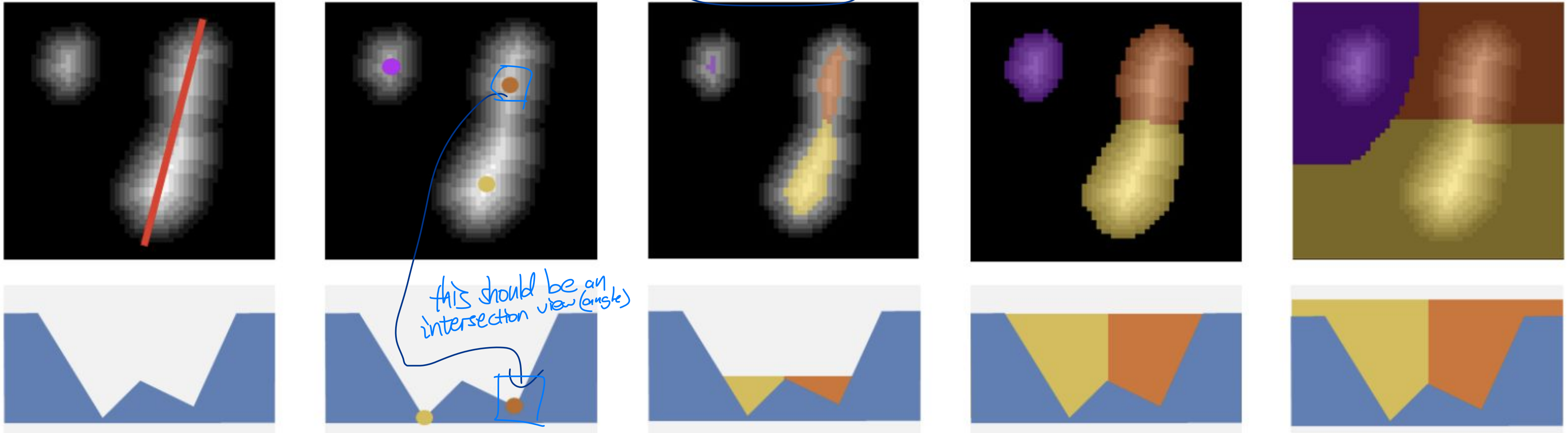
Brief Introduction to the Contour Trees

MA Student D. Tony Sun (dachao.sun@yu.edu; dsun1@mail.yu.edu) | Yeshiva University | Tuesday November 19, 2024 New York, N.Y.

The notion of "persistence homology," generally as a tool/method/approach, is what my study was focused on around 2016-2017 in the field of computer science, before a master's thesis (which ended up not using any of it). For a "space" specifically formatted data, say a grid or a network, *persistence homology* is focused on "topological features at different spatial resolutions" (Wikipedia page). In fact, the "persistence" part is actually explicitly defined in a particular tool and a particular data structure, as far as how to quantify (by a scalar measure) how "persistent" the dataset is, when it is subject to some examination, or even transformations.

(an relevant illustration from https://imagej.net/plugins/interactive-watershed)



Context of the Contour Trees (Carr, H. et al., 2003) Morse Th

Context & definition of what persistence homology—which I did not consolidate—which is (nonetheless) based upon a geometric object, such as triangle mesh.

My understanding: it is a **holistic tool** somehow (as compared to homotopy that focuses on identifying certain features) that encodes many things at once. The primary/only data structure I studied and struggled a lot with back then is this Contour Trees (this work gave a $n \log(n)$ algorithm which I tried implementing in C++) originally from radar data analysis, a.k.a. remote sensing or geography/GIS context. Perhaps, I didn't see anything or information missing from this contour tree structure that it does not capture, although I'm not thinking that it's perfect, either. I'll now organize these notes until next class time. This is a data structure/tool which I'm always hoping to talk and advocate more about: it is descriptive, encoding just about the right amount of information/features. Here the "persistence" corresponds to a scalar-function measure, also possibly known as a "tolerance threshold" value such that any "noise" (white noise, regular noise) with too small variations will not filtered-out as long as they are below this threshold. So noise is usually excluded from the resulting tree, hence yielding an informative tree...

