2D Computer Graphics

Diego Nehab

Summer 2020

IMPA

Introduction

What this course is about

Computer processing of 2D visual content

What this course is about

Computer processing of 2D visual content

Little to no focus on user interaction

- Not enough time...

What this course is about

Computer processing of 2D visual content

Little to no focus on user interaction

- Not enough time...

Why 2D?

- Counter to intuition, it is more demanding than 3D

What this course is about

Computer processing of 2D visual content

Little to no focus on user interaction

- Not enough time...

Why 2D?

- Counter to intuition, it is more demanding than 3D
- Everyday use of computers is almost exclusively 2D

What this course is about

Computer processing of 2D visual content

Little to no focus on user interaction

- Not enough time...

Why 2D?

- Counter to intuition, it is more demanding than 3D
- Everyday use of computers is almost exclusively 2D
- There are plenty of 3D courses out there

Course information

Teaching assistant

- Pedro Souza
- Lab time?

Course information

Teaching assistant

- Pedro Souza
- Lab time?

Course webpage

- `http://www.impa.br/~diego/teaching/vg`

Course information

Teaching assistant

- Pedro Souza
- Lab time?

Course webpage

- `http://www.impa.br/~diego/teaching/vg`

Discussion list

- `https://groups.google.com/d/forum/impa-2020-0-2dcg`

2D visual content

You are familiar with *images*

- Matrices where each entry is a color
- BMP, JPG, GIF, PNG, EXR, etc

2D VISUAL CONTENT

You are familiar with *images*

- Matrices where each entry is a color
- BMP, JPG, GIF, PNG, EXR, etc

Cameras can capture them

2D visual content

You are familiar with *images*

- Matrices where each entry is a color
- BMP, JPG, GIF, PNG, EXR, etc

Cameras can capture them

Artists can create or edit them with special software

- E.g., Gimp, Adobe Photoshop

2D visual content

You are familiar with *images*

- Matrices where each entry is a color
- BMP, JPG, GIF, PNG, EXR, etc

Cameras can capture them

Artists can create or edit them with special software

- E.g., Gimp, Adobe Photoshop

They can be *directly* displayed or printed

2D visual content

We will focus on *vector graphics*

- Layers of colored shapes
- PDF, SVG, AI, EPS, CGM, etc

2D visual content

We will focus on *vector graphics*

- Layers of colored shapes
- PDF, SVG, AI, EPS, CGM, etc

What you see in screens, other than photos and videos

2D visual content

We will focus on *vector graphics*

- Layers of colored shapes
- PDF, SVG, AI, EPS, CGM, etc

What you see in screens, other than photos and videos

Can be created by artists using special software

- E.g., Inkscape, Adobe Illustrator

2D visual content

We will focus on *vector graphics*

- Layers of colored shapes
- PDF, SVG, AI, EPS, CGM, etc

What you see in screens, other than photos and videos

Can be created by artists using special software

- E.g., Inkscape, Adobe Illustrator

Or by anyone that has ever used a word processor

2D visual content

We will focus on *vector graphics*

- Layers of colored shapes
- PDF, SVG, AI, EPS, CGM, etc

What you see in screens, other than photos and videos

Can be created by artists using special software

- E.g., Inkscape, Adobe Illustrator

Or by anyone that has ever used a word processor

Must be *rendered* into images before displayed or printed

Resolution and scalability

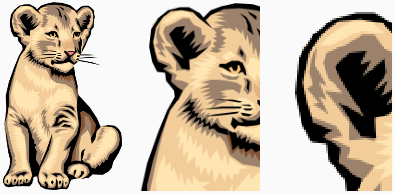
Images have a fixed, finite resolution

Resolution and scalability

Images have a fixed, finite resolution

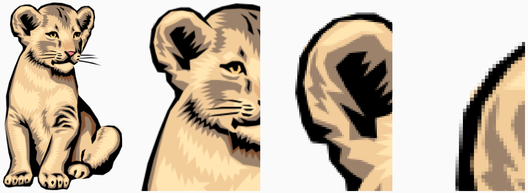
Resolution and scalability

Images have a fixed, finite resolution



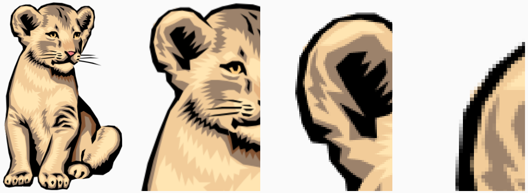
Resolution and scalability

Images have a fixed, finite resolution



Resolution and scalability

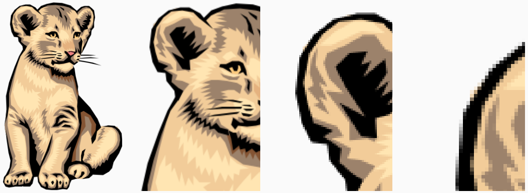
Images have a fixed, finite resolution



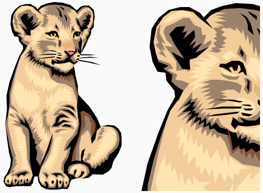
Vector graphics are *scalable*

Resolution and scalability

Images have a fixed, finite resolution

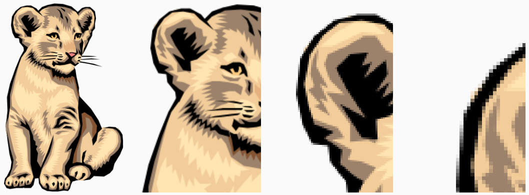


Vector graphics are *scalable*

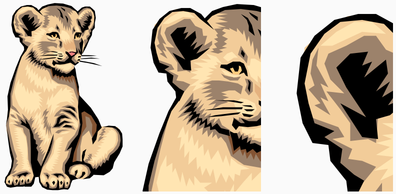


Resolution and scalability

Images have a fixed, finite resolution

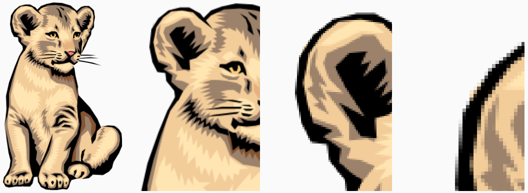


Vector graphics are *scalable*

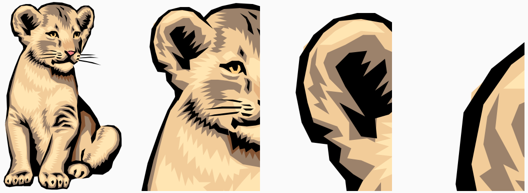


Resolution and scalability

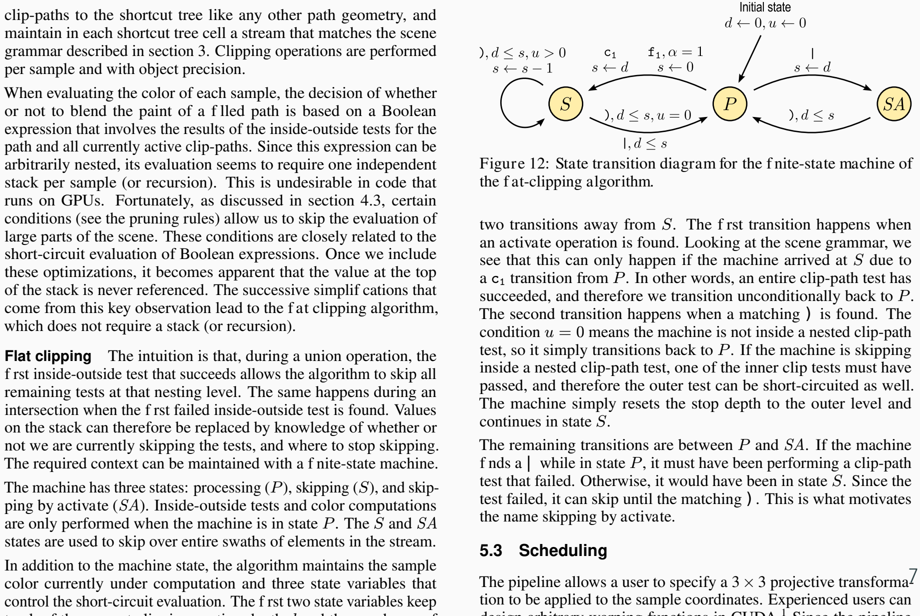
Images have a fixed, finite resolution



Vector graphics are *scalable*



Vector graphics are everywhere



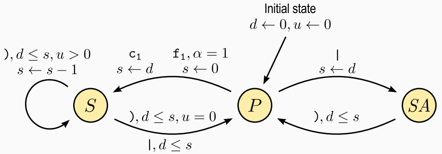
clip-paths to the shortcut tree like any other path geometry, and maintain in each shortcut tree cell a stream that matches the scene grammar described in section 3. Clipping operations are performed per sample and with object precision.

When evaluating the color of each sample, the decision of whether or not to blend the paint of a f lled path is based on a Boolean expression that involves the results of the inside-outside tests for the path and all currently active clip-paths. Since this expression can be arbitrarily nested, its evaluation seems to require one independent stack per sample (or recursion). This is undesirable in code that runs on GPUs. Fortunately, as discussed in section 4.3, certain conditions (see the pruning rules) allow us to skip the evaluation of large parts of the scene. These conditions are closely related to the short-circuit evaluation of Boolean expressions. Once we include these optimizations, it becomes apparent that the value at the top of the stack is never referenced. The successive simplif cations that come from this key observation lead to the f at clipping algorithm, which does not require a stack (or recursion).

Flat clipping The intuition is that, during a union operation, the f rst inside-outside test that succeeds allows the algorithm to skip all remaining tests at that nesting level. The same happens during an intersection when the f rst failed inside-outside test is found. Values on the stack can therefore be replaced by knowledge of whether or not we are currently skipping the tests, and where to stop skipping. The required context can be maintained with a f nite-state machine.

The machine has three states: processing (P), skipping (S), and skipping by activate (SA). Inside-outside tests and color computations are only performed when the machine is in state P. The S and SA states are used to skip over entire swaths of elements in the stream.

In addition to the machine state, the algorithm maintains the sample color currently under computation and three state variables that control the short-circuit evaluation. The f rst two state variables keep



$$\begin{tikzcd}
 & & \text{Initial state } d \leftarrow 0, u \leftarrow 0 \arrow[d] & & \\
S \arrow[loop left, "\texttt{)}, d \le s, u > 0;\ s \leftarrow s-1"] \arrow[rr, bend right, "\texttt{)}, d \le s, u = 0"'] \arrow[rr, bend right=50, "\texttt{|}, d \le s"'] & & P \arrow[ll, bend right, "\texttt{c}_1;\ s \leftarrow d \quad \texttt{f}_1, \alpha = 1;\ s \leftarrow 0"'] \arrow[rr, bend left, "\texttt{|};\ s \leftarrow d"] & & SA \arrow[ll, bend left, "\texttt{)}, d \le s"]
\end{tikzcd}$$

Figure 12: State transition diagram for the f nite-state machine of the f at-clipping algorithm.

two transitions away from S. The f rst transition happens when an activate operation is found. Looking at the scene grammar, we see that this can only happen if the machine arrived at S due to a $\texttt{c}_1$ transition from P. In other words, an entire clip-path test has succeeded, and therefore we transition unconditionally back to P. The second transition happens when a matching **)** is found. The condition $u = 0$ means the machine is not inside a nested clip-path test, so it simply transitions back to P. If the machine is skipping inside a nested clip-path test, one of the inner clip tests must have passed, and therefore the outer test can be short-circuited as well. The machine simply resets the stop depth to the outer level and continues in state S.

The remaining transitions are between P and SA. If the machine f nds a **|** while in state P, it must have been performing a clip-path test that failed. Otherwise, it would have been in state S. Since the test failed, it can skip until the matching **)**. This is what motivates the name skipping by activate.

