Instituto Militar de Engenharia
Departamento de Engenharia de Sistemas
Relatório Técnico nº 106/DE9/04

editores: Prof. Ulf Bergman
Prof. Paulo Fernando Ferreira Rosa

Departamento de Engenharia de Sistemas

título: Algoritmo de oclusão para tratamento de objetos em um SIG 3D

autores: Fábio André M. Porto, D.Sc.
Jauvane C.de Oliveira, Ph.D.
Ermírio de Siqueira Coutinho

RT 106/DE9/Set 04

Algoritmo de oclusão para tratamento de objetos em um SIG 3D

Fábio André Machado Porto, D.Sc.
Jauvane C.de Oliveira, Ph.D.
Ermírio de Siqueira Coutinho

Instituto Militar de Engenharia
Depto de Engenharia de Sistemas
Pça Gal Tibúrcio 80
CEP 22290-270 Rio de Janeiro - RJ
e-mail : fabio.porto@epfl.ch; jauvane@lncc.br; ermirio@ime.eb.br

Abstract

This work introduces an occlusion algorithm which was developed for deployment in optimization in scene generation for three-dimensional Geographical Information Systems (3D-GIS). We have adopted a conservative strategy through occlusion pre-processing for fix objects. The results of such processing are later stored in the same Database Management System (DBMS) of the GIS. During a simulation of the virtual environment the visible objects are then recovered based on occlusion volumes, generated in the previous step, through queries on the DBMS spatial indexing system. The goal is to reduce the amount of objects which are recovered even though not needed for the rendering of the scenario. Such filtering mechanism allows an optimized processing and enabling a feature-rich, fluid rendering of the scenes.

1. Introdução:

Durante muitos anos a visualização de uma porção da superfície terrestre tem sido feita utilizando-se uma projeção em duas dimensões do terreno real, seja através de cartas topográficas em papel, ou de Sistemas de Informação Geográfica (SIG). Tal restrição imposta pela tecnologia existente na época, exige grande capacidade de abstração por parte do usuário e limita a análise espacial.

Com o advento dos Sistemas de Realidade Virtual (RV) e Ambientes Virtuais Colaborativos (AVC), surgiu a possibilidade de visualização da superfície terrestre tal como ela se apresenta na natureza. Os Ambientes Virtuais Colaborativos (AVC) são um caso especial de Sistemas de Realidade Virtual (RV), onde diferentes usuários podem interagir através do ambiente de simulação. Se aliarmos a interface de visualização em três dimensões presente em sistemas de RV ou AVC com a capacidade de efetuar-se consultas sobre características dos dados constituintes do ambiente de simulação (metadados) e análises topológicas ou espaciais sobre estes, têm-se um SIG tridimensional (SIG-3D). Com a utilização dessas tecnologias, é possível fazer-se análises espaciais com maior facilidade e realismo, permitindo a visualização e extração de um volume muito maior de informação.

Da análise de artigos já publicados sobre o assunto, observa-se que a maioria dos Sistemas de Realidade Virtual armazena seus dados no Sistema de Arquivos do Sistema

Operacional ou em Bancos de Dados proprietários, criados para atender a um sistema específico. Algumas iniciativas surgiram para armazenar os dados em Sistemas Gerenciadores de Banco de Dados (SGBD). No caso de SIG-3D, existem soluções que apenas unem a visualização tridimensional dada pelo sistema de RV a um SIG bidimensional, onde são feitas as consultas e análises, mantendo-se os dados do sistema de RV armazenados no sistema de arquivos e empregando o SGBD apenas para o SIG. A utilização de um SGBD único para o armazenamento de dados e metadados em SIG-3D leva a este benefícios aliados à segurança, integração, uniformidade de acesso e independência de dados, entre outros.

Um pré-requisito fundamental em aplicações de RV, AVC e SIG-3D é a sensação de imersão no ambiente e para que isso seja possível durante a simulação, é imprescindível a manutenção de uma taxa constante de exibição, medida em quadros por segundo (*frames per second – fps*). Neste contexto, a utilização de SGBDs requer o desenvolvimento de técnicas que otimizem a recuperação e o armazenamento dos dados constituintes do ambiente, colaborando no alcance das taxas de exibição requeridas. Neste contexto, um dos aspectos relevantes é a identificação dos dados (modelos tridimensionais dos objetos, objetos 3D ou feições) necessários e suficientes para a formação de uma cena a partir de um observador. Vários fatores podem se fazer importantes neste momento, tais como: alcance visual do observador, oclusão entre objetos, interesse semântico na formação da cena e instante de tempo. Dentre esses fatores, a identificação dos objetos desnecessários à geração da cena, por se encontrarem oclusos por outros mais próximos ao observador, reduz drasticamente a quantidade de dados a serem manipulados, principalmente durante simulações em ambientes urbanos.

Considerando-se o que foi exposto, este trabalho aborda o problema de recuperação de modelos tridimensionais de objetos em SGBD Relacional-Objeto, buscando otimizá-lo com o emprego de um algoritmo de oclusão desenvolvido especificamente para este sistema. O trabalho se insere em um contexto maior de desenvolvimento de um sistema completo de apoio à recuperação e exibição de objetos 3D em aplicações de AVC e SIG-3D.

2. Organização deste Relatório:

Na seção 3 (três) enumeram-se as premissas adotadas que norteiam este trabalho; na seção 4 (quatro) aborda-se alguns trabalhos relacionados ao assunto; na seção 5 (cinco) apresenta-se o conceito de envoltória e explica-se a sugestão de utilização de duas envoltórias por objeto para que o emprego do algoritmo apresentado neste trabalho resulte numa abordagem conservadora. Na seção 6 (seis) apresenta-se o algoritmo de oclusão desenvolvido durante o presente trabalho, cuja implementação é descrita a seção 7 (sete) e avaliada na seção 8 (oito). Por fim, a seção 9 (nove) conclui o presente trabalho.

3. Premissas adotadas:

Desde o início do desenvolvimento do presente trabalho as seguintes premissas nortearam o trabalho:

- *adoção de um SGBD no armazenamento de objetos para RV:* considera-se que os benefícios trazidos pela utilização de um SGBD justificam sua adoção, bem como permitem o desenvolvimento de novas técnicas voltadas para um eficiente armazenamento e recuperação dos objetos;

- *tratar oclusão ao nível do SGBD é uma boa estratégia:* evita a busca no SGBD e transmissão para a camada de visualização de dados que não serão utilizados para a geração da cena na simulação. Em simulações de regiões urbanas, representa uma diminuição drástica da quantidade de dados processada, o que pode ser decisivo para garantir a sensação de imersão no ambiente, através da sustentação da taxa mínima de quadros gerada pela camada de visualização;

- *estruturas de indexação espaciais existentes podem ser utilizadas para o problema de oclusão 3D:* dada a posição do observador, através de uma estrutura de indexação espacial, existente no SGBD, pode-se de maneira ágil determinar-se quais objetos estão oclusos através dos volumes de oclusão gerados em um pre-processamento.

4. Alguns trabalhos relacionados:

O problema de tratamento de oclusão é uma das principais áreas de estudo no ramo da Computação Gráfica, pois ocorre sempre que se deseja gerar uma visão (bidimensional) a partir de um cenário tridimensional. Tem aplicação direta em jogos, animação e simuladores de vôo ou de outros sistemas, entre outros.

O tratamento de oclusão, em aplicações de computação gráfica, é feito geralmente pelo *hardware* gráfico presente na placa gráfica do computador. Nesse caso, o ambiente completo é gerado em memória e a posição de todos os objetos, passada para a placa gráfica, que através de um algoritmo de oclusão (como o Z-buffer (CATMULL, 1974), por exemplo) define para cada cena em separado e em tempo real, quais objetos são visíveis, descartando todos os demais dados. Em simulações de ambientes complexos, como as simulações de cenários urbanos que geralmente se passam em ambientes abertos, estão presentes no mínimo centenas ou milhares de objetos, cada um geralmente composto por uma quantidade ainda maior de polígonos, dependendo de sua complexidade. Desta maneira, apenas a utilização das técnicas de oclusão da placa gráfica se torna impraticável, devido principalmente a dois aspectos:

1) o volume de dados a ser carregado na memória principal é muito grande, o que exige grande quantidade de memória;

2) em ambientes urbanos, geralmente apenas uma pequena parcela dos objetos existentes é visível a cada instante, sendo que quase a totalidade dos objetos se encontra oclusa. Isso significaria um enorme desperdício de recursos computacionais, pois a maior parte dos dados carregados a partir da memória secundária seria descartada na formação de cada quadro pela placa gráfica.

Assim, se faz necessária a aplicação de alguma técnica de oclusão antes que esses dados sejam enviados para a geração final da cena. Na simulação de ambientes

urbanos, sempre há um tratamento prévio de oclusão, antes do envio de dados para o processamento da placa gráfica.

As técnicas aplicadas nessa fase preliminar, podem ser classificadas em dois grupos: conservadoras e não-conservadoras. No primeiro grupo, estão as técnicas de oclusão que garantem que todos os objetos que não estão oclusos aparecerão na cena final, mesmo que para isso, sejam mandados para a placa gráfica alguns poucos objetos que seriam desnecessários. Para aplicações em áreas que exigem exatidão nos cenários simulados, tais como cartografia, arquitetura, planejamento urbano, ocupação do solo, cadastro territorial, entre outras, a ausência na cena de algum objeto que seria visível no mundo real pode ser crucial. Para estes, apenas as técnicas conservadoras devem ser empregadas, mesmo que a um custo maior de processamento. Este custo maior inviabiliza o seu processamento em tempo real, em decorrência da complexidade dos cenários, que torna o tratamento de oclusão computacionalmente caro. Desta maneira, é preferível optar-se pelo tratamento de oclusão para os objetos estáticos da cena em pré-processamento, com o conseqüente armazenamento dos resultados em memória secundária ou em um SGBD. Considera-se, de forma conservadora, que a parte dinâmica de uma cena é produzida por objetos de pequena dimensão, sem efeito de oclusão sobre os demais elementos da cena. Já para determinadas aplicações, tais como jogos ou algum outro tipo de simulação que não exija exatidão, a utilização de técnicas não-conservadoras permite o processamento em tempo de execução e simplifica a implementação dos aplicativos de visualização. (COHEN et al, 2001) apresenta em detalhes o problema da oclusão e as principais técnicas desenvolvidas para tratá-lo, comparando-as e classificando-as segundo uma taxonomia adotada. O trabalho, desenvolvido em conjunto por pesquisadores do MIT, AT&T Labs e Universidades do Chipre e de Tel-Aviv, apresenta um completo levantamento sobre pesquisas na área e uma ótima fonte de referência para o assunto.

Dentre as diversas soluções analisadas, apresentar-se-á uma solução não-conservadora, por sua abordagem inovadora, e três conservadoras.

A primeira destas soluções, desenvolvida por pesquisadores da Duke University e da University of Illinois, ambas nos EUA, e apresentada em (AGARWAL et al, 2001), trata a oclusão em ambientes urbanos utilizando-se uma técnica de oclusão não-conservadora e executada em tempo real (não pré-processada), se caracterizando como uma exceção à regra. Esta solução introduz uma nova abordagem para tratamento de oclusão que consiste em calculá-la fazendo uso da coerência temporal de quadros – grande semelhança entre quadros consecutivos numa cena, constituídos pelos mesmos objetos, com pequenas variações em suas posições. Para sinalizar a validade da coerência temporal, é adotado um *time stamp* para cada cena, válido até um tempo-limite. Com isso, terminada a validade da cena, esta deve ter sua oclusão reprocessada. Quadros que exijam reprocessamento são chamados de quadros críticos (*critical frames*) e ocorrem quando há alguma mudança significativa no contexto da cena: mudança na direção de deslocamento do observador ou remoção de oclusores. A oclusão processada em tempo de execução permite alterações dinâmicas na cena e tratamento de oclusão de objetos móveis. Por outro lado, para que fosse possível prescindir-se de um pré-processamento, adotou-se um compromisso (*tradeoff*) entre precisão e velocidade de processamento, resultando numa estratégia não-conservadora de oclusão. Com isso, apesar dos autores afirmarem que a taxa de erros (quantidade de objetos que deveriam estar presentes na cena, mas que foram considerados oclusos) é muito pequena, isso

inviabilizaria certas aplicações, como as militares, tratadas no presente trabalho, podendo ser aproveitada futuramente alguma dessas estratégias para o tratamento de oclusão de objetos móveis.

As demais soluções aqui apresentadas tratam o problema da oclusão de uma forma conservadora. Em (COHEN, 1998), desenvolvido na Universidade de Tel-Aviv (Israel), é apresentada uma estratégia de oclusão baseada no particionamento do ambiente em células bidimensionais, a partir das quais são feitos, em um pré-processamento, os testes de oclusão com grandes oclusores e obtidos seus conjuntos de objetos possivelmente visíveis (Potential Visibility Set – PVS). Desta maneira, dentro de cada célula, se garante que o mesmo conjunto de objetos será sempre visível.

