

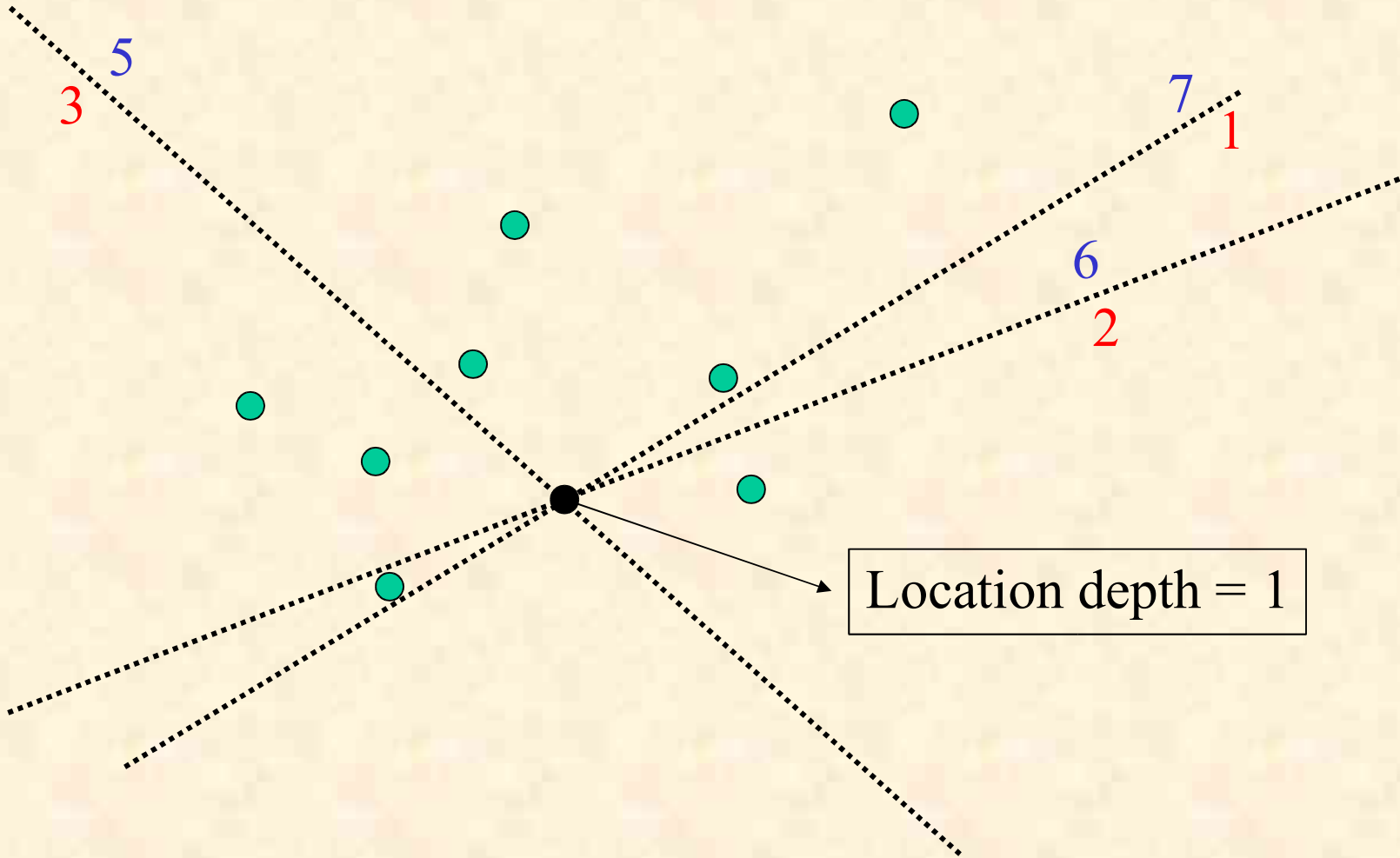
Hardware-Assisted Computation of Depth Contours

Shankar Krishnan (AT&T)

