Contornos Sugestivos em Superfícies Implícitas

João Proença, Joaquim A. Jorge
Dep. Engª. Informática
Instituto Superior Técnico, UTL
jrpp@mega.ist.utl.pt, jorgej@acm.org

Mário Costa Sousa
Dep. of Computer Science
University of Calgary, Canada
mario@cpsc.ucalgary.ca

Resumo

Este artigo apresenta um sistema que combina superfícies implícitas, definidas a partir de nuvens densas de pontos, com rendering rápido baseado em linhas. Concebemos um novo processo para extrair contornos sugestivos rapidamente, espalhando partículas ao longo da superfície, que identificam zonas de interesse, e aplicando métodos de clustering e encaixe de linhas. Adicionalmente, melhorámos alguns dos métodos mais recentes para extrair silhuetas e arestas, usando o poder descritivo da representação da superfície. Esta permite-nos obter heurísticas para a determinação rápida de curvatura e também suporta a regeneração local de elementos visuais em operações de edição da superfície. Apresentamos exemplos visuais que ilustram a alta qualidade dos desenhos obtidos com a nossa aplicação, bem como o alto detalhe que ela consegue proporcionar para os modelos mais complexos, e também mostramos que os nossos tempos de processamento revelam um melhor desempenho comparativamente a outras abordagens semelhantes, para o mesmo número de pontos.

Palavras Chave

Superfície implícita, silhueta, aresta, contorno sugestivo, sistema de partículas, Non-Photorealistic Rendering.

1. INTRODUÇÃO

Os métodos de desenho não fotorealista (*Non-Photorealistic Rendering* – NPR) são normalmente usados para transmitir a forma de objectos tridimensionais de uma maneira semelhante às conhecidas técnicas de desenho artístico. A maioria do trabalho recente nesta área, que efectua a extracção em tempo real de linhas a partir de objectos 3D, focou-se principalmente em malhas poligonais. No entanto, apesar de as malhas serem estruturas adequadas para visualização, elas apresentam problemas sérios para a modelação geométrica. Nomeadamente, o consumo de memória aumenta com a dimensão da forma e o nível de detalhe e é difícil manter a consistência topológica em operações de edição. Existem outras representações que são mais adequadas para este tipo de operações e para as quais se devem criar novas técnicas de *rendering* directo. Uma dessas representações é a superfície implícita, que oferece uma definição matemática simples e flexível que deriva de uma função potencial.

Neste artigo, apresentamos técnicas para a extracção rápida de linhas reveladoras da forma de superfícies implícitas MPU (*Multi-level Partition of Unity*) [Ohtake03], que permitem representar modelos complexos a partir de nuvens densas de pontos. Estas técnicas foram aplicadas a um sistema já existente [Araujo04] que permite a edição interactiva de objectos complexos definidos através da MPU. O utilizador consegue aplicar operações de modelação através de uma interface caligráfica que converte traços 2D em alterações da forma 3D. A aplicação das nossas técnicas neste sistema é justificada pela nossa motivação em usar estes métodos de *rendering* num ambiente no qual a edição da superfície é suportada.

Figura 1 - Superfície MPU computada com erro relativo 5×10^{-4}. Esquerda: o modelo *David's Head* com 59994 partículas espalhadas pela superfície. Direita: o mesmo modelo desenhado pelo nosso sistema usando silhuetas, arestas e contornos sugestivos.

A nossa abordagem inspirou-se nos métodos apresentados por [Foster05] para construir um sistema de partículas sobre a superfície implícita, extrair silhuetas e arestas a partir das posições dessas partículas e remover linhas ocultas usando *surfels*. No entanto, esta abordagem não suporta contornos sugestivos e utiliza uma representação implícita diferente (*BlobTree*), que suporta a definição de objectos a partir de CSG (*Constructive Solid Geometry*). Tivemos um maior interesse na MPU, uma vez que permite a representação de modelos complexos a partir de

grandes conjuntos de pontos que podem ser obtidos a partir de *scans* 3D de objectos reais. Uma das principais vantagens de usar a MPU é a estrutura *octree* [Ohtake03] associada, que estabelece uma divisão celular 3D do espaço circundante ao objecto, na qual o nível de subdivisão celular se torna maior em áreas de elevada complexidade. Destacamos em baixo as principais contribuições do nosso trabalho:

- Apresentamos uma nova técnica para extrair contornos sugestivos de superfícies implícitas, através da identificação de partículas que se encontram em áreas da superfície onde um contorno sugestivo existe, o agrupamento destas (*clustering*) e a aplicação de algoritmos de encaixe de linhas. Este processo funciona até quando a informação de curvatura local não é bem comportada, o que faz com que a nossa técnica seja generalizável à maioria das representações implícitas, não só as MPU.
- Usamos a informação de subdivisão celular da MPU como heurística que melhora, de várias formas, a precisão e desempenho da simulação do sistema de partículas e do processo de extracção de linhas. De facto, consultando o nível hierárquico de cada célula, conseguimos determinar heuristicamente comprimentos de linha para contornos, que se tornam mais curtos em áreas de grande detalhe (níveis celulares profundos) e mais compridos em regiões mais planas (células próximas da raiz da *octree*).
- Também usamos a *octree* da MPU para estabelecer as posições iniciais das partículas ao longo da superfície, o que nos proporciona um processo automático para escolher o número de partículas que é adequado para cada modelo, sem necessidade de intervenção do utilizador, contrariamente a outros métodos [Foster05] que usam inicializações aleatórias e requerem ajustes manuais.
- Damos suporte para elementos dinâmicos, usando a *octree* da MPU para permitir que as partículas e as linhas independentes da vista se regenerem localmente depois de operações de modelação, até em superfícies com dezenas de milhares de pontos.

Todos estes elementos são conjugados num sistema que combina a expressividade do desenho baseado em linhas com o poder descritivo da MPU para a visualização e modelação de objectos definidos por nuvens densas de pontos.

2. TRABALHO RELACIONADO

A maioria dos métodos desenvolvidos em NPR, que incluem silhuetas, arestas ou contornos sugestivos, focam-se em malhas poligonais e têm vindo a ganhar destaque nos últimos anos [Isenberg03, Gooch99, Sousa03, DeCarlo03, DeCarlo04]. No entanto, não tem havido muito trabalho a abordar NPR sobre superfícies implícitas e só nos anos mais recentes surgiram as primeiras abordagens [Bremer98, Elber98, Plantinga03, Foster05].