O (WONKA et al, 2000), desenvolvido na Universidade de Tecnologia de Vienna (Áustria), apresenta uma variação da solução baseada na discretização do ambiente virtual em células. Neste trabalho, os testes de oclusão são feitos a partir de alguns pontos obtidos por amostragem nas arestas de cada célula. Para cada um desses pontos é obtido um PVS. Da união dos PVS de cada ponto resulta um PVS relativo a toda a célula. Desta maneira, o algoritmo trata não só a oclusão decorrente de grandes oclusores, como também a decorrente da fusão de pequenos. A técnica foi testada em um modelo da cidade de Vienna e foram relatados resultados positivos.

Também utilizando a divisão do ambiente em células, foi desenvolvido em um projeto-conjunto do MIT (EUA) e do INRIA (França), um algoritmo que pré-processa a oclusão conservadora a partir da fusão de pequenos oclusores (SCHAUFLER et al, 2000). Os voxels (pixels tridimensionais) constituintes do cenário são classificados em *opacos* (internos a objetos), *vazios* (não pertencem a nenhum objeto) e *limites* (limitam os objetos). Após a classificação, as regiões de voxels vazios (consideradas de circulação de observadores) são divididas em unidades chamadas *células de visão* (viewcell). A partir dessas *viewcells* são traçadas linhas ligando seus limites aos limites dos objetos, definindo *espaços* (volumes) *de oclusão*. O algoritmo testa a fusão de pequenos oclusores para tornar mais eficiente a oclusão. São apresentados resultados comprovando a eficiência do algoritmo.

5. Envoltórias:

A criação de envoltórias visa simplificar o teste de oclusão. Pode assumir as formas de esfera, cubo, paralelepípedo, elipsóide ou outras mais complexas de modo a se adequar à forma da maior parte dos objetos que representa ou em função de detalhes físicos do sistema. Podem também ser chamadas de volumes envolventes mínimos ou *envelopes*. Em Silva (2002) é feito um estudo aprofundado do assunto. Nesse estudo, dentre outras coisas, foram analisadas as formas de envoltórias que apresentavam a melhor adaptação média a objetos de formas genéricas, a fim de testar sua oclusão. Após os testes, o referido trabalho conclui que uma das formas para envoltória que melhor se adaptou a objetos de desenho genérico foi a pastilha, ilustrada na figura 5.1.

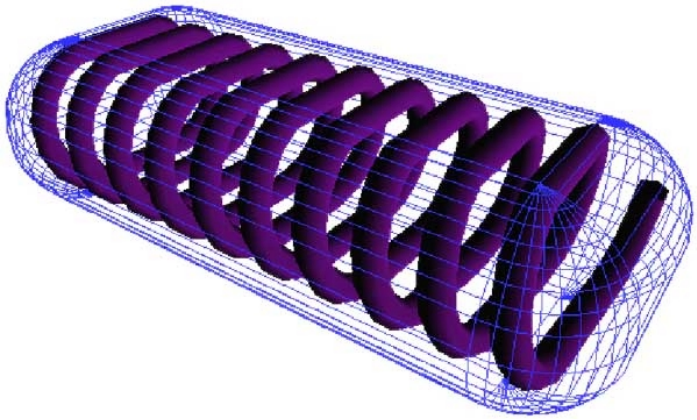

Figura 5.1 – Exemplo de envoltória na forma de pastilha (SILVA, 2002).

Analisando-se em Silva (2002) os paralelepípedos (chamados de "caixas", naquele trabalho) na adaptabilidade a objetos de forma genérica, observa-se que seu desempenho não fica muito abaixo do obtido com a utilização de pastilhas. Considerando-se seu desempenho próximo ao das pastilhas e a fim de se adequar à forma de particionamento do espaço empregada pelas estruturas de indexação do SGBD, apesar da pastilha ser a forma que melhor se adaptou a objetos de desenhos genéricos, optou-se por adotar paralelepípedos como envoltórias. Foram adotadas duas envoltórias por objeto, uma para quando ele se encontra na condição de oclusor e outra para quando é testado para se verificar se está ocluso para o observador ou não. Este cuidado se revelou necessário para manter a oclusão conservadora, adotada neste SIG-3D. Uma envoltória só engloba um objeto de cada vez. Cada objeto possui apenas duas envoltórias: uma interna e outra externa. Não são tratados objetos que precisem da união de mais de uma envoltória externa ou interna. Para gerá-las, deve-se abrir o arquivo VRML (ISO, 1997) que o representa, identificar suas coordenadas limites e criar manualmente um bloco que as envolva, uma vez que apesar de existir uma função envelope (OGC, 1998), esta está especificada apenas para figuras geométricas bidimensionais até o momento.

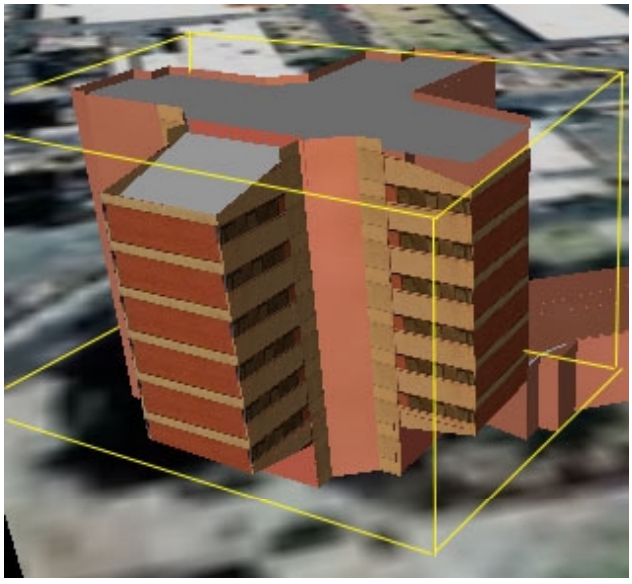

Figura 5.2 - Envoltória para teste de oclusão do objeto: menor paralelepípedo, com base no plano XY (nível do solo), que contém todo o objeto. Vista oblíqua. *Adaptado* (THE UNIVERSITY OF ARIZONA, 2004).

Imagine-se o caso de um oclusor irregular, conforme apresentado nas figuras 5.2 e 5.3. Caso a envoltória apresentada fosse utilizada, os objetos que aparecem logo abaixo dos quartos em balanço, apontados pela seta, seriam considerados oclusos, não sendo apresentados pela e a cena ficaria inconsistente.

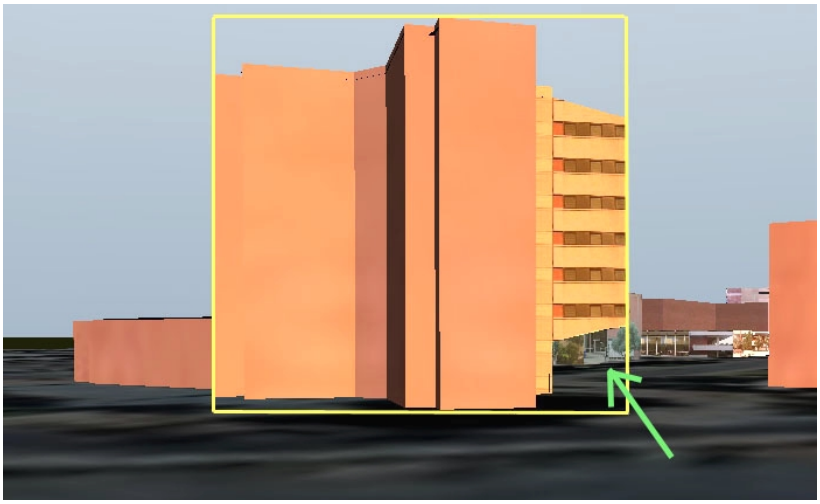

Figura 5.3 - Envoltória para teste de oclusão do objeto: menor paralelepípedo, com base no plano XY (nível do solo), que contém todo o objeto. Vista lateral. *Adaptado* (THE UNIVERSITY OF ARIZONA, 2004).

Já na figura 5.4, foi usada uma envoltória que considera a oclusão de uma maneira conservadora, considerando-se sempre a pior hipótese, de modo a garantir cenas consistentes durante a simulação. Esta será a envoltória utilizada quando o objeto for testado como oclusor.

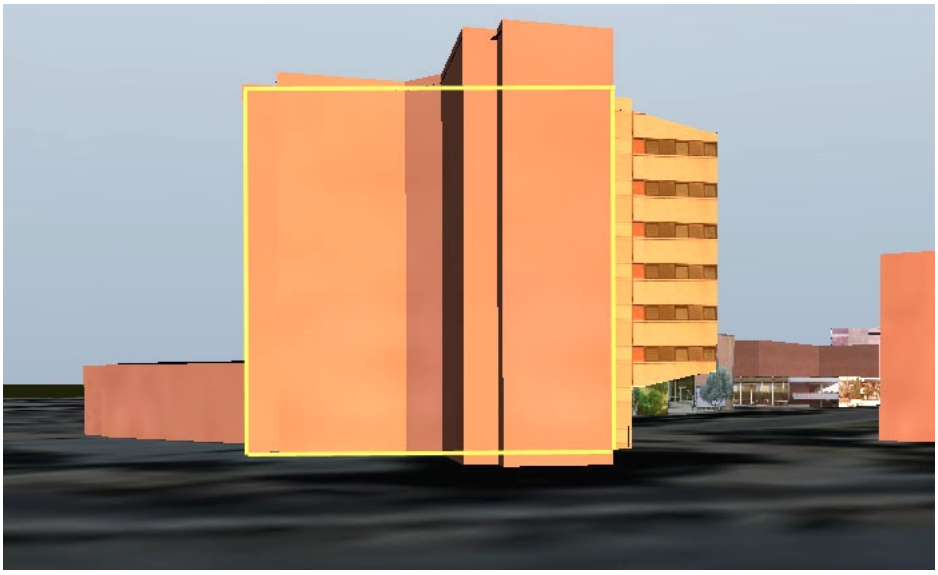

Figura 5.4 - Envoltória para objeto oclusor: maior paralelepípedo, com base no plano XY (nível do solo) todo contido no objeto. *Adaptado* (THE UNIVERSITY OF ARIZONA, 2004).

Resumindo, temos como envoltória, em função da situação do objeto:

- *objeto cuja oclusão é testada:* adotou-se o menor paralelepípedo, com base no plano XY (nível do solo), que contivesse todo o objeto, incluindo seus prolongamentos;

- *objeto como oclusor:* adotou-se o maior paralelepípedo, com base no plano XY (nível do solo), que possa estar contido no objeto.

Desta maneira, garante-se sempre a correta oclusão, de acordo com a estratégia de oclusão conservadora adotada.

6. Algoritmo de oclusão para tratamento de objetos em um SIG 3D:

Em um SIG-3D, como o que motiva este trabalho, deve-se dar especial atenção à necessidade de fluidez da imagem nas cenas que geram o ambiente virtual, garantindo assim a sensação de imersão, durante a simulação. Dentre os fatores responsáveis para atender a tal exigência, destaca-se a necessidade de um taxa mínima de reposição de quadros. Para que isso seja possível, se faz necessária a otimização de todo o processo de carga dos dados utilizados na geração da cena que, em um SIG-3D, se encontram armazenados em um SGBD. Para que se possa identificar quais etapas da carga de dados podem ser otimizadas e como organizá-los em disco a fim de proceder a essa otimização, se faz necessário o entendimento de como funciona um ambiente de SIG-3D como o que motivou este trabalho.

6.1. Justificativa do emprego de um SGBD:

Nesse tipo de sistema, é muito grande o número de objetos tridimensionais a serem armazenados. Para se quantificar o volume de dados existentes, basta imaginar o número de objetos tridimensionais necessários na representação de uma cidade, por exemplo (cada edificação, poste e hidrante, para citar apenas alguns exemplos, é representado por um objeto tridimensional diferente). Deve-se observar que os objetos tridimensionais estão espacialmente distribuídos e que cada um possui diversos atributos. Para ilustrar, no caso de uma ponte, esta possui, além de sua representação visual (forma, cor e textura), dados sobre sua localização (coordenadas e orientação) e atributos que armazenam suas características (carga dinâmica máxima aplicada, tipo de pavimento, capacidade estática de carga, ano de construção, tipo de fundação e órgão responsável pela manutenção, por exemplo).

Em resumo, observa-se que:

- é grande o volume de dados e atributos a serem armazenados;
- para agilizar o armazenamento e a recuperação dessas informações, seria interessante uma organização que levasse em conta a posição (coordenadas) de cada um dos objetos tridimensionais;
- durante a visualização poderão ocorrer alterações permanentes sobre os objetos tridimensionais (demolição por acidentes, bombas, etc);
- como se trata de um AVC, possuindo provavelmente vários usuários (aqui chamados de observadores, cada um representado no sistema por um *avatar*), para cada um será gerada uma visualização diferente. Mais ainda,

por se tratar de uma simulação de emprego militar, as ações de um usuário que alterarem o cenário, como por exemplo, a destruição de uma ponte, devem aparecer para todos que compartilham a exibição desse objeto tridimensional. Sendo assim, faz-se necessário um controle de concorrência nos acessos aos dados que compõe a cena durante a simulação.

