AVALIAÇÃO DE ESTRATÉGIAS ALGORÍTMICAS NA VISUALIZAÇÃO DE CAMPOS TRI-DIMENSIONAIS

Gabriel Freire de Andrade
Manuel João Próspero

Universidade Nova de Lisboa
Departamento de Informática da FCT
Quinta da Torre
2825 Monte de Caparica

Sumário

Esta comunicação baseia-se num trabalho de criação de um ambiente para a visualização de dados científicos, concretamente a visualização em volume de campos tri-dimensionais, constituídos por medidas de grandezas contínuas, escalares vectoriais e multivariável, nos nós de uma grelha. Analisam-se diversas técnicas de visualização e compara-se a sua aplicabilidade aos fins em vista.

Descreve-se, em seguida, a implementação de dois algoritmos de visualização, analisando-se os problemas encontrados e as técnicas para os eliminar, com discussão dos resultados conseguidos.

1 - Introdução

A visualização de dados científicos é uma área que se tem revelado cada vez mais importante nos últimos anos. As técnicas de visualização têm tido um grande desenvolvimento, com a passagem de gráficos simples uni ou bi-dimensionais a técnicas mais sofisticadas, apoiadas nos desenvolvimentos drásticos ocorridos no *hardware* gráfico, como a visualização tri-dimensional em volume recorrendo a técnicas de visualização foto-realística. Designam-se por técnicas de visualização em volume as técnicas que permitem visualizar a totalidade de um volume numa única imagem.

Estas técnicas permitem uma percepção global de um conjunto de dados que é impossível obter de outra forma, permitindo uma melhor análise e interpretação dos dados científicos.

Apesar deste facto, a utilização da visualização de dados em geral e das técnicas mais avançadas em particular não está tão difundida como seria de desejar. A razão principal é a falta de *software* adequado. O relatório de McCormick, de 1987 [McCormick], veio chamar a atenção da comunidade científica para esta questão.

Programar propositadamente aplicações de visualização a partir de bibliotecas de subrotinas gráficas (GKS, PHIGS) é excessivamente moroso e obriga o utilizador a preocupar-se com detalhes de programação, o que o desvia do objectivo da sua investigação. A maior parte das aplicações de visualização desenvolvidas com um determinado objectivo fica demasiado comprometida com esse objectivo para se poder adaptar facilmente a outros fins.

Têm sido usadas aplicações para animação (p. ex. MOVIE.BYU) na geração de imagens de dados científicos, mas estas aplicações não se adaptam bem à visualização científica. Os inconvenientes prendem-se com o tipo de dados de entrada e com o ênfase colocado no realismo da visualização.

Recentemente foram desenvolvidas várias aplicações para visualização científica, como o apE e o AVS (ver [Dyer] e [Upson1]). Trata-se de aplicações altamente flexíveis, que permitem visualizar, de diferentes formas, uma grande diversidade de tipos de dados recorrendo a várias técnicas. Tanto o apE como o AVS são sistemas programáveis, suportando uma linguagem de programação visual, desenhada à volta da abstracção do fluxo de dados. O facto de serem sistemas programáveis torna-os muito flexíveis, mas pode ser um obstáculo à sua manipulação por um utilizador final não especialista.

Esta comunicação baseia-se num trabalho de criação dum ambiente para a visualização de dados científicos, consistindo em campos de dados definidos numa grelha tri-dimensional, resultantes da execução duma simulação ou obtidos por um sistema de aquisição de dados.

O objectivo não é obter um sistema com a flexibilidade do apE ou do AVS, mas um sistema de capacidades mais modestas, que permita visualizar os campos de dados referidos de uma forma que facilite a sua análise e interpretação, usando técnicas de visualização em volume.

Muito embora se pretenda obter a máxima generalidade possível, escolheu-se para domínio de aplicação a Mecânica de Fluidos, sendo os dados a visualizar provenientes de rigorosas simulações.

Pretende-se, com o desenvolvimento deste sistema, analisar os algoritmos para visualização em volume existentes e os problemas que surjam na sua implementação. Nesta comunicação, os autores pretendem justificar as técnicas de visualização seleccionadas e que foram convenientemente adaptadas para se proceder a uma implementação piloto. Concretamente, implementou-se um algoritmo de extracção de iso-superfícies por conversão em polígonos, bem como um algoritmo de visualização em volume por *ray-casting*.

2 - Requisitos

O problema central consiste na visualização dos resultados da simulação do comportamento de um fluido num determinado volume. Os resultados da simulação constituem um *array* de dados que correspondem à estimação de determinadas propriedades em pontos conhecidos do interior do volume. Estas propriedades consistem em grandezas vectoriais (velocidade) e grandezas escalares (pressão e temperatura, entre outras). É ainda necessário visualizar a turbulência do fluido, materializada por um valor escalar (energia cinética), juntamente com a velocidade.

Em geral, os dados em cada ponto podem ser compostos por grandezas escalares e vectoriais, interessando não só visualizar uma qualquer dessas grandezas, mas também visualizar simultaneamente duas ou mais dessas grandezas.

Os pontos em que os dados são definidos constituem uma grelha cuja geometria é especificada junto com os dados de cada problema. Designam-se estes pontos por nós da grelha. Os dados constituem um *array* tri-dimensional, correspondendo cada elemento a um determinado nó. A geometria consiste na definição da posição de cada nó no espaço (em coordenadas cartesianas), e pode ser dada explícita ou implicitamente. Neste último caso, existe alguma regularidade, sendo dados elementos que permitem calcular a posição de cada nó. A grelha divide o volume em células hexaédricas.

[Speray] (citado por [Collins]) propõe uma classificação dos tipos de geometria da grelha, que também será aqui adoptada:

Assim, uma grelha que divide o volume em células hexaédricas, sendo definida por um *array* tri-dimensional das posições de cada nó, designa-se por grelha estruturada, por contraposição às grelhas não estruturadas. Nestas, as células têm qualquer outra forma, não existindo a conectividade que a definição por um *array* impõe.

Uma grelha estruturada pode ser curvilínea (caso geral), ou rectilínea, caso em que as células são paralelepípedos rectângulos. Neste caso, a grelha pode ser definida por um conjunto de planos, paralelos aos eixos de um referencial. Assim, pode-se definir a geometria da grelha por

meio de três *arrays* que contenham cada um as distâncias à origem dos planos perpendiculares a um dos três eixos.

Particularizando ainda mais, uma grelha rectilínea pode ser regular, caso em que a distância entre os planos paralelos é constante. Se essa distância for a mesma para os planos paralelos aos três eixos, a grelha denomina-se cúbica.

