社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－CG－125 (10)
2006／11／16

ボリュームキャッチャーの拡張と応用

大和田 茂¶ フランク ニールセン¶ 五十嵐 健夫¶†
¶ソニー コンピュータサイエンス研究所
{sowd | frank.nielsen}@acm.org
†東京大学 JST PRESTO
takeo@acm.org

ボリュームデータをセグメンテーションすることは,ボリュームデータ処理および分析において最も大切な操作の一つであるが,これを効率的に行うことは難しいということが知られている.我々はレンダリングされた画像上をクリックしたりストロークを描くなど,いくつかの簡単で直感的なインターフェースを用いてこの作業を行う方法を提案する.我々は,ボリュームキャッチャーシステム[1]の前処理として二次元画像処理を行うことによって,様々なユーザーインターフェースを実現する.

An Extension and Application of the Volume Catcher System

Shigeru Owada¶ Frank Nielsen¶ Takeo Igarashi¶†
¶Sony Computer Science Laboratories, Inc.
†The University of Tokyo / JST PRESTO

Selecting a region of interest (ROI) within unsegmented volume data is one of the fundamental operations in volume data processing and analysis, yet it is difficult to perform the task efficiently. This paper proposes several simple and intuitive sketching user interface tools for the selection task, in which the user can directly click or draw a stroke on the volume-rendered object on the screen. The main contribution is that the user's input is pre-processed in 2D domain before applying the traditional 2D-to-3D stroke elevation algorithm (the Volume Catcher system [1]). We tested the system with real-world examples to verify the effectiveness of our approach.

1. 背景

ボリュームセグメンテーションは,CT スキャナなどから取得されたボリュームデータを,意味のある領域に分割する操作のことである.この操作は領域の形状や連結性を調べたり,可視化や種々の測定を行う上で大変重要で不可欠な処理である.この重要さは数十年前から認識されており多くの手法が提案されてきたが,今もって完全な自動手法(どんな対象に対しても人間の直感に合致する結果を返す手法)は現れていない.その理由の一つは,理想的なセグメンテーションには観察者の主観的な認識や固有の要求を反映させる必要があるため,それを明示的に指定することなしにコンピュータに理解させることは本質的に困難だからである.実際,エッジやテクスチャの情報を用いた局所的情報だけではよいセグメンテーションを行うことは難しく,グローバルな情報をユーザーインターフェースや対象ごとにモデル化された知識を与える手法が数多く提案されている.我々は様々な対象に一般的に用いることができる手法に関心があるため,ユーザーインターフェースによる解決を目指している.

対象が二次元画像でなく三次元ボリュームデータである場合に問題になるのは,ユーザーが指定するべき情報がすでに三次元であり,通常の二次元の入出力デバイスにはマッチしないという点である.この問題に対するトラディショナルな解決法としては,断層画面の一つをとり出し,その画像上に情報を与えるという手法が最も広く使われている[2].この手法は,三次元の座標を二段階に分けて直接指定する(断層画像の選択+二次元情報)というものであるが,より手間の少ない解決法として,二次元にレンダリングされた画像の上に情報を加えると,奥行き情報を自動的に計算する手法が近年相次いで発表された[1][3].大和田らの手法[1]はボリュームキャッチャーと呼ばれ,ユーザーが対象領域の輪郭線をなぞるというユーザーインターフェースにな

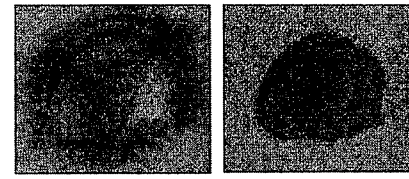
っている.この手法では,ユーザーが与える情報は線であり,その線が必ずしも閉じていなくてもよい(領域情報でなくてよい)ので,遮蔽があっても問題ないという利点がある.一方,輪郭をある程度正確になぞらなければならないという欠点がある.一方 Yuan らの Volume Cutout システム[3]では,前景領域と背景領域をラフに塗りわける事により情報を与えるようになっており,ユーザーの入力に正確さは全く必要ないという点は大きな利点だが,視点から見えない領域があるとうまく動かないという欠点がある.このように,ユーザーインターフェースの面から言えば,この二つの既存手法は一長一短である.

