Segmentação de Imagens Médicas via Esboços

Vasco Gervásio Joaquim A. Jorge
Departamento de Engenharia Informática
INESC-ID/IST/Universidade Técnica de Lisboa
R. Alves Redol, 9, 1000-029 Lisboa, Portugal
vmrg@immi.inesc-id.pt, jorgej@acm.org

Resumo

Segmentação de imagens médicas é uma área que dada a sua dificuldade é hoje em dia principalmente resolvida por métodos manuais, pois os resultados dos métodos automáticos estão ainda aquém do desejado. Uma forma de melhorar os resultados é através de uma abordagem intermédia entre estas duas vertentes, onde o utilizador começa por esboçar uma aproximação do contorno inicial, que é depois adaptada automaticamente ao contorno pretendido. As imperfeições do resultado são depois corrigidas através de uma nova técnica que permite progressivamente melhorar o resultado com poucas interacções simples.

Palavras-Chave

Segmentação, Snakes, Correcções Interactivas.

1. INTRODUÇÃO

Segmentação de imagens médicas é um problema difícil de resolver, devido à qualidade das imagens e à necessidade de obter uma grande precisão. Por estes motivos, actualmente, a abordagem mais comum consiste num operador a efectuar a segmentação manualmente, ponto a ponto, o que torna o processo bastante lento.

Assim sendo, neste artigo, propomos um sistema que permite acelerar o processo, mantendo a precisão. Para tal, dos vários algoritmos possíveis escolhemos os modelos de contorno activos (*active contour models*), vulgarmente chamados de *snakes* [Kass 88], pois são uma solução simples e eficiente, que tira partido de uma inicialização manual para permitir segmentar uma determinada região numa imagem, ao contrário de outros algoritmos que segmentam toda a imagem automaticamente [Vu 06, Antonelli 05], mas cuja precisão é difícil de controlar, pois são baseados em valores limite (*thresholds*). No entanto, as *snakes*, como mostraremos mais à frente, também têm as suas limitações na qualidade da segmentação que obtêm, e para as ultrapassar apresentamos uma nova técnica, a que chamámos *oversketching snakes*.

O nosso método é composto pelos seguintes passos: (a) o utilizador esboça uma aproximação inicial do contorno pretendido; (b) é aplicada uma *snake* para ajustar o contorno; (c) o utilizador corrige o contorno resultante desenhando sobre ele, onde seja necessário; (d) cada uma destas correcções locais é ajustada por um algoritmo que adapta os seus parâmetros implicitamente, conforme o tipo de correcção; (e) o utilizador continua a refinar o contorno até estar satisfeito com o resultado (ver **Figura 1**). Este método permite obter uma solução mais rápida que uma solução exclusivamente manual, mais perfeita que só usando *snakes* e mais simples que os métodos "puramente" automáticos, pois não necessita de reparametrizações explícitas.

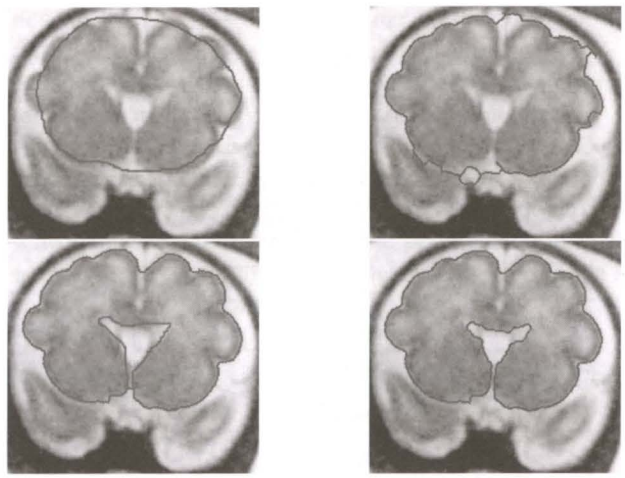

Figura 1. De cima para baixo, da esquerda para a direita tem-se: Esboço inicial; Resultado após a GVF *snake*; Extensão do contorno após três correcções anteriores; Resultado final.

Este artigo está organizado da seguinte forma: na secção 2 apresentamos outras técnicas interactivas que permitem melhorar a segmentação. Na secção 3 descrevemos dois algoritmos de *snakes*, *snakes* normais [Kass 88] e Gradient Vector Flow (GVF) *snakes* [Xu 98]. Na secção 4 explica-

mos, em maior detalhe, a nossa técnica e como ela permite melhorar os outros algoritmos. Na secção 5 analisamos resultados de segmentação e, finalmente, na secção 6 apresentamos as conclusões e trabalho futuro.