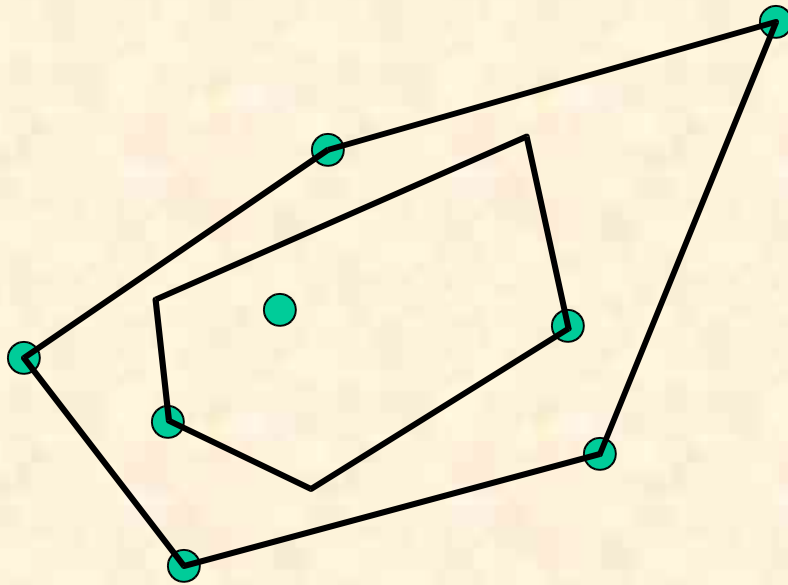
Nabil Mustafa (Duke)

Suresh Venkatasubramanian (AT&T)

Depth Contours



Depth Contours



k -contour: set of all points having location depth k

Every n -point set has an $n/3$ -contour [Helly]

The Tukey median is the deepest contour.

We wish to compute the set of all depth contours

What they are used for...



- Special case of general non-parametric data analysis tool
- Proposed by Tukey as a visualization/analysis method for scatter plots
- Have application to hypothesis testing, robust statistics

What is known...

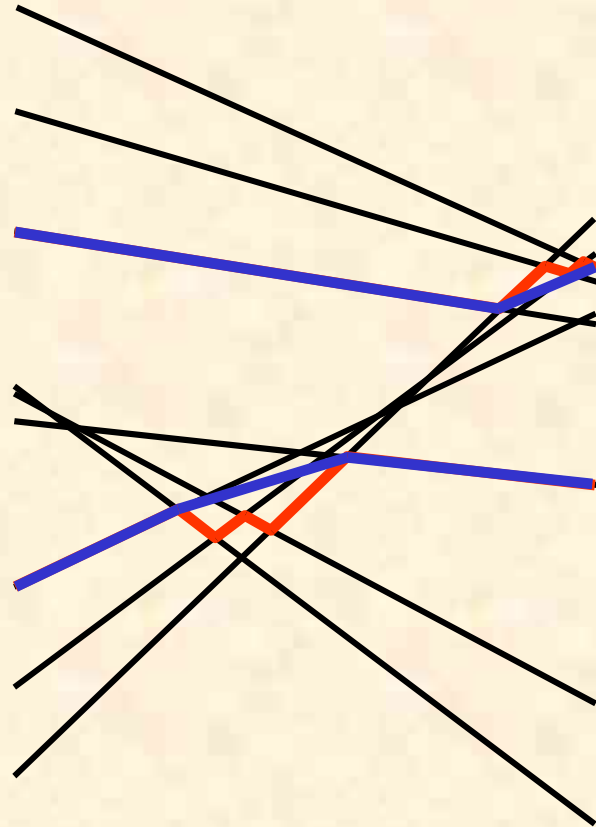
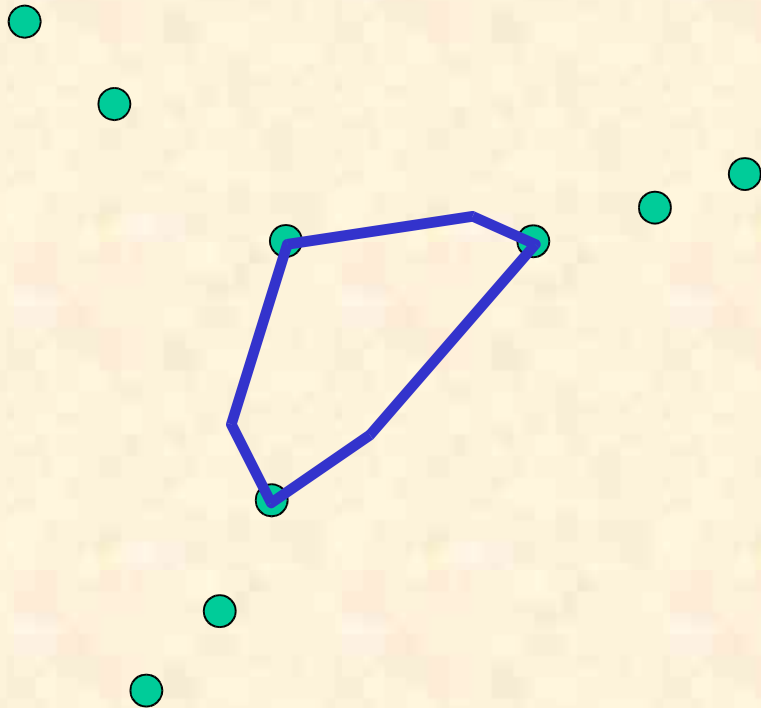
- The set of all contours can be computed in time $O(n^2)$ [MRRS⁴01]
- Any single contour can be computed in time $O(n \log^4 n)$ ($O(n \log^5 n)$ for the Tukey median) [M91]