As grelhas regulares e cúbicas possuem a geometria mais simples, pelo que o ponto de partida para a implementação de qualquer algoritmo de visualização é a sua implementação para grelhas regulares. Os dados provenientes das simulações referidas são definidos em grelhas rectilíneas, pelo que será necessário adaptar os algoritmos também a este tipo de grelhas.

A visualização de grelhas curvilíneas é consideravelmente mais complexa que a dos tipos mais simples. A estrutura menos regular das grelhas curvilíneas impede que se façam várias simplificações correntes nos algoritmos para grelhas regulares ou rectilíneas. Isto reflecte-se na literatura especializada, em que só recentemente surgiram artigos que tratam da visualização deste tipo de grelhas (ver [Neeman], [Wilhelms2] e [Gallagher]).

[Collins] classifica ainda as grelhas de acordo com a maneira como é feita a amostragem dos dados. No caso presente, os dados consideram-se definidos apenas nos nós da grelha. Sempre que for necessário obter o valor num outro ponto arbitrário, recorre-se à interpolação dos valores dos nós que formam os vértices da célula em que o ponto se encontra. Assume-se que os campos de dados variam de forma linear no interior das células, podendo então recorrer-se a métodos lineares de interpolação. Esta aproximação é adequada quando se considera que os valores dos nós são medidas pontuais (reais ou resultado de uma simulação) de um campo contínuo.

Noutras aplicações, é mais exacto considerar que o valor de um nó é o valor médio do campo no interior de uma célula que o envolve. Neste caso, considera-se que o valor dos dados em qualquer ponto no interior de uma célula é igual ao valor médio dessa célula. Este caso é mais frequente nas aplicações médicas (TAC, RNM, etc).

A aproximação escolhida para a amostragem dos dados tem implicações sobre o processo de visualização. A assunção de que o valor do campo é constante dentro de cada célula leva, para se obter imagens de aspecto contínuo, à necessidade de mais células (i.e. grelhas de maior dimensão) do que se se assumir uma variação linear do campo no interior das células. Em compensação, no processo de visualização, é inútil analisar-se o campo no interior das células.

Assumindo que os campos variam linearmente no interior das células, pode-se obter uma imagem de aspecto contínuo a partir de uma grelha relativamente larga. Mas a variação do campo obriga a analisar o seu valor em vários pontos no interior de cada célula, sob pena de não se obter o aspecto contínuo pretendido.

A variedade de formas em que se podem apresentar os dados de entrada implica uma atenção especial sobre a sua representação. Esta deve permitir ao sistema a determinação automática do tipo dos dados, da sua estrutura e das operações possíveis, assim como conter outros atributos que possam auxiliar a interpretação (dimensões, unidades, etc) (ver [Cunha1], [Cunha2] e [Farrell]).

3 - Formas de Visualização dos Dados

Discutem-se, nesta secção, as técnicas de visualização que melhor podem servir os fins em vista, tendo em conta as características dos dados que se pretende observar.

3.1 - Campo escalar

A representação de campos escalares num volume é um problema bem conhecido e encontram-se na literatura, sobre o assunto, numerosas técnicas para o tratar. No entanto, grande parte destas técnicas utilizam a aproximação do valor constante no interior das células.

Grande parte das técnicas assumem, também, que os dados estão definidos em grelhas regulares. A adaptação para grelhas rectilíneas é relativamente fácil, o mesmo não se passando com as grelhas curvilíneas, como já foi referido.

Outra hipótese é proceder à reamostragem dos dados originais de forma a obter uma grelha regular. Contudo, isto nem sempre é desejável, pois pode originar a perda de pormenores da imagem ou introduzir artefactos.

Algumas técnicas procedem a uma classificação dos valores do campo, procurando identificar zonas no volume pertencentes a cada classe. Estas técnicas destinam-se sobretudo a tratar dados provenientes de volumes compostos por vários materiais diferentes. A classificação procura então identificar as zonas de cada material no volume. Esta situação ocorre sobretudo em dados médicos (TAC, RNM), procurando a classificação identificar os diferentes tipos de tecido.

Tratando-se de dados que representam uma grandeza contínua, o que se pretende é apenas visualizar a grandeza em cada ponto do campo, não cabendo aqui considerar processos de classificação.

A primeira forma a considerar na representação de campos escalares, a iso-superfície, é uma das técnicas mais clássicas ([Lorensen] e [Wilhelms1]). Trata-se de construir superfícies contendo todos os pontos do volume em que o campo tem um determinado valor. Isto é o equivalente tri-dimensional das iso-linhas nas representações bi-dimensionais.

As superfícies são aproximadas por primitivas geométricas (polígonos, ou *patches* bi-cúbicos ([Gallagher])). O processamento é feito célula por célula, determinando para cada uma um conjunto de polígonos que aproxime a intersecção da célula com a superfície. A posição dos vértices destes polígonos é calculada por interpolação.

Em [Lorensen] processa-se uma célula de cada vez, aproximando a superfície por triângulos. Em [Wilhelms1] utiliza-se a subdivisão recursiva do volume, gerando uma representação tipo *octree* que é depois utilizada para atravessar apenas os voxels que contêm iso-superfícies.

Os polígonos resultantes são depois visualizados com as técnicas habituais, o que resulta em superfícies opacas, com possibilidade de adicionar efeitos de iluminação para melhorar a percepção da forma das superfícies. Nada impede que se visualizem simultaneamente várias superfícies.

Para se poder examinar o espaço interior limitado pelas iso-superfícies, espaço esse que pode conter outras iso-superfícies quando se visualizam várias, é desejável prever a possibilidade de fazer cortes, assim como controlar interactivamente a visibilidade de cada superfície. Deve-se, também, considerar a hipótese de visualizar superfícies translúcidas.

Outra forma de visualização possível é a visualização directa em volume (ver [Wilhelms2], [Sabella], [Levoy] e [Upson2]), isto é, sem passar por uma representação por primitivas geométricas. Esta tem a vantagem de evitar os artefactos que podem ser introduzidos pela aproximação por primitivas geométricas, além de ser bastante mais flexível, pois permite visualizar gamas de valores, ou mesmo todo o campo, de uma forma contínua, o que se adequa melhor à natureza contínua dos campos a visualizar. Com efeito, uma iso-superfície introduz uma classificação binária dos dados, o que não é o mais adequado para interpretar um campo contínuo.

Além disso, a representação das iso-superfícies por primitivas geométricas pode ocupar um espaço de memória excessivo, principalmente se a grelha for de grandes dimensões e se se representar simultaneamente um grande número de superfícies [Upson2].