5.3 Scheduling

The pipeline allows a user to specify a 3×3 projective transformation to be applied to the sample coordinates. Experienced users can

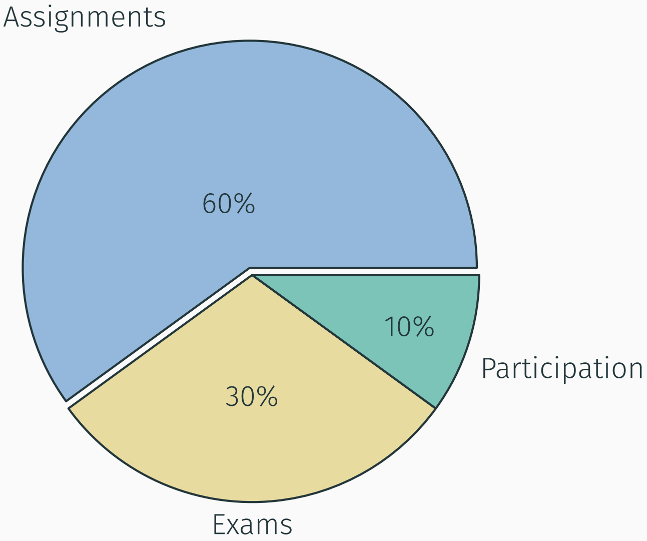
Vector graphics are everywhere

Vector graphics are everywhere

Inkscape 0.45

Evaluation

Grading

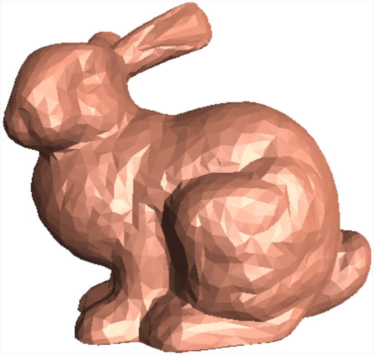


Assignments

1. Triangles, circles, and polygons

Assignments

1. Triangles, circles, and polygons

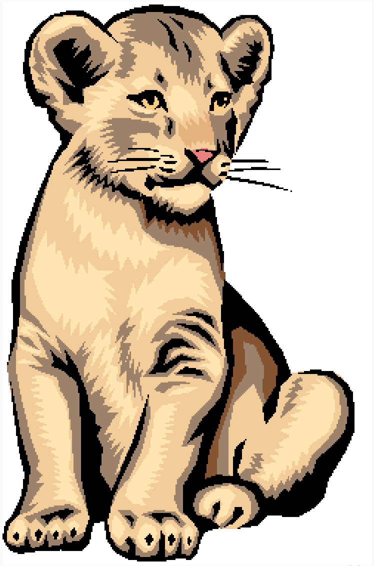


Assignments

1. Triangles, circles, and polygons

Assignments

1. Triangles, circles, and polygons



Assignments

1. Triangles, circles, and polygons
2. Add path rendering

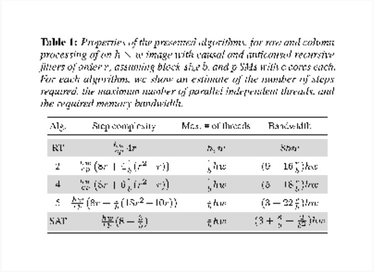


Table 1: *Properties of the presented algorithms, for row and column processing of an $h \times w$ image with causal and anticausal recursive filters of order r, assuming block size b, and p SMs with c cores each. For each algorithm, we show an estimate of the number of steps required, the maximum number of parallel independent threads, and the required memory bandwidth.*

Alg.	Step complexity	Max. # of threads	Bandwidth
RT	$\frac{hw}{cp} 4r$	h, w	$8hw$
2	$\frac{hw}{cp}\left(8r + \frac{4}{b}(r^2 + r)\right)$	$\frac{1}{b}hw$	$(9 + 16\frac{r}{b})hw$
4	$\frac{hw}{cp}\left(8r + \frac{6}{b}(r^2 + r)\right)$	$\frac{1}{b}hw$	$(5 + 18\frac{r}{b})hw$
5	$\frac{hw}{cp}\left(8r + \frac{1}{b}(18r^2 + 10r)\right)$	$\frac{1}{b}hw$	$(3 + 22\frac{r}{b})hw$
SAT	$\frac{hw}{cp}(8 + \frac{4}{b})$	$\frac{1}{b}hw$	$(3 + \frac{8}{b} + \frac{4}{b^2})hw$

Assignments

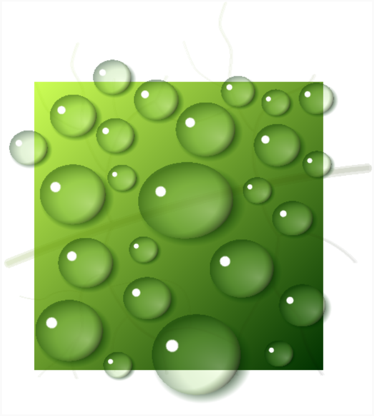
1. Triangles, circles, and polygons
2. Add path rendering

Assignments

1. Triangles, circles, and polygons
2. Add path rendering

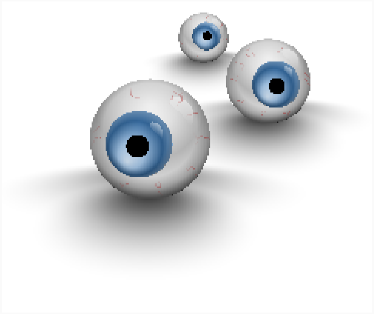
Assignments

1. Triangles, circles, and polygons
2. Add path rendering
3. Add transparency and gradients



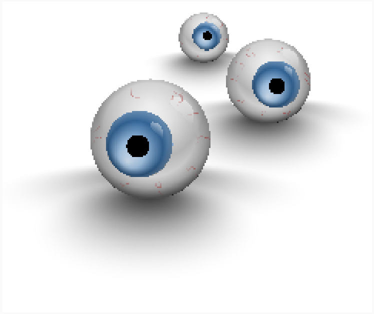
Assignments

1. Triangles, circles, and polygons
2. Add path rendering
3. Add transparency and gradients



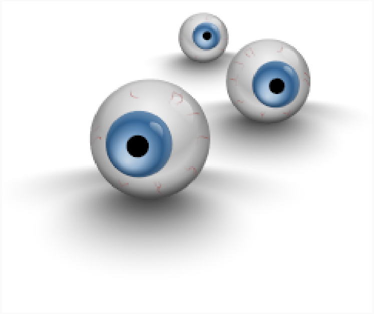
Assignments

1. Triangles, circles, and polygons
2. Add path rendering
3. Add transparency and gradients
4. Add implicit intersection tests



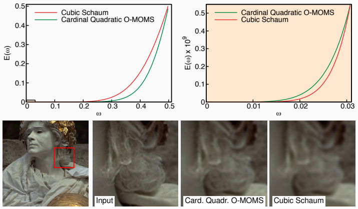
Assignments

1. Triangles, circles, and polygons
2. Add path rendering
3. Add transparency and gradients
4. Add implicit intersection tests
    4.1 Add anti-aliasing



Assignments

1. Triangles, circles, and polygons
2. Add path rendering
3. Add transparency and gradients
4. Add implicit intersection tests
   4.1 Add anti-aliasing
   4.2 Add texture mapping

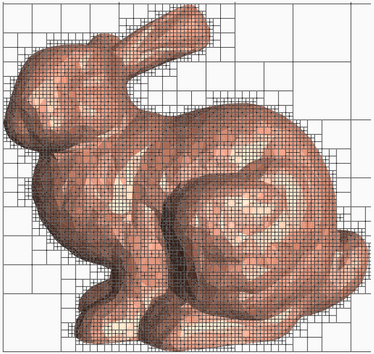


(a) PSNR=∞ , SSIM =1.0 (b) PSNR=32.21, SSIM=0.94 (c) PSNR=29.86, SSIM=0.90

Fig. 2. Comparison between the quadratic O-MOMS, a 3rd-order interpolator proposed by Blu et al. [4], and a 4th-order cubic by Schaum [32]. Even with its lower order, O-MOMS's error kernel shows a better behavior overall in most of the Nyquist interval (top left). Detail (top right) shows that Schaum's is only better for a tiny portion of the spectrum near the origin. Comparison of 30 consecutive rotations conf rm the better approximation qualities of the O-MOMS interpolator.

Assignments

1. Triangles, circles, and polygons
2. Add path rendering
3. Add transparency and gradients
4. Add implicit intersection tests
   4.1 Add anti-aliasing
   4.2 Add texture mapping
5. Add acceleration



Overview of lectures

Class 2: Geometry and transformations

Properties preserved by a group of transformations

- Euclydean
- Affine
- Projective

Class 2: Geometry and transformations

Properties preserved by a group of transformations

- Euclydean
- Affine
- Projective

Representations for points, vectors, and transformations

Class 2: Geometry and transformations

Properties preserved by a group of transformations

- Euclydean
- Affine
- Projective

Representations for points, vectors, and transformations

Focus on using transformations to solve geometric problems

Class 3: Vector graphics

Seminal work by Warnock and Wyatt [1982]

- PostScript, PDF, SVG
- RVG: our own representation

Class 3: Vector graphics

Seminal work by Warnock and Wyatt [1982]

- PostScript, PDF, SVG
- RVG: our own representation

Layers, shapes, and paints

Class 3: Vector graphics

Seminal work by Warnock and Wyatt [1982]

- PostScript, PDF, SVG
- RVG: our own representation

Layers, shapes, and paints

Basic rasterization loop

Class 3: Vector graphics

Seminal work by Warnock and Wyatt [1982]

- PostScript, PDF, SVG
- RVG: our own representation

Layers, shapes, and paints

Basic rasterization loop

Inside-outside test for triangles, polygons, and circles

Class 3: Vector graphics

Seminal work by Warnock and Wyatt [1982]

- PostScript, PDF, SVG
- RVG: our own representation

Layers, shapes, and paints

Basic rasterization loop

Inside-outside test for triangles, polygons, and circles

Assignment 1 posted: triangles, circles, and polygons

Class 4–5: Parametric curves

From polygons to *paths*

Class 4–5: Parametric curves

From polygons to *paths*

Splines, Lagrangian interpolation, B-splines

Class 4–5: Parametric curves

From polygons to *paths*

Splines, Lagrangian interpolation, B-splines

Bézier curves

- Bernstein basis
- Derivative, degree elevation
- Affine reparameterization, subdivision
- Intersection, monotonization
- Flattening

Class 4–5: Parametric curves

From polygons to *paths*

Splines, Lagrangian interpolation, B-splines

Bézier curves

- Bernstein basis
- Derivative, degree elevation
- Affine reparameterization, subdivision
- Intersection, monotonization
- Flattening

Rational Bézier curves

- Required for circular arcs

Class 6: Floating-point and root-finding

Representation of paths

- Converting other primitives to paths

Class 6: Floating-point and root-finding

Representation of paths

- Converting other primitives to paths

Floating-point representation and properties

- Numerical issues

Class 6: Floating-point and root-finding

Representation of paths

- Converting other primitives to paths

Floating-point representation and properties

- Numerical issues

Iterative root-finding methods

- Bisection
- Newton-Raphson
- *Safe* Newton-Raphson

Class 6: Floating-point and root-finding

Representation of paths

- Converting other primitives to paths

Floating-point representation and properties

- Numerical issues

Iterative root-finding methods

- Bisection
- Newton-Raphson
- *Safe* Newton-Raphson

Two simple methods for finding roots of polynomials

- Power basis
- Bernstein basis

Class 6: Floating-point and root-finding

Representation of paths

- Converting other primitives to paths

Floating-point representation and properties

- Numerical issues

Iterative root-finding methods

- Bisection
- Newton-Raphson
- *Safe* Newton-Raphson

Two simple methods for finding roots of polynomials

- Power basis
- Bernstein basis

Assignment 2 posted: path rendering

Class 7: Color and compositing

Radiometry

- Physics of light

Class 7: Color and compositing

Radiometry

- Physics of light

Photometry

- *Perception* of light

Class 7: Color and compositing

Radiometry

- Physics of light

Photometry

- *Perception* of light

Representation of colors by computer

- sRGB, XYZ
- Gamma correction

Class 7: Color and compositing

Radiometry

- Physics of light

Photometry

- *Perception* of light

Representation of colors by computer

- sRGB, XYZ
- Gamma correction

Transparency

- Seminal work by Porter and Duff [1984]
- Pre-multiplied alpha

Class 8: Gradient paints

Procedural way of defining spatially varying colors

Class 8: Gradient paints

Procedural way of defining spatially varying colors

2D map + color ramp

- Linear gradient
- Radial gradient

Class 8: Gradient paints

Procedural way of defining spatially varying colors

2D map + color ramp

- Linear gradient
- Radial gradient

Mesh gradients

- Gouraud shaded triangle mesh
- Coons patch mesh
- Tensor-product patch mesh

Class 8: Gradient paints

Procedural way of defining spatially varying colors

2D map + color ramp

- Linear gradient
- Radial gradient

Mesh gradients

- Gouraud shaded triangle mesh
- Coons patch mesh
- Tensor-product patch mesh

Assignment 3 posted: transparency and gradients

Class 9: Resultants and implicit curves

Moving towards an implicit test for intersections

- Avoid costly root-finding

Class 9: Resultants and implicit curves

Moving towards an implicit test for intersections

- Avoid costly root-finding

Implicit form of parametric polynomial curves

Class 9: Resultants and implicit curves

Moving towards an implicit test for intersections

- Avoid costly root-finding

Implicit form of parametric polynomial curves

Resultant

- Sylvester form
- Cayley-Bezout form

Class 9: Resultants and implicit curves

Moving towards an implicit test for intersections

- Avoid costly root-finding

Implicit form of parametric polynomial curves

Resultant

- Sylvester form
- Cayley-Bezout form

Affine implicitization

Class 10–11: Differential geometry

Planar parametric curves

Class 10–11: Differential geometry

Planar parametric curves

Rectification, and arc length

Class 10–11: Differential geometry

Planar parametric curves

Rectification, and arc length

Arc-length parameterization

Class 10–11: Differential geometry

Planar parametric curves

Rectification, and arc length

Arc-length parameterization

Curvature, offset, and evolute

Class 10–11: Differential geometry

Planar parametric curves

Rectification, and arc length

Arc-length parameterization

Curvature, offset, and evolute

Inflections

Class 10–11: Differential geometry

Planar parametric curves

Rectification, and arc length

Arc-length parameterization

Curvature, offset, and evolute

Inflections

Double-points

Class 10–11: Differential geometry

Planar parametric curves

Rectification, and arc length

Arc-length parameterization

Curvature, offset, and evolute

Inflections

Double-points

Stroking

Class 12: Abstract segments

The design of a segment primitive for rendering

Class 12: Abstract segments

The design of a segment primitive for rendering

Implicit test instead of root-finding

- Idea fails in general
- But works in a limited region of space

Class 12: Abstract segments

The design of a segment primitive for rendering

Implicit test instead of root-finding

- Idea fails in general
- But works in a limited region of space

Outside that region, we use simpler tests

- Bounding-box test
- Auxiliary line tests

Class 12: Abstract segments

The design of a segment primitive for rendering

Implicit test instead of root-finding

- Idea fails in general
- But works in a limited region of space

Outside that region, we use simpler tests

- Bounding-box test
- Auxiliary line tests

Assignment 4 posted: implicit intersection tests

Class 13: Digital images and anti-aliasing

Proper definition of *digital image*

Class 13: Digital images and anti-aliasing

Proper definition of *digital image*

Rendering as an approximation problem

Class 13: Digital images and anti-aliasing

Proper definition of *digital image*

Rendering as an approximation problem

Ideal sampling theory

- Introduction to Fourier transforms
- Whittaker-Nyquist-Kotelnikov-Shannon theorem
- Aliasing