Bremer e Hughes [Bremer98] apresentaram métodos para o *rendering* de linhas sobre superfícies implícitas, que usam *ray-intersection* para determinação de pontos sobre a superfície e processos de integração numérica para fazer com que eles se aproximem das silhuetas e as sigam. O mesmo tipo de técnicas baseadas em raios são usadas para posicionar pequenos traços internos e efectuar remoção de linhas ocultas (RLO). Mais recentemente, Foster et al. [Foster05] propuseram técnicas que combinaram estes métodos com algumas das ideias apresentadas por Elber [Elber98], como a utilização de um sistema de partículas Witkin-Heckbert [Witkin94] para espalhar pontos sobre a superfície em vez de algoritmos de *ray-intersection* com elevado peso computacional. Usando uma variação do método *Shrink-wrap* [vanOverveld04], é possível extrair também arestas das partículas através da identificação de *straddle points* ao longo de faces da superfície. Adicionalmente, um método de RLO que usa *surfels* também é apresentado neste artigo. Em termos de sistemas de partículas do tipo Witkin-Heckbert, Levet et al. [Levet06] apresentaram recentemente uma técnica que utiliza processamento geométrico para encontrar posições iniciais quase óptimas para as partículas, o que melhora o subsequente tempo de simulação, mas a sua abordagem requer cálculos mais complexos neste passo do que os nossos. [Jepp06] também estendeu o desenvolvimento do sistema de partículas de [Foster05] para criar *smarticles* que usam *flocking behaviour* para a procura de descontinuidades e desenho de linhas sobre superfícies implícitas.

Em termos de contornos sugestivos, eles formam um conceito bastante recente que foi introduzido por DeCarlo et al. [DeCarlo03, DeCarlo04]. Estes dois artigos apresentam definições matemáticas para contornos sugestivos numa superfície e a sua extracção de malhas poligonais. [DeCarlo03] apresenta as definições formais e duas abordagens algorítmicas para produzir imagens estáticas a partir de objectos 3D (uma no espaço objecto e outra no espaço imagem). Em [DeCarlo04] os autores estendem o sistema anterior para desenhar contornos sugestivos em tempo real, usando técnicas de análise do movimento das linhas e algoritmos rápidos para eliminação de polígonos no processo de procura de linhas. Estes artigos estabeleceram as bases para o desenho de contornos sugestivos a partir de malhas, mas não existem muitos artigos que abordem a sua extracção a partir de outro tipo de representações. [Burns05] apresenta técnicas para a extracção de silhuetas e contornos sugestivos de dados volumétricos, usando *level-sets* auxiliares, mas os seus métodos parecem ser mais adequados a informação volumétrica e não apenas para superfícies únicas como os nossos. [Schmidt06] tenta desenhar estas linhas sobre superfícies implícitas, mas são utilizadas malhas de baixa resolução num passo intermédio da extracção de linhas, o que pode ser um factor limitativo em alguns cenários. Um dos nossos objectivos era o de evitar este tipo de abordagem porque o processo de reconstrução da malha em operações CSG não é trivial sobre uma superfície implícita. Daquilo que conhecemos, não existem artigos publicados que apresentem a extracção de contornos sugestivos a partir de superfícies sem recorrer a uma malha.

A nossa abordagem utiliza as definições de contornos sugestivos apresentadas em [DeCarlo03] para identificar partículas na superfície MPU, criar *clusters* dessas partículas e formar linhas de contorno através da aplicação de algoritmos de encaixe de linhas aos *clusters*. Este método inspirou-se no trabalho apresentado por [Barla05], que aplica *clustering* geométrico para simplificação de desenhos com linhas. No entanto, os seus métodos baseiam-se na premissa de que existe um grande conjunto de linhas a simplificar, o que não é o nosso caso. Adicionalmente, esta técnica opera no espaço imagem, o que impossibilitaria a utilização da nossa abordagem de RLO baseada em *surfels* para remover contornos sugestivos. Como tal, a nossa estratégia evoluiu para a extracção de curvas que representam os grupos de pontos 3D. [Gumhold01] apresentou métodos para a extracção de linhas de descontinuidade de nuvens de pontos, onde são construídas *minimum-spanning-trees* sobre os *clusters* e depois procede-se ao corte dos ramos curtos das mesmas para obter as linhas representativas. A nossa abordagem principal para o encaixe de linhas adaptou este método às características dos contornos sugestivos, mas outras abordagens que encaixavam curvas a conjuntos de pontos foram também consideradas, as quais apresentaremos na Secção 5.3.

2.1 A Superfície MPU

Uma superfície implícita [Bloomenthal97] é definida através da função potencial $f(\mathbf{x})$ como o conjunto de pontos $\mathbf{x}=(x,y,z)$ que respeitam a condição:

$$f(\mathbf{x}) = iso \qquad (1)$$

onde *iso* é um valor constante. No nosso caso, onde $iso=0$, os valores positivo ou negativo de $f(\mathbf{x})$ indicam que $\mathbf{x}$ está dentro ou fora do volume do objecto, respectivamente. Estas representações, por natureza, proporcionam uma definição compacta e flexível para superfícies muito complexas e fazem com que as operações de *blending* sejam fáceis de aplicar.

A MPU é um dos vários tipos de superfícies implícitas que se destaca por proporcionar um método para a construção eficiente de modelos a partir de nuvens densas de pontos, obtidas por *sampling* da superfície de objectos complexos. A sua estrutura é composta por três elementos: uma *octree* de células espaciais cúbicas que cobrem o objecto; funções quadráticas que aproximam a forma local em cada célula e funções de peso que fazem o *blend* das funções locais e, como tal, oferecem-nos uma aproximação controlada pela precisão de uma superfície implícita de forma eficiente. A construção da MPU é orientada pela subdivisão da estrutura *octree*, onde as células tornam-se mais pequenas e numerosas em áreas onde as posições das partículas e normais sugerem uma maior curvatura. Estendemos este modelo para suportar operações de edição interactiva da forma. A enumeração espacial permite-nos marcar as células locais da MPU que estão envolvidas na operação de edição e reconstruir a MPU localmente a partir de um novo conjunto de pontos de controlo, o que beneficia o desempenho do sistema [Araujo04].

3. ABORDAGEM

Os nossos métodos seguem uma abordagem de alto nível semelhante à de Foster et al. [Foster05], na qual partículas são distribuídas ao longo da superfície. Silhuetas, contornos sugestivos e arestas que evidenciam descontinuidades superficiais são então extraídas das partículas relevantes. No entanto, enquanto Foster et al. mantinham o sistema de partículas num estado de simulação constante (as posições das partículas estão sempre a ser actualizadas num ciclo contínuo), nós apenas deixamos que a simulação se desenvolva por um número máximo de iterações que são suficientes para a uma distribuição aceitável de pontos ao longo da superfície. Isto resulta num tempo de simulação limitado, ao fim do qual muitos recursos computacionais são libertados para os algoritmos de extracção de linhas. Também oferecemos ao utilizador a possibilidade de gravar o estado do sistema de partículas para um ficheiro, permitindo a sua reutilização em cada carregamento do mesmo objecto na aplicação, sem ser necessário repetir a simulação novamente.