Esta técnica implica a especificação de funções que, a cada valor possível do campo escalar, atribuem valores de cor e opacidade. A imagem resultante será a projecção de todos os pontos do campo num plano de projecção, sendo as cores dos pontos projectados no mesmo ponto do plano de projecção acumuladas de acordo com a sua opacidade. Deste modo, as zonas de opacidade nula não contribuirão em nada para a imagem, enquanto as de opacidade não nula serão visualizadas como uma zona da cor calculada pela função e com uma intensidade dependente da opacidade. As zonas com opacidade menor que o máximo são translúcidas, deixando ver as zonas que estão atrás

de si com uma intensidade que depende da sua opacidade. A imagem resultante tem a aparência de um gel semi-transparente com zonas de cores diferentes conforme o valor do campo.

Os algoritmos utilizados podem basear-se em *ray-casting* ou processamento célula por célula ([Levoy] e [Upson2]).

Os algoritmos de *ray-casting* analisam o volume com raios, originados no plano de projecção e que intersectam o volume. Para cada raio traçado, vão-se integrando os valores de cor e opacidade, interpolados a intervalos de amostragem regulares ao longo do raio, enquanto atravessa o volume. O pixel da imagem correspondente ao raio recebe, assim, um valor que é uma combinação de todos os valores do campo que o raio atravessou.

Os algoritmos célula por célula projectam cada célula no plano de projecção, determinando quais os pixels que são afectados pela célula. A cor de cada um destes pixels é obtida por integração no interior da célula, similar à do *ray-casting*. Os valores resultantes para os pixels são combinados com os que já se encontram no plano, originando um efeito semelhante ao obtido pelo *ray-casting*. Este tipo de algoritmos tem a vantagem de ser mais facilmente implementável em sistemas distribuídos.

Esta técnica também pode ser utilizada para visualização de iso-superfícies pelo uso de uma função de opacidade nula para todos os valores exceptuando determinadas gamas estreitas de valores, centradas nos limiares das iso-superfícies desejadas. A função de cor deve atribuir cores uniformes dentro de cada gama, e variando duma para outra. As superfícies são aproximadas com uma precisão dependente do intervalo de amostragem do raio e ficam com um aspecto diferente das obtidas por aproximação com polígonos, pois terão uma espessura que reflecte a razão de variação do campo. Nas zonas de maior variação a espessura será menor, e vice-versa. Este efeito é mais uma vantagem para a interpretação do campo.

A largura das gamas de valores opacas também pode ser variada para se obter superfícies com espessura diferente. Aumentando a largura das gamas até estas se tocarem obtém-se uma visualização contínua de uma gama larga de valores, com o valor discriminado pela cor.

3.2 - Campos vectoriais

O problema da visualização de campos vectoriais não está tão explorado como o dos campos escalares. A técnica mais vulgar em 2D, a utilização de ícones (setas, por exemplo) para representar os vectores, torna-se rapidamente demasiado confusa em 3D, devido à sobreposição de numerosos ícones.

As técnicas anteriores podem ser usadas na representação de uma função escalar do campo vectorial. Pode-se, por exemplo, representar o módulo do campo ou o produto interno dos vectores do campo com um vector de direcção conhecida. Contudo, é evidente que numa conversão de vector para escalar se perde informação, que pode ser importante ou não, consoante a natureza do campo e as características que se pretende observar.

Este inconveniente pode ser minimizado combinando a visualização de duas ou mais funções do campo vectorial, como seja tomar uma para modelar a cor das iso-superfícies da outra, através das mesmas técnicas usadas para os campos escalares. Por exemplo, pode-se modelar a cor das iso-superfícies do módulo do campo de acordo com a direcção do campo. A modelação de cores pelas grandezas a visualizar será discutida na secção seguinte.

O seguimento de partículas é uma técnica de visualização específica para campos vectoriais. Nesta técnica, e nas suas variantes, segue-se o movimento de partículas virtuais, injectadas em pontos definidos pelo utilizador, sujeitas à acção do campo e considerando os vectores como vectores velocidade [Hearn]. A imagem regista a trajectória das partículas, obtida calculando a posição das partículas a intervalos de tempo regulares.

Observando o percurso de um número suficiente de partículas, obtém-se uma boa percepção do campo. O número de partículas e os pontos onde são injectadas têm evidentemente uma grande

influência na facilidade de interpretação do resultado. Por isso, deve-se dar grande atenção aos critérios de determinação destes parâmetros, assim como possibilitar a selecção interactiva dos pontos de introdução das partículas.

Esta técnica sofre, em menor grau, do problema da representação por ícones: se se utilizarem demasiadas partículas a sobreposição das suas trajectórias torna a imagem resultante demasiado confusa.

Uma variante desta técnica utiliza fitas para materializar a trajectória das partículas. As fitas podem mostrar a vorticidade do campo, por meio da rotação da sua secção.

A técnica do seguimento de partículas pode ser combinada com a visualização do módulo do campo vectorial. Pode-se usar a cor das trajectórias para visualizar o módulo do campo (ou também outra grandeza qualquer). Também é possível sobrepor simplesmente uma representação por seguimento de partículas com uma representação do módulo do campo obtida por outra técnica.

3.3 - Campo multivariável

Neste caso trata-se de visualizar simultaneamente duas ou mais grandezas, de tipos iguais ou diferentes. Um exemplo deste caso é a representação simultânea da velocidade e da energia cinética da turbulência.

A visualização de campos vectoriais por meio da visualização de duas funções escalares pode ser vista como um caso particular deste, permitindo o emprego das mesmas técnicas. Em [Hearn] sugerem-se várias técnicas para campos multivariável.

Uma hipótese é a utilização de ícones de diversas formas, actuando cada grandeza numa característica dos ícones (forma, cor, etc). Mas, como foi referido acima no caso de campos 3D, a utilização de ícones resulta em imagens demasiado confusas, devido à sobreposição de numerosos ícones.

Uma técnica que não tem este inconveniente é a associação de atributos. Trata-se de obter uma representação de uma das grandezas segundo uma técnica habitual, associando outras grandezas a atributos dessa representação.

No caso das iso-superfícies, a segunda grandeza fica associada à cor das superfícies.

No caso do seguimento de partículas, além da cor pode-se alterar a espessura da linha que representa a trajectória.

Para a visualização directa em volume pode-se utilizar para funções de cor e opacidade funções de várias variáveis, correspondendo cada uma delas a uma grandeza a visualizar. As possibilidades são muitas, podendo-se, por exemplo, usar uma grandeza para controlar a opacidade e outra para a cor, ou então associar cada grandeza a um espaço de cor separado.

Quando se utiliza uma grandeza para modelar uma cor existem várias possibilidades. A mais vulgarmente utilizada é a construção de uma paleta, como sejam as cores do espectro, e associar a gama de variação da grandeza à paleta. É também possível associar cada componente da cor (vermelho, verde e azul, ou tom, saturação e intensidade) a uma grandeza diferente. Deste modo, a imagem conterá muito mais informação. Por exemplo, permite representar um vector sem perda de informação. Contudo, pode ser difícil interpretar as combinações de cores, e até mesmo variações na reprodução da cor pelos equipamentos terminais podem conduzir a graves erros de interpretação.