2. TRABALHO RELACIONADO

Não existem muitos trabalhos em interacção, além da necessária na inicialização, com *snakes* e algoritmos semelhantes, particularmente no domínio difícil das imagens médicas, [Foo 06] e [Hug 99] são duas dessas referências. Em [Foo 06] são apresentados alguns métodos semi-automáticos de segmentação, a maior parte para 3D, que estão divididos em três classes: **guiados pelo utilizador**, nos quais o utilizador altera constantemente os dados de entrada durante o processo de segmentação; **intervenção do utilizador**, semelhante ao anterior mas as alterações só são feitas quando são detectados dados incorrectos, após as quais o processo de segmentação é então reiniciado ou simplesmente prossegue; **correcção ou manipulação dos resultados finais pelo utilizador**, onde o utilizador só faz correcções depois do processo de segmentação ter acabado. As melhorias interactivas, apresentadas neste artigo, pertencem a esta última classe (ver **Figura 2**).
Em [Hug 99] são mostradas algumas técnicas interactivas para melhorar a segmentação, no caso específico das *snakes*.

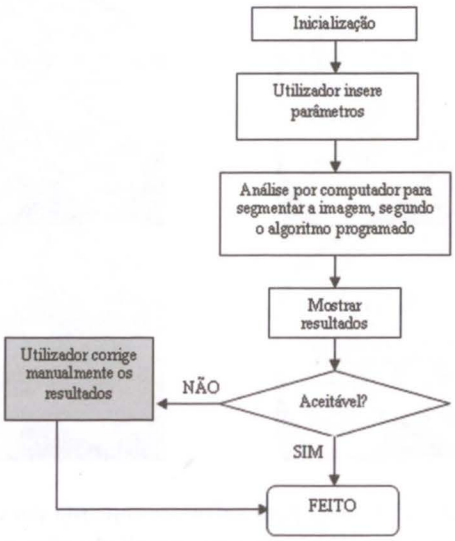

Figura 2. Gráfico genérico de um processo de segmentação com correcção ou manipulação dos resultados finais.

Algumas das técnicas em [Foo 06] só podem ser utilizadas por utilizadores especializados, pois têm de ter um bom conhecimento de como o algoritmo funciona para poderem alterar os seus parâmetros.
Outros algoritmos em [Foo 06], assim como em [Hug 99], permitem a manipulação do contorno, mas só ponto a ponto. Estas técnicas são demoradas, pouco naturais e podem ser complicadas na escolha do ponto a mover.

Por outro lado, noutro domínio, remover um objecto do fundo de imagens a cores, existe um maior interesse em interacção. Em [Li 04] e [Rother 04] a segmentação é feita por variantes do algoritmo Graph Cut ([Boykov 01]) e são apresentadas algumas formas de melhorar o resultado da segmentação. Em [Li 04] o utilizador pode desenhar uma banda e o contorno é aperfeiçoado só nessa banda e em [Rother 04], através do Graph Cut, o utilizador desenha traços, de forma a adicionar ou remover partes ao objecto de interesse.
No entanto, como estes algoritmos extraem um objecto do fundo da imagem baseados em diferenças de cores, não podem ser usados em todas as imagens. Em particular, em imagens médicas, várias vezes não existe uma diferença de cor significativa entre o objecto a ser segmentado e o resto da imagem.

3. SNAKES

Nas duas próximas secções introduzimos dois algoritmos de *s*nakes e mostramos algumas das suas limitações.

3.1. Snakes Normais

Uma *snake* é uma curva, cuja convergência se obtém quando se verifica um equilíbrio entre as suas forças internas e as suas forças externas.
As suas forças internas controlam a tensão da curva, ao aproximar e afastar os vários pontos entre si e a sua rigidez, tornando a *snake* mais ou menos suave. A cada um destes factores estão associados, respectivamente, os pesos α e β.
Por outro lado, as suas forças externas são responsáveis pela atracção pelos pontos de interesse na imagem, que no nosso caso são pontos de contorno. Para mais detalhes de como ambas as forças são calculadas ver [Kass 88].
Na **Figura 3** mostramos alguns exemplos da aplicação do algoritmo.