Class 13: Digital images and anti-aliasing

Proper definition of *digital image*

Rendering as an approximation problem

Ideal sampling theory

- Introduction to Fourier transforms
- Whittaker-Nyquist-Kotelnikov-Shannon theorem
- Aliasing

Shift-invariant approximation spaces

- Ideal sampling reduces to sinc as generator
- Discussion of the box case
- Both are *orthogonal* spaces

Class 14: Anti-aliasing and texture mapping

The anti-aliasing integral

- Analytic solutions are not possible

Class 14: Anti-aliasing and texture mapping

The anti-aliasing integral

- Analytic solutions are not possible

Conflation of *coverage* with *opacity*

- Problem with *correlated mattes*
- Problem with gamma correction

Class 14: Anti-aliasing and texture mapping

The anti-aliasing integral

- Analytic solutions are not possible

Conflation of *coverage* with *opacity*

- Problem with *correlated mattes*
- Problem with gamma correction

Supersampling

- Monte Carlo integration
- Effect of sample distributions on variance

Class 14: Anti-aliasing and texture mapping

The anti-aliasing integral

- Analytic solutions are not possible

Conflation of *coverage* with *opacity*

- Problem with *correlated mattes*
- Problem with gamma correction

Supersampling

- Monte Carlo integration
- Effect of sample distributions on variance

Texturing filtering

- Mipmaps
- Anisotropic filtering

Class 15–16: Acceleration data structures

Classical acceleration data structures

- Space partition
  - Quadtree, K-d tree, and BSP

Class 15–16: Acceleration data structures

Classical acceleration data structures

- Space partition
  - Quadtree, K-d tree, and BSP
- Bounding volume hierarchy
  - R-tree

Class 15–16: Acceleration data structures

Classical acceleration data structures

- Space partition
  - Quadtree, K-d tree, and BSP
- Bounding volume hierarchy
  - R-tree

Specific for vector graphics

- Adaptation of quadtree and R-tree
- Shortcut tree
- Shortcut regular grid

Class 15–16: Acceleration data structures

Classical acceleration data structures

- Space partition
    - Quadtree, K-d tree, and BSP
- Bounding volume hierarchy
    - R-tree

Specific for vector graphics

- Adaptation of quadtree and R-tree
- Shortcut tree
- Shortcut regular grid

Assignment 5 posted: acceleration

Class 17: Typesetting

History of typesetting

- Calligraphy
- Gutenberg's printing press

Class 17: Typesetting

History of typesetting

- Calligraphy
- Gutenberg’s printing press

Unicode

Class 17: Typesetting

History of typesetting

- Calligraphy
- Gutenberg's printing press

Unicode

Fonts

- Metafont, TTF, Type 1, OpenType
- Metrics, shaping, kerning, ligatures
- Hinting, ClearType

Class 17: Typesetting

History of typesetting

- Calligraphy
- Gutenberg's printing press

Unicode

Fonts

- Metafont, TTF, Type 1, OpenType
- Metrics, shaping, kerning, ligatures
- Hinting, ClearType

Paragraph

- Hyphenation and justification
- Seminal work by Knuth and Plass [1981]
- Micro-typography

Class 18: Stroked primitives

Definition

- Dashing and decorations

Class 18: Stroked primitives

Definition

- Dashing and decorations

Two different approaches to rendering

- Using distance to generator

Class 18: Stroked primitives

Definition

- Dashing and decorations

Two different approaches to rendering

- Using distance to generator
- Converting to filled primitives

Class 18: Stroked primitives

Definition

- Dashing and decorations

Two different approaches to rendering

- Using distance to generator
- Converting to filled primitives

Two conversion methods

- Flattening the generator

Class 18: Stroked primitives

Definition

- Dashing and decorations

Two different approaches to rendering

- Using distance to generator
- Converting to filled primitives

Two conversion methods

- Flattening the generator
- Outputting curved outlines

Class 18: Stroked primitives

Definition

- Dashing and decorations

Two different approaches to rendering

- Using distance to generator
- Converting to filled primitives

Two conversion methods

- Flattening the generator
- Outputting curved outlines

Required approximations

- To arc length

Class 18: Stroked primitives

Definition

- Dashing and decorations

Two different approaches to rendering

- Using distance to generator
- Converting to filled primitives

Two conversion methods

- Flattening the generator
- Outputting curved outlines

Required approximations

- To arc length
- To offset and evolute

Class 19: Screen-space effects

Blur

- Direct convolution
- In frequency domain
- Recursive filter
- Monte Carlo

Class 19: Screen-space effects

Blur

- Direct convolution
- In frequency domain
- Recursive filter
- Monte Carlo

Clipping

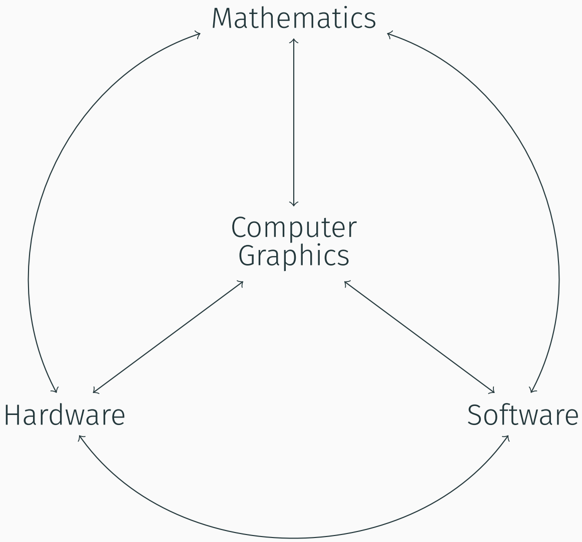
- Per pixel or per sample
- Vatti's algorithm [1992]

Class 20: Other rendering algorithms

Active edge list algorithm [1967]

NVPR [2012]

Old-school graphics


Mathematics
Computer Graphics
Hardware
Software

Displays from 1980–1990

CGA (Color Graphics Array) (1981)

- 16KB of video memory
- Text: 80×25 with 8×8 characters
- Graphics: 320×200 4 bpp, 640×200 1bpp

Displays from 1980–1990

CGA (Color Graphics Array) (1981)

- 16KB of video memory
- Text: 80×25 with 8×8 characters
- Graphics: 320×200 4 bpp, 640×200 1bpp

VGA (Video Graphics Array) (1987)

- 256KB of video memory
- Text mode 80×25 with 9×16 characters
- Graphics: 320×240 8bpp, 640×480 4bpp

Displays from 1980–1990

CGA (Color Graphics Array) (1981)

- 16KB of video memory
- Text: 80×25 with 8×8 characters
- Graphics: 320×200 4 bpp, 640×200 1bpp

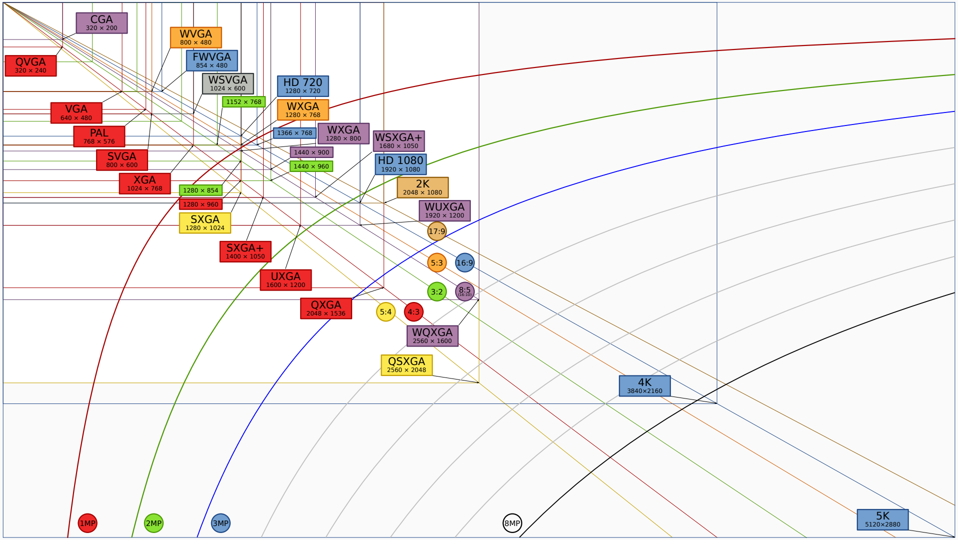
VGA (Video Graphics Array) (1987)

- 256KB of video memory
- Text mode 80×25 with 9×16 characters
- Graphics: 320×240 8bpp, 640×480 4bpp

SVGA (Super Video Graphics Array) (1989)

- Graphics: 800×600 4bpp, 640×480 8bpp

Screen resolution

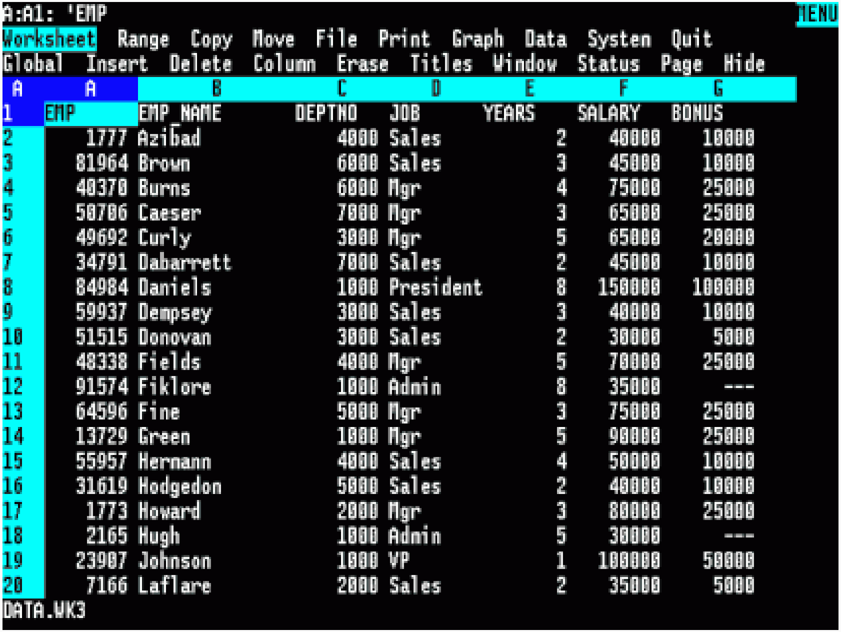


Text mode

Code page 437

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	NUL 0000 ***0***	☺ 263A ***1***	☻ 263B ***2***	♥ 2665 ***3***	♦ 2666 ***4***	♣ 2663 ***5***	♠ 2660 ***6***	• 2022 ***7***	◘ 25D8 ***8***	25CB ***9***	◙ 25D9 ***10***	♂ 2642 ***11***	♀ 2640 ***12***	♪ 266A ***13***	♫ 266B ***14***	☼ 263C ***15***
1_	25BA ***16***	◄ 25C4 ***17***	↕ 2195 ***18***	‼ 203C ***19***	¶ 00B6 ***20***	§ 00A7 ***21***	▬ 25AC ***22***	↨ 21A8 ***23***	2191 ***24***	↓ 2193 ***25***	→ 2192 ***26***	2190 ***27***	∟ 221F ***28***	↔ 2194 ***29***	▲ 25B2 ***30***	25BC ***31***
2_	SP 0020 ***32***	! 0021 ***33***	" 0022 ***34***	# 0023 ***35***	$ 0024 ***36***	% 0025 ***37***	& 0026 ***38***	' 0027 ***39***	(0028 ***40***	) 0029 ***41***	* 002A ***42***	+ 002B ***43***	, 002C ***44***	- 002D ***45***	. 002E ***46***	/ 002F ***47***
3_	0 0030 ***48***	1 0031 ***49***	2 0032 ***50***	3 0033 ***51***	4 0034 ***52***	5 0035 ***53***	6 0036 ***54***	7 0037 ***55***	8 0038 ***56***	9 0039 ***57***	: 003A ***58***	; 003B ***59***	< 003C ***60***	= 003D ***61***	> 003E ***62***	? 003F ***63***
4_	@ 0040 ***64***	A 0041 ***65***	B 0042 ***66***	C 0043 ***67***	D 0044 ***68***	E 0045 ***69***	F 0046 ***70***	G 0047 ***71***	H 0048 ***72***	I 0049 ***73***	J 004A ***74***	K 004B ***75***	L 004C ***76***	M 004D ***77***	N 004E ***78***	O 004F ***79***
5_	P 0050 ***80***	Q 0051 ***81***	R 0052 ***82***	S 0053 ***83***	T 0054 ***84***	U 0055 ***85***	V 0056 ***86***	W 0057 ***87***	X 0058 ***88***	Y 0059 ***89***	Z 005A ***90***	[005B ***91***	\ 005C ***92***	] 005D ***93***	^ 005E ***94***	_ 005F ***95***
6_	` 0060 ***96***	a 0061 ***97***	b 0062 ***98***	c 0063 ***99***	d 0064 ***100***	e 0065 ***101***	f 0066 ***102***	g 0067 ***103***	h 0068 ***104***	i 0069 ***105***	j 006A ***106***	k 006B ***107***	l 006C ***108***	m 006D ***109***	n 006E ***110***	o 006F ***111***
7_	p 0070 ***112***	q 0071 ***113***	r 0072 ***114***	s 0073 ***115***	t 0074 ***116***	u 0075 ***117***	v 0076 ***118***	w 0077 ***119***	x 0078 ***120***	y 0079 ***121***	z 007A ***122***	{ 007B ***123***	| 007C ***124***	} 007D ***125***	~ 007E ***126***	⌂ 2302 ***127***