Do acima exposto, verifica-se a conveniência em se utilizar um sistema gerenciador de bancos de dados (SGBD) para armazenamento dos objetos tridimensionais em um ambiente de SIG-3D.

6.2. Algoritmos de Oclusão:

A fim de se justificar o emprego de algoritmos de oclusão ao nível da camada de persistência, apresentar-se-á o funcionamento hipotético de um SIG-3D, mostrando gradativamente as possibilidades de otimização, no que diz respeito à carga de dados. Imaginou-se um SIG-3D cuja visualização estivesse em *1ª pessoa*. Neste caso, a cena a ser gerada é a que seria vista por uma pessoa inserida no ambiente onde se encontram os objetos de consulta do SIG, tal como ocorre em um ambiente virtual ou em jogos *(video-games)* em 1ª pessoa.

6.2.1. Abordagem hipotética de um SIG-3D sem otimizações quanto à carga de dados:

Uma abordagem possível para a geração das cenas, durante a simulação, seria a carga em memória de todos os dados que compõem o ambiente. Como exemplo, imaginemos que desejamos executar consultas de SIG sobre um ambiente que representa uma cidade e que para tal, iniciemos com o observador colocado em uma de suas ruas. A fim de facilitar o raciocínio, mas sem perda de generalidade, imaginemos que este observador encontra-se parado e sempre olhando numa mesma direção. Para a geração da cena, recuperam-se do SGBD os dados relativos à representação tridimensional de todos os objetos constituintes da cidade. Com isso, percebe-se porém, que a maior parte dos objetos recuperados era desnecessária para a geração das cenas, por se encontrar atrás do observador. Sendo assim, devemos buscar reduzir o conjunto de objetos recuperados de modo a se aproximar ao máximo do conjunto de objetos estritamente necessários à geração da cena (ou seja, os objetos que serão visíveis ao observador naquele momento) e assim reduzir o tempo gasto na recuperação de objetos que serão descartados na geração da cena. A primeira melhoria que podemos introduzir para diminuir o número de objetos desnecessários é recuperar apenas aqueles que estejam à frente do observador. Para isso, definiremos seu campo de visão.

6.2.2. Primeira otimização – recuperação apenas dos objetos no campo de visão do observador:

Podemos definir o campo de visão do observador, como sendo a região do espaço visível pelo mesmo, em um determinado instante e dada por sua posição (A), direção de visada, amplitude do campo visual (α) e acuidade visual (alcance da visão do observador, ou seja, maior distância na qual o observador consegue perceber um objeto). Desta forma, podemos considerar o campo de visão como sendo um cone, cuja base seria o setor de uma esfera. A fim de simplificarmos o entendimento e a implementação do conceito, vamos considerar sua projeção no plano sobre o qual se

encontra o observador (neste caso, o plano XY), conforme ilustrado na figura 6.1 abaixo. Tem-se então que a acuidade visual do observador é dada pela distância AB (= distância AC). Devemos observar que a altura do triângulo ABC não tem obviamente o mesmo comprimento que um dos lados (AB ou AC), sendo que a representação ideal para a projeção do campo de visão do observador, em função de sua acuidade visual seria um setor circular. Considerando-se a acuidade visual de um observador como um valor aproximado, decidiu-se por sua representação como um triângulo, ao invés de um setor circular, com a finalidade de se simplificar os cálculos de suas coordenadas limites. Em função desses quatro parâmetros (posição do observador, direção de visada, amplitude do campo visual (α) e acuidade visual), é possível definir as coordenadas limites do campo de visão do observador (A,B e C).

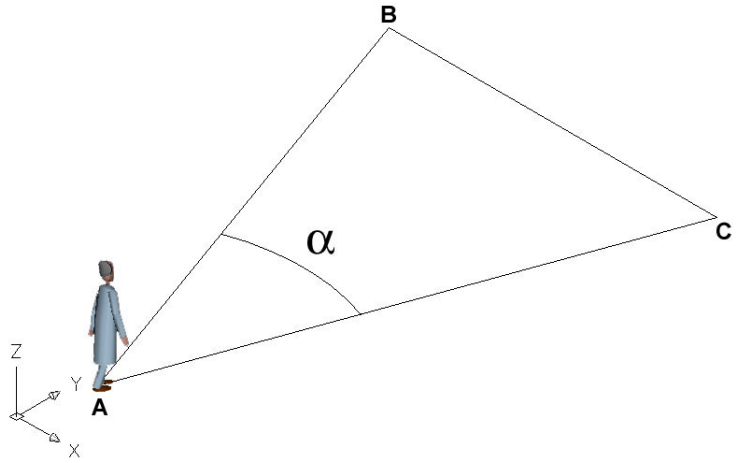

Figura 6.1 – Campo de visão do observador. Ilustrou-se a projeção do campo de visão no plano XY, a fim de se facilitar a compreensão.

O emprego do conceito de campo de visão, reduz em muito a quantidade de objetos desnecessários recuperados do SGBD. Para ilustrar a quantidade de objetos que, com o emprego deste conceito, não são recuperados desnecessariamente, imagine-se um ambiente urbano como o ilustrado na figura 6.2. Nele, encontra-se achurado o campo de visão do observador. Os objetos que devem ser recuperados se encontram desenhados em vermelho. Os objetos em preto não serão recuperados para a formação da cena.

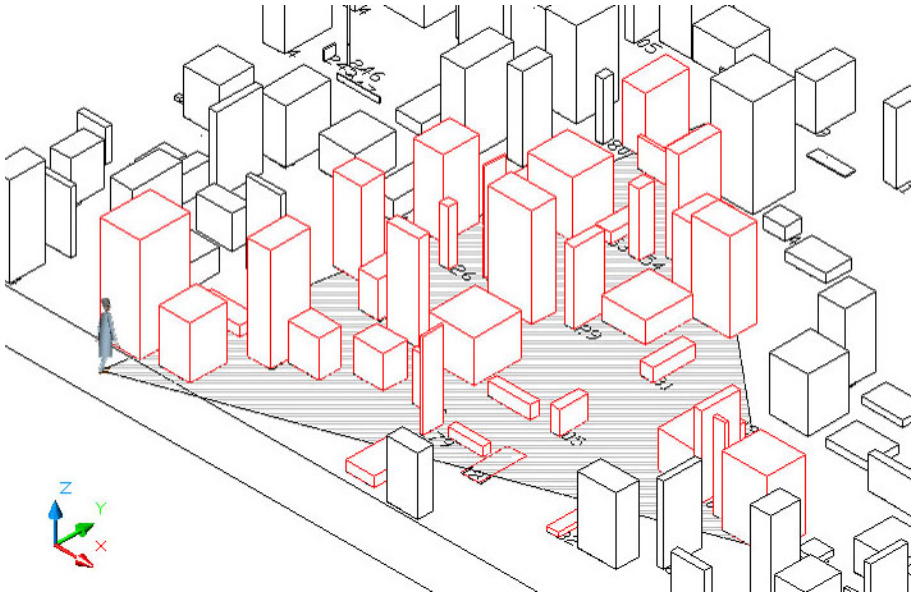

Figura 6.2 – Exemplo de utilização do conceito de campo de visão. Obs: o avatar do observador se encontra fora de escala a fim de facilitar sua visualização na figura.

Observa-se que com relação a todos os objetos constituintes de uma cidade virtual, apenas uma pequena parte deles foi recuperada, utilizando-se o conceito de campo de visão. Apesar disso, mesmo recuperando-se apenas os objetos dentro do campo de visão do observador, muitos continuam desnecessários para a formação da cena. Observando-se a figura 6.3, percebe-se que na confecção da cena que ilustra a visão do avatar, apenas os blocos mais próximos serão usados, pois vários encontram-se atrás dos que estão em primeiro plano, conforme ilustrado no figura 6.4.

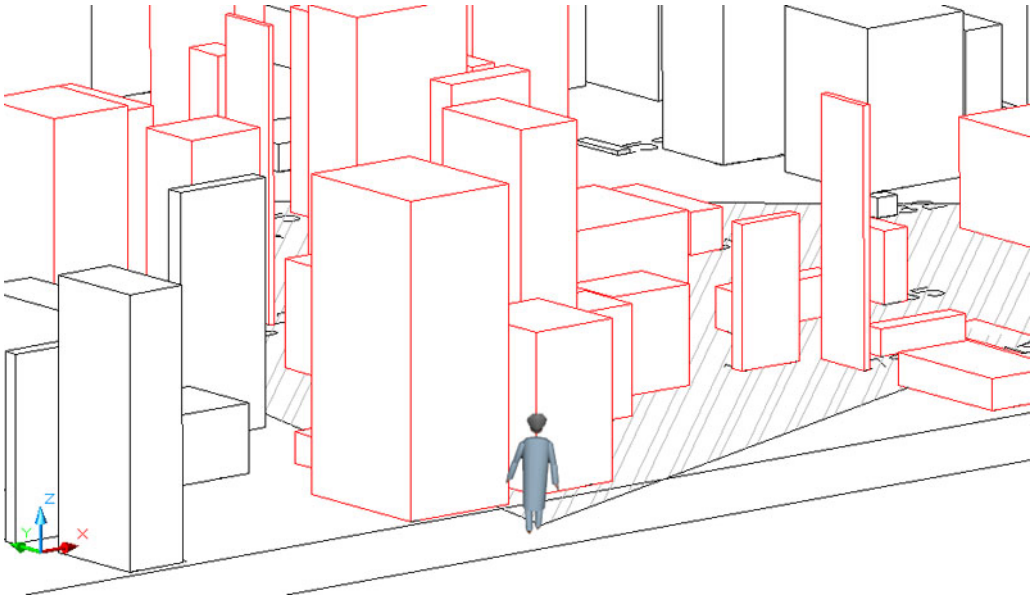

Figura 6.3 – Campo de visão do observador. Obs: o avatar do observador se encontra fora de escala a fim de facilitar sua visualização na figura.

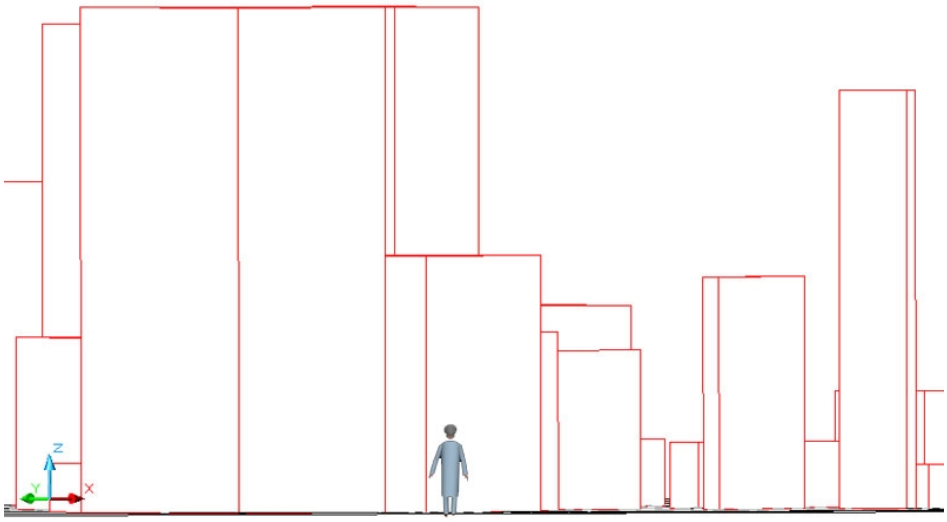

Figura 6.4 – Cena semelhante a que representa a visão do observador. Obs: o avatar do observador se encontra fora de escala a fim de facilitar sua visualização na figura e foi colocado só para dar um referencial do ponto de vista na cena.

Nota-se, então, a necessidade de mais uma otimização, a fim de aproximar o conjunto dos objetos selecionados para formar a cena, do constituído pelos objetos visíveis pelo observador.

6.2.3. Segunda otimização – Tratamento de Oclusão sobre os objetos no campo de visão do observador:

A fim de se evitar o problema apresentado na seção acima, ou seja, a recuperação de objetos que apesar de estarem no campo de visão do observador, não são úteis na composição da cena por não serem visíveis por este, deve-se submetê-los a um tratamento de oclusão. Para se ilustrar o conceito de Oclusão, imagine-se a situação apresentada na figura 6.5 abaixo.