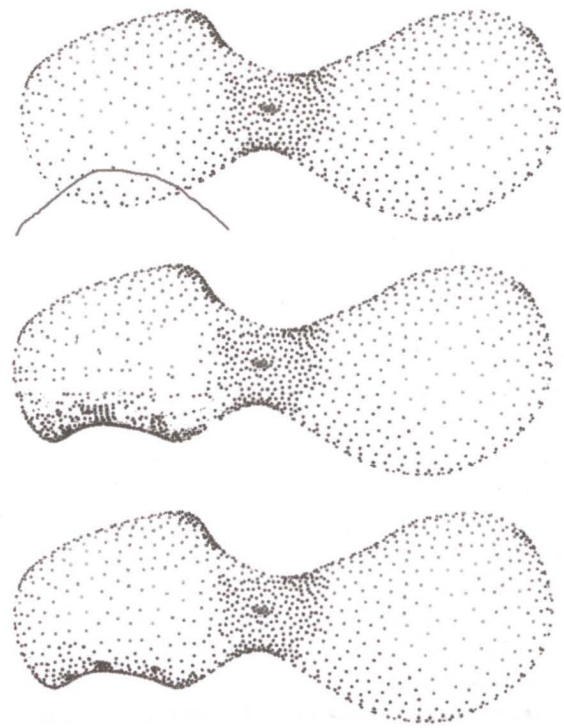

Figura 2 - Edição da superfície. Cima: o utilizador define um corte CSG sobre um objecto simples, usando a linha azul. Centro: a superfície MPU é localmente alterada e são geradas novas partículas nessa área. Baixo: as partículas são redistribuidas por uma simulação local, que dura alguns segundos.

As silhuetas e os contornos sugestivos são linhas dependentes do ponto de vista, por isso regeneramo-las sempre que há uma alteração das condições de visualização ou quando o sistema de partículas está em simulação. As arestas por seu lado, que representam descontinuidades geométricas, são independentes do ponto de vista, por isso apenas as geramos uma vez depois de algumas iterações do sistema de partículas e depois aplicamos as trans-

formações 3D apropriadas sempre que as condições de visualização se alteram.

Foster et al. tratam a representação implícita como uma *black-box*, i.e. os seus métodos apenas consultam a função potencial para extrair o seu valor e gradiente, o que faz com que a sua abordagem seja compatível com quase qualquer tipo de superfície implícita. Apesar da nossa abordagem também só necessitar da mesma informação matemática, ela explora a informação espacial fornecida pela *octree*, o que nos cria uma dependência relativamente às estruturas de dados associadas com a MPU. Isto inclui usar a *octree* para determinações de proximidade de linhas e pontos, em vez de se usar uma grelha espacial comum para o efeito.

O nosso sistema suporta operações de modelação através de uma interface caligráfica [Araujo04]. Depois de uma operação de edição, marcamos as células *octree* afectadas e começamos uma nova simulação do sistema de partículas confinada a essas células. Isto faz com que a regeneração das partículas seja muito mais eficiente, porque apenas afecta os pontos locais. Isto é ilustrado na Figura 2, onde podemos ver claramente a redistribuição de partículas depois de uma edição da forma da superfície. Uma vez que as arestas são elementos independentes da vista, elas também são regeneradas localmente. Apesar do nosso trabalho se ter preocupado principalmente com o *rendering* de superfícies implícitas, era importante assegurar que a modelação da superfície continuava a ser possível e eficiente com as nossas técnicas, uma vez que é uma das principais motivações para se usarem estas representações em detrimento de outras, especialmente no caso da MPU onde a reconstrução local é directamente suportada em operações CSG e de *blending*.

Finalmente, para efectuar remoção de linhas ocultas utilizámos uma abordagem baseada em *surfels*, como em [Foster05]. *Surfels* são elipses orientadas ou círculos que são usados em *point-based rendering* de superfícies [Pfister00]. A nossa abordagem utiliza texturas circulares dentro de polígonos quadriláteros posicionados ligeiramente atrás de cada partícula usando-se a normal da superfície para orientação. Apesar deste método não garantir uma oclusão total para todos os casos, é muito eficaz desde que exista uma boa distribuição de partículas ao longo da superfície.

4. O SISTEMA DE PARTÍCULAS

O nosso sistema de partículas constitui um desenvolvimento sobre a abordagem de Foster et al. [Foster05], a qual é baseada no modelo de Witkin e Heckbert [Witkin94]. Enquanto os seus métodos posicionam partículas em posições iniciais aleatórias, nós utilizamos os elementos da MPU para obter uma melhor distribuição inicial. O nosso objectivo é atingir maiores concentrações de pontos em áreas de elevada curvatura. Uma vez que as células *octree* são mais pequenas e numerosas nessas áreas, criamos um número k de partículas aleatórias dentro de cada célula, onde k é um valor fixo (usualmente $k = 1$), e obtemos uma distribuição inicial que já está próxima do nosso objectivo final, o que poupa um número considerável de iterações. Obtemos assim um conjunto de n partículas, $P_i, i \in [1,n]$, escolhidos do conjunto de pontos de controlo da MPU que já se encontram sobre a superfície. Cada uma destas partículas é registada na célula correspondente da *octree*, permitindo a utilização de algoritmos rápidos para determinação de proximidades de pontos, através de pesquisa de células adjacentes.

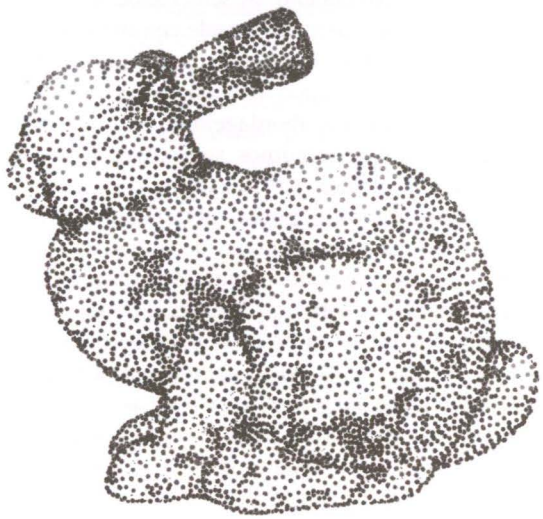

Figura 3 - 10000 partículas espalhadas ao longo da superfície do modelo *Bunny*, depois de 20 iterações da simulação do sistema de partículas. Os pontos vermelhos correspondem a células *octree* de elevada profundidade, pontos verdes a baixa profundidade.