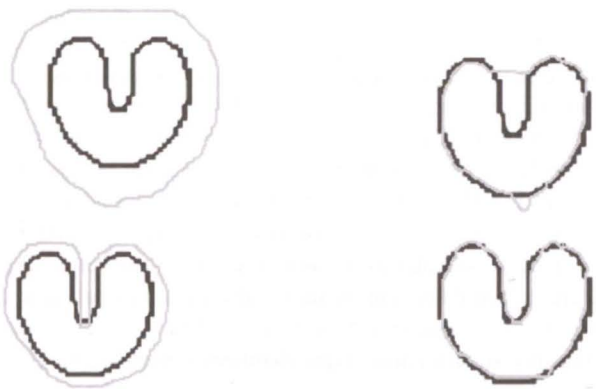

Figura 3. As imagens da esquerda mostram o esboço inicial e as da direita o resultado depois da aplicação das *snakes* normais.

Como se pode ver, este algoritmo apresenta alguns problemas, nomeadamente no alcance das forças externas e na sua direcção em concavidades (ver **Figura 4**), o que

faz com que o esboço inicial tenha de ser desenhado bem próximo do contorno desejado para ser atraído e que haja dificuldades de convergência nas concavidades.

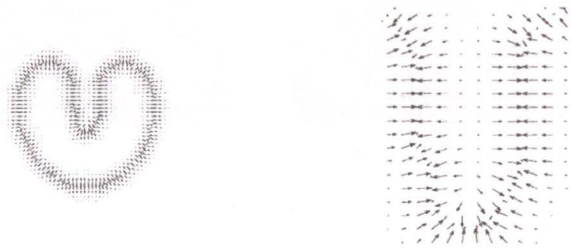

Figura 4. A imagem da esquerda mostra as forças externas da *snake* normal e a da direita uma aproximação dessas forças na concavidade.

3.2. Gradient Vector Flow (GVF) Snakes

A *snake* GVF é um algoritmo que procura resolver as limitações das *snakes* normais, apresentadas na secção anterior. Para tal recorre a uma nova formulação de forças externas, a que os autores chamam de *GVF field*, mantendo as mesmas forças internas.
O *GVF field* é obtido através da soma das suas componentes u e v. Em [Xu 98] mostra-se uma solução numérica para obter os valores destas componentes, através de uma fórmula iterativa.

Com esta alteração os resultados são melhores do que com as *snakes* normais (ver **Figura 5**), pois as forças externas

Figura 5. Aplicação da GVF *snake*.

têm agora um maior alcance e apontam para o interior das concavidades (ver **Figura 6**).

No entanto, os resultados pioram quando os contornos são pouco nítidos ou quando existem vários contornos próximos, pois nesse caso umas vezes é escolhido um contorno e outras vezes outro (ver **Figura 7**).

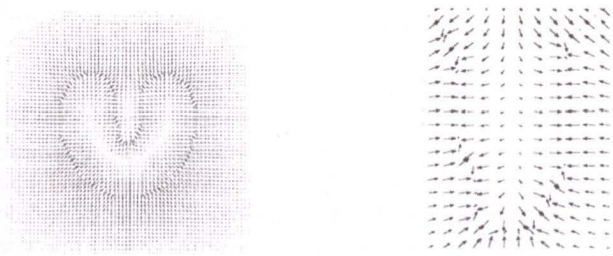

Figura 6. A imagem da esquerda mostra as forças externa da GVF *snake* e a da direita uma aproximação dessas forças na concavidade.

Figura 7. A imagem da esquerda mostra a imagem original, a do meio o esboço inicial e a da direita apresenta o resultado da GVF *snake*, sem mais interacção.

4. OVERSKETCHING SNAKES

Nesta secção apresentamos a nossa abordagem. Começamos por motivar a sua necessidade, de seguida descrevemo-la e por fim analisamo-la.

