma série de processos nas implementações do sistema: o mais comum e talvez o mais importante é a detecção de colisões entre os objectos em movimento e o mundo, e entre os próprios objectos em movimento. A detecção de colisões é uma componente fundamental em várias aplicações na área da computação gráfica, realidade virtual (RV) e simulação na área da física. O maior desafio deste tipo de detecção em tempo real é conseguir determinar quais são os objectos em contacto ou colisão sem comprometer (em demasia) o desempenho da aplicação.

Existem muitos factores para que um mundo virtual simule um ambiente real, um desses factores pode ser por exemplo analisar as características físicas e comportamentais de um ou vários humanos virtuais. Uma das primeiras características a ter em conta para a navegabilidade realista destes pedestres simuladores, é a capacidade de perceberem a presença de zonas no ambiente, onde dadas as suas características humanas desejadas, estarão aí privados de movimentação, limitando-se assim, a zonas de possibilidade ou impossibilidade de passagem ou navegação no ambiente, projectando no mundo virtual uma simulação do real, onde a lei da física regente nos indica que o mesmo espaço não pode ser ocupado por mais de um tipo de matéria. A possibilidade de considerar a detecção de colisão que ocorre durante a navegação em ambientes virtuais, é uma técnica fundamental para o realismo e orientação entre os vários objectivos em deslocação espacial. Esta técnica, custosa de aplicar a mundos desconhecidos (mesmo aqueles unicamente aptos à navegação planar), deverá identificar o determinado momento e local da simulação onde existe contacto entre os diferentes objectos virtuais quando pelo menos um deles se encontra em movimento.

Incluir simulação de multidões virtuais humanas em modelos estáticos ou dinâmicos, projecta actualmente ainda um custo muito elevado, não só ao nível da renderização, mas principalmente em relação à detecção de colisões. Se adicionarmos a possibilidade e capacidade das personagens poderem navegar pelo modelo de uma

forma livre[14], existe a necessidade de recorrer a métodos mais rápidos e mais abrangentes, sem nunca descurando o aspecto realista desejado de apresentação, comportamento e detecção de colisão.

Considerações Prévias

Neste capítulo é descrito o desenvolvimento de um método eficiente na implementação de detecção conservadora de colisões entre avatares e o mundo 3D. Os modelos dos mundos 3D que servirão de base à integração de personagens virtuais poderão no limite ser uma "sopa" de polígonos não relacionados enquanto modelação/geometria diz respeito, quer isto dizer, que o método proposto não restringe nem impõem condições à estrutura da malha poligonal da cena a habitar. O método está apto para lidar com mundos arbitrariamente complexos, sem que isto possa comprometer a performance e escalabilidade. Existe ainda uma clara imposição de eficácia e eficiência para a computação de detecção de colisões em ambientes 3D multi-nível com alto desempenho em tempo real.

O projecto foi desenvolvido nas linguagens C/C++. Foi utilizada a biblioteca OpenGL como meio de suporte de apresentação/comportamento/animação das personagens e mundo 3D envolvente.

A primeira preocupação foi utilizar uma rotina eficiente e capaz para a importação do modelo (mundo 3D) a ser povoado. Seguiu-se a extracção da informação relevante e necessária aos objectivos do projecto apartir desse mundo 3D. A utilização de um mapa de alturas 2D fornecido a partir dos valores armazenados no buffer de profundidade (*z-Buffer ou depth buffer*) captado desde uma vista superior, à imagem do que foi utilizado em [2], foi o primeiro passo da investigação para a detecção de colisões.

Na captação do mapa de profundidades seria utilizada uma projecção ortográfica, onde não haveria distorção da imagem quando a câmara se movimentasse no eixo do Z. O tratamento, e a estruturação dos dados recolhidos e armazenados em memória serviriam para testes de colisão e definição comportamental dos agente com o ambiente envolvente.

[14] O termo livre aqui descrito reflecte a liberdade de movimentação do agente no ambiente 3D a povoar, de acordo com todas as possibilidades disponibilizadas de deslocação fornecidas por uma estrutura 3D.

O último passo seria a incorporação de um elevado número de personagens virtuais em movimentação, bem como uma diversidade de avatares ou agentes variada, implementação, experimentação e testes de navegação multi-nível com foco na detecção de colisões em tempo real.

4.1 | Descrição Geral

O método aqui proposto estabelece mecanismos eficientes de detecção conservadora de colisões em mundos 3D virtuais com vários níveis de altura navegáveis (multi-nível). Um exemplo deste tipo de ambientes pode ser observado na Figura 32. Neste mundo sintético o avatar poderá navegar nos 3 andares, subindo e descendo as rampas, escadas, ou outros pequenos obstáculos. Deverá ser capaz de detectar colisões com os automóveis, pilares, e outros objectos presentes na cena. Deverá também ter a capacidade de não cair de uma altura elevada (queda dos andares superiores).

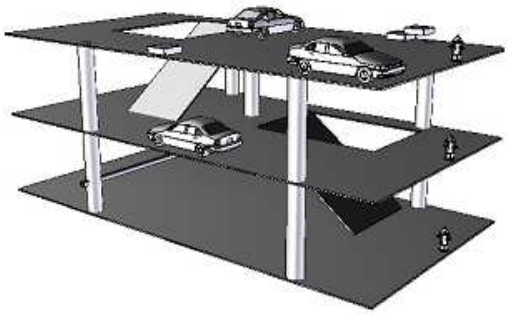


Figura 32 – Mundo Virtual Multi-nível

O principal objectivo do método aqui presente, é fornecer uma forma eficiente de controlo da posição do avatar, prevenindo por consequência, as colisões com o mundo e os objectos presentes no mesmo.

É assumido que a altura de cada um dos avatares presentes no mundo (*avatarHeight*) seja conhecida.

O método é iniciado extraindo automaticamente informação do mundo 3D a povoar, de modo a poder determinar quais as áreas de navegação para os diferentes avatares,

bem como a actualização da altura em tempo real a que se devem posicionar quando em movimento.

Isto é conseguido pela fatiação do mundo 3D recorrendo a diversos planos horizontais. Para cada fatia, são armazenados em memória dois grupos de valores: a altura a que é efectuado o corte, e todo o mapa de alturas do plano obtido com essa fatia. O processo de fatiação tem em consideração o valor das alturas dos vários avatares de modo a poder decidir a que altura, a próxima fatia deva ser obtida. Este processo de fatiação poderá gerar um grande número de fatias porque é assumido não haver nenhum tipo de conhecimento anterior do modelo ou ambiente 3D a povoar. Uma fatiação nestas condições iria supor um enorme consumo de memória para o armazenamento de todo o mapa de alturas por fatia processada. No entanto, somente algumas destas fatias serão absolutamente necessárias para uma correcta navegação e controlo do espaço navegável. É então estabelecido um algoritmo de verificação da necessidade absoluta da indispensabilidade de guardar cada fatia processada, e eliminação daquelas que não acarretem informação adicional, de modo a produzir um controlo eficiente da memória física essencial para a navegação em tempo real. Este processo será detalhado na secção 4.3. - Etapa de Pré Processamento: Fatiação do Mundo Virtual

A etapa de pré processamento é razoavelmente rápida devido à principal carga da operação computacional se situar ao nível do *rendering* do mapa de profundidades exigido para cada fatia envolvida. Para o *rendering* do mapa de profundidades não é necessário a instalação de shaders, componentes de luzes, ou outros tipos de efeitos de luminosidade que normalmente provocam o abrandamento do tempo de *rendering*. Além disso, somente a primeira fatia incorporará toda a cena e assim um *rendering* do mapa de profundidades total do ambiente, porque à medida que a fatiação percorre a cena 3D, cada vez menos geometria será envolvida neste processo, onde o tempo de *rendering* das fatias finais será muito inferior ao tempo consumido pelas fatias iniciais.

O mundo virtual quando submetido à simulação e visualização, depois da fase de pré processamento concluída, as camadas ou fatias resultantes são usadas para representar duas importantes peças de informação:

- A altura da base de colocação de cada um dos avatares envolvidos,

- As áreas com espaços livres de movimentos por onde cada avatar se poderá movimentar.

Este processamento é também muito simples desde o ponto de vista computacional, acarretando pequenas quantidades de pesquisas nas fatias guardadas em fase de pré processamento. A simplicidade de todo este processo permite executar detecção de colisões de forma conservativa, com milhares de avatares movimentando-se sobre um mundo virtual com um nível de complexidade bastante elevado. Os passos de desenvolvimento e teste em tempo real serão detalhados a seguir.

4.2 Etapa de Detecções de Colisão em Tempo Real

De modo a simplificar e facilitar o processo de apresentação do método, e sem perder generalidade, os esquemas exemplificativos seguintes serão apresentados em 2D, representando secções no plano XY desde uma determinada posição do mundo virtual.

Assim, e tomando como referência o mundo virtual apresentado na Figura 33, vamos assumir que a fase de pré processamento computou duas fatias. As duas linhas horizontais a negrito representam os vários Clip Planes usados para efectuar o *rendering* do mapa de profundidades, e a legenda correspondente indica a altura a que cada fatia está posicionada em relação ao volume do mundo virtual. Neste exemplo, a primeira fatia é tomada com um Clip Plane de Y=13,6, e a segunda fatia é guardada para um Y=5,9. As várias caixas com os números sobre cada linha das fatias apresentadas, representam os píxeis no mapa de alturas, com os números a indicarem o valor da altura guardado para essa posição.

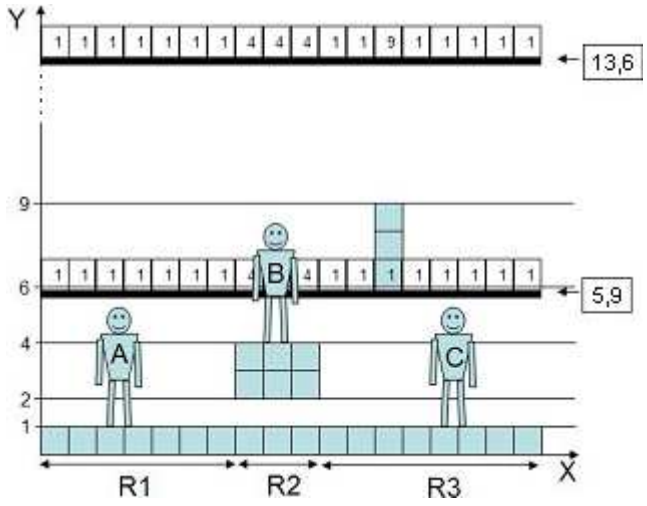


Figura 33 – Mundo fatiado com avatares

Por defeito cada avatar apresenta três parâmetros iniciais: a largura, a altura e o salto. Os dois primeiros estão relacionados com as dimensões espaciais do avatar, o terceiro indicará o quanto ou a que altura o avatar poderá subir ou descer. Este último parâmetro poderá ainda ser configurável com valores diferentes para subidas ou descidas. À imagem da vida real, os avatares poderão ter maior capacidade de salto na descida comparado com a possibilidade aquando a subida de obstáculos.

Focados novamente no exemplo da Figura 33, e assumindo que um avatar tem uma capacidade de salto constante e menor a 3 unidades, então o movimento dos avatares será limitado às regiões apresentadas na imagem, neste exemplo o avatar A poderá apenas movimentar-se na região R1, o avatar B na região R2 e o avatar C na região R3. Todo este tipo de informação pode ser extraído das duas fatias assinaladas na imagem. Cada avatar efectuará a leitura da informação da fatia posicionada acima da sua cabeça. Assim os avatares A e C basear-se-ão na fatia inferior, posicionada em Y=5,9, enquanto que o avatar B fará a leitura do píxel correspondente na fatia superior, em Y=13,6. Se o avatar B tentar mover-se para a região R1, saberá que terá de mover-se de uma região de altura 4 para outra de altura 1. Assumindo que o salto dos avatares seja inferior a 3 unidades, então o movimento será classificado de ilegal. Similarmente os avatares A e C não poderão mover-se para a região R2.

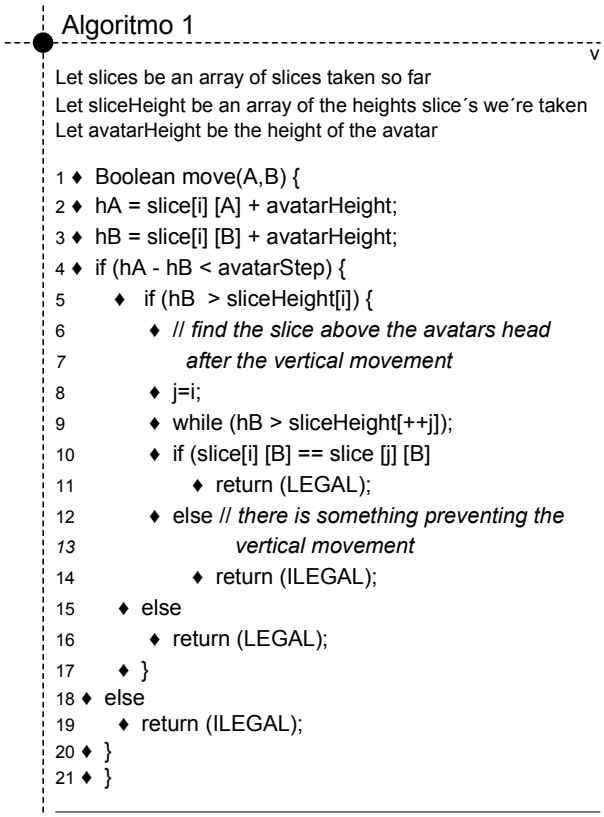
O mundo é assim decomposto em células (correspondentes a píxeis) no eixo do X (nos casos 3D a decomposição será nos planos XZ). Cada avatar será posicionado numa destas células a uma altura (*height*) em particular, tomando a altura do topo da

sua cabeça em consideração (os avatares A e C estarão a uma altura = 5,5, e o avatar B a uma altura = 8,5).

Quando um avatar pretender mover-se para uma célula vizinha, o movimento é decomposto num movimento vertical seguido de um outro movimento horizontal. Primeiro será necessário aferir se a capacidade de salto do avatar é suficiente para vencer as diferenças de alturas entre a célula actual e a célula de destino. Assumindo que capacidade de salto é inferior à altura requerida para a deslocação, então só será testada a magnitude do movimento vertical. Assumindo que esta magnitude não é superior à capacidade de salto do avatar, e consequentemente o avatar têm condições de superar os obstáculos, então é necessário testar se o espaço alvo está ou não ocupado.

Se após o movimento vertical a nova altura a que se situará o topo da cabeça do avatar ficar ainda abrangido pela mesma fatia, então o movimento é legal. Se por exemplo o avatar estiver em subida no mundo virtual, e a altura do topo da sua cabeça estar já acima da fatia processada anteriormente, então será necessário usar uma nova fatia de referência para esse avatar, mais precisamente a próxima fatia que se encontrar acima da sua cabeça após o movimento de subida de modo a poder validar este e outros movimentos futuros.

O Algoritmo 1 descreve em detalhe todo este processo:



Algoritmo 1 – Algoritmo de teste em tempo real para avaliar a legalidade do movimento do avatar

Resumindo o processo de avaliação da legalidade dos movimentos dos avatares, e focando algumas linhas do Algoritmo 1 juntamente com a Figura 34 de apoio exemplificativo, podemos analisar algumas situações de teste ao algoritmo relacionadas com determinadas condições de movimentação dos avatares presentes num novo cenário. De referir inicialmente que a função *move* do Algoritmo 1 (linha 1) receberá a posição actual e de destino do avatar (variável A e B respectivamente) e responderá com a informação de possibilidade ou não de deslocação. Notar ainda que as variáveis hA e hB (linhas 2 e 3) estabelecem a altura da posição actual e futura dos avatares.

Através da Figura 34 podemos notar a tentativa de movimento de dois avatares (avatar A e C) em sentidos opostos. Consideremos o salto dos avatares (*avatarStep*) igual a 2 unidades. A tentativa de deslocação em uma célula do avatar A será considerada legal visto que o valor de hA e hB se mantém igual (5.5 unidade) logo a sua diferença (0 unidades) ser inferir ao salto do avatar (linhas 4, 5 e 16). Caso este avatar continue e efective a sua deslocação para a célula seguinte, notar agora que o novo hB se encontrará já sob a abrangência da fatia acima (*slice[j]*) devido ao novo posicionamento superar um degrau de altura=1. No entanto, ainda poderá suportar

este movimento já que a diferença do hA com o novo hB ainda se mantém inferior ao valor de salto do avatar e os valores de altura de destino da duas fatia ser o mesmo (linhas 4, 10 e 11).

No caso da tentativa de movimento do avatar C em uma unidade, também o novo hB se encontrará já sob a abrangência da fatia acima (slice[j]), mas agora mesmo que a diferença do hA com o novo hB permita o movimento, os valores no mapa de alturas para o mesmo destino nas diferentes fatias é já diferente, o que retorna um valor ilegal de possibilidade de locomoção (linhas 4, 5, 10 e 14), já que a altura do avatar seria superior ao espaço em altura presente na cena para esta movimentação.

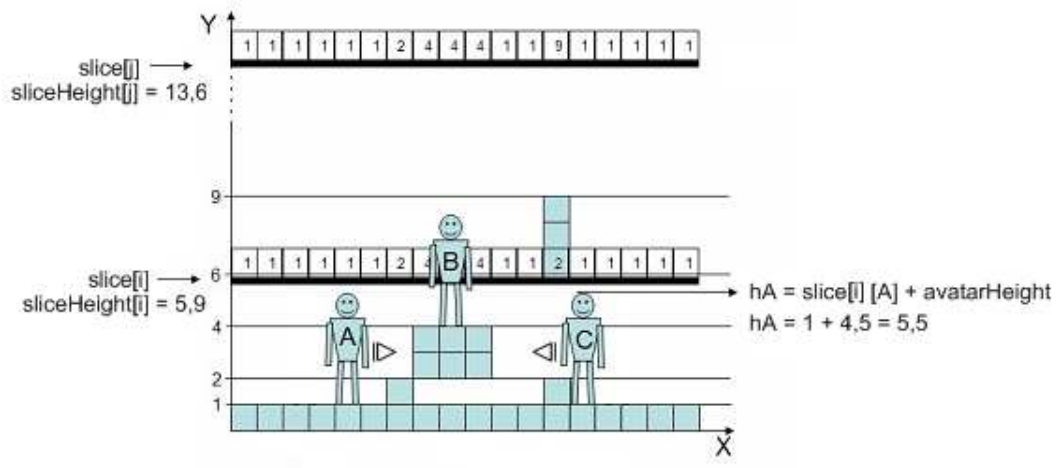


Figura 34 – Cena exemplificativa de apoio ao Algoritmo 1

O teste de legalidade de todas as movimentações de subidas ou descidas está assim garantido com a correspondência e adaptação do Algoritmo 1 às condições físicas de qualquer ambiente virtual. A detecção de colisões entre avatares é também resolvida usando uma estratégia similar. Um bit extra é guardado em cada uma das células anteriores, a qual indicará em tempo real o estado de ocupação desta célula. Este bit deverá ser testado antes do movimento do avatar de modo a não permitir colisões entre estes. A descrição deste processo será abordado mais adiante nos capítulos 5 e 6.

4.3 Etapa de Pré Processamento: Fatiação do Mundo Virtual

Esta secção detalha a etapa de pré processamento do método onde se apresentam alguns exemplos ilustrando situações comuns de navegação multi-nível. O resultado desta etapa será a quantificação e guarda das fatias e correspondentes células com os valores em altura necessárias e essenciais à correcta condição de navegabilidade e detecção de colisões exigida.

4.3.1 Setup

Inicialmente são alinhados e computados os eixos constituindo o paralelepípedo envolvente do mundo (entre um Plano Z-Near e um Z-Far). Este processo poderá ser implementado sem um custo adicional quando o modelo do mundo está a ser carregado. Os valores máximos e mínimos de cada eixo do mundo virtual a carregar são guardados em variáveis *maxX, minX, maxY, minY, maxZ, minZ*. É então colocada uma câmara (vista) ortográfica no topo do mundo virtual, "olhando" para baixo ao longo do eixo Y, obtendo um volume de visualização que englobe o "*bounding box*" previamente estabelecido. O Near Plane é colocado acima do valor maxY guardado, e o Far Plane é colocado abaixo do valor minY (os planos mais a escuro da Figura 35 representam o Near e Far Planes).

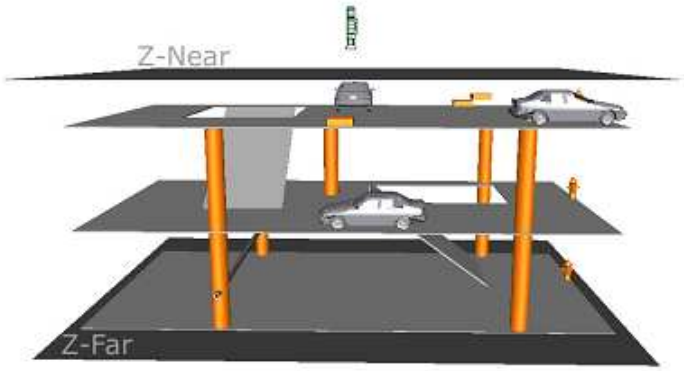


Figura 35 – Mundo Virtual Multi-nível

Retomando os esquemas exemplificativos em 2D de modo a facilitar a apresentação do método, a Figura 36 identifica a representação de uma secção no plano XY neste tipo cena.

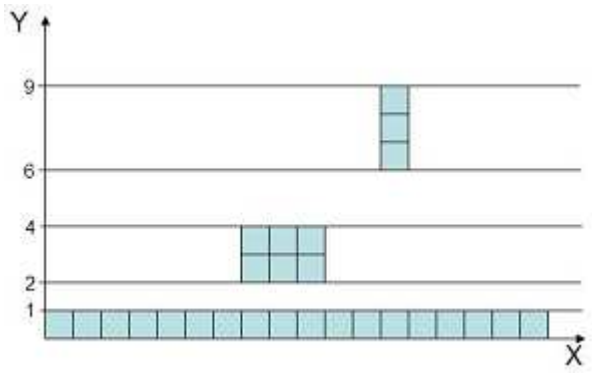


Figura 36 – Simplificação do mundo Virtual

O Near Plane é colocado assim a uma altura constante definida pela Equação 1.

$$hNearPlane = maxY + avatarHeight + resolution$$

Equação 1 – Colocação do NearPlane

Onde a variável *resolution* (resolução) indicará o erro vertical máximo durante a descretização de um mundo virtual para o objectivo da detecção de colisões. Este valor não influenciará a capacidade de o avatar se posicionar de acordo com a altura do chão dada pelo mapa de profundidades. Na prática, este valor *resolution* implica que o avatar terá por exemplo este valor de espaçamento entre a sua altura máxima e um qualquer limite ou objecto colocado acima deste, para poder testar a passagem por debaixo.

O Far Plane é também uma constante para cada mundo virtual importado, o seu posicionamento é definido pela Equação 2.

$$hFarPlane = minY - resolution$$

Equação 2 – Colocação do FarPlane

Dentro destes limites é efectuado o *rendering* e processado o mapa de profundidades de modo a obter o mapa de alturas da totalidade do ambiente. Assumindo um avatar com uma altura de 4.5 unidades, e uma resolução de 0.1 unidades, a altura da primeira fatia seria tomada em Y=13.6 segundo a Figura 37, dado que o registo máximo de altura aqui é de 9 unidades, que juntamente com as 4.5 unidades de altura do avatar e o valor de resolução, pressupõem a altura da primeira fatiação (e colocação do Z-Near).

Assumindo que *max(i)* é a altura máxima guardada na fatia i. A próxima fatia, fatia *i+1*, será tomada a uma altura definida pela Equação 3.

$$nextSliceHeight = Max(i) - resolution$$

Equação 3 – Definição da altura de corte da próxima fatia

As duas primeiras fatias são apresentadas na Figura 37.

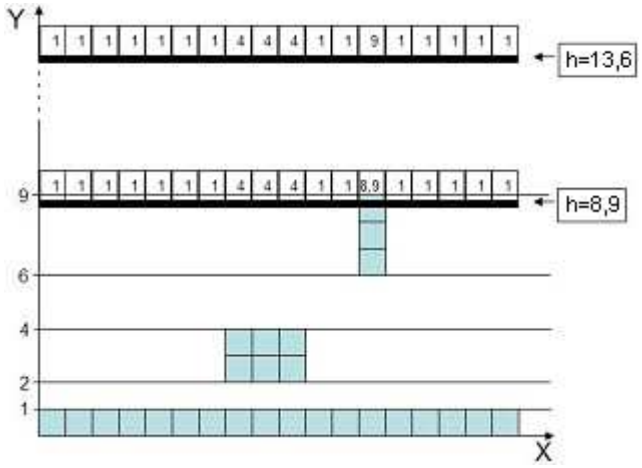


Figura 37 – Primeira e Segunda Fatias

A terceira fatia, de acordo com a equação 3, deveria ter tomada a uma altura de Y=8.8. A altura das fatias seguintes vai sendo definida pelo parâmetro resolução até Y=5.9. Ver Figura 38.

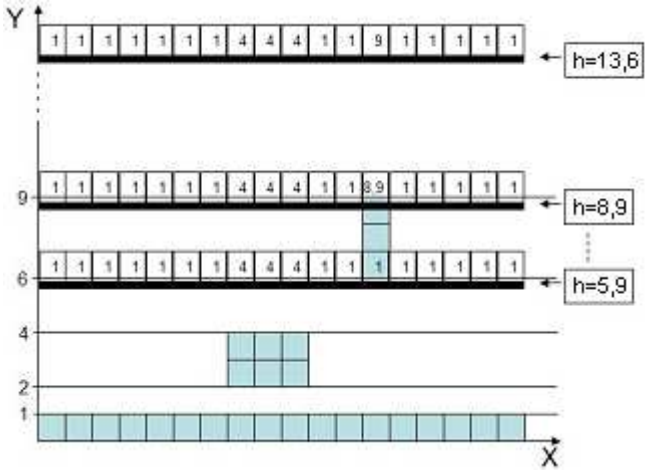


Figura 38 – Fatiação até altura de 5,9

O valor máximo de altura no mapa de profundidades presente na última fatia (Y=5.9) é 4, e desde que a altura dos avatares não desça deste valor (no caso actual é 4.5)

então não há necessidade de continuar a fatiação. Assumindo que a resolução definida é de 0.1, serão computadas e testadas 32 fatias durante todo o processo, respectivamente a fatia à altura de 13.6 e o bloco de fatias entre as alturas 8.9 e 5.9 inclusive com distanciação do valor resolução. No entanto somente duas das 32 computadas serão requeridas e essenciais para uma correcta detecção de colisões, nomeadamente a primeira (h=13,6) e última fatia (h=5,9). Todas as outras fatias e correspondestes valores guardados não acarretam qualquer informação adicional relevante para uma correcta detecção de colisões.

4.3.2 | Avaliação das Fatias

Uma fatia é usada por um avatar para executar a detecção de colisões quando esta é a primeira a situar-se logo acima da sua cabeça. Considerando uma fatia em particular, se a diferença entre a altura a que esta fatia foi tomada e a altura máxima guardada no mapa de alturas das suas células for menor à altura do avatar, a célula nunca será usada para teste. Este é o caso da segunda fatia na Figura 38, onde a altura guardada é na realidade igual à altura tomada da fatia. Todos os outros valores desta segunda fatia são iguais aos correspondentes na terceira fatia, consequentemente a segunda fatia e os correspondentes valores de alturas não acarretam nenhuma nova informação, podendo assim serem excluídos. Este raciocínio deixa-nos no final com unicamente 2 fatias úteis (Figura 38), a primeira tomada a 13.6 unidades e seguinte a 5.9 unidades de altura.

O exemplo da Figura 39 mostra outro caso onde algumas fatias podem ser ignoradas ou não incluídas no teste de colisões. Consideremos as três fatias presentes na Figura 39. A fatia tomada a 7.9 unidades de altura pode ser descartada, já que toda a informação contida está também presente na 1ª ou 3ª fatia, excepto o valor 7.9 que poderá ser considerado inútil, como se verá de seguida. As cruzes em cada fatia (Figura 40) indicam as células que nunca serão usadas na detecção de colisões devido ao avatar ter a sua cabeça sempre acima destas posições em particular, onde a célula com o valor 7.9 que servia de excepção antes, é agora também descartada. Ou seja, e nas condições presentes na cena da Figura 39, a possível fatia reguladora de navegação tomada a uma altura de 7.9 poderá em todas as condições de navegação ser substituída pela seguinte de maior altura (13.6).

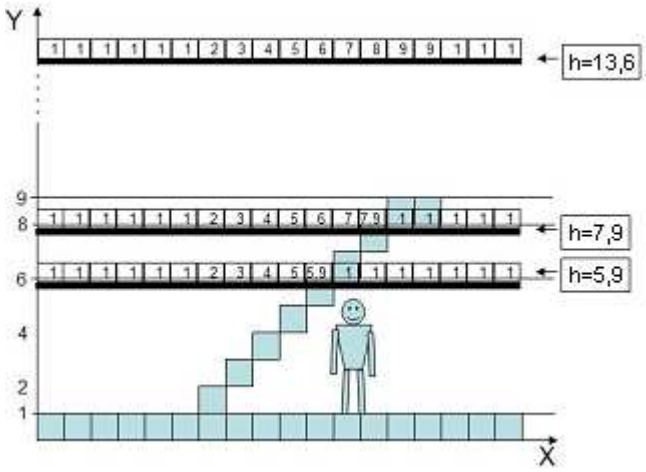


Figura 39 – Exemplo com escadas

Figura 40 – Camadas retiradas do exemplo anterior com definição de células descartadas

Estão identificadas assim as duas situações onde uma dada fatia poderá ser descartada. O algoritmo que detecta estas situações e descarta as fatias desnecessárias é apresentado no Algoritmo 2. O algoritmo é chamado cada vez que uma nova camada é computada, e executa o teste de conteúdo relevante da nova fatia processada.

Na primeira situação de descarte de fatias computadas (relativa ao caso da Figura 38) e identificada no Algoritmo 2 são processadas e contabilizadas nas linhas 13 e 14 como descartadas (*DISMISS_CURRENT_SLICE*) todas as fatias, quando o valor de altura de todas as células dessa fatia igualar a soma das iguais na fatia anterior (*Equal*), mais aquelas consideradas inúteis devido à sua impossibilidade de actuação no processo (*Useless*), este processamento situa-se desde a linha 5 à 10.
Na segunda situação de substituição e descarte de fatias computadas (relativa aos casos das Figura 39 e Figura 40) e identificadas no Algoritmo 2 são processadas como substituíveis aquelas que nunca chegarão a ser usadas por não acrescentarem valor acrescentado (linhas 20,22 e 23) e contabilizada a troca destas pela informação da fatia posterior quando válida à sua substituição (linhas 17,18 e 19).

Resumindo, se este teste não achar conteúdo relevante, então a fatia será descartada. Se a fatia contiver informação nova, então o algoritmo testa se a fatia anterior poderá ser substituída pela fatia actual.

Algoritmo 2

```
Let slices be an array of slices taken so far
Let sliceHeight be an array of the heights slice´s we´re taken
Let avatarHeight be the height of the avatar
Define INVALID as being below minY

1 testSlice(i) {
2 ♦  auxSlice = new Slice();
3 ♦  countEqual = 0;
4 ♦  countUseless = 0;
5 ♦  for every cell a in slice[i] {
6       ♦  if (sliceHeight[i] - slice[i][a] < avatarHeight)
7             ♦  countUseless ++;
8       ♦  else if (slice[i][a] == slice[i-1][a]) {
9             ♦  auxSlice[a] = INVALID;
10            ♦  countEqual ++;
11      ♦  }
12 ♦  }
13 ♦  if (countEqual + countUseless == number of cells in slice)
14       ♦  return (DISMISS_CURRENT_SLICE)
15 ♦  // are there more then two slices?
16 ♦  if (i>1) {
17       ♦  for every cell a in slice[i] {
18          ♦  if ((auxSlice[a] == INVALID) || (slice[i-2] [a] == slice[i-1][a])
19                || (sliceHeight[i-1] - slice [i-1][a] < avatarHeigth))
20          ♦  count ++;
21       ♦  }
22       ♦  if (count == number of cells in slice)
23          ♦  return (REPLACE_PREVIOUS_SLICE)
24 ♦  }
25 ♦  return(KEEP_SLICES);
26 }
```

Algoritmo 2 – Função de teste às fatias computadas

4.3.3 | Fatiação

No Algoritmo 3 é detalhado todo o processo de fatiação, assumindo que a *bounding box* foi estabelecida e o Y varia entre maxY e minY.

Algoritmo 3

```
1 Slice_World(maxY, minY) {
2 ♦ sliceHeight[0] = maxY + avatarHeight + resolution;
3 ♦ slice[0] = ComputeSlice(sliceHeight[0]);
4 ♦ max = maximum(slice[0]);
5 ♦ i=1;
6 ♦ while (max > minY + avatarHeight) {
7       ♦ sliceHeight[i] = max – resolution;
8       ♦ slice[i] = ComputeSlice(sliceHeight[i]);
9       ♦ max = maximum(slice[i]);
10      ♦ test = testSlice(i);
11      ♦ if (test == KEEP_SLICES)
12            ♦ i++;
13      ♦ else if (test == REPLACE_PREVIOUS_SLICE) {
14            ♦ slice[i-1] = slice[i];
15            ♦ sliceHeight [i-1] = sliceHeight[i];
16      ♦ }
17 ♦ }
18 }
```

Algoritmo 3 – Algoritmo de Fatiação

A Figura 41 exemplifica o processo de fatiação de uma outra determinada cena de forma a demonstrar algumas situações com enquadramento no Algoritmo 3. Serão exemplificados alguns procedimento da fatiação na relação directa com as respectivas linhas do correspondente algoritmo.

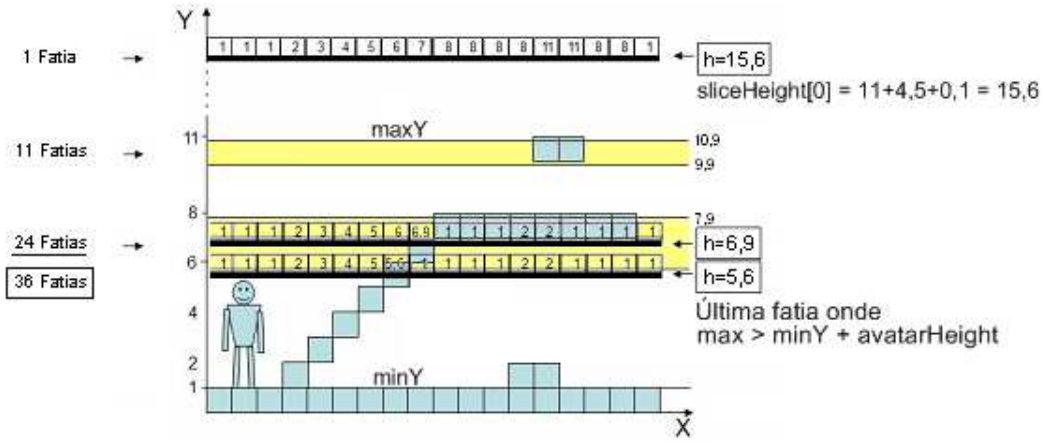


Figura 41 – Cena exemplificativa de apoio ao Algoritmo 3

Assim vemos que o processo de fatiação se desenvolverá entre a 1ª fatia a um valor de altura = 15.6 (*maxY + avatarHeight + resolution*) e um valor de altura mínimo = 5.6 (*minY + avatarHeight + resolution*).

A primeira fatia a guardar será sempre a de altura máxima (neste caso a 15.6). Durante o seguinte bloco de fatiação descendente (linha 6) serão computadas fatias sucessivas, mas unicamente em zonas onde exista verdadeira probabilidade de encontrar fatias que acarretem informação útil à navegação. Estas zonas estão demarcadas a cinzento e situam-se entre os valores de altura 10.9 até 9.9, e entre os valores de altura 7.9 até 5.6 (valor de altura mínimo de fatiação). Durante este processo serão computadas fatias contínuas com diferença de alturas no valor da resolução definida (linha 7). O que gerará 36 fatias computadas para a cena da Figura 41.
Durante a computação de cada uma destas 36 fatias é realizado um teste de validade às mesmas (*testSlice(i)*) fornecido pelo Algotitmo2, onde se decidirá pelo seu descarte (linha 11 e 13), manutenção (linha 11 e 12), ou então pela sua substituição pela fatia computada posteriormente (linhas 13, 14 e 15).
No final, as fatias guardadas e consideradas relevantes para uma correcta navegação nas condições presentes na Figura 41, serão somente 3 (às alturas de 15.6, 6.9 e 5.6). De realçar que nesta situação a fatia a 6.9 de altura será indispensável devido à possibilidade do avatar poder movimentar-se em locais (os dois escalões na base do mundo) onde as informações desta fatia serão diferentes dos valores da fatia logo acima (15.6), não permitindo assim o seu descarte.

4.3.4 Detalhes da Implementação

Como mencionado anteriormente, o método executa o *rendering* do mapa de profundidades, sendo transformado num mapa de alturas organizado em fatias, que não são mais que matrizes de células, onde cada célula corresponde a uma altura num determinado píxel, e respectivamente um determinado espaço virtual a simular.

Efectuar o *rendering* do mapa de profundidades para os objectivos propostos inicialmente é conceptualmente uma ideia sadia de modo a obter em altura um mapeamento de coordenadas espaciais, no entanto, na prática, alguns problemas poderão figurar. Estes problemas estão relacionados com o modo como as primitivas gráficas são usadas para a modelação da cena, bem como a resolução do z-Buffer utilizado.

É comum modelar uma parede através do uso de triângulos verticais. Estes triângulos são perpendiculares ao Near Plane e Far Plane, resultando daí uma área de projecção

igual a zero. Assim, por norma este tipo de polígonos não são renderizados, e por conseguinte a altura destes polígonos não é guardada no z-Buffer. Considerando a cena 3D apresentada na Figura 32, quando tomada uma fatia abaixo do terceiro nível, são obtidos resultados diferentes se adicionar-mos a opção de apresentar as linhas no topo de todos os polígonos. A Figura 42-A mostra o mapa de profundidades (mais escuro significa maior altura) obtido quando utilizamos um *rendering* dos polígonos em modo "*fill mode*", e a Figura 42-B mostra para a mesma cena o mapa de profundidades quando são desenhadas as linhas no topo de cada polígono renderizado (nota: as figuras foram alteradas ao nível do contraste de modo a ilustrar como maior clareza as suas diferenças).

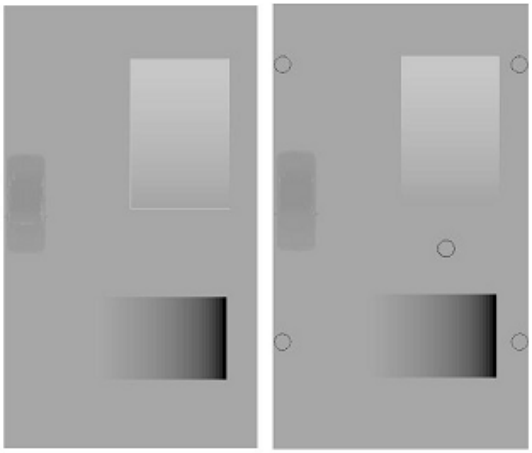


Figura 42 – Mapas de Profundidades (esquerda: polígonos em modo "*fill*"; direita: polígonos em modo "*lines*" impostas no topo do corte)

As diferenças entre as figuras A e B situam-se ao nível das colunas presentes na cena. Estas estão representadas por polígonos verticais que não entram no *rendering* quando usamos o comando OpenGl "*fill mode*". Impor a apresentação das linhas no topo dos polígonos produz a visibilidade dos contornos das colunas e a sua altura a este nível, o que é um passo em direcção ao objectivo pretendido. No entanto quando convertemos o mapa de profundidades em alturas, poderão verificar-se a existência de erros entre a altura real e o valor de altura guardado no mapa de alturas nos píxeis relacionados com a geometria de algumas estruturas presentes (neste caso das colunas), devido principalmente à resolução no nosso z-Buffer e os possiveis diferentes posicionamentos do Near Plane em relação ao Far Plane.