いずれにしても,これらのユーザーインターフェースは様々な可能性の中から意図的に,あるいは発想的に提案された単体の手法であり,普遍的なフレームワークではない.そこで本稿では,ボリュームキャッチャーの前処理として画像処理モジュールを組みこむことによって,これら二種の方法を含む,様々なユーザーインターフェースを実現できるプラットフォームを提案する.これにより,これまで二次元画像セグメンテーションで提案されてきた多くのユーザーインターフェースがそのまま三次元に導入できるようになると考えられる.

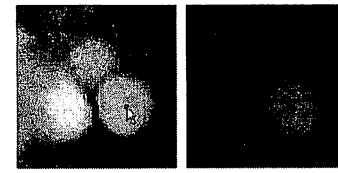
2. ユーザーインターフェース

我々の手法では,まずボリュームデータを読みこみ,任意の可視化パラメータ・視点からボリュームレンダリングする.この時,対象となる領域がなるべくよく見えるようにする.我々の現在の実装では,透明度調整,およびエッジ強調を行うことができる[4].

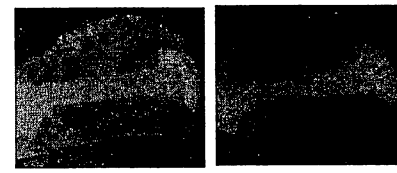
意図する領域を選択する操作としては,本稿では以下の 5 種類のユーザーインターフェースを提案する.1. 輪郭線トレース (領域の輪郭をなぞる.元のボリュームキャッチャーと同じ) 2. クリック選択(ほぼ丸く見える領域の中心をクリックする) 3. 中心線トレース(柱状の領域の中心線を描く) 4. Grabcut (領域を囲むバウンディングボックスを指定する) 5. 前景・背景スケッチ(前景・背景をラフに塗る.Volume Cutout と同じ) である(図 1)



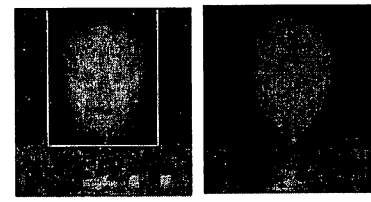
(a) Contour tracing tool



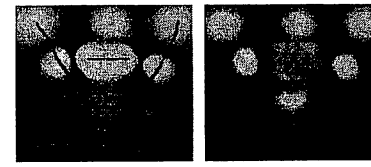
(b) Click selection tool



(c) Axis tracing tool



(d) Grabcut tool



(e) Foreground/Background sketching tool

図 1 : 提案するセグメンテーション手法.我々のフレームワークにより,様々なユーザーインターフェースが実現できる.

3. 実装

提案システムはクリックやストローク入力などがユーザーによって与えられると,現在のボリュームレンダリング画像とともに二次元画像処理モジュールにそれらを渡す.画像処理モジュールは出力として,ユーザーが選択したいと思われる領域の(二次元的な)輪郭線を生成する.この結果は,ボリュームキャッチャーシステムに入力される(図 2).しかし,このフレームワークでは,これ以外にも多くの二次元の領域選択ユーザーインターフェースを用いることができると考えられる(例 : 文法ベ

ースのセグメンテーションなど[5])

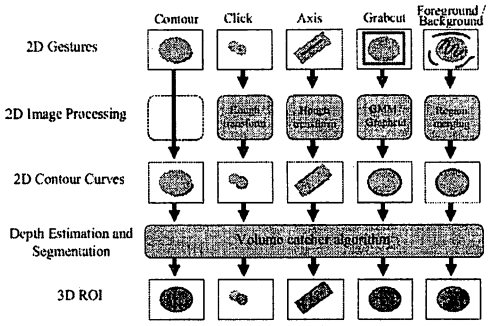


図 2：システムの概要

3.1 二次元画像処理モジュール

二次元画像処理モジュールの各機能の実装方法を以下に述べる.繰り返しになるが,モジュールの出力は,領域の境界線の部分集合である.