O processo de simulação do sistema de partículas é composto por um conjunto de iterações no qual, em cada passo, cada partícula sofre forças de atracção e repulsão, movendo-se de acordo com estas. O cálculo destas forças, bem como do movimento, é efectuado da mesma forma que em [Foster05], com a diferença de que computamos o factor de distribuição da repulsão δ_i de uma forma diferente. Este valor influencia directamente a força repulsiva e controla a densidade local das partículas. Foster et al. calculam-no utilizando valores de Hessiana extraídos da função potencial, para determinar a curvatura média local. Isto introduz um peso computacional significativo; nós evitamos isto por completo ao usar a informação de profundidade da célula *octree* na MPU. A profundidade indica o nível de subdivisão celular e como tal é uma boa heurística para determinar a complexidade local da superfície, o que nos dá uma fórmula mais simples para δ_i:

$$\delta_i = weight\left(\frac{d_i - d_{\min}}{d_{\max} - d_{\min}}\right) \qquad (2)$$

onde d_i é a profundidade da célula para a partícula P_i e $d_{\max}$ e $d_{\min}$ são as profundidades máxima e mínima onde existem partículas no modelo (estas são determinadas durante o passo inicial de geração de partículas). A função *weight* é uma *B-Spline* quadrática que estabelece um valor mais elevado para δ_i se a partícula está numa

das células com menor profundidade (regiões menos encurvadas da superfície) e vice versa. Com esta técnica conseguimos obter bons resultados de forma fácil e rápida, nos quais as partículas ficam bem distribuídas ao longo da superfície e reflectem a curvatura com menos passos e um custo inferior por partícula (Figuras 1 e 3).

5. CONTORNOS SUGESTIVOS

Contornos sugestivos [DeCarlo03] são linhas formadas por pontos que, para uma determinada posição da câmara, correspondem a pontos que formam uma silhueta que surge numa posição de visualização próxima (de distância radial inferior a 90 graus), não correspondendo essa silhueta a nenhuma outra de um ponto de visualização mais próximo (radialmente). O que esta definição nos diz informalmente é que se observarmos um contorno sugestivo a partir de uma determinada posição de visualização, uma silhueta aparecerá na mesma área com uma pequena variação na posição da câmara.

Os processos descritos por DeCarlo et al. para a extracção de contornos sugestivos no espaço objecto utilizam uma definição que depende da derivada direccional da curvatura radial da superfície. Esta tem de ser calculada a partir de valores principais de curvatura, que são normalmente computados através da Hessiana da função potencial $f(\mathbf{x})$ numa superfície implícita. É difícil encontrar representações implícitas que nos forneçam valores de Hessiana rápidos e numericamente estáveis e a MPU não é excepção. Torna-se particularmente difícil obter estes valores perto de descontinuidades, onde é muito provável encontrar-se um contorno sugestivo, porque a função potencial da MPU não é bem comportada. Como tal, a nossa abordagem utiliza outra definição para contornos sugestivos mencionada por DeCarlo et al. no seu algoritmo do espaço imagem, nomeadamente o conjunto de mínimos do produto interno entre o vector de visualização e a normal da superfície na direcção do vector de visualização projectado. Esta definição requer apenas a avaliação de valores de gradiente $\nabla f(\mathbf{x})$ para se obter a normal da superfície.

O nosso processo de extracção de contornos sugestivos começa com a identificação de partículas em áreas de contornos sugestivos. Depois criamos *clusters* de conjuntos dessas partículas próximas umas das outras e aplicamos algoritmos de encaixe de linhas a cada *cluster* para obter os contornos sugestivos. Nas próximas três secções descrevemos estes métodos em detalhe.

5.1 Identificação de Partículas

O vector de visualização normalizado $\mathbf{v}$ para uma posição de superfície $\mathbf{x}$ e posição de visualização $\mathbf{c}$ é definido por:

$$\mathbf{v} = \frac{(\mathbf{x}-\mathbf{c})}{\|(\mathbf{x}-\mathbf{c})\|} \tag{3}$$

Consideremos que $\mathbf{w}$ é a projecção de $\mathbf{v}$ no plano tangente à superfície em $\mathbf{x}$ com normal $\mathbf{n}$, obtida através do gradiente normalizado $\|\nabla f(\mathbf{x})\|$. Os contornos sugestivos podem ser definidos como o conjunto de mínimos de $(\mathbf{n}\cdot\mathbf{v})$ na direcção de $\mathbf{w}$ [DeCarlo03].

Usando esta definição, identificamos partículas que se encontram próximas de um contorno sugestivo estimando se $(\mathbf{n}\cdot\mathbf{v})$ encontra um mínimo local nas suas posições. Consideremos dois pontos $\mathbf{x}_i^+$ e $\mathbf{x}_i^-$ definidos por:

$$\mathbf{x}_i^+ = \mathbf{x}_i + k_d\mathbf{w} \tag{4}$$

$$\mathbf{x}_i^- = \mathbf{x}_i - k_d\mathbf{w} \tag{5}$$

Onde $\mathbf{x}_i$ é a posição da partícula P_i e k_d depende da escala e controla a distância entre $\mathbf{x}_i^+$ e $\mathbf{x}_i^-$ (para os modelos testados usámos $k_d = 0.5$, Figura 4). Consideremos também que dp_i, dp_i^+ e dp_i^- são os valores $(\mathbf{n}\cdot\mathbf{v})$ em $\mathbf{x}_i$, $\mathbf{x}_i^+$ e $\mathbf{x}_i^-$ respectivamente (obtidos pela consulta do gradiente $\nabla f(\mathbf{x})$). Se as seguintes inequações se verificarem:

$$dp_i < dp_i^+ \tag{6}$$

$$dp_i < dp_i^- \tag{7}$$

podemos concluir que existe um mínimo local entre $\mathbf{x}_i^+$ e $\mathbf{x}_i^-$ e, se k_d for suficientemente pequeno, $\mathbf{x}_i$ é uma boa estimativa desse mínimo. Adicionalmente, aplicamos o *threshold* de estabilidade mencionado em [DeCarlo03], que exclui áreas onde o vector de visualização está quase normal à superfície:

$$\theta_{sc} < \cos^{-1}(dp_i) \tag{8}$$

onde θ_{sc} é um ângulo definido pelo utilizador. Se as inequações (6), (7) e (8) se verificarem, consideramos que P_i pertence a um contorno sugestivo.

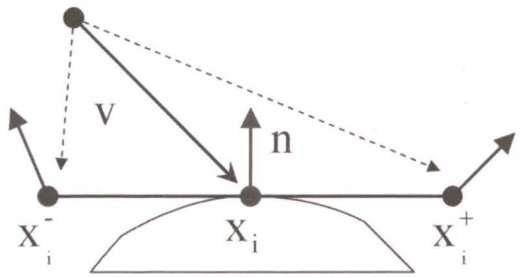

Figura 4 - As posições $\mathbf{x}_i^+$ e $\mathbf{x}_i^-$ são calculadas sobre a superfície plana definida por n e $\mathbf{x}_i$, na direcção projectada de v. Os vectores normais azuis são computados pela avaliação do gradiente da função potencial em $\mathbf{x}_i^+$ e $\mathbf{x}_i^-$.