4 - Algoritmos de Visualização

Tendo em conta as considerações anteriores, decidiu-se proceder à implementação experimental de um algoritmo de extracção de iso-superfícies por aproximação por triângulos (triangulação) e de um de visualização directa em volume por *ray-casting*.

As técnicas de visualização para esta implementação experimental foram escolhidas tendo em conta as condições do problema atrás referidas. Decidiu-se implementar algoritmos básicos para estas técnicas, de modo a avaliar o seu comportamento nas condições particulares do problema, procedendo depois aos melhoramentos e optimizações (tanto em termos de qualidade dos resultados como de eficiência) que se revelassem necessários.

Os algoritmos implementados foram o de *Marching Cubes*, de Lorensen [Lorensen], de triangulação de iso-superfícies, e um algoritmo de visualização em volume por *ray-casting*, segundo [Levoy]. Estes são representantes típicos das duas técnicas referidas de visualização.

Numa primeira fase procedeu-se à implementação dos algoritmos para visualização de grelhas regulares, tendo-se depois adaptado os algoritmos para grelhas rectilíneas.

Implementou-se também um algoritmo de reamostragem de grelhas rectilíneas para grelhas regulares, com o objectivo de comparar a performance dos algoritmos de visualização de grelhas regulares e rectilíneas sobre dados equivalentes.

Os algoritmos foram testados em campos gerados para o efeito, a partir de funções matemáticas, antes de serem usados com os dados "reais", provenientes das simulações referidas no início do texto. A dimensão dos campos variava entre as 10 e 20 células de lado, aproximadamente.

4.1 - Triangulação de iso-superfícies

Na triangulação de iso-superfícies seguiu-se o algoritmo original de Lorensen. O campo é percorrido célula por célula e determinam-se quais os vértices da célula que estão no interior ou no exterior da superfície, atribuindo-lhes um valor booleano. Este valor é usado para indexar uma tabela de casos de intersecção que permite obter os triângulos correspondentes à intersecção da superfície com a célula.

A tabela de casos de intersecção contém, para cada vector de valores dos vértices, uma série de protótipos de triângulos. Designa-se por protótipo de triângulo a indicação das arestas que contêm cada vértice do triângulo.

A posição exacta dos vértices dos triângulos é obtida por interpolação linear ao longo da aresta indicada no protótipo. Os triângulos que compõem a superfície ficam assim completamente determinados, podendo ser então visualizados utilizando o sistema gráfico de base (graPHIGS), com as técnicas habituais de visualização com sombreamento, etc..

Como há 8 vértices, a tabela deve conter exactamente 2^8=256 entradas. Como enunciado por Lorensen, estes 256 casos podem ser reduzidos a 14 por duas propriedades de simetria: em primeiro lugar, os casos com um padrão de vértices negado (bit a bit) têm o mesmo padrão de intersecção, o que reduz os casos a metade; em segundo lugar, muitos casos são idênticos a menos de uma rotação da célula.

Para se obter um sombreado da superfície (método de Gouraud) é necessário fornecer ao graPHIGS vectores normais à superfície em cada vértice de cada triângulo. Estes vectores são calculados a partir do gradiente da superfície. A normal é obtida normalizando-se o vector gradiente, que é estimado nos nós da grelha usando diferenças centrais ao longo dos três eixos coordenados e é depois interpolado nos vértices dos triângulos. Os vectores normais são guardados junto com a posição dos vértices e passados ao graPHIGS.

Este algoritmo tem problemas conhecidos, nomeadamente o de poder produzir resultados incorrectos nalguns casos ambíguos em que não é possível determinar a iso-superfície apenas pelo valor dos vértices [Collins]. Estas ambiguidades podem fazer com que superfícies pareçam tocar-se onde na verdade não se toquem e vice-versa.

Este problema não pode ser eliminado mantendo o princípio do algoritmo. Os métodos propostos para eliminar este problema (ver [Wilhelms3], [Koide], [Purvis] e [Upson3]) repousam todos em algoritmos bastante diferentes.

Sendo as ambiguidades a nível da célula, contudo, o problema não se revelou importante para a utilização prática do algoritmo. Em imagens geradas a partir de dados reais e em condições normais não se detectaram ambiguidades graves.

4.2 - Visualização em volume por *ray-casting*

Como já foi referido, a visualização por *ray-casting* baseia-se na obtenção de amostragens do valor do campo em intervalos regulares ao longo de raios partindo da posição de cada pixel da imagem. Os valores das amostragens de cada raio são combinados por meio de duas funções, que aplicam os valores do campo em valores de opacidade e cor.

A implementação feita baseou-se no artigo de Marc Levoy ([Levoy]). Assim, para cada raio, procede-se à acumulação de cor e opacidade da frente para trás (sendo a frente o plano de projecção). A opacidade e a cor correntes são compostos com as da amostragem, em cada ponto desta, de acordo com as fórmulas seguintes:

opacidade := opacidade + op(amostragem)*(1-opacidade)

cor := cor + cl(amostragem)*(1-opacidade)

em que **opacidade** e **cor** são os valores correntes da opacidade e cor, e **op()** e **cl()** são as funções referidas.

Cada raio é descrito, numa forma paramétrica, por uma origem e uma direcção. A origem coincide com o centro do pixel correspondente no plano de projecção. A direcção é um vector com a direcção de projecção. Qualquer ponto do raio pode ser representado por um valor real, o seu parâmetro. Para obter as coordenadas cartesianas basta somar (vectorialmente) à origem o vector direcção multiplicado pelo parâmetro. O comprimento do vector direcção é igual à distância entre amostragens desejada, de modo que se numerarmos os pontos de amostragem a partir do plano de projecção, o parâmetro de qualquer ponto de amostragem ao longo do raio coincide com o seu número.

Para traçar cada raio, o algoritmo começa por determinar os parâmetros dos pontos de entrada e saída do raio no volume. Em seguida, calculam-se as funções "tecto" e "chão" do ponto de entrada e de saída respectivamente, obtendo-se o número da primeira e última amostragens dentro do volume.

Em cada um destes pontos procede-se à interpolação do valor do campo, aplicando-se em seguida a função de opacidade ao valor obtido. Se a opacidade for zero, passa-se imediatamente ao ponto seguinte, senão aplica-se também a função de cor ao valor interpolado, compondo-se estes valores com os valores correntes do raio de acordo com as fórmulas acima. Estes valores são colocados a zero antes de se processar cada raio. Para se avançar de um ponto para outro basta somar ao ponto corrente o vector direcção. A cor do pixel é a cor corrente do raio depois de ter atravessado o volume.