Tomando em linha de conta a resolução actual do z-Buffer, tipicamente limitada a 24 ou 32 bits, sendo não linear, poderão aparecer algumas discrepâncias aquando a leitura de valores de alturas tomados com por exemplo diferentes possíveis posições do Near Plane. Por exemplo, um mesmo ponto 3D medido com diferentes Near Planes poderá apresentar diferentes alturas no z-Buffer. Se considerarmos um z-Buffer de 24 bits, esta diferença no geral será muito pequena, e poderá ser contabilizada usando um valor de erro consentido (*threshold*) quando testamos igualdades entre estes valores. Outro dos factores de influência para esta discrepância de valores medidos é a correspondência entre o posicionamento do Near Plane em relação ao Far Plane.

A solução encontrada foi o recurso à utilização de Clip Planes (Figura 43) que permite superar esta e outra característica específica do z-Buffer: a não linearidade juntamente com os possíveis erros de arredondamento. Quando consideramos o mapa de profundidade para duas fatias tomadas com dois diferentes valores de Z-Near, os valores da profundidade guardada para o mesmo píxel, poderão ser diferentes devido à não linearidade do z-Buffer. Com o recurso á utilização de Clip Planes e mantendo fixos os valores nos planos Z-Near e Z-Far, conseguimos garantir que para os mesmos píxeis, as alturas guardadas nas diferentes fatias serão sempre as mesmas.

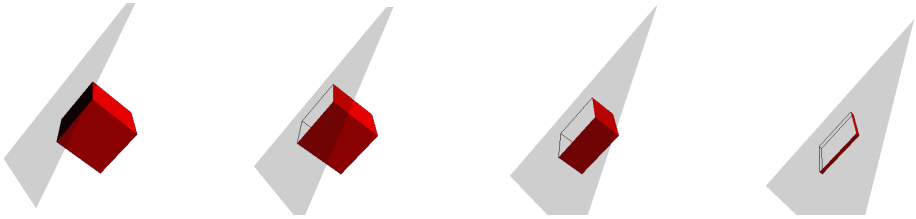


Figura 43 – Segmentação com reconstituição do polígono dada pela utilização de vários Clip-Planes

Uma outra característica relevante tem a ver com a precisão do mapa de alturas, e consequentemente memória requerida por fatia. Se consideramos um mundo onde cada unidade seja correspondente a um centímetro, então um mapa de alturas de 16 bits permitir-nos-á lidar com cenas até 655.36 metros. Seria assim possível, efectuar detecção de colisões até esta altura com erros de até 1 centímetro. Isto seria suficiente para cobrir o actual maior edifício no mundo: A Torre Taipei 101, com 509 metros. No entanto podemos ainda considerar que por vezes e em algumas situações particulares este centímetro de erro possa ainda ser excessivo. Uma solução possível

seria guardar a diferença de alturas ou profundidades nas fatias, em vez da altura actual. A altura a que cada fatia é tomada pode ser usada assim de uma forma mais precisa e abrangente, mantendo os valores de alturas para cada píxel de profundidade. Em tempo real somente seria requerido uma simples operação de subtracção extra. Este método é assim consequentemente não limitativo para valores de precisão, mesmo considerando a guarda de valores de alturas utilizando outras resoluções do z-Buffer.

Em jeito de análise crítica poderá ainda haver implicações menos precisas quando usamos um valor de resolução de corte limitado. Isto poderá suceder quando a altura/profundidade actual requererem maior precisão do que aquela disponível inicialmente. Estes valores poderão ser considerados inválidos, e uma fatia extra adicionada de modo a garantir que em qualquer situação onde exista uma área navegável por um avatar, exista uma fatia a uma altura apropriada para estabelecer a correcta ajuda à navegação. Se a variação de alturas é suficientemente pequena tal que o valor da precisão não é um valor aceitável, um bit adicional poderá ser usado como indicador de célula ocupada ou não pelo avatar. Um array destes bits poderá ser considerado ainda para colisões inter avatares, como se verá demonstrado adiante.

4.4 Preparação, Experiências e Testes

4.4.1 Importação dos Modelos Representando o Espaço e as Personagens Povoadoras

Existem muitos "importadores" para OpenGL capazes de carregar diferentes formatos de ficheiros 3D. Talvez os de uso mais corrente sejam os importadores de formatos 3DS e OBJ. Os 3DS porque se tornaram num formato mais ou menos standard no mundo da computação gráfica devido à tradição e sucesso da sua aplicação raiz (3DStudio). Os de formato OBJ, também muito utilizados devido ao conteúdo se encontrar em 2 ficheiros de texto "limpo", um contendo informação visível da localização dos vértices, faces e polígonos no espaço dos diversos objectos do modelo 3D, gerando a estrutura do mesmo (*.obj); e outro complementar com informação sobre os materiais, luzes e cores a associados ao primeiro (*.mtl).

Na importação de modelos representando o espaço a povoar (ambiente de simulação e navegação), foi decidido utilizar o código importador de ficheiros de utilização livre OBJ de Nate Robins[15] (GLM), devido às especificidades deste tipo de ficheiros descritos anteriormente.

O GLM é uma biblioteca escrita em C++ e OpenGl, capaz de ler, escrever e manipular ficheiros Wavefront-obj. É composta pelos ficheiros glm.c e glm.h. Inclui rotinas para leitura e escrita de ficheiros obj e mtl, controlo de suavidade na apresentação de objectos, controlo das dimensões e escala da cena, cálculo das normais em parte ou totalidade da cena, harmonização de texturas, ligação ou exclusão de vértices próximos ou redundantes respectivamente, geração de projecções específicas (mapa planar ou esfera), entre outras.

A biblioteca de importação de ficheiros Wavefront-obj de Nate Robins foi sensivelmente alterada no trabalho inicial que dá origem à investigação neste capítulo, de forma a estabelecer e definir a posição "inaugural" dos agentes na base do mundo importado e também adequar este mundo importado às proporções dos agentes, entre outras (escala e posição, smooth, material, polígonos, etc).

Para o caso da importação dos modelos representando as personagens virtuais foi utilizada a biblioteca de animação Cal3D [142] de Bruno Heidelberger (já referida anteriormente), que representa os níveis de abstracção no âmbito geométrico e cinemático das personagens virtuais c/ comportamento aqui utilizadas. Esta biblioteca anima um modelo/personagem, recorrendo à sua associação a um esqueleto virtual. Neste método de animação, uma acção como o andar é levada a cabo deslocando o conjunto de triângulos associados aos "ossos" em questão, neste caso aqueles formando as pernas. De modo semelhante ao esqueleto humano, o movimento das pernas leva ao movimento adequado da pélvis e dos braços criando uma animação credível de uma personagem em movimento.

O Cal3D é independente da plataforma e não está refém de nenhuma API gráfica específica. Algumas das suas características, que levaram à sua escolha e utilização são:

- Animação baseada em esqueleto;

[15] nate@pobox.com, http://www.pobox.com/~nate

- Animação concorrente e fluida, i.e: uma dada personagem encontra-se a andar e em dado momento levanta o braço (para saudar alguém que passa) sem que isso o impeça de continuar a andar;
- Meshes progressivas para diferentes níveis de detalhe (LOD);
- Sistema de simulação de vestuário e cabelo (aumenta a credibilidade);
- Plugins de exportação (3D Studio Max, Milkshape 3D entre outros);
- API simples e "limpa" (C++ e C);
- Software livre, seguindo a licença GNU LGPL[16].

O conjunto de acções ou animações desempenhadas por cada modelo, é definido num ficheiro de configuração, que guarda e estrutura a informação física do avatar, o esqueleto, as animações, as *meshes* e os materiais/texturas. As várias animações possíveis são disponibilizadas, definidas e utilizadas em scripts criados pelo programador/utilizador.

As personagens virtuais do Cal3D incluídas na aplicação terão de manter e actualizar em cada instante características comuns a todas elas, as quais irão ser adquiridas através dos mecanismos aqui apresentados, tais como:

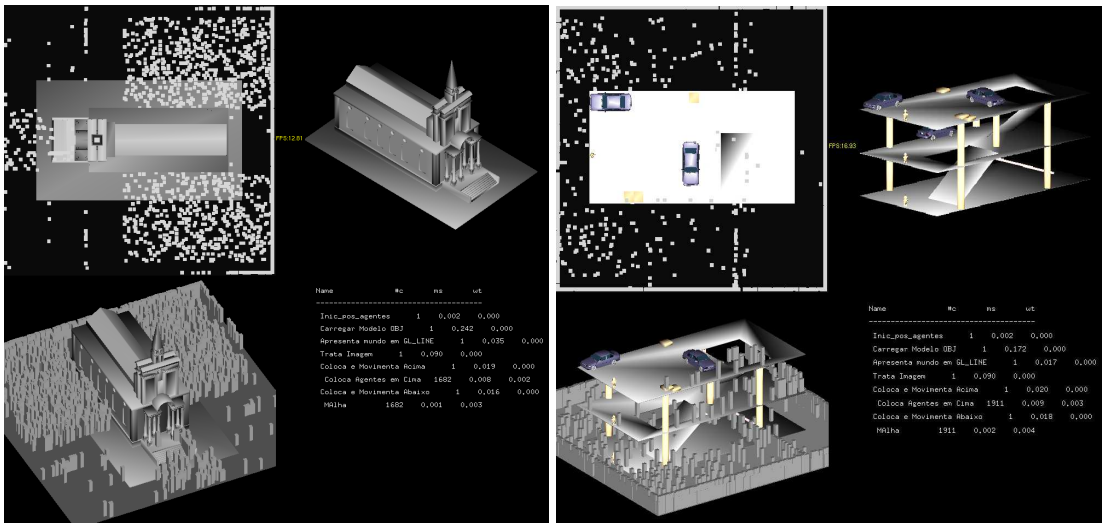
- A posição no mundo em determinado instante;
- A orientação no mundo, i.e: em que direcção se encontra orientado;
- Informação relativa ao destino que pretende atingir;
- Colisão com o ambiente e outros avatares;
- Conjunto de acções realizáveis pelo avatar. Este conjunto depende da biblioteca de animação utilizada, neste caso do Cal3D;
- O próprio modelo 3D. O conjunto de pontos, triângulos e texturas que formam o avatar.

Foram também definidas áreas de visualização (*viewports*) onde será apresentada a simulação nos mundos 3D importados (Figura 44). Assim foi criada uma vista superior ao modelo importado (*viewport1 - superior esquerda*), outra de vista lateral (*viewport2 - inferior esquerda*), com opção de rotação e zoom sobre o modelo, e mais duas (*viewport3 e 4 - superior e inferior direita)* com informação diversa sobre os modelos importados, e especificações da simulação em tempo real respectivamente. A definição inicial proposta para cada uma foi de 512*h* x 512*w,* que se achou

[16] GNU Lesser General Public License

adequada para os primeiros modelos de animação testados. No entanto, o código foi estruturado de forma a permitir alterar facilmente esta definição, desde a janela inicial de carregamentos de definições da animação.

De modo a poder proporcionar uma animação mais personalizada em relação ao *display* nas *viewports* definidas, foi implementado um algoritmo que fomenta a interacção dos utilizadores através de algumas teclas e botões especiais que provoca alterações principalmente na vista lateral (inferior esquerda) – *viewport*2 (incluída na Figura 44 A e B), tais como: rotação, translação da câmara ou modelo, e zoom da área a visualizar.



A B

Figura 44 – Apresentação final das *viewports* : vista superior e lateral rotativa/zoom da animação, vista isolada do modelo importado e dos valores de algumas tarefas desempenhadas (*profile*)

Existia também o interesse de poder guardar as animações apresentadas, quer em imagens isoladas ou sequenciais (vídeo) num formato que mantenha a qualidade mas com taxas de compressão aceitáveis, como por exemplo o formato JPEG.

Assim, na captura de imagens estáticas de diferentes formatos e qualidade foi utilizada a biblioteca OPenIL - DevIL [143], de licença livre seguindo a norma GNU LGPL. A Developer's Image Library (DevIL) é uma biblioteca desenvolvida para ser utilizada por programadores na construção de aplicações que necessitem uma grande capacidade de carregar e guardar imagens, é de fácil integração em código externo e de fácil aprendizagem. O controlo final das imagens é deixado ao programador

podendo ler, guardar, converter, manipular, filtrar e apresentar uma grande variedade dos formatos mais utilizados para gravar e compactar imagens.

4.4.2 | Testes Experimentais

Foram efectuados experiências e testes em vários mundos 3D. Destacam-se aqui em pormenor a cena da garagem 3D (ver Figura 32) de modo a ilustrar o conceito, e o modelo de uma "*Power-Plant*" (Figura 45- modelo disponível em http://www.cs.unc.edu/~geom/Powerplant/) para demonstrar a sua aplicabilidade para cenas muito complexas, cada vez mais um standard nos dias de hoje.

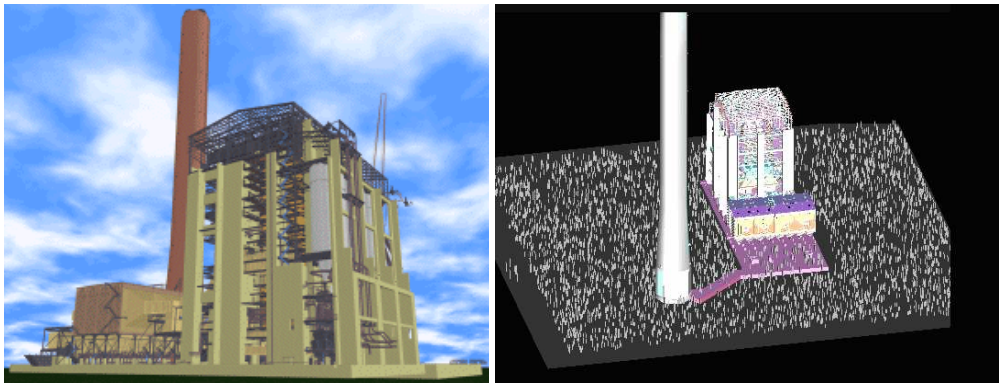


Figura 45 – PowerPlant Model: 13 milhões de triângulos

As fatias tomadas para a cena da Garagem 3D são apresentadas na Figura 46. O tempo de pré processamento requerido para a fatiação de toda a cena (86 fatias computadas) e para o descarte e eliminação das fatias consideradas não necessárias foi menor a um segundo para *viewports* de 256x256 (ver tabela de tempos de fatiação em segundos apresentado na seguinte secção de resultados e testes).

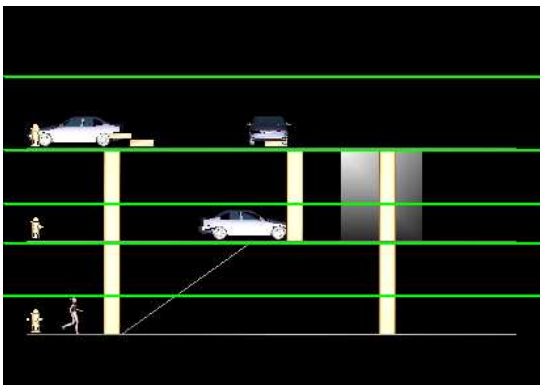


Figura 46 – Fatias obtidas e guardadas na cena da Garagem

A utilização deste método permite-nos também incorporar à animação agentes de alturas diferentes. Estes poderão ter situações de navegação divergente, devido a algumas possibilidades ou limitações, inerentes à sua altura. Como se comprova pela Figura 47, agentes de alturas inferiores poderão caminhar por locais onde outros agentes de alturas superior poderão colidir (neste exemplo uma barra colocada a uma determinada altura, impede a passagem de alguns agentes mais altos, deixando passar por debaixo, outros de altura inferior).

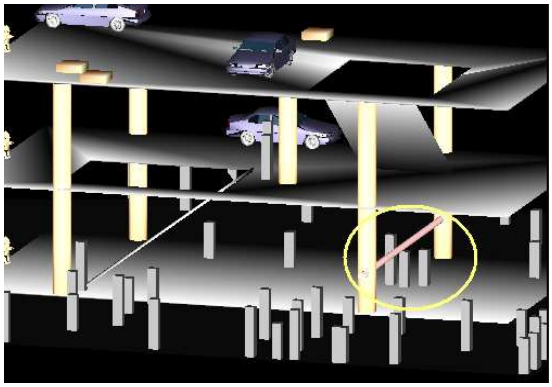
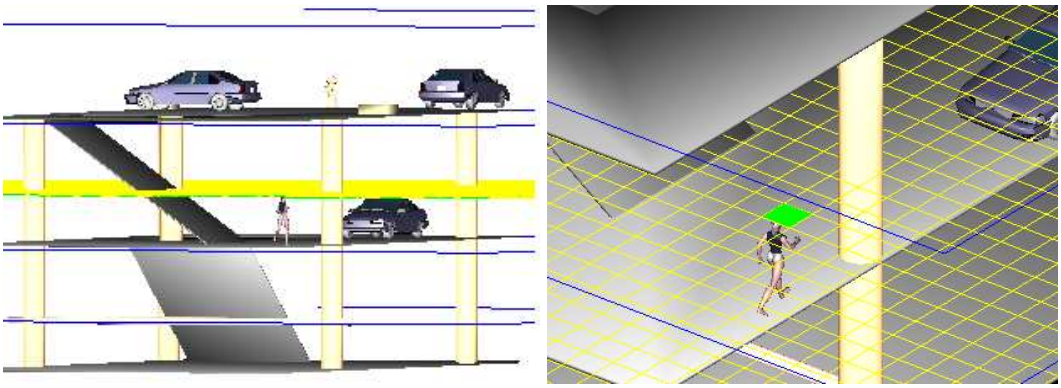


Figura 47 – Colisão ou não dependendo da altura dos agentes

Para poder monitorizar ou comparar a velocidade/performance real da animação gerada com ou sem testes de colisão, e para os diversos mundos 3D e personagens virtuais carregadas, é possível visualizar em tempo real o número de frames por segundo que ocorre a cada instante para cada um destes casos na animação.

De modo a obter um quadro claro da distribuição temporal das tarefas desempenhadas no pré-carregamento e durante a simulação, foi incorporada uma classe "*profiling*" capaz de devolver respostas rápidas e localizadas sobre o desempenho dos diferentes módulos que compõem a aplicação, por exemplo: quais as funções onde se gasta mais tempo, quais as funções que são invocadas mais vezes, quais os picos de congestionamento, etc. Permite ainda descobrir situações que dificilmente seriam detectáveis sem esta diferenciação sectorial. Esta utilização/implementação de um *profiler* trás as vantagens de facilidade de gestão da performance de todo e parte do sistema, onde num *display* em tempo real se vão percebendo os consumos dos recursos ao longo da animação. É este o tipo de informação apresentado em tempo real na *viewport* 4 da Figura 44.

Outra funcionalidade adicionada à animação foi a de proporcionar ainda a visualização de uma malha de células que representa visualmente e em 3D a matriz de células em actual navegação. Isto passou por incluir uma classe demonstrativa da navegação dos agentes e correspondente detecção de colisões, composta por células (com tamanho maior ao realmente utilizado) que variam entre: Verde – Livre, Amarelo – Degrau ultrapassável, e Vermelho – Colisão (Figura 48-B).
Ainda nesta visualização, outra implementação foi a identificação do mesmo género e ao mesmo tempo de uma classe demonstrativa da fatia actual de teste de alturas (identificada na Figura 48-A).



A B

Figura 48 – Malhas visíveis adicionais para visualizar a aplicação do método utilizado para identificar colisões.

4.4.3 Resultados dos Testes

Foram implementados testes para avaliar a performance na fase de fatiação e posterior detecção de colisões. A cena da Garagem 3D levou 10 milissegundos a efectuar o *rendering*. Os resultados estão apresentados na Tabela 2. Para propósitos referenciais, o tempo tomado para desenhar um avatar (representado graficamente nestas condições por uma simples caixa) é também presente nos tempos anunciados. O número de avatares testados situou-se entre os 500 e um milhão de avatares. Como se pode comprovar, o método escala de forma linear de acordo com o número de avatares apresentados, como seria de esperar. Também de notar, que o tempo de mover um avatar inclui também a decisão da nova direcção a ser tomada em caso de colisão, e o respectivo teste da nova ocupação. Nesta altura, o algoritmo que toma a decisão sobre qual a nova direcção quando uma colisão ocorre, é uma simples escolha

aleatória entre as opções esquerda, direita, frente, atrás. No capítulo seguinte serão apresentados algoritmos mais "inteligentes" para este tipo de tarefa.

Nr de Avatares	Desenhar Avatares	Mover Avatares
500	2	2
1.000	5	4
1.500	7	6
2.000	9	8
5.000	23	20
10.000	48	39
100.000	467	395
1.000.000	4.780	3.990

Tabela 2 – Performance resultante da detecção de colisão na cena da garagem (tempo em milisegundos)

Os testes foram executados num PC com as seguintes características:

CPU: - Intel Pentium4 3 Gz
RAM: - 1,5 Gb Ram
GPU: - NVIDIA - GeForce 5950 Ultra 256 Mb

Foram submetidas a teste outro tipo de cenas, para apurar o número de fatias tomadas e guardadas, nomeadamente: Um cubo, a garagem (vista anteriormente), um edifício de uma igreja, e a *Power-Plant* (*from the Walkthru Project at Stanford University*). A Tabela 3, mostra o número de fatias tomado inicialmente, e os realmente guardados para cada um destes mundos 3D. Todos os testes assumem um avatar com um equivalente a 1,75 metros, e uma resolução de 10 cm.

Cena 3D	Computadas	Guardadas
Cubo	42	2
Garagem	86	5
Igreja	159	7
Power-Plant	835	85

Tabela 3 – Fatias computadas e guardadas

O tempo de fatiação destes mundos tendo em conta as características do PC utilizado para o efeito é apresentado em segundos na Tabela 4. Este tempo incorpora a computação das fatias, e a sua análise para aceitação, descarte ou substituição, e a preparação das mesmas à ajuda de navegação.

Cena 3D	Viewports	Tempo de Fatiação
Cubo 3D	256 x 256	0.4
	512 x 512	2.1
Garagem	256 x 256	0.9
	512 x 512	5.6
Igreja	256 x 256	2.3
	512 x 512	12.3
Power-Plant	256 x 256	192
	512 x 512	1060

Tabela 4 – Tempo de fatiação dos mundos 3D em segundos

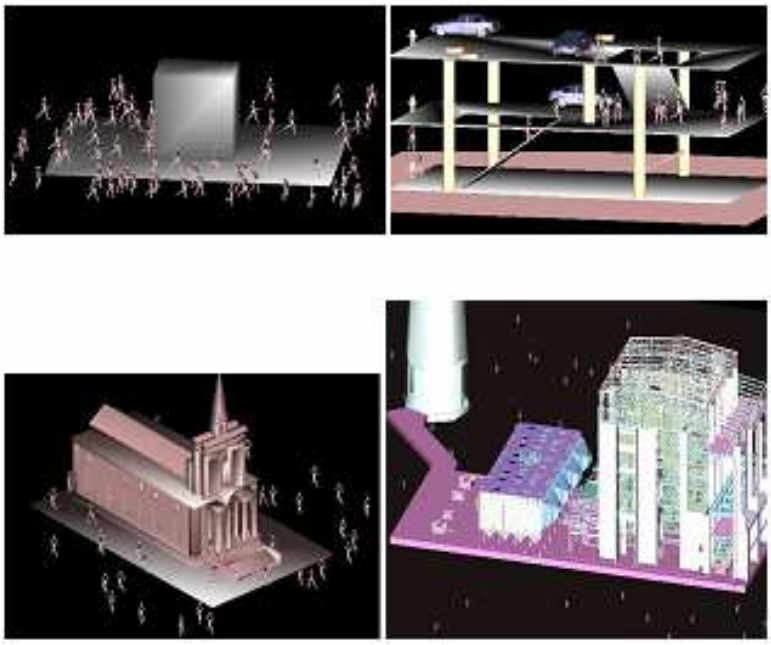


Figura 49 – Ambiente de testes: Cubo, Garagem, Power-Plant e Igreja (No sentido dos ponteiros do relógio desde a imagem superior esquerda)

Todos os testes e resultados subjacentes a este capítulo, foram contidos e apresentados no artigo [134], em Junho de 2006 na SIACG'2006 – 3rd Ibero-American Symposium in Computer Graphics.

CAPÍTULO 5

Planeamento de Rotas em Multi-nível para Navegação 3D Real

Nos dias que correm, os ambientes virtuais são cada vez mais preenchidos com humanos virtuais, como uma componente intrínseca e indissolúvel, procurando assim um maior realismo e atractividade da animação. Estes humanos virtuais (sejam considerados avatars ou agentes) poderão incorporar várias capacidades ou habilidades (*skills*), que quando mescladas, permitirão produzir comportamentos mais complexos e credíveis. Entre estes comportamentos, a capacidade de navegação num mundo virtual revela-se de uma enorme importância. A reprodução deste comportamento fundamental, requer o endereçamento para a investigação de tópicos como o estudo e análise da topologia do modelo, do ambiente envolvente, técnicas de detecção de colisões, e estratégias de planeamento e procura de caminhos ou rotas de navegação. De modo a poder alcançar estes objectivos, terá de haver uma aprendizagem do contexto a povoar, e a sua propensão à navegação usando modelos de interacção apropriados.

Os vários modelos de localização e planeamento de rotas de navegação já considerados no estado da arte, recorrem ao método de construção (em tempo real ou não) de trajectórias de movimentação de personagens, conhecido como *pathfinder*. Esta é uma técnica recorrente em IA para fornecer possíveis alternativas de rotas entre dois pontos, gerando um mapa bidimensional ou tridimensional das soluções encontradas no ambiente de navegação. Pode ser aplicada a uma personagem singular ou a um conjunto destas que poderão agregar objectivos comuns (um exército em locomoção de um ponto a outro, por exemplo). O curso fornecido por este conjunto de heurísticas poderá nem sempre ser o melhor trajecto, dependendo localmente do significado atribuído à palavra "melhor". Um dos problemas que poderá atingir a relatividade desta questão, poderá por exemplo ser a

necessidade imediata de respostas num jogo de acção ou estratégia. Outro problema na génese da complexidade e qualidade das respostas de navegabilidade entre pontos durante um determinado percurso, recai sobretudo em aspectos da simulação utilizando ambientes sem a prévia análise da sua geometria e especificação de zona navegáveis.

Para se alcançar uma etapa final eficiente de localização de rotas (*path-finding*) sobre um determinado modelo, muita da investigação e "*intelligence*" terá de ser centrada no anterior planeamento (*path-planning*) do percurso dadas as condições do "teatro de operações". Estas duas etapas estão intimamente ligadas, sendo que o sucesso e eficiência da última está dependente da eficácia da primeira. Este planeamento terá de focalizar esforços na aprendizagem do contexto a povoar, proporcionando um estádio de preparação transparente para o utilizador, e essencial para a futura tomada de decisões de navegação independente ou colectiva. Inovar nesta fase (centrada sobretudo na circunstância da aprendizagem do contexto ambiental) é essencial para obter resultados também inovadores nas etapas seguintes, originando uma base de metodologias e informação que definirão a qualidade na área da navegação autónoma.

Ainda neste enquadramento, a qualidade de navegabilidade a gerar não se mede simplesmente pelas possibilidades abertas e factíveis de atingir, rege-se pelas boas práticas, pela eficiência e escalabilidade a demonstrar, pela aplicação de técnicas conjuntas de inteligência artificial e computação gráfica em sistemas interactivos de simulação autónoma, pela adequação dos algoritmos às necessidades requeridas, e pela capacidade e qualidade de apresentação de todas estas componentes da simulação em tempo real com um elevado grau de independência face a muitas condicionantes. Este grau de independência abrange por exemplo, poder utilizar qualquer geometria de um modelo 3D como base da simulação comportando objectos virtuais em movimento (representando por exemplo pedestres em deslocação), assim como ter a capacidade de explorar todos os sectores espaciais potencialmente acessíveis no modelo para cada tipo de personagens, tentando promover de forma idêntica ao que deparamos na realidade, a deslocação com possibilidades de movimentação mais alargadas, incorporando também estruturas admitindo vários níveis de altura, representativas normalmente de interiores de edifícios, túneis, ou pontes, onde a navegação deixa de ser restrita a um só plano.

A possibilidade de agentes autónomos poderem lidar com os seus intentos independentes de posicionamento e movimentação na estrutura de navegação, é uma das aptidões mais importantes em termos comportamentais. Esta pretensão, traduzida em desenvolvimentos na área de *path planning/finding*, projecta a incorporação de múltiplas decomposições de procura nos espaços navegáveis, associando para o facto, normalmente um grafo hierárquico como um meio de mais eficientemente gerir todo este processo. Este tipo de implementação está bem documentado na já vasta literatura direccionada a este foco de investigação. No entanto, devido à quase total inexistência de um método capaz de mapear automaticamente ambientes 3D de forma escalável, não foram consecutivamente e naturalmente aplicadas estas técnicas para navegação multi-nível. Os artigos bem documentados que suportam esta ideia, por exemplo os estudos de Pettré ou Steed [3, 82, 86] apresentados anteriormente, produzem grafos com um número muito elevado de nodos, muitas das vezes de gestão incomportável para apresentações em tempo real de simulação em ambientes mais complexos, como é reconhecido pelos próprios autores, o que limita de modo constrangedor, a planificação a mundos mais simplistas, com condições contidas em termos de complexidade e navegabilidade multi-nível, em resumo, a mundos praticamente planares, como os que anteriormente haviam sido exclusivamente considerados.

No desenvolvimento deste capítulo, o modelo proposto e apresentado nos testes de aplicação e eficiência de *path planning* tridimensional, apresenta de uma forma inovadora o planeamento e configuração das deslocações através de uma granularidade de pesquisa desigual no tratamento da informação de localização espacial, distinguindo os diferentes planos de navegação bidimensional encontrados e as suas transições, e baseando a navegação tanto no teste singular das células que compõem a malha multí-nível, como também na localização dos nodos (composição de células) em relação ao tipo de espaço do ambiente em que o avatar se apresenta ou pretende alcançar. Ou seja, e tomando o exemplo do planeamento e mapeamento para uma liberdade de navegação orientada a um mundo 3D, que poderia ser aqui exemplificado por um edifício de habitação real com vários pisos como aqueles que comummente servem de habitação a milhões de pessoas nas diversas cidades, sucederia catalogar numa fase de pré-processamento, os vários andares e planos de navegação à mesma altura, e as zonas individuais de acesso e interligação entre estes, normalmente escadas ou rampas. Assim é conseguido e organizado um conjunto de nodos representativos das diferentes superfícies que podemos encontrar a diferentes alturas. A navegação neste espaço será posteriormente e a nível local controlada

pelas informações contidas nas células individuais da malha multi-nível, a nível global, o controlo será representado pelas diferentes áreas de catalogação (conjunto de células) nas diferentes alturas, e à conexão entre estas através das superfícies adjacentes com a função da sua interligação das áreas catalogadas (estrutura de ligação de nodos) possibilitando navegação virtual humana entre os vários andares.

Os nodos resultantes da hierarquização do espaço, incorporam a distinção entre zonas sensivelmente planares ou zonas em declive, onde (por norma) as segundas permitirão o acesso às várias zonas de movimentação planar. Esta configuração é perfeitamente representativa das condições principalmente interiores, que podemos verificar em edifícios reais. Assim é normal, existirem zonas planares e as respectivas superfícies em declive de ligação a estas. Existem também neste modelo de catalogação, vantagens de actuação e comportamentos implícitos que poderão ser incluídos às personagens virtuais, quando presentes nesta diferenciação de espaços. Já que em superfícies em declive, a velocidade e cansaço de navegação pode variar substancialmente dependendo da direcção em relação ao declive, assim como, é também normal nestas áreas mais exíguas de transição e passagem entre objectivos de deslocação espacial, acontecerem e se concentrarem situações de engarrafamentos e colisões inter-avatares mais recorrentes, possibilitando facilmente identificar e antecipar este tipo de constrangimentos na navegação através da possível guarda do declive em relação à área catalogada, quando se envolvem grandes quantidades de personagens em movimento. A posterior colocação de pontos guia nos limites das zonas de subdivisão de actuação diferenciadas, estabelecendo as interligações de nodos, irão ser deduzidos de forma rápida, já que para estruturas mesmo muito complexas os nodos de pesquisa limitam-se a áreas alargadas produzindo grafos pouco numerosos de nodos e respectivas interligações. A indicação de continuidade do percurso, dados por estes guias de ligação entre nodos, possibilita subdividir também as áreas e tempos de pesquisa entre a totalidade das células da grelha, o que em tempo real se demonstrou muito importante no dinamismo e flexibilidade na tomada de decisões sobre o posicionamento e rotas a tomar.

Com a utilização desta abordagem de *path planning*, poderá ser diferenciada a questão da escolha entre a simplicidade ou o detalhe entre navegação local ou navegação mais abrangente (a mais simples poderá ser a mais rápida, mas poderá não proporcionar uma navegação mais racional e realista, fazendo uso do conhecimento do espaço tridimensional para conseguir decisões de melhor qualidade).

A utilização neste modelo de técnicas de grelhas de células, promove a fácil geração da estrutura fazendo uso do hardware aquando do *rendering*, possibilitando flexibilizar a partição de modo a incrementar a pesquisa em cenas mais complexas e de maior dimensão, subdividindo processos em vários níveis de importância e custo. Conseguir maior autonomia e simplicidade neste processo de mapeamento da área navegável, passará por conseguir lidar com uma independência total da possível complexidade estrutural do ambiente 3D recorrendo a mecanismos de conversão em proporcionando imagens mais simples (*Image based*).

Também a navegação inter-agentes ganha com esta catalogação, influenciando por exemplo os comportamentos associados a avaliações desde vários e possíveis ângulos de visão associados e em confronto entre as várias superfícies dos modelos tridimensionais. Onde por exemplo dois agentes podem avaliar as suas posições e intenções no espaço multi-nível e decidir em função disso, por exemplo, qual deles deverá alterar a rota inicialmente traçada de modo a evitar uma colisão. Outro exemplo poderá ser o de um ou vários agentes poderem esperar em zonas inicias ou finais de transição entre estes espaços por outros mais atrasados, se por exemplo ambos fizerem parte do mesmo grupo, já que é reconhecido que estas zonas são altamente influenciadoras da performance subjacente às deslocações colectivas entre as personagens. Também a possibilidade de uma análise das secções mais ou menos preenchidas com pedrestes em deslocação, servindo esta informação de base decisória na alteração e re-planeamento de rotas de deslocação em tempo real, é fundamentada neste tipo de catalogação de espaços.

Resumindo e como nota preparativa à leitura dos desenvolvimentos que se seguem, foi conseguido isolar, dividir e catalogar dinamicamente em grafos e de uma forma automática todas as áreas independentes e em multi-nível, agrupando diferentes espaços e categorias de navegação bidimensional para posterior actuação em interligação, proporcionando uma plataforma capaz de suportar um número relativamente grande de agentes em navegação, tratando com eficiência e uma naturalidade superior simulações mais flexíveis. De seguida é explanado todo o processo que conduziu à obtenção de um modelo capaz de garantir as pretensões dos objectivos iniciais.

5.1 | Extracção do Grafo de Subdivisão Espacial a partir da Topologia do Modelo

Nesta secção irão ser apresentados os vários estádios que conduziram á construção de um modelo integrado e optimizado de navegação, planeamento e localização de destinos específicos em diferentes níveis de altura, numa simulação virtual 3D.

Como já referido, o modelo agora apresentado, tem por base e advêm da evolução do trabalho elaborado (e apresentado no capítulo anterior), já testado e apresentado em congresso da especialidade [134], o qual, fornece um eficiente sistema de colisões multi-nível em mundos 3D virtuais desconhecidos à partida. Este modelo realiza uma pré-análise e processamento do mundo virtual 3D, de modo a reunir condições para uma correcta navegação.

É proposto agora, uma nova vertente de desenvolvimento integrado, a navegação multinível em ambientes virtuais 3D totalmente desconhecidos, com procura de percursos e localizações de objectivos espaciais. É apresentado na Figura 50 um mundo 3D multi-nível navegável (prolongamento do utilizado anteriormente) que servirá de base para explicar e exemplificar todo o processo de *path planning/findind* aqui desenvolvido.

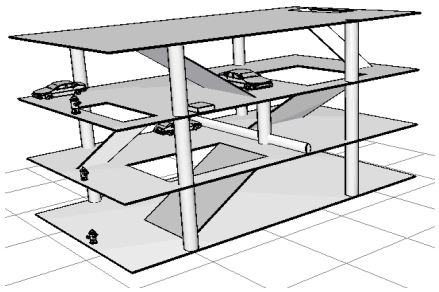


Figura 50 – Modelo 3D virtual multi-nível para testes

Neste mundo um avatar poderá navegar nos 4 andares, subir as rampas ou outros pequenos obstáculos, detectar colisões com carros ou pilares e não cair desde um nível mais elevado. Deverá ainda ser capaz de encontrar uma localização de destino desde uma qualquer localização inicial, navegando através dos vários andares, encontrando um caminho viável e possível caso exista.

Poderemos dividir em 5, as etapas de desenvolvimento deste modelo:

1. Recolha e preparação da informação.
2. Catalogação da informação utilizando algoritmos de processamento de imagem.
3. Definição e localização de zonas de interligação.
4. Construção do grafo hierárquico associado.
5. Navegação, procura e planeamento de percursos através de algoritmos apropriados.

5.1.1 Recolha e Preparação da Informação

A investigação e trabalho apresentado no capítulo anterior abarcando o processo de fatiação do mundo 3D para a detenção de colisões, não perderá sequência limitando-se só a essa tarefa. A partir da implementação deste processo, resultava desde logo, uma decomposição do mundo 3D a navegar, que poderia ser aproveitado para a geração de uma rede de espaços de navegação praticável e mais eficiente em navegação multi-nível, de modo a poder localizar neste tipo de ambiente (em sobreposição) destinos espaciais específicos.

É apresentado nas Figura 51 e Figura 52, todo este processo para o mundo 3D multi-nível em teste.