4.1. Motivação

Os dois principais problemas das *snakes* normais, alcance limitado na captura do contorno e dificuldade em convergir são tratados pelas GVF *snakes*, mas mesmo as *snakes* normais podem superá-los se o utilizador redesenhar o esboço inicial mais cuidadosamente (ver **Figura 3**). No entanto, ao fazê-lo, todo o processo tinha de ser repetido novamente, era mais difícil desenhar o esboço inicial, de forma a ser semelhante ao contorno pretendido, e não são dadas garantias de convergência, pois a adaptação ao contorno pode melhor nalgumas zonas, mas piorar noutras.
Assim sendo, em vez do utilizador voltar a desenhar todo o esboço desde o início, por que não permitir-lhe redesenhar só as partes incorrectas? Para responder a esta questão desenvolvemos uma nova técnica, que permite ao utilizador esboçar sobre contornos anteriormente obtidos de forma a corrigi-los, *oversketching snakes* (ver **Figura 8**).

Figura 8. A imagem à esquerda mostra o esboço correctivo de uma das imagens resultado da Figura 3 e a do meio o resultado. A imagem à direita mostra o resultado final após outra correcção.

4.2. Descrição

A técnica proposta consiste nos seguintes passos: Enquanto a imagem a segmentar é carregada aplica-se um filtro passa-alto, de forma a eliminar os *pixels* de baixo gradiente (ruído), e os contornos são realçados aumentando o valor do gradiente dos *pixels* restantes. Escolhemos estes filtros, pois por serem simples o seu cálculo é eficiente. Para além disso, como os seus efeitos nas imagens são facilmente previsíveis, ao contrário do que acontecia com fil-

tros mais complexos, existe um menor risco de serem eliminados involuntariamente pormenores pouco nítidos, que no caso das imagens médicas podem ser importantes.
Nesta fase são também efectuados cálculos de pré-processamento de forma a obter, para cada *pixel* da imagem, o valor do gradiente, através do operador de Sobel, e as componentes u e v necessárias para o algoritmo da GVF *snake*. Optámos por efectuar estes cálculos nesta altura para toda a imagem, em vez de localmente, em cada esboço efectuado, pois desta forma apesar de gastarmos um pouco mais de tempo a carregar a imagem, tornamos a parte interactiva do processo mais rápida, o que é preferível.

Quando a imagem acaba de ser carregada o utilizador esboça uma aproximação do contorno desejado, que é então ajustada através da GVF *snake* (ver **Figura 9**).
O objectivo deste passo é fornecer ao utilizador uma es-

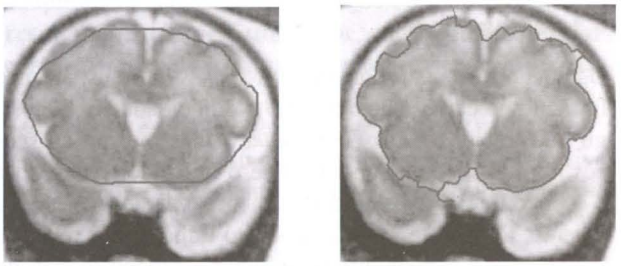

Figura 9. Da esquerda para a direita tem-se: o esboço inicial e o resultado após a GVF *snake*.

timativa do contorno desejado, que sirva como ponto de partida para as possíveis correcções. Assim sendo, nesta fase, o mais importante é que pelo menos partes do esboço inicial sejam atraídas pelo contorno pretendido e como tal é dada ênfase à componente das forças externas. Para além disso, para facilitar que o esboço seja atraído por cantos colocámos o valor de β a 0.

O resultado da GVF *snake* é analisado e se o utilizador está satisfeito com ele o processo acaba.
Caso contrário, ele faz um novo esboço, mais próximo do resultado pretendido, de forma a substituir parte do resultado anterior. Para tal, obtêm-se os pontos do contorno inicial, mais próximos dos pontos limite do novo esboço, e substituem-se o menor número de pontos entre eles pelos pontos do novo esboço (ver **Figura 10**). O resto do

Figura 10. Exemplo de substituição de parte de um contorno por um novo esboço.

contorno inicial mantém-se inalterado.

A estes novos pontos e somente a eles é aplicada uma variante da GVF *snake*, para ajustar a correcção ao contorno (ver **Figura 11**), com três principais modificações:

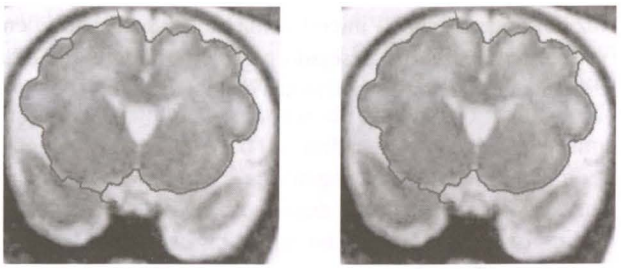

Figura 11. Da esquerda para a direita tem-se: uma correcção (a vermelho) e o resultado após ser adaptada.