CGA TEXT USER INTERFACE

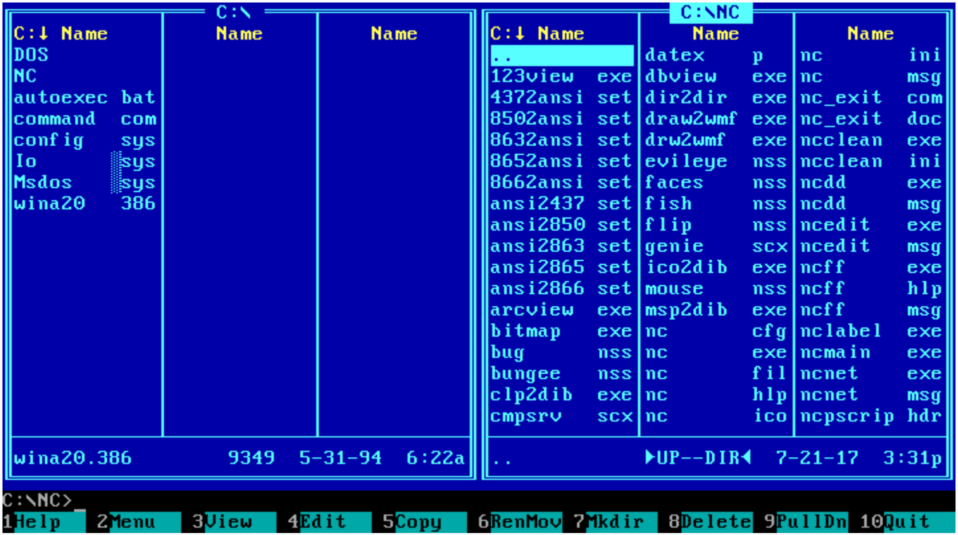


```
A:A1: 'EMP                                                                  MENU
Worksheet  Range  Copy  Move  File  Print  Graph  Data  System  Quit
Global  Insert  Delete  Column  Erase  Titles  Window  Status  Page  Hide
 A       A        B               C       D        E         F        G
1   EMP      EMP_NAME        DEPTNO   JOB       YEARS     SALARY   BONUS
2       1777 Azibad             4000 Sales          2     40000    10000
3      81964 Brown              6000 Sales          3     45000    10000
4      40370 Burns              6000 Mgr            4     75000    25000
5      50706 Caeser             7000 Mgr            3     65000    25000
6      49692 Curly              3000 Mgr            5     65000    20000
7      34791 Dabarrett          7000 Sales          2     45000    10000
8      84984 Daniels            1000 President      8    150000   100000
9      59937 Dempsey            3000 Sales          3     40000    10000
10     51515 Donovan            3000 Sales          2     30000     5000
11     48338 Fields             4000 Mgr            5     70000    25000
12     91574 Fiklore            1000 Admin          8     35000      ---
13     64596 Fine               5000 Mgr            3     75000    25000
14     13729 Green              1000 Mgr            5     90000    25000
15     55957 Hermann            4000 Sales          4     50000    10000
16     31619 Hodgedon           5000 Sales          2     40000    10000
17      1773 Howard             2000 Mgr            3     80000    25000
18      2165 Hugh               1000 Admin          5     30000      ---
19     23907 Johnson            1000 VP             1    100000    50000
20      7166 Laflare            2000 Sales          2     35000     5000
DATA.WK3
```

Code page 437

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
8_	Ç 00C7 ***128***	ü 00FC ***129***	é 00E9 ***130***	â 00E2 ***131***	ä 00E4 ***132***	à 00E0 ***133***	å 00E5 ***134***	ç 00E7 ***135***	ê 00EA ***136***	ë 00EB ***137***	è 00E8 ***138***	ï 00EF ***139***	î 00EE ***140***	ì 00EC ***141***	Ä 00C4 ***142***	Å 00C5 ***143***
9_	É 00C9 ***144***	æ 00E6 ***145***	Æ 00C6 ***146***	ô 00F4 ***147***	ö 00F6 ***148***	ò 00F2 ***149***	û 00FB ***150***	ù 00F9 ***151***	ÿ 00FF ***152***	Ö 00D6 ***153***	Ü 00DC ***154***	¢ 00A2 ***155***	£ 00A3 ***156***	¥ 00A5 ***157***	₧ 20A7 ***158***	ƒ 0192 ***159***
A_	á 00E1 ***160***	í 00ED ***161***	ó 00F3 ***162***	ú 00FA ***163***	ñ 00F1 ***164***	Ñ 00D1 ***165***	ª 00AA ***166***	º 00BA ***167***	¿ 00BF ***168***	⌐ 2310 ***169***	¬ 00AC ***170***	½ 00BD ***171***	¼ 00BC ***172***	¡ 00A1 ***173***	« 00AB ***174***	» 00BB ***175***
B_	░ 2591 ***176***	▒ 2592 ***177***	▓ 2593 ***178***	│ 2502 ***179***	┤ 2524 ***180***	╡ 2561 ***181***	╢ 2562 ***182***	╖ 2556 ***183***	╕ 2555 ***184***	╣ 2563 ***185***	║ 2551 ***186***	╗ 2557 ***187***	╝ 255D ***188***	╜ 255C ***189***	╛ 255B ***190***	┐ 2510 ***191***
C_	└ 2514 ***192***	┴ 2534 ***193***	┬ 252C ***194***	├ 251C ***195***	─ 2500 ***196***	┼ 253C ***197***	╞ 255E ***198***	╟ 255F ***199***	╚ 255A ***200***	╔ 2554 ***201***	╩ 2569 ***202***	╦ 2566 ***203***	╠ 2560 ***204***	═ 2550 ***205***	╬ 256C ***206***	╧ 2567 ***207***
D_	╨ 2568 ***208***	╤ 2564 ***209***	╥ 2565 ***210***	╙ 2559 ***211***	╘ 2558 ***212***	╒ 2552 ***213***	╓ 2553 ***214***	╫ 256B ***215***	╪ 256A ***216***	┘ 2518 ***217***	┌ 250C ***218***	█ 2588 ***219***	▄ 2584 ***220***	▌ 258C ***221***	▐ 2590 ***222***	▀ 2580 ***223***
E_	α 03B1 ***224***	ß 00DF ***225***	Γ 0393 ***226***	π 03C0 ***227***	Σ 03A3 ***228***	σ 03C3 ***229***	µ 00B5 ***230***	τ 03C4 ***231***	Φ 03A6 ***232***	Θ 0398 ***233***	Ω 03A9 ***234***	δ 03B4 ***235***	∞ 221E ***236***	φ 03C6 ***237***	ε 03B5 ***238***	∩ 2229 ***239***
F_	≡ 2261 ***240***	± 00B1 ***241***	≥ 2265 ***242***	≤ 2264 ***243***	⌠ 2320 ***244***	⌡ 2321 ***245***	÷ 00F7 ***246***	≈ 2248 ***247***	° 00B0 ***248***	∙ 2219 ***249***	· 00B7 ***250***	√ 221A ***251***	ⁿ 207F ***252***	² 00B2 ***253***	25A0 ***254***	NBSP 00A0 ***255***

VGA textual user interface



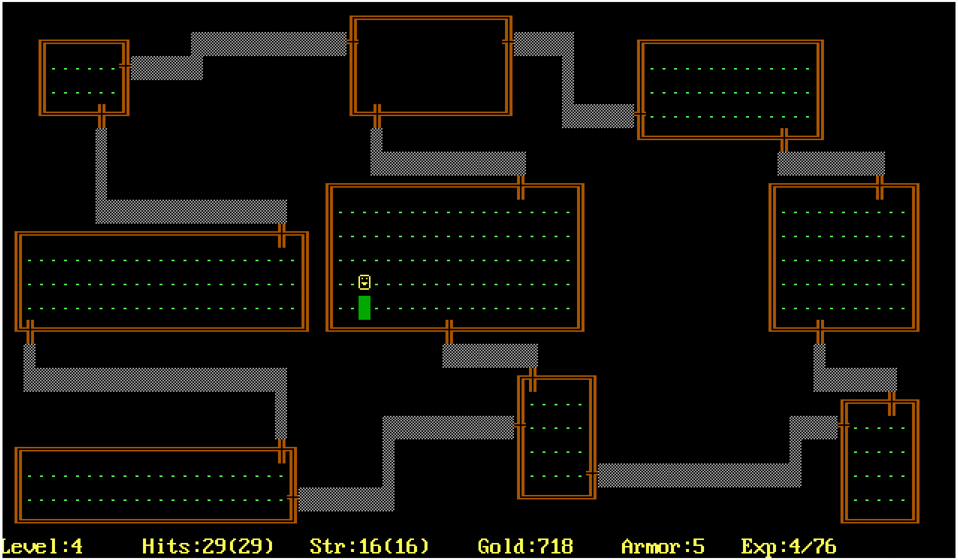
```
                  C:\                                       C:\NC
C:↓ Name     │    Name     │    Name     ║ C:↓ Name     │    Name      │    Name
DOS          │             │             ║ ..           │ datex      p │ nc       ini
NC           │             │             ║ 123view  exe │ dbview   exe │ nc       msg
autoexec bat │             │             ║ 4372ansi set │ dir2dir  exe │ nc_exit  com
command  com │             │             ║ 8502ansi set │ draw2wmf exe │ nc_exit  doc
config   sys │             │             ║ 8632ansi set │ drw2wmf  exe │ ncclean  exe
Io       sys │             │             ║ 8652ansi set │ evileye  nss │ ncclean  ini
Msdos    sys │             │             ║ 8662ansi set │ faces    nss │ ncdd     exe
wina20   386 │             │             ║ ansi2437 set │ fish     nss │ ncdd     msg
             │             │             ║ ansi2850 set │ flip     nss │ ncedit   exe
             │             │             ║ ansi2863 set │ genie    scx │ ncedit   msg
             │             │             ║ ansi2865 set │ ico2dib  exe │ ncff     exe
             │             │             ║ ansi2866 set │ mouse    nss │ ncff     hlp
             │             │             ║ arcview  exe │ msp2dib  exe │ ncff     msg
             │             │             ║ bitmap   exe │ nc       cfg │ nclabel  exe
             │             │             ║ bug      nss │ nc       exe │ ncmain   exe
             │             │             ║ bungee   nss │ nc       fil │ ncnet    exe
             │             │             ║ clp2dib  exe │ nc       hlp │ ncnet    msg
             │             │             ║ cmpsrv   scx │ nc       ico │ ncpscrip hdr
wina20.386        9349   5-31-94   6:22a ║ ..            ►UP--DIR◄  7-21-17   3:31p
C:\NC>_
1Help  2Menu  3View  4Edit  5Copy  6RenMov 7Mkdir 8Delete 9PullDn 10Quit
```