As funções de cor e opacidade poderão ter como argumentos, além do valor do campo, a posição no espaço do ponto de amostragem (permitindo implementar *depth-cueing* e cortes) e o gradiente do campo, previamente calculado (permitindo adicionar algum sombreado).

As imagens obtidas são bastante elucidativas da distribuição dos valores do campo pelo volume. Contudo a escolha das funções de cor e opacidade revelou-se bastante crítica para a clareza dos resultados.

A ocorrência de fenómenos de *aliasing* revelou-se, também, um problema. Depende de factores como o número de amostras por raio e a velocidade de variação das funções de cor e opacidade. Designa-se por frequência de amostragem o número de amostras por raio.

Experimentaram-se diversas funções para a cor e opacidade, procurando-se obter os resultados mais fáceis de interpretar e com um mínimo de *aliasing*.

Quanto à frequência de amostragem, verificou-se serem necessárias frequências importantes para que o *aliasing* não se torne demasiado evidente, o que agrava a ineficiência do algoritmo que, como seria de esperar de um algoritmo de *ray-casting*, não é muito rápido.

Verificou-se que a ineficiência do algoritmo, mesmo com frequências de amostragem relativamente baixas, impede uma utilização interactiva. A lentidão do algoritmo levou a implementar as duas optimizações propostas por [Levoy], que consistem na terminação adaptativa do raio e na utilização de *octrees* para permitir saltar zonas vazias do volume.

Procurou-se analisar a eficácia destas técnicas de optimização, assim como o efeito da variação de determinados parâmetros destas técnicas (limite de opacidade para a terminação adaptativa e níveis máximo e mínimo das *octrees*) na eficiência do algoritmo.

Globalmente, conseguiu-se uma melhoria considerável da eficiência, mas não a ponto de permitir a utilização interactiva.

4.2.1 - Técnicas de optimização

A terminação adaptativa consiste simplesmente em terminar o raio assim que a opacidade acumulada é tal que as amostras que ainda faltam não podem influenciar muito a cor final. A eficácia desta técnica depende da percentagem de zonas quase opacas do volume, sendo completamente ineficaz para volumes inteiramente translúcidos.

A implementação é trivial, bastando interromper a progressão do raio quando a opacidade acumulada é maior que um determinado limite. O valor deste limite (no máximo 1) representa um compromisso entre a qualidade da imagem e o grau de optimização obtido.

Com os campos testados na prática, o valor sugerido por [Levoy] de 0.95 revelou-se um bom compromisso, permitindo obter ganhos de eficiência da ordem dos 15% a 50%, sem degradação notória da qualidade da imagem. A diminuição para 0.85 não permtiu um ganho substancialmente maior (18% a 60%), começando-se a notar artefactos na imagem. Estes artefactos são mais notórios com frequências de amostragem baixas.

A optimização por *octrees* consiste na utilização de uma estrutura em *octree* (veja-se também [Wilhelms1]), construída previamente, para identificar zonas de opacidade nula que podem ser saltadas por não contribuírem em nada para a imagem. A *octree* é uma subdivisão hierárquica do volume. O nível 0 da *octree* corresponde a todo o volume e o nível M corresponde a uma divisão em subvolumes iguais às células da grelha. Os níveis intermédios correspondem, cada um, à divisão do anterior em 8 subvolumes. O subvolume de cada nível contém um valor booleano que indica se a opacidade do subvolume é 0 ou maior que 0.

O processo de geração da *octree* é um processo recursivo. Começa-se por analisar o volume inteiro, analisando-se depois recursivamente os seus subvolumes. A recursividade termina quando se analisam as células da grelha. A análise de cada subvolume é feita usando a função de opacidade que será usada depois para o cálculo da cor dos raios.

O algoritmo de visualização com a *octree* tem uma estrutura diferente do original. O avanço do raio é feito subvolume por subvolume, no nível corrente da *octree*. Consoante esse subvolume tenha opacidade nula ou não, salta-se o subvolume ou sobe-se um nível para examinar os seus

subvolumes. Quando se chega ao nível M, procede-se então à acumulação de cor e opacidade como anteriormente, em pontos de amostragem a espaços regulares dentro do subvolume. Depois de se atravessar um subvolume, deve-se verificar se foi atravessada a fronteira de um subvolume de nível inferior e, nesse caso, desce-se de nível.

A eficácia desta optimização depende largamente do conteúdo da imagem, sendo mais eficiente quanto mais zonas completamente transparentes houver no volume, o que é o inverso da técnica de terminação adaptativa. Isto resulta num melhoramento considerável se a função opacidade for zero para certas gamas do valor do campo. Se a opacidade nunca for zero, esta optimização é completamente ineficaz.

Na prática, verificou-se que esta optimização é realmente eficaz para volumes com espaços vazios, tendo-se alcançado 70% de melhoria para alguns ficheiros de teste. Mas nalguns ficheiros com dados reais verificou-se o oposto, isto é, uma pioria do tempo de execução, mesmo garantindo que existem espaços vazios. Num caso verificou-se uma pioria da ordem dos 20%. Este facto deve-se à complexidade extra introduzida pelo cálculo das intersecções com os subvolumes, bem como das subidas e descidas de nível, que só é compensada se existirem grandes espaços de opacidade nula.

O tempo de geração da *octree* não revelou ser excessivo, com tempos de CPU da ordem dos 0.66 s para um volume de 20x20x20 células e 5.4 s para um volume de 40x40x40 células.

O algoritmo utiliza dois parâmetros, nível_max e nível_min, para se evitar perder tempo a examinar os subvolumes de níveis próximos de 0, que quase nunca serão transparentes, e as de níveis próximos de M, para os quais o tempo poupado por se poderem saltar não compensa o tempo do cálculo da intersecção. Analisou-se o efeito da variação destes parâmetros, tendo-se concluido que os melhores resultados se obtêm com nível_min igual a 2 e nível_max igual a M-3 ([Levoy] refere os valores de 2 e M-2 para estes parâmetros).

Pelo facto de poder conduzir a uma pioria da eficiência em certas condições, esta técnica de optimização deve ser sempre opcional, ao passo que a terminação adaptativa pode ser sempre utilizada sem inconvenientes.

4.2.2 - Funções de cor e opacidade

As funções de cor e opacidade do algoritmo de *ray-casting* devem ser tais que permitam discriminar os valores do campo que interessa observar. Na prática, as funções usadas podem ser de dois tipos (ver [Upson2]):

- as funções em escala (a opacidade varia de zero até ao máximo, linearmente com o valor do campo, e a cor atravessa vários valores que permitam discriminar bem o valor do campo);
- a opacificação de determinadas gamas (a opacidade é nula excepto para as gamas de valores que interessam, dentro das quais a cor toma valores diferentes de modo a permitir identificá-las).