- O alcance da GVF *snake* é reduzido, pois como o utilizador está a corrigir parte do resultado o mais provável é que a sua correcção se encontre próxima do contorno pretendido, logo deve-se prevenir a influência de contornos afastados (ver **Figura 12**).

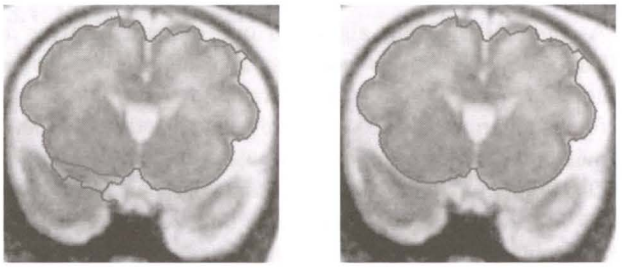

Figura 12. Da esquerda para a direita tem-se: uma correcção (a vermelho) e o resultado após ser adaptada.

- De forma a obter um contorno mais perfeito, mais suave, o valor de β é aumentado (ver **Figura 13**).

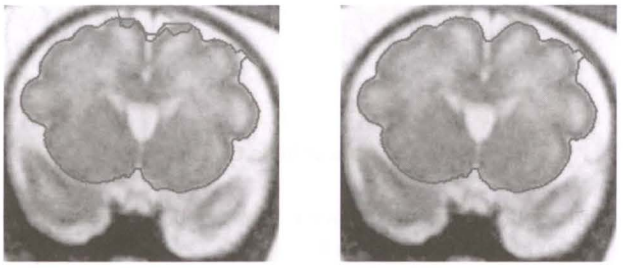

Figura 13. Da esquerda para a direita tem-se: uma correcção (a vermelho) e o resultado após ser adaptada.

- Se o novo esboço é pequeno, o mais certo é que se encontre bem próximo do contorno desejado. Por outro lado, o algoritmo GVF não é muito preciso com muito poucos pontos. Assim sendo, neste caso, os pontos do novo esboço são simplesmente aproximados por uma *spline* (ver **Figura 14**).

Para além das modificações introduzidas, o nosso algoritmo permite ainda não só corrigir o contorno, mas refiná-lo. Deste modo, em vez de ser feito um esboço ini-

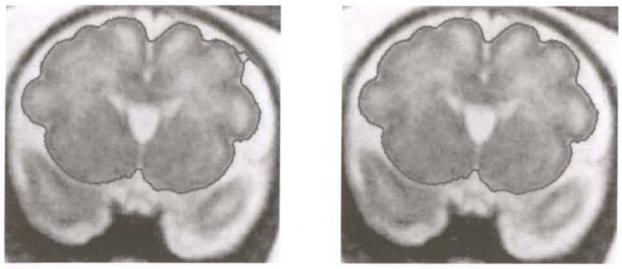

Figura 14. Da esquerda para a direita tem-se: uma correcção (a vermelho) e o resultado após ser adaptada.

cial complexo pode-se fazer um mais simples, corrigi-lo e depois acrescentar pormenores com outro esboço, melhorando o resultado final (ver **Figura 15**).

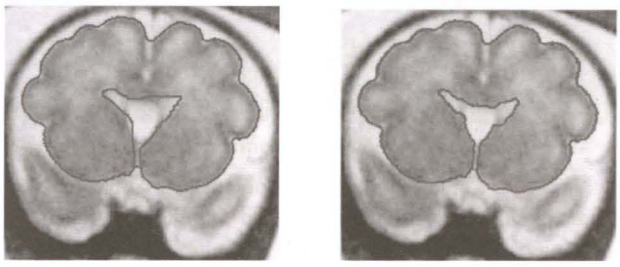

Figura 15. Da esquerda para a direita tem-se: uma extensão do contorno (a vermelho) e o resultado final.

O processo termina quando o utilizador estiver satisfeito com o resultado.