ASCII Art

```
                               /T /I
                              / |/ | .-~/
                          T\ Y  I  |/  /  _
         /T               | \I  |  I  Y.-~/
        I l   /I       T\ |  |  l  |  T  /
     T\ |  \ Y l  /T   | \I  l   \ `  l Y
 __  | \l   \l  \I l __l  l   \   `  _. |
 \ ~-l  `\   `\  \  \\ ~\  \   `. .-~   |
  \   ~-. "-.  `  \  ^._ ^. "-.  /  \   |
.--~-._  ~-  `  _  ~-_.-"-." ._ /._ ." ./
 >--.  ~-.   ._  ~>-"    "\\   7   7   ]
^.___~"--._    ~-{  .-~ .  `\ Y . /    |
 <__ ~"-.  ~       /_/   \   \I  Y   : |
   ^-.__           ~(_/   \   >._:   | l______
       ^--.,___.-~"  /_/   !  `-.~"--l_ /     ~"-.
              (_/ .  ~(   /'     "~"--,Y   -=b-. _)
               (_/ .  \  :           / l      c"~o \
                \ /    `.    .     .^   \_.-~"~--.  )
                 (_/ .   `  /     /       !       )/
                  / / _.   '.   .':      /        '
                  ~(_/ .   /    _  `  .-<_
                    /_/ . ' .-~" `.  / \  \          ,z=.
                    ~( /   '  :   | K   "-.~-.______//
                      "-,.    l   I/ \_    __{--->._(==.
                       //(     \  <    ~"~"     //
                      /' /\     \  \     ,v=.  ((
                    .^. / /\     "  }__ //===-  `
                   / / ' '  "-.,__ {---(==-
                 .^ '       :  T  ~"   ll       -Row
                / .  .  . : | :!        \\
               (_/  /   | | j-"          ~^
                 ~-<_(_.^-~"
```

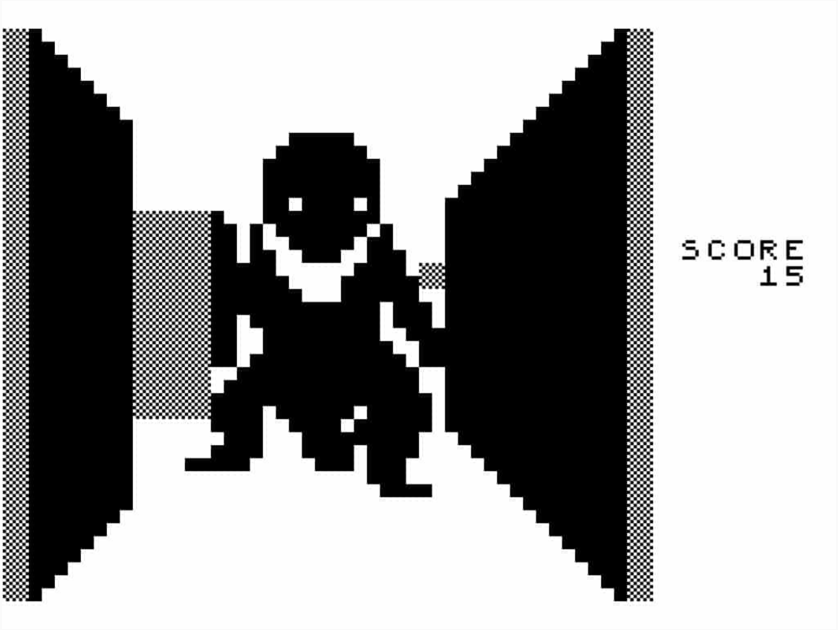
ASCII Art

Rogue (1980)

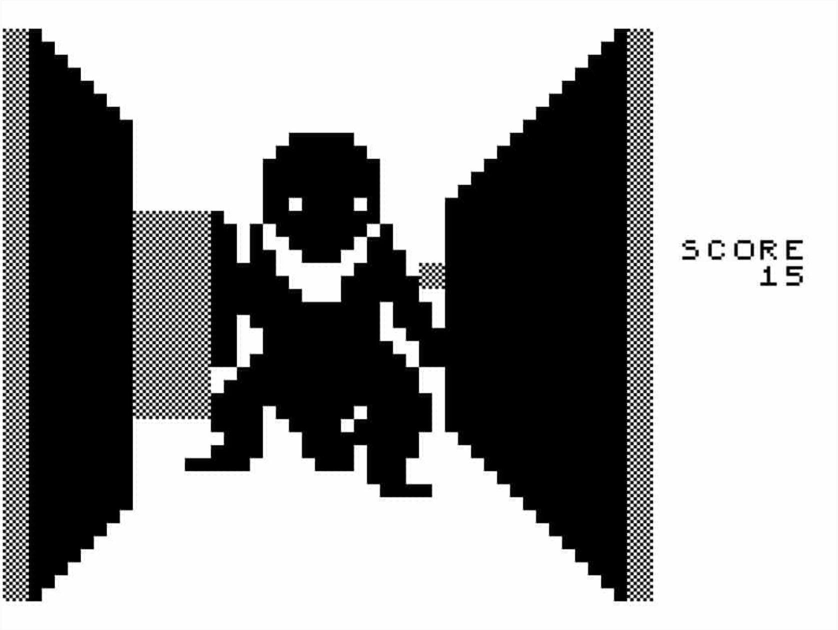


```
Level:4    Hits:29(29)   Str:16(16)    Gold:718    Armor:5   Exp:4/76
```

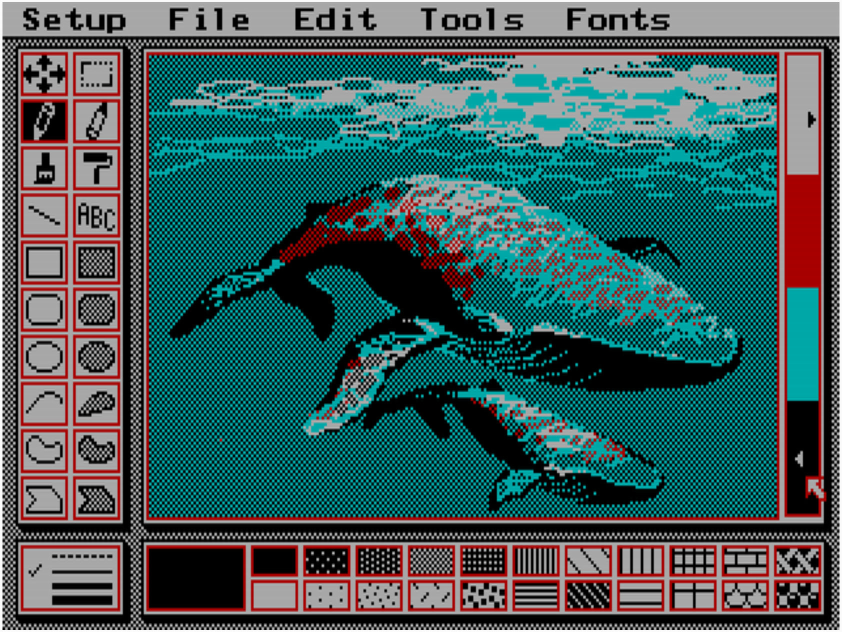
3D monster maze (1981)



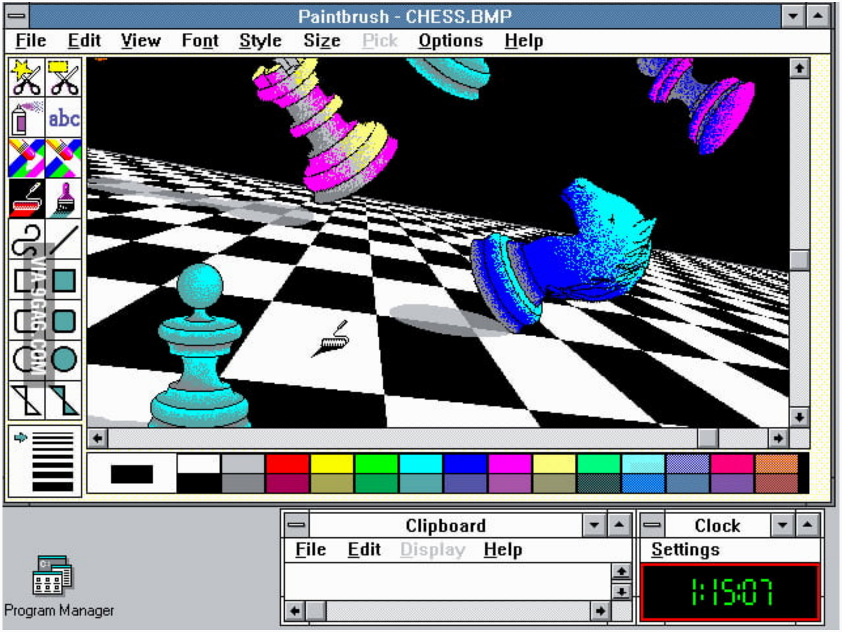
3D monster maze (1981)



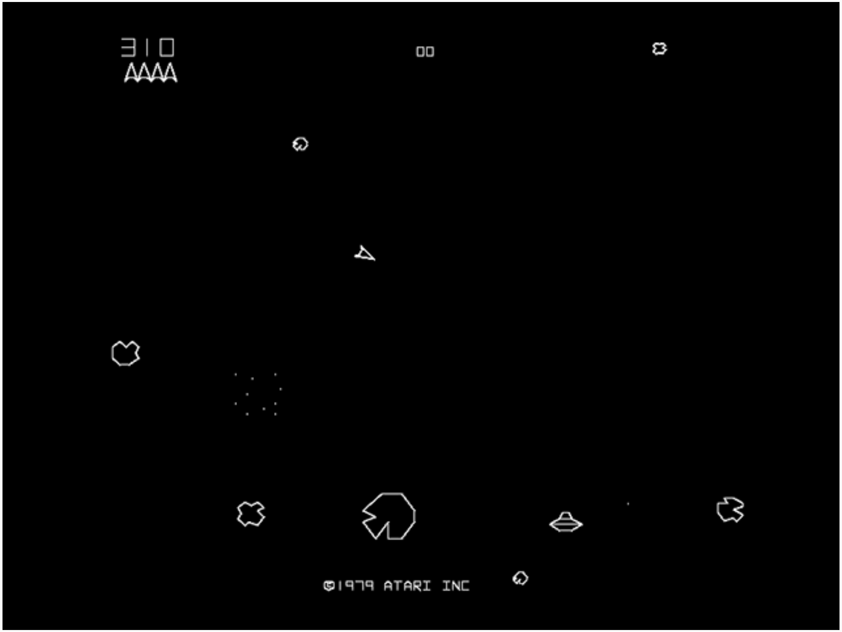
CGA graphical user interface



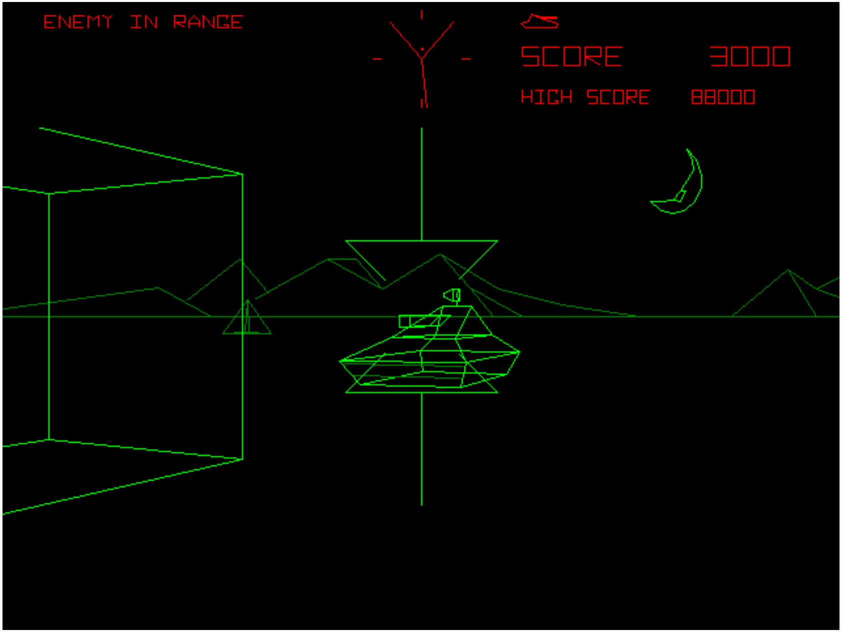
SVGA graphical user interface



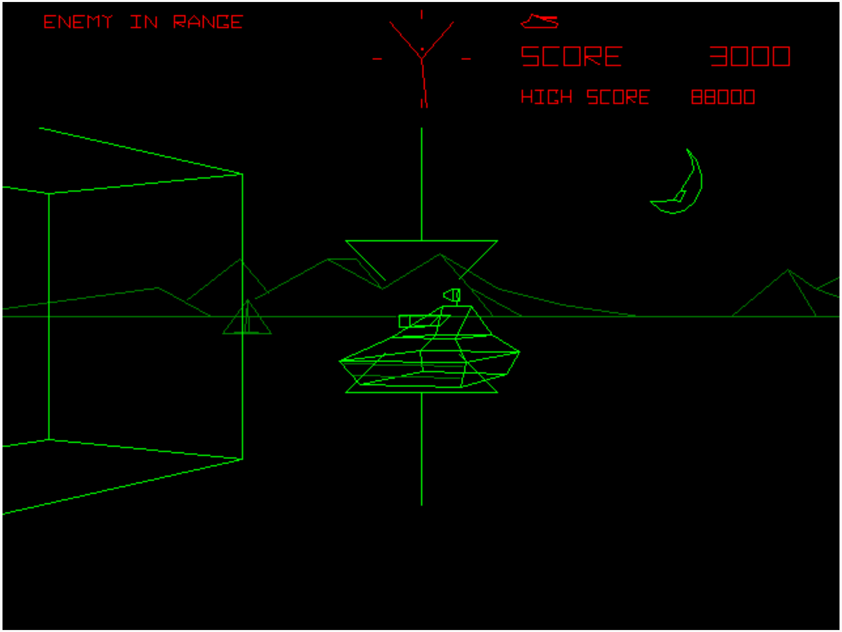
Asteroids (1979)



Battlezone (1980)



Battlezone (1980)



Graphics primitives

The screen can be seen as a $W \times H$ matrix of pixels

- Pixel at coordinates (x, y) has color c

Graphics primitives

The screen can be seen as a $W \times H$ matrix of pixels

- Pixel at coordinates (x, y) has color c

Assume we have two graphics primitives

```
set_pixel(img, x, y, c)
hline(img, x1, x2, y, c)
```

Graphics primitives

The screen can be seen as a $W \times H$ matrix of pixels

- Pixel at coordinates (x, y) has color c

Assume we have two graphics primitives

```
set_pixel(img, x, y, c)
hline(img, x1, x2, y, c)
```

How do we

- draw an arbitrary line?