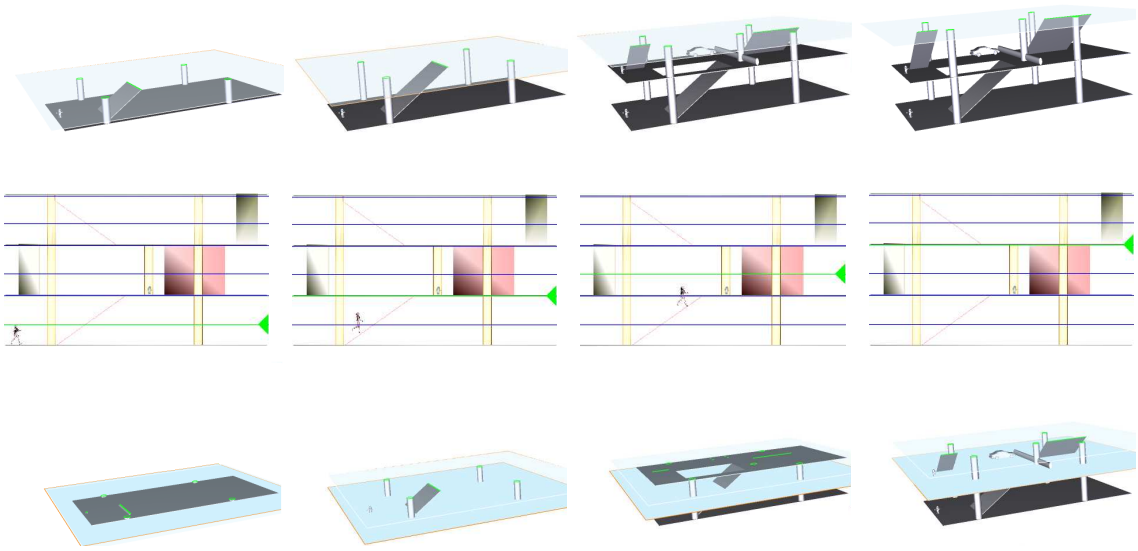


Figura 51 – Áreas computadas e realmente utilizadas para os primeiros 4 níveis de fatiação

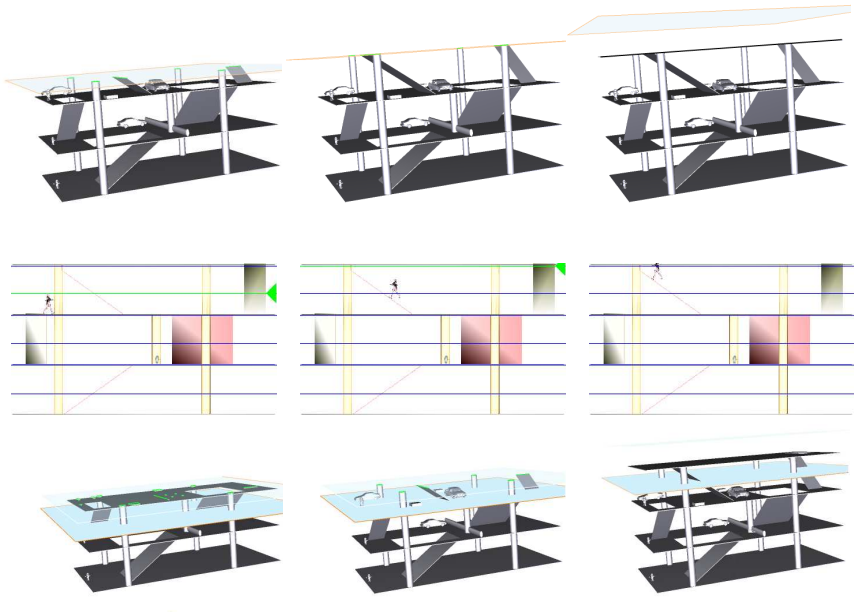


Figura 52 – Áreas computadas e realmente utilizadas para os seguintes níveis fatiação

A partir da observação das imagens anteriores, podemos visualizar as fatias de dados resultantes da etapa de fatiação (imagens superiores), a utilização em tempo real da fatia correspondente associada à navegação de um determinado avatar no mundo virtual, correspondendo num dado momento aquela que se posicione logo acima à altura da sua cabeça num dado momento (imagens intermédias), e as áreas do mundo 3D realmente necessárias (áreas entre *clip planes*) para a navegação entre transições multi-nível e respectivamente para cada posição de altura de um dado avatar (imagens inferiores).

Foram numa primeira abordagem, utilizadas as áreas directamente resultantes deste processo (imagens inferiores) de subdivisão do espaço navegável, preparando assim o terreno (*path planning*) para uma futura e correcta pesquisa de percursos (*path finding*) entre esta.

Após uma análise mais profunda, esta primeira abordagem resulta, como se comprova pelas imagens inferiores das Figuras 44 e 45, numa subdivisão não totalmente propícia a uma correcta subdivisão do espaço representativo de edifícios interiores. O termo propício não significa aqui disfuncional mas sim, não propriamente adaptado. Existem algumas ineficiências que esta directa utilização do resultante directo da fatiação provocaria, entre elas destacam-se: a não catalogação independente de espaços planos e espaços de elevação (que certamente representaria uma mais valia na questão da imposição a estas áreas de pesos de navegação

distintos); a geração de áreas muito numerosas e reduzidas no caso da utilização de vários avatares de alturas diferentes com uma fatiação mais intensiva (principalmente sobre as áreas de elevação) para poder gerir as possibilidades de colisão destes; dificuldades de poder obter uma navegação totalmente correcta e eficiente no caso de projectarmos (como era objectivo inicial) a planificação e pesquisa de percursos sobre uma base de utilização composta unicamente por imagens binárias, para assim podermos a este nível conseguir independência da complexidade do ambiente 3D a povoar. Era necessária uma alteração a esta estrutura de sub-divisões iniciais, de forma a poder responder a estas potenciais limitações.

Este novo passo requerido de selecção e catalogação de novas áreas mais ajustadas a um correcto suporte ao planeamento de percursos, faz uso, e parte inicialmente da estrutura antes definida. A implementação do processo de segmentação das cenas virtuais em altura, descrito anteriormente, facilita a identificação das fatias em altura (*slices*) onde existiriam ou não locais representativos de zonas de acesso a níveis de alturas diferentes, podendo estas zonas serem elevações graduais com imposição de uma fatia para a sua gestão de navegabilidade. Falamos de escadas ou rampas de acesso que servem de transição entre áreas de maior abrangência de navegação, como é o caso do exemplo que vamos seguir. Este processo foi facilmente conseguido, dado que o algoritmo já existente de definição e estabelecimento de camadas consegue situar fatias com estas características (através de um processo de descarte consecutivo de *slices* equivalentes [134], visto no capitulo anterior).

Após o estabelecimento das fatias de dados unicamente necessárias e fundamentais a uma correcta navegação 3D, foram identificadas quais destas intersectavam e ao mesmo tempo continham informação sobre a deslocação de navegabilidade entres os diferentes níveis de altura de um edifício virtual. Tomando o exemplo demonstrativo da cena anterior, na Figura 53 podemos ver identificados 7 fatias de informação consideradas fundamentais para uma correcta navegação 3D, onde 3 delas são referenciadas e correspondem à localização de rampas, como zonas identificativas de transição entre zonas de navegação sobrepostas em altura.

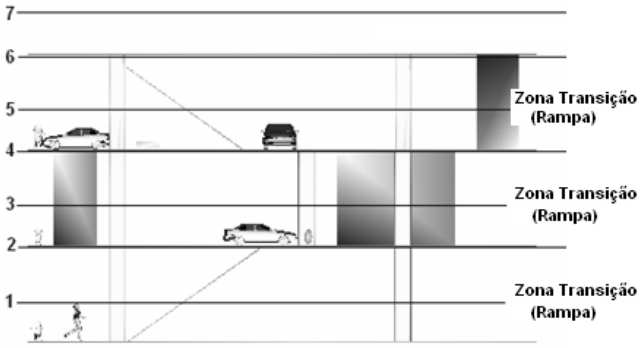


Figura 53 – Fatiação para navegação em altura e detecção de zonas de transição

De acordo com esta identificação, foi definido um algoritmo de localização e diferenciação das diferentes regiões navegáveis, gerando imagens identificativas de zonas de navegação bem definidas (sensivelmente à mesma altura), e zonas de transição para ligações às anteriores (constituídas neste exemplo em particular por rampas). O Algoritmo 4 descreve em detalhe todo este processo. Este Algoritmo agrega mecanismos de preparação, obtenção e catalogação das diferentes zonas de navegação às diferentes alturas, descritos e apresentados em 3 secções distintas (#A, #B e #C).

Algoritmo 4

```
♦ Let Avatar_Step = How much the avatar can jump up or down (both)
♦ Let sliceHeight[0] = 0
♦ Zone [x] [y] [i] = false        // Boolean values (initially all false)
♦ minY = Floor
♦ Int x, y = spatial localization
-----------------------------------------------------------------------
//#A      // For all slices

1 ♦ For all the slices obtained (i = 1 to 7, Figure 46)
2 ♦ {
3   ♦ For all cells (a) in these slices
4   ♦{
5       ♦ If ( slice[i][a] == minY ) { zone [i] [j] [area]= true;}  // minY= floor
6       ♦ If ( slice[i][a] >= sliceHeight[i-1] and
7             slice[i][a] <= sliceHeight[i-1] + Avatar_Step )
8                   ♦ {
9                         ♦ zone [x] [y] [i]= true;
10    ♦ }♦ } ♦ }
-----------------------------------------------------------------------
//#B      // only for the slices where there are ramps or stairs

14 ♦ For all directions of navigation
15 ♦ {
16   ♦ For all slices with ramps or stairs (i = 1, 3 and 5, Figure 46)          // area = Slice
17    ♦ {
18    ♦ For all cells (a) in these slices
19     ♦ {
20        ♦ If ( zone [x] [y] [i] == true )
21         ♦ {
22            ♦ While ( slice[i][a] <= sliceHeight[i+1] and
23                  slice[i][a] >= sliceHeight[i-1] + Avatar_Step )
24            ♦ {
25               ♦ If ( slice[i][a] <= Avatar_Step or                    // Obstacles smaller than jump
26                     slice[i][a] + avatar's height <= sliceHeight[i] )   // If it fits in height
27               ♦ {
28                  // Fill with surmountable (FloodFill) in the different directions
29                     ♦ zone [x +1] [y] [i]= true or zone [x -1] [y] [i]= true or
30                           ♦ zone [x] [y+1] [i]= true or zone [x ] [y-1] [i]= true;
31 ♦ }♦ }♦ } ♦ }♦ }♦ }
-----------------------------------------------------------------------
//#C      // only for the slices where there are no ramps or stairs

35 ♦ For all directions of navigation
36 ♦ {
37   ♦ For all slices without ramps or stairs (i = 2, 4 and 7, Figure 46)
38    ♦ {
39        ♦ Repeat the Previous "FIlodFill" algorithm (lines 20 to 30 ) for the slices where there
40        are no stairs, except in areas already covered by stairs, in order to find
41        locations with gaps of less height achievable and turn in also surmountable.
42 ♦ }♦ }
```

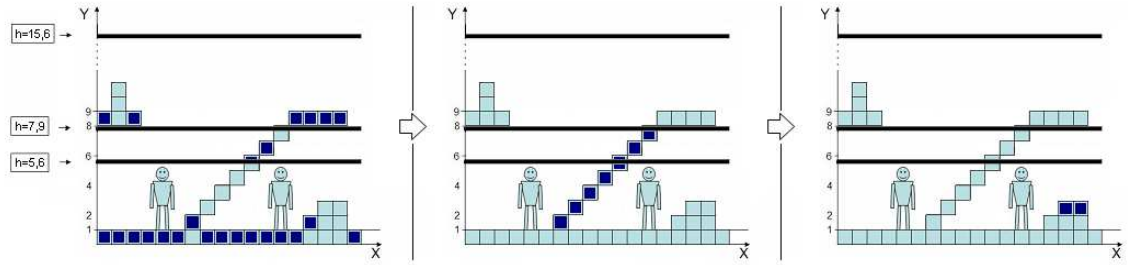
Algoritmo 4 – Obtenção de zonas navegáveis

As figuras que se seguem servem de apoio exemplificativo ao algoritmo, onde podemos analisar algumas situações de teste relacionadas com determinadas condições de subdivisão e catalogação de zonas de navegação presentes em cenas mais simples, para uma maior facilidade de entendimento de todo o processo.

A secção #A descreve o implementar da 1ª etapa do Algoritmo 4 para o tratamento e preparação da informação, considerando nesta fase todas as fatias achadas relevantes à navegação no exemplo da Figura 54 (3 neste caso, às alturas de 5.6, 7.9 e 15.6). Consideremos por exemplo um salto ou a capacidade possível de superar obstáculos para os avatares (*Avatar_Step*) igual a 1.5 unidades. Começando a análise pela fatia de menor altura até à de maior altura (linha 1), nesta fase irão ser consideradas como zonas navegáveis (linhas 5-9) todas as células acima da fatia anterior em teste até à altura do *Avatar_step* (de realçar que para a análise da 1ª fatia é considerada a fatia anterior igual ao minY, representativa da base de navegação).

A secção #B descreve o processo de análise e extensão das fatias representativas das consideradas zonas de transição, neste caso identificadas por escadas. Para o exemplo da Figura 54 é envolvida unicamente a fatia a uma altura de 5.6, onde se consideram unicamente as células achadas relevantes (onde *zone[x][y][i]=true*) na etapa #A para esta fatia (linha 20). É provocado de seguida um teste à possível extensibilidade a partir destas células para as células vizinhas seguintes com valores em alturas superáveis de acordo com o *Avatar_step* definido (Processo *FloodFill* – linhas 25-30), delimitando como terminação deste processo as alturas das fatias posteriores e anteriores (linhas 22, 23).

Por fim, a secção #C estabelece um novo teste às células não consideradas no processo da secção #A, relativas somente em fatias não consideradas de transição (neste caso as fatias às alturas de 7.9 e 15.6). Nesta etapa é agora possível identificar locais de acesso decorrentes da possibilidade de navegação neste tipo de áreas de acordo com as possibilidades de transposição de obstáculos por parte dos avatares. É assim repetido o processo de *FloodFill* idêntico ao da secção anterior, aplicado agora às células com valores de *zone[x][y][i]=false* mas com possibilidades de navegação de acordo com o valor de *Avatar_step* definido (linhas 39-41).

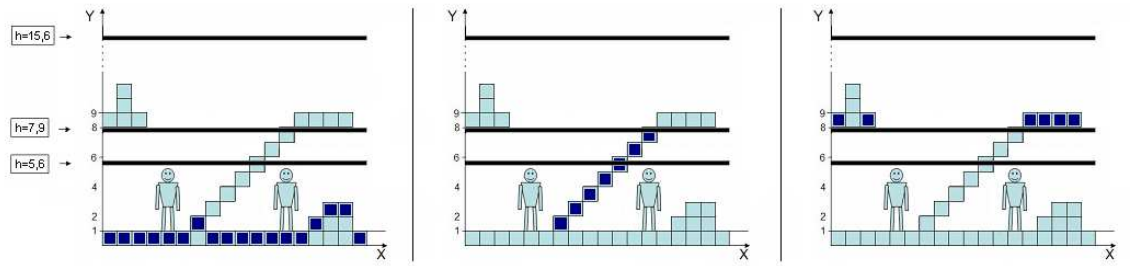


#A #B #C

Figura 54 – Exemplificação das diferentes etapas no Algoritmo 4 com a definição das células aptas à navegação

Durante este processo vão sendo catalogadas as células com possibilidades de navegação (células preenchidas a negro) indexadas directamente com as diferentes zonas de navegação a diferentes alturas. Estes valores de catalogação de células e zonas são armazenados no array *Zone[x][y][i]*, onde o *x* e o *y* identificam bidimensionalmente a células no plano, e o *i* identifica a diferente zona de catalogação em altura.

Na Figura 55 podemos ver agora identificada a catalogação das células consideradas livres e aptas à navegação devolvidas pelas etapas identificadas anteriormente organizadas em zonas de actuação independentes relativas às fatias processadas e envolvidas no processo. De acordo com o exemplo que temos vindo a seguir, são obtidas assim as zonas 1, 2 e 3 de actuação com as respectivas células consideradas ou desconsideradas aptas ou livres à sua navegação por parte das características do ambiente juntamente com a dos avatares envolvidos.



Zona[1] Zona[2] Zona[3]

Figura 55 – Catalogação de zonas independentes de navegação

Por fim e focando agora o aspecto global de navegabilidade com subdivisão em vários blocos de navegação representado na Figura 56, podemos verificar que algumas destas células consideradas aptas, se irão sobrepor em diferentes zonas de catalogação (células com cruzes), configurando um aspecto essencial para uma fase seguinte (em pormenor mais adiante) de identificação de locais de transição entre estas zonas catalogadas.

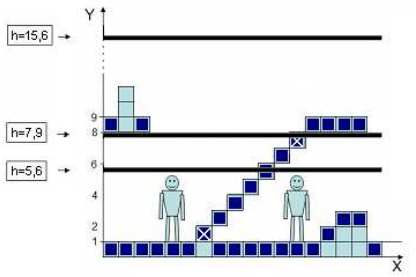


Figura 56 – Visão global das células aptas à navegação com distição de interligação entre as zonas independentes

Retomando de novo a base exemplificativa do mundo 3D virtual anteriormente apresentado como exemplo na Figura 50, podemos observar que o implementar deste algoritmo, gera as zonas de navegação multi-nível apresentadas na Figura 57. Este tipo de informação foi convertido em imagens binárias ou bi-nível (onde a claro são apresentadas zonas de possível navegação e a escuro o oposto) podendo ser mantidas em memória ou guardadas em ficheiros sequenciais de imagens, para posterior análise e processamento. Para a opção de guarda das imagens em ficheiros externos recorreu-se á aplicação *open-source* "DevIL" [143].

Figura 57 – Imagens binárias geradas definindo locais de navegação em altura com base na cena da Figura 50

Através da recolha, análise e armazenamento destas imagens representativas dos espaços navegáveis, podemos detectar o chão de navegação dos 4 pisos do nosso mundo virtual (imagens A, C, E e G), e aquelas que identificam as rampas de acesso a estas superfícies (imagens B, D e F). De realçar o facto de a imagem A estabelecer o piso 0 do nosso edifício virtual e por isso ser navegável em toda a sua extensão com a excepção das zonas de colisão (por debaixo das escadas até a altura do avatar, pilares, e outros obstáculos neste caso uma boca de incêndios). Outro factor de realce, é a rampa de acesso mais larga da imagem F não ter continuidade a partir da altura onde o avatar colidiria com o tecto do andar superior (imagem G) dado que esta passagem está obstruída. Identificam-se ainda que pequenos escalões que se situam a uma

altura inferior ao possível salto do agente, podendo assim ser ultrapassáveis, já não aparecem identificados na imagem E.

5.1.2 Catalogação da Informação Utilizando Algoritmos de Processamento de Imagem

O passo seguinte consistiria em subdividir espacialmente, e catalogar como zonas independentes e navegáveis, cada uma das áreas a branco das imagens obtidas anteriormente. De modo a exemplificar este processo e recorrendo à análise das imagens da Figura 57, identificamos por exemplo na figura C, 2 zonas separadas e independentes de navegação no mesmo piso, e nas figuras D e F encontramos rampas de acesso a níveis superiores ou inferiores que também deverão ser tratadas de forma independente e não como a mesma opção de acesso. Assim deixaremos de ter uma divisão espacial por imagem para passarmos a ter uma divisão espacial por área independente para posteriormente ser processada como tal.
Esta necessidade de processamento e análise digital das imagens geradas foi colmatada utilizando a aplicação open-source OpenCv [144], desenvolvida inicialmente pela Intel e distribuída gratuitamente na Internet em código c/c++ para novos desenvolvimentos e testes com vista à sua evolução e optimização.

> "*This library is mainly aimed at real time computer vision. Some example areas would be Human-Computer Interaction (HCI); Object Identification, Segmentation and Recognition; Face Recognition; Gesture Recognition; Motion Tracking, Ego Motion, Motion Understanding; Structure From Motion (SFM); and Mobile Robotics.*"
>
> *In Opencv Intel´s web page.*

Refinando e integrando a parte da aplicação destinada à detecção de contornos incorporada na suite de processamento digital OpenCv, conseguiu-se mapear todos os polígonos numa determinada imagem, devolvendo a composição destes em número de vértices e a localização espacial dos mesmos (Figura 58). Assim e para cada uma das imagens da Figura 57, obtiveram-se as detecções e especificações de contornos para cada polígono na imagem (com indicação da localização das coordenadas em píxeis, dos vértices dos polígonos reconhecidos) de forma a catalogar espacialmente zonas de navegação independentes na mesma imagem e por conseguinte ao mesmo nível de altura.

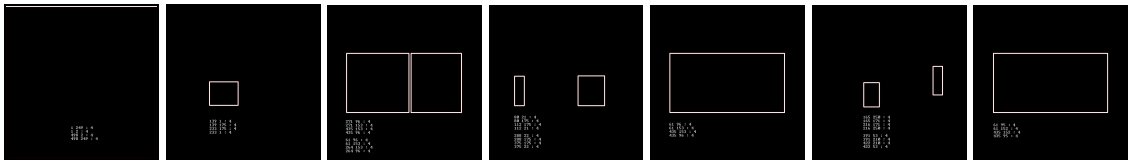
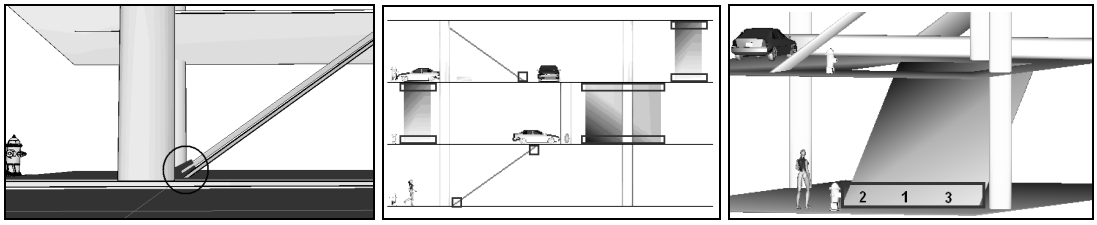


Figura 58 – Detecção e mapeamento da localização independente de diferentes áreas de navegação

5.1.3 | Definição e Localização de Zonas de Interligação

Após a catalogação das localizações espaciais das diferentes zonas navegáveis a várias alturas, seria necessário estabelecer uma ligação de transição entre as mesmas, ou seja determinar quando existe ou não acesso de umas as outras aquando da navegação dos avatares pelo mundo virtual. Não só será necessário aferir a existência de interligação entre as diferentes áreas resultantes, como também especificar pontos onde essas interligações serão consumadas durante o processo de navegação (chamados também de pontos guia ou *waypoints*). Assim, e auxiliados agora pela disposição das áreas navegáveis a diferentes alturas com base nas imagens processadas da Figura 57 e Figura 58, vemos que as possíveis interligações se situarão nas imagens contíguas aquela que está a ser testada. Por exemplo, a área navegável da Figura 58-B só terá ou não ligações (*waypoints*) com as áreas navegáveis situadas na Figura 58-A ou Figura 58-C, assim como só será testada a existência de ligações entre as duas áreas independentes "auferidas" na Figura 58-C com as áreas navegáveis e independentes resultantes da análise da Figura 58-B e Figura 58-D, e assim sucessivamente.

Para poder estabelecer uma ligação entre as várias áreas navegáveis a diferentes alturas foi comparada a localização espacial das mesmas. Quando estas áreas de ligação em teste tiverem locais comuns em altura e se confinarem a uma passagem válida para o avatar, então será imposta uma ligação válida entre essas áreas, onde o avatar dará por consumada a passagem de uma a outra área de navegação. Este processo no global é exemplificado na figura seguinte.



A B C

Figura 59 – Detecção de zonas de interligação em altura e definição de pontos de acesso

Na Figura 59-A podemos identificar as camadas de navegação presentes nas Figura 57-A e B acopladas ao ambiente 3D. Através desta vista exemplificativa é possível compreender de uma forma mais fácil a identificação de zonas de ligação entre as diferentes áreas navegáveis em altura. Assim e realçado pelo circulo da Figura 59-A, é apresentado o local de intersecção à mesma altura das duas áreas de navegação em teste (a escuro a extensa área do chão no piso 0, e a claro a área navegável correspondente à rampa de acesso ao 1º piso).

É importante referir, que neste processo é utilizada em simultâneo a informação dada pelas Figura 57 (definindo as áreas de possível navegação e as áreas interditas à navegação) e a informação fornecida pelas imagens da Figura 58 (onde se identificam as independências e individualidades de zonas navegáveis ao mesmo nível de alturas). É como se trata-se de uma sobreposição ordenada das imagens da Figura 57 catalogadas com a independência da Figura 58 e se identifique quais as áreas a branco que se intersectam à mesma altura entre imagens contíguas.

A extensão deste processo aos mundos virtuais fornece localizações espaciais de interligação entre as diferentes alturas, como se pode comprovar no exemplo da Figura 59-B, onde os quadrados e os rectângulos identificam este tipo de zonas de junção no mundo virtual que nos têm servido de exemplo.

Uma vez definida cada uma destas zonas, torna-se importante encontrar pelo menos um ponto pertencente a ambas, o qual servirá como identificador do destino intermédio que o avatar tomará aquando da transição entre localizações de duas ou mais zonas de navegação interligadas. De forma a optimizar a navegação, foram definidos 3 pontos guia em cada uma das interligações para uma melhor performance e aspecto mais realista da navegação. Estes, podem ser vistos a título demonstrativo na Figura 59-C, onde o ponto nº 1 se situa precisamente no centro do acesso, e os pontos 2 e 3 se situam a ¼ e a ¾ da extensão da ligação. Só um destes, será o

escolhido pelo avatar como guia de destino intermédio durante a passagem por essa zona de interligação, o eleito será aquele que se encontrar mais próximo do avatar quando este alcance esta zona, e servirá de registo de passagem para uma área de altura distinta.

5.1.4 Construção do Grafo Hierárquico Associado

Após a preparação dos dados em zonas independentes de navegação a diferentes alturas, com as ligações (*waypoints*) entre estas também já bem definidas, estão reunidas as condições para a construção de um grafo hierárquico que servirá de planificador e regulador das zonas intermédias de passagem entre os pontos de origem e destino de navegação. Terá as funções de devolver o caminho mais curto entre dois pontos, caminhos alternativos a este, informar da existência ou não de ligação caso esta se verifique ou não, quais as áreas sucessoras e antecessoras em relação a uma determinada localização, entre outras.

Resumidamente, o grafo utilizado é constituído por listas de adjacência implementadas com listas ligadas com a definição de nodos, e sequências (*arrays*) de apontadores para esses mesmos nodos. Dinamicamente são construídas as sequências de apontadores, iguais ao número máximo de vértices (nodos) presentes em cada grafo, também estes gerados dinamicamente. Cada nodo corresponderá assim a cada uma das zonas independentes de navegação a diferentes alturas, e os apontadores às suas ligações ou pontos guia.

Da aplicação deste grafo hierárquico ao modelo 3D em exemplo, obtém-se o esquema ordenado representado na Figura 60.

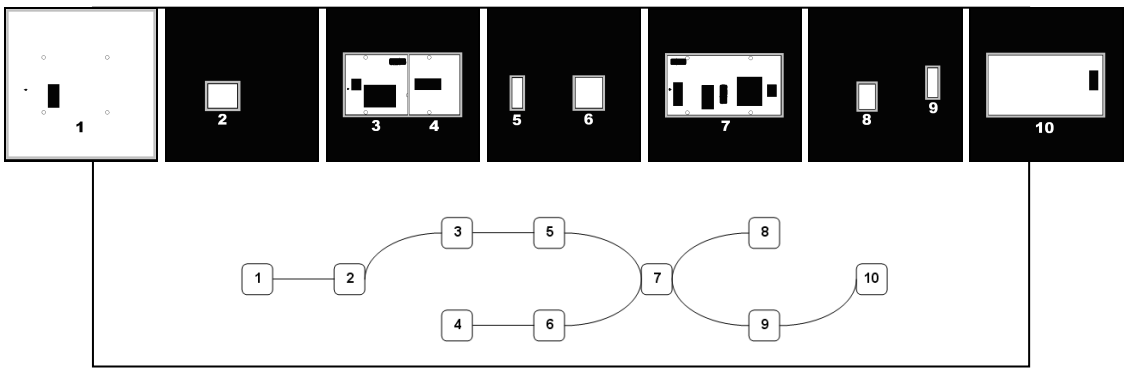


Figura 60 – Grafo extraído da subdivisão espacial ao mundo virtual em exemplo

5.2 Estabelecimento de Rotas de Navegação Através do Grafo

5.2.1 Navegação e Procura de Percursos Através de Algoritmos Apropriados

Estabelecidas em pré-processamento todas as alternativas de navegação multi-nível dadas pelo grafo hierárquico, o próximo passo seria optimizar a movimentação dos avatares durante o percurso entre os vários nodos. Um ponto crucial no estabelecimento de um "*roadmap*" de navegação entre os pontos de origem e destino é a localização em tempo real no grafo dos respectivos nodos de localização de partida e de chegada.

A localização do ponto de chegada e respectivo nodo é definido pelo utilizador quando estabelece o destino a dar aos avatares. O ponto de partida traduzido na localização espacial e respectivo nodo no grafo é calculado com base na localização actual do avatar. Desde já, salientar que durante o processo de "*path planning/finding*" no modelo aqui apresentado, podemos inúmeros pontos e nodos de partida para um mesmo destino quando aplicamos instruções em grupo a muitos avatares em localizações e alturas dispersas no ambiente 3D. Assim, e para a construção dos *roadmaps* e alternativas a estes, são inicialmente adquiridas todas as posições e respectivos nodos de partida de todos os avatares envolvidos, num processo de localização da sua posição actual nas fatias dadas pela Figura 57, com as zonas independentes navegáveis da Figura 58 e por último a sua incorporação nos nodos definidos no processo de construção do grafo a partir da imagem apresentada na Figura 60.

Foi já referido que o método de detecção de colisões aqui utilizado com base em mapas de alturas considerados a vários níveis durante a fatiação horizontal do mundo 3D estava já numa fase de avançada experimentação e funcionalidade. Teriam agora de se encontrar direcções lógicas de navegação aquando da detecção das referidas colisões, tendo como destino os pontos intermédios (*waypoints*) e pontos finais de navegação (*path finding component*).

O algoritmo de teste utilizado para *path-finding* utilizado para navegação neste espaço hierárquico é baseado no muito conhecido algoritmo A*, com a introdução de

algumas alterações associadas à navegação em altura, tentando colmatar assim algumas ineficiências deste, quando confrontado com ambientes deste género.

5.2.1.1 | Utilização do Algoritmo A* com Algumas Modificações

Para implementar este tipo de procedimento de pesquisa em grafos utilizando o algoritmo A*, é usual utilizar duas listas de nós:

- OPEN: nós que foram gerados e tiveram a função heurística de "proximidade" da solução aplicada, mas que ainda não foram examinados (ou seja, ainda não foram gerados os seus sucessores). Esta lista é realmente uma lista de prioridade, na qual os elementos com maior valor da função heurística de proximidade da solução serão os mais promissores.
- CLOSED: nós que já foram examinados. É necessário manter esta lista para evitar pesquisar percursos já examinados anteriormente.

Descrição dos passos do algoritmo A* base implementado:

1. Inicie com OPEN contendo só o nó inicial. Atribua para esse nó: g=0 ; h' ao resultado da aplicação da função heurística; f' = 0 + h' = h'. CLOSED será a lista vazia.
2. Até encontrar uma solução, repetir: Se não existem nós em OPEN, falhar. De outra forma, encontre o nó com menor valor de f'. Chame a esse nó BESTNODE. Retire-o de OPEN e coloque-o em CLOSED. Verifique se BESTNODE é uma solução. Se for, retorne a solução e termine. Senão gerar todos os sucessores de BESTNODE. Para cada SUCESSOR faça:
    a. Coloque o SUCESSOR a apontar para BESTNODE.
    b. Calcule g(SUCESSOR)=g(BESTNODE)+custo do percurso entre BESTNODE e SUCESSOR.
    c. Verifique se SUCESSOR se encontra na lista OPEN. Se sim, chame a esse nó OLD. Como este nó já existe, pode desprezar-se SUCESSOR. Deve ainda decidir-se se a ligação parental de OLD se deverá manter ou mudar. Se g(OLD)>g(SUCESSOR) então o pai de OLD muda para o pai de SUCESSOR e g(OLD) muda para g(SUCESSOR).

d. Se SUCESSOR não estava em OPEN, verifique em CLOSED. Se está, chame a esse nó OLD e adicione-o à lista de sucessores de BESTNODE. Se g(OLD)>g(SUCESSOR) então o pai de OLD muda para o pai de SUCESSOR e g(OLD) muda para g(SUCESSOR). Além disso será necessário propagar esta alteração a todos os descendentes de OLD, ou seja, o valor da função g de todos os descendentes será diminuída de g(OLD)-g(SUCESSOR). Cada propagação terminará num nó que esteja ainda em OPEN ou num que não tenha descendentes. Esta propagação será um percurso do tipo primeiro em profundidade que alterará os valores de g (e, consequentemente os valores de f'), terminando em cada ramo que não tenha filhos ou então num nó para que tenha já sido encontrado um melhor percurso. Esta última condição é fácil de verificar, dado que um nó apontará sempre para o pai que seja melhor em termos de percurso. Quando se propaga para um determinado nó, verifique se o pai inscrito nesse nó é o nó de onde veio. Se sim, continuar a propagação, senão verifique se a solução obtida por este percurso é melhor do que a do percurso anterior. Se a solução resultante da propagação for melhor do que a existente, alterar o pai do nó e o seu valor de g, e continuar a propagação, senão parar a propagação.
e. Se SUCESSOR não pertencia nem a OPEN nem a CLOSED, coloca-se SUCESSOR em OPEN, adiciona-se SUCESSOR à lista de filhos de BESTNODE. Calcule g(SUCESSOR) e f'(SUCESSOR).

A operação do algoritmo é simples. Procede em passos, expandindo um nó em cada passo, até gerar um nó que corresponde a uma solução. Em cada passo, o nó a expandir é o nó considerado mais promissor (nó com menor valor de f(n), se for uma minimização) da lista de nós não expandidos. É então gerada a lista de nós sucessores do nó seleccionado e para cada um deles é aplicada a função f(n), sendo acrescentados à lista OPEN de nós, se ainda não tiverem sido gerados.

Com um número considerável de unidades para implementações de *pathfinding* no "tabuleiro" e um mapa de pesquisa razoavelmente grande, é normal notar um elevado uso de processamento de CPU e perda consequentemente de performance da animação. Isto é verdade até para os profissionais que projectam jogos como "*Starcraft*" ou "*Age of Empires*", que necessitam de recorrer a soluções personalizadas

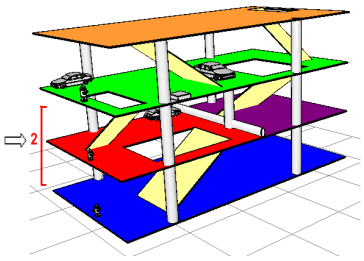
de modo a poder incrementar a velocidade. Podemos encontrar várias destas soluções em [145], algumas delas são aqui sumariadas:

- Considerar um mapa menor ou com menos unidades.
- Não acoplar *pathfinding* para muitas unidades de cada vez. Gerar grupos de avatares e realizar procuras de percurso alternadamente.
- Considerar usar grelhas com divisões (células) maiores para o mapa, introduzindo se necessário uma interpolação. Isto reduz o número total de nós procurados para encontrar o caminho. Podendo usar áreas grandes para caminhos mais longos, e alternando para pesquisas mais acuradas, usando áreas menores ao chegar perto do objectivo.
- Para caminhos longos considerar utilizar caminhos pré-calulados.
- Considerar pré-processar o mapa para descobrir que áreas são inacessíveis, áreas estas também chamadas de ilhas ou lugares inalcançáveis.
- Num ambiente cheio de labirintos, considerar marcar manualmente nós que não conduzem a lugar algum, como becos sem saída.

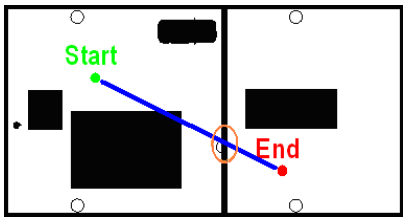
As implementações de pré-processamento de análise e decomposição do ambiente, referenciadas nas secções anteriores, facilitariam assim e muito, as alterações a implementar na aplicação algoritmo A* para as condições exigidas. De seguida são expostas algumas situações onde houve alteração ao algoritmo inicial A* de modo a torná-lo mais eficiente em navegação autónoma em altura para um número elevado de avatares.

- Resultados Instantâneos:

A decomposição e catalogação do espaço numa estrutura hierárquica de interligação de nodos possibilitou rápidos resultados a acessos não alcançáveis, onde a pesquisa A* gastaria tempo e recursos para no final resultar (como se poderia verificar inicialmente) numa impossibilidade de acesso. É o caso do 2º nível de alturas do exemplo do mundo 3D da Figura 61, onde há uma clara impossibilidade de deslocação à mesma altura entre localizações opostas e mais distantes devido a uma barreira que divide a plataforma em duas secções separadas e independentes para a navegação.



Mundo 3D - Exemplo



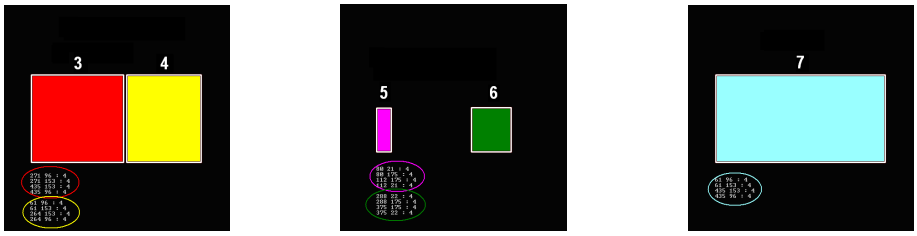
2º Nível Altura

Figura 61 – Catalogação de áreas acessíveis/inacessíveis através do pré-processamento

- Pesquisa Dinâmica:

A utilização de algoritmos de processamento de imagem para análise da independência de zonas navegáveis à mesma altura em diferentes localizações espaciais, possibilitou a criação de mapas de análise de navegação inferiores e mais eficientes através de uma alocação dinâmica do espaço de pesquisa.

Assim, e pegando no exemplo do modelo 3D em exame e nas imagens da Figura 62 representativas do 2º e 3º Piso e rampas de acesso entre estes, vemos que um avatar que se encontre algures no 2º Piso na Região 3, e que pretenda deslocar-se para a região 4 desse mesmo Piso, terá de deslocar-se ao 3º Piso (região 7) através da rampa 5 (Figura 63, 1 e 2), e finalmente aceder à região 4 pela rampa 6 (Figura 63, 3 e 4). Durante este trajecto, serão efectuadas 5 pesquisas diferentes através do algoritmo A* (nas regiões 3,5,7,6 e 4 consecutivamente). Cada uma destas terá como objectivo alcançar o destino intermédio de ligação entre as diferentes zonas (*waypoints*), excepto na última pesquisa (região 4) onde se procurará o destino e objectivo final. O fundamental para o incremento da eficiência, é que cada pesquisa se restrinja à região em actuação do avatar (via coordenadas espaciais resultantes inicialmente da aplicação dos algoritmos de processamento de imagens), sendo estas regiões identificadas e passadas dinamicamente desde o pré-processamento à área de pesquisa do algoritmo A* em tempo real, diminuindo desta forma a dimensão dos mapas de pesquisa A* e resultando num incremento importante em relação aos tempo de resposta e animação.