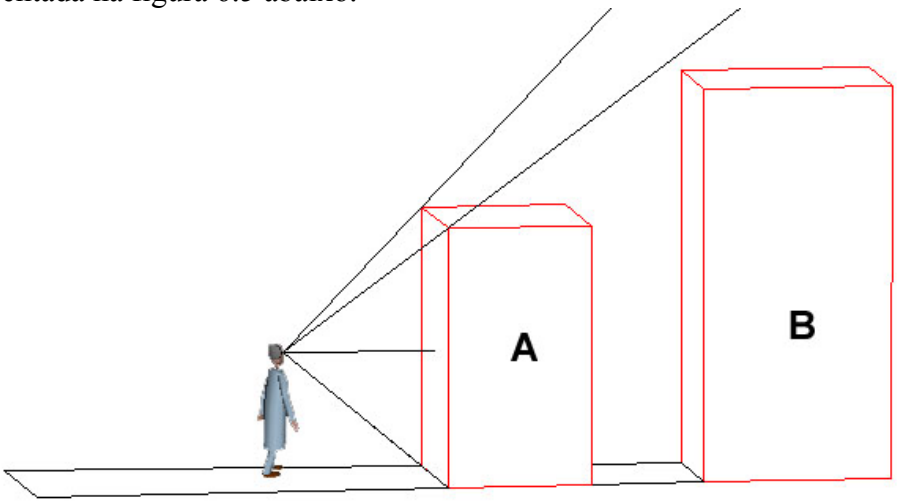

Figura 6.5 – Ilustração do conceito de Oclusão: B objeto ocluso por A em relação ao observador. Os traços em preto representam as várias linhas de visada do observador. Em nenhuma delas ele consegue ver o objeto B.

Apesar do observador estar olhando no sentido de B, ele não pode vê-lo porque este se encontra escondido (ocluso) por A. Diz-se que o objeto B está ocluso por A em relação ao observador, se este último ao olhar no sentido de B só consegue ver o objeto A, que encontra-se a sua frente e encobre B. No caso ilustrado, A recebe o nome de oclusor de B em relação ao observador, ou apenas, oclusor, enquanto B é o objeto ocluso por A em relação ao observador, ou apenas, objeto ocluso.

Em um SIG-3D, como o que motivou este estudo, considera-se o tratamento de oclusão em dois níveis:

- **na recuperação de objetos do banco de dados que serão enviados à camada de aplicação:** visa reduzir o volume de dados a ser recuperado a partir do banco de dados durante a geração de uma cena. Este tratamento é feito a nível do SGBD;

- **na geração da cena ("renderização"):** visa gerar a cena bidimensional que será exibida ao observador a partir da posição espacial de cada objeto recuperado do banco de dados e é feito a nível da camada de aplicação.

Como vários estudos já foram desenvolvidos sobre tratamento de oclusão na geração da cena e o assunto, apesar de sua evolução contínua, já se encontra bastante pesquisado e como este trabalho busca se concentrar apenas na camada de persistência de um SIG-3D, o estudo aqui desenvolvido se atém ao primeiro caso apresentado acima: tratamento de oclusão na recuperação de objetos do banco de dados que serão enviados à camada de aplicação. Apesar de toda a pesquisa bibliográfica feita durante o desenvolvimento deste trabalho, não foi encontrada nenhuma referência a algum estudo que abordasse a oclusão da maneira aqui conduzida, podendo-se portanto considerar este tratamento de oclusão como uma contribuição do presente trabalho.

6.2.4. Estratégia de tratamento de oclusão desenvolvida:

A implementação de objetos 3D em SGBD deve considerar as técnicas adotadas nestes sistemas para tornar eficiente sua recuperação. Em particular, o acesso a dados em SGBD se torna eficiente com a adição de técnicas de projeto físico. Diferentemente dos sistemas de aplicação, tais como os de computação gráfica, onde boa parte das técnicas vislumbra dados carregados em memória, SGBD consideram estruturas criadas (pré-processadas) armazenadas em disco, acessadas por estruturas de indexação (tais como árvores B+, R, GiST, entre outras) adotadas para oferecer acesso direto ou semi-direto aos objetos procurados. Desta maneira, torna-se desejável que no SIG-3D aqui proposto, tanto os objetos constituintes do ambiente virtual, quanto resultados da aplicação do algoritmo de oclusão sobre esses objetos, possam fazer uso das estruturas de indexação do SGBD. Essa exigência, influenciou de maneira determinante o desenvolvimento do algoritmo de oclusão para esta aplicação em dois aspectos:

- **granularidade do teste de oclusão:** geralmente em um sistema de RV, diferentemente do que ocorre neste que se propõe, cada objeto do ambiente virtual é composto por diversos elementos atômicos constituintes que unidos formam sua representação tridimensional. Dependendo do sistema, esses elementos atômicos, constituintes da forma dos objetos, podem ser *voxels* (*pixels* tridimensionais), triângulos ou polígonos. São esses os elementos considerados

no tratamento de oclusão nos algoritmos empregados na fase de geração da cena. Ao contrário do que ocorre na geração da cena, a estratégia a ser adotada no tratamento de oclusão entre objetos armazenados em SGBD deve considerá-los como objetos únicos e indivisíveis, tal como são armazenados no SGBD, a fim de usufruir os benefícios provenientes do uso das estruturas de indexação;

- **armazenamento dos resultados provenientes da execução do algoritmo de oclusão:** uma vez que os resultados provenientes da execução, em pré-processamento, do algoritmo de oclusão serão armazenados para serem utilizados durante a execução do SIG-3D, eles devem, de alguma maneira, fazer uso das estruturas de indexação e de outras facilidades proporcionadas pelo SGBD, a fim de agilizar o fornecimento de dados para a camada de visualização.

Com isso em mente, partiu-se para o desenvolvimento do algoritmo cuja estratégia aqui proposta considera como premissa a criação em pré-processamento (*offline*) de estruturas que privilegiem o tipo de acesso vislumbrado pelas aplicações alvo.

Durante o desenvolvimento do algoritmo de oclusão, duas hipóteses foram consideradas. Na primeira, tentou-se utilizar o conceito de fusão de oclusores, conforme apresentado por Schaufler et al (2000). Durante a implementação desta hipótese, observou-se que ela não tratava corretamente a oclusão de objetos com alturas diferentes, ocasionando falhas na determinação da oclusão. O motivo dessas falhas foi a diferença entre a abordagem feita no referido artigo da adotada neste trabalho: enquanto no artigo a fusão de oclusores é feita a nível de *voxels*, aqui tentou-se fazer a nível de objetos, o que não se mostrou possível, merecendo no futuro, talvez, um estudo mais específico. A tentativa baseou-se numa solução de oclusão bidimensional (plano XY), posteriormente estendida para considerar as cotas dos objetos (plano Z), numa abordagem conhecida como 2,5D, onde para cada ponto do plano bidimensional corresponde uma altura (ou cota), conforme ilustrado na figura 6.6. Sendo assim, partiu-se para a segunda hipótese, onde adotou-se uma solução totalmente tridimensional.

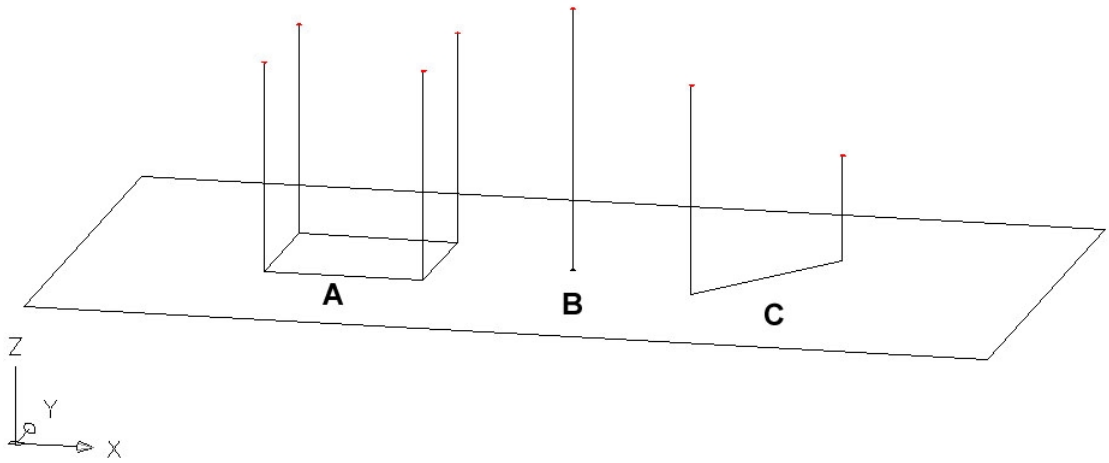

Figura 6.6 – Ilustração representativa de uma solução em 2,5D em perspectiva. Em preto: objetos em 2,5D (projeção dos objetos tridimensionais no plano XY). Em vermelho: valores das cotas para cada ponto. Com isso, observa-se que o objeto A é um cubo, o B, um ponto acima do plano XY e o C, uma reta oblíqua em relação ao plano XY.

Analisando-se novamente a figura 6.5, pretende-se modificá-la de modo a encontrar uma região dentro da qual se garanta que não é possível observar o objeto B. Tal região pode ser gerada, traçando-se retas ligando as extremidades do oclusor e o do objeto ocluso, conforme ilustrado na figura 6.7.

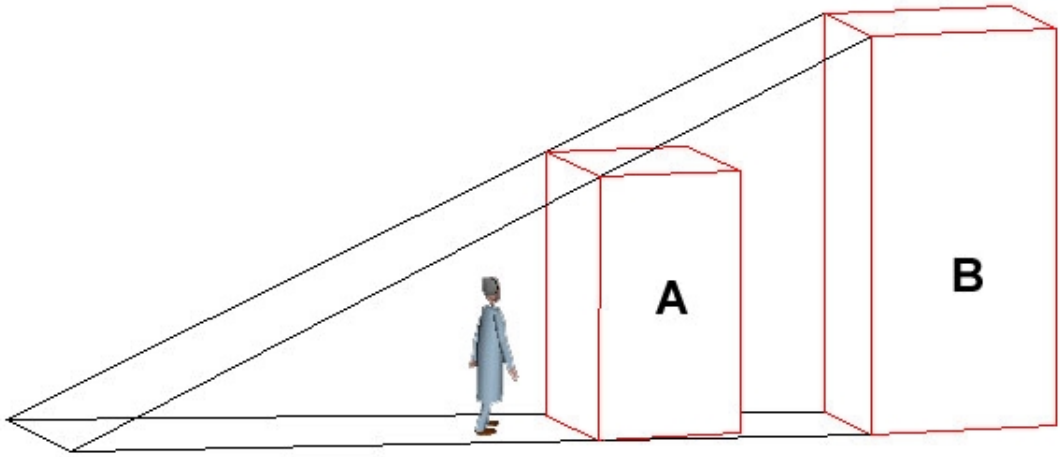

Figura 6.7 – Ilustração de uma região dentro da qual não seja possível ver-se o objeto B.

A região, dentro da qual o objeto B se encontra ocluso por A em relação ao observador, será chamada de Volume de Oclusão de B em relação a A. Também pode-se entender esse volume como sendo a região de sombra gerada por A para uma linha de luz que existisse no topo do bloco B. Para se criar uma regra clara de como gerar esse volume, adotou-se a nomenclatura apresentada na figura 6.8.

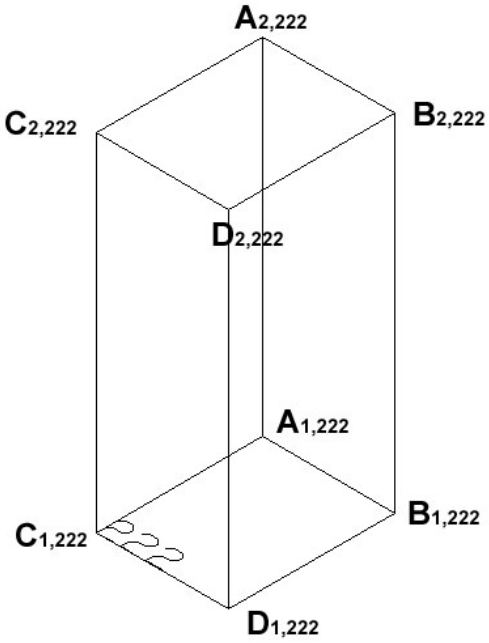

Figura 6.8 – Nomenclatura usada para os vértices dos blocos na geração do volume de oclusão. No exemplo, um bloco de número 222.

Tomando-se por base a figura 6.7, nota-se que o volume de oclusão é obtido a partir do prolongamento das retas que ligam vértices homólogos do oclusor e do objeto

ocluso. Deve-se escolher vértices de modo que a reta não cruze o interior de cada bloco. Considerando-se as figuras 6.7 e 6.8, verificamos que foram escolhidos os vértices C e D de cada bloco. Note-se que caso a escolha recaísse sobre os vértices A e B, obter-se-iam retas que cortariam o interior dos blocos, conforme a figura 6.9.