4.3. Análise

A técnica descrita na secção anterior permite ultrapassar não só as falhas das *snakes* normais mas também das GVF *snakes*:

- Auxilia a convergência, pois o resultado depois da correcção é garantidamente melhor que o anterior, ou no pior dos casos igual, pois se por qualquer motivo, má correcção do utilizador, for pior o seu efeito, ele pode ser cancelado (botão Undo); Auxilia a convergência, pois o resultado depois da correcção é habitualmente melhor que o anterior e garantidamente nunca é pior, pois se por qualquer motivo, má correcção do utilizador, o fosse, o seu efeito podia ser cancelado (botão Undo);
- é eficiente, pois todas as correcções e cálculos correspondentes são só locais;
- é fácil de usar, pois o utilizador interage de uma forma simples e natural ao desenhar alguns traços, em vez de definir os esboços ponto a ponto ou recorrendo a formas fixas (círculos). Além disso ele não precisa de conhecer o sistema em detalhe, pois todas as mudanças de parâmetros são feitas automaticamente, sem a sua intervenção;
- Apesar dos seus resultados mudarem consoante os esboços, diferentes inicializações/correcções produzem diferentes resultados, como os resultados podem ser corrigidos, as *oversketching snakes* são mais robustas que os algoritmos referidos anteriormente.

5. RESULTADOS

Nesta secção analisamos o resultado de várias segmentações de imagens, através de *oversketching snakes*. Para cada uma das imagens mostramos o esboço inicial, o contorno adaptado pela GVF *snake*, com as correcções necessárias a vermelho, e o contorno final.

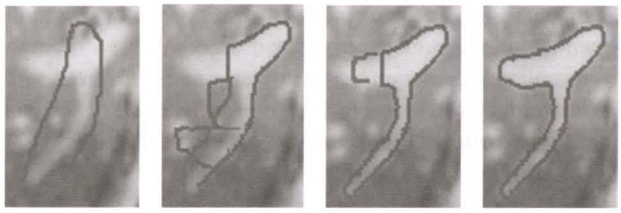

Figura 16. Da esquerda para a direita tem-se: o esboço inicial, o resultado da GVF *snake* (a azul) com a primeira correcção (a vermelho), o resultado anterior com outra correcção e o resultado final.

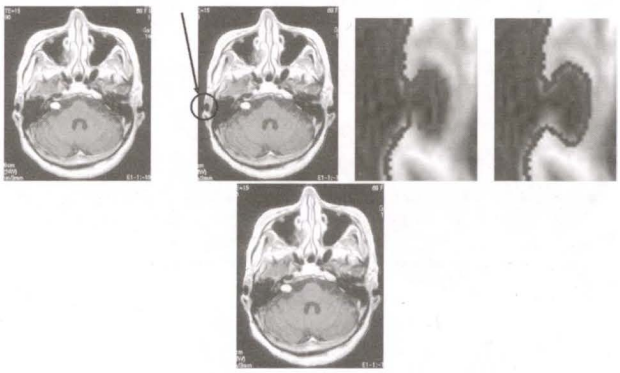

Figura 17. Em cima, da esquerda para a direita, tem-se: o esboço inicial, o resultado da GVF *snake* (a azul) com uma correcção assinalada pelo círculo, o zoom à área anterior com a correcção (a vermelho) e o zoom do resultado. Em baixo está o resultado final.

Figura 18. Da esquerda para a direita tem-se: o esboço inicial, o resultado da GVF *snake* (a azul) com uma correcção (a vermelho) e o resultado final.

	Número de pontos do esboço inicial	Tempo da GVF *snake* (em seg.)	Número de correcções	Tempo total de utilizador para as correcções (em seg.)
Figure 16	355	0.351	2	4.486
Figure 17	1917	3.515	1	2.293
Figure 18	356	0.170	1	2.624
Figure 19	577	0.210	3	7.876
Figure 20	310	0.260	3	9.216
Figure 21	903	1.352	6	19.090
Figure 22	887	1.303	4	15.436
Figure 23	754	0.751	8	28.571

Tabela 1. Diferentes medidas dos vários resultados de segmentação.

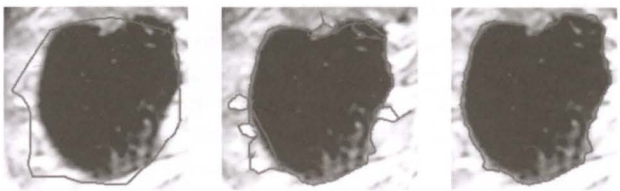

Figura 19. Da esquerda para a direita tem-se: o esboço inicial, o resultado da GVF *snake* (a azul) com três correcções (a vermelho) e o resultado final.