2º Piso Rampas de Acesso 3º Piso

Figura 62 – Catalogação de áreas acessíveis com independência

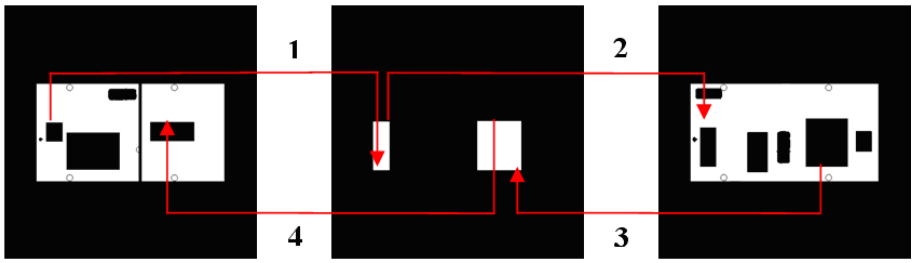


Figura 63 – Identificação dos destinos intermédios de ligação entre percursos.

- Imposição de Pesos ou Proibições:

Outra afinação implementada ao algoritmo A* inicial é a navegação com recurso a custos de terreno variável. Assim o terreno deixa de ter unicamente as opções: passável ou não-passável, para incorporar custo de movimento de maior ou menor peso, dependendo do virtual esforço que o avatar teria de suportar aquando da necessidade de subir ou descer em altura para atingir determinados objectivos. Existem nesta catalogação também, vantagens de actuação e comportamentos implícitos que poderão ser incluídos às personagens virtuais, quando presentes nesta diferenciação de espaços. Já que em superfícies em declive, a velocidade e cansaço de navegação pode variar substancialmente dependendo da direcção em relação ao declive, assim como, é também normal nestas áreas mais exíguas de transição e passagem entre objectivos de deslocação espacial, acontecerem e se concentrarem situações de engarrafamentos e colisões inter-avatares mais recorrentes, possibilitando desta forma identificar e antecipar facilmente este tipo de constrangimentos na navegação através da possível guarda do declive em relação à área catalogada, quando se envolvem grandes quantidades de personagens em movimento. O avatar poderá também ser impedido de deslocar-se para zonas, onde se sabe através do pré-processamento inicial, que resultariam em resultados morosos ou de posterior inversão de direcção.

No exemplo da Figura 64, apresentam-se duas situações de simulação onde se exemplifica a navegação com recurso a custos de terreno variável ou proibição de espaços. Na imagem da esquerda poderemos impor pesos mais elevados às células em zonas de navegação em declive, de acordo com a sua largura ou inclinação (aumentando o previsível esforço e atenção dos avatares). Na imagem da direita, podemos observar outro exemplo, de um avatar com o objectivo de direcção a uma das bocas de incêndio, tendo uma opção clara de destino à sua frente (no piso inferior) sem necessidade de alteração de direcção, pelo meio encontra uma rampa que poderá encontrar-se sobrelotada ou inadequada, e assim definida (temporariamente ou não) com a imposição de proibição à navegação sobre esta superfície, obrigando o avatar a deslocar-se para outra opção de destino (neste caso para uma boca de incêndios no mesmo andar), mesmo que aparentemente este trajecto se apresente inicialmente como mais complexo ou moroso.

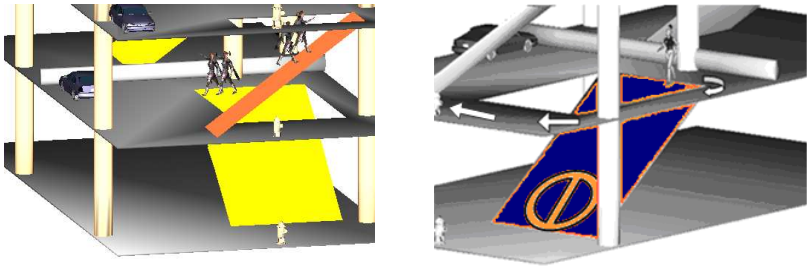


Figura 64 – Navegação com recurso a custos de terreno variável

Resumidamente e em jeito de conclusão nestes pontos, as melhorias implementadas ao algoritmo A* e adaptadas ao tipo de navegação abordado e requerido, geram:

- Execuções de tarefas em tempo "mais real", reduzindo os tempos de resposta;
- Menor consumo e acesso à memória durante a pesquisa;
- Controlo de importâncias e custos de navegação, recorrendo à subdivisão de processos em vários escalões;
- Um melhor controlo da heurística na navegação realmente 3D;
- Rápidas localizações em altura e soluções imediatas de impossibilidades de acesso através do acesso ao grafo hierárquico registado em pré-processamento;
- Possibilidade de pesquisas com destinos faseados, mais curtos e mais suaves.

5.3 | Testes e Exemplos da Aplicação do Modelo

Foram realizados testes e simulações de navegação em vários modelos 3D desconhecidos à partida, com um número elevado de avatares, de forma a medir a aplicabilidade da estrutura hierárquica criada pelo modelo aqui especificado, conjuntamente às implementações adicionadas ao algoritmo A*, o qual foi denominado de A* hierárquico. Os testes foram executados num PC com as seguintes características:

CPU: - Intel Pentium4 3 Gz
RAM: - 1,5 Gb Ram
GPU: - NVIDIA - GeForce 5950 Ultra 256 Mb

De forma a preparar um ambiente de testes apropriado, foram considerados dois mundos virtuais com os mesmos percursos, mas com características diferentes. Inicialmente um mundo 3D poderá ser transformado num problema em 2D, como se pode observar na Figura 65. Foi testado em ambos os mundos o modelo de pesquisa hierárquico aqui descrito, e o método de pesquisa clássico A*. A grelha de pesquisa para cada fatia resultante da estruturação hierárquica foi composta por 512x512 píxeis. De ressalvar no entanto, que na versão hierárquica, somente uma parte desta área (onde o avatar está situado), é necessária para a procura em cada nodo.

Quando foram considerados testes com um elevado número de avatares, a sua representação gráfica foi simplificada a um cubo (*6 polygons*) de forma evitar que a performance gráfica pudesse causar uma forte influência na performance e influenciar desse modo a performance dos resultados da simulação/animação.

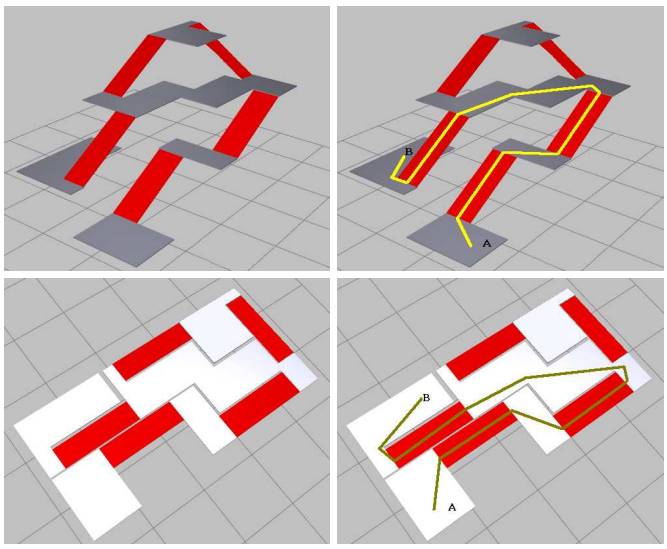


Figura 65 – Mundo 3D e o seu equivalente em 2D

Em análise, estava o desempenho de 2 algoritmos de *path-finding* (A* 2D Clássico e A* 3D Hierárquico), sobre uma malha de pesquisa de 512x512 (262.144 células) por cada camada de navegação em altura. Os mundos virtuais a povoar, foram "carregados" sobre esta malha onde os avatares se deslocariam entre localizações de origem e destino.

A informação que consta da Tabela 5 mostra os resultados em segundos para ambos os métodos de acordo com a simulação nos ambientes da Figura 65. O período de espera reflecte o tempo dispendido para a computação inicial do caminho a traçar. O período de movimento representa o tempo requerido para a movimentação dos avatares desde a localização de início (ponto A), até à localização de destino (ponto B). De ressalvar, que os pontos identificados na Figura 65 são meramente referencias, ou seja, foi considerada uma pequena área em volta deste ponto. Muito importante de referir, é que cada um destes avatares em teste pesquisará de forma individual o destino final com base na sua origem, gerando assim uma muito maior quantidade de processamento do que na situação de se situarem num mesmo grupo onde houvesse um único cálculo de rota para todo grupo. Isto implica que cada um deles estabeleça o seu próprio caminho, assim sendo, serão computados tantos percursos como os avatares presentes em cada um dos testes.

É claro a partir dos dados apresentados na Tabela 5, que existe uma diferença substancial no período de espera (computação dos percursos) entre os dois métodos. Como esperado o modelo A* 3D hierárquico desenvolvido é muito mais rápido nesta etapa, já que, o percurso é traçado e computado em vários períodos do espaço e não durante a totalidade de percurso entre o ponto inicial e final. Mover os avatares, após a computação dos percursos, é sensivelmente equivalente para ambas as situações.

	1 avatar			100 avatares			500 avatares			1.000 avatares			5.000 avatares		
	Período espera	Período movimento	Total	Período espera	Período movimento	Total	Período espera	Período movimento	Total	Período espera	Período movimento	Total	Período espera	Período movimento	Total
2D	0	1,71	1,7	18,9	2,75	22	79	6,6	86	149	12	161	846	52	898
3D	0	1,45	1,5	0	2,57	2,6	0,45	6,53	7	1,07	11,58	13	5,38	49,66	55
Fps	564 fps			329 fps			125 fps			71 fps			15 fps		

2D – A* 2D Clássico
3D – A* 3D Hierárquico

Tabela 5 – Resultados da simulação para o mundo da Figura 65. Os resultados são apresentados em segundos.

	10 avatares		50 avatares		100 avatares	
	Tempo em Colisões	Total	Tempo em Colisões	Total	Tempo em Colisões	Total
2D	14,5	16,2	43,2	45,2	104,8	112
3D	0,4	2,05	4,67	6,6	10,53	13,1
Fps	528 fps		445 fps		329 fps	

2D – A* 2D Clássico
3D – A* 3D Hierárquico

Tabela 6 – Resultados da simulação para o mundo da Figura 60 com detecção de colisões entre os avatares. Os resultados são apresentados em segundos.

Os resultados oferecidos pela Tabela 6 mostram os valores para o mesmo problema, mas neste caso considerando a detecção de colisões entre os avatares. Uma colisão é detectada, na tentativa de um avatar tentar ocupar uma posição já assumida espacialmente por outro avatar, aqui o percurso é re-computado para este avatar, definindo a posição actual como um novo ponto de partida. Os períodos de tempo gastos após colisão reportados na Tabela 6, representam assim todas as novas computações de caminhos alternativos ao traçado inicialmente para os avatares que se encontrem em colisão.

Novamente, é claro que a representação 3D hierárquica apresenta uma substancial evolução em relação ao conceito clássico da pesquisa A* (conceito de pesquisa não hierarquizado). Esta evolução deve-se principalmente ao facto da pesquisa no modelo de procura hierárquico se restringir a uma re-computação dentro do mesmo nodo, não abrangendo a totalidade da grelha de navegação possível entre os pontos inicias e finais primários.

Num segundo teste, foi usado o mundo 3D virtual da Figura 66, e testada a performance entre os 2 caminhos aí representados (Start-End1 e Start-End2). Este teste utilizou unicamente o modelo 3D hierárquico associado agora a uma verdadeira navegação multi-nível. O ponto de partida (Start) localiza-se no centro da plataforma de base inicial, o primeiro objectivo de destino (End1) encontra-se no canto esquerdo do último piso (conforme perspectiva da Figura 66), e o segundo destino (End2) situa-se por debaixo da rampa que acede à secção direita no primeiro piso. A grelha de procura manteve-se nos 512x512 para cada fatia de teste. Como referido anteriormente, somente o nodo do grafo associado em tempo real a cada avatar servirá de espaço de procura, reduzindo substancialmente a área de pesquisa, excepto no piso do chão, onde a quase totalidade da grelha é "caminhável". Os valores

resultantes (tempos testados em segundos) de deslocações entre a origem e destinos, com e sem detecção de colisões entre os avatares, são apresentados respectivamente nas Tabela 7 e Tabela 8.

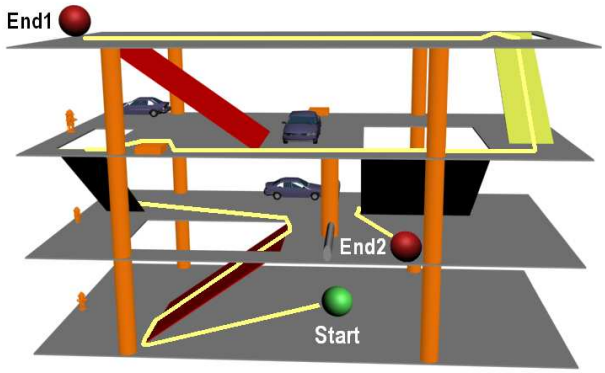


Figura 66 – Mundo realmente 3D com 7.820 Polígonos. Testes de animação e pesquisa e de procura de objectivos espaciais

	1 avatar			100 avatares			500 avatares			1.000 avatares			5.000 avatares		
	Período Espera	Período Movimento	Total	Período Espera	Período Movimento	Total	Período Espera	Período Movimento	Total	Período Espera	Período Movimento	Total	Período Espera	Período Movimento	Total
End1	0	3,11	3,11	1,14	6,48	7,62	2,98	13,55	16,5	9,23	20,19	29,4	46,9	90,72	138
End2	0	2,26	2,26	0,93	3,3	4,23	2,75	11,23	14	7,19	15,93	23,1	31	73,9	105
Fps	478 fps			297 fps			114 fps			65 fps			15 fps		

End1 – Trajecto Start – End1, usando o A* 3D Hierárquico
End2 – Trajecto Start – End2, usando o A* 3D Hierárquico

Tabela 7 – Resultados da simulação para o mundo da Figura 66 sem detecção de colisões entre os avatares. Os resultados são apresentados em segundos.

	10 avatares		50 avatares		100 avatares	
	Tempo em Colisões	Total	Tempo em Colisões	Total	Tempo em Colisões	Total
End1	0,47	3,6	9,1	13,7	24,68	32,3
End2	0,35	2,87	8,9	12,4	24,82	29,1
Fps	528 fps		445 fps		329 fps	

End1 – Trajecto Start – End1, usando o A* 3D Hierárquico
End2 – Trajecto Start – End2, usando o A* 3D Hierárquico

Tabela 8 – Resultados da simulação para o mundo da Figura 26 com detecção de colisões entre os avatares. Os resultados são apresentados em segundos.

De recordar, como mencionado anteriormente, que cada avatar "implementa" o seu próprio caminho, havendo assim tantas procuras A* hierárquicas quantos avatares

presentes na cena. É sabido, que esta não é uma situação óptima nestes casos em particular, onde todos partilham a mesma localização de destino, mas demonstra bem a escalabilidade da aplicação do algoritmo. Esta implementação, não é de todo, uma restrição do nosso modelo, já que se pode lidar facilmente com grupos de personagens associados a um outro nível de hierarquia de gestão de grupos ou multidões de personagens virtuais, e computado um único percurso para um dos líderes de cada grupo (investigação incorporada no capítulo seguinte).

Quando comparados os resultados das Tabela 7 e Tabela 8, permanece evidente um forte impacto na performance quando adicionada a componente de detecção de colisões entre os avatares. É no entanto de notar, que a componente de detecção de colisões de per si, não se mostrou como um factor relevante a este incremento, como demonstrado em [146]. O aumento do tempo de deslocação espacial deve-se sobretudo à re-computação das pesquisas dos percursos, cada vez que existia uma colisão potencial.

A utilização do algoritmo A* provoca o início da animação somente quando estiver encontrado o percurso até determinado objectivo, o que gera alguns momentos de espera de acordo com a dimensão da grelha de pesquisa, e posicionamento dos alvos. Assim, no início ou após atingidos pontos intermédios de pesquisa, poderá haver um tempo de computação da animação, que é o tempo destinado à nova procura do "*roadmap*" seguinte. Estes valores são por norma insignificantes, excepto quando utilizados um número elevado de avatares, juntamente com percursos de difícil resolução em mapas de grelhas de grandes dimensões (como é o caso do 512x512 aqui utilizado nas varia fatias, resultantes do mapeamento do mundo virtual 3D).

A Tabela 9 apresenta os resultados parciais médios para movimentações de 1.000 avatares, durante os trajectos entre as localizações de Start e End1 e End2 respectivamente. Os tempos são apresentados em segundos, e reflectem principalmente, os períodos de computação de novos percursos intermédios, quando os avatares atingem localizações de novas procuras de trajectos em grafos de maiores dimensões (neste caso, normalmente à saída das rampas).

	1.000 avatares			
		Chegada	Partida	Diferença
Trajecto1:	**Inicio**		3,81	3,81
Star - End1	Cimo rampa 1	7,92	8,39	0,47
	Cimo rampa 2	12,57	17,18	4,61
	Cimo rampa 3	22,1	22,44	0,34
	FIM	**29,42**		
			Espera Total:	9,23

	1.000 avatares			
		Chegada	Partida	Diferença
Trajecto2:	**Inicio**		3,81	3,81
Star - End2	Cimo rampa 1	7,92	8,39	0,47
	Cimo rampa 2	12,57	14,29	1,72
	Baixo rampa 3	19,56	20,75	1,19
	FIM	**23,12**		
			Espera Total:	7,19

Tabela 9 – Resultados parciais com 1.000 avatares para simulações no mundo da Figura 66 sem detecção de colisões entre os avatares. Os resultados são apresentados em segundos.

Em resumo, e após a verificação dos testes efectuados, é possível validar este modelo capaz de processar uma qualquer "sopa de polígonos", de modo a poder obter informação das suas acessibilidades, e construir uma estrutura de navegação hierárquica, que combinada com o algoritmo A*, promove uma solução completa, favorecendo a eficiência e escalabilidade. A única etapa a lidar com a geometria poligonal das cenas é a etapa inicial, onde é abstraído o mapa de profundidades do ambiente 3D a simular. Todas as outras etapas, são *Image Based*.

O processo é totalmente automatizado, sem necessidade de qualquer intervenção por parte do utilizador, e capaz de lidar com a estrutura multi-nível de edifícios 3D, de uma forma natural. Este modelo, cria um número sustentável de sub-divisões (nodos), mesmo verificando que por vezes até seriam desejáveis alguns mais (por exemplo no caso do piso inferior do modelo 3D em teste). O que poderá ser conseguido através de métodos de partição de espaços planares bem conhecidos, relatados no estado da arte e referenciadas no capítulo 3 desta tese.

As imagens seguintes, mostram algumas sequências de testes e simulações de navegação utilizado o modelo de planeamento e procura de percursos aqui produzido, aplicado a alguns modelos 3D.

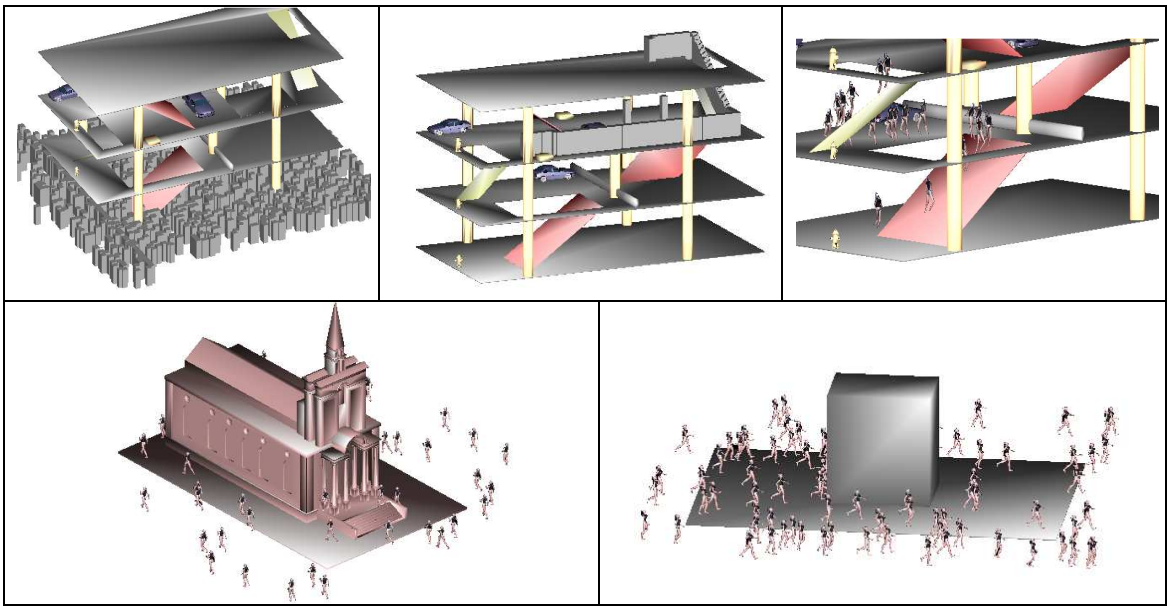


Figura 67 – Imagens de testes noutros mundos virtuais

Todos os testes e resultados subjacentes a este capítulo, foram submetidos a conferência internacional, gerando o artigo [147] aceite à SCCG08 – Spring Conference on Computer Graphics, em Abril de 2008, Bratislava, Eslováquia.

CAPÍTULO 6

Simulação Comportamental de Pedestres em Ambientes 3D

Durante os últimos anos, a modelação virtual tornou-se comum em diversas áreas, como o entretenimento, a simulação, as aplicações arquitectónicas, entre outras. A construção destes modelos virtuais é, com a excepção dos jogos de computador, ainda pouco orientada para a integração de personagens virtuais que os habitem. Povoar estes modelos, com entidades virtuais possuindo níveis de autonomia associados às necessidades pretendidas da simulação, é uma tarefa bastante árdua, envolvendo várias áreas ainda em desenvolvimento, como se pude comprovar pelos exemplos enumerados no seguimento desta tese.

Presentemente, os esforços em representar o comportamento humano num ambiente virtual requerem por norma um modelo do processo da observação do mundo e a consequente sintetização da informação recolhida. A maioria dos sistemas é formado desta combinação: como resposta a um subconjunto do estado do ambiente virtual baseado nos sentidos; e reagindo desde uma combinação de respostas programadas já planeadas, ou obtidas em tempo real por métodos heurísticos. Posteriormente, as decisões e acções aplicadas podem ser explicitamente programadas para as mais variadas alterações no ambiente comportamental. A IA pode disponibilizar paralelamente a "sensibilidade" a alguns aspectos do ambiente virtual, e tomar decisões em resposta ao que vai descobrindo.

A indeterminabilidade e a complexidade do comportamento humano são também recorrentemente um problema para os investigadores. O ramo da IA, que trabalha principalmente com os fundamentos da psicologia, tenta ainda produzir um modelo testado, completo e válido, correspondente ao conceito não determinístico do comportamento humano. A questão coloca-se: será que algum dia um destes modelos

será capaz de alcançar o degrau significativo de anular as ainda enormes diferenças entre o pensamento humano e processo computacional.

6.1 | Descrição Geral

A importância de modelos avançados de inteligência artificial, no processo de gestão abrangente de todas as componentes que pretendam virtualizar o conceito intrínseco ao comportamento dos seres humanos, actuando de forma individual ou em grupo, pode gerar importantes mais valias. Um dos exemplos é a estruturação de esquemas que reduzam a pressão dos recursos computacionais do sistema quando comparado com um nível de controlo por unidade, permitindo assim um maior detalhe no comportamento do grupo e uma maior profundidade no seu planeamento. Porque por exemplo, e ao contrário do mundo real, é perfeitamente razoável para um sistema de IA, ter uma entidade global que poderá ser directamente controlada, dirigindo assim indirectamente muitas das personagens virtuais presentes na simulação. Alternativamente, interacções entre unidades de entidades singulares poderão também ser usadas de modo a produzir comportamentos mais complexos. Individualmente o software inerente a cada entidade poderá ser composto de simples acções comportamentais, que quando em grupo, poderão também exprimir e mostrar comportamentos emergentes, que aparentam muito mais inteligência do que a simples estrutura de um só poderia sugerir.

As simulações computacionais quando utilizam grandes multidões, com centenas ou milhares de indivíduos, requerem modelos simples e eficientes, mas capazes de proporcionem uma descrição acurada da realidade simulada, tal como acontece nos diversos tipos de modelos encontrados no livro de Schreckenberg [148]. Assim, outro desafio claro é a progressão em termos de eficiência e desempenho da própria animação. Um sistema que possa simular realisticamente os comportamentos de humanos, e ao mesmo tempo consiga um aspecto visual agradável e realista (com um *rendering* de qualidade e em tempo-real) das personagens virtuais em movimento, pode ajudar o utilizador a extrair conclusões mais exactas a partir da observação do mundo em simulação, e usar essas conclusões para optimizar o sector para o qual o modelo simulacional foi desenvolvido, como por exemplo o caso das acessibilidades em construções urbanísticas.

Os comportamentos associados a uma simulação de agentes[17] virtuais representando humanos, podem ser classificados em comportamentos de alto-nível, envolvendo principalmente aspectos relacionados com a forma de como os agentes interagem entre eles, condicionam os seus comportamentos dependendo dessas interacções, surgindo conceitos como autonomia, inter ajuda, comunicação, comportamento individual e de grupo, etc; e em comportamentos de baixo-nível abarcando os aspectos relacionados com funções fundamentais de base, necessárias para atingir necessidades que suportem outros comportamentos de mais alto nível, integrando a interacção com o ambiente e outros agentes de modo a evitar colisões e consequente planeamento de rotas ou caminhos de navegação. Este tipo de comportamentos de mais baixo-nível foi já analisado nos capítulos anteriores.

6.1.1 O Ambiente

Não podemos falar de agentes autónomos sem mencionar aspectos relacionados com a análise do ambiente, pois sem esta, seria impossível a existência destes entes autónomos. Abaixo seguem características identificadores dos diversos tipos de ambientes, conforme Russel e Norvig [149]:

Acessível/Inacessível: Se o mecanismo sensorial do agente consegue aceder ao estado completo do ambiente é dito que o ambiente é acessível. Quando o agente possui somente uma visão local do ambiente, o ambiente é chamado de inacessível.

Determinístico/Não Determinístico: Um ambiente é dito determinístico se o próximo estado do ambiente é completamente determinado pelo estado actual e pelas acções seleccionadas pelos agentes.

Factual/Não Factual: A experiência do agente é dividida em episódios, cada um destes consiste na percepção/acção do agente. A qualidade da acção depende somente do episódio actual, não dependendo de acções que ocorreram nos episódios anteriores. Estes são os chamados ambientes factuais.

[17] Termo herdado nesta fase, para uma diferenciação de nível superior em relação aos avatares

Dinâmico/Estático: Se o ambiente mudar enquanto o agente efectuar processamento inteligente, então o ambiente é chamado de dinâmico, caso contrário é dito que o ambiente é estático. Se o ambiente não muda com a passagem do tempo, mas existe alteração na performance do agente, então o ambiente é considerado semi-dinâmico.

Discreto/Contínuo: Se existe um número limitado de percepções e acções o ambiente é discreto. Por exemplo, no caso de um agente que joga xadrez, existe um número fixo de movimentos por turno, é dito que o ambiente é discreto. No caso de um agente que dirige um táxi o ambiente é contínuo – a velocidade do veículo está entre um intervalo contínuo.

Ambientes inacessíveis, não-factuais, dinâmicos, contínuos e não-determinísticos são considerados os mais difíceis de serem implementados, pois os agentes possuem uma visão local do ambiente, as suas acções dependem das acções ocorridas no passado, o ambiente modifica-se enquanto o agente está executando o seu processo de inferência, as suas acções são expressas em intervalos contínuos (por exemplo, a aceleração), e os estados futuros do ambiente não são determinados somente pelas acções dos agentes, outros eventos ocorrem também podendo provocar alteração no ambiente.

6.1.2 | Autonomia em Humanos Virtuais

Humanos virtuais podem ser considerados como entidades de software com graus variáveis de autonomia, inseridos em ambientes virtuais imersivos ou não. As suas características e os seus movimentos simulam basicamente o comportamento humano, não se tratando de tentar criar um ser humano computacional "à imagem e semelhança do real", mas gerar entidades que conduzam o utilizador à ilusão e sensação, de interagir ou visualizar uma determinada personagem ou um conjunto destas, com comportamentos semelhantes aos reais.

Como visto anteriormente, quando as acções de um humano virtual são controladas total ou parcialmente pelo utilizador, diz tratar-se de um avatar, quando completamente "controlado" por software, diz tratar-se de um agente autónomo ou simplesmente agente virtual, sintético ou digital.

Os actores sintéticos exigem modelação a vários níveis, que vão desde a representação e animação de corpos individuais, até à movimentação de grandes conjuntos de agentes, formando multidões, que se devem comportar ao mesmo tempo de forma natural (evitando obstáculos e outros agentes, seguindo caminhos plausíveis, terem comportamentos adequados ao ambiente onde se inserem, etc), mas não necessariamente padronizada, gerando um conjunto de individualidades naturalmente heterogéneo.

As características que permitem verosimilhança comportamental a um ser humano virtual poderão ser divididas em:

a) Aspectos dinâmicos: a movimentação de uma ou mais personagens numa cena virtual envolve escolhas de caminhos, prevenção de colisões, etc, que exigem um planeamento inicial básico, submetido a objectivos de mais baixo nível;

b) Aspectos expressivos: as expressões faciais e os movimentos do corpo têm grande importância na transmissão de estados emocionais e intenções, sendo fundamental na capacidade comunicativa da personagem virtual;

c) Aspectos cognitivos e emocionais: personagens verosímeis devem apresentar um certo nível de autonomia, imprevisibilidade, intenções e metas próprias, podendo apresentar diferentes personalidades;

d) Aspectos sociais: quando apresentadas em grupo, as personagens devem comportar-se segundo regras de interacção, dependentes da cultura particular e individual que representam.

Neste seguimento, segundo [42] e [150], as propriedades fundamentais a incluir num agente representativo de um humano virtual seriam:

- Actividade: Poder iniciar interacção, ter comportamento determinado autonomamente, apresentar-se com credibilidade.

- Interactividade: entender as requisições e situações ambientais, saber responder a esclarecimentos sobre questões feitas por ele, apresentar iniciativas mistas entre o agente e utilizador.

- Reactividade: responder imediatamente a interrupções ou situações críticas imprevisíveis, saber pedir esclarecimentos, saber fazer criticas.

- Pró-Actividade: antecipar necessidades, adoptar as suas metas e fornecer comentários não solicitados.

- Contínuo Temporal: o agente é um processo que está continuamente em execução.

- Comunicação: a capacidade de trocar informações com outros agentes computacionais e com utilizadores humanos.

- Adaptação (aprendizagem): habilidade de extrair conhecimento a partir de experiências anteriores e adaptá-las sucessivamente ao seu comportamento perante o ambiente.

- Carácter: esta propriedade refere-se aos agentes que possuem uma personalidade e um estado emocional.

À tentativa de caracterizar e implementar a autonomia num contexto artificial, foi associado o termo agente na simulação de humanos reais, no seu sentido lato mas geral de "sistema artificial com capacidade de percepção e acção", possuidor de comportamento construído pela própria experiência. O ideal de um agente racional é que ele aja de forma a maximizar o seu desempenho a cada vez que o agente percebe o ambiente [149].

A inteligência artificial utilizada nestes agentes, inclui o planeamento, raciocínios e decisões para as tarefas em questão, além do processamento de linguagem natural para a compreensão das questões e necessidades do utilizador, e a consequente produção de respostas plausíveis.

6.1.3 O Conceito de Grupo

A evolução sobre o comportamento do grupo é uma questão central na evolução biológica e no estudo da evolução comportamental e a sua compreensão, e, consequentemente também, na simulação de um grande número de agentes representando humanos virtuais. Etólogos apresentam custos e benefícios sobre a vida em grupo, e tentam perceber o modo como estes factores interagem no contexto envolvendo populações.

Um grupo não é somente um aglomerado de pessoas que, por qualquer razão, se encontra num determinado lugar. Mesmo que no momento e por alguma razão haja algum tipo de interacção, essa não é uma condição suficiente e necessária para que o possamos considerar. Considerar um grupo como sistema é, do ponto de vista epistemológico, um recurso a que o cientista lança mão para melhor compreender os factos da sua observação. Este sistema existe porque há um conjunto de partes diferenciadas que em determinado momento se juntaram e se organizaram para atingir determinada finalidade. Se acontecer algum fenómeno a uma das partes que constitui o sistema, por certo que as outras partes vão ressentir-se disso e, por consequência, todo o sistema irá reagir.

Por exemplo, consideremos as vantagens térmicas que os animais de sangue quente têm mantendo-se juntos, as vantagens hidro-dinâmicas para os peixes que navegam em cardumes, bem como as vantagens de aerodinâmica e protecção para as aves em bandos, o risco do aumento da incidência de doenças em multidões, bem como muitos outros riscos e vantagens da vivência em grupo [151].

Deverá haver assim diferenciação em definir e simular comportamentos de grupo e comportamentos em multidão.

6.1.4 IA na Simulação de Comportamentos em Multidão e Grupo

A modelação e simulação baseada em agentes têm-se mostrado uma abordagem interessante para investigar sistemas mais complexos. É a área de sistemas multi-agentes (SMA) que estuda o comportamento de um grupo organizado ou não, de

agentes autónomos que cooperam entre si para realizar actividades de resolução de problemas que estão além das suas capacidades individuais.

Conseguir um modelo de simulação, de modo geral, consiste em reproduzir artificialmente uma situação ou um fenómeno natural. Os modelos de simulação baseiam-se na ideia de que programas informáticos podem exibir o comportamento do sistema real e desta forma, transportar um modelo do mundo real para um artificial, no qual hipóteses específicas podem ser exploradas, possibilitando o processo de resolução de problemas em ambientes de aprendizagem. A simulação pode assim ser usada para a realização de experiências de maneira análoga à que realizamos num laboratório real.

Ao contrário dos animais, que por norma não se movem aleatoriamente mas são guiados pela sua percepção do ambiente ou de outros animais [152], os agentes representando humanos virtuais, podem perceber outros agentes dentro de seu campo de visão, e interagir com eles de diversas formas. Os agentes virtuais usam as suas individualidades como a sociabilidade, a comunicação, o conforto, a percepção, a memoria e as suas habilidades para se comportarem ou simplesmente se movimentarem. Depois de se agruparem, eles agem de acordo com certos modelos de partilha e competitividade.

Concentrados agora em exemplos de modelação de comportamentos com um grande número de personagens virtuais, vemos que por norma, são introduzidos conceitos de multidões hierárquicas formado grupos e individualidades, com diferentes características e aspirações, como por exemplo se pode verificar em [153]. Neste campo, podemos também encontrar modelos baseados unicamente em comportamentos mais essenciais ou primários presenciados na natureza da convivência em multidão ou grupo. Um dos exemplos deste tipo de aplicações pode ser verificado em [154], onde são aplicados grupos de comportamentos básicos aos agentes virtuais ou a robôs móveis, tais como: evitar colisão, seguir outros agentes, formar grupos e dispersar. Existe também a possibilidade de agrupar alguns destes comportamentos gerando outros mais complexos.

A simulação de agentes virtuais em ambientes que podem ser modificados em tempo real (dinâmicos) é outra vertente de estudo. Aqui, são utilizados modelos dinâmicos e não lineares (como pode ser comprovado em [155]) de definição de rotas dinâmicas, juntamente com comportamentos de relacionamento em grupo, de modo a simular

um grande número de agentes. Estes modelos contemplam normalmente agentes com comportamentos de baixo-nível, para evitar colisões e encontrar um caminho que conduza a um objectivo que pode ser fixo ou móvel. Os agentes são dotados de orientação, e interagem com o ambiente através do conhecimento da posição e velocidade dos objectos.

Para simular comportamentos mais complexos e autónomos, em ambientes dinâmicos ou não, são utilizados modelos para simulação de multidões baseados na combinação de regras e máquinas de estado finitas (FSM[18]), para controlar o comportamento dos agentes dentro do conceito de multi-camadas. Em cada nível destas camadas são controlados tipos de comportamentos específicos e diferenciados, resultando o conjunto de comportamentos da totalidade das camadas, um conceito escalável e poderoso de simulação. Nos níveis mais altos, as regras seleccionam comportamentos complexos baseados nos estados dos agentes e do ambiente. Nos níveis baixos, comportamentos complexos são implementados por máquinas de estado finitas. Exemplos deste tipo de modelos podem ser vistos por exemplo em [156] e [5]. Um sistema de classificação dedicado à simulação comportamental com entidades virtuais foi também desenvolvido por Sanza et al [157]. As principais características deste sistema são a habilidade de gerar comportamentos autónomos e adapta-los às mudanças do ambiente permitindo as entidades progredirem automaticamente.

Em simulações de influência directa entre indivíduos, como por exemplo a competitividade entre ambos, encontramos trabalhos como o modelo competitivo Lotka-Volterra [158] [159], com demonstrações por equações diferenciais ordinárias para um número de indivíduos de duas espécies competitivas. Sem a presença de outra, cada espécie desenvolve-se de acordo com um crescimento linear e uma taxa de mortalidade quadrática, onde a ultima depende da capacidade de subsistência do ambiente para a espécie correspondente. A influência de uma espécie por outra é dada por um termo de produto cruzado que representa o efeito competitivo entre elas.

Cada vez mais, programadores fazendo uso das experiências e dos resultados atingidos em investigações na simulação comportamental de humanos virtuais em grupo ou multidão, constroem ferramentas de software facilmente integradas em

[18] Finite State Machine

aplicações gráficas específicas de animação e geração de modelos virtuais, de forma a poder integrar naturalmente autonomia individual ou em grupo às animações 2D ou 3D geradas para os mais variados propósitos. Seguidamente, são descritas algumas destas principais aplicações comerciais, que de forma isolada ou integradas, se dedicam à simulação comportamental de humanos virtuais em grupo ou multidão via animação por computador:

- Softimage/Behavior (http://www.softimage.com/products/behavior/) – Um exemplo deste tipo de ferramentas é o *toolkit* focado em animação por computador que faz uso de FSMs hierárquicas de modo a permitir melhor representação e controlo de sistemas em tempo-real, em multidões complexas, permitindo por exemplo que o animador forneça "inteligência" à multidão e a personagens individuais inseridos nela. Esta ferramenta também é provida de um suporte visual para a criação e edição de FSMs, suporta também uma linguagem de script para aumentar sua flexibilidade.

- BlenderPeople (http://www.harkyman.com/bp.html) – Blender People é um conjunto de aplicações desenvolvidas em Python scripts para a aplicação Blender, que em conjunto com uma base de dados mySQL, permite a geração em larga escala da dinâmica de multidões, incluindo (mas não limitado) a cenários de combate. O BlenderPeople 0.8 é a última *release* sobre o conceito de software livre – GNU LGPL.

- NetLogo (http://ccl.northwestern.edu/netlogo/) – NetLogo é uma plataforma de programação em multi-agente, e modelação de ambientes. NetLogo é da autoria de Uri Wilensky (1999) e está em constante desenvolvimento com ligações directas ao CCL (Center for Connected and Learning). NetLogo permite através da aplicação HubNet a participação e elaboração de sistemas de simulação de agentes.

- Massive (http://massivesoftware.com/) – Um dos pioneiros em sistemas 3D de animação para a geração de efeitos visuais em filmes e televisão provendo algum tipo de autonomia às personagens, reconstruindo grandes grupos de personagens virtuais com alguma independência e reutilização, ficando famoso pela recreação virtual da trilogia do Senhor dos Anéis.