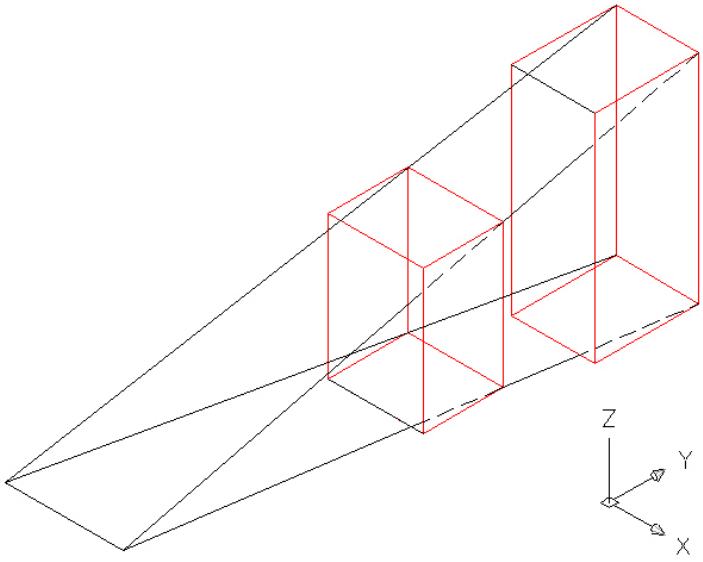

Figura 6.9 – Escolha incorreta dos vértices para geração do volume de oclusão. As linhas tracejadas representam o trecho da reta secante ao objeto.

Sendo assim, considerando-se a escolha dos vértices corretos, obtém-se o volume de oclusão, ilustrado em verde na figura 6.10, ou seja, se o observador se encontrar em qualquer posição dentro desse volume, não terá como ver diretamente o bloco 175.

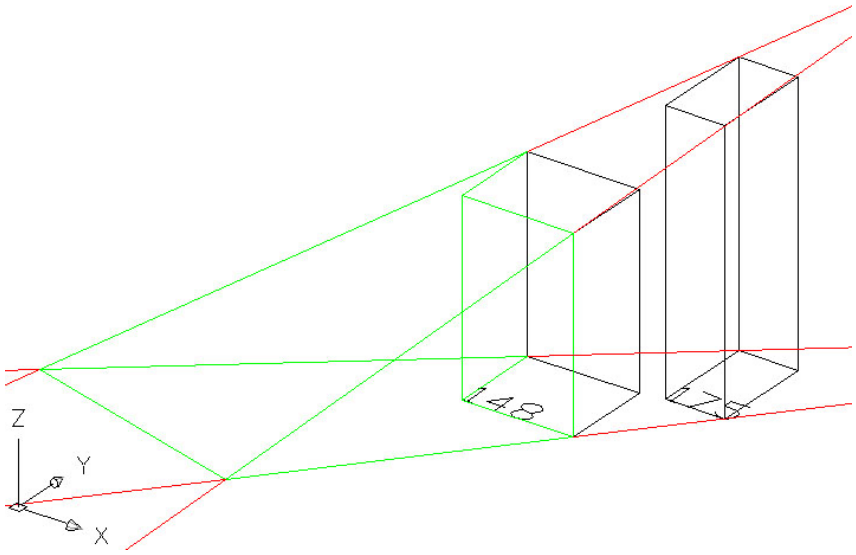

Figura 6.10 – Volume de oclusão gerado por retas que ligam pontos homólogos do oclusor (bloco 148) e do objeto ocluso (bloco 175).

É óbvio que em um ambiente denso de objetos, como numa simulação de ambientes urbanos, os volumes de oclusão relativos aos diversos blocos existentes poderão se sobrepor. Como isso, considere-se um observador totalmente inserido em dois volumes de oclusão ao mesmo tempo: como sua permanência em cada volume

acarreta não ser possível ver o objeto ocluso correspondente, nesta situação, o observador não poderia ver dois objetos (cada um correspondendo a um volume de oclusão).

Na figura 6.11, ilustrou-se um ambiente denso de objetos, onde ocorrem sobreposições de volumes de oclusão. Os volumes de oclusão (em verde na figura 6.10) tiveram sua forma simplificada para blocos em azul na figura 6.11 e foram colocados ao longo de uma rua, supondo-se que o observador só possa transitar por ela. Observe que um grande número de volumes de oclusão se sobrepõem.

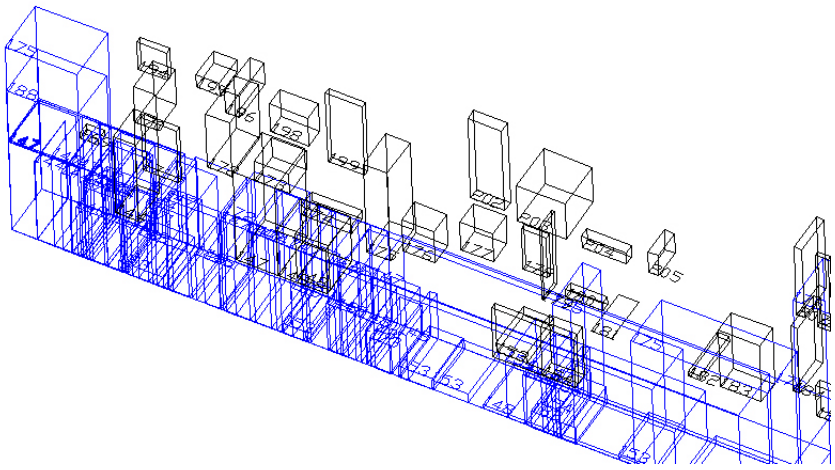

Figura 6.11 – Sobreposição de volumes de oclusão relativos à combinação de diferentes objetos existentes na figura. Em preto os objetos que geram a oclusão e em azul os volumes de oclusão, cuja forma foi aqui simplificada para blocos.

Observando-se a figura 6.11, percebe-se que um pequeno número de objetos gera um grande número de volumes de oclusão, uma vez que cada objeto deve testar sua oclusão com todos os objetos que estejam entre ele e a rua (neste caso). Durante a simulação, dependendo da estratégia adotada, tal quantidade de volumes de oclusão pode significar um alto custo de processamento no momento de se determinar quais objetos se encontram oclusos para o observador. É neste aspecto que o acesso a dados utilizando-se uma estrutura de indexação revela sua eficiência, conforme já abordado, sendo um dos fatores determinantes na definição da estratégia aqui adotada. Durante a execução do SIG-3D, de posse da posição do observador, o SGBD é capaz de responder rapidamente a uma consulta da camada de aplicação sobre quais volumes contêm sua coordenada, fazendo uso de uma estrutura de indexação gerada sobre o campo que contém a representação dos volumes de oclusão.

6.2.5. Adaptações da técnica de oclusão ao uso de estruturas de indexação espacial:

Conforme abordado na seção anterior, a definição de quais objetos não são visíveis ao observador é feita a partir do uso de um índice sobre o campo que contém a representação tridimensional dos volumes de oclusão: a camada de aplicação passa as coordenadas do observador para a camada de persistência; de posse dessas coordenadas, através de uma estrutura de indexação, o SGBD identifica todos os volumes de oclusão

que as contêm e através destes, quais objetos se encontram oclusos para aquele observador. Desta maneira, percebe-se a importância da adequação entre os MBB e os volumes de oclusão, pois se não observada, pode induzir a erros no tratamento de oclusão.

Observando-se a figura 6.10 e as considerações já feitas de como as estruturas de indexação apresentadas costumam particionar o espaço em MBB, conclui-se que a simples indexação do volume de oclusão, tal como ele se apresenta na referida figura, poderia causar distorções que acabariam por prejudicar a estratégia conservadora adotada para o tratamento de oclusão. O problema de adaptação do volume de oclusão à forma de particionamento do espaço pela estrutura de indexação do SGBD é ilustrado na figura 6.12 abaixo, considerando-se como área de circulação dos observadores, uma rua existente à frente de dois blocos.

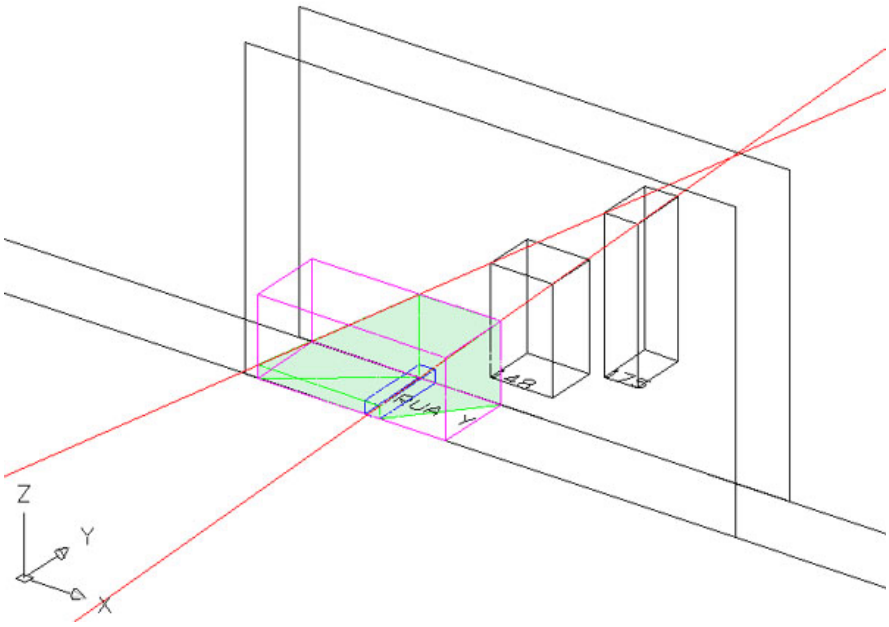

Figura 6.12 – Em verde está o volume da oclusão do bloco 148 sobre o 175, limitado pelas linhas vermelhas, que ligam pontos homólogos dos dois blocos e pelos planos perpendiculares à rua, gerados pelos meios-fios . Em rosa, o MBB que inclui o volume de oclusão. Em azul, o MBB adotado para preservar o tratamento conservador da oclusão.

Na figura 6.12, considera-se a região em verde como sendo o volume da oclusão do bloco 148 sobre o 175. Ao gerar-se um índice sobre esse volume, obtém-se a partir de seus pontos extremos o menor bloco que o envolve (MBB), representado na figura na cor rosa. Observa-se que este bloco abrange tanto o volume de oclusão, como regiões fora deste. Sendo assim, seria necessário um tratamento de erro após a localização do volume de oclusão pelo índice, a fim de verificar se o observador se encontra dentro do volume de oclusão, ou entre este e seu MBB, o que acrescentaria mais um teste ao processamento. Lembrando-se que o teste de oclusão aplicado nesta etapa do funcionamento de um SIG-3D (recuperação de objetos do banco de dados que serão enviados à camada de aplicação) visa apenas reduzir o custo computacional na recuperação de objetos desnecessários à geração da cena, um aumento no número de testes poderia tornar este tratamento ineficiente. Considerando-se isso e tendo em vista o emprego de uma estratégia conservadora no tratamento de oclusão, se optou por adaptar a forma do volume de oclusão aos MBB criados pela estrutura de indexação.

Desta maneira, o volume de oclusão adotado passou a ser dado pelo maior MBB interno ao volume original e limitado pelos meios-fios da rua, que na figura 6.12 está representado pelo bloco em azul. Assim se manteve a estratégia conservadora do tratamento de oclusão ao nível do SGBD e aproveitou-se o acesso eficiente promovido pela estrutura de indexação, causando pouca interferência no número de objetos considerados visíveis a serem enviados à camada de aplicação.

Ao longo do presente trabalho, admite-se que em um ambiente urbano, a movimentação dos observadores fica limitada aos espaços existentes entre os objetos constituintes da cidade. A adoção desta estratégia não tira a generalidade desta solução, uma vez que pressupõe-se o tratamento de oclusão para ambientes externos (*outdoor*). No caso de haver necessidade de visualização do interior desses objetos (prédios e casas), deve-se adotar um tratamento de oclusão para ambientes internos (*indoor*), que difere bastante do aqui abordado, além de fugir aos objetivos deste trabalho.

6.3. Estratégia adotada para o tratamento de oclusão:

O teste de oclusão entre os diversos objetos constituintes do ambiente virtual é feito em um pré-processamento. Preferiu-se esta estratégia em relação ao processamento da oclusão em tempo de execução do SIG-3D pelos seguintes motivos:

- o volume de dados para a criação de um ambiente virtual, notadamente um ambiente virtual urbano, é muito grande, o que inviabilizaria uma execução do algoritmo em tempo real;

- grandes oclusores são representados, em um ambiente urbano, na maioria das vezes, por objetos estáticos, tais como prédios, muros, casas e outras construções. Geralmente, objetos móveis ocupam uma pequena área na cena final gerada pelo ambiente de visualização e nela permanecem por pouco tempo, sendo logo necessário apresentar os objetos que se encontravam por trás do mesmo. Desta maneira, o custo computacional para se calcular a oclusão gerada por um objeto móvel dificilmente compensa os recursos empregados para a recuperação de objetos oclusos e possivelmente desnecessários à geração da cena;

- para recuperar apenas os objetos visíveis, pode-se usar a estrutura de índice do SGBD, aproveitando suas funcionalidades e dispensando novas implementações;

- como se trata de um SIG-3D militar, no caso de destruição de objetos, basta criar-se uma rotina que além de apagar ou modificar a aparência do objeto atingido, também encontre os volumes de oclusão gerados por ele e os remova. Essa busca se torna fácil pois a tabela que contém a representação do objeto se relaciona com a tabela contendo os volumes de oclusão através de chaves primárias e estrangeiras.