Utilizamos os mínimos de $(\mathbf{n}\cdot\mathbf{v})$ para identificar pontos de contornos sugestivos no espaço objecto, em vez dos zeros de curvatura radial usados por DeCarlo et al., uma vez que estes últimos parecem não funcionar fiavelmente em MPUs, porque são descontínuos ao longo das funções de *blending*. Esta é uma nova aplicação desta definição para contornos sugestivos e também permite a sua extracção de superfícies implícitas sem se recorrer a uma estrutura auxiliar como uma malha poligonal.

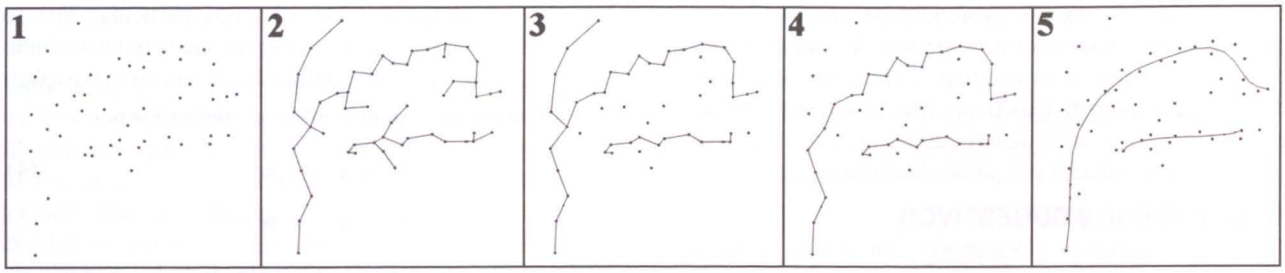

Figura 5 - Extracção de contornos sugestivos. 1: partículas de contornos sugestivos. 2: dois *clusters* são identificados e as MSTs são construídas. 3: abordagem do corte da MST. 4: abordagem do caminho mais longo na MST. 5: através da abordagem de caminho mais longo e usando *smoothing* de linhas, obtemos os contornos sugestivos.

5.2 Clustering de Pontos

Depois de identificarmos partículas de contornos sugestivos, agrupamos *clusters* de pontos que estão próximos uns dos outros na superfície. Seguidamente, a nossa abordagem constrói um grafo dos k vizinhos mais próximos para cada *cluster*. Começamos por escolher uma partícula arbitrária P e procurar vizinhos no espaço 3D dentro de um raio fixo. Este raio é multiplicado pelo valor δ_i da célula *octree* respectiva de forma a confinar esta procura a um espaço mais limitado em áreas de maior curvatura. A partir destas partículas circundantes, escolhemos as k mais próximas (normalmente usamos k = 3) e esses tornam-se os vizinhos de P no grafo. Para cada vizinho repetimos esta procura de pontos próximos, ignorando sempre as partículas já inspeccionadas. O grafo do *cluster* fica completo quando a procura não encontra novos vizinhos. Criamos *clusters* adicionais através da aplicação deste processo a outras partículas que não foram inspeccionadas anteriormente. Devido à natureza dinâmica do raio de procura, é possível ter partículas de um *cluster* a encontrar partículas próximas de outros *clusters*. Nestas situações, os dois *clusters* são unidos.

5.3 Encaixe de Linhas

Depois do *clustering* de partículas, finalmente aplicamos um algoritmo de encaixe de linhas para extrair linhas que representam a topologia de cada *cluster*. A nossa abordagem inspirou-se nos métodos apresentados em [Gumhold01] e começa por criar uma *minimum-spanning-tree* (MST) do grafo do *cluster*, usando-se o algoritmo de Kruskal optimizado. Este processo minimiza as distâncias geométricas entre pontos.

Uma vez que a MST obtida pode ter um elevado número de ramos curtos, temos de aplicar um processo de simplificação para extrair as linhas dos contornos sugestivos. Para tal, temos duas opções. A primeira consiste no cálculo do caminho mais longo na MST e utiliza os arcos correspondentes para formar o contorno sugestivo. Para encontrar este caminho, escolhemos uma partícula arbitrária P e encontramos partícula mais longínqua P_f através de uma procura *breadth-first*. Depois encontramos a partícula P_f' mais longe de P_f. O caminho mais longo é o que liga estas duas partículas. A segunda abordagem consiste em remover os ramos curtos. Para efectuar este passo, percorrermos todas as partículas na MST que têm mais de duas ligações a outras partículas (nós de intersecção) e analisamos cada ramo incidente. Se o ramo liga a outro nó de intersecção, este mantém-se intacto. Se o ramo acaba num nó folha da árvore e é composto por menos de k_{trim} (normalmente três) partículas, é descartado. Em qualquer uma das abordagens, as linhas obtidas sofrem um passo final de *smoothing*.

Outras Abordagens

Também foram tentados outros métodos para encaixe de linhas, mas a abordagem da MST provou ser bastante superior a esses. Um dos métodos usa um processo de expansão de linhas semelhante à expansão do *cluster* descrito na Secção 5.2. Para cada *cluster* estimamos um ponto extremo para o contorno sugestivo através de um encaixe de uma recta 2D (através do método dos mínimos quadrados). Depois projecta-se cada partícula sobre essa linha recta e escolhemos um dos pontos das extremidades da projecção. Essa partícula torna-se no primeiro ponto da *polyline* do contorno sugestivo e extraímos um novo ponto da média das posições das partículas vizinhas no grafo. Aplicamos este processo recursivamente ao vizinhos dessas partículas, sem nunca repetir uma partícula, até que todas as partículas do *cluster* tenham sido exploradas. Apesar deste processo produzir resultados aceitáveis, é muito dependente da estimação do ponto inicial, que não funciona bem para certas topologias do *cluster*. Quando o ponto extremo é incorrectamente determinado, o contorno sugestivo produzido poderá até sair completamente para fora do *cluster*.

Noutra abordagem, tentámos também encaixar uma curva polinomial 2D pelos mínimos quadrados ao *cluster*. Existiram porém dois problemas maiores com esta abordagem. Primeiro, é difícil estimar com precisão o grau do polinómio adequado a partir da dimensão e topologia do *cluster*, o que muitas vezes resultava na produção de curvas instáveis. Adicionalmente, uma vez que este processo opera no espaço 2D (uma vez que se torna muito complexo de efectuar em 3D), perderíamos a capacidade de RLO providenciada pelas *surfels*, que usa texturas 3D para cobrir linhas que são construídas no espaço objecto. Para resolver estes problemas poderíamos ter usado *splines* 3D, mas teríamos de enfrentar o mesmo problema da abordagem anterior, uma vez que seria necessário determinar os pontos extremos do *cluster*. Além do mais, a MST provou ser um método bastante rápido e eficaz e

com menor peso computacional do que uma interpolação sobre uma *spline* 3D.