- CrowdIT (http://www.crowdit.worldofpolygons.com/) – A gestão de diferentes ciclos de emoções para a animação de multidões e a habilidade para os usar e aceder aos seus estados de modo lógico e natural é uma das tarefas mais importantes na animação de agentes em multidão. O Crowd IT gera com facilidade bibliotecas de animações, armazena-as em ficheiros de dados e efectua a sua reutilização para os diferentes agentes numa multidão. Esta aplicação é desenvolvida para as mais recentes versões do 3D Studio Max.

- Micro-PedSim (http://people.revoledu.com/kardi/research/pedestrian/) – Denominada como ferramenta inovadora para pesquisa microscópica em SMA (sistema multi-agentes) para simulação de tráfico e agentes pedestres. O output da simulação poderá ser apresentado em simulações em tempo real, em visualizações 3D, traçagem de percursos, ou medidores de performance que poderão ser exportados e apresentados em MSExcel ou Matlab.

- SpirOps Crowd (http://www.spirops.com/) – SpirOps é um desenvolvimento de Middleware com principal aplicação em experiências de inteligência artificial e simulação de percursos de agentes. Principalmente usada em jogos de computador.

No entanto, estas aplicações (que maioritariamente são proprietárias), não resolvem eficientemente ou de todo, e de uma forma escalável (e por vezes "amigável"), o problema traçado inicialmente, ou seja, lidar com uma grande quantidade de agentes em interacção, simulando eficientemente colisões, gerando o planeamento de rotas, e automatizando comportamentos autónomos em ambientes realmente 3D, desconhecidos à partida.

6.2 | Arquitectura de Suporte a Comportamentos de mais Alto-Nível

Para os propósitos definidos nesta tese, o modelo de comportamentos de mais alto-nível associado a pedestres virtuais irá surgir assente na base das sequências das investigações e enquadramentos anteriores, utilizando comportamentos globais, e permitindo também ser decomposto em unidades mais elementares de

comportamento individual. Nomeadamente, contará com as seguintes especificações de base:

- Lidar com vários níveis navegáveis de terrenos sobrepostos em altura.
- Evitar obstáculos, permitindo ao agente evitar a colisão como objectos estáticos.
- Atingir objectivos, conduzindo o agente até ao destino.
- Evitar pedestres, o que permite ao agente evitar a colisão como outros agentes em deslocação.
- Ter um comportamento específico, associado a outros agentes, grupo de agentes, objectos ou localizações em particular.

Estas cinco unidades deverão produzir um *output* de base "estável" na forma como serão idealizados e influenciados à posteriori os respectivos movimentos e comportamentos das personagens virtuais. Esta aproximação é similar às regras usadas nos boids, onde a combinação dos *outputs* é uma obtenção simples de um conjunto da soma de forças. O modelo comportamental implementado incorpora requisitos (na mesma linha de investigação) de eficácia e eficiência, e leva em linha de conta os vários níveis navegáveis de terrenos sobrepostos em altura, não gerando conflitos e impasses na geração de rotas quando estabelecidas neste tipo de ambiente. Este tipo de interacção deverá ser alargada a qualquer tipo de ambiente, diferenciando autonomia individual e de grupo, bem como a distinção do tipo de personagens na simulação de acordo com os seus objectivos iniciais, e o histórico da sua actuação no ambiente de simulação.

6.2.1 Geração de Personagens Virtuais Representativas de Agentes

A aplicação resultante foi prevista para incorporar um grande número de personagens (até 10.000 agentes, sem perder realismo). Cada um destes agentes teria de ser independente nas decisões que tomasse (comportamento individual), bem como ser capaz de se enquadrar numa comunidade de entidades similares, regendo-se através das regras sociais e de grupo (comportamento de grupo) dessa comunidade. Antevendo cada um dos estados destas entidades, estes teriam de ser mantidos em memória, com capacidade de gestão de informação associada às suas próprias características individuais e de grupo, actualizando o seu estado actual nas mais

diversas situações envolvidas em tempo real. Todos estes estados terão um papel importante na "disposição mental" do agente, influenciando a tomada de decisões em relação ao seu comportamento e interacção num mundo partilhado com outros agentes virtuais. A concepção destas entidades foi gerada a partir da biblioteca Cal3D [142] de Bruno Heidelberger.

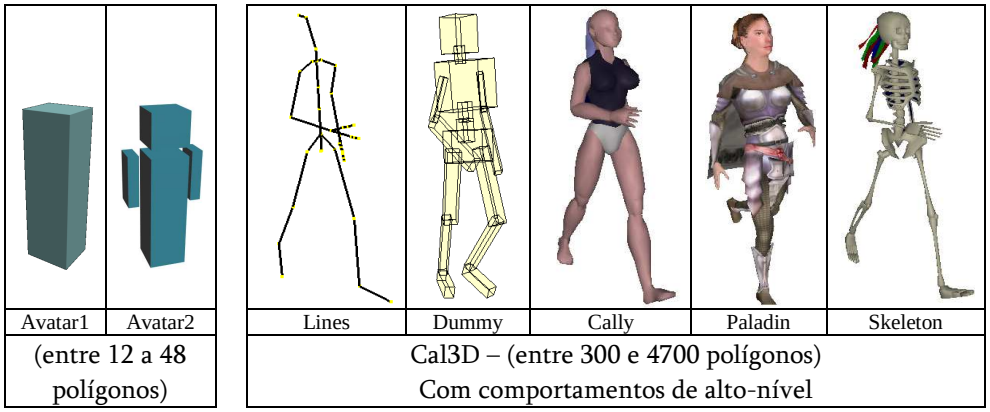


Figura 68 – Personagens virtuais disponíveis nas simulações

Durante o carregamento da aplicação, é apresentado ao utilizador a decisão do tipo de agentes a envolver na animação (Figura 68), bem como o número destes. Se o número escolhido ultrapassar um determinado valor pré-estabelecido, que tem em linha de conta a dimensão do mundo já pré-carregado e o garante da qualidade da simulação prevista, é diminuído o nível de detalhe escolhido para todos os tipos de agentes a carregar, para que a animação se mantenha fluida e realista. Outra opção para estes casos, é a aplicação poder incorporar personagens (neste caso chamados de avatares) mais simples, com um número muito reduzido de polígonos com comportamento limitado (baixo-nível) perante os outros agentes (resumindo-se à navegação e detecção de colisões no mundo virtual e inter-agente).

De modo a definir um ponto de partida para a colocação das personagens virtuais a incorporar no ambiente de simulação, foi estabelecido que todas elas seriam inicialmente posicionadas aleatoriamente na base navegável do mundo a povoar. Em consequência, foi construído um módulo de alocação e localização inicial em altura dos agentes, este, estabelece uma leitura do mundo carregado a partir da aplicação GLM (já mencionada no capítulo 4) onde é estabelecido o patamar de colocação inicial.

De modo a evitar a sobreposição na colocação entre agentes, e com objectos existentes na cena sobre o plano estipulado de colocação inicial, este módulo detecta a posição inicial de todos os agentes e compara com a mesma posição do mapa de alturas no nível de fatiação mais baixo e inicial de teste de colisão para todos os agentes. Se a posição de algum destes coincidir com a posição de alguma célula com uma altura elevada e inultrapassável para os agentes, então é gerada uma nova posição aleatória para este agente, e logo após, testada do mesmo modo, e assim sucessivamente até se encontrar uma posição inicial aceitável (Figura 69).

Após se estabelecer a colocação inicial de todos os agentes em altura, um novo módulo posiciona-os, ou aleatoriamente em x,y e z no plano encontrado (Figura 69), ou em locais específicos a definir pelo utilizador da aplicação.

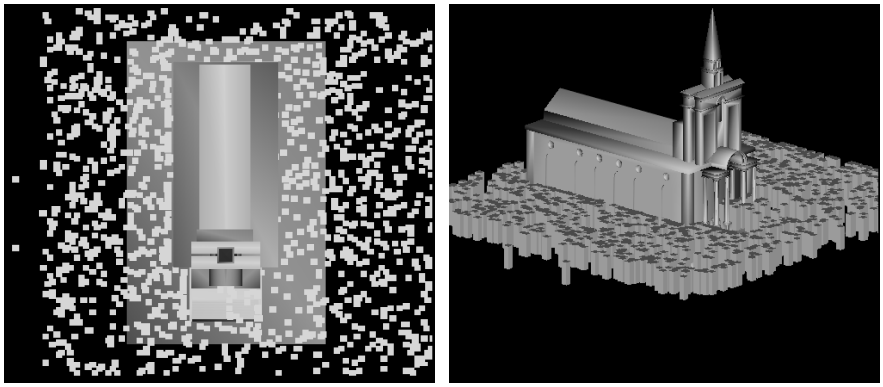


Figura 69 – Exemplo da colocação automatizada inicial de 2000 agentes (tipo Avatar1) no modelo Igreja

6.2.2 | Gestão da Ocupação Espacial dos Agentes

De forma a permitir conhecer a localização global inter agentes e, por consequência, garantir uma navegação na qual estes não colidam, prevendo essa situação antecipadamente, e alterando as rotas de navegação de acordo, bem como estabelecer regras ou comportamentos associados à proximidade com outros agentes (do mesmo grupo social ou não), é fundamental manter em memória a sua localização de ocupação espacial em tempo real.

Esta ocupação requer alocação dinâmica com grande volatilidade, já que durante o caminhar de um agente, as várias localizações espaciais por onde este passa serão

conotadas com valores de ocupação quando este se encontrar sobre elas, e imediatamente identificadas como desocupadas quando o agente tiver deixado estas células. Haverá assim a necessidade de manter um ciclo de teste de ocupações do espaço disponível à navegação durante a simulação, com a função de controlar estas situações.

Todo este processo além do objectivo principal de gestão da ocupação do espaço, manterá também, registos de valores identificativos para a distinção do tipo de agente aí presente. Um exemplo simples deste processo poderá ser observado na Figura 70, onde o agente "Cally" se encontra em deslocação desde a posição A até a posição B. Os espaços a cinza-claro representam o trajecto por este utilizado. Os espaços ocupados actualmente são representados a negro para os agentes "Paladin" e a cinza escuro para a posição presente do agente em deslocação "Cally".

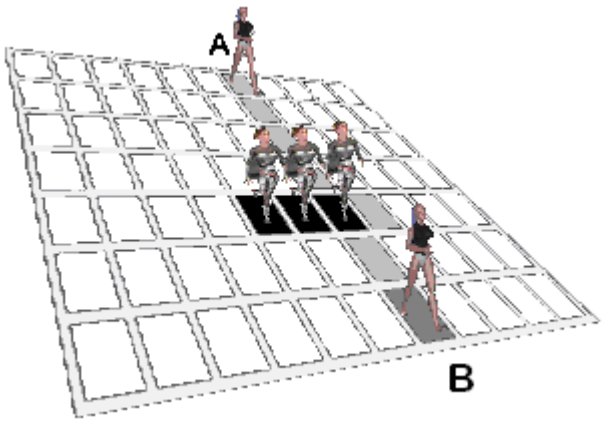


Figura 70 – Matriz de ocupação espacial de agentes

A aplicação deste mecanismo assenta numa composição de células de memória, organizadas numa matriz multi-dimensional de nome "*Walkability*" de 1 byte de ocupação por célula, onde valores 0 representam espaços vazios e de 1 a 7 as várias ocupações dos diferentes tipos de agentes envolvidos na simulação (num primeiro caso de testes até 7 personagens distintas em simulação). Este valor de memória poderá no entanto ser aumentado para abarcar populações mais diversificadas.

Em relação ao possível contacto inter-agente aquando da deslocação pelo ambiente, existem 3 possibilidades de relacionamento global dentro do modelo definido:

1. Dois agentes podem avaliar as suas posições e decidir qual deles deverá alterar a rota inicialmente traçada de modo a evitar a colisão.

2. Um agente pode esperar por outros mais atrasados, se ambos fizerem parte do mesmo grupo.

3. Um agente ao conhecer outro, pode avaliar os seus parâmetros sociais, e agir de acordo com estes em relação ao novo agente. Neste caso, grupos e comportamentos associados passam a ser dinâmicos (como se pode comprovar em exemplos detalhados mais adiante em testes de simulação comportamental).

Para poder simular com uma versatilidade e diversidade populacional superior, as células constituintes da grelha de navegação terão de ser reduzidas em tamanho ao menor valor possível, o que no limite corresponderá a um píxel. Isto provoca a necessidade de ter matrizes maiores e consequentemente maior ocupação e gestão de memória. Foi decidido tornar esta decisão flexível, e poder especificar esta condição (memória *vs* versatilidade) aquando a importação do ambiente 3D (também de acordo com os agentes a serem incluídos).

Com uma grelha de células regular reduzida aos píxeis, podemos simular agentes de todos os tamanhos de uma forma precisa. Podemos ter por exemplo, elefantes e ratos como personagens virtuais de interacção no ambiente 3D, cada um deles ocupando espaços diferentes e precisos na grelha de navegação e correspondentemente no ambiente 3D, transportando as necessidades de implementação realista que isso acarreta. Este valor é estabelecido pela variável inicial «largura do agente» que estabelece a quantidade de píxeis (células) que cada agente em particular irá ocupar na grelha.

É importante recordar que os mundos a simular serão desconhecidos à partida, possibilitando também uma enorme diversidade de ambientes. A grelha de navegação terá de se adaptar ao mundo virtual e não o contrário, daí, células reduzidas a píxeis, representarem uma estrutura de interesse superior de modo a caracterizar o espaço de navegação com elevada precisão. Veja-se as diferenças de deslocação na Figura 71 entre dois casos similares com agentes de diferentes tamanhos, utilizando grelhas de navegação também com diferentes tamanhos de células.

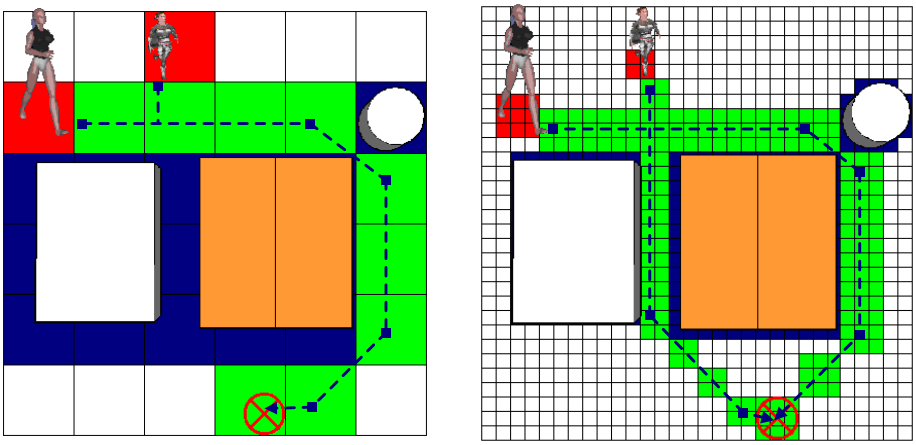


Figura 71 – Grelhas de navegação com diferentes dimensões de células

Nas duas situações para o mesmo ambiente virtual e para a mesma situação, podemos observar a deslocação por caminhos ou rotas diferentes, onde os dois avatares de tamanhos distintos se direccionam para um objectivo comum. Após o enquadramento das diferentes grelhas de navegação à mesma cena, são detectadas quais as células livres e aptas à navegação (coloridas a branco), e as ocupadas e inabilitadas à navegação (a negro). Após esta verificação, são traçados os percursos (a cinza) sobre as células a branco. Pela observação do percurso representado no exemplo da figura anterior, e para o mesmo objectivo espacial (representado com um X), é possível identificar que a grelha com o tamanho de células mais reduzido, proporciona a representação de condições de navegabilidade mais realista, optando sempre para os vários tipos de agentes associados, pela navegação pelos percursos mais directos e mais rápidos quando estes estiverem disponíveis.

6.2.3 Níveis de Interacção dos Agentes

Na relação entre comportamentos de baixo e alto nível, podem ser identificadas várias etapas de abstracção. Em cada uma destas etapas serão introduzidas faculdades cada vez mais evoluídas às entidades manipuladas. A Figura 72 mostra esta interligação, onde os sucessivos níveis de abstracção introduzem crescentes aptidões às entidades manipuladas, que nesta etapa, serão consideradas de agentes.

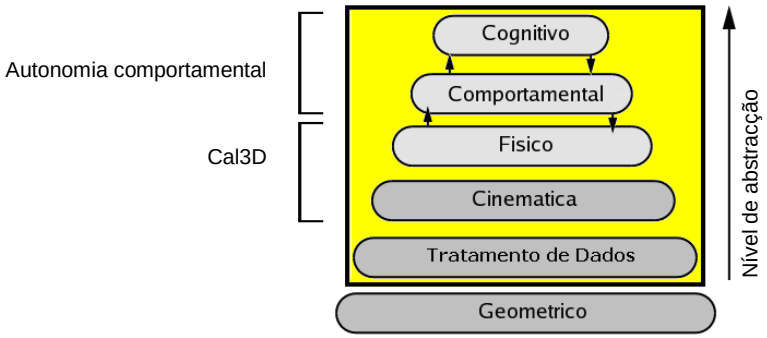


Figura 72 – Interligação entre níveis ou camadas de aptidões comportamentais

As camadas inferiores, física e cinemática, foram implementadas na biblioteca Cal3D. As animações resultantes são disponibilizadas às camadas superiores numa forma elementar, por exemplo, a deslocação bípede (o caminhar). O domínio principal agora é então, formado pelas restantes camadas, cognitiva, comportamental e física (para ligação aos níveis inferiores). Cada nível de abstracção proporciona informação ao seu nível superior e, de modo semelhante, cada nível controla o seu nível inferior.

Os dois níveis superiores, traduzidos neste capítulo em implementação de certo grau de autonomia e reacção comportamental, são representados em decisões, com a definição de objectivos, e execução de acções associadas, bem como a definição de métodos de apreensão de conhecimento do mundo onde o agente está inserido. Esse conhecimento é logicamente tido em conta pelo nível comportamental.

Detalhando, o nível físico engloba o agente composto por sensores e actuadores. Aqui se estabelece o diálogo com o ambiente com base na dualidade percepção/acção. A este nível incorporam-se os processos comportamentais, introduzindo a actividade que converte o agente físico num agente situado. Cada comportamento está ligado a um conjunto de características extraídas dos sensores (os estímulos) e determina a acção a gerar, dependendo da configuração particular dos estímulos. O comportamento é portanto, um ciclo sensitivo-motor que liga a percepção à acção (internamente) e a acção à percepção (externamente) através do ambiente. Finalmente, o nível cognitivo contém o conhecimento de vários processos que estruturam o conhecimento e gerem os comportamentos do nível comportamental. Este nível introduz a capacidade de racionalização e converte o agente situado num agente cognitivo, capaz de satisfazer os critérios de avaliação. Muito sucintamente um agente cognitivo pode ser pensado, como um sistema de

estruturas auto-organizativas que se comunicam com o mundo externo através de determinadas interfaces especializadas, também por vezes denominadas de esquemas. Cada uma das interfaces do agente pode especializar-se em tratar os diferentes aspectos do meio ambiente onde o agente está inserido.

6.2.4 | Definição e Estruturação de Interacções da Arquitectura

Podemos agora construir um diagrama que represente a personagem virtual e o seu comportamento reactivo, ou seja, o modelo 3D situado no mundo, que toma determinadas acções baseadas nos seus objectivos, nos seus conhecimentos e nos estímulos recebidos nesse mesmo mundo. De modo genérico, este diagrama chamado "Módulo Agente", pode ser representado pelo seguinte esquema:

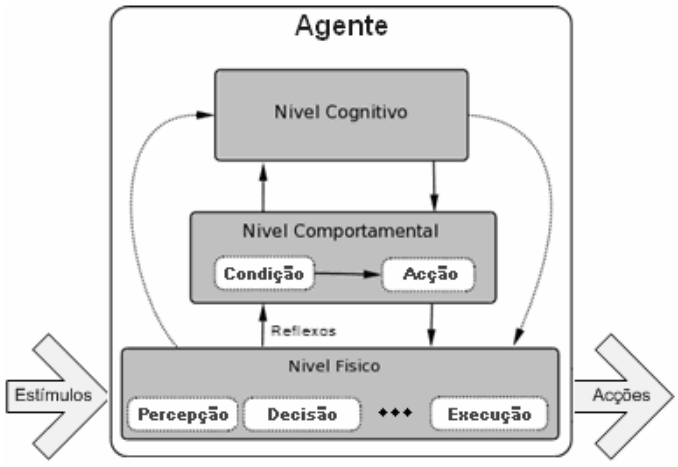


Figura 73 – Módulo Agente – Visão geral do sistema comportamental para cada agente

Neste módulo representado pela Figura 73, podemos observar os fluxos de informação para e pelo agente, como ela é obtida, o seu processamento e as consequências desse processamento. A título exemplificativo tomem-se as seguintes situações, integrando algumas etapas do processo de resposta a partir do Módulo Agente:

1. O agente é estimulado: surge no seu campo de percepção outro agente.
2. De um modo reflexivo, o agente toma a acção de alterar o seu percurso de modo a evitar tal obstáculo.

3. O agente é seu conhecido e sendo assim cumprimenta-o, tomando a acção de acenar com o braço direito.

Deste modo, um agente pode ser visto, conceptualmente, como uma função de domínio Estímulos e contradomínio Acções:

$$f_{agente}(Estímulos) = Acções$$

Equação 4 – Definição da função comportamental estímulos *vs* acções

Há pois uma separação clara entre o meio exterior, de onde recebe e actua, e o meio interior de processamento, entre o estado externo e o estado interno. O estado externo é apreendido e o estado interno com propriedades como cansaço, simpatia, agressividade, são processados pelo nível comportamental e cognitivo (com conhecimento mais complexo como objectivos, emoções, memória, etc.), e quando determinadas condições são satisfeitas as acções apropriadas são executadas, por exemplo, "se um sinal luminoso está vermelho, então não atravessar a rua".

Refira-se ainda a existência de comportamento consciente e inconsciente. O anterior ponto 3 dos fluxos de informação para e pelo agente, é um exemplo do primeiro tipo, enquanto que a situação no ponto 2 é um exemplo de comportamento inconsciente, reflexivo.

O domínio desta etapa é então formado pelos níveis de abstracção de ordem cognitiva, comportamental e física, onde são definidos métodos de apreensão do conhecimento do mundo onde a personagem virtual está inserida. Este conhecimento do mundo e nível de suporte a comportamentos de mais alto-nível pode ser representado pelo esquema da Figura 74, em que comportamentos de baixo-nível serão projectados para representar uma arquitectura capaz de estabelecer a ponte com comportamentos de mais alto-nível definidos nas secções seguintes.

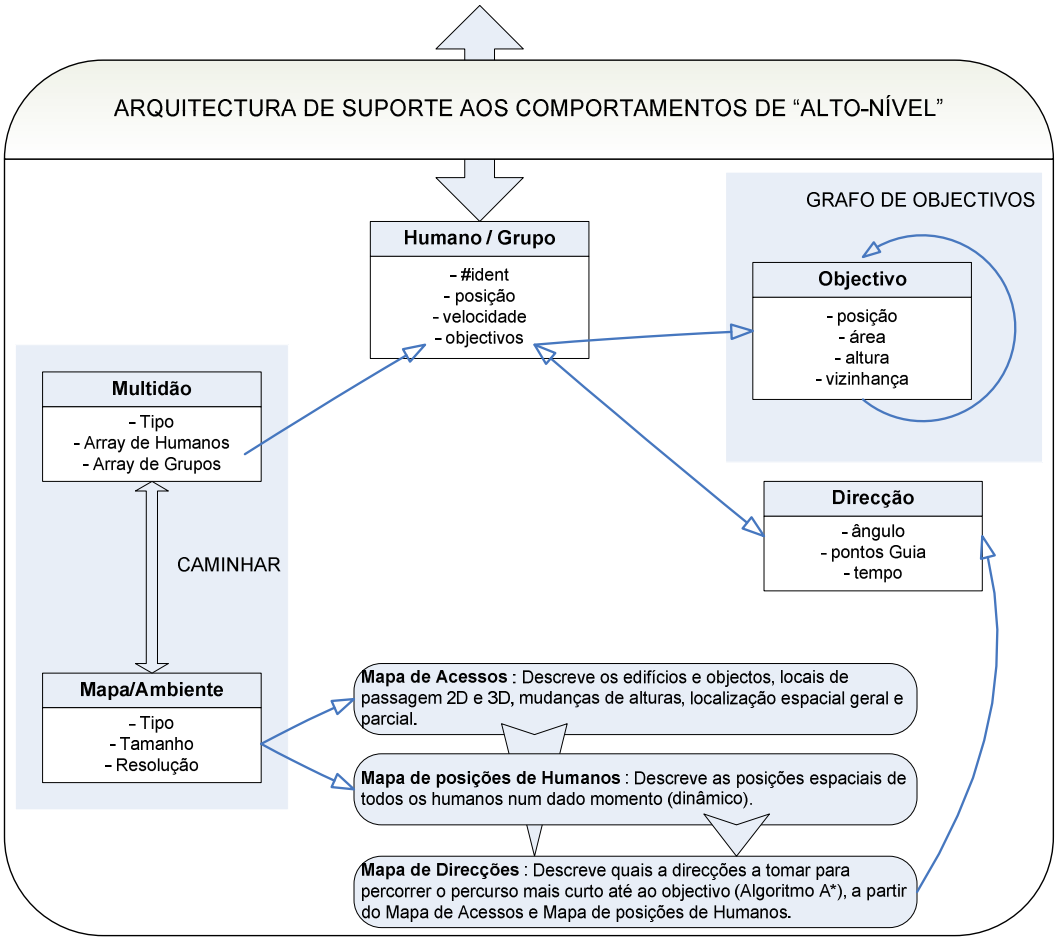


Figura 74 – Base da plataforma agregando comportamentos de baixo-nível

Esta arquitectura direcciona todas as pesquisas de baixo nível para o módulo Humano/Grupo, tais como o conhecimento do ambiente envolvente para detecção de colisões com o ambiente, localização global inter agentes para conseguir garantir uma navegação livre de colisões inter agentes, detecção de objectivos de deslocação espacial, estabelecimento de forma dinâmica do traçado de rotas de deslocação tendo em conta, por exemplo, a direcção, pontos de guia, e o tempo.

6.3 | Comportamentos de Alto Nível – Máquina de Estados

Na maioria das ocasiões, os agentes não são desenvolvidos de forma independente, mas como entidades que constituem um sistema. Este sistema é denominado normalmente de sistema multi-agente (SMA) [160]. Neste caso os agentes devem ou podem interactuar entre si. As interacções mais habituais como são as de informar ou consultar outros agentes, permitem troca de informação entre eles, perceberem

aquilo que cada um deles realiza, e julgar ou raciocinar acerca do papel em jogo pelos diferentes agentes que constituem o sistema. A comunicação entre agentes realiza-se por meio de uma linguagem de comunicação denominada (*ACL –Agent Communication Language*).

Os SMA (Sistemas Multi-Agentes) possuem habitualmente uma complexidade estrutural e de funcionamento considerável, sendo normalmente impossível determinar à partida, o conjunto de comportamentos e as actividades concretas que irão ser executadas pelo sistema. Entre exemplos sobejamente conhecidos de investigação e desenvolvimento abarcando este tipo de conhecimento, encontram-se os trabalhos de Nick Jennings, Mike Wooldridge, Gerhard Weiss, G. O'Hare, entre muitos outros, onde se destacam os seguintes trabalhos dos mesmos autores [161-163].

Independentemente de serem colaboradores ou competidores, os agentes que integram um SMA irão interagir uns com os outros – um requisito operacional mínimo. Esta interacção não é indesejada, mas sim intrínseca ao próprio conceito de agente que pressupõe a sociabilidade do SMA como forma de um agente atingir os seus próprios objectivos. As interacções pressupõem que os agentes conheçam outros agentes presentes no ambiente, ou pelo menos, que estejam a par da sua existência, ou seja, que os agentes sejam projectados como verdadeiros membros de uma sociedade multi-agente. No entanto, é desejável que estas interacções sejam convenientemente coordenadas pois caso contrário a sociedade de agentes pode degenerar numa sociedade completamente descoordenada e caótica.

Actualmente, uma das formas mais vulgares de implementar agentes reactivos baseia-se no uso de máquinas de estados [164]. Numa abordagem deste género é necessário descriminar e guardar os estados da mente do agente, e construir condições de transições entre estes. A forma mais popular de descrever uma arquitectura apoiada em máquinas de estados é recorrendo a autómatos finitos deterministas (AFD) [165] ou, mais especificamente, usando transdutores, que não são mais que um AFD em que a cada transição se pode associar um símbolo de saída [164]. Desta forma, torna-se mais fácil estruturar o estado de uma mente e simular comportamentos, pois a partir de cada estado, existe um conjunto de estímulos-resposta diferentes.

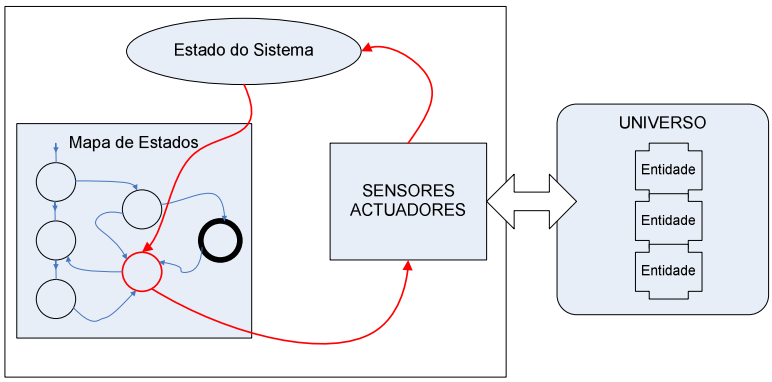


Figura 75 – Arquitectura reactiva de uma máquina de estados

Identificando uma arquitectura reactiva de uma máquina de estados, a Figura 75 ajuda a perceber todos estes processos de transferência de informação e reacção. São agora apresentados em detalhe os diversos procedimentos nos diferentes módulos:

- **Mapa de Estados** – Neste módulo está definido a máquina de estados. Assim, a cada momento os estímulos que vêm do exterior actualizam o estado interno do sistema, o qual pode mudar o estado actual do mapa, e de seguida são aplicados os estímulos ao estado actual, o qual devolve um conjunto de acções correspondentes.

- **Estado Sistema** – O estado do sistema pode ser qualquer conjunto de variáveis no qual o sistema se baseia para coordenar as transições e/ou estados do sistema.

- **Sensores e actuadores** – Interfaces com o mundo em que os sensores actualizam o estado do sistema e os actuadores executam as acções do sistema.

Um exemplo recorrente em agentes onde se podem identificar as vantagens do uso de transdutores é na representação de estados emocionais. Poderá ser aqui exemplificada uma possível transição de estados de um agente (Figura 76) que, partindo de uma situação de repouso, e ao receber um determinado estímulo ou estímulos, é efectuada uma transição para o estado que o estímulo originou.

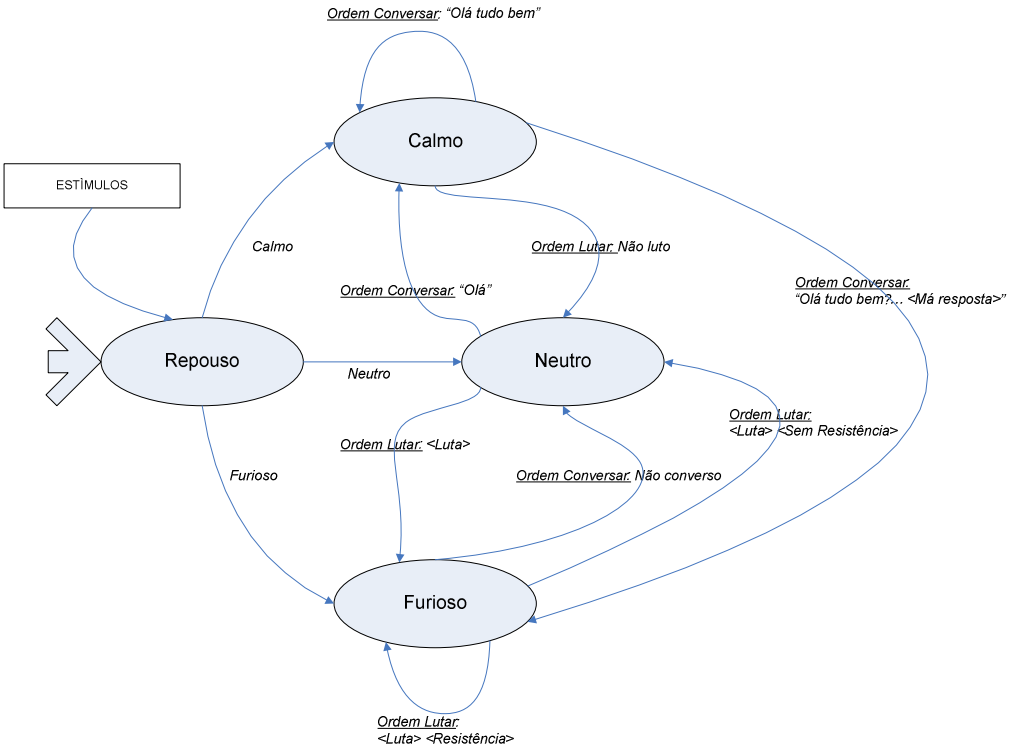


Figura 76 – Exemplo de um transdutor para representar estados emocionais simples

Focados neste exemplo concreto (Figura 76), e se a condição de um agente desde uma situação de "Repouso" evoluir para um estado "Calmo", e recebe um estímulo para conversar com alguém, ele continua calmo se a conversa correr bem, senão pode mudar para o estado "Neutro" ou "Furioso". Por outro lado, se receber a indicação para lutar com alguém, ele poderá devolver uma mensagem a dizer que não actuará de acordo, porque está calmo e relaxado e, possivelmente, mudará de estado. Dependendo do estado actual, também é possível ter várias respostas diferentes face a um mesmo estímulo. Este é um exemplo bastante simplista, mas, contudo, revela alguns dos estados modelados na aplicação prática deste trabalho usando transdutores.

6.3.1 Arquitectura Mental e Cognitiva

Na tentativa de alcançar um degrau significativo de simulação realista idêntica ao comportamento humano, objectivos mais avançados de IA apontam que deverão ser produzidos comportamentos credíveis para observadores *naïf´s*, e suficientemente convincentes para que os *experts* deste tipo de assuntos aceitem os comportamentos dos agentes como sendo naturais. Mas logicamente a avaliação da "humanidade" dos

agentes, poderá ser altamente subjectiva, e dificilmente avaliada numa escala fixa e pré-existente.

O desenho do sistema da arquitectura mental e cognitiva envolvida nas experiências práticas desta tese divide o progresso cognitivo em quatro módulos muito específicos: percepção, modelo mental, decisão de objectivos e a resolução da acção a tomar. Cada módulo tenta simular fraquezas e forças da percepção humana e o seu processamento cognitivo, incluindo reacções instintivas, erros de percepção, degradação da memória, decisões dependentes do contexto, inferência, entre outras.

A estrutura básica de um sistema de modo a poder controlar uma entidade virtual, deverá no geral, seguir o paradigma anteriormente referido de "percepção – decisão – acção" (ou o usado pelo exército "observação – orientação – decisão – acção"). Este modelo da fluência entre a informação e a acção conduz-nos ao esquema de interacção com o ambiente virtual descrito pela Figura 77.

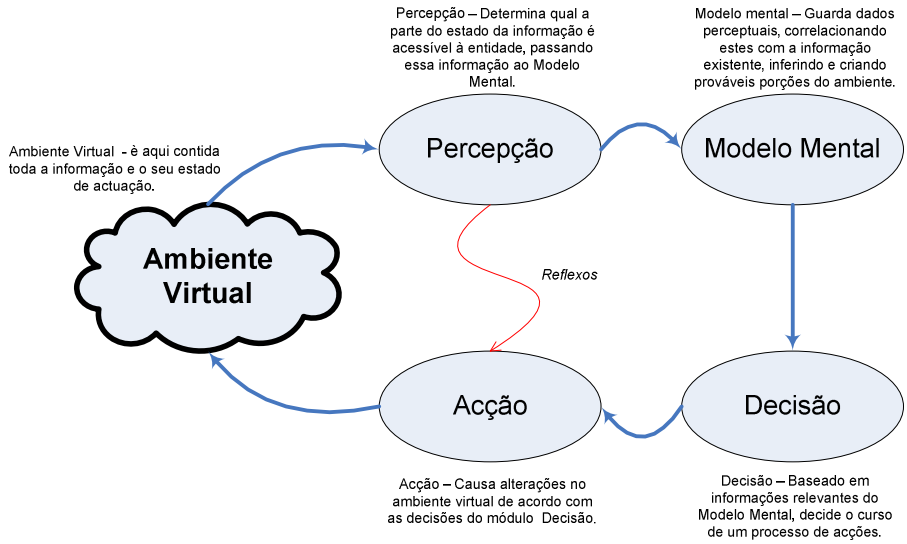


Figura 77 – Modelo 4 módulos de IA

São apresentados em detalhe as especificidades de cada um destes módulos, com especial realce nas implementações de potenciais decisões, problemas e possíveis soluções. A interacção entre os módulos e entidades será também alvo de análise.

Percepção:

Define o modo como cada entidade humana percebe o ambiente virtual. Em paridade aos sentidos humanos, é determinada que parte do estado da informação é acessível à entidade associada. Este modo pode ser circunscrito a alguns aspectos em que a componente máquina é limitativa.

Um sistema que simule a percepção humana terá também de comportar o erro. Mesmo faltando uma parte da informação, será possível com base nos detalhes já conhecidos elaborar objectivos e suposições (que serão em alguns casos falíveis), este é um dos propósitos da IA, estabelecer comparações com o comportamento humano.

A modelação de alguns tipos distintos de reacções deve favorecer um elemento adicional de realismo, o reflexo instantâneo, de modo a que os utilizadores obtenham um *feedback* imediato (em acções) que os componentes de IA deverão devolver de acordo com percepções específicas.

Modelo mental:

Por norma os humanos, baseiam as suas decisões em mais do que a imediata percepção de um dado acontecimento. Um completo modelo mental de todas as entidades envolvidas no ambiente seria o melhor método de proporcionar decisões de alto nível mental, mas proibitivo em termos de memória e tempo de processamento, necessário ao suporte das consultas.

A informação é perecível, por duas razões. Primeira, a informação de localizações, velocidade, e objectivos actuais é dinâmica. Consequentemente o modelo mental deverá degradar este conhecimento no tempo, tornando-o cada vez de menor importância ou confiança. A segunda razão é a memória não ser perfeita. Mesmo algumas qualidades estáticas como tipo de entidades, ou algum tipo de alinhamento poderão não ser perfeitamente recordados após passado algum tempo, especialmente se entretanto existiram muitas transferências de informações no modelo mental. A confusão de algumas destas memórias poderá ser usada em algumas situações.

A inferência também poderá ser um problema difícil de simular. Esta tende a ser mais falível do que a directa observação, por razões óbvias, e deverá ser dependente de conhecimento acumulado anteriormente.

Outro conceito que entra no modelo mental é a confiança, sendo uma boa solução para resolução de problemas como a inferência e a degradação da informação, bem como para modelar alguns traços de personalidade.

Finalmente, a partilha de informações entre as entidades é um objectivo também desejado. Sem este, acções de coordenação, partilha de objectivos, enriquecimento de conhecimentos sem a vivência, etc, serão impossíveis de simular e acrescentar ao modelo.