Deste modo, uma vez carregadas todas as feições constituintes do ambiente virtual no SGBD, executa-se o teste de oclusão. Os volumes de oclusão gerados são armazenados no SGBD, juntamente com seus respectivos números identificadores do oclusor e do objeto ocluso. Assim, o SIG-3D encontra-se pronto para iniciar uma simulação.

Ao iniciar-se o sistema, deve-se fornecer as coordenadas iniciais do observador, sua acuidade visual, amplitude do campo visual e direção de visada. Usando esses parâmetros, a camada de aplicação calcula as coordenadas dos pontos que delimitam o campo de visão. Esses dados, coordenadas iniciais do observador e limites do campo de visão, são enviados ao SGBD em forma de consulta. De posse das coordenadas do observador, o SGBD, usando um índice sobre o campo de representação espacial dos volumes de oclusão, recupera todos os que contenham a coordenada do observador. Desta maneira, obtém-se uma lista de objetos oclusos para o observador naquele momento. O SGBD recupera então todos os objetos que estejam total ou parcialmente contidos no campo de visão, usando as coordenadas dos pontos limites que lhe foram remetidas através de uma consulta e um índice sobre o campo que contém as envoltórias desses objetos. O SGBD retorna então para a camada de aplicação, como resultado da consulta, a representação em VRML de cada um dos objetos pertencentes ao conjunto daqueles contidos total ou parcialmente no campo de visão e que não estejam oclusos. Finalmente a camada de visualização, com base nos objetos recuperados, gera ("renderiza") a cena e a apresenta para o usuário.

6.4. Pre-fetch:

Para que se possa garantir uma taxa de reposição de quadros suficiente para não se prejudicar a interatividade com o sistema durante a simulação, deve se ter um fluxo de dados suficiente entre o SGBD e a camada de visualização. Numa simulação, podem ocorrer movimentos bruscos e de grande amplitude do observador e que necessitem de uma grande quantidade de novos objetos para a geração da cena que não compunham a anterior. Para que mesmo nessas situações possa se garantir a fluidez necessária da imagem, é interessante recuperar-se um conjunto maior de objetos (pre-fetch) do que apenas os necessários para se gerar a cena corrente.

No caso do observador parado, uma solução possível é adotar-se como campo de visão uma semi-esfera (apenas a parte acima do plano XY) de raio igual à acuidade visual, que pode ser aproximada para sua projeção sobre o plano XY. Sendo assim, ao determinar-se quais serão os objetos a serem recuperados do SGBD para a camada de aplicação, ao invés de fazer-se a interseção entre o conjunto de objetos não-oclusos com os contidos no cone de visão, fazer-se-á a interseção daqueles com os contidos em um círculo de raio igual à acuidade visual. Este será o procedimento adotado no presente trabalho.

No caso do observador estar se movimentando, deve-se considerar sua velocidade, seu sentido de deslocamento e o tempo médio observado pela camada de aplicação entre a consulta aos dados e a disponibilidade da cena já gerada para se recuperar antecipadamente dados correspondentes à próxima cena. Esses parâmetros devem ser avaliados em função da capacidade de processamento gráfico da plataforma utilizada. Este tratamento não foi abordado no presente trabalho, uma vez que depende de parâmetros influenciados pela camada de visualização.

7. Implementação:

Nesta seção descreve-se a implementação do protótipo desenvolvido para validação das idéias já apresentadas. Inicialmente, apresenta-se o ambiente de

desenvolvimento. Em seguida, discute-se algumas simplificações adotadas, passando-se finalmente pelas etapas de: criação do ambiente de prototipação; desenvolvimento do algoritmo de oclusão e preparação do ambiente físico no SGBD.

7.1. Ambiente de desenvolvimento:

No ambiente de desenvolvimento para este trabalho, procurou-se adotar apenas *softwares livres*, por ser uma solução que numa futura implantação de um SIG-3D vá redundar numa redução significativa de despesas. Exceção a essa regra foi a utilização do programa AutoCAD para visualização dos resultados. Com isso, para este teste foram usados o SGBD PostgreSQL-7.4, com seus módulos PostGIS-0.8.0 (que implementa o padrão OpenGIS sobre o PostgreSQL) e Geos-1.0.0 (Módulo de análise topológica), todos instalados em um computador compatível com PC (Athlon 2.0+, 512MB de RAM e disco rígido de 40 GB) com Sistema Operacional Linux, distribuição RedHat 9.0. Para manutenção do SGBD se usou o phpPGAdmin 2.1.4.12 e para implementação das rotinas que geravam o ambiente virtual, executavam as rotinas de oclusão e que convertiam os resultados em AutoLISP para visualização no AutoCAD, utilizou-se o Java 2 Plataform Standard Edition (SE) Software Development Kit (SDK) 1.4.1_03.

7.2. Simplificações adotadas:

No caso específico deste estudo, algumas suposições foram adotadas de modo a simplificar os cálculos realizados na implementação do algoritmo de oclusão, sem no entanto causar perda de generalidade na solução. As simplificações assumidas foram:

- *Restrição do movimento do observador às ruas da cidade virtual:* neste trabalho, foi considerado que o observador poder-se-ia movimentar apenas dentro de regiões denominadas ruas. Esta restrição não tira a generalidade da solução, uma vez que o mesmo tratamento que será apresentado adiante pode ser aplicado ao caso geral, implicando apenas no aumento da quantidade de cálculos envolvidos;

- *Posicionamento de ruas:* em um arquivo vetorial, uma curva é geralmente aproximada por vários segmentos de reta, sendo o comprimento destes dependente da escala de apresentação. Sendo assim, a adoção de uma rua reta não tira a generalidade da solução, uma vez que outros traçados de ruas podem ser consideradas como um conjunto de pequenas retas. Também a adoção de uma rua paralela a um eixo coordenado não tira a generalidade da solução, simplificando apenas os cálculos para a confirmação da eficácia do algoritmo. Sendo assim, foi adotada nesta implementação uma rua reta e paralela ao eixo das abscissas;

- *Posicionamento das envoltórias:* o alinhamento das envoltórias com os eixos coordenados foi adotado a fim de se simplificar os cálculos. Considera-se que não há perda de generalidade, uma vez que sempre é possível se obter tais envoltórias, mesmo que o objeto se encontre oblíquo aos eixos coordenados e a adoção de dois tipos diferentes de envoltórias permite manter a oclusão conservadora dos objetos em qualquer posição.

Considerando-se essas simplificações passou-se a preparação do ambiente, descrita na seção seguinte.

7.3. Preparação dos dados:

A fim de permitir a avaliação dos testes necessários durante a implementação do algoritmo, foi criada de forma aleatória uma cidade virtual. Para entender como essa cidade foi criada, algumas observações devem ser feitas:

- o modelo tridimensional de cada objeto constituinte do ambiente virtual é armazenado sob a forma de um arquivo VRML, em um campo CLOB no SGBD;

- para o teste de oclusão, é considerada apenas a envoltória de cada objeto;

- são criadas duas envoltórias para cada objeto, uma considerando-o oclusor, outra para quando este for testado como objeto possivelmente ocluso;

- ruas, estradas, linhas de transmissão, rios e elevações do terreno merecem uma modelagem especial. Geralmente sua representação é extensa e inviável de ser feita em apenas um arquivo VRML. Neste caso, uma maneira simplificada de se resolver o problema de representação seria particioná-los em um conjunto de vários arquivos em VRML, que juntos representariam toda a feição. O problema não foi abordado neste trabalho, sendo sugerido como trabalhos futuros.

Na próxima seção, descrever-se-a a geração da cidade virtual utilizada para testes do sistema.

7.4. Geração da Cidade Virtual:

Como já mencionado anteriormente, o teste de oclusão é feito sobre as envoltórias do objeto, considerando-se a condição deste, de possível ocluso ou oclusor. Sendo assim, optou-se por gerar uma cidade sem os arquivos VRML, utilizando-se apenas as envoltórias, a fim de testar-se a implementação do algoritmo de oclusão desenvolvido no presente trabalho. A geração da cidade, por meio de uma rotina, programada em Java, consistiu na criação de diversos blocos, espaçados de maneira também aleatória e distribuídos numa área quadrada de 1000x1000 unidades de comprimento (UC). Desta maneira, se cada UC for considerada equivalente a 1 metro (m), tem-se uma área de 1x1 km. O tamanho dos blocos que representam envoltórias também foi calculado de maneira aleatória, abrangendo um intervalo de valores em UC que variassem desde o tamanho de um hidrante (0,5 UC) até o tamanho de um edifício de aproximadamente 33 andares (100 UC). No meio dessa área, entre as ordenadas de valores y = 475 UC e y = 525 UC, foi deixado um espaço vazio, a fim de representar uma rua de 50 UC de largura. O movimento dos observadores, conforme abordado anteriormente, será restrito aos limites dessa rua. A fim de ilustrar-se a ordem de grandeza do volume de dados gerado nesta fase, a cidade foi aleatoriamente construída com 326 blocos, representando envoltórias (o que corresponde à 326 tuplas).

Como o desenvolvimento da camada de visualização não faz parte do escopo do presente trabalho, evitou-se desenvolver um aplicativo para a visualização dos

resultados da aplicação do algoritmo de oclusão. A visualização foi toda feita utilizando-se o programa de CAD (Computer Aided Design) AutoCAD, da Autodesk. Para isso foi desenvolvida uma aplicação em Java, que transformava a saída do resultado das consultas ao SGBD em um arquivo AutoLISP, linguagem de programação do AutoCAD. Desta maneira, obteve-se a visualização da cidade virtual, conforme apresentado na figura 7.1.

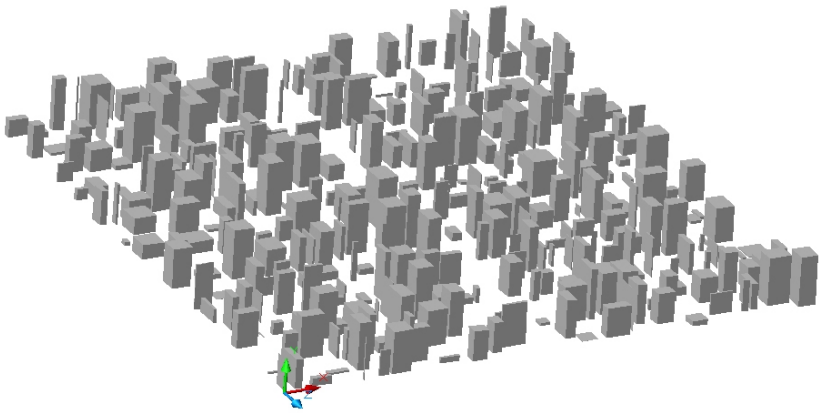

Figura 7.1 - Vista geral da Cidade Virtual.

7.5. O algoritmo de oclusão:

Na implementação feita neste trabalho, considerou-se, conforme exposto no início desta seção, uma região com 1000 x 1000 UC e cortada por uma rua de 50 UC paralela ao eixo das abscissas e eqüidistante dos limites da cidade.

O algoritmo de oclusão percorre a tabela com as envoltórias, testando para cada uma, sua oclusão com todas as demais que se encontrem entre ela e a rua. Se o teste fosse feito com todas as outras envoltórias, teríamos uma complexidade de $O(n^2)$. Sendo assim, pode-se dizer que sua complexidade é bem menor que esse valor.