Figura 20. Da esquerda para a direita tem-se: o esboço inicial, o resultado da GVF *snake* (a azul) com três correcções (a vermelho) e o resultado final.

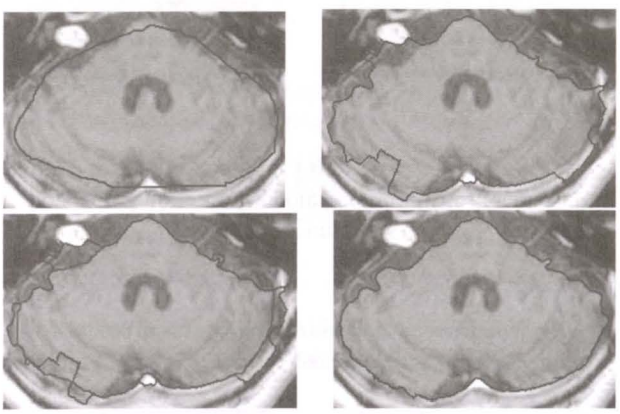

Figura 21. Da esquerda para a direita, de cima para baixo tem-se: o esboço inicial, o resultado só da GVF *snake*; o resultado anterior (a azul) com seis correcções (a vermelho) e o resultado final.

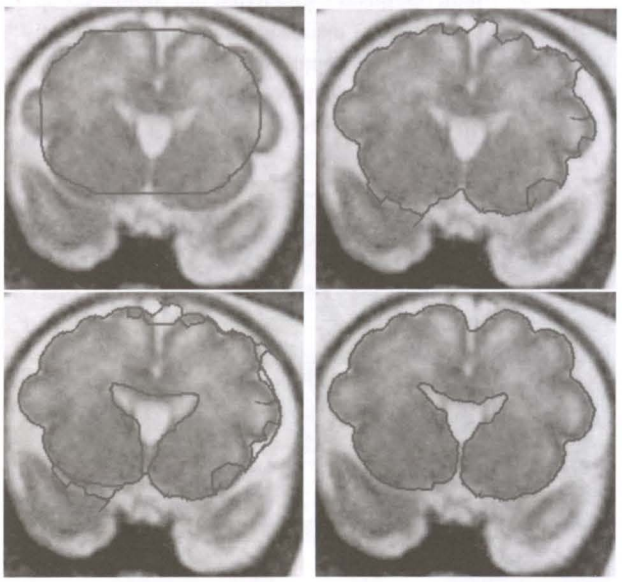

Figura 22. Da esquerda para a direita, de cima para baixo, tem-se: o esboço inicial, o resultado só da GVF *snake*; o resultado anterior (a azul) com quatro correcções (a vermelho) e o resultado final.

A **Tabela 1** mostra, para cada imagem, o número de pontos do esboço inicial, que têm entre eles a distância máxima de dois *pixels*, o tempo que a GVF *snake* inicial demorou, o número de correcções necessárias para obter o resultado final e o tempo de utilizador necessário para fazer essas correcções.

Estes resultados mostram que, apesar da GVF *snake* ser eficiente, em várias ocasiões necessita de ser parcialmente corrigida, em particular em situações com contornos pouco definidos (**Figuras 16**, **20** e **21**), com diferentes contornos próximos uns dos outros (**Figuras 19**, **21** e **23**) e para corrigir detalhes locais (**Figuras 17** e **18**). Dependendo da qualidade do esboço inicial o número de correcções pode aumentar (**Figuras 22** e **23**), mas apesar de serem necessárias mais algumas interacções obtém-se na mesma o resultado desejado.

Cada uma das correcções demora, aproximadamente, entre 2 e 4 segundos, dependendo da sua complexidade. Este