Decisões:

Após a concepção de uma "imagem" do ambiente envolvente, este módulo de IA toma decisões sobre as acções a realizar para alcançar os objectivos pretendidos. Estas decisões resultam da combinação dos estados emocionais das personagens virtuais, da sua percepção do ambiente e dos conhecimentos e modelos mentais adquiridos e organizados anteriormente.

Estas decisões deverão ser delineadas através da comparação entre todas as possibilidades disponíveis em cada conjuntura particular, podendo haver saltos de um estado para o outro muito diferentes, dependendo das situações. Isto deverá aumentar a flexibilidade do sistema permitindo codificar alguns estados intuitivos e emocionais característicos dos humanos.

Este tipo de configuração permitirá facilmente atribuir qualidades a agentes individuais tais como: bravura, agressividade, insegurança, precaução, cansaço, etc, de modo a criar um largo espectro de densidade de comportamentos individuais a partir de um pequeno número de propriedades e capacidades. Terá de haver também a simulação de decisões com base em repostas a estímulos externos específicos, como por exemplo: ameaças, vulnerabilidades percebidas, curiosidade, etc…

Acções:

Os objectivos, são na generalidade abrangentes na atitude necessária perante o mundo e o seu ambiente imediato. Uma vez tornado um objectivo activo, a IA necessita descrever quais as acções específicas apropriadas para a total ou parcial execução desse objectivo. É possível organizar uma lista de todas as acções praticáveis, devido às limitações lógicas e necessárias da relação dos agentes com o ambiente virtual escolhido. No entanto, uma escolha inteligente do subconjunto de combinações das acções elementares é por norma suficiente para prover os agentes

de uma flexibilidade satisfatória para lidar com o mundo. Existem dois processos que poderão utilizar-se aqui, que dizem respeito ao número de acções possíveis:

1. O primeiro utiliza uma longa lista provocando uma maior flexibilidade, onde o desenhador poderá implementar acções com combinações mais específicas e apropriadas à simulação. No entanto, um grande conjunto de acções requer um algoritmo muito detalhado, com dificuldade de escolha entre qual ou quais conjugações de acções usar em cada situação.
2. Por outro lado, uma lista pequena de acções para cada objectivo torna a programação e a análise de actuações muito mais fácil, e permite a simplicidade de actuação no ambiente virtual. Assim as reacções serão mais robustas e rápidas. No entanto alguns problemas poderão surgir: uma lista muito curta de possíveis acções torna-as muito previsíveis (poderá também ser vantajoso se quisermos observar facilmente padrões subjacente à simulação). Mas estes poderão violar alguns dos objectivos do projecto, pela forma de ser dos humanos "aparentemente" não determinista.

Módulo	Desenho e Implementação Inicial de Objectivos
Percepção	• Simulação sensorial. • Quantidade de informação recebida dependente do tempo e da localização. • Possibilidade de perceber falsa informação. • Interactivo com os modelos mentais. • Simulação de acções a partir de reflexos.
Modelo Mental	• Representação das outras entidades no ambiente. • Conhecimento dividido em porções individuais, utilizado em situações individuais. • Conhecimento com degradação temporal. • Simulação de inferências para necessidades simples (ex localização). • Representação e utilização da confiança em situações conhecidas. • Modelo mental partilhado entre elementos do grupo.
Decisão	• Lista de objectivos "instintivos". • Algoritmo para assignar pesos a cada objectivo. • Algoritmo de modo a escolher a acção apropriada para resolver determinado objectivo. • Esquema de re-arranjo temporal quando a situação se altera.
Acção	• Pequena lista de acções. • Complexidade média nas acções tomadas. • Muitas acções aplicáveis a mais do que um objectivo. • Feedback e reflexos em alguns objectivos.

Tabela 10 – Implementação inicial de objectivos de IA em agentes reactivos

6.4 | Configuração Comportamental

6.4.1 | Arquitectura Mental e Cognitiva

Na configuração da arquitectura mental e cognitiva aplicada aos agentes nos testes práticos levados a cabo nos trabalhos desta tese, cada personagem possuiu um estado interno representado por um aglomerado ou conjunto de propriedades pessoais. Estas propriedades usadas para modelar o seu comportamento, são automaticamente inferidas desde as condições iniciais da animação e actualizadas de acordo com a dinâmica da animação (por exemplo o fugir de um agente agressivo). Existem assim, um conjunto de propriedades idênticas, comuns a todas as personagens, como por exemplo: Posição; Orientação; Destino; Estado de movimento; Velocidade; entre outras.

Todo o agente disponibiliza estas propriedades permitindo a sua utilização na definição do comportamento. Cada uma destas propriedades é um identificador desse domínio com a correspondência de um valor atribuído numa escala pré-definida (neste caso em espaços decimais entre 0 e 1). A linguagem criada permite a (re)definição das propriedades de cada instância de um determinado tipo de agente do seguinte modo:

```
propriedades
{
        forca: 0.8;
        agressividade: 0.7;
        estado: 0; // repouso=0, calmo=0.1, neutro=0.2, etc
        cansaço: 0.2;
        inteligencia: 0.4;
        posicao_nodo_grafo: 8; //#ident do nodo no grafo de localização
        posicao_espacial_nodo: (900.0, 250.0, 10.0); //x,y,z
}
```

6.4.2 | Definição Comportamental

Esta é a parte principal da especificação da linguagem criada para suporte de definição comportamental. Aqui é estabelecido o modo como o agente interage com o mundo e os outros. As instruções definidas serão interpretadas criando-se uma estrutura de dados que constitui o "cérebro" do personagem. As várias instruções definidas utilizarão as propriedades internas e os estímulos apreendidos (mesmo

propriedades de outras personagens) e realizarão acções, representadas pelas animações previamente associadas. Existem três grupos fundamentais na definição destas acções comportamentais:

1. Scripts
2. Condições
3. Decisões

6.4.2.1 | Scripts

Scripts são conjuntos (identificados) de instruções sequenciais. Estas instruções são basicamente as acções que a personagem executará. São várias as instruções de script possíveis:

1. Scripts / Acções:

Acções previamente definidas podem ser executadas em pontos tempo/espaço arbitrários, ou específicos, de acordo com condições ambientais. Por exemplo despoletar a acção "*acenar".* Para além desta projecção de acções, existem algumas inerentes a todo o agente, nomeadamente a acção "*walk*" até ao ponto XYZ. Esta acção é responsável pela identificação da melhor rota até ao ponto especificado e pelo efectivo deslocamento do modelo por essa mesma rota. A acção "*walk*", como já referido, é um caso especial de acção. Ao contrário das definidas pelo utilizador, esta necessita de argumentos, nomeadamente o destino (referido pelas coordenadas no mundo) e possivelmente a velocidade (lento, normal ou rápido). Por exemplo: Clicar com o botão esq do rato no destino espacial a tentar alcançar, é o encapsulamento por exemplo para o comando: "walk" (450.0, 0.0, 950.0, 5, normal) //x,y,z, nodo, velocidade;

2. Conjunto não determinístico de instruções:

A linguagem permite também que o agente criado escolha aleatoriamente uma de um conjunto de instruções. Isto possibilita a criação de comportamentos não lineares, não determinísticos. É definido um conjunto de instruções e a cada uma é associada um peso, uma probabilidade. Com base nestas probabilidade uma instrução é escolhida e executada normalmente. Por exemplo:

```
escolhe {
        "coçar":0.8,
        "walk" ( 450.0, 0.0, 950.0, 5, normal): 0.2
};
```

Assim há uma probabilidade de 8/10 de que o Agente irá coçar a cabeça e 2/10 em como irá andar até um ponto especificado. A escolha é aleatória, porém o criador tem a possibilidade de indicar uma determinada preferência. Para tornar as personagens mais credíveis é necessário que estas escolhas reflictam o estado do Agente. Estas decisões constituem outro grupo de definição no comportamento.

3. Definição de propriedades:

As várias propriedades inerentes à condição que moldará o comportamento de cada agente são inicialmente definidas, mas poderão ser alteradas ao longo da "vida" do agente. Por exemplo:

```
define cansaço 0.4;
```

Um exemplo da alteração dos valores da propriedade "cansaço" é quando uma dada personagem se desloca sobre uma zona em declive, e neste caso quando estiver em subida. Estas zonas com declive estão bem catalogadas no mapa hierárquico, organizado por nodos e respectivas ligações. Como saber então, quando um agente se encontrar nestas zonas de declive, se devemos afectar esta propriedade ou não (se em subida aumentará o cansaço, já em descida este estabiliza ou até diminuirá). Nos desenvolvimentos apresentados é estabelecida a entrada (*waypoin*t) nesta zona de acção e assim estabelecida a direcção que o agente tomará e respectiva inclinação ascendente ou descendente a afectar. Esta especificação é bastante mais eficiente em relação à opção de uma constante consulta ao mapa de alturas em tempo real para cada nova célula de posicionamento futuro. A utilização das propriedades e a sua alteração nos scripts permite também uma utilização curiosa. As propriedades são utilizadas como memória, como variáveis. Assim é possível definir, por exemplo, uma propriedade estado e o conjunto de scripts, condições e decisões redefinirão este estado, criando-se muito facilmente um mecanismo associado a uma máquina de estados finita (FSM).

6.4.2.2 | Decisões

Aqui se definem as regras pelas quais o agente se rege. Estas regras constituem o seu comportamento. Estas regras (chamadas aqui regras de decisão) derivam do par já visto anteriormente:

$$Condição \Rightarrow Acção$$

Esta regra determina a execução da Acção quando a Condição associada for satisfeita. Estas regras de decisão podem ser comparadas às instruções *if..then..else* tão comuns em linguagens de programação. Por exemplo, a decisão:

```
define DECISAO("estados") {
       se em nodo com declive, então: o cansaço aumenta, então: "reduzir a
       velocidade de locomoção";
}
```

levará o agente a deslocar-se mais devagar "para poupar energia", estando numa zona em declive. Em termos dos vários tipos de instruções possíveis nesta linguagem, uma regra de decisão define-se então como:

se *Condição* **então** *Instrução* **senão** ...

A instrução refere-se a uma instrução tal como definida em Scripts (como acções, definição de propriedades, etc.). A Condição permite a verificação da validade (com vários pesos) de determinadas propriedades e situações. Pode ser uma condição simples ou um conjunto de instruções condicionais.

6.4.2.3 | Condições

Em última instância, uma condição é uma comparação de valores. Do ponto de vista do utilizador uma condição é uma comparação entre propriedades (internas ou externas) ou Valores. Foram definidos dois tipos de condições:

Condição simples

Uma condição simples enuncia uma comparação entre valores e/ou propriedades de objectos (por objecto entenda-se um agente ou um objecto definido no mundo). Assim:

$$Condicao(V_1, op, V_2) \rightarrow \{true, false\}$$

onde:

$Vn \equiv$ Valor(es) da propriedade de um objecto;
$op \equiv$ Operador condicional {=,!=,<,>,<=,>=}.

Equação 5 – Definição de uma condição comportamental simples

Condição ponderada

Uma condição ponderada surge da necessidade de um sistema onde as decisões sejam tomadas baseadas numa dada situação, derivada do estado interno e externo do ambiente. Define-se como um conjunto de factores pesados, ou seja, um conjunto de propriedades de objectos e pesos que determinam a influência dessa propriedade na condição global. Quando uma condição deste tipo é invocada, o sistema utiliza as propriedades dos objectos, especificadas de maneira a criar um peso entre 0.0 e 1.0 para cada uma. Podem utilizar-se as propriedades directamente, ou incorporando uma dada "tolerância". Deste modo utilizou-se também um sistema de comparações "fuzzy" (e não boolena). Assim:

$$CondicaoPonderada(Factor_1 Peso_1, ..., Factor_n Peso_n) \rightarrow [0,1]$$

Equação 6 – Definição de uma condição comportamental ponderada

Uma soma pesada é calculada com base nos valores e pesos associados aos vários factores. A seguinte função é utilizada para o cálculo da proximidade entre valores, Compara quão perto se encontra o valor entrada da referência, dentro de uma dada tolerância:

$$y = w\left(\frac{entrada - referência}{tolerância}\right)$$

onde:

y ≡ Valor "fuzzy" calculado;
w ≡ Função em forma de "sino" (utilizou-se o coseno).

Equação 7 – Definição de uma condição comportamental baseada em tolerâncias

De modo a tornar esta definição de condições mais intuitiva e clara, a linguagem possibilita o seu agrupamento em conjuntos podem ser referenciados, nas decisões. Assim por exemplo, a decisão de cumprimentar uma personagem pode ser ponderada com base nos sentimentos dessa personagem, nas suas propriedades. O agente "simpatiza com" outro se este último for, sobretudo, simpático, mas também pela proximidade das suas inteligências.

6.4.2.4 | Exemplo de Comportamento

Os scripts criados pelo utilizador são lidos e processados pela biblioteca. Daqui resultam várias estruturas que serão guardadas e associadas a cada agente.

O motor comportamental do agente, ou seja, os mecanismos que lhe dão 'vida', é relativamente simples:

```
enquanto( agente existe )
        recebe_estimulos();
        verifica_decisoes();
        se(em movimento)
                actualiza_posicao(tempo_passado);


verifica_decisoes()
        para todo D pertence a Behavior.Decisoes
                se (verifica D.condicao)
                        executa(D.accao);
```

Este pseudo-código exemplificativo ilustra os procedimentos básicos necessários. A cada iteração (uma iteração pode ser um ciclo de render do OpenGL), o agente é actualizado. Esta actualização passa pela recepção dos estímulos externos e a verificação das decisões. Aqui, para todo o par Condição, Acção definido, a Condição é verificada e se validada é executada a Acção associada.

6.5 | Testes de Simulação Comportamental

Para testar algumas das variedades de comportamento disponíveis e acoplados aos diferentes agentes envolvidos nas simulações, foi utilizado num primeiro exemplo o modelo 3D (Figura 78), que tem vindo a servir de testes durante os desenvolvimentos nesta tese, como ambiente de simulação, e dois tipos de agentes diferentes (Paladin e Cally).

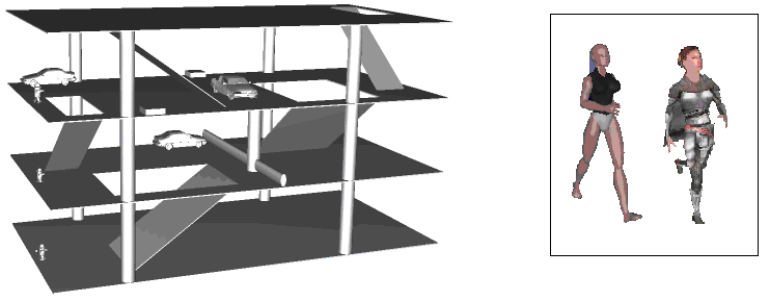


Figura 78 – Ambiente envolvente e personagens a inserir da simulação

A definição do modelo de personagens virtuais a incluir, é realizada num ficheiro de configuração externo, onde se especificam as *meshes*, as animações, os materiais e as texturas do modelo. Este ficheiro é carregado para a criação do modelo utilizando especificações do Cal3D. Para cada agente, é necessária a especificação do conjunto de animações fornecidas pela biblioteca de animação, e a sua associação às acções exequíveis.

Existe uma característica implementada que distingue o tipo de acções/animações: existem acções momentâneas, como o acenar da mão para cumprimentar, e acções persistentes como o andar. Uma personagem neste sistema pode assim efectuar várias acções simultaneamente, em que diferentes actividades podem interagir de diversas formas, por exemplo: "o agente A pode estar a cumprimentar alguém mas entretanto decide coçar a cabeça com a mesma mão devido a uma pancada anterior". Estas acções são concorrentes entre si. No entanto o coçar da cabeça não impede o agente de continuar andar. A linguagem utilizada permite assim a distinção entre tais animações. Para resolver este problema o algoritmo desenvolvido possibilita a definição de grupos de acções disjuntas. Cada acção de um grupo compete com as restantes deste grupo, isto é, quando uma determinada acção está em execução, se uma outra acção do mesmo grupo é executada, a antiga é gradualmente interrompida (o seu peso no algoritmo passa a ser menor) enquanto que a nova é gradualmente

iniciada (o seu peso toma valores mais elevados). Foi necessário definir assim, uma transição adequada entre as animações que requerem este tipo de multiplicidade de escolhas.

Como descrito, cada agente possuiu um conjunto de propriedades pessoais que caracterizam o seu estado interno. Estas propriedades podem ser atribuídas no algoritmo e utilizadas para modelar o seu comportamento (por exemplo fugir de um agente agressivo), podem ser elas: estado (parado ou em andamento), localização (valor x, y, z actual), destino (valor x, y, z destino), velocidade, força, simpatia, grupo a que pertence, etc. Os valores destas propriedades podem variar numa escala de 1/10, de modo a proporcionar uma tendência (dinâmica) maior ou menor em executar determinadas acções de acordo com os parâmetros do seu estado interno. É utilizada, uma implementação de transdutores para representar estados emocionais.

Ainda a nível comportamental é estabelecido o modo como o agente interage com o mundo e os outros agentes. As instruções definidas serão interpretadas criando-se uma estrutura de dados que constitui o modelo mental "cérebro" da personagem de acordo com o modelo da Figura 77. As várias instruções definidas utilizarão as propriedades internas e os estímulos apreendidos (mesmo propriedades de outros agentes e objectos do mundo) e realizarão acções, representadas pelas animações previamente associadas. O algoritmo desenvolvido, possibilita a geração autónoma de comportamentos não determinísticos, através de um conjunto de instruções, onde a cada uma é associada um peso, ou por outras palavras, uma probabilidade de acontecer. Com base nestas probabilidade uma instrução (ou um conjunto delas) é escolhida e executada. Para tornar as personagens mais credíveis é necessário que as suas escolhas reflictam o estado interno do agente, este, composto por uma definição inicial, vai sendo moldado pelas interacções sucessivas com o ambiente e outros agentes na simulação.

Uma outra implementação é a capacidade de comunicação entre os agentes. Com esta funcionalidade o agente comunica determinado elemento e este será apreendido (é um estimulo) por outros (se em distância adequada). Representa a capacidade da fala do ser humano. Podem-se enviar valores direccionados a um agente particular ou a todos os que se encontrem dentro do alcance auditivo estipulado (raio de acção).

Por último, existem as decisões, que definem as regras pelas quais cada agente se rege. Estas regras determinam a execução da Acção quando a Condição associada for

satisfeita (modelo 4 módulos de IA da Figura 77). Estes regulamentos de decisão podem ser comparadas às instruções *if…then…else* tão comuns em linguagens de programação. De um ponto de vista simplista uma condição é uma comparação entre propriedades (internas ou externas) ou valores. Assim podemos ter condições simples, ou ponderadas. Estas últimas são definidas como um conjunto de factores com vários pesos, isto é, um conjunto de propriedades de objectos e afectações que determinam a influência dessa propriedade na condição global. Quando uma condição deste tipo é invocada, o sistema utiliza as propriedades dos objectos, especificadas de maneira a criar uma escala de ponderação específica para cada uma.

Retomando agora ao exemplo da simulação dos dois tipos de agentes inseridos no modelo 3D em teste, podemos observar alguns destes tipos de especificações de autonomia expostos (Figura 79).

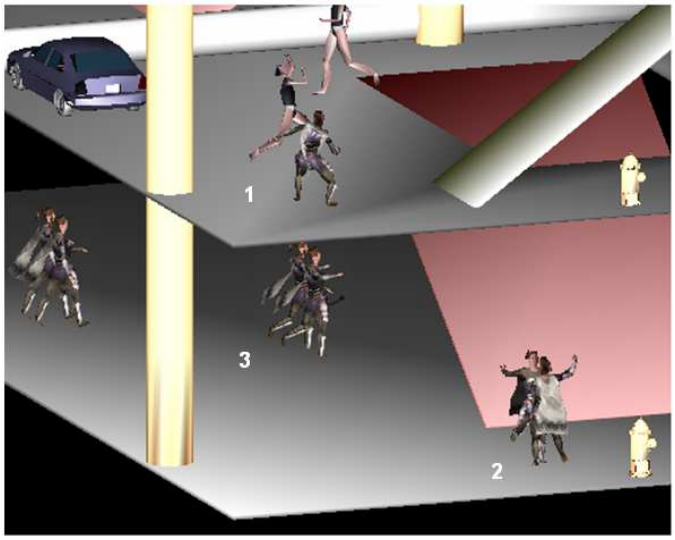


Figura 79 – Simulação exemplificativa de alguns comportamentos desenvolvidos pelos agentes

Como foi referido antes, foram adicionados nesta simulação exemplificativa apenas 2 tipos de agentes. Inicialmente alguns agindo em grupo, outros individualmente num primeiro momento.

Podemos verificar no piso superior (situação 1), um agente Cally e um Paladin em confronto. Estes após o encontro (definido num raio de detecção, podendo ser variável entre os vários tipos de agentes e a sua condição interna) e uma comunicação inicial, produziram determinados comportamentos hostis originando a

luta entre ambos. Não muito longe dali (também dentro de um raio pré-definido) um outro agente Cally acorre à zona de luta, após confirmar (através da percepção ou comunicação inter-agente) que outra personagem do seu grupo está em estado de combate e provavelmente necessitando de ajuda.

No piso inferior (situação 2), e ao início da rampa que permite aceder ao piso superior, dois agentes movimentando-se e agindo de modo individual encontram-se ocasionalmente, trocam cumprimentos enquanto se aproximam (tarefas simultâneas), e possivelmente poderão formar um grupo e agir deste modo a partir daqui (devido ao seu estado actual ser propenso ou não a novos relacionamentos).

Mais ao fundo, e ainda no piso inferior (situação 3), um dos grupos de agentes dirige-se para uma localização específica indicada pelo programador, o grupo mais atrasado persegue-os, pois tem por missão inicial seguir e reagir aos comportamentos do grupo à sua frente (grupo líder).

Por fim referir, que todos os testes e resultados subjacentes a este capítulo, foram contidos e submetidos às conferencias internacionais gerando os artigos [166, 167], de Maio e Junho de 2008, na 3IA2008 – The 11th International Conference on Computer Graphics and Artificial Intelligence, e CISTI2008 – 3ª Conferencia Ibérica de Sistemas y Tecnologías de la Información respectivamente.

CAPÍTULO 7

Conclusões

O fecho de um trabalho de índole científica, para além de pretender essencialmente, a elaboração e reflexão final acerca das suas principais características e contributos, deve também permitir uma visão integrada do esforço desenvolvido, focando eventuais dificuldades que tenham surgido ao longo do trabalho científico realizado na consequência da elaboração da tese.

A inovação que este tipo de estudos deve necessariamente conter, obriga a que cada projecto seja entendido como o início de um processo que em muitos dos casos, não se esgota no trabalho já consumado. Abrem-se assim novas "avenidas" de pesquisa e trabalho. Esta abertura de horizontes renovados propicia o despertar para novos filões de investigação, o que por si só, é um contributo muito importante tendo em vista a evolução do conhecimento científico, e a melhoria das condições de simulação de mundos virtuais tridimensionais com um grande números de personagens que os habitem.

A identificação de um conjunto de propostas, que visem a realização de trabalhos complementares futuros e consequentes do trabalho apresentado, para além de possibilitarem o enriquecimento científico do projecto, podem ainda, focar a aplicabilidade dos contributos em novos contextos práticos, tendo em vista a melhoria da escalabilidade, eficiência e naturalidade das animações apresentadas comportando agentes autónomos em interacção, bem como a promoção, aplicação e afinação da transversalidade de actuação neste tipo de projectos.

7.1 | Enquadramento dos Resultados

A execução deste projecto, que agora se conclui, comportou um vasto conjunto de motivações e consequente produção e realização científica, mas também uma série de dificuldades. A primeira, e porventura entre as mais pertinentes, surgiu imediatamente a seguir ao início do mesmo e relacionava-se com o teor do trabalho a desenvolver.

A busca da razão de ser do projecto, a procura de um foco de necessidade ou instabilidade no domínio alvo da pesquisa, a necessidade de uma justificação científica e inovadora, aliados à intenção de fundar as bases do trabalho científico que tivesse uma aplicabilidade prática na área da animação e simulação por computador, constituíram uma das principais dificuldades do primeiro terço do período previsto para a concretização do projecto.

Uma outra que surgiu numa fase mais avançada do trabalho, mas que por outro lado e ao mesmo tempo se caracterizou como muito motivadora, foi a indispensabilidade de reunir conceitos, fundamentos e produção científica avançada, em quatro áreas distintas mas "contíguas" (computação gráfica, inteligência artificial, análise comportamental e processamento de imagem), para que agregadas, conseguissem "atacar" e fazer a diferença na abordagem ao tema do problema formulado inicialmente.

Uma vez ultrapassadas as hesitações mencionadas, poder-se-á referir, que o desenrolar do trabalho seguinte decorreu em conformidade com o previsto, realçando o posterior enquadramento natural, de todo o desenvolvimento científico propriamente dito, descrito nos capítulos 4, 5 e 6, mesmo com a transversalidade requerida pelas várias áreas de investigação.

As dificuldades no desenvolvimento de simulações neste tipo de cenários habitados, reflectidas no seu planeamento, especificação e construção de sistemas e tendências para modelar o processo cognitivo inerente aos sistemas multi-agentes (SMA) na emulação de comportamentos humanos, muito provavelmente reside no facto deste tipo de representações depender e requerer investigação em várias linhas. Nestas destacam-se principalmente: a especificação para permitir a visualização em tempo real de um elevado número de personagens virtuais associadas a comportamentos de baixo-nível (área científica maioritariamente de CG); e a geração de agentes

inteligentes para permitir o controlo e definição de comportamento dos mesmos (área científica maioritariamente de IA), surgindo naturalmente a sensação de maior credibilidade e autonomia, de simulações em ambientes tridimensionais desconhecidos à partida, com características idênticas aos presentes no mundo real.

Em síntese, a combinação de ideias e tecnologias sobre o campo de pesquisa em ambientes virtuais, é, como foi já verificado, um âmbito muito alargado e concorrido. Este trabalho que agora chega a uma fase de autocrítica, como uma etapa mais do processo de construção de melhor ciência, pendeu com um peso superior (como inicialmente programado) para a área da CG, de modo a estabelecer uma base sólida e testada, capaz de suportar futuros trabalhos científicos em áreas transversais relacionadas com este processo. De facto, vários são os grupos de investigação que se versam sobre os diferentes aspectos do problema. A opção pelos domínios de investigação relacionados com estas áreas de investigação deve-se, fundamentalmente a vários motivos que por si só, justificariam os pré-requisitos de um projecto de Doutoramento.

7.2 | Apresentação de Resultados

Na génese e posterior estabelecimento de mecanismos comportamentais em personagens autónomas virtuais com conduta aceitável e realista, existe uma clara direcção observada nos últimos tempos, onde são traduzidos esforços e trabalho científico em tentar integrar resultados provenientes da interligação de conhecimentos especializados, para que seja possível, com uma maior credibilidade, gerar seres virtuais portadores de algum tipo de raciocínio cognitivo e comunicacional, simulando tarefas comuns em sociedade.

Um dos primeiros passos a percorrer na área da computação gráfica, para que este propósito pudesse ter sucesso, era a criação de condições para poder acomodar naturalmente as futuras personagens sintéticas em ambientes específicos mais realistas que serviriam de base, envolvência, ou mesmo de componente de teste, em determinadas simulações específicas. Neste projecto, são usados ambientes realmente 3D, arbitrários e desconhecidos à partida como suporte à simulação.

7.2.1 | Detecção de Colisões Multi-nível

A detecção de colisões computadas entre superfícies virtuais para análise de objectos em interferência ou intersecção com a respectiva localização dos pontos de colisão na malha poligonal que compõem a geometria dos objectos envolvidos, é muito importante para as mais variadas investigações em diversas áreas da ciência. Neste enquadramento, na tentativa de explorar ambientes 3D mais complexos como espaços de navegação com múltiplos níveis de altura, caracterizando por exemplo áreas interiores de edifícios, que incluam sobreposição de estruturas, espaços em declive, etc, é desejável que as aplicações mantenham os participantes (agentes ou avatares) em movimento numa posição e condição de navegação e altura realista em relação ao terreno e espaço envolvente.

Neste tipo de terrenos ou cenas multi-nível, as técnicas de detecção de colisões comummente utilizadas não se aplicam. Simulações em ambiente desconhecidos à partida com respostas de colisão em tempo real, são abordagens onde terão de ser aplicadas especificações mais eficientes. É precisamente esta a problemática (a detecção de colisões multi-nível em tempo real), que se pretendeu ver solucionado a outro nível, no trabalho apresentado no seguimento do capítulo 4 desta tese.

7.2.1.1 | Motivações e Análise Geral

O caso da detecção de interferências precisas e específicas em tempo real de um número elevado de objectos em movimentação, incluídos em ambientes virtuais complexos, com milhares ou até milhões de polígonos na composição da especificidade da geometria, não é comportável, tanto a nível de recursos computacionais requeridos, como a nível da complexidade envolvida. E nos casos da simulação da movimentação de pedestres virtuais, é também potencialmente desnecessária. A precisão aqui, pode situar-se a um nível muito mais condensado e menos específico ou preciso, pode confinar-se à detecção de colisões entre o volume de cada personagem virtual e o volume dos objectos presentes no ambiente, sejam eles estáticos ou dinâmicos.

Neste âmbito, são frequentemente utilizadas algumas técnicas denominadas de *bounding boxes* e *bounding spheres*, que circundam os objectos em movimento com envoltórios tridimensionais, considerados como um todo e indexados aos objectos

assignados para a detecção de intersecções. Estes métodos, se bem que de uma forma mais abrangente continuam a ter a necessidade de testar intersecções entre polígonos, onde a complexidade da geometria inicial da cena iria novamente ser limitante e decisiva para a eficiência e realismo da apresentação.

Numa análise a outras abordagens, poder-se-ia proceder a uma partição tridimensional do espaço a povoar com base em métodos do tipo BSP com octree´s, mas, como já referenciado, projecta um elevado tempo de construção, e mais importante, um muito significativo e limitante custo de simulação indexado à complexidade do ambiente. Outra objecção a este tipo de suporte à detecção de colisões, é a abrangência do teste muitas vezes estar confinada à básica detecção de colisões, com tratamento igual para todo o tipo de objectos sem capacidade de distinção entre objectos possivelmente ultrapassáveis, ou determinantemente inultrapassáveis para as simulações das condições humanas projectadas nas personagens virtuais (por exemplo um muro de 2 metros deverá ser um obstáculo, mas não um de 20 centímetros). Teria de existir assim uma estrutura paralela representativa da condição das características dos obstáculos a serem ou não testados e percebidos durante a exploração do ambiente virtual.

A ideia que passa pela representação discreta e estática a partir de um mapa de alturas, conseguido por exemplo desde uma vista superior de um qualquer modelo 3D, via rasterização, mantendo em memória e para cada posição espacial previamente estabelecida (desde simples píxeis até composições regulares de grupos destes) os valores da profundidade do terreno, fornecidos pelo z-Buffer, complementa as duas situações necessárias anteriormente identificadas (a detecção e o tipo de colisão) numa única estrutura, que poderá ser facilmente identificável em fase de pré-processamento de um qualquer ambiente proposto à simulação. Este tipo de solução que pretende reduzir o processo computacional da detecção de colisões, através de uma discretização do espaço a navegar, desloca grande parte dos processos computacionais requeridos para a área de actuação dos componentes do hardware gráfico, reduzindo a complexidade dos algoritmos necessários à detecção de colisões, possibilitando uma escalabilidade muito superior em relação ao nível da complexidade do ambiente a povoar, bem como no número de agentes usados na simulação. Os exemplos mais conhecidos utilizando esta plataforma de actuação, são as famosas simulações de sistemas para visualização de personagens virtuais autónomas em ambientes virtuais urbanos, apresentados em vários artigos por Chrysanthou, Tecchia, Loscos e Conroy (detalhados no capítulo 3), dando ênfase à

componente da eficiência, escalabilidade e exibição em tempo real. O problema, é que não deixava ainda de ser, um quadro de aparência de navegação tridimensional, limitando-se a simulações 2,5D.

É no laboratório de realidade virtual (VRlab) do Swiss Federal Institute of Technology, liderado por Daniel Thalmann, um dos pioneiros em investigações utilizando humanos virtuais, onde se ensaiaram as primeiras tentativas para a construção de um sistema ou plataforma, capaz de simular em ambientes desconhecidos, deslocações e respectivas detecções de colisão de algumas personagens virtuais, em navegação livre e verdadeiramente tridimensional. No entanto, nos primeiros estudos efectuados neste laboratório, a ausência importante da condição do conhecimento das características físicas dos obstáculos a serem ou não testados para a detecção de colisões foi relevante, já que, não foi considerada a altura/profundidade da cena ou dos obstáculos (terceira dimensão) de uma forma contínua. Mais recentemente, em 2006 e paralelamente ao espaço temporal das investigações envolvidas nesta tese, um grupo de investigadores do VRLab, desta vez encabeçados por Pettré, compõem artigos científicos de referência, onde reportam explicitamente a necessidade de construção de plataformas apropriadas para mapear de forma automatizada um qualquer modelo tridimensional, de modo a poder simular e planear navegação em tempo real com um nível de independência e realismo superior. É então apresentado o desenvolvimento de uma solução que recorrendo à extracção dos valores em cada ponto da geometria da cena, facultados pelo z-Buffer, interligando aqueles adjacentes numa sectorização do espaço vertical de tamanho igual à altura dos avatares, quando a sua equidistância não superar esta altura de referência. Após esta primeira preparação da cena, segue-se a localização dos espaços livres resultantes às diferentes alturas e a sua delimitação de navegação horizontal, recorrendo a vários cilindros que mapeariam condições de navegação neste espaço, criando desta forma uma estrutura em forma de grafo, onde os nodos representariam os cilindros e as interligações as intersecções entre estes. Este método apresenta-se no entanto ainda, do ponto de vista da eficiência do pré-processamento e preparação do mapeamento e adaptações necessárias a uma correcta navegação largamente influenciado por uma série de fases distintas e morosas, de modo a conseguir primeiro assimilar o por vezes imenso grafo resultante e respectivas conexões e em seguida conseguir identificar, a que níveis de guarda de informação de acessos em altura se situariam os avatares aquando a sua navegação em espaços de deslocação multi-nível. Outro factor ainda de menor-valia, é a impossibilidade de aproveitar e enquadrar a técnica usada nesta plataforma de actuação também para a

detecção de colisões inter-avatares, já que este método não fomenta esta singularidade e dinamismo, tendo de recorrer-se por exemplo, a outros métodos já conhecidos com as respectivas limitações associadas.

Analisadas as actuais e manifestas necessidades de actuação neste tipo de estruturas virtuais, conclui-se que extrair a partir daí, através de métodos inovadores, modelos da representação espacial capazes de simular navegação autónoma e realista, apresentava ainda um desenvolvimento muito deficitário, onde por norma não é produzida especial atenção à navegação livre do agente em qualquer tipo de modelo tridimensional como factor de realismo e pluralidade, e quando existem tentativas neste sentido, estas contemplam ainda demasiadas limitações quando usamos ambientes mais complexos ou um número mais elevado de personagens virtuais em simulação em tempo-real. Estava assim bem definida a zona de actuação, na tentativa de resolver os principais constrangimentos, responsáveis pela ineficiência ou mesmo inexistência da resolução do problema em lidar com este tipo de demanda das novas exigências nas simulações de pedestres.

7.2.1.2 | Especificações do Desenvolvimento e Análise dos Resultados Auferidos

Uma das mais valias resultantes das investigações nesta tese prendeu-se com a evolução de um sistema que preenchesse as lacunas enumeradas nesta área de actuação. A utilização de técnicas de discretização do espaço utilizando projecções e mapas de alturas recorrendo ao z-Buffer com outro nível de profundidade e eficiência, valendo-se de estruturas de dados multi-nível, simples e bem organizadas, apoiadas nas potencialidades do hardware gráfico, e inferidas com dinamismo das estruturas tridimensionais a povoar, provaram conseguir suportar um modelo global de navegação robusto, escalável e muito eficiente em todas as vertentes, comportando simulações com distinção clara de zonas de transição entre as várias alturas de navegação, considerando a altura dos objectos em colisão de forma contínua, facultando informações rápidas das características e tipo de objectos em colisão, para que e em tempo real, fosse possível uma simulação com uma elevada independência de um grande número de personagens em movimento em estruturas ou ambientes virtuais complexos com milhões de polígonos na composição da sua geometria. Este sistema lida ainda perfeitamente com os chamados obstáculos verticais, de forma simples e eficaz. Está também, perfeitamente adaptado às

exigências das detecções de colisão entre as próprias personagens virtuais, utilizando para tal a mesma lógica no mapeamento do espaço, mas recorrendo a um dinamismo de mapeamento e pesquisa necessário em simulação de ambientes dinâmicos, mostrando-se perfeitamente capaz e eficiente na detecção das interferências inter-avatares em simulações com milhares de personagens, sem perda exponencial da eficiência e realismo na apresentação, quando aumenta o número de pedestres a incluir, ou são requeridas cenas mais complexas como ambiente de simulação.

Numa primeira fase, foi apresentado um método para a detecção de colisões multi-nível para mundos 3D complexos e arbitrários. Foram desenvolvidos e reunidos mecanismos de determinação em tempo real do mundo onde o agente actuará, isto é, foram providenciados os meios de percepção autónoma da área envolvente e determinação dos locais cujo deslocamento (bípede) é possível em qualquer mundo 3D (com possibilidade do apoio do utilizador eventualmente para casos muito específicos) recorrendo à discretização do mundo, em oposição à especificação manual do mapa, já que, se por um lado, animações e *scripts* pré definidas oferecem um máximo controlo ao desenhador e programador, não são realmente flexíveis e escaláveis para uma vasta área povoada em ambientes virtuais, em simulações complexas ou desempenhos não lineares.

Os testes mostraram que a detecção de colisões em ambientes tridimensionais, complexos e multi-nível pode comportar um número elevado de personagens virtuais em tempos de *rendering* muito aceitáveis, obtendo animações fluidas e com possibilidade de interacção em tempo real. O número de avatares testados e incluídos em diversos exemplos de simulação utilizando os mais variados (em complexidade, tamanho e diversidade) modelos de personagens 3D, atingiram o milhão de avatares. Como se pude comprovar, este método escala a eficiência do tempo gasto a desenhar e movimentar o número de personagens apresentadas de forma linear, de acordo com o que seria presumível, com base nos objectivos e previsões iniciais. Foi demonstrado que a performance em *run-time* não é significativamente afectada ou influenciada pela complexidade do modelo ou mundo virtual 3D que serve de base à simulação. Mundos mais complexos, requerem logicamente um maior número de fatias geradas e guardadas, mas a performance do sistema não é afectada pelo número de fatias utilizadas (a menos que a memória requerida para o efeito exceda a memória disponível, estes casos são improváveis de se verificar como foi demonstrado no capítulo 4).

A memória *footprint* é perfeitamente aceitável para os exemplos testados, mesmo utilizando modelos superiores a 10 milhões de polígonos (no caso da *Power-Plant*). O número de fatias de informação gerado após a decomposição do modelo é mantido em memória, no menor valor possível e necessário para uma correcta identificação de zonas de colisão ou livres para o movimento, testando sucessivamente quais destas puderam ser descartadas, e rotuladas como dispensáveis e sem acrescento de nova informação, e assim não mantidas em memória (podemos verificar que por exemplo no caso da *Power-Plant*, foram computadas 835 fatias, para só 85 destas serem achadas realmente necessárias e efectivamente mantidas em memória para apoio à movimentação das personagens).