7.5.1. Funcionamento do Algoritmo de Oclusão:

Para proceder-se ao teste de oclusão, é primeiramente analisada a posição relativa entre a envoltória cuja oclusão está sendo testada, aqui chamada de *test-box* e seu possível oclusor. Esse teste é feito comparando-se as coordenadas dos pontos homólogos das duas envoltórias, compreendendo um dos quatro casos abaixo (considerando-se o sentido dos eixos coordenados):

- possível oclusor totalmente à esquerda do *test-box*:

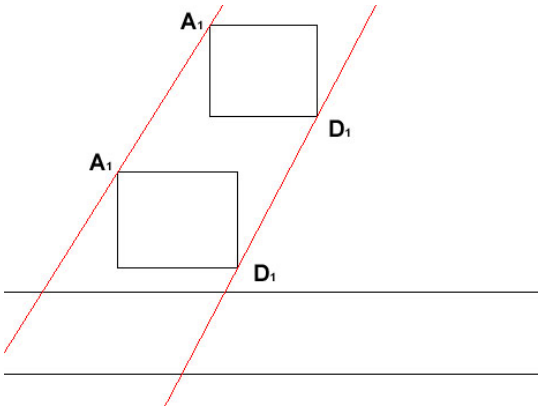

Figura 7.2 – Possível oclusor totalmente à esquerda do *test-box*. (Vista de topo)

- possível oclusor com uma extremidade à esquerda e outra à direita do *test-box*:

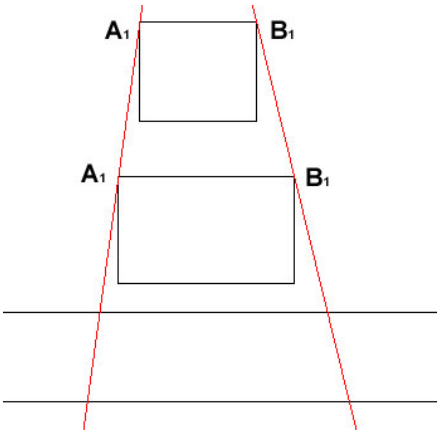

Figura 7.3 – Possível oclusor com uma extremidade à esquerda e outra à direita do *test-box.* (Vista de topo)

- possível oclusor entre as extremidades do *test-box*:

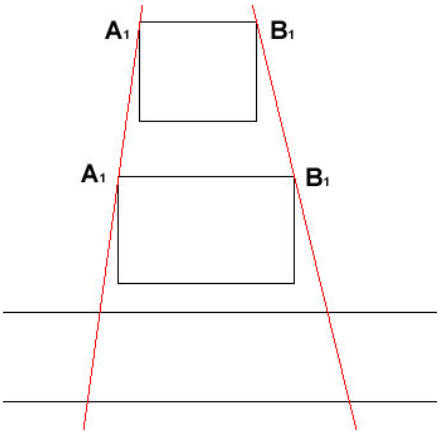

Figura 7.4 – Possível oclusor entre as extremidades do *test-box.*(Vista de topo)

- possível oclusor totalmente à direita do *test-box*:

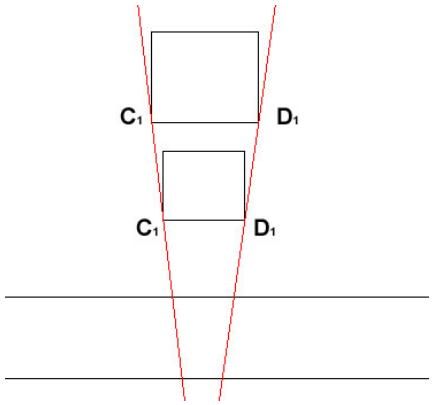

Figura 7.5 – Possível oclusor totalmente à direita do *test-box*.(Vista de topo)

Como a finalidade desta implementação é comprovar o funcionamento da estratégia de oclusão desenvolvida, o algoritmo de oclusão foi implementado contemplando apenas metade acima da rua (norte) da cidade virtual, pois para a outra metade, a implementação dos testes é análoga, mudando apenas os testes de ordenadas para saber se o oclusor está realmente entre o *test-box* e a rua.

Após o teste de posição relativa entre as envoltórias que terão sua oclusão verificada, calcula-se a equação das quatro retas que ligam os pontos homólogos, definindo o volume de oclusão, conforme mostrado na figura 7.6.

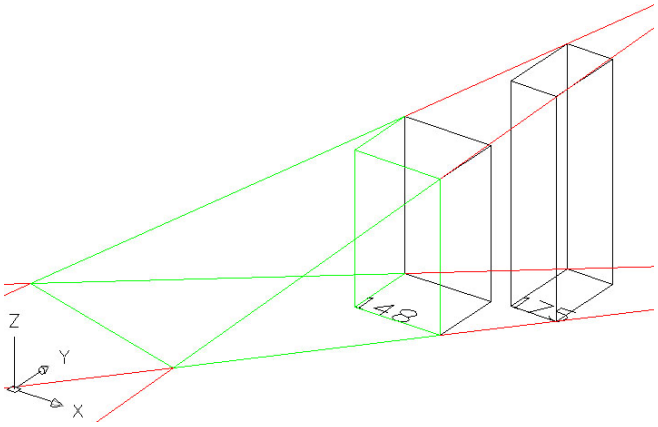

Figura 7.6 – Ilustração da determinação da equação de cada uma das quatro retas que definem o volume de oclusão. (Vista oblíqua)

Determinadas as equações de retas, determina-se o valor da coordenada onde cada uma das retas cruza os planos perpendiculares à rua sobre os meios-fios (figura 7.7), determinando-se os pontos ilustrados na figura 7.8, que representam o maior quadrado contido entre as retas que ligam pontos homólogos das envoltórias e que contém parte dos meios-fios. Faz-se um teste para verificar se algum dos pontos está fora da área da cidade. Se isto ocorrer com os quatro, não é gerado volume de oclusão, uma vez que este estaria fora da cidade e portanto fora da área de circulação do observador. Se ocorrer com apenas dois, estes dois são substituídos pelas coordenadas dos pontos mais próximos, que estejam no meio-fio no limite da cidade (mantendo a cota determinada pelo cálculo inicial).

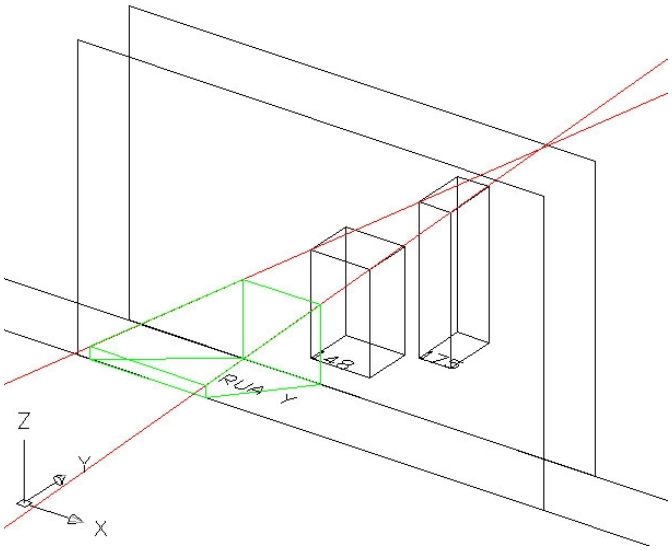

Figura 7.7 – Ilustração dos planos perpendiculares a rua Y sobre os meios-fios e do volume de oclusão gerado por eles e pelas retas que ligam pontos homólogos. (Vista Oblíqua)

Para cada um desses quatro pontos, determina-se qual possui a menor cota. Forma-se então um paralelepípedo formado pelos quatro pontos da base e com altura igual à menor cota entre os pontos 11, 12, 21 e 22. Este paralelepípedo é o volume de oclusão adaptado ao MBB, conforme explicado anteriormente e ilustrado em azul na figura 6.12. Armazena-se no SGBD, na tabela de Volumes de Oclusão (Voloclu) o id do bloco oclusor, o id do bloco ocluso e as dimensões do volume de oclusão, usando-se o tipo de dado *BOX3D*. Repete-se o processo para cada possível oclusor que esteja entre o *test-box* e a rua e depois para cada *test-box*, gerando-se todos os volumes de oclusão.

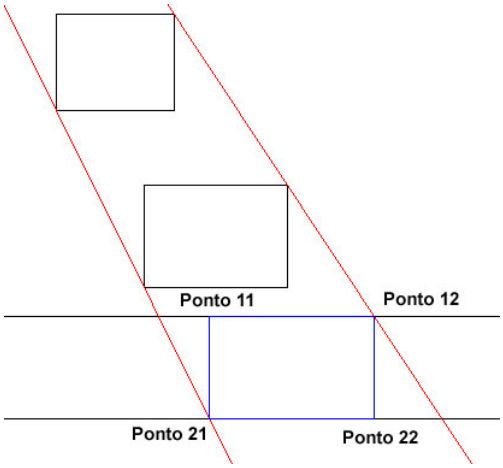

Figura 7.8 – Pontos determinados na interseção das retas que ligam pontos homólogos das envoltórias com os planos dos meios-fios.(Vista de topo)

O próximo passo é indexar todos os volumes de oclusão e todas as envoltórias constituintes da cidade. O índice sobre o campo das envoltórias auxiliará consultas para determinar quais são as feições no campo visual do observador, enquanto que o índice sobre o campo dos volumes de oclusão, determinará quais são os volumes de oclusão que estão no local onde se encontra o observador, determinando quais serão portanto as

feições que estão oclusas para ele. Do cruzamento dessas informações, recuperam-se apenas as feições potencialmente visíveis na geração da cena.

7.5.2. Casos em que o algoritmo não detecta a oclusão:

O volume de oclusão é gerado a partir do paralelepípedo máximo que contenha os dois meios-fios e que esteja contido no volume limitado pelas quatro retas que ligam pontos homólogos e pelos planos perpendiculares sobre os meios-fios da rua. Numa situação em que o *test-box* e o oclusor se encontrem muito distantes, pode ocorrer de não ser possível desenhar um quadrado entre os pontos de interseção das retas que ligam pontos homólogos com os limites da rua, conforme ilustrado na figura 7.9. Este problema será referenciado ao longo deste trabalho como "oclusão estreita" e também pode ocorrer em alguns casos nos quais haja uma combinação dos fatores tamanhos das duas envoltórias próximos e posição relativa entre eles. Nas duas situações citadas, o algoritmo não detecta a oclusão da maneira como está implementado, ou seja, ele não gera um volume de oclusão para uma envoltória que não é visível.

Como este problema é conseqüência da exigência de se encontrar um quadrado que contenha os dois meios-fios da rua, uma maneira de resolvê-lo é dividindo-se a rua ao meio, permitindo a criação de dois volumes de oclusão menores, conforme ilustrado na figura 7.10. Assim, apesar de ainda haver chance de um objeto ocluso ser recuperado como possivelmente visível, a sua possibilidade já era prevista na manutenção de uma estratégia conservadora para o tratamento de oclusão. Pode-se dividir a rua em mais linhas paralelas, de modo a diminuir o tamanho dos volumes de oclusão, o que significa aumentar a área coberta. Quanto mais se dividir a rua por linhas paralelas, menor será a granularidade dos volumes de oclusão e portanto, maior será sua aproximação ao volume real, que é aquele formado pelas quatro retas que ligam os pontos homólogos das envoltórias (desenhado em verde na figura 6.12).

Para adotar a solução aqui apresentada, são poucas as modificações necessárias no algoritmo. Além da rotina de subdivisão da rua, exige apenas que o algoritmo utilize para limite as novas ordenadas obtidas, ao invés das ordenadas da rua original (y = 475 UC e y = 525 UC, no exemplo ilustrado no começo desta seção.

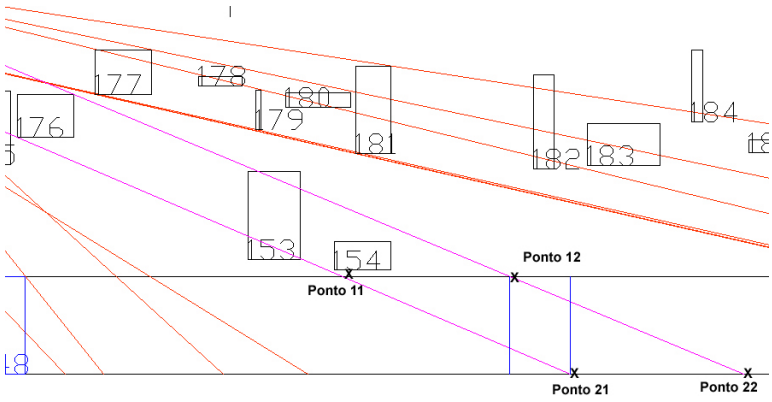

Figura 7.9 – Caso em que falha o algoritmo. Em rosa, as linhas que ligam pontos homólogos da envoltória de número 176 e outra distante, que não aparece na figura.(Vista de topo)

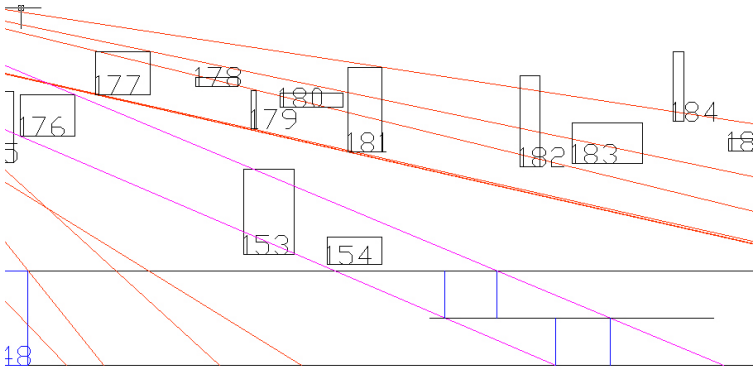

Figura 7.10 – Solução para o problema de oclusão estreita com divisão da rua.(Vista de topo)

Na próxima seção, será mostrada uma consulta que atesta o funcionamento do algoritmo desenvolvido.

8. Avaliação do Algoritmo de oclusão em SGBD:

Inicialmente foi previsto o desenvolvimento de um aplicativo em modo texto que simulasse as consultas feitas pela camada de aplicação e comparasse os resultados, no que diz respeito ao tempo de resposta e número de objetos retornados, com e sem a utilização das otimizações introduzidas no sistema. Ao contrário do planejado, não foi possível efetuar-se tais testes. Desta maneira, apresentam-se consultas em linha de comando feitas que ilustram da mesma maneira os resultados obtidos.