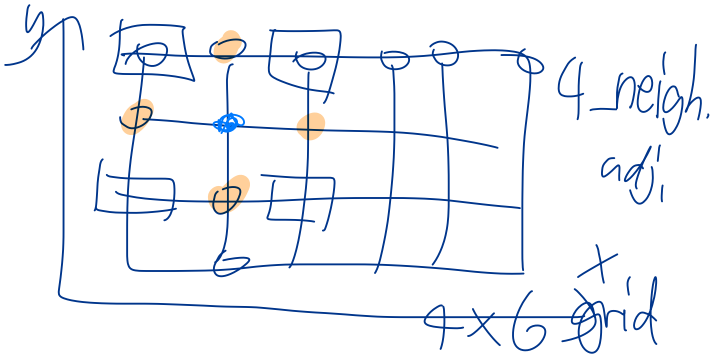
ELSEVIER

Computational Geometry

Volume 24, Issue 2, February 2003, Pages 75-94

Computing contour trees in all dimensions

Hamish Carr [a], Jack Snoeyink [b], Ulrike Axen [c]



Reflection: 2-D Example from My 2016 Notes

From a 2-D data with colorized components, associated to each "persistence level" where I think using many colors was just to "diversify" as a visualization technique—the colors are simply labels... although this algorithm is quite tedious to implement in C/Python code when I did it then (also sorting-based), but it was 6-7 years ago when I was still a bit “mechanic” as a learner; so, I can probably understand it more concisely today and do a precise presentation (if not too practical nor a detailed tutorial). Let me start planning ingredients of this topic to present next Tuesday/the week after. In particular, this algorithm belongs to the so-called “sort-and-scan” algorithms which are based on sorting (over some scalar quantity). And then, the necessity is to recognize neighbors for each element; after that, not only grid-based data, but also other structured data (i.e., a network) can be extracted a CT from.

Seam Carving vs. Contour Tree: "merge tree"

Topology-Aware Scalar Data Resizing?