Graphics primitives

The screen can be seen as a $W \times H$ matrix of pixels

- Pixel at coordinates (x, y) has color c

Assume we have two graphics primitives

```
set_pixel(img, x, y, c)
hline(img, x1, x2, y, c)
```

How do we

- draw an arbitrary line?
- fill an arbitrary polygon?

Line drawing

Bresenham, J. E. 1965. “Algorithm for computer control of a digital plotter”. *IBM Systems Journal.*

Line drawing

Bresenham, J. E. 1965. “Algorithm for computer control of a digital plotter”. *IBM Systems Journal.*

Integer endpoints

Line drawing

Bresenham, J. E. 1965. “Algorithm for computer control of a digital plotter”. *IBM Systems Journal.*

Integer endpoints

Incremental

- No divisions
- (almost) No multiplications

Line drawing

Bresenham, J. E. 1965. “Algorithm for computer control of a digital plotter”. *IBM Systems Journal*.

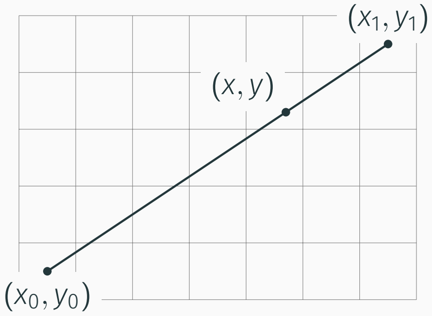
Integer endpoints

Incremental

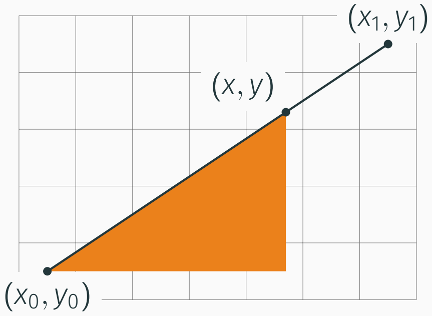
- No divisions
- (almost) No multiplications

Leave no gaps

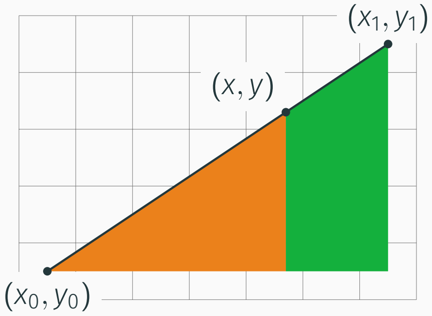
Line drawing



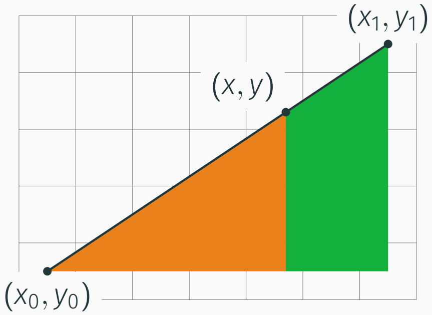
Line drawing



Line drawing

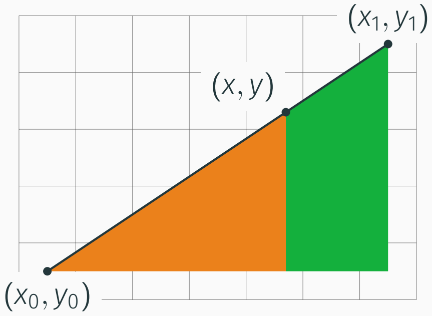


Line drawing



$$\frac{x - x_0}{y - y_0} = \frac{x_1 - x_0}{y_1 - y_0}$$

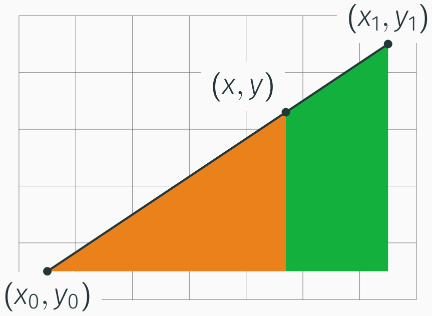
Line drawing



$$\frac{x - x_0}{y - y_0} = \frac{x_1 - x_0}{y_1 - y_0}$$

$$(y_1 - y_0)(x - x_0) - (x_1 - x_0)(y - y_0) = 0$$

Line drawing

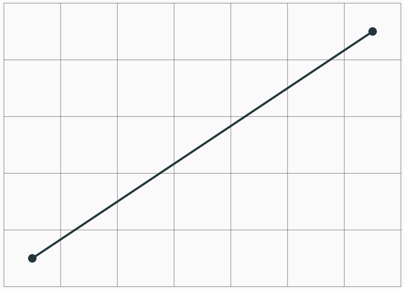


$$\frac{x - x_0}{y - y_0} = \frac{x_1 - x_0}{y_1 - y_0}$$

$$(y_1 - y_0)(x - x_0) - (x_1 - x_0)(y - y_0) = 0$$

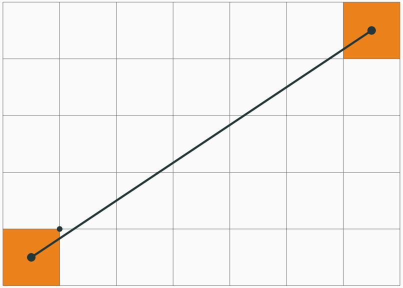
$$\boxed{\ell(x, y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

Line drawing



$$\boxed{\ell(x, y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

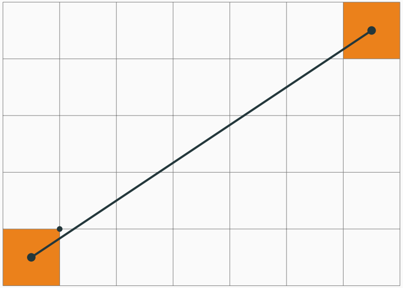
Line drawing



$$\boxed{\ell(x, y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

Line drawing

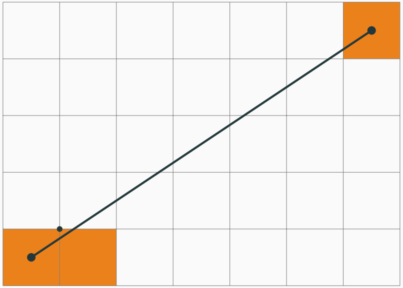


$$\boxed{\ell(x,y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0,y_0) = \ell(x_1,y_1) = 0 \qquad \ell(x_0+\tfrac{1}{2}, y_0+\tfrac{1}{2}) = dy - dx$$

$$\ell(x+1,y) - \ell(x,y) = 2dy \qquad \ell(x,y+1) - \ell(x,y) = -2dx$$

Line drawing

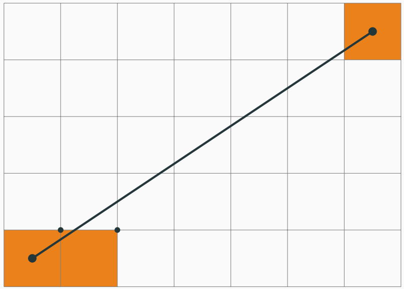


$$\boxed{\ell(x, y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

$$\ell(x + 1, y) - \ell(x, y) = 2dy \qquad \ell(x, y + 1) - \ell(x, y) = -2dx$$

Line drawing

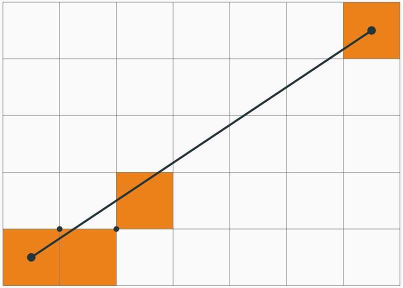


$$\boxed{\ell(x, y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

$$\ell(x + 1, y) - \ell(x, y) = 2dy \qquad \ell(x, y + 1) - \ell(x, y) = -2dx$$

Line drawing

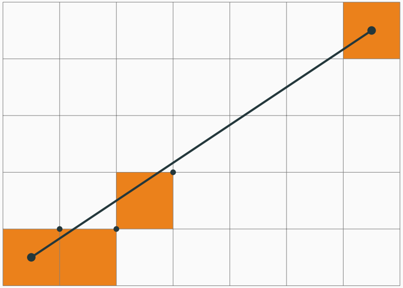


$$\boxed{\ell(x,y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

$$\ell(x+1, y) - \ell(x,y) = 2dy \qquad \ell(x, y+1) - \ell(x,y) = -2dx$$

Line drawing

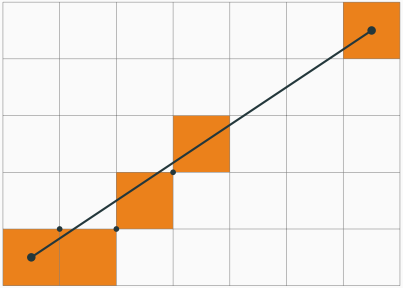


$$\boxed{\ell(x,y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

$$\ell(x+1, y) - \ell(x,y) = 2dy \qquad \ell(x, y+1) - \ell(x,y) = -2dx$$

Line drawing

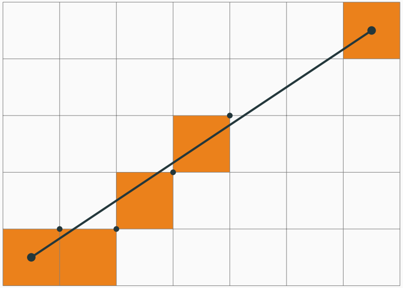


$$\boxed{\ell(x,y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0,y_0) = \ell(x_1,y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

$$\ell(x+1,y) - \ell(x,y) = 2dy \qquad \ell(x,y+1) - \ell(x,y) = -2dx$$

Line drawing

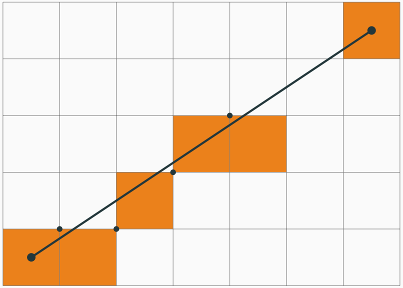


$$\boxed{\ell(x,y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

$$\ell(x+1, y) - \ell(x,y) = 2dy \qquad \ell(x, y+1) - \ell(x,y) = -2dx$$

Line drawing

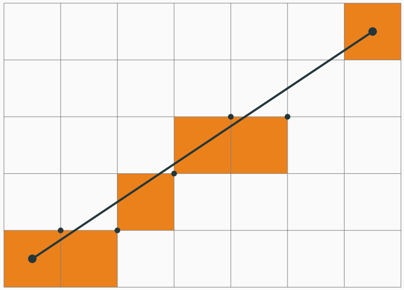


$$\boxed{\ell(x, y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

$$\ell(x + 1, y) - \ell(x, y) = 2dy \qquad \ell(x, y + 1) - \ell(x, y) = -2dx$$

Line drawing

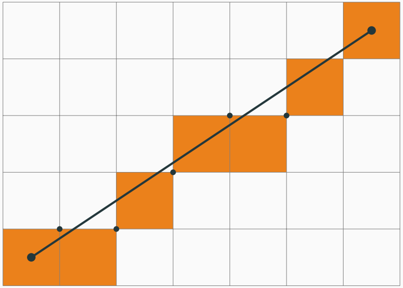


$$\boxed{\ell(x, y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

$$\ell(x + 1, y) - \ell(x, y) = 2dy \qquad \ell(x, y + 1) - \ell(x, y) = -2dx$$

Line drawing

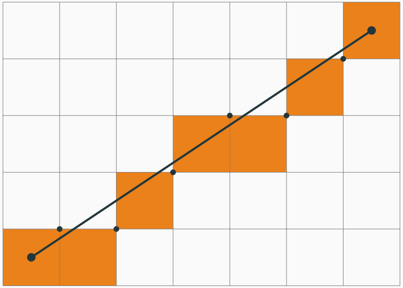


$$\boxed{\ell(x,y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

$$\ell(x+1, y) - \ell(x,y) = 2dy \qquad \ell(x, y+1) - \ell(x,y) = -2dx$$

Line drawing



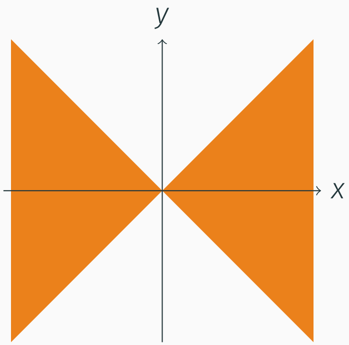
$$\boxed{\ell(x, y) = 2dy\,(x - x_0) - 2dx\,(y - y_0) = 0}$$

$$\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \qquad \ell(x_0 + \tfrac{1}{2}, y_0 + \tfrac{1}{2}) = dy - dx$$

$$\ell(x + 1, y) - \ell(x, y) = 2dy \qquad \ell(x, y + 1) - \ell(x, y) = -2dx$$

Line drawing

```
local function linex(img, x1, y1, x2, y2, set_pixel)
  local dx, dy = x2 - x1, y2 - y1
  local sx, sy = sign(dx), sign(dy)
  dx, dy = sx * dx, sy * dy
  assert(dx >= dy)
  local f = dy - dx
  dx, dy = dx*2, dy*2
  local x, y = x1, y1
  set_pixel(img, x, y)
  while x ~= x2 do
    x = x + sx
    f = f + dy
    if f > 0 then
      f = f - dx
      y = y + sy
    end
    set_pixel(img, x, y)
  end
end
```