6. SILHUETAS E ARESTAS

Como no sistema de partículas, os nossos métodos para extracção de silhuetas e arestas são semelhantes a [Foster05] com quatro melhoramentos de destaque. No que se segue, providenciamos contexto suficiente para explicar as técnicas base e destacamos as nossas alterações. Para determinar silhuetas, começamos por identificar partículas P_i com posição $\mathbf{x}_i$ que verificam a seguinte inequação:

$$|\mathbf{v}_i \cdot \mathbf{n}_i| < k_c \qquad (9)$$

onde k_c é controlado pelo utilizador (usamos $k_c = 0.05$ para a maioria dos modelos apresentados), $\mathbf{v}_i$ é o vector de visualização e $\mathbf{n}_i$ é a normal da superfície em $\mathbf{x}_i$. Esta inequação estabelece uma área da superfície onde se considera que as partículas estão próximas da silhueta real (chamemos-lhe área de silhueta). A partir de cada partícula identificada, construímos uma *polyline* da silhueta iterativamente através de um processo de integração numérica. Usamos o método preditor/corrector apresentado por Foster et al., efectuando dois movimentos em cada iteração. O primeiro segue uma direcção de silhueta estimada com uma dimensão de passo fixa, enquanto o segundo resulta de dois vectores de correcção computados para assegurar que a linha se mantém próxima da silhueta e na superfície. A nossa abordagem utiliza uma dimensão de passo dinâmica, multiplicando-a pelo valor δ_i obtido da célula *octree* da MPU (descrita na Secção 4) onde ocorre o movimento. Isto proporciona uma melhor precisão da *polyline* da silhueta em áreas de maior curvatura, porque os respectivos segmentos da linha se tornam mais curtos, enquanto melhora o desempenho em áreas de menor curvatura, onde os segmentos se tornam mais longos (*Melhoramento 1*). Este processo pára se (1) detectamos que a silhueta forma um ciclo, (2) sempre que se detecta a proximidade de outra silhueta ou (3) quando a seguinte inequação não é verificada:

$$|\mathbf{v} \cdot \mathbf{n}| < k_c \cdot k_e \qquad (10)$$

Esta inequação é semelhante a (9) com a excepção do parâmetro k_e, que expande a área de silhueta onde a curva se pode desenvolver. Com a presença deste parâmetro, podemos estabelecer um valor k_c menor para limitar o numero de partículas de silhueta identificadas e permitir que o processo de construção da linha tenha uma maior área de expansão (usando $k_e = 2$ ou $k_e = 3$). Uma vez que é comum ter partículas que originam a mesma silhueta, esta técnica melhora o desempenho geral ao mesmo tempo que mantém a qualidade e continuidade das curvas extraídas (*Melhoramento 2*). Depois de termos computado todas as silhuetas, efectuamos uma analise 2D para encadear pares de *polylines* que estão a uma certa distancia uma da outra e seguem orientações semelhantes.

Para a extracção de arestas, identificamos partículas que se encontram nas áreas relevantes da superfície e construímos linhas a partir dessas posições através de métodos de integração numérica. Começamos por descobrir pares de partículas (*straddle points*) perto uma da outra, que apresentam uma grande diferença entre as suas normais da superfície. Foster et al. efectuam este passo na fase de simulação do sistema de partículas, comparando a normal em cada partícula antes e depois de um movimento. Formalmente, para uma certa partícula P_i com normal inicial $\mathbf{n}_i$ e normal $\mathbf{n}_i'$ depois do movimento, consideramos que as duas posições respectivas $\mathbf{x}_i$ e $\mathbf{x}_i'$ são *straddle points* se:

$$\hat{a}ngulo(\mathbf{n}_i, \mathbf{n}_i') > k_a \qquad (11)$$

onde $\hat{a}ngulo(\mathbf{v}_1, \mathbf{v}_2)$ retorna o ângulo entre dois vectores e k_a é o ângulo *threshold* (normalmente usamos $k_a = 0.15$). Apesar deste método ser usualmente eficaz e extrair suficientes *straddle points* para um bom resultado visual, existem situações onde poderá ser incapaz de encontrar arestas devido às nossas optimizações no sistema de partículas. De facto, as partículas que são colocadas inicialmente com densidades quase óptimas ao longo da superfície (ver Secção 4) podem levar a uma fase de simulação onde ocorrem variações muito pequenas nas suas posições. Apesar disto nos dar um bom desempenho, pode também limitar as oportunidades para identificar *straddle points*. Como tal, concebemos um método alternativo para resolver este problema que é aplicado quando o método de Foster et al. falha. Consiste em procurar pontos próximos dentro de um raio fixo à volta de cada partícula e comparar as normais para detectar pares que respeitam a Inequação (11) (*Melhoramento 3*).

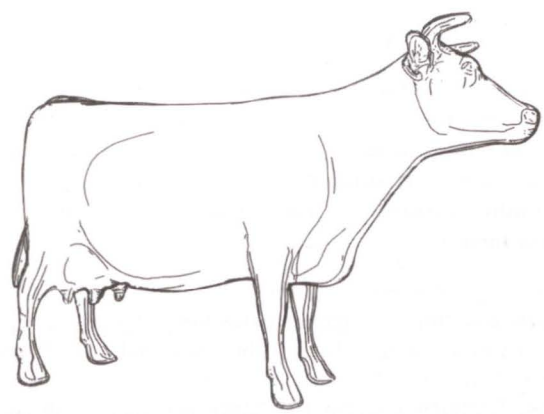

Figura 6 – Silhuetas, arestas e contornos sugestivos no modelo *Cow*.

Depois de identificarmos *straddle points*, movemo-los ao longo das respectivas faces separadas pela aresta, enquanto tentamos manter uma direcção paralela a esta. Enquanto isto é efectuado, estimamos pontos intermédios que se encontram próximos da aresta, construindo a respectiva *polyline*. Como no caso das silhuetas, utilizamos o valor δ_i da célula *octree* da MPU para ajustar dinamicamente

o passo do movimento à curvatura local (*Melhoramento 4*). O processo pára quando a Inequação (11) deixa de ser verificada ou quando outra aresta é detectada na zona circundante dos *straddle points*.