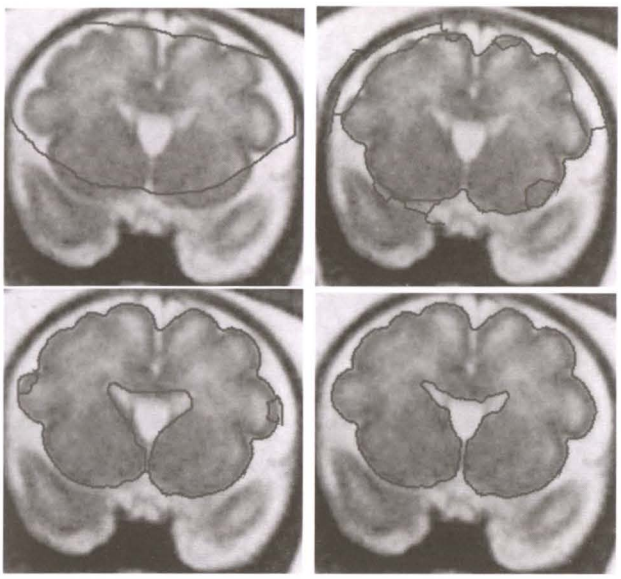

Figura 23. Da esquerda para a direita, de cima para baixo, tem-se: o esboço inicial, o resultado da GVF *snake* (a azul) com cinco correcções (a vermelho), o resultado após as correcções anteriores (a azul) com mais três correcções (a vermelho) e o resultado final.

tempo é aceitável, pois cada correcção envolve vários passos (o utilizador tem que identificar a sua necessidade, efectuá-la e esperar que se ajuste), só são necessárias algumas correcções e estas correcções possibilitam uma clara melhoria nos resultados.

6. CONCLUSÕES E TRABALHO FUTURO

A técnica apresentada neste artigo permite melhorar as segmentações obtidas por outros algoritmos, tanto na sua qualidade como na sua complexidade, através de interacção com o utilizador. Apesar desta intervenção tornar o processo mais lento, esta abordagem híbrida parece aconselhável no domínio das imagens médicas, pois, por um lado, é bem mais rápida que a solução manual existente e, por outro, melhora a precisão do resultado ao tirar partido da experiência do operador, através de algumas acções locais simples.

No futuro vamos fazer testes com utilizadores peritos em segmentação de imagens médicas, de forma a avaliarmos a nossa abordagem não só quantitativamente, ao comparar, por exemplo, as diferentes áreas segmentadas no nosso sistema com as produzidas numa segmentação manual, mas também qualitativamente. Outras áreas de possíveis desenvolvimentos futuros incluem usar esta técnica com outros algoritmos mais avançados [Cheng 06], de forma a reduzir o número de correcções, mantendo a qualidade do resultado, e testar a possibilidade de usar algoritmos diferentes para, por um lado, obter a aproximação do contorno inicial e, por outro, fazer as correcções necessárias.

Referências

[Antonelli 05] Michela Antonelli, Beatrice Lazzerini, e Francesco Marcelloni. Segmentation and reconstruction of the lung volume in CT images. Em *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*, páginas 255–259, New York, NY, USA, 2005. ACM Press.

[Boykov 01] Yuri Y. Boykov e Marie-Pierre Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. *Eighth IEEE International Conference on Computer Vision, 2001. ICCV 2001. Proceedings.*, 1:105–112, 2001.

[Cheng 06] Jierong Cheng e Say Wei Foo. Dynamic directional gradient vector flow for snakes. *IEEE Transactions on Image Processing*, 15(6):1563–1571, Jun. 2006.

[Foo 06] Jung Leng Foo. A survey of user interaction and automation in medical image segmentation methods. Relatório técnico, Iowa State University - Human Computer Interaction, Janeiro 2006.

[Hug 99] Johannes Hug, Christian Brechbuhler, e Gabor Szekely. Tamed snake: A particle system for robust semi-automatic segmentation. Em *MICCAI*, páginas 106–115, 1999.

[Kass 88] M. Kass, A. Witkin, e D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, Janeiro 1988.

[Li 04] Yin Li, Jian Sun, Chi-Keung Tang, e Heung-Yeung Shum. Lazy snapping. *ACM Trans. Graph.*, 23(3):303–308, 2004.

[Rother 04] Carsten Rother, Vladimir Kolmogorov, e Andrew Blake. "Grabcut": interactive foreground extraction using iterated graph cuts. Em *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, páginas 309–314, New York, NY, USA, 2004. ACM Press.

[Vu 06] Randy H. Vu, Rangaraj M. Rangayyan, e Graham S. Boag. Multi-seed segmentation of the primary tumor mass in neuroblastoma using opening-by-reconstruction. Em *BioMed'06: Proceedings of the 24th IASTED international conference on Biomedical engineering*, páginas 242–249, Anaheim, CA, USA, 2006. ACTA Press.

[Xu 98] Chenyang Xu e Jerry L. Prince. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, 7(3):359–369, Mar. 1998.