Pelas experiências efectuadas verificou-se que as funções mais úteis eram as opacificações de gamas. Podem-se visualizar gamas isoladas ou associar várias para visualizar uma gama extensa com várias cores.

Dedicou-se algum tempo a testar diferentes formas para estas funções. Verificou-se que o modo como a opacidade varia influencia, não só a facilidade de interpretação da imagem resultante, como a gravidade dos problemas de *aliasing*. Assim, verificou-se ser conveniente evitar saltos bruscos nos valores das funções, devendo todas as transições de valores ser feitas gradualmente. Os melhores resultados foram obtidos com funções em forma de triângulo, com o pico centrado na gama a opacificar.

Usando funções em triângulo, pode-se obter uma função em escala definindo uma função de gama suficientemente extensa, pelo que não é necessário incluir explicitamente funções em escala. As funções de cor e opacidade são, portanto, definidas por um somatório de funções em triângulo.

A variação da cor dá origem a um fenómeno que melhora o aspecto e a facilidade de interpretação das imagens: a cor atinge valores baixos (cor mais escura) próximo dos extremos da gama; como a opacidade ainda é maior que zero, isto origina uma zona parcialmente opaca e mais escura que o centro da gama. Por outras palavras, as regiões "luminosas" do volume estão cercadas por uma fina parede escura, mas opaca. Esta parede nota-se menos nas superfícies perpendiculares ao raio de visão, e mais nas superfícies quase paralelas ao raio (o raio atravessa uma maior porção da zona escura). Por isso, obtém-se aproximadamente o efeito de um sombreamento, com as suas vantagens para a interpretação da imagem tridimensional, sem os custos de um verdadeiro cálculo do sombreamento.

5 - Campos multivariável

A necessidade de visualização de campos vectoriais e de duas ou mais grandezas simultaneamente levaram à extensão das técnicas anteriores de modo a permitir o processamento de campos multivariável.

Dada uma grelha com várias grandezas definidas, o sistema deve permitir seleccionar as grandezas a visualizar e, ainda, introduzir eventuais funções de conversão, necessárias, por exemplo, para se obter o módulo de uma grandeza vectorial.

5.1 - Técnicas de visualização para campos multi-variável

Utilizou-se a técnica de associação de atributos, referida em 3.3, com as técnicas de visualização implementadas.

Assim, as iso-superfícies obtidas pelos *Marching Cubes* em relação a uma variável são coloridas de acordo com o valor de outra variável (ou de uma combinação de várias) nos pontos da superfície. A cor é obtida por meio de funções que aplicam cada valor possível de uma variável numa cor, semelhantes às utilizadas no algoritmo de *ray-casting*.

No algoritmo de *ray-casting*, as funções de cor e opacidade do *ray-casting* podem passar a ser definidas como funções de várias variáveis, tantas como as variáveis que se pretende observar. Isto permite, por exemplo, controlar com uma variável as zonas em que outra é visualizada, usando uma variável para definir a opacidade e outra para definir a cor, ou então opacificar gamas de valores de várias variáveis, com uma cor diferente para cada uma.

5.2 - Dados e atributos

Introduziu-se uma operação de selecção das variáveis a visualizar, de entre todas as que estão definidas na grelha, o que é conveniente para evitar guardar em memória e interpolar variáveis que não vão ser utilizadas. É neste ponto que se podem também introduzir eventuais funções de conversão de vector para escalar, ou outras. As funções de transformação podem ter parâmetros extra, além do dado a transformar. Estes parâmetros são definidos pelo utilizador e armazenados com a transformação.

Surge assim a distinção entre o conjunto de variáveis que são fornecidas como dados de entrada e o conjunto de variáveis para a visualização, que será um subconjunto do primeiro, com alguns valores eventualmente transformados. Convencionou-se designar os primeiros por dados e os segundos por atributos.

As funções de cor e opacidade operam sobre um vector de atributos, e identificam cada atributo pelo seu índice no vector de atributos.

5.3 - Funções de cor e opacidade para campos multi-variável

As funções de cor e opacidade para o *ray-casting* podem continuar a basear-se na associação de funções em forma de triângulo. Contudo, é necessário acrescentar a cada gama de valores o índice do atributo a usar para essa gama.

Para se poder controlar a opacidade e a cor por variáveis diferentes, é necessário manter também o índice do atributo que controla a opacidade e do atributo que controla a cor.

Nos *Marching Cubes*, a cor das iso-superfícies passa a ser definida por uma função, em vez de uma constante, como no caso escalar. Esta função pode ser definida de maneira idêntica às funções de cor do *ray-casting*, permitindo assim modelar a cor das superfícies pelo valor de qualquer das variáveis.

6 - Discussão dos resultados

A aplicação dos algoritmos às simulações do Departamento de Mecânica da FCT permitiu verificar a sua utilidade para analisar dados reais. As maiores grelhas testadas tinham uma dimensão de 23x25x17 células (grelha rectilínea) e 40x40x40 células (grelha regular).

A visualização de iso-superfícies permite visualizar a distribuição de um determinado valor do campo com grande precisão. Pequenas irregularidades são imediatamente reveladas por saliências ou reentrâncias, que o sombreamento permite evidenciar. A visualização simultânea de várias iso-superfícies com diferentes cores permite visualizar uma gama de valores, mas, dependendo da sua distribuição, uma superfície pode ocultar outras por completo. Para examinar estes casos seria útil a possibilidade de se fazer um corte no conjunto de superfícies.

A geração das iso-superfícies é bastante rápida, sendo o tempo gasto da mesma ordem que o necessário para carregar a grelha de um ficheiro para memória. Depois de geradas as superfícies, o sistema graPHIGS permite visualizá-las com uma rapidez suficiente para permitir a variação interactiva do ponto de vista.

O algoritmo de *ray-casting* permite visualizar desde toda a gama de valores do campo de uma forma contínua, com a opacidade a variar de zero até ao máximo ao longo da gama e a cor a atravessar diversos valores, até gamas estreitas de valores, que aparecerão opacas num meio transparente, eventualmente com cores diferentes. Estas são uma aproximação das iso-superfícies mas, ao contrário do caso anterior, podem ser parcialmente transparentes e têm espessura.

O *ray-casting* permitiu identificar, nos dados analisados, zonas do volume com o valor cumprindo determinadas condições, por exemplo, zonas quentes num campo de temperatura, ou gerar iso-superfícies aproximadas parcialmente transparentes. Estas ficam mais difusas que as obtidas pelos *Marching Cubes*, não permitindo a identificação de variações tão pequenas, mas permitindo uma melhor visão do conjunto devido à sua transparência.