7. RESULTADOS E DISCUSSÃO

As nossas técnicas para a extracção de silhuetas e arestas provaram ser eficazes desde que a superfície MPU forneça precisão suficiente. Normalmente esta depende do número e densidade de pontos no *dataset* do objecto. Também somos capazes de produzir contornos sugestivos para os objectos usando as nossas técnicas de identificação de partículas e encaixe de linhas.

Para as duas abordagens que usámos para encaixar linhas sobre a MST dos contornos sugestivos (Secção 5.3), a abordagem do caminho mais longo garante-nos que se extrai uma única linha contínua para cada *cluster*, o que a torna geralmente na melhor opção para evitar artefactos causados pela ramificação da MST. No entanto, quando temos contornos sugestivos que quase se intersectam, a abordagem de encurtar a MST consegue distinguir algumas linhas que se intersectam quando os respectivos *clusters* são unidos. Esta é de facto uma das maiores dificuldades a ultrapassar em todo o processo: existe sempre uma certa probabilidade de dois ou mais contornos sugestivos distintos serem englobados pelo mesmo *cluster* de partículas por causa da sua proximidade. Como tal, damos a hipótese ao utilizador de escolher o método desejado para cada modelo.

Modelo	Pontos Dataset	Sistema Partículas	Sem Cont. Sugestivos	Com Cont. Sugestivos
Bunny	69451	10000	3.4	1.9
Cow	92864	8334	3.4	1.8
Igea	268686	13665	3.7	1.4
Armadillo	345944	28360	1.6	0.5
Dragon	480076	26984	1.8	0.6
David	827181	59994	0.9	0.2

Tabela 1 Resultados de desempenho do nosso sistema. Cada linha indica o modelo, número de pontos no *dataset*, número de pontos no sistema de partículas, *frame-rate* com silhuetas e arestas (em *frames* por segundo) e *frame-rate* com contornos sugestivos adicionalmente.

A Tabela 1 apresenta os resultados de desempenho obtidos para os objectos apresentados nas Figuras 1, 3, 6, 7, 8 e 9. Estes resultados foram obtidos usando um Pentium IV 3.6Ghz com 2 gigabytes de RAM e uma placa de vídeo NVIDIA Quadro FX 3400, correndo o Windows XP SP2. Todas as *frame-rates* foram obtidas depois da simulação do sistema de partículas se ter completado (normalmente, esta inclui 20 iterações).

O número de partículas no sistema é sempre determinado pela estrutura da superfície implícita MPU, uma vez que colocamos uma partícula por cada célula da *octree*. Esta estratégia revelou ser muito eficaz para todos os modelos testados e também se torna automática, sem ser preciso o utilizador ajustar o número de partículas necessárias para uma cobertura correcta da superfície. Tal como os resultados sugerem, o número de partículas não é proporcional ao número de pontos no *dataset*, porque também é influenciado pela topologia da superfície.

Os resultados de desempenho mostram que é possível obter *frame-rates* interactivas com objectos de complexidade média, até quando extraímos contornos sugestivos. No entanto, a dimensão do sistema de partículas e a complexidade do modelo normalmente afectam o desempenho, uma vez que influenciam o número de avaliações da função implícita que são efectuadas e o peso computacional associado. Isto também explica porque a extracção de contornos sugestivos tem uma grande influência na *frame-rate*, uma vez que implica um grande aumento do número de avaliações. Apesar disto, a nossa abordagem compara-se favoravelmente com [Foster05] no sentido de que, para uma *frame-rate* semelhante, somos normalmente capazes de desenhar modelos com seis vezes mais partículas, ao mesmo tempo que incluímos contornos sugestivos, que não são suportados pela sua abordagem. Também é importante mencionar que, apesar das abordagens baseadas em superfícies implícitas (como a nossa e a de Foster et al.) serem actualmente menos eficientes que as baseadas em malhas, é importante melhorar constantemente estes métodos de *rendering*, porque a flexibilidade descritiva fornecida por uma superfície implícita é vital em modelação de superfícies e não pode ser obtida numa malha poligonal.

Em termos de resultados visuais, verificamos que, para densidades superiores de pontos no *dataset*, a MPU consegue representar a superfície com maior precisão e o processo global de extracção de linhas produz melhores resultados. Isto explica porque os modelos *David's Head* e *Phlegmatic Dragon* parecem produzir as imagens mais impressionantes (Figuras 7 e 8).

Existem no entanto algumas notas importantes a ter em conta em relação às limitações do nosso sistema. Uma vez que muitos dos nossos métodos são baseados em Foster et al., herdámos muita da dependência de ajuste de parâmetros pelo utilizador, nomeadamente em termos de ajustar a repulsão entre partículas e alguns parâmetros da extracção de silhuetas à escala e topologia do modelo 3D. Outra limitação é a necessidade de nuvens de pontos quase completas para se obterem bons resultados. Estas são normalmente obtidas a partir do *scan* 3D de objectos reais e é comum apenas algumas partes da superfície serem amostradas no *dataset*. Apesar do sistema de partículas ser capaz de lidar com buracos ocasionas na representação MPU, não é capaz de distribuir pontos eficazmente sobre uma superfície incompleta. A nossa técnica também não garante que todas as silhuetas, contornos sugestivos e arestas sejam extraídos de cada superfície, uma vez que a identificação destas linhas, para cada posição de visualização, depende da existência de partículas colocadas estrategicamente junto delas. Esta dependência é ainda mais importante para contornos sugestivos, uma vez que são gerados a partir de *clustering* de partículas.

Isto resulta em curvas instáveis, o que se torna visível quando a câmara se move.

8. CONCLUSÕES

Apresentámos técnicas para extrair silhuetas, contornos sugestivos e arestas directamente de superfícies implícitas MPU, que são capazes de representar superfícies a partir de grandes conjuntos de pontos. Os nossos métodos beneficiam da informação espacial oferecida pela estrutura MPU, permitindo-nos posicionar automaticamente partículas ao longo da superfície e ajustar o sistema de partículas e a extracção de linhas à forma local. Esta representação implícita também oferece suporte à modelação do objecto, ao permitir-nos re-gerar elementos visuais afectados por edições. Melhorámos técnicas *state-of-the-art* para desenho de superfícies implícitas baseado em linhas e apresentámos um método para extracção de contornos sugestivos a partir desta representação.

A nossas técnicas geram desenhos expressivos e precisos de objectos muito complexos, especialmente nos *datasets* mais detalhados e completos. Os nossos resultados apresentam melhores níveis de desempenho em comparação a sistemas semelhantes para o mesmo número de partículas. Continuam a existir algumas áreas para trabalho futuro. Entre elas, poderemos descobrir novas formas de definir a MPU junto das arestas da superfície para conseguirmos desenhar linhas mais precisas e contínuas. Também existe algum trabalho que pode ser feito para melhorar a coerência temporal dos contornos sugestivos.