Line drawing

```lua
local function set_pixelyx(img, y, x)
  set_pixel(img, x, y)
end

function line(img, x1, y1, x2, y2)
  local dx, dy = math.abs(x2-x1), math.abs(y2-y1)
  if dx > dy then
    linex(img, x1, y1, x2, y2, set_pixel)
  else
    linex(img, y1, x1, y2, x2, set_pixelyx)
  end
end
```

Polygon filling

(?) Wylie, C. et al. 1967. “A hidden surface algorithm for computer generated halftone pictures”. *Proceedings Fall Joint Computer Conference.*

Integer endpoints

Incremental

- No divisions
- (almost) No multiplications

Leave no gaps

Polygon filling

(?) Wylie, C. et al. 1967. “A hidden surface algorithm for computer generated halftone pictures”. *Proceedings Fall Joint Computer Conference.*

Integer endpoints

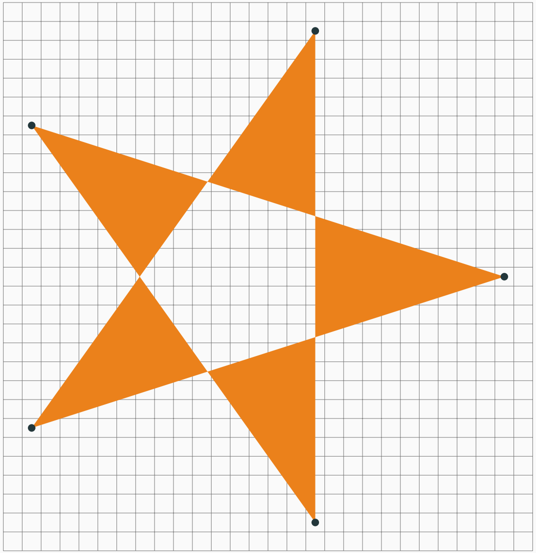
Incremental

- No divisions
- (almost) No multiplications

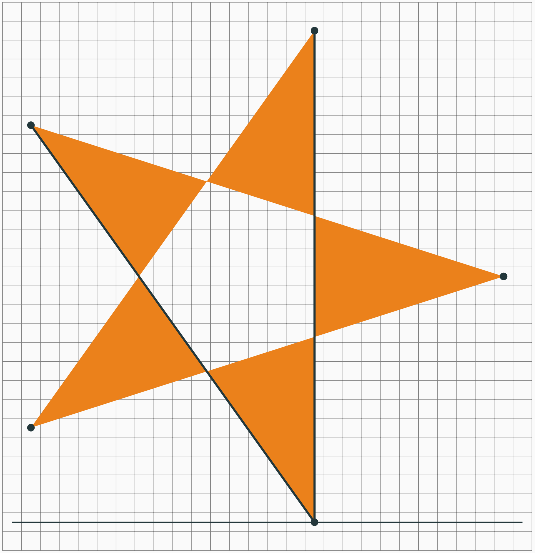
Leave no gaps

Use spatial coherence

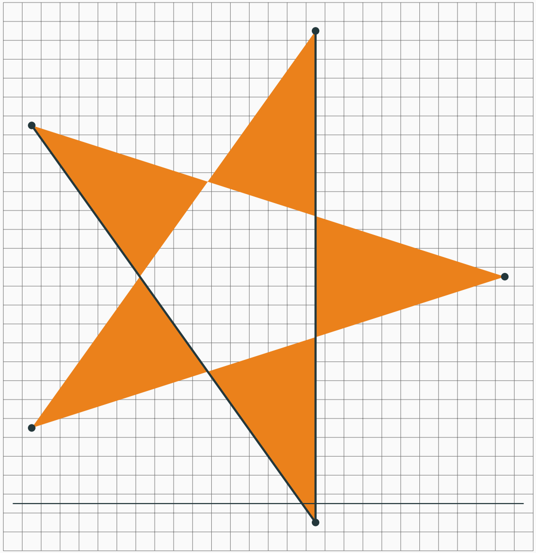
Polygon filling



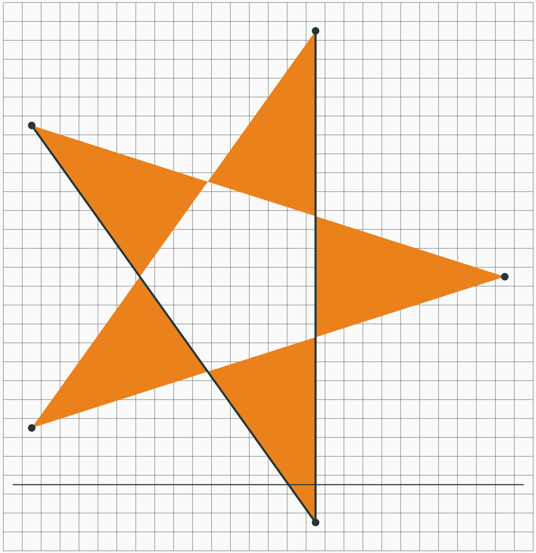
Polygon filling



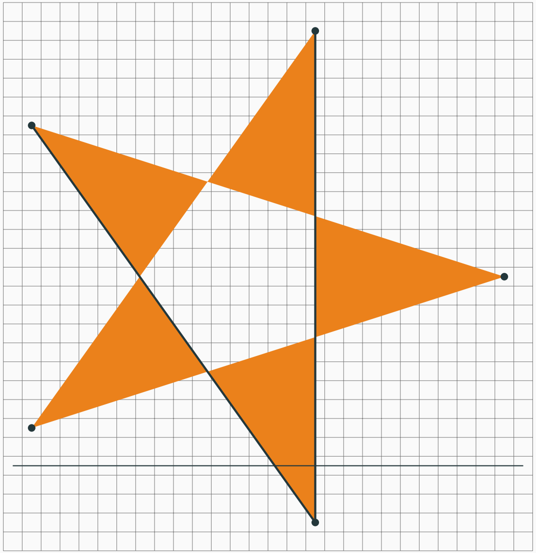
Polygon filling



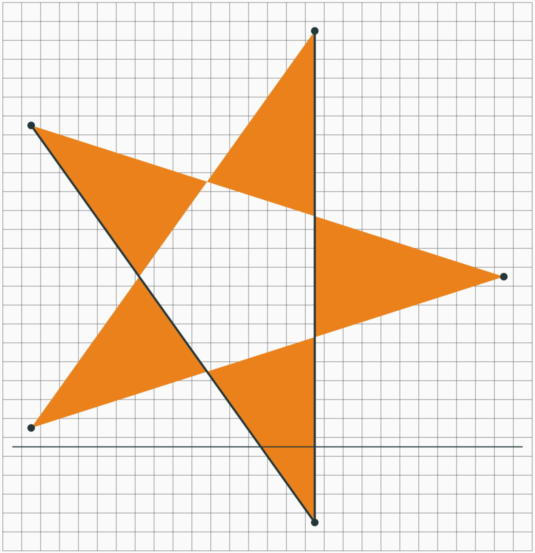