Para modelos de ambientes 3D razoavelmente menores (inferiores a 100.000 polígonos), a percentagem de utilização e armazenamento de matrizes em relação ao total de fatias testadas é na ordem dos 5%, este valor duplica quando utilizamos modelos mais complexos (em torno de 1 milhão de polígonos) e com possibilidade de navegação interior em vários pisos em altura.

O tempo de fatiação (fase de pré-processamento) também está (com é lógico) directamente ligado à complexidade do modelo a povoar, à diversidade de tamanhos a incluir nas personagens virtuais, e à definição da resolução de corte das fatias (valor *resolution* definido inicialmente no algoritmo de fatiação) que queremos impor à precisão das colisões em altura com os objectos no mundo. Este tempo em condições de teste de acordo com as implementações no capitulo 4, segue uma distribuição exponencial decrescente em relação ao número de polígonos associados ao modelo, sendo que para vistas (*viewports*) em resoluções de 515x512 píxeis utilizando modelos com menos de 100 mil polígonos temos uma média de relação de 12 fatias por segundo, entre 100 mil e 1 milhão de polígonos atingiu-se uma média de relação de 4 fatias por segundo, e para mais de 1 milhão de polígonos a média situa-se numa relação de 2 fatias por segundo. Para resoluções de 1024x1024 este valor de fatias geradas *versus* tempo consumido, vê-se reduzido em média três vezes para todas as complexidades de mundos povoados. Como é lógico o tamanho dos agentes definidos para cada um dos modelos a fatiar (integrando proporcionalidade de escala ao modelo) é preponderante e influencia estes tempos de pré-processamento, sendo que a complexidade dos modelos antes testada é uma complexidade de tamanho em altura do ambiente *versus* tamanho dos agentes a incluir, graduados numa escala que por exemplo, e para os casos da cena da garagem, atinge o valor de altura

correspondente a 6,5 agentes, e a *Power-Plant* atinge o valor correspondente a aproximadamente 52 agentes.

O mecanismo proposto, permite ainda agrupar na simulação, agentes de tamanho e alturas díspares. Através da simples imposição ou edição inicial da altura ou largura de determinado agente ou grupo de agentes, o mecanismo de fatiação é adaptado a todas as diferentes alturas das personagens virtuais a incluir na simulação, controlando assim a definição de corte em altura no mundo 3D e armazenando as fatias resultantes (matrizes de dados) com a informação necessária à movimentação para todas e cada uma das alturas e largura das personagens, conseguindo-se assim neste método um vínculo entre a escalabilidade e diversidade para as distintas formas e tamanhos das personagens.

O método proposto de discretização de espaço dadas as circunstâncias do ambiente e qualidade de animação, promove numa fase posterior a localização de espaços multi-nível, deixando transparecer a sua flexibilidade, reutilização e expansão. Este novo desenvolvimento passa pela localização e interligação automática de determinados sectores específicos e característicos de navegação realmente 3D, como são os diferentes planos horizontais de navegação e todas as superfícies de transição entre estes (como escadas, rampas ou planos desnivelados de acesso), capacitando assim a construção de um grafo e respectivas interligações permitindo navegação multi-nível, abstraído da rasterização e do mapeamento inovador antes apresentado. Esta nova especificação é essencial para o sucesso da simulação com objectivos de navegação espaciais específicos, planeando rotas e caminhos de deslocação entre os objectos estáticos e dinâmicos, entre o ponto de partida e um qualquer ponto de destino navegável localizado no ambiente virtual tridimensional.

7.2.2 | Path Planning/Finding Hierárquico

Como anteriormente referido, nem sempre o caminho óptimo será a linha recta até ao destino, navegar contornando *versus* poder ultrapassar passando por cima ou por baixo, poderá ser considerado o melhor e menos custoso percurso. É também importante abordar a questão da escolha da negociação entre a simplicidade ou optimização mais detalhada e mais abrangente (a mais simples poderá ser a mais rápida, mas não proporcionará uma navegação com decisões de alta qualidade). Neste processo, a geração e a procura do espaço navegável terão grande vantagem em ser

automatizadas e geradas desde a geometria, de modo a poder proporcionar automaticamente condições abrangentes à fase do planeamento de trajectos.

Um planeamento e configuração manual da navegação acrescentariam muito tempo e esforço de optimização. Daí as vantagens das técnicas de subdivisão espacial em grelhas de células proporcionando a fácil geração da estrutura através do uso do hardware aquando do *rendering*, a possibilidade de usar esta partição de modo a incrementar a pesquisa em cenas de maior dimensão, e ainda a possibilidade de subdividir os processos de *path-finding* em vários níveis de importância e custo [6]. A utilização deste método favorece a representação do espaço a ser pesquisado através de grafos, a qual poderá ser muito útil, por exemplo, na colocação de pontos guia nos limites de alguns obstáculos com indicação de continuidade do percurso, ou de pontos intermédios entre os nodos para uma navegação mais directa e flexível entre estes.

7.2.2.1 | Motivações e Análise Geral

A habilidade de adquirir uma representação espacial do ambiente como posteriormente a localização de determinados pontos específicos neste, é essencial para o sucesso de qualquer navegação, principalmente em ambientes desconhecidos à partida. Foram já apresentados no capítulo 3 alguns algoritmos que lidam com esta tarefa, mas nenhum deles resolve o problema satisfatoriamente sobre todas as circunstâncias. Alguns algoritmos trabalham somente em duas dimensões e assim inadequados na aplicação a qualquer superfície. Outros não executam tarefas em tempo real, ou não operam totalmente autónomos, necessitando de alguma interacção por parte do programador na fase de pré processamento. Há ainda aqueles que apresentam níveis de escalabilidade e eficiência pouco "recomendáveis" para uma correcta simulação em ambientes desconhecidos mais complexos.

A utilização de grafos com muitos nodos e respectivas interligações tendem a ocupar demasiada memória com decréscimo da performance e com constrangimentos claros em simulações autónomas em tempo real (mais nodos e interligações causam uma menor eficiência na pesquisa, como acontece em [3]). Grafos mais simples libertando mais memória são recomendáveis, com a preocupação de não correr o risco deste tipo de representações ser menos cuidada, possibilitando enquadrar todo o espaço

navegável do ambiente como potencial solução às diversas pesquisas e planeamento de percursos.

Aquando da pesquisa nestes espaços sectoriais do ambiente de navegação, o algoritmo A*, universalmente conhecido pela sua utilização em *path-finding*, contempla alguns problemas em encontrar percursos quando trata ambientes e cenas em 3D real [168, 169]. A função heurística trabalha muito bem somente em situações de um mundo ou ambiente plano. A explicação é simples, a função normal de heurística que utiliza este algoritmo não utiliza as direcções "para cima" ou "para baixo"de uma forma especial [170]. Quer dizer que assume que o agente ou avatar poderá caminhar directamente por exemplo para cima sem ter em linha de conta essa aptidão, esforço, performance, etc. Por norma, alguns profissionais do ramo necessitam de recorrer a algumas soluções de modo a poder incrementar, também e principalmente, a velocidade de pesquisa deste algoritmo [145].

Alguns investigadores [168,169] de modo a melhorar o algoritmo A* e correspondente heurística, fazem o uso de meta-informação já existente e disponível no mapa, tal como portas, quartos, chão, e outros departamentos. Este tipo de catalogação contempla porém a necessidade do pré conhecimento e o assinalar destas áreas na construção da meta-informação a associar a cada ambiente virtual. Também o método de reduzir a quantidade de células visitadas durante a procura em cenas mais complexas é muito interessante, usado em edifícios com espaços multi-nível, através da partição da extensão da estrutura de células de forma mais eficiente. Um domínio mais específico e menos pesquisados são as técnicas de redução de grafos baseadas em combinações recursivas de nodos em *clusters* de modo a ajustar estruturas hierárquicas mais eficientes [6].

Em todas estas especificações, a inclusão de um número elevado de personagens virtuais em movimento, deveria ser escalável e não limitativo da qualidade ou complexidade do ambiente, bem como no número de personagens a incorporar com propósitos de deslocação espacial em grupo ou singularmente. Assim, o objectivo de trabalho abordado no capítulo 5 desta tese com claras vantagens à aplicação de algoritmos de *path-finding*, visou isolar, dividir e catalogar dinamicamente em grafos e de uma forma automática (como outro tipo de meta-informação) a representação multi-nível e em sobreposição do ambiente 3D interior e exterior dos edifícios, de modo a melhorar a eficiência e naturalidade da navegação (como em situações de mudanças de nível em altura, isolando escadas ou rampas de acesso a outros espaços)

proporcionando um número relativamente grande de agentes em navegação. Foi criada uma estrutura de informação hierárquica na ajuda à tomada de decisões (de alto nível para o *path-planning,* e de baixo nível para o *path-finding*). O algoritmo opera suportado em simples imagens (*image based*) e com sub-divisões (*tessellation)* independente, não ficando "escravo" da complexidade da estrutura poligonal do ambiente 3D.

7.2.2.2 | Especificações do Desenvolvimento e Análise dos Resultados Auferidos

Neste campo, estiveram em análise técnicas de planeamento de pesquisa de percursos (*path planning*) endereçando e tentando solucionar problemas de localização de percursos em mundos virtuais complexos, arbitrários, e principalmente com navegação multi-nível. Este tipo de situação, poderá ser ocorrência habitual nas já muitas construções virtuais dos dias de hoje, tais como estádios, edifícios industriais, edifícios habitacionais com vários pisos de navegação interior, onde este método se adaptaria perfeitamente não sendo conceptualmente limitativo a nenhum tipo particular de especificação do ambiente. Estas técnicas incorporaram a recolha e preparação da informação, a catalogação da informação utilizando algoritmos inovadores através do processamento de imagem, a definição e localização de áreas de interligação entre zonas navegáveis a diferentes alturas, e a construção de um grafo hierárquico associado. Este planeamento e preparação da informação do espaço navegável fez uso de técnicas ou algoritmos para a geração de um mapa 3D, e posterior aplicação de algoritmos de *path finding* sobre este.

Em termos de resultados, a decomposição e catalogação do espaço tridimensional numa estrutura hierárquica de interligação de nodos, possibilitou resultados imediatos a acesso não alcançáveis, onde a pesquisa através do algoritmo A* gastaria tempo e recursos para finalizar igualmente numa impossibilidade de acesso. A utilização destes nodos e a respectiva distribuição de pontos intermédios no percurso (*waypoints*), permitiram também uma divisão da trajectória global em várias subdivisões (organizadas por acessos a diferentes alturas), o que possibilitou ganhos de eficiência consideráveis em relação a outros algoritmos já conhecidos, de modo a produzir procuras de percursos faseados com um comportamento mais performante.

A utilização de algoritmos de processamento de imagem para análise da independência de zonas navegáveis à mesma altura em localizações espaciais diferentes, possibilitou obter independência de esforço relativa à complexidade do mundo 3D a simular. Diminuiu também a necessidade de um denso armazenamento para a guarda de mapas de análise e navegação, e ao mesmo tempo permitiu facultar consultas mais eficientes através de uma alocação dinâmica do espaço de pesquisa, necessitando assim de efectuar consultas em tempo real a menores quantidades de informação correspondente à memória necessária a uma correcta navegação, resultando em animações mais fluidas. Esta Implementação é principalmente visível quando se utiliza um número elevado de agentes, e com maior complexidade (em termos de definição visual e propriedades comportamentais de mais alto-nível).

Este tipo de desenvolvimento permite que as informações extraídas do terreno deixem de suportar unicamente duas opções: colisão ou não-colisão, para incorporar custos de movimento, dependendo do virtual esforço que o agente teria de suportar aquando da necessidade de subir ou descer rampas ou escadas de ligação em altura para atingir determinados objectivos, afectando também o estado interno da própria personagem virtual (por exemplo, aquando de uma subida, o estado interno desta na vertente esforço será incrementado, podendo alteraram-se subsequentemente outros comportamentos associados em consequência do aumento ou diminuição de algumas faculdades).

O método descrito é uma nova aproximação em direcção ao pré-processamento de uma qualquer sopa de polígonos desconhecida à partida, de modo a poder ser tratada, deduzida e obtida informação importante das suas potencialidades de assessibilidade, através da construção de uma estrutura hierárquica de navegação, que combinada com o algoritmo A* (original ou alterado), oferece uma solução completa e eficiente. A única etapa que trata directamente com a sopa poligonal da geometria é a etapa inicial, onde os mapas de alturas representativos do mundo a povoar são obtidos. Todas as outras etapas são baseadas em imagens de navegação (*image based*) deduzidas a partir da primeira fase. Sendo essencialmente um algoritmo baseado em imagens, a performance não sofre significativamente quando a contagem poligonal aumenta. Além disso, o método aqui proposto não declina qualquer topologia em particular, não provocando afectação directa da sopa de polígonos aos mapas de navegação onde o sistema de *path planning* é desempenhado.

O processo é totalmente automatizado sem qualquer intervenção do utilizador, sendo capaz de lidar com movimentação multi-nível de uma forma natural. Um dos processos sempre presente em relação à sua importância de actuação em todas as etapas de intervenção deste trabalho, no global, foi promover a reutilização e enquadramento de todas as etapas evolutivas da investigação científica. Neste ponto em particular, a especificação do algoritmo de *path planning* é portador de um legado base, capaz de extracção de recursos de navegação, acessibilidades, e construção da estrutura hierárquica da navegação de uma forma ordenada e pensada para o enquadramento nos requisitos das etapas seguintes.

Este método proporciona um número de nodos sustentável. Reconhecendo que novas divisões em espaços mais amplos e planares poderão ser desejáveis, podem ser aplicadas nestes casos técnicas de subdivisão de espaço sobejamente conhecidas para estes tipos ambientes, já detalhadas no capitulo 3 do estado da arte.

Uma vez que o resultante desta etapa de trabalho é, por construção, uma estrutura de navegação hierárquica, quando combinada com algoritmos de *path finding* tradicionais obtém-se um resultado com números promissores, mesmo quando consideradas colisões inter-avatar. Do resultado dos testes aos exemplos propostos em análise, podem reter-se algumas considerações gerais, tais como: o resultado muito satisfatório da aplicação de técnicas que lidam realmente com a navegação 3D em altura para os casos de navegação em ambientes desconhecidos à partida; o pré processamento utilizado (*path planning*) é importante e decisivo para a performance e correcta inclusão de algoritmos de pesquisa de rotas em multi-nível; a possibilidade de guardar imagens, representativas do espaço navegável, em ficheiros externos, com possibilidade de serem reutilizadas em animações futuras com benefícios de eficiência na fase de pré processamento utilizando o mesmo ambiente; e no geral os algoritmos testados lidam bem com a maioria das situações normais e possíveis de encontrar na vida real.

7.2.3 | Simulação Comportamental

As animações convencionais têm por objectivo recriar a vida através das habilidades artísticas de um animador que transforma as suas observações, experiências, e intuições em personagens virtuais mais próximas ao ser humano. A maioria das ferramentas projectadas para este fim tem por objectivo um controlo manual das

imagens, formas e movimentos. Contudo, um objectivo de teste mais ambicioso para a simulação trata de tentar alcançar entidades complexas possuidoras de comportamentos dinâmicos, crenças e intenções. Além disso, como habitantes de um mundo dinâmico e imprevisível, precisam de incorporar autonomia para se puderem adaptar à mudança. Isto significa que necessitam de ser capazes de perceberem o ambiente envolvente e decidir o que é necessário fazer para alcançarem alguns objectivos de acordo com o seu comportamento actuando em individualidade ou grupo, com as acções relevantes tomadas a serem transformadas em acções de controlo motor. O projecto de um sistema de animação comportamental abrangia assim as questões do desenvolvimento de entidades autónomas dotadas de algumas capacidades e características importantes, como: percepção do ambiente, decisão, acção, memória, capacidade pró-ativa, adaptação, comunicação, entre outros. É neste campo de actuação, que muitos investigadores têm trabalhado para simular de forma realista estas capacidades, com o objectivo de dotar agentes com comportamentos próximos aos humanos.

É reconhecido pelo autor, que a investigação neste campo se posicionou a um nível mais preambular deste ramo de ciência específica que estuda e tenta simular o comportamento humano em todas as suas facetas. O trabalho efectuado neste apartado, teve como missão fundamental, o implementar de certos mecanismos de virtualização da vida (em existência individual e de grupo) aplicada a pedestres humanos, que pudessem actuar, dar suporte e seguimento, bem como servir de teste mais real à base hierárquica construída para povoar ambientes virtuais realmente 3D desconhecidos à partida, incorporando às entidades virtuais alguns tipos de autonomia e poder de decisão.

Neste contexto, o trabalho de progresso e melhoria nas animações e relações comportamentais entre as personagens sintéticas, em situações particulares, centrou-se essencialmente no aumento da sua credibilidade, no chamado *suspension of disbelief* [171]. Como se pude observar, foram várias as implementações e desenvolvimentos inseridos na biblioteca de simulação comportamental conseguida. Foi prestada atenção à criação de uma plataforma de comportamento global, realçando a linguagem e métodos ou mecanismos de regulamentação autónoma das actividades a praticar pelas entidades virtuais. Nestes desenvolvimentos, foi previsto e houve uma especial atenção à possibilidade da sua expansão, considerando a implementação de controlos sob o formato *"core"*, facilmente expansíveis e especializados à posteriori.

Com base na arquitectura de suporte a comportamentos de mais alto-nível, providenciada pelos desenvolvimentos descritos e testados no capítulo 6, foi criada uma linguagem de definição de conduta para as personagens, possibilitando uma forma simples e intuitiva de aplicação de comportamentos inerentes a estas, mas com elevado potencial e liberdade, resultando numa plataforma de personagens virtuais (agora denominadas de agentes) mais "humanas" e credíveis a nível comportamental. No entanto os sistemas multi-agentes (SMA) possuem vulgarmente uma complexidade estrutural de funcionamento considerável, sendo por norma difícil determinar à partida, o conjunto de comportamentos e actividades concretas que irão ser executadas pelo sistema.

Ao permitir que os vários componentes que integram a definição de um agente (e as suas propriedades) sejam acedidos pela linguagem, e aliando o conjunto de instruções de alto nível (condições e decisões) implementadas, desenvolveu-se uma estrutura com potencial para utilização em diversas áreas. Através da implementação de transdutores, a linguagem permitiu a criação de vários *scripts*, contendo uma série de acções a realizar como que argumentos de um filme virtual. Ampliando a sua utilização desde o conhecimento do mundo e do seu estado interno e externo, foi também conseguido conceber *scripts* "dinâmicos" onde o argumento não é determinístico, e depende unicamente da situação, do tempo, do estado interno e do envolvimento das personagens no respectivo ambiente num determinado momento.

O sistema de processamento cognitivo foi organizado em quatro módulos muito específicos: percepção, modelo mental, decisão de objectivos e a resolução de acções a executar. Cada um destes, tenta anexar virtudes e fraquezas da percepção humana e o seu processamento cognitivo associado, incluindo reacções instintivas, erros de percepção, degradação da memória, decisões dependentes do contexto, inferência, entre outros. A capacidade de comunicação entre os agentes é outra das funcionalidades estudadas, estes incorporam a possibilidade de comunicar determinados factos aos seus vizinhos, especialmente se pertencerem ao mesmos grupo social, alterando comportamentos e respectivos estados internos, unicamente pela influência da comunicação. Representa a capacidade da fala e capacidade de influenciar por este meio, subjacentes a qualquer ser humano. Podem-se convencionar alcances auditivos, "enviando" valores para a globalidade de personagens aí posicionados, ou direccioná-los a um agente ou grupo destes em particular.

A este nível simulacional, foram planeados, experimentados e auferidos resultados satisfatórios de observação relevante assente no enquadramento das investigações anteriores. De realçar também a possibilidade de definição de grupos de acções conjuntas complementares, e disjuntas concorrenciais, estabelecendo limites e transições naturais entre estas. Em termos de adequação e escalabilidade, pode considerar-se que a biblioteca desenvolvida, poderá ser logicamente evoluída, e utilizada em vários campos de actuação, como por exemplo: jogos de computador, simulação de multidões, simulação de ambientes urbanos, estudo de espaços em edifícios, interfaces de comunicação, criação de ambientes históricos, arqueológicos, sociais, etc.

7.3 | Trabalho Futuro

Na realização de um qualquer trabalho científico é normal surgirem uma série de questões que, por limitações do próprio projecto, normalmente de ordem temporal, ou por se desenquadrarem dos objectivos centrais da pesquisa, ficam por desenvolver ou esclarecer, abrindo-se novos horizontes de investigação para trabalho futuro.

Deste modo, a proposta de futuros desenvolvimentos, relacionados com a pesquisa efectuada, não deverão ser entendidos como deficiências do projecto, mas sim como contributos que visam o enriquecimento do conhecimento e de melhor ciência nos domínios da investigação desenvolvida. Assim, é possível estabelecer alguns tópicos de trabalho futuro que poderão ser interessantes de explorar, baseado nos problemas debatidos e nas conclusões retiradas, que passo a enunciar.

Em relação à fatiação dos mundo a utilizar, e em casos de excepção, quando a definição de resolução ou precisão da colisão em altura for muito "apreciada", com necessidade de utilização de valores entre a fatiação muito diminutos, o que provocaria maiores necessidades de memória, então uma possível direcção a seguir seria a exploração de algoritmos que utilizem matrizes esparsas, e conseguir gerir a negociação entre o consumo de memória *versus* performance de uma forma mais aprofundada;

Em recentes implementações ao nível do hardware, com utilização de "*floating point buffers*", a cena poderia ser renderizada com recurso à utilização de *shaders*

apropriados de modo a realizar as duas fases de *rendering* e construção do mapa de alturas numa só;

Uma outra possibilidade de evolução, reside na avaliação da fiabilidade e performance do uso de um algoritmo *out-of-core* que permitisse armazenar e devolver os valores e posições das matrizes ao programa base de controlo de colisões com base nestes valores;

Um outro importante ramo de investigação de *path planning/finding* que poderá ser evoluído, é aquando a utilização de ambientes dinâmicos, com testes de colisões entre agentes em tempo real e o espaço envolvente contendo outros tipos de objectos também em movimento;

De modo a aportar maior realismo às simulações, poderão ser consideradas técnicas de vanguarda associadas ao hardware e software, actuando por exemplo, na definição de sombras e tipos de texturas e a interacção destas com os seres virtuais;

A criação de uma memória virtual, permitindo a recordação de eventos e percursos, poderia ser uma mais valia em termos de melhorar a eficiência e incorporar maior conhecimento e aprendizagem aos agentes. Possibilitando assim a aprendizagem pelas experiências "vividas" ao longo do tempo, incorporando diferentes níveis de memoria associada à aprendizagem, recordações e instintos derivados de imagens em memória de curto e longo prazo. Por exemplo, se um determinado caminho se encontra-se obstruído, e provavelmente mantido nesta condição durante algum tempo, as novas regras de direccionamento espacial teriam de incorporar em linha de conta esta condição temporária;

Também a expansão no desenvolvimento do módulo cognitivo de modo a dilatar os sentimentos, necessidades e incertezas dos agentes, seria potencialmente uma área de actuação que ainda muito poderia evoluir, de modo a incorporar maiores capacidades cognitivas aos agentes, logo maior realismo e uma maior valia à simulação;

BIBLIOGRAFIA

[1] C. Reynolds, "Flocks, Herds, and Schools: A Distributed Behavioral Model," presented at SIGGRAPH 87, Computer Graphics 21(4), 1987.

[2] F. Tecchia, Y.Chrysanthou, "Real-time Visualisation of Densely Populated Urban Environments: a Simple and Fast Algorithm for Collision Detection," presented at Eurographics, Swansee, UK, 2000.

[3] J. Pettre, P. Ciechomski, J. Maim, B. Yersin, J.P. Laumond,D. Thalmann, "Real-time navigaton crowds: scalable simulation and rendering," *Computer Animation And Virtual Worlds 2006*, vol. 17- 445-455, 2006.

[4] L. Gonzaga da Silveira, S. Musse, "Real-time generation of populated virtual cities," presented at ACM symposium on Virtual reality software and technology, Limassol, Cyprus, 2006.

[5] F. Tecchia, C. Loscos, R. Conroy, Y. Chrysanthou, "Agent Behaviour Simulator (ABS): A Platform for Urban Behaviour Development," presented at Games Technology 2001(GTEC'2001), Hong Kong, 2001.

[6] N. M. B. Sturtevant, "Partial pathfinding using map abstraction and refinement," presented at The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intel-ligence Conference, 2005.

[7] D. Tan, G. Robertson, M. Czerwinski, "Exploring 3D Navigation: Combining Speed-coupled Flying with Orbiting," presented at CHI 2001 Conference on Human Factors in Computing Systems, 2001.

[8] A. Maximmiano, B. Jonnson, E. VAsconsellos, J. Marcovicth, W. O´Keefe, *Administração do Processo de Inovação Tecnológica*: Editora Atlas, 1980.

[9] S. Kemmis, R. Taggart, "The Action Research Reader," *Deakin University Press, Victoria*, 1988.

[10] I. Sutherland, "Sketchpad: A Man-Machine Graphical Communications System," in *MIT/Lincoln Lab*, vol. Ph.D.: Massachusetts Institute of Technology, 1963.

[11] F. Rogers, R. Earnshaw, *Computer Graphics Techniques:Theory and Practice*, vol. CG3010: Springer – Verlag, 1990.

[12] R. Aylett, M. Luck, "Applying Artificial Intelligence to Virtual Reality: Intelligent Virtual Environments," *Applied Artificial Intelligence*, vol. 14, pp. 3-32, 2000.

[13] R. Aylett, M. Cavazza, "Intelligent Virtual Environments – A state of the art report," presented at Eurographics Conference, Manchester, UK., 2001.

[14] L. Deusdado, "Ensino da Língua Gestual Assistido por Personagens 3D Virtuais," in *Escola de Engenharia - Informática*, vol. Mestrado. Braga: Mestrado, Universidade do Minho, 2002.

[15] L. Nedel, "Animação por computador: Evolução e tendências," *UNIJUÍ*, pp. 87-114, 2000.

[16] D. Thalmann, "New Generation of Synthetic Actors: the Real-Time and Interactive Perceptive Actors," presented at Pacific Graphics, Taiwan, 1996.

[17] M. Thalmann, D. Thalmann, "Complex Models for Animating Synthetic Actors," *IEEE Computer Graphics and Applications*, vol. 11 nº5, 1991.

[18] N. Badler, C.Phillips, B.Webber, "Simulating Humans," presented at Computer graphics, animation, and control, 1999.

[19] M. Wooldridge, N. Jennings, "Agent Theories, Architectures, and Languages: A Survey," presented at ECAI'94 – Workshop on Agent Theories, Architectures and Languages, Amsterdam, The Netherlands, 1995.

[20] J. Ingham, "What is an Agent?" Centre for Software Maintenance, University of Durhan, Durhan, London Technical Report #6/99, 1997.

[21] P. Maes, "Artificial Life Meets Entertainment: Lifelike Autonomous Agents," *Communications of ACM*, vol. 38, n. 11, pp. 108-114, 1995.

[22] LOTRO, "The Lord of the Rings Online," in http://www.lotro.com, 2007.

[23] R. Parent, "Computer Animation, Algorithms and Techniques," *Morgan Kaufman*, 2001.

[24] N. Pettigrew, "The Stop-Motion Filmography: A Critical Guide to 297 Features Using Puppet Animation," *McFarland & Company Inc*, 1999.

[25] A. Watt, M. Watt, *Advanced Animation and Rendering Techniques, theory and practice*: Addison Wesley, 1992.

[26] A. LaMothe, J. Ratcliff, M. Seminatore, D. Tyler, *Tricks of the Game Programming Gurus*: Sams Publishing, 1994.

[27] R. de Araujo, A. Battaiola, "Jogos 3D Interativos Multiusuários na Internet: Estado Atual, Perspectivas e Desafios," *Relatório Interno do DC/UFSCar*, 1998.

[28] J. Banks, *Principles of Simulation: Handbook of Simulation - Principles, Methodology, Advances, Applications and Practices*: Jerry Banks (editor). Wiley-Interscience, 1998.

[29] A. Law, et all, *Simulation Modelling and Analysis*, vol. 3º: McGraw-Hill, 2000.

[30] J. Rickel, S. Marsella, J. Gratch, R. Hill, D. Traum, W. Swartout, "Toward a New Generation of Virtual Humans for Interactive Experiences," *IEEE Intelligent Systems*, vol. 17, n. 4, 2002.

[31] J. Gratch, J. Rickel, E. Andre, N. Badler, J. Cassell, E. Petajan, "Creating Interactive Virtual Humans: Some Assembly Required," *IEEE Intelligent Systems*, vol. 17, n. 4, 2002.

[32] G. Anastassakis, T. Ritching, T. Panayiotopoulos, "Multi-agent Systems as Intelligent Virtual Environments," *LNAI 2174*, pp. 381-395, 2001.

[33] J. Foley, A.Van Dam, S. Feiner, F. Hughes, *Computer Graphics: Principles and Practice*, vol. 1174p: USA:Addison-Wesley, 1990.

[34] X. Pueyo, D. Tost, "A Survey of Computer Animation," presented at Computer Graphics Forum, Amsterdam, 1988.

[35] T. Möller, E. Haines, *Real-Time Rendering*: A K Peters, Ltd, 1999.

[36] A. Watt, F. Policarpo, *3D Games – Real-Time Rendering and Software Technology*: Addison-Welsey, 2001.

[37] H. Moravec, *O Futuro da Inteligência Humana e Robótica (or. Mind Children,The Future of Robot and Human Intelligence)*: Gradiva, Lisboa, 1988.

[38] S. Russell, *Inteligencia Artificial: Un enfoque moderno*: Prentice - Hall. México, 1996.

[39] M. Juchem, R. Bastos, "Engenharia de sistemas multiagentes: uma investigação sobre o estado da arte," *Technical Report Series, number 014, April 2001, PUCRS.*, 2001.

[40] M. Wooldridge, "Intelligent agents. In: WEISS, G. (Ed.) Multiagent Systems," *A Modern Approach to Distributed Artificial Intelligence, MIT Press*, 1999.

[41] M. Thalmann, D. Thalmann, "Artificial Life and Virtual Reality," 1994.

[42] S. Franklin, A. Graesser, "Is it an Agent, or just a Program? ATaxonomy for Autonomous Agent," presented at Third International WorkShop on Agent Theories, Architectures, and Languages, Springer-Verlag, 1997.

[43] M. Wooldridge, "Intelligent Agents. In: Multiagent Systems," *The MIT Press*, 1999.

[44] R. Firby, "An investigation into reactive planning in complex domains," presented at Sixth National Conference on Artificial Intelligence, 1987.

[45] M. Shoppers, "Universal Plans for Reactive Robots in Unpredictable Environments," presented at InternationalConference on Artificial Intelligence, IJCAI-87, 1987.

[46] Walkinside, in *http://www.walkinside.com/*, 2006.

[47] B. Ulicny, D. Thalmann, "Crowd simulation for virtual heritage," presented at First International Workshop on 3D - Virtual Heritage, Geneve, 2002.

[48] W. Shao, D. Terzopoulos, "Autonomous pedestrians," presented at ACM Siggraph/Eurographics Symposium on Computer Animation (SCA´05), Los Angeles, California, 2005.

[49] S. Musse, "Human Crowd Modelling with Various Levels of Behaviour Control," in *Lausanne: EPFL*, vol. PhD, 2000.

[50] S. Grand, D. Cliff, "Creatures: Entertainment software agents with artificial life," *Autonomous Agents and Multi-Agent Systems*, vol. 1, n. 1, pp. 39-57, 1998.

[51] A. Nijholt, J. Hulstijn, "Multimodal Interactions with Agents in Virtual Worlds," *Future Directions for Intelligent Information Systems and Information Science, Physica-Verlag: Studies in Fuzziness and Soft Computing*, 2000.

[52] S. Donikian, "The Virtual Museum," in *http://www.irisa.fr/prive/donikian/virtual_museum.html.*, 2001.

[53] L. Chittaro, R. Ranon, "New Directions for the Design of Virtual Reality Interfaces to E-Commerce Sites," presented at 5th International Conference on Advanced Visual Interfaces, ACM Press, New York, 2002.

[54] M. Cavazza, F. Charles, S. Mead, A. Strachan, "Virtual Actors' Behaviour for 3D Interactive Storytelling," presented at Eurographics Conference, 2001.

[55] J. Rickel, W. Johnson, "Integrating Pedagogical Capabilities in a Virtual Environment Agent," presented at 1st International Conference on Autonomous Agents, ACM Press, 1997.

[56] P. Jiménez, F. Thomas, et al., "Collision detection: A survey," *Computers & Graphics*, vol. 25(2), pp. 269-285, 2001.

[57] M. Lin, S. Gottschalk, "Collision detection between geometric models: A survey," presented at IMA Conference on Mathematics of Surfaces, 1998.

[58] J. Cohen, M. Lin, et al., "I-Collide: An interactive and exact collision detection system forlarge-scale environments," presented at ACM Interactive 3D Graphics Conference., 1995.

[59] S. Gottschalk, M. Lin, et al., "OBB-tree:A hierarchical structure for rapid interference detection," presented at ACM SIGGRAPH, 1996.

[60] Y. Cai, Z. Fan, H. Wan, S. Gao, B. Lu, K.Lim, "Hardware-accelerated collision detection for 3D virtual reality gaming," *Simulation Gaming*, vol. 37; 476, 2006.

[61] G. Baciu, S-K. Wong, et al., "RECODE:An image-based collision detection algorithm," *Journal of Visualization and Computer Animation*, vol. 10(4), pp. 181-192, 1999.

[62] K. Hoff III, A. Zaferakis,et al., "Fast and simple 2D geometric proximity queries using graphics hardware," presented at ACM Symposium on Interactive 3D Graphics, 2001.

[63] Y. Kim, M. Lin, et al., "Fast penetration depth estimation using rasterization hardware and hierarchical refinement," presented at Workshop on Algorithmic Foundations of Robotics (WAFR), 2002.

[64] J. Lombardo, M.Cani, et al., "Real-time collision detection for virtual surgery," presented at Computer Animation '99, 1999.

[65] P. Hubbard, "Interactive collision detection," presented at IEEE Symposium on Research Frontiers in Virtual Reality, 1993.

[66] B. Naylor, J. Amanatides, W. Thibault, "Merging BSP Tress Yields Polyhedral Set Operations," presented at ACM Computer Graphics, 1990.

[67] H. Samet, *The Design and Analysis of Spatial Data Structures, Series in Computer Science*: Addison-Wesley, 1990.

[68] A. Wilson, E. Larsen, D. Manocha, C. Lin, "Partitioning and handling massive models for interactive collision detection," presented at Eurographics Conference, Computer Graphics Forum, 1999.

[69] N. Govindaraju, S. Redon, M. Lin, D. Manocha, "Cullide: Interactive collision detection between complex models in large environments using graphics hardware," presented at Eurographics/SIGGRAPH Graphics Hardware Workshop, 2003.

[70] D. Knott, Pai D., "Cinder: Collision and interference detection in real-time using graphics hardware," presented at Graphics Interface '03, 2003.

[71] S. Kimmerle, N. Matthieu, "Hierarchy accelerated stochastic collision detection," presented at Vision Modeling, and Visualization, 2004.

[72] M. Winter, Stamminger M, "Depth-buffer based navigation," presented at In Vision, Modeling, and Visualization Conference Proceedings (VMV), 2004.

[73] S. Vosinakis, T. Panayiotopoulos, "A tool for constructing 3d environments with virtual agents," *Multimedia Tools Applications*, vol. 25, 2, pp. 253–279, 2005.

[74] S. Redon, Y. Kim, M. Lin, D. Manocha, J. Templeman, "Interactive and continuous collision detection for avatars in virtual environments," presented at IEEE International Conference on Virtual Reality, 2004.

[75] J. Lengyel, M. Reichert, B. Donald, D. Greenberg, "Real-Time robot motion planning using rasterizing computer graphics hardware," *Computer Graphics*, vol. 24(4), pp. 327-335, 1990.

[76] J. Rossignac, A. Megahed, B. Schneider, "Interactive Inspection of Solids: Cross-sections and interferences," *Computer Graphics*, vol. 26(2):353-360, 1992.

[77] K. Myszkowski, O. Okunev, T. Kunii, "Fast collision detection between complex solids using rasterizing graphics hardware," *The Visual Computer*, vol. 11(9): 497-512, 1995.

[78] G. Still, "Crowd Dynamics," vol. PhD thesis: University of Warwik, UK, 2000.

[79] S. Stylianou, M. Fyrillas, Y. Chrysanthou, "Scalable pedestrian simulation for virtual cities," presented at Proceedings of the ACM symposium on Virtual reality software and technology, Hong Kong, 2004.

[80] B. Moulin, et all, "MAGS Project: Multi-Agent GeoSimulation and Crowd Simulation," presented at Proceediings of the COSIT'03 Conference Ittingen, Switzerland, 2003.

[81] A. Treuille, S. Cooper, Z. Popovic, "Continuum Crowds," presented at SIGGRAPH 2006, Boston, USA, 2006.

[82] A. Steed, "Efficient Navigation Around Complex Virtual Environments," presented at ACM Symposium on Virtual Reality Software and Technology (VRST-97), ACM Press, 1997.

[83] J. Mackinlay, S. Card, G. Robertson, "Rapid controlled movement through a virtual 3D workspace," presented at Computer Graphics, 1990.

[84] S. Bandi, D. Thalmann, "Space discretization for efficient human navigation," presented at Eurographics '98, Computer GraphicsForum, 1998.

[85] S. Bandi, D. Thalmann, "An Adaptive Spatial Subdivision of the Object Space for Fast Collision Detection of Animated Rigid Bodies," presented at Eurographics '95 Conference Proceedings, Computer Graphics Forum, 1995.

[86] J. Pettre, J.P. Laumond,D. Thalmann, "A naviagtion graph for real-time crowd animation on multilayered and uneven terrain," presented at The First International Workshop on Crowd Simulation (V-CROWDS´05), Lausanne, Switzerland, 2005.

[87] M. Cherif, "Motion planning for all-terrains vehicles: a physical modeling approach for coping with dynamic and contact interaction constraints," *IEEE transactions on Robotics and Automation (ICRA)*, 1999.

[88] A. Hait, T. Siméon, M. Taix, "Algorithms for rought terrain trajectory planning," *Advanced Robotics*, vol. 14(6), 1999.

[89] B. Salomon, M. Garber, M. Lin, D. Manocha, "Interactive navigation in complex environments using path planning," presented at Symposium on Interactive 3D graphics, ACM Press, New York, NY, USA, 2003.

[90] R. Darken, J. Sibert, "Navigating in large virtual worlds," *International Journal of Human-Computer Interaction*, vol. 8, 1, pp. 49–72, 1996.

[91] T. Li, H. Ting, "An intelligent user interface with motion planning with 3d navigation," presented at IEEE VR, 2000.

[92] J. Latombe, *Robot Motion Planing*. Boston: Kluwer Academic Publishers, 1991.

[93] L. Kavraki, J. Latombe, "Randomized preprocessing of configuration space for fast path planning," presented at IEEE Conference on Robotics and Automation, 1994.

[94] M. Overmars, P. Svestka, "A probabilistic learning approach to motion planning," presented at Algorithmic Foundations of Robotics, A. K. Peters, Wellesley, MA., 1995.

[95] M. Pinter, "Toward More Realistic Pathfinding," in *http://www.gamasutra.com/features/20010314/pinter_01.htm*, 2001.

[96] R. Bukowski, "Second generation walkthrough system," presented at 7th International Conference on Virtual Systems and MultiMedia, Berkeley, California, 2001.