9. AGRADECIMENTOS

Os modelos são cortesia de: Digital Michelangelo Project 3D Model Repository (*David's Head*), Stanford 3D Scanning Repository (*Bunny, Cow* e *Armadillo*), Cyberware (*Igea*) e UTIA, Academia de Ciências da República Checa, e CGG, Universidade Técnica em Praga (*Phlegmatic Dragon*).

10. REFERÊNCIAS

[Araujo04] B. R. de Araujo e J. A. P. Jorge. Curvature dependent polygonization of implicit surfaces. *Proc. of SIB-GRAPI'04*, pp. 266–273, 2004.

[Barla05] P. Barla, J. Thollot, e F. Sillion. Geometric clustering for line drawing simplification. *Proc.of the Eurographics Symposium on Rendering*, 2005.

[Bloomenthal97] J. Bloomenthal. *Introduction to Implicit Surfaces, First Edition (The Morgan Kaufmann Series in Computer Graphics)*. Agosto 1997.

[Bremer98] D. J. Bremer e J. F. Hughes. Rapid approximate silhouette rendering of implicit surfaces. *Proc. of Implicit Surfaces '98*, pp. 155–164, Junho 1998.

[Burns05] M. Burns, J. Klawe, S. Rusinkiewicz, A. Finkelstein, e D. DeCarlo. Line drawings from volume data. *ACM Trans. Graph.*, 24(3):512–518, 2005.

[DeCarlo03] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, e A. Santella. Suggestive contours for conveying shape. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 22(3):848–855, Julho 2003.

[DeCarlo04] D. DeCarlo, A. Finkelstein, e S. Rusinkiewicz. Interactive rendering of suggestive contours with temporal coherence. *NPAR'04*, pp. 15–24, Junho 2004.

[Elber98] G. Elber. Line art illustrations of parametric and implicit forms. *IEEE Trans. Vis. Comp. Graph.*, 4(1):71–81, Jan./Mar. 1998.

[Foster05] K. Foster, P. Jepp, B. Wyvill, M. C. Sousa, C. Galbraith, e J. A. Jorge. Pen-and-ink for BlobTree implicit models. *Computer Graphics Forum*, 24(3):267–276, 2005.

[Gooch99] B. Gooch, P.-P. J. Sloan, A. Gooch, P. Shirley, e R. Riesenfeld. Interactive technical illustration. In *Proc. of the 1999 Symposium on Interactive 3D Graphics*, pp. 31–38, 1999.

[Gumhold01] S. Gumhold, X. Wang, e R. MacLeod. Feature extraction from point clouds. *Proc. of the 10th International Meshing Roundtable*, pp. 293–305, Outubro 2001.

[Isenberg03] T. Isenberg, B. Freudenberg, N. Halper, S. Schlechtweg, e T. Strothotte. A developer's guide to silhouette algorithms for polygonal models. *IEEE Comput. Graph. Appl.*, 23(4):28–37, 2003.

[Jepp06] P. Jepp, B. Wyvill, e M. C. Sousa. Smarticles for sampling and rendering implicit models. Theory *and Practice of Computer Graphics 2006*, pp. 39–46, 2006.

[Levet06] F. Levet, X. Granier, e C. Schlick. Fast sampling of implicit surfaces by particle systems. *Proc. of Shape Modeling International*, page 39, Julho 2006.

[Ohtake03] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, e H.-P. Seidel. Multi-level partition of unity implicits. *ACM Trans. Graph.*, 22(3):463–470, 2003.

[Pfister00] H. Pfister, M. Zwicker, J. van Baar, e M. Gross. Surfels: surface elements as rendering primitives. *Proc. of SIGGRAPH '00*, pp. 335–342, 2000.

[Plantinga03] S. Plantinga e G. Vegter. Contour generators of evolving implicit surfaces. *SM '03: Proc. of the eighth ACM symposium on Solid modeling and applications*, pages 23–32, 2003.

[Schmidt06] R. Schmidt, T. Isenberg, e B. Wyvill. Interactive pen and ink rendering for implicit surfaces. In *ACMSIGGRAPH 2006 Conference Abstracts and Applications*. ACM Press, 2006.

[Sousa03] M. C. Sousa, K. Foster, B. Wyvill, e F. Samavati. Precise ink drawing of 3D models. *Computer Graphics Forum*, 22(3):369–379, 2003.

[vanOverveld04] K. van Overveld e B. Wyvill. Shrinkwrap: An efficient adaptive algorithm for triangulating an iso-surface. *The Visual Computer*, 20(6):362–379, 2004.

[Witkin94] A. P. Witkin e P. S. Heckbert. Using particles to sample and control implicit surfaces. *Proc. of SIGGRAPH '94*, pages 269–277, 1994.

Figura 7 – O modelo *David's Head* desenhado através de silhuetas, arestas e contornos sugestivos.

Figura 8 - O modelo *Phlegmatic Dragon* desenhado através de silhuetas, arestas e contornos sugestivos.

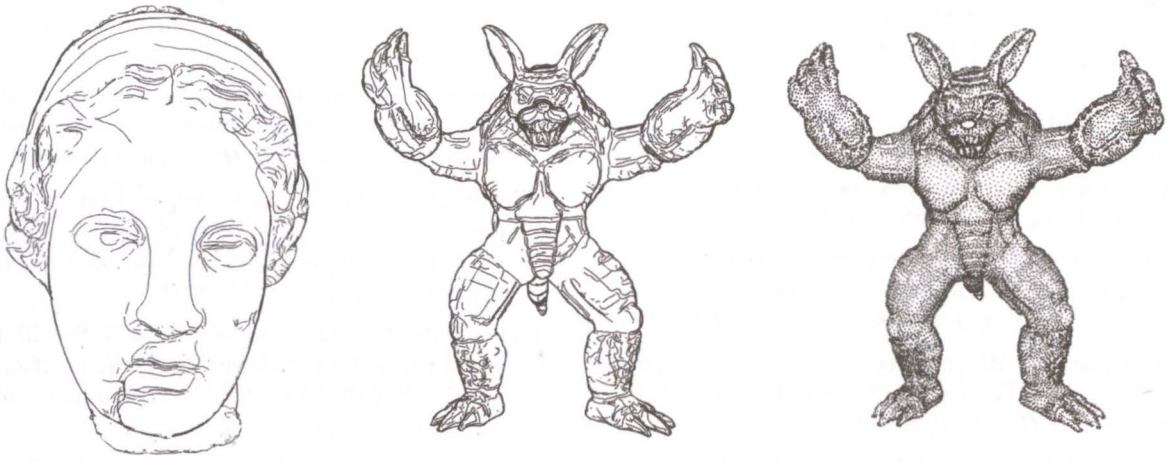

Figura 9 – Os modelos *Igea* e *Armadillo* desenhados através de silhuetas, arestas e contornos sugestivos.