Goal Of Our Work

- Practical efficient algorithms for computing depth contours.
- Programs like HALFMED, BAGPLOT used in the computational statistics community.
- [MRRS⁴01] presents a fast implementation.

Our algorithm makes use of graphics hardware to compute depth contours significantly faster than before.

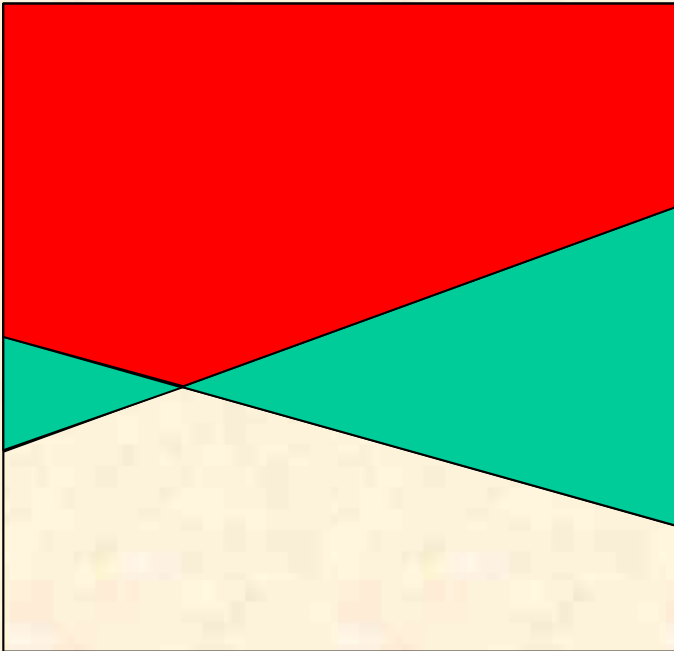
Contours \leftrightarrow ~~k~~-levels



k-contour: dual of convex hulls of k- and n-k levels in dual.

We compute k-contour by computing the convex hulls.

Main Algorithm



- Increment stencil buffer above dual line.
- After all lines are drawn, stencil buffer contains depths of each dual pixel
- Draw primal line for each dual pixel that lies on a dual line.

Use Depth-Test in framebuffer to only record (at each primal pixel) the line whose dual pixel has least depth.

Correctness

- Two duals are used to ensure that any pair of lines intersect in the dual window in at least one dual.
- Thus, depths of dual pixels are recorded correctly.
- The set of lines corresponding to dual pixels covers the primal plane.
- For every primal point p , there is at least one primal line whose dual point lies on the dual of p .

Experimental Results

N	SGI Octane		SGI Onyx		Linux/GeForce2	
	clock()	Elapsed	clock()	Elapsed	clock()	Elapsed
50	0.93	1.36	1.29	1.34	0.6	0.53
100	1.45	2.84	1.98	2.21	0.94	0.85
250	2.26	5.39	3.2	3.5	1.49	1.87
500	2.78	6.99	3.98	4.32	1.93	1.68
750	3.01	8.06	4.43	5.38	2.21	3.29
1000	3.27	8.96	4.83	5.97	2.49	3.81
2500	3.76	13.61	5.97	7.87	4.00	6.30
5000	4.31	20.99	7.14	10.76	6.34	10.45
7500	4.84	28.23	8.88	13.43	8.7	14.56
10000	5.45	35.48	10.34	16.18	11.09	18.59

Graphics Pipeline Modelling

- We model the graphics engine as a pipeline with basic operations and buffers.
- We propose asymptotic costs for basic operations, based on empirical studies of the performance of different chips.
- We analyse the performance of our algorithm in terms of these operations: our algorithm running time is $O(nW^3)$ (frame buffer is of size $W \times W$).

Open Questions

- Is there an efficient kinetic algorithm to maintain depth contours ?
- Can we compute contours in higher dimensions approximately/efficiently ?
- There are general notions of non-parametric data depth for which efficient algorithms are needed.
- Further work on understanding and modelling behaviour of graphics chips.