Polygon filling



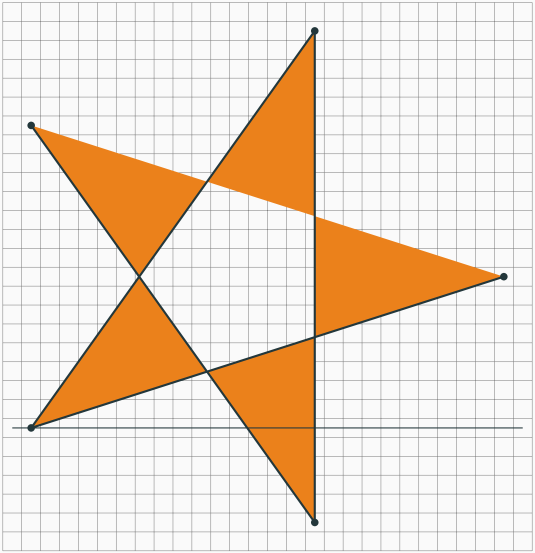
Polygon filling



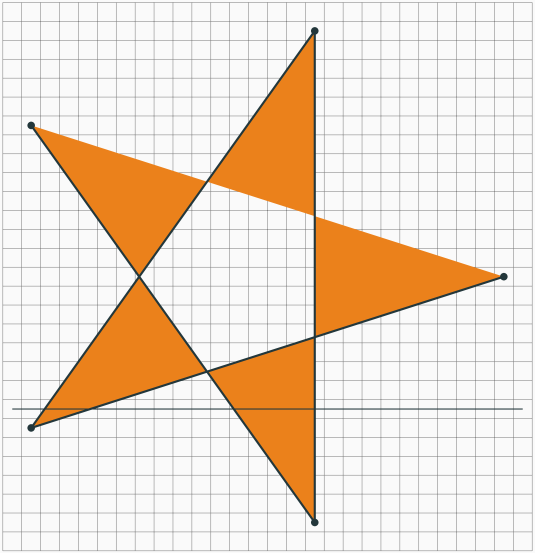
Polygon filling



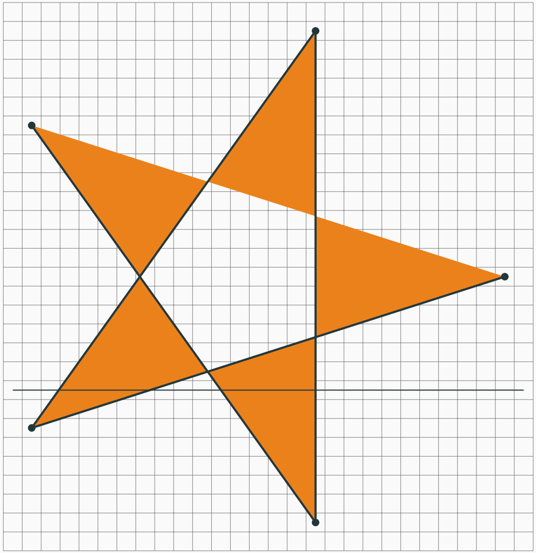
Polygon filling



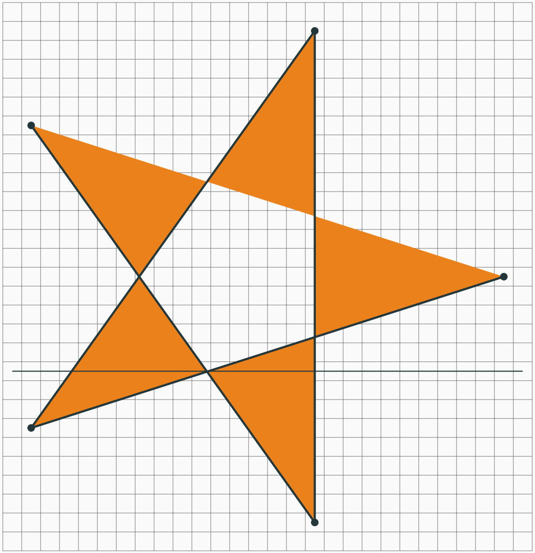
Polygon filling



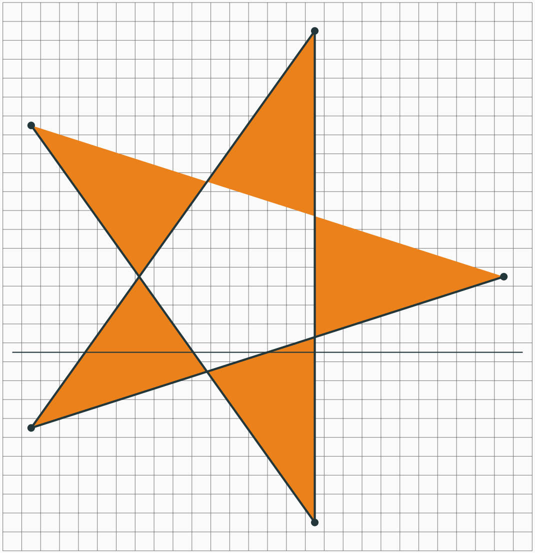
Polygon filling



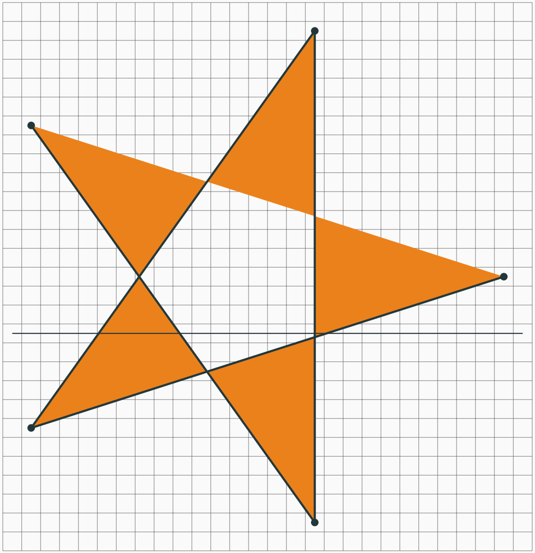
Polygon filling



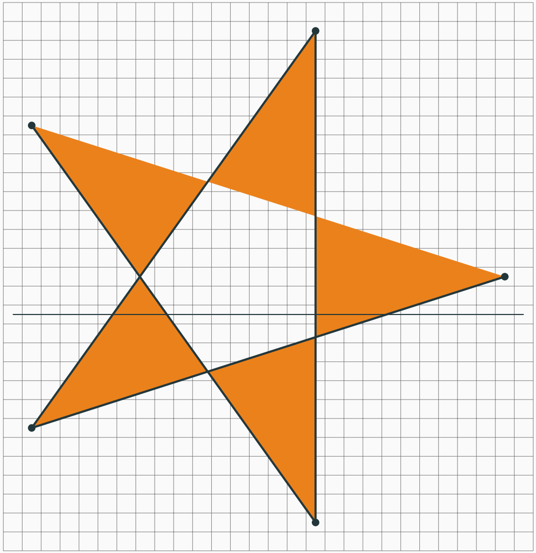
Polygon filling



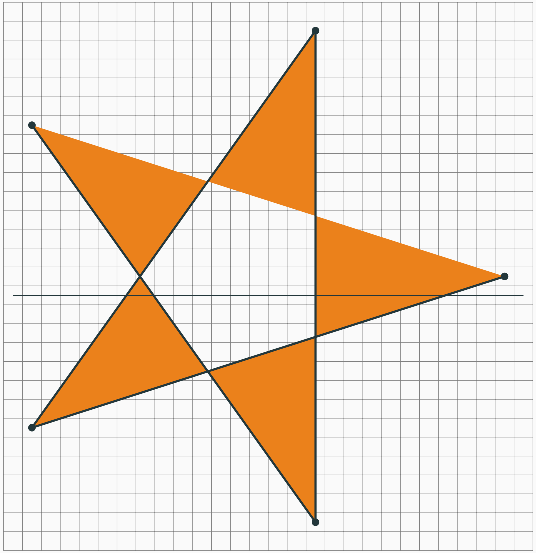
Polygon filling



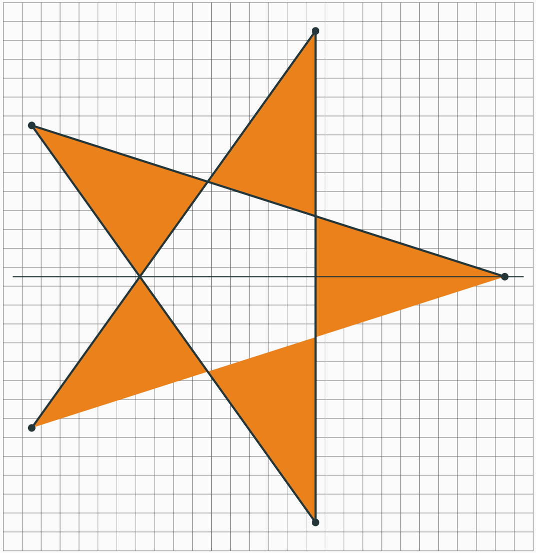
Polygon filling



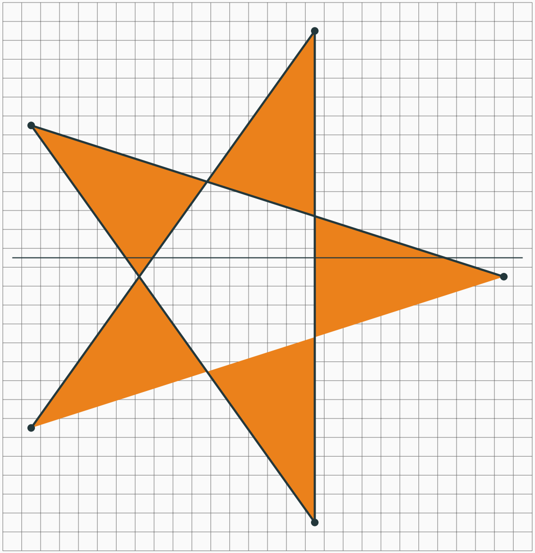
Polygon filling



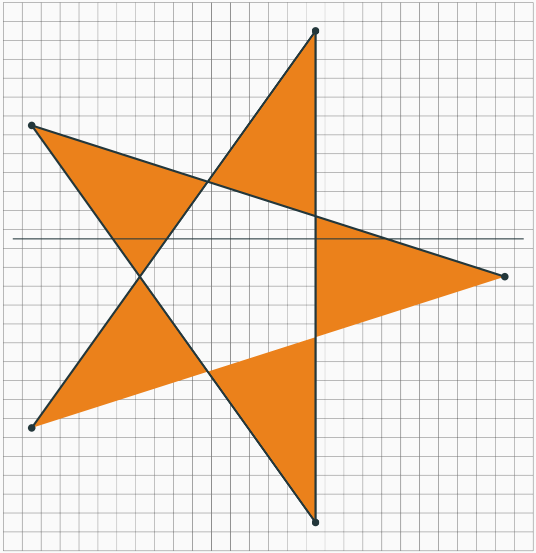
Polygon filling



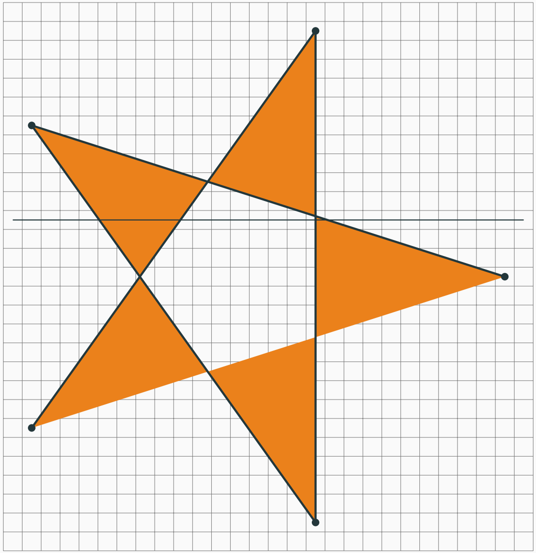
Polygon filling



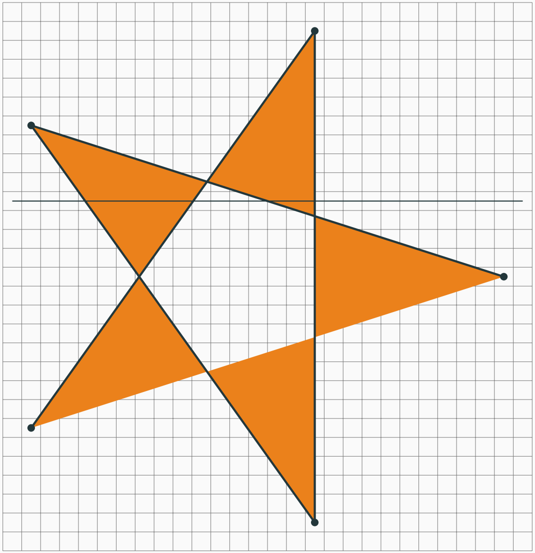
Polygon filling



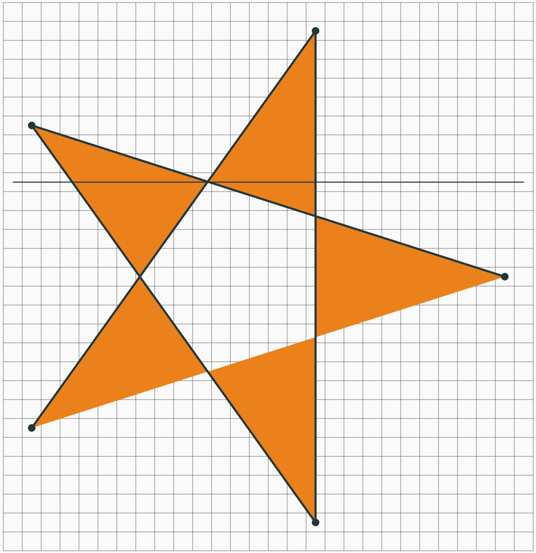
Polygon filling



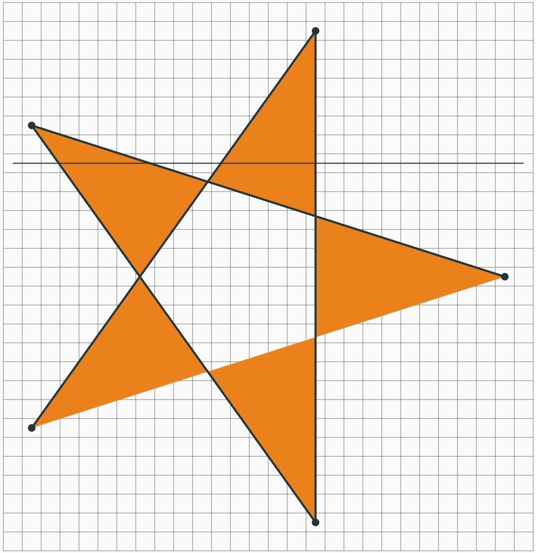
Polygon filling



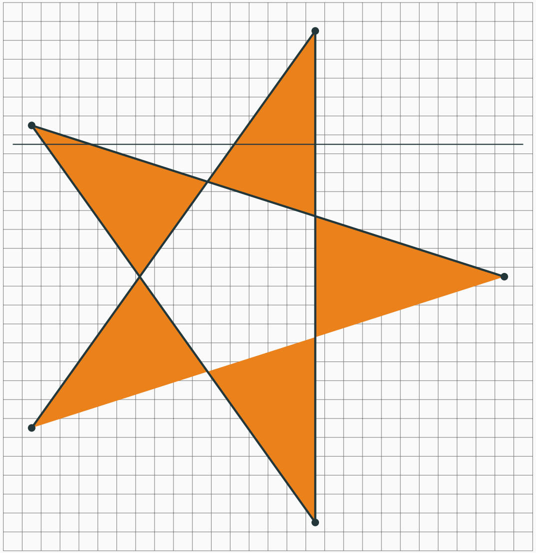
Polygon filling



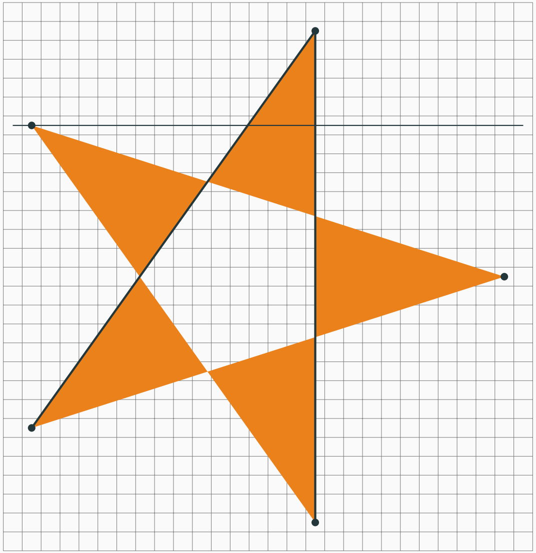
Polygon filling



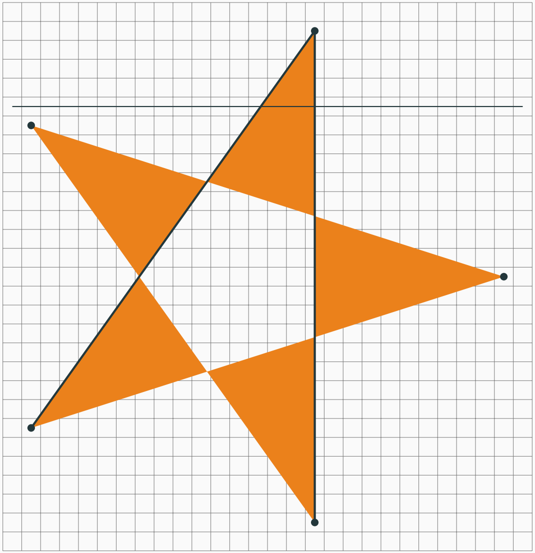
Polygon filling