Para averiguar-se o funcionamento do algoritmo de oclusão, considerou-se um observador cujos olhos estavam a 1,75 UC do solo e que se encontrava nas coordenadas x = 5 UC e y = 500 UC, ou seja, no começo da rua, em seu meio.

Fizeram-se duas consultas: a primeira recuperando tudo que estivesse em seu campo de visão, de acordo com as observações sobre *prefetch* da seção 6.4 (adotando como campo de visão um círculo com centro no observador e raio igual à sua acuidade visual, aqui considerada de 500 UC – como a função *distance* do OpenGIS não estava implementada para o tipo de dado *BOX3D*, aproximou-se o círculo pelo seu MBB, de modo que pudesse ser usado índice para agilizar consulta); e a segunda, verificando quais os volumes se encontram oclusos. Disto, gerou-se um arquivo em AutoLISP, contendo o resultado da subtração do conjunto de todos os objetos contidos no campo de visão do observador, do conjunto dos objetos oclusos a partir do ponto considerado. O arquivo em AutoLISP foi carregado no AutoCAD e gerou uma apresentação que confirmava o funcionamento do algoritmo para este caso. O banco de dados foi nomeado “testegeos” e as tabelas e atributos, conforme modelo lógico, já apresentado.

```
testegeos=# select envmax
            from feicao
            where envmax && 'BOX3D(0 0 0, 505 1000 0)'::box3d;
```

Figura 8.1 – Consulta solicitando os objetos contidos no campo de visão (visíveis) de um observador localizado no ponto de coordenadas x = 5 UC e y = 500 UC (no meio da via, no começo da rua) e com acuidade visual de 500 UC.

```
testegeos=# select id_ocluso, id_oclusor
            from voloclu
            where intersects(represent, 'POINT(5 500 1.75),-1');
```

Figura 8.2 – Consulta solicitando os objetos oclusos para um observador localizado no ponto de coordenadas x = 5 UC e y = 500 UC (no meio da via, no começo da rua) e cujos olhos se encontram a 1,75 UC do solo.

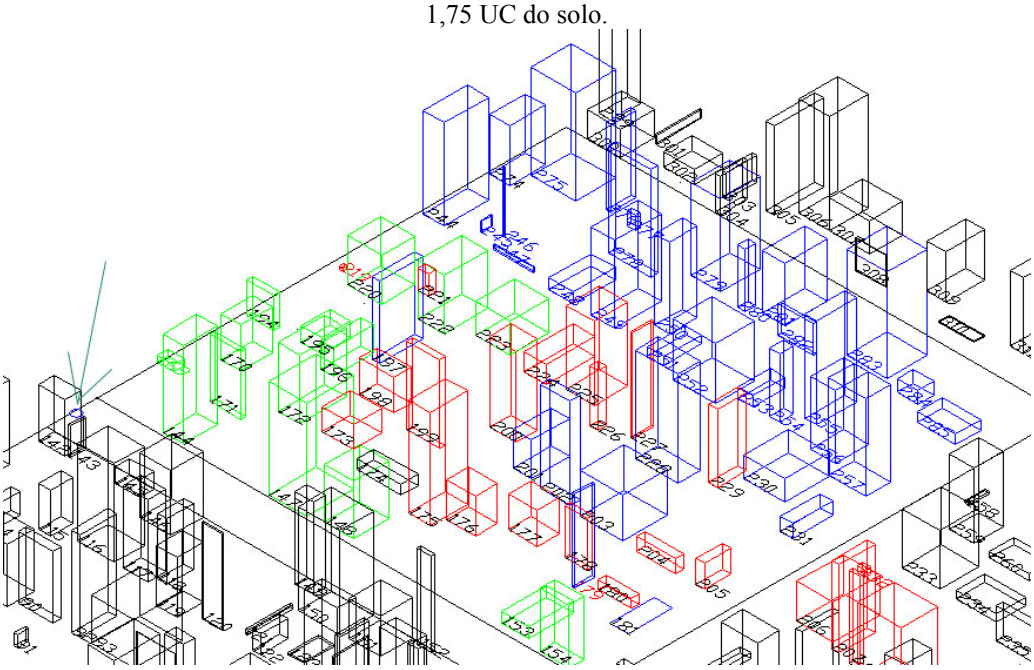

Figura 8.3 – Resultado das consultas solicitando objetos visíveis e oclusos para um observador localizado no ponto de coordenadas x = 5 UC e y = 500 UC (no meio da via, no começo da rua) e cujos olhos se encontram a 1,75 UC do solo. Em verde claro os objetos visíveis, em vermelho os oclusos, em azul os que deveriam ser considerados oclusos, mas não foram tratados como tal. A seta verde escuro indica o observador, representado por um círculo azul. (Vista Oblíqua)

O resultado das duas consultas foi convertido em AutoLISP, gerando a saída em AutoCAD ilustrada nas figuras 8.3 e 8.4. Observar que o algoritmo de oclusão só foi implementado para a porção norte da cidade (acima da rua).

Nas figuras que ilustram o resultado da consulta, estão em verde claro as envoltórias das feições visíveis para o observador. Em vermelho as feições oclusas e em azul as que foram erradamente consideradas como possivelmente visíveis. Percebe-se falha no algoritmo, cujos motivos foram descobertos após análise dos dados: existência de "oclusão estreita" e de volumes de oclusão que não estão totalmente contidos na cidade. O segundo caso ocorre no momento da determinação dos pontos que delimitarão o volume de oclusão. Para isso, faz-se um teste a fim de verificar se algum dos pontos está fora da área da cidade. Desse teste duas hipóteses podem ocorrer:

- *Volume de oclusão fora da área da cidade:* se os quatro pontos estiverem fora da área da cidade, não é gerado volume de oclusão, pois este ficaria fora da área de circulação do observador;

- *Apenas parte do volume de oclusão fora da área da cidade:* se ocorrer menos de quatro pontos fora da cidade, tem-se um volume de oclusão que ocupa ao menos uma pequena parte da área de circulação do observador. Se houver só um ponto, não há como traçar-se uma linha perpendicular aos meios-fios, pois a solução não se mostraria

conservadora. Para essa linha, é necessário ter-se no mínimo dois pontos. Os dois que caem fora da área de circulação do observador são substituídos pelas coordenadas dos pontos mais próximos, que estejam no meio-fio no limite da cidade (mantendo a cota determinada pelo cálculo inicial).

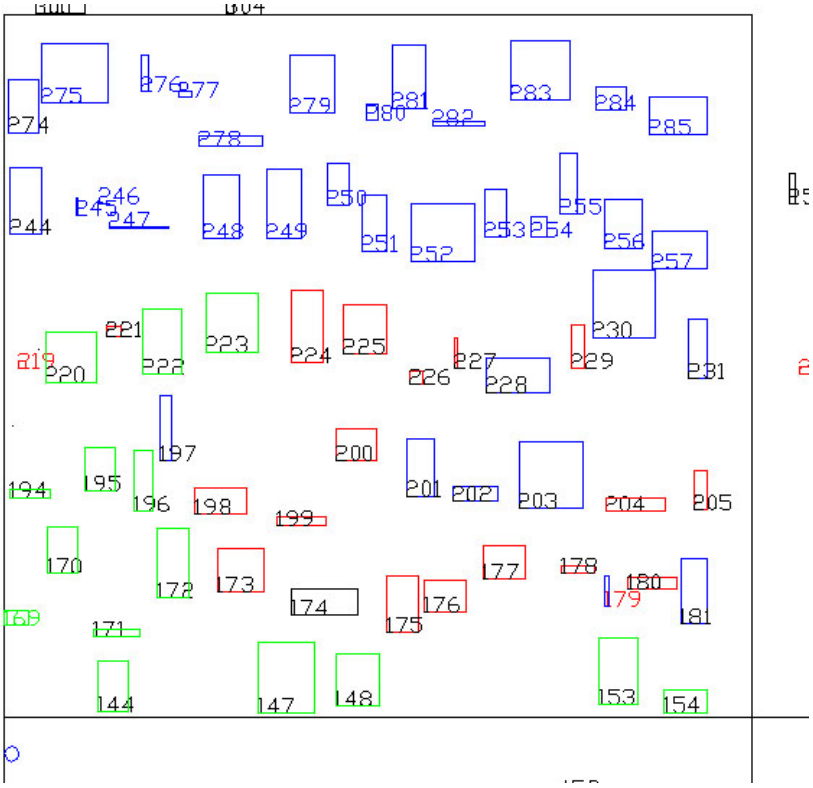

Figura 8.4 – Resultado das consultas solicitando objetos visíveis e oclusos para um observador localizado no ponto de coordenadas x = 5 UC e y = 500 UC (no meio da via, no começo da rua) e cujos olhos se encontram a 1,75 UC do solo. Em verde claro os objetos visíveis, em vermelho os oclusos, em azul os que deveriam ser considerados oclusos, mas não foram tratados como tal. A seta verde escuro indica o observador, representado por um círculo azul. (Vista de topo)

Mesmo considerando-se a ineficiência do algoritmo da maneira como está implementado, percebe-se que uma boa quantidade de objetos foi considerada oclusa, deixando portanto, de ser recuperada desnecessariamente. Além disso, a solução do problema da oclusão estreita, resolverá a maioria dos casos. Mesmo com essas soluções, alguns objetos ainda serão recuperados desnecessariamente, o que não invalida o método, uma vez que adotou-se uma estratégia conservadora, onde se opta por um equilíbrio (*tradeoff*) entre precisão do método e custo de processamento.

Para avaliar-se a quantidade de objetos manipulados durante a execução deste algoritmo, de um total de 164 objetos testados na região acima da rua (norte), foram gerados 322 volumes de oclusão. A aplicação da correção do algoritmo tende a aumentar consideravelmente o número de volumes de oclusão gerados.

9. Conclusão:

O tratamento de oclusão no nível de memória secundária requer o desenvolvimento de técnicas especiais que permitam acesso rápido, e se possível direto, aos objetos visíveis a partir de um dado ponto no espaço e acuidade visual do observador. É importante observar que esse modelo é completamente distinto dos utilizados atualmente no tratamento de oclusão realizado em memória principal, nos quais todos os objetos se encontram disponíveis para análise, prestando-se, por exemplo, a uma análise exaustiva de visibilidade. Neste particular, o presente trabalho inova não somente com a proposta de algoritmo para o tratamento de oclusão mas também com a análise do problema a partir de uma ótica distinta, abrindo oportunidade para o avanço no tratamento do problema a nível de sistemas de bancos de dados.

Neste sentido desenvolveu-se uma nova técnica de oclusão, conservadora e pré-processada, cujos resultados são armazenados no SGBD e utilizados durante a consulta feita pela aplicação de visualização para a geração do ambiente virtual.

Assim, pode-se afirmar que os principais resultados deste trabalho foram:

- sugerir um método de armazenamento de modelos de objetos tridimensionais utilizando-se VRML (Virtual Reality Modeling Language) em um campo do tipo CLOB (Character Large Object) de uma tabela no SGBD;

- desenvolver uma estratégia de tratamento de oclusão que otimiza o desempenho do SIG-3D, aproveitando-se diversos conceitos e facilidades disponíveis no SGBD;

10. Referências Bibliográficas:

AGARWAL, P.K., HAR-PELED, S., WANG, Y., **Occlusion Culling for Fast Walkthrough in Urban Areas**, in Eurographics, 2001.

CATMULL, E. **A Subdivision Algorithm for Computer Display of Curved Surfaces.** Ph.D. thesis, University of Utah, December 1974.

COHEN-OR, D., CHRYSANTHOU, Y., SILVA, C., DURAND, F. **A survey of visibility for walkthrough applications**. SIGGRAPH Course Notes # 30, 2001.

COHEN-OR, D., FIBICH, G., HALPERIN, D, ZADICARIO, E. **Conservative visibility and strong occlusion for viewspace partitioning of densely occluded scenes**. Computer Graphics Forum, 17(3):243–254, 1998.

ISO - INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. **ISO/IEC 14772-1 – Information technology – Computer graphics and image processing – The Virtual Reality Modeling Language – Part 1: Functional specification and UTF-8 encoding.** 1997.

OGC - OPEN GEOSPATIAL CONSORTIUM. **OpenGIS: Abstract Specification Topic 1: Feature Geometry.** Wayland, Massachusetts, EUA, 1998.

SCHAUFLER, G., DORSEY, J., DECORET, X., SILLION, F. **Conservative volumetric visibility with occluder fusion**. Proceedings of SIGGRAPH 2000, pages 229–238, July 2000.

SILVA, H. **Tratamento eficiente de visibilidade através de árvores de volumes envolventes**. Dissertação de Mestrado, PUC, 2002.

THE UNIVERSITY OF ARIZONA. **Campus VRML Model**. Disponível em: <http://ag.arizona.edu/agnet/icac/vrml>. Acesso em: 10 ago. 2004.

WONKA, P., WIMMER, M., SCHMALSTIEG, D. **Visibility preprocessing with occluder fusion for urban walkthroughs**. Rendering Techniques 2000: 11th Eurographics Workshop on Rendering, pages 71–82, June 2000.