Compararam-se os resultados da visualização de grelhas rectilíneas com os da visualização das mesmas convertidas para grelhas regulares por reamostragem. A perda de pormenor é visível, especialmente se a grelha rectilínea tem células muito pequenas. A *performance* dos algoritmos não é muito afectada, o que faz com que, na maioria dos casos, seja preferível processar directamente as grelhas rectilíneas.

O tempo necessário para gerar uma imagem depende principalmente do número total de amostras. Mantendo constantes as dimensões da imagem, este número depende da frequência de amostragem e das dimensões do volume. O tempo varia ainda com o conteúdo da grelha, já que as

amostras de opacidade nula têm um tratamento mais simples. A utilização das técnicas de optimização torna o tempo ainda mais dependente do conteúdo da grelha.

Assim, para 5 amostras por célula e um volume de 20x20x20 células obtiveram-se tempos entre 370 e 740 s. As dimensões da imagem são de 512x512 pixels. Para o mesmo volume e 20 amostras por célula, os tempos foram entre 1440 e 1460 s.

7 - Conclusão

Analisadas as técnicas de visualização existentes, procurou-se seleccionar as mais adequadas à visualização de campos tri-dimensionais representando funções contínuas. A implementação dos dois algoritmos referidos permitiu sentir os problemas, de complexidade e eficiência, postos pela visualização de campos 3D nas condições referidas.

Há, no entanto, fenómenos de *aliasing* inerentes a essa mesma implementação. Embora não se tendo referido na comunicação, houve necessidade de resolver tais problemas na prática, nomeadamente quanto à interferência entre a frequência de amostragem e as variações do valor da opacidade e da cor.

O resultado final permitiu visualizar com sucesso os campos provenientes das simulações de Mecânica de Fluidos referidas na introdução, não devendo haver problemas em utilizá-lo para quaisquer outros domínios em que se necessite de visualizar campos tri-dimensionais, escalares, vectoriais e multivariável, representando funções contínuas, definidos em grelhas de pequena dimensão.

Junta-se, como Apêndice, um conjunto de quatro figuras que servem de ilustração ao texto da comunicação e foram obtidas pelos programas nele referidos. A linguagem de programação utilizada foi C, numa *workstation* RISC/6000 530H da IBM e dispondo de um processador gráfico especializado. Como sistema gráfico escolheu-se o graPHIGS, também da IBM.

Agradecimentos

Os autores desejam agradecer ao Prof. Dr. José Dias, do Departamento de Mecânica da FCT-UNL, autor das simulações e dados com que foram testados os algoritmos de visualização, pelas sugestões quanto às técnicas de visualização (do ponto de vista da interpretação dos dados por um especialista), bem como pela apreciação dos resultados obtidos.

8 - Bibliografia

[Collins] **B.M.Collins** "Data Visualization", *IBM UK Scientific Centre 1/92*

[Cunha1] **J.D.Cunha, A.P.Cláudio, M.B.Carmo.** "Tipos de dados visualizáveis -- aplicação no domínio da Engenharia Civil". *Actas do IV EPCG*, pp: 19, 11/91

[Cunha2] **J.D.Cunha.** "Visualizable data types -- Gen. purp. mapping in the visualization pipeline". *Eurographics Workshop(2)*, 4/91

[Dyer] **D.S.Dyer.** "A dataflow toolkit for visualization". *IEEE Computer Graphics & Applications*, 10(4) pp: 60, 7/90

[Farrell] **E.J.Farrell, P.A.Appino, D.P.Foty,..** "Visual interpretation of multidimensional computations and transistor design". *IBM Journal of Research and Development*, 35(1) pp: 26, 1991

[Gallagher] **R.S.Gallagher, J.C.Nagtegaal.** "An efficient 3D visualization technique for finite element models and ...". *ACM Computer Graphics*, 23(3) pp:185, 7/89

[Hearn] **D.D.Hearn, P.Baker.** "Scientific Visualization" *Eurographics'91 Tutorial Note 6, Eurographics Technical Report Series*, 9/91, ISSN 1017-4656

[Koide] **A.Koide, A.Doi, K.Kajioka.** "Polyhedral Approximation Approach to Molecular Orbital Graphics". *Journal of Molecular Graphics*, 4(3) pp:149, 9/86

[Levoy] **M.Levoy.** "Efficient ray-tracing of volume data". *ACM Transactions on Graphics*, 9(3) pp:245, 7/90

[Lorensen] **W.E.Lorensen, H.Cline.** "Marching cubes: a high resolution 3D surface construtions algorithm". *ACM Computer Graphics*, 21(4) pp:163, 7/87

[McCormick] **B.H.McCormick,T.A.Fanti,M.D.Brown.** "Visualization in scientific computing". *ACM Computer Graphics*, 21(6), 11/87

[Neeman] **H.Neeman.** "A decomposition algorithm for visualizing irregular grids". *ACM Computer Graphics*, 24(5) pp: 49, 11/90

[Purvis] **D.G.Purvis, C.Culberson.** "On the Graphical Display of Molecular Electrostatic Force-Fields and Gradient...". *Journal of Molecular Graphics*, 4(2) pp: 89, 6/86

[Sabella] **P.Sabella.** "A rendering algorithm for visualizing 3D scalar fields". *ACM Computer Graphics*, 22(4) pp: 51, 8/88

[Speray] **D.Speray, S.Kennon.** "Volume probes: interactive data exploration on arbitrary grids". *ACM Computer Graphics*, 24(5) pp: 5, 1990

[Upson1] **C.Upson,T.Faulhaber,D.Kamins,D.Laid.** "The application visualization system: a computational environment for scientific visualization". *IEEE Computer Graphics & Applications*, 9(4) pp: 30, 7/89

[Upson2] **C.Upson, M.Keeler,** "V-buffer visible volume rendering". *ACM Computer Graphics*, 22(4) pp: 59, 8/88

[Upson3] **C.Upson.** "The visual simulation of amorphous phenomena". *Visual Computer*, 2(5) pp:321, 9/86

[Wilhelms1] **J.Wilhelms, A. van Gelder.** "Octrees for faster isosurface generation". *ACM Computer Graphics*, 24(5) pp: 57, 1990

[Wilhelms2] **J.Wilhelms, J.Challinger.** "Direct volume rendering of curvilinear volumes". *ACM Computer Graphics*, 24(5) pp: 41, 1990

[Wilhelms3] **J.Wilhelms, A. van Gelder.** "Topological Considerations in Isosurface Generation - Extended Abstract". *Computer Graphics*, 24(5) pp: 79, 11/90