[97] L. Arns, C. Neira, "Effects of physical and virtual rotations and display device on users of an architectural walkthrough," 2004.

[98] O. Kreylos, "Path Finding In Complex Maps And The Black Art Of Linear Algebra," in *http://graphics.cs.ucdavis.edu/~okreylos/Private/AlgorithmCorner/*, 2005.

[99] F. Lamarche, S. Donikian, "Crowd of virtual humans: a new approach for real time navigation in complex and structured environments," presented at Eurographics 2004, Volume 23. Nr 3., 2004.

[100] J. Kuffner, "Goal-directed navigation for animated characters using real-time peth planning and control," *Lectures Notes in Computer Science*, vol. 1537, pp. 171-179, 1998.

[101] P. Hart, N. Nilsson, B.Raphael, "A formal basis for the heuristic determination of minimum cost paths," presented at IEEE Transactions on Systems Science and Cybernetics SSC4 (2), 1968.

[102] B. Stout, "Smart Moves: Intelligent Path-Finding," in *http://www.gamasutra.com/features/19970801/pathfinding.htm*, 2001.

[103] M. Lanctot, N. Sun, C.Verbrugge, ""Path-finding for Large Scale Multiplayer Computer Games," presented at GameOn-NA, 2006.

[104] M. Berg, M. Kreveld, M. Overmars, O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*. Heidelberg, New York: Springer-Verlag, Berlin, 2000.

[105] O. Arikan, A. Forsyth, "Efficient multi-agent path planning," presented at In Computer Animation and Simulation ‘ O1, Springer-Verlag, 2001.

[106] K. Hoff III, T. Culver, J. Keyser, M. Lin, D. Manocha, "Fast Computation of Generalized Voronoi Diagrams Using Graphics Hardware," presented at SIGGRAPH, 1999.

[107] K. Hoff III, T. Culver, J. Keyser, M. Lin, D. Manocha, "Interactive motion planning using hardware-accelerated computation of generalized voronoi diagrams," presented at International Conference on Robotics and Automation, 2000.

[108] C. Andújar, P. Vàzquez, M. Fairén, "Way-Finder: guided tours through complex walkthrough models," presented at Computer Graphics Forum, 2004.

[109] A. Sud, E. Andersen, S. Curtis, M. Lin, D. Manocha, "Real-time Path Planning for Virtual Agents in Dynamic Environments," presented at IEEE Virtual Reality 2007, Charlotte, USA, 2007.

[110] C. Loscos, D. Marchal, A. Meyer, "Intuitive Crowd Behaviour in Dense Urban Environments using Local Laws," presented at Tpcg, 2003.

[111] F. Dapper, E. Prestes, M. Idiart, A. Nedel, P.Luciana, "Simulating Pedestrian Behavior with Potential Fields," presented at Computer Graphics International 2006 (CGI), Hangzhou, China, 2006.

[112] X. Tu, D. Terzopoulos, "Artificial Fishes: Physics, Locomotion, Perception, Behavior," presented at Computer Graphics: Proceedings of SIGGRAPH'94, ACM Press, 1994.

[113] M. Dorigo, L.M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem," presented at IEEE Transactions on Evolutionary Computation, 1997.

[114] D. Brogan, J. Hodgins, "Group Behaviors for Systems with Significant Dynamics," *Autonomous Robots, vol 4, pp 137-153*, 1997.

[115] D. Koeppel, "Massive attack," http://www.popsci.com/popsci/science/article/0, 390918-1,00.html, Ed., 2005.

[116] V. M. Predtechensky, A.I. Milinsky, *Planning for foot traffic flow in buildings*. New Delhi, 1978: Amerind Publishing Co, 1978.

[117] E. Lardeux, "Modélisation des Mouvements de Foule," in *PhD thesis of the university of Rennes*: France, 1992.

[118] D. Helbing, P. Molnár, "Self-organization phenomena in pedestrian crowds," presented at F. Schweitzer (ed.), Gordon and Breach, London, 1997.

[119] D. Helbing, I.Farkas, T. Vicsek, "Simulating Dynamical Features of Escape Panic," in *Nature*, vol. 407, 2000.

[120] A. Braun, S. Musse, B. Bodmann, "Um modelo Para Comportamentos Individuais em Simulação de Multidões," in *Revista Scientia, São Leopoldo, RS, Brasil*, vol. 14(1), 2003.

[121] S. Musse, et al., "PetroSim - Um Sistema para Simulação de Multidões em Situações de Emergência," in *Revista Petro & Química - Petróleo - Gás - Petroquímica - Química*, vol. № 268, 2005, pp. 60-65.

[122] E. Bouvier, E. Cohen, "Simulation of Human Flow with Particle Systems," 1994.

[123] R. B. Tilove, "Local obstacle avoidance for mobile robots based on the method of artificial potentials," presented at IEEE International Conference on Robotics and Automation, 1990.

[124] K. Kant, S. Zucker, "Planning collision-free trajectories in time-varying environment: a two-level hierarchy," presented at IEEE International Conference on Robotics and Automation, 1988.

[125] P. Willemsen, J. Kearney, H.Wang, "Ribbon Networks for Modeling Navigable Paths of Autonomous Agents in Virtual Urban Environments," 2003.

[126] H. Wang, J. Kearney, J.Cremer, P. Willemsen, "Autonomous Agents Trough Intersections in Virtual Urban Enviroments," 2004.

[127] T. Tsubouchi, S. Arimoto, "Behavior of a mobile robot navigated by an 'iterated forecast and planning' scheme in the presence of multiple moving obstacles," presented at IEEE International Conference on Robotics and Automation, 1994.

[128] F. Feurtey, "Simulating the Collision Avoidance, Behavior of Pedestrians," 2000.

[129] K. Forbus, W. Wright, "Some notes on programming objects inTheSims," 2001.

[130] M. Kallmann, D.Thalmann, "Modeling objects for interaction tasks," presented at Eurographics Workshop on Animation and Simulation1998, 1998.

[131] J. K. Lau Manfred, "Behavior Planning for Character Animation," presented at ACM SIGGRAPH / Eurographics Symposium on Computer Animation, Los Angeles, CA, 2005.

[132] J. C. Teixeira, M. Figueiredo, "Virtual Environments meet Architectural Design Requirements," *Modelling and Graphics in Science and Technology*, pp. 207-219, 1995.

[133] B. Duffy, G. Hare, A. Martin, et al., "Agent Chameleons: Agent Minds and Bodies," presented at Conference on Computer Animation and Social Agents (CASA 2004), Switzerland, 2004.

[134] A. Ramires, L. Deusdado, "Efficient Conservative Collision Detection for Populated Virtual Worlds," presented at Ibero-American Symposium on Computer Graphics - SIACG(06), Santiago de Compostela, Spain., 2006.

[135] B. Bounabat, "A New Formal Approach for the Specification and the Verification of Multi-agent Reactive Systems Operating Modes," *Information Processing and Technology*, vol. Nova Science Publishers, New York, EUA, pp. 25-47, 2001.

[136] M. Wooldridge, *An Introduction to MultiAgent Systems*: John Wiley & Sons, 2002.

[137] K. Perlin, A. Golberg, "Improv: A System for Scripting Interactive Actors in Virtual Worlds," *Computer Graphics*, vol. 29, 1996.

[138] N. Badler, et al., "Parameterized Action Representation for Virtual Human Agents," *Embodied Conversational Characters*, vol. MIT Press, Cambridge, pp. 256-284, 2000.

[139] J. Funge, X. Tu, D. Terzopoulos, "Cognitive Modeling: Knowledge, Reasoning and Planning for Intelligent Characters," presented at SIGGRAPH'99, Los Angeles, CA, 1999.

[140] J. Torres, L.Nedel, R. Bordini, "Using the BDI Architecture to Produce Autonomous Characters in Virtual Worlds," presented at Fourth International Workshop on Intelligent Virtual Agents (IVA 2003), Kloster Irsee, Germany, 2003.

[141] Y. Demazeau, "From Interactions to Collective Behaviour in Agent-based Systems," presented at European Conference on Cognitive Science, Saint-Malo, France, 1995.

[142] Cal3D, in *http://cal3d.sourceforge.net*, 2004.

[143] DevIL, in http://openil.sourceforge.net/, 2006.

[144] OpenCv, in http://www.intel.com/technology/computing/opencv/, 2006.

[145] P. Lester, in http://www.policyalmanac.org/games/aStarTutorial_port.htm, 2004.

[146] A. R. Fernandes, L. Deusdado, "Efficient Conservative Collision Detection for Populated Virtual Worlds," presented at Ibero-American Symposium on Computer Graphics - SIACG(06), Santiago de Compostela, Spain., 2006.

[147] L. Deusdado, A.R. Fernandes, O. Belo, "Path Planning for Complex 3D Multilevel Environments," presented at SCCG08 –Spring Conference on Computer Graphics, Bratislava, Eslováquia, 2008.

[148] M. Schreckenberg, S. Sharma, *Pedestrian and Evacuation Dynamics*: Germany: Springer-Verlag New York,Inc, 2002.

[149] S. Russel, P. Norvig, *Artificial Intelligence: A Modern Approach*: New Jersey: Prentice-Hall, 1995.

[150] G. Tecuci, "Building Intelligent Agents," *Academic Press*, pp. 320p, 1998.

[151] J. Krebs, N. Davies, *An Introduction to Behavioural Ecology*, vol. 3rd edition: Blackwell Scientific Publications Ltd, Oxford, 1981.

[152] C. Balkenius, "Natural Intelligence in Artificial Creatures," vol. Phd thesis. Sweden: Lund University Cognitive ScienceKungshuset, 1995.

[153] S. Musse, D. Thalmann, "Hierarquical model for real time simulation of virtual human crowds," *IEEE Transactions on Visualization and ComputerGraphics*, vol. 7,no.2, pp. 152–164, 2001.

[154] M. Mataric, *Designing and understanding adaptative group behaviors*: Waltham, 1995.

[155] S. Goldenstein, E. Large, D. Metaxas, "Non-linear dynamical system approach to behavior modeling," *The Visual Computer*, pp. 349–364, 1999.

[156] B. Ulincny, D. Thalmann, "Crowd simulation for interactive virtual environments and vr training systems," pp. 163–170, 2001.

[157] C. Sanza, O. Heguy, Y. Duthen, "Evolution and cooperation of Virtual entities with classifier," presented at Proceedings of Eurographics, 2001.

[158] A. Lotka, "Undaped Oscillations derived from the mass action," vol. no.42, pp. 1595–1599, 1920.

[159] V. Volterra, "Variations and fluctuations of a number of individuals in animal species living together," *Mem. Acad. Lincei*, vol. no. 2, pp. 31–113, 1926.

[160] M. Huhns, M. Singh, "Readings in Agents," *Readings in Agents.*, vol. Chapter 1, 1-24, 1998.

[161] G. O'Hare, N. Jennings, *Foundations of Distributed Artificial Intelligence*. Hardcover, 1996.

[162] M. Wooldridge, *Introduction to Multi-Agent Systems*. Paperback, 2001.

[163] G. Weiss, "Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence," *MIT Press*, 1999.

[164] P. Sweetser, "Current AI in games: A review." School of ITEE, University of Queensland, 2003.

[165] S. Rabin, "Implementing a State Machine Language," presented at AI Game Programming Wisdom (ed. S. Rabin), Charles River Media, Hingham, MA, 2001.

[166] L. Deusdado, O. Belo, A.R. Fernandes, "Pedestrians Behavior Simulation in Real 3D Environments," presented at 3IA´2008 – The Eleventh International Conference on Computer Graphics and Artificial Intelligence, Athens, Grécia, 2008.

[167] L. Deusdado, O. Belo, A.R. Fernandes, "Simulação Comportamental de Pedestres Em Ambientes 3D Desconhecidos," presented at CISTI 2008 – Conferencia Ibérica de Sistemas y Tecnologías de la Información, Vigo, Espanha, 2008.

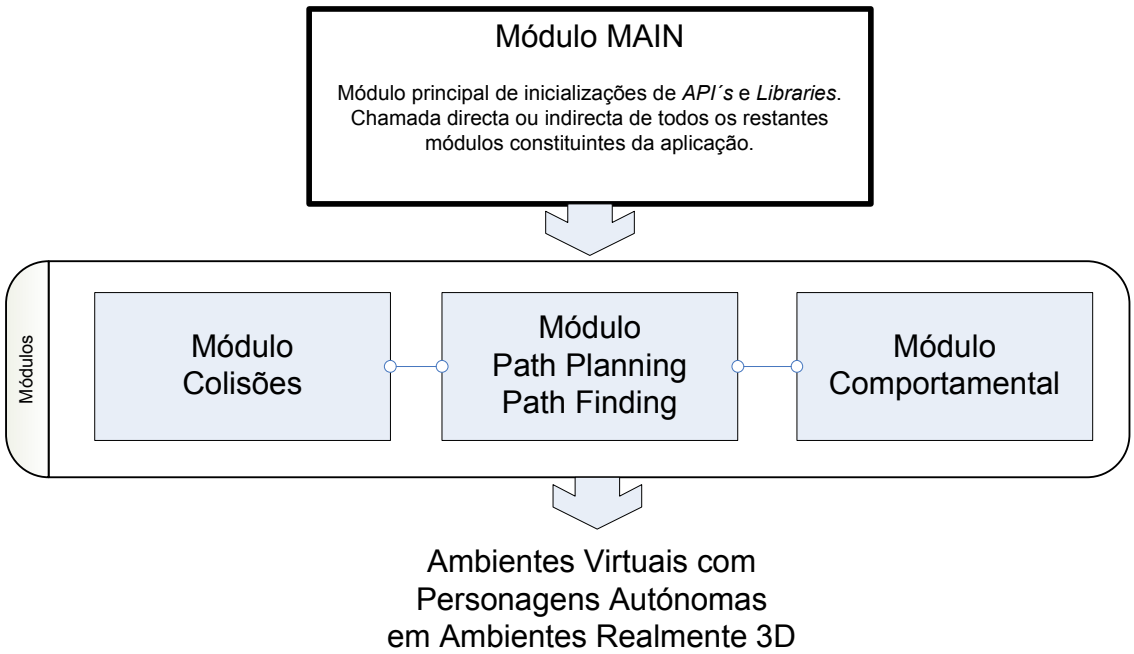
[168] R. Holte, M. Perez, R. Zimmer, A. MacDonald, "Hierarchical A*: Searching abstraction hierarchies efficiently," presented at The Thirteenth National Conference on Artificial Intelligence (AAAI-96), 1996.

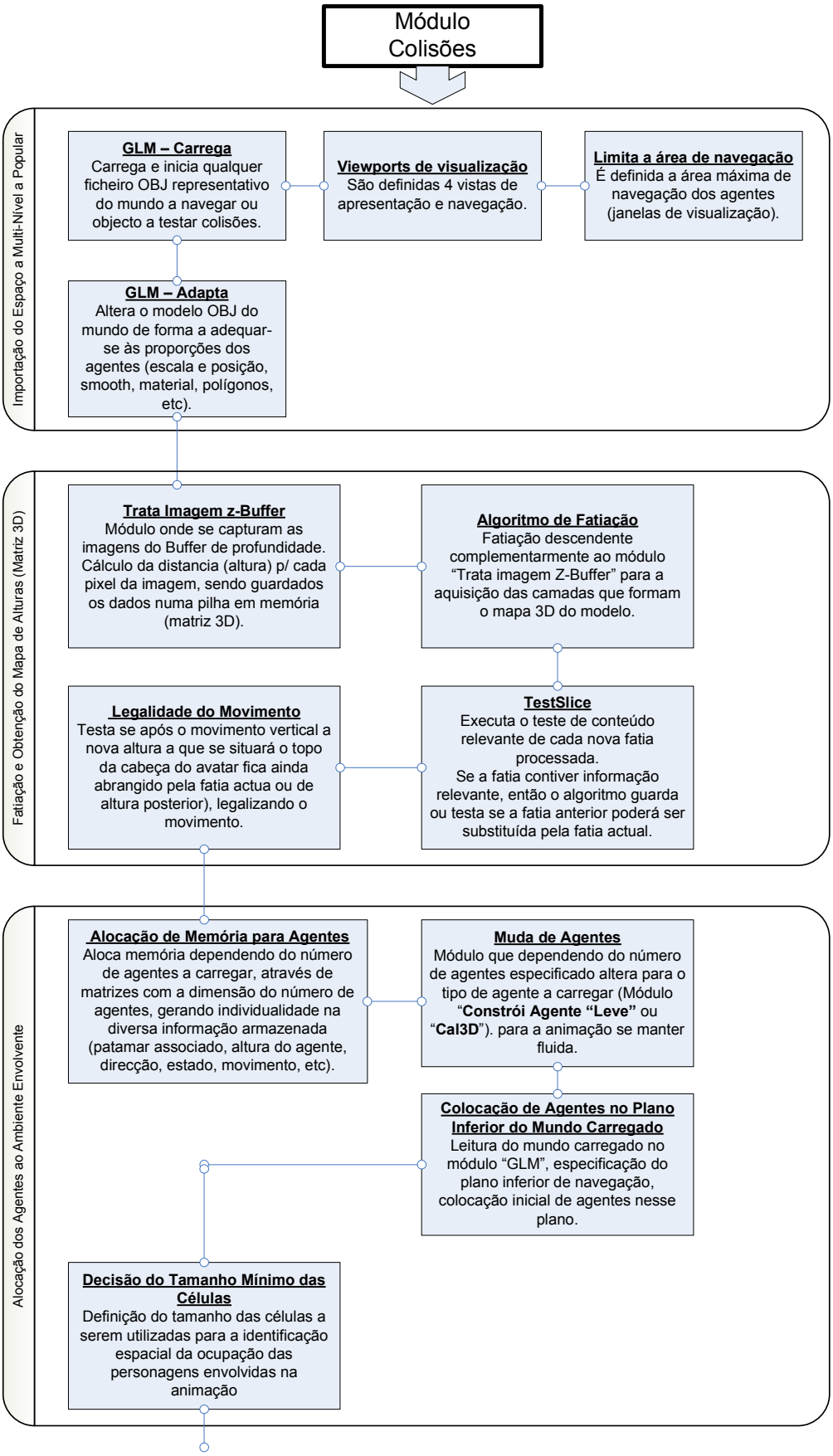
[169] D. Maio, S. Rizzi, "A hybrid approach to path planning in autonomous agents," presented at Second International Conference on Expert Systems for Development, 1994.

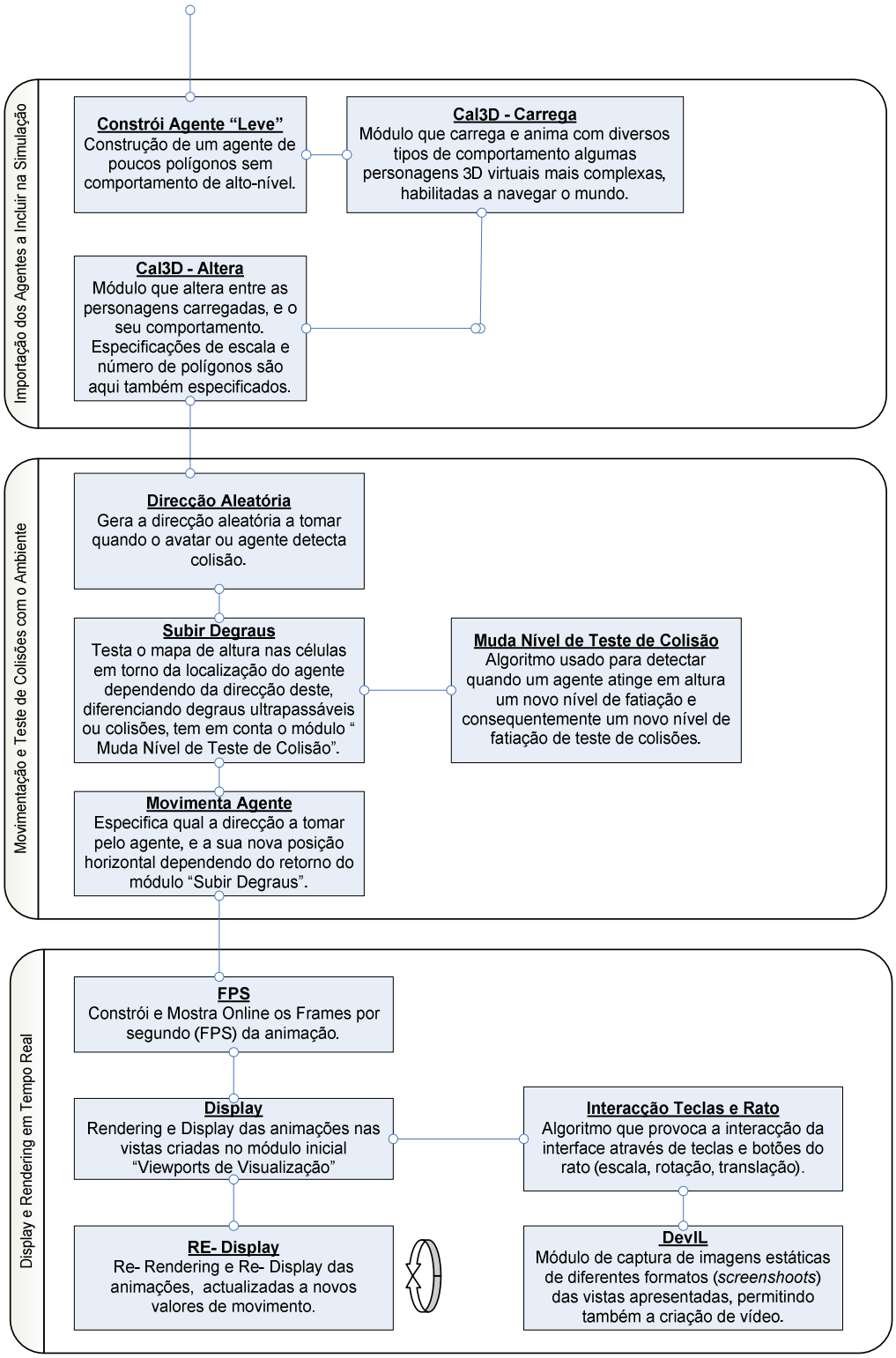
[170] T. Fröhlich, D. Kullmann, "Autonomous and Robust Navigation for Simulated Humanoid Characters in Virtual Environments," presented at First International Symposium on Cyber Worlds (CW'02), 2002.

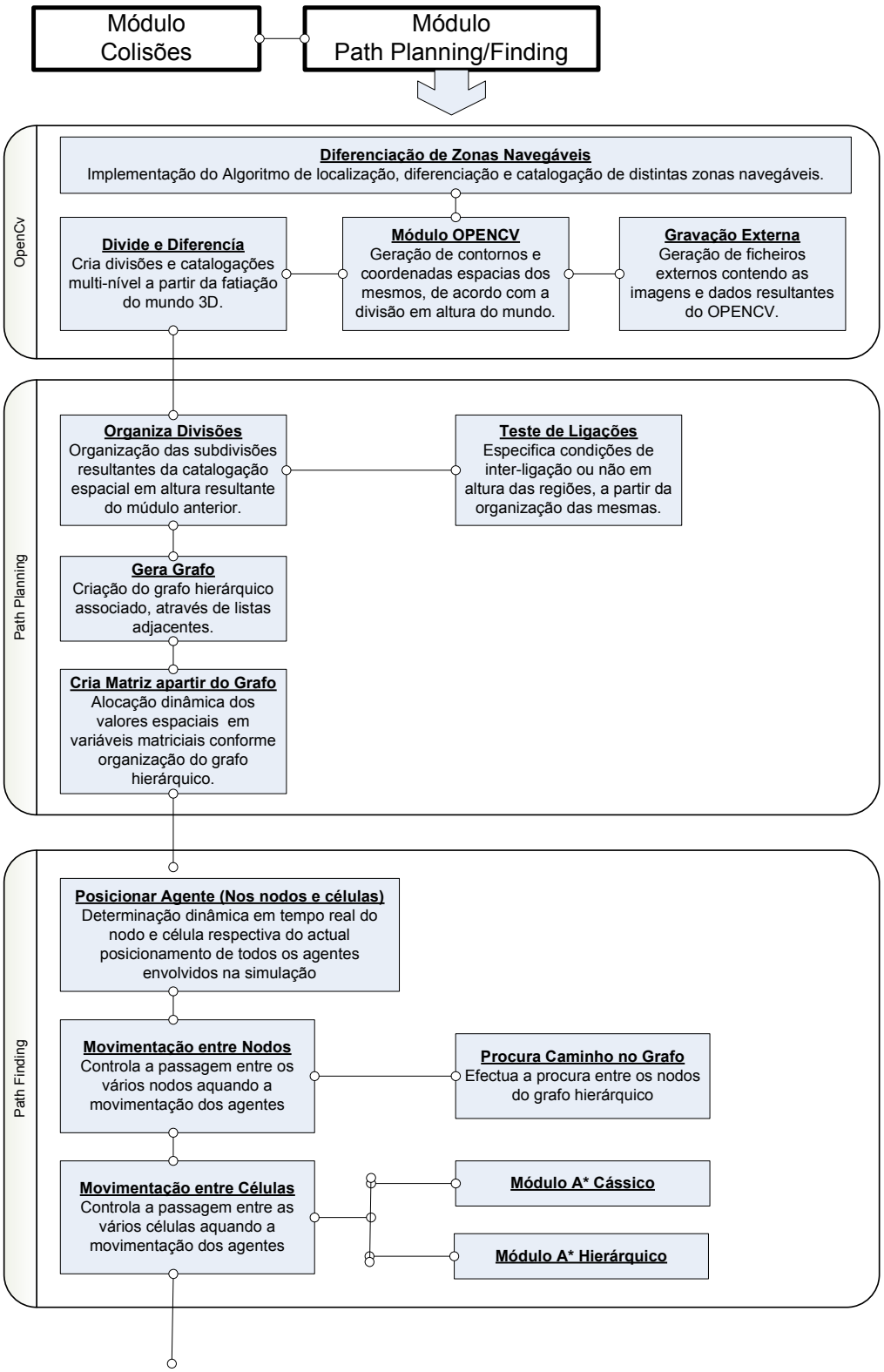
[171] S. T. Coleridge, *Biographia Literaria*. Chapter XIV, 1817.

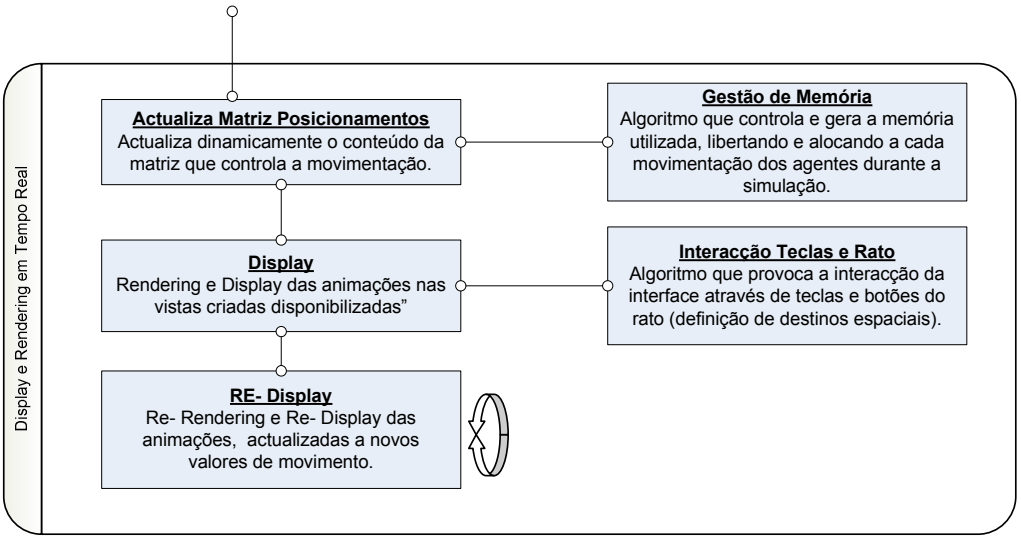
Anexo 1: Módulo Principal

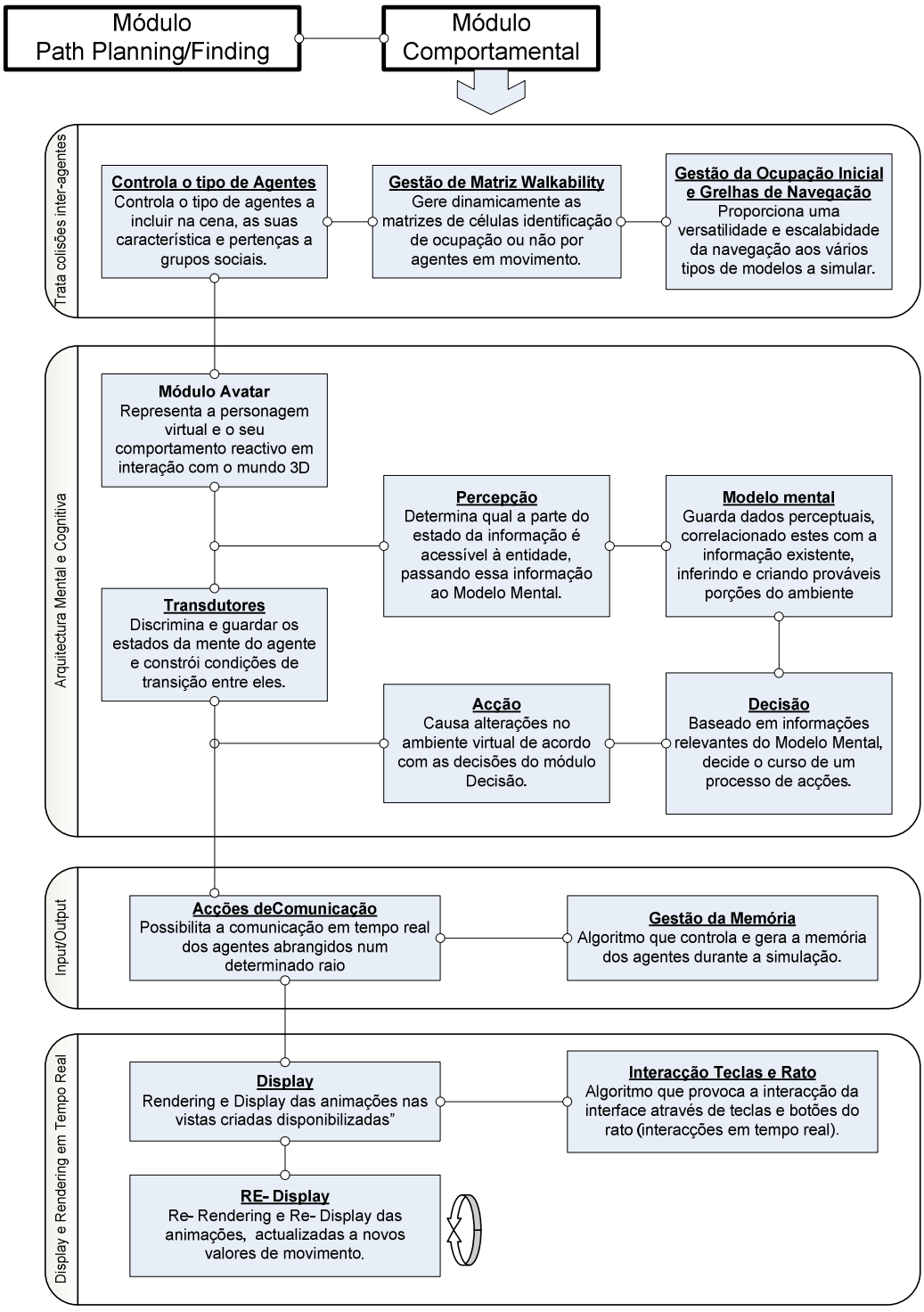



Módulo Colisões
Importação do Espaço a Multi-Nível a Popular
GLM – Carrega
Carrega e inicia qualquer ficheiro OBJ representativo do mundo a navegar ou objecto a testar colisões.
Viewports de visualização
São definidas 4 vistas de apresentação e navegação.
Limita a área de navegação
É definida a área máxima de navegação dos agentes (janelas de visualização).
GLM – Adapta
Altera o modelo OBJ do mundo de forma a adequar-se às proporções dos agentes (escala e posição, smooth, material, polígonos, etc).
Fatiação e Obtenção do Mapa de Alturas (Matriz 3D)
Trata Imagem z-Buffer
Módulo onde se capturam as imagens do Buffer de profundidade. Cálculo da distancia (altura) p/ cada pixel da imagem, sendo guardados os dados numa pilha em memória (matriz 3D).
Algoritmo de Fatiação
Fatiação descendente complementarmente ao módulo "Trata imagem Z-Buffer" para a aquisição das camadas que formam o mapa 3D do modelo.
Legalidade do Movimento
Testa se após o movimento vertical a nova altura a que se situará o topo da cabeça do avatar fica ainda abrangido pela fatia actua ou de altura posterior), legalizando o movimento.
TestSlice
Executa o teste de conteúdo relevante de cada nova fatia processada.
Se a fatia contiver informação relevante, então o algoritmo guarda ou testa se a fatia anterior poderá ser substituída pela fatia actual.
Alocação dos Agentes ao Ambiente Envolvente
Alocação de Memória para Agentes
Aloca memória dependendo do número de agentes a carregar, através de matrizes com a dimensão do número de agentes, gerando individualidade na diversa informação armazenada (patamar associado, altura do agente, direcção, estado, movimento, etc).
Muda de Agentes
Módulo que dependendo do número de agentes especificado altera para o tipo de agente a carregar (Módulo "Constrói Agente "Leve" ou "Cal3D"). para a animação se manter fluida.
Colocação de Agentes no Plano Inferior do Mundo Carregado
Leitura do mundo carregado no módulo "GLM", especificação do plano inferior de navegação, colocação inicial de agentes nesse plano.
Decisão do Tamanho Mínimo das Células
Definição do tamanho das células a serem utilizadas para a identificação espacial da ocupação das personagens envolvidas na animação


Importação dos Agentes a Incluir na Simulação
Constrói Agente "Leve"
Construção de um agente de poucos polígonos sem comportamento de alto-nível.
Cal3D - Carrega
Módulo que carrega e anima com diversos tipos de comportamento algumas personagens 3D virtuais mais complexas, habilitadas a navegar o mundo.
Cal3D - Altera
Módulo que altera entre as personagens carregadas, e o seu comportamento. Especificações de escala e número de polígonos são aqui também especificados.
Movimentação e Teste de Colisões com o Ambiente
Direcção Aleatória
Gera a direcção aleatória a tomar quando o avatar ou agente detecta colisão.
Subir Degraus
Testa o mapa de altura nas células em torno da localização do agente dependendo da direcção deste, diferenciando degraus ultrapassáveis ou colisões, tem em conta o módulo " Muda Nível de Teste de Colisão".
Muda Nível de Teste de Colisão
Algoritmo usado para detectar quando um agente atinge em altura um novo nível de fatiação e consequentemente um novo nível de fatiação de teste de colisões.
Movimenta Agente
Especifica qual a direcção a tomar pelo agente, e a sua nova posição horizontal dependendo do retorno do módulo "Subir Degraus".
Display e Rendering em Tempo Real
FPS
Constrói e Mostra Online os Frames por segundo (FPS) da animação.
Display
Rendering e Display das animações nas vistas criadas no módulo inicial "Viewports de Visualização"
Interacção Teclas e Rato
Algoritmo que provoca a interacção da interface através de teclas e botões do rato (escala, rotação, translação).
RE- Display
Re- Rendering e Re- Display das animações, actualizadas a novos valores de movimento.
DevIL
Módulo de captura de imagens estáticas de diferentes formatos (screenshoots) das vistas apresentadas, permitindo também a criação de vídeo.


Módulo Colisões
Módulo Path Planning/Finding
OpenCv
Diferenciação de Zonas Navegáveis
Implementação do Algoritmo de localização, diferenciação e catalogação de distintas zonas navegáveis.
Divide e Diferencía
Cria divisões e catalogações multi-nível a partir da fatiação do mundo 3D.
Módulo OPENCV
Geração de contornos e coordenadas espacias dos mesmos, de acordo com a divisão em altura do mundo.
Gravação Externa
Geração de ficheiros externos contendo as imagens e dados resultantes do OPENCV.
Path Planning
Organiza Divisões
Organização das subdivisões resultantes da catalogação espacial em altura resultante do múdulo anterior.
Teste de Ligações
Especifica condições de inter-ligação ou não em altura das regiões, a partir da organização das mesmas.
Gera Grafo
Criação do grafo hierárquico associado, através de listas adjacentes.
Cria Matriz apartir do Grafo
Alocação dinâmica dos valores espaciais em variáveis matriciais conforme organização do grafo hierárquico.
Path Finding
Posicionar Agente (Nos nodos e células)
Determinação dinâmica em tempo real do nodo e célula respectiva do actual posicionamento de todos os agentes envolvidos na simulação
Movimentação entre Nodos
Controla a passagem entre os vários nodos aquando a movimentação dos agentes
Procura Caminho no Grafo
Efectua a procura entre os nodos do grafo hierárquico
Movimentação entre Células
Controla a passagem entre as vários células aquando a movimentação dos agentes
Módulo A* Cássico
Módulo A* Hierárquico


Display e Rendering em Tempo Real
Actualiza Matriz Posicionamentos
Actualiza dinamicamente o conteúdo da matriz que controla a movimentação.
Gestão de Memória
Algoritmo que controla e gera a memória utilizada, libertando e alocando a cada movimentação dos agentes durante a simulação.
Display
Rendering e Display das animações nas vistas criadas disponibilizadas"
Interacção Teclas e Rato
Algoritmo que provoca a interacção da interface através de teclas e botões do rato (definição de destinos espaciais).
RE- Display
Re- Rendering e Re- Display das animações, actualizadas a novos valores de movimento.


Módulo Path Planning/Finding
Módulo Comportamental
Trata colisões inter-agentes
Controla o tipo de Agentes
Controla o tipo de agentes a incluir na cena, as suas característica e pertenças a grupos sociais.
Gestão de Matriz Walkability
Gere dinamicamente as matrizes de células identificação de ocupação ou não por agentes em movimento.
Gestão da Ocupação Inicial e Grelhas de Navegação
Proporciona uma versatilidade e escalabidade da navegação aos vários tipos de modelos a simular.
Arquitectura Mental e Cognitiva
Módulo Avatar
Representa a personagem virtual e o seu comportamento reactivo em interação com o mundo 3D
Percepção
Determina qual a parte do estado da informação é acessível à entidade, passando essa informação ao Modelo Mental.
Modelo mental
Guarda dados perceptuais, correlacionado estes com a informação existente, inferindo e criando prováveis porções do ambiente
Transdutores
Discrimina e guardar os estados da mente do agente e constrói condições de transição entre eles.
Acção
Causa alterações no ambiente virtual de acordo com as decisões do módulo Decisão.
Decisão
Baseado em informações relevantes do Modelo Mental, decide o curso de um processo de acções.
Input/Output
Acções deComunicação
Possibilita a comunicação em tempo real dos agentes abrangidos num determinado raio
Gestão da Memória
Algoritmo que controla e gera a memória dos agentes durante a simulação.
Display e Rendering em Tempo Real
Display
Rendering e Display das animações nas vistas criadas disponibilizadas"
Interacção Teclas e Rato
Algoritmo que provoca a interacção da interface através de teclas e botões do rato (interacções em tempo real).
RE- Display
Re- Rendering e Re- Display das animações, actualizadas a novos valores de movimento.