1. 輪郭線トレース
これは元々のボリュームキャッチャーの機能そのものである.ユーザーの入力は,そのまま出力される.
2. クリック選択
これは,レンダリングされた画像を微分した画像中から,Hough 変換によって円形の部分を見つける処理を行う.円の中心はクリック位置であるので,求めるべきパラメータは一つ,半径のみである.最もスコアの高い半径がみつかったら,その円を領域の輪郭線として出力する.
3. 中心線トレース
これも Hough 変換により実装可能である.クリック選択と同様に画像を微分し,ユーザーが描いた軸からのオフセットをパラメータとしてスコアが最大となるところを見付ける.
4. Grabcut
Grabcut 法は Rother らによって提案された二次元画像セグメンテーションのための手法である[6].Talbot らによる実装が公開されている[7]ので,これを呼び出すことにより我々のシステムに組みこんだ.結果として得られる領域の輪郭をトレースして出力とする.
5. 前景・背景スケッチ
このユーザーインターフェースは元々は Boykov らによるシステム[8]が最初だと思われるが,Volume Cutout や Lazy Snapping[9] など,多くのシステムで採用されている.我々はこのユーザーインターフェースを使って,Statistical Region Merging 法[10]と呼ばれるセグメンテーション手法のバイアスを与えることとした(ストロークを構成する点が,それぞれバイアス点としてアルゴリズムに入力される).

3.2 ボリュームキャッチャー

この部分はほとんど既存手法のボリュームキャッチャー[1]と同一であるが,連結でない入力輪郭線への対応が加えられている.

すでに述べたように,ボリュームキャッチャーシステムではストロークは連結である必要はない.そのため,輪郭線がはっきりしない部分は省いてもよい.ここで入力されなかった部分は後段のボリュームセグメンテーションステージにおいて,二次元のレンダリング画像よりはるかにリッチな情報を用いて探されることになる.

連結でない輪郭線が与えられた場合に問題となるのは,それぞれの輪郭線に独立に奥行きを与えてしまうと,近接した輪郭線の奥行きが大きく異なってしまう可能性があるということである.我々はこの問題を解決するために,複数の輪郭線に由来するスイープ面をパラメータ空間内で連結し,その上でパスを求めることとした(図 3).

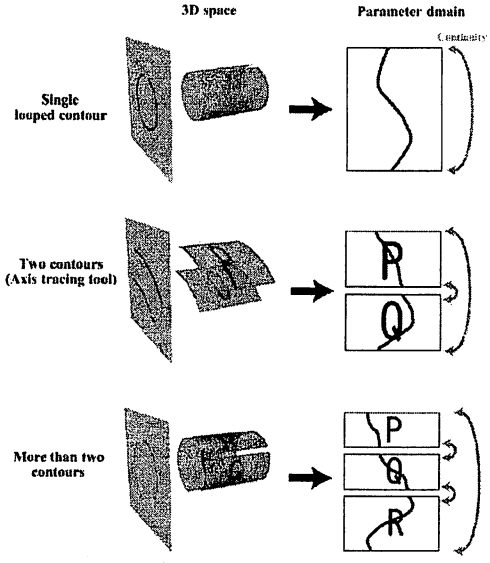


図 3： 連結でない輪郭線の処理

4. 結果と考察

我々は提案システムを様々な実データに適用した(図 4).ほとんどの場合,ユーザーはこれまでの場合に比べてよりセレクションタスクに注意を払わなくてよくなった.とりわけクリック選択ツールは大変便利かつ頑健であった.また,前景・背景選択ツールも,比較的頑健によい結果を返すため有用であった.一方中心線トレースは意図した結果が得られない事が多かった.調べるとこれは Hough 変換が意図した結果を返さない事が多いためであった.Grabcut もまた,バウンダリボックスを指定するだけではうまくいかないことがあった.これは,扱う対象が色のバラエティに乏しかった事が原因として考えられる.ただ,Grabcutにはストロークを加えていくことで細かな修正を行う機能がついているので,それを用いれば有用であった.以上から言える事は,ユーザーインターフェース自体の良し悪しよりも,裏で動くセグメンテーションアルゴリズムの動作の安定度に成功率が大きく左右されるということである.

今後はシステムが出力した結果の誤りを補正するための機能を追加する予定である.

謝辞

我々の用いたデータの一部は Visible Korean Human dataset (http://vkh3.kisti.re.kr/new) から得た.また,GrabCut システムの実装コードを公開されている Justin Talbot 氏に感謝する.

[1] S. Owada, F. Nielsen, and T. Igarashi. Volume catcher. In SI3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games, pages 111–116, New York, NY, USA, 2005. ACM Press.
[2] F.-Y. Tzeng, E. B. Lum, and K.-L. Ma. A novel interface for higher dimensional classification of volume data. In Proceedings of IEEE Visualization 2003, pages 505–512. IEEE, 2003.
[3] X. Yuan, N. Zhang, M. X. Nguyen, and B. Chen. Volume cutout. The Visual Computer (Special Issue of Pacific Graphics 2005), 21(8–10):745–754, 2005.
[4] B. Lichtenbelt, R. Crane, and S. Naqvi. Introduction to volume rendering. Prentice-Hall, Inc., 1998.
Press, 2002.
[5] F. Han and S. C. Zhu. Bottom-up/top-down image parsing by attribute graph grammar. In ICCV, pages 1778–1785, 2005.
[6] C. Rother, V. Kolmogorov, and A. Blake. "grabcut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph., 23(3):309–314, 2004.
[7] J.F. Talbot, and X. Xu. Implementing Grabcut,. http://www.justintalbot.org/
[8] Y. Boykov and M.-P. Jolly. Demonstration of segmentation with interactive graph cuts. In ICCV, page 741, 2001.
[9] Y. Li, J. Sun, C-K. Tang, and H-Y Shum. Lazy snapping. ACM Trans. Graph., 23(3):303–308, 2004.
[10] R. Nock and F. Nielsen. Grouping with bias revisited. In A. B. L.-S. Davis, R. Chellapa, editor, IEEE International Conference on Computer Vision and Pattern Recognition, pages 460–465. IEEE CS Press, 2004. (Applet http://www.csl.sony.co.jp/person/nielsen/SRMb/)

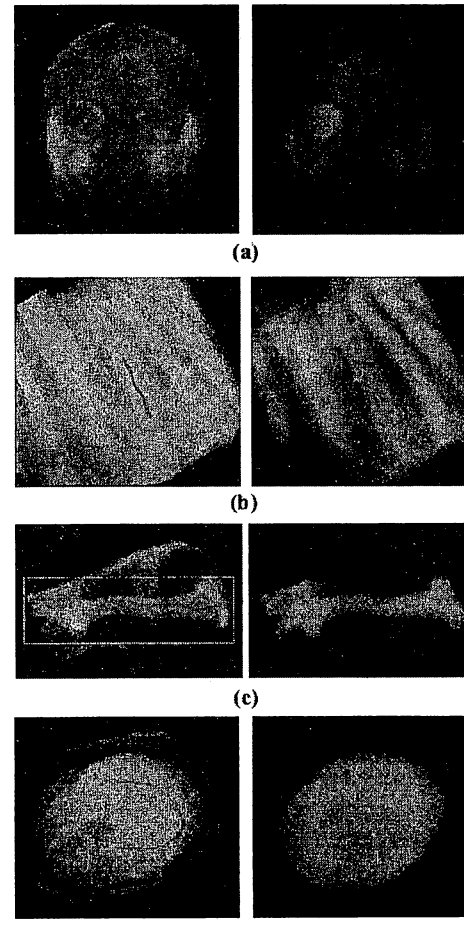


図 4： その他の実験結果