Apêndice

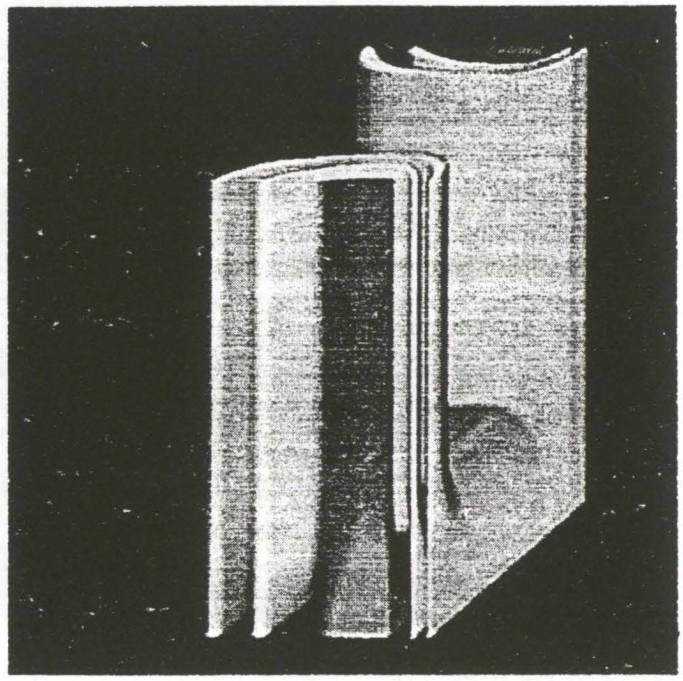

Figura 1 - Opacificação de 3 gamas, por *ray-casting*. O campo consiste num "U" de amostras de valor 1 num volume de valor 0. A transição entre os valores é suavizada pela interpolação.

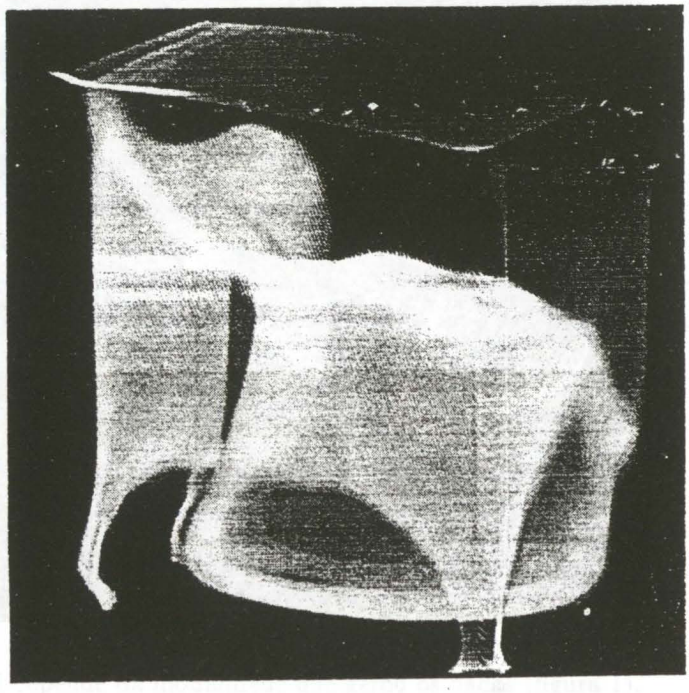

Figura 2 - Opacificação de uma gama, por *ray-casting*. O campo é obtido de uma simulação da energia cinética da turbulência de um fluido num volume paralelepipédico rectângulo com uma abertura, aquecido numa das faces.

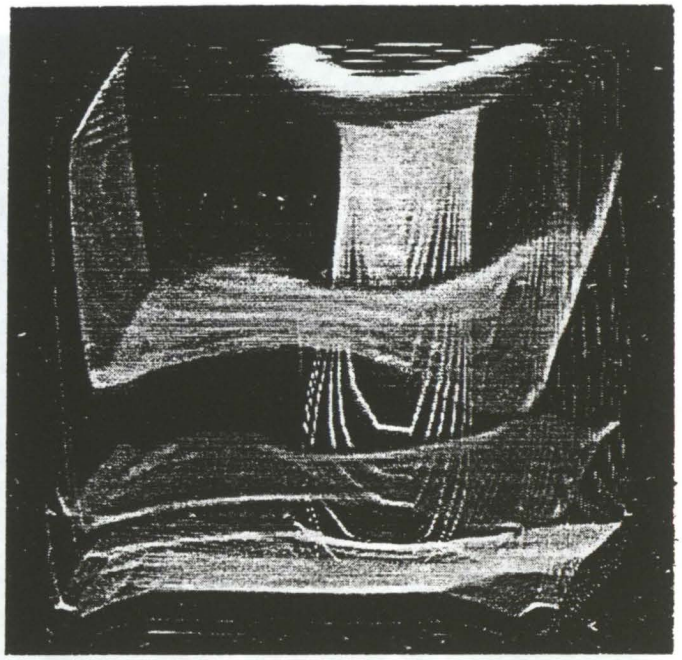

Figura 3 - Obtenção de 3 iso-superfícies e opacificação de uma gama extensa, por *ray-casting*. O campo é obtido da simulação da temperatura de um fluido nas mesmas condições da figura anterior. Nota-se aqui a ocorrência de *aliasing*.

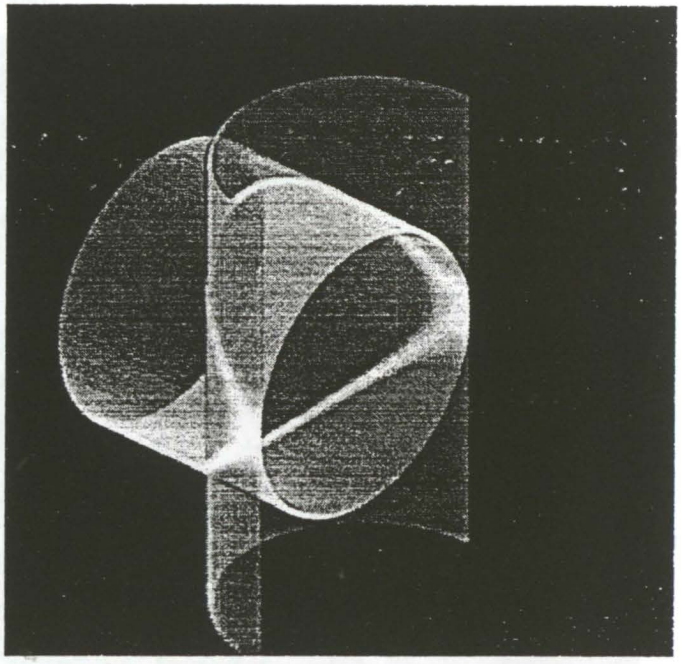

Figura 4 - Visualização simultânea de duas grandezas escalares. Ambas consistem no valor da distância em relação a um eixo, horizontal para uma e vertical para outra. Obteve-se uma iso-superfície de cada uma, por *ray-casting*. O volume foi dividido por um plano de corte.