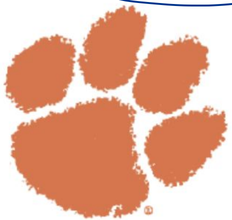
VC Lunch

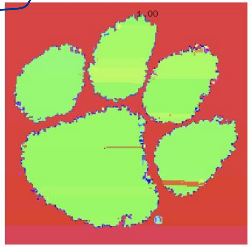
Friday April 1st, 2016

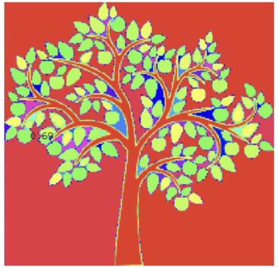
$f(2,1) = 57$

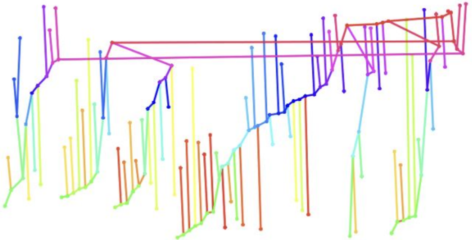
$f(3,4) = 57$

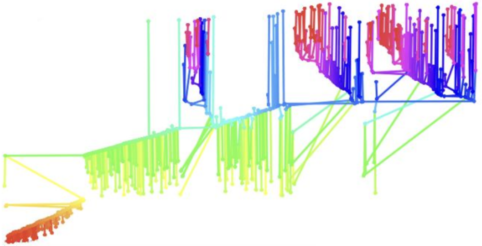
$x \cdot Row + y$











Carr's 2004 Doctoral Dissertation (British Columbia) with 200+ Pages

Topological Manipulation of Isosurfaces

by

Hamish Carr

B.Sc.(Hons.) 1987 University of Manitoba,

LL.B. 1990 University of Manitoba,

B.C.Sc.(Hons.) 1998 University of Manitoba,

M.Sc. 2000 University of British Columbia

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

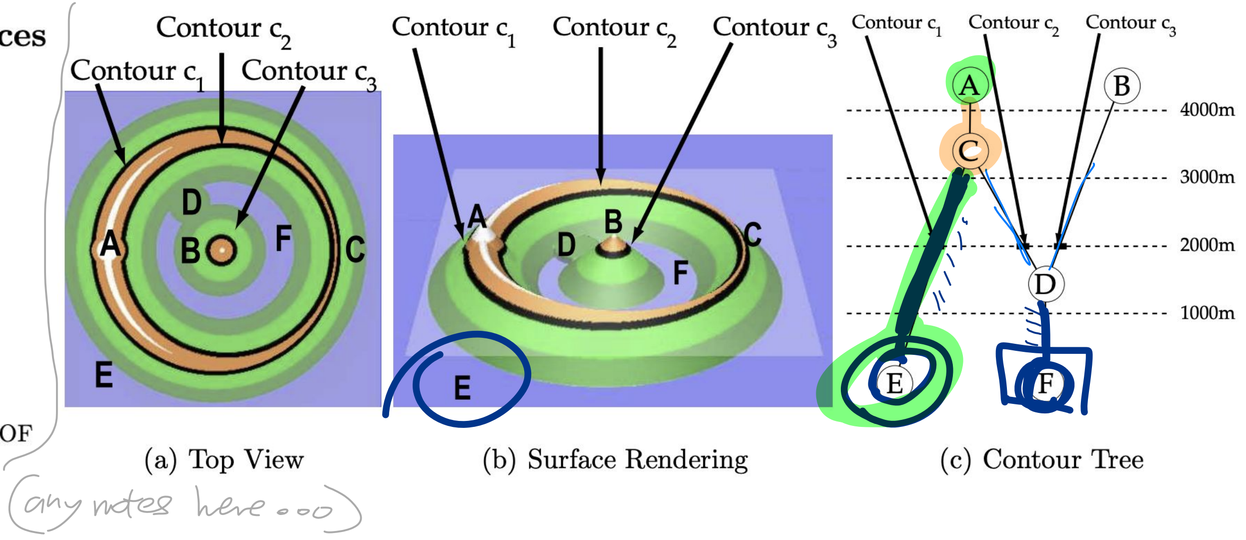
in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)

The University of British Columbia

April 2004



(a) Top View (b) Surface Rendering (c) Contour Tree

Figure 2.3: Example of a volcanic crater lake.
The contour tree expresses the adjacency of the coloured regions in the top view. Region E is adjacent to a green region, which is adjacent to contour c_1, which is adjacent to C. C is adjacent to two disjoint regions: A and contour c_2, and so on. The contour tree shows these relationships: here, the isovalue of each regions is used as the vertical dimension in (c).

The Basic Algorithm "For All Dimensions" and a Note on Neighborhoods

Def. (Level Sets) Start with n "points" $\{p_1, p_2, \ldots, p_n\}$ in a set in $\mathbb{R}^d$

Each point has a scalar measure, $\{h_1, h_2, \ldots, h_n\}$

and an overall, piecewise-linear interpolated function f such that

$f(p_i) = h_i$, for all $i = 1, 2, \ldots, n$

Then the level set is in regard to a/some specific function value, that is all points mapped to this value. Namely, it is the inverse image

$$f^{-1}(c) = \{p \in \mathbb{R}^d \mid f(p) = c\}$$

by Carr's (2003) notation, where each p here is still from the dataset above, as the context ......

(Section 4 of Carr et al, 2003)

§4.1 "Join and Split Trees"

Steps at a high level)
1. find/compute "join tree"
2. find "split tree"
3. merge them to form a merge tree
4. (simplify)

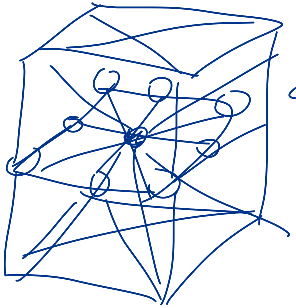
4.1 Join and split trees

Define a *join component* to be a connected component of the set $\{p \in \mathbb{R}^d \mid f(p) \leq x\}$. We will label a join component J_α^β if it is created at α and destroyed at β – where we think of the parameter as time in the same way as when we name level set components. By this definition, if two points belong to the same component of the level set, then they must belong to the same join component. Thus, each join component corresponds to at least one component of the level set, and possibly more.

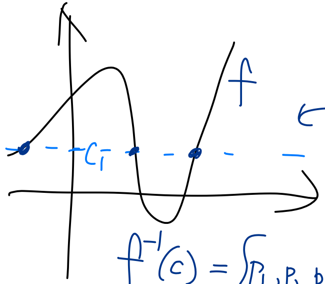
Define the *join tree* as a graph whose edges represent join components. One vertex, the root of the tree, represents the entire space. Other

The neighboring points/elements, for each element in dataset, are the essence here... and this makes it quite "geometrically"

normally known in advance (implicit) as well as pre-stored like an adj. list



← trying to say adjacent points in a 3-D image (grid)



$f^{-1}(c) = \{p_1, p_2, p_3\}$

← "level set" basically an inverse image in 1-D function

Dimension	Level Set (at Scalar Value c)
$\mathbb{R}^1$	set of points (often separate)
$\mathbb{R}^2$	isolines, or isocurves (must be closed)
$\mathbb{R}^3$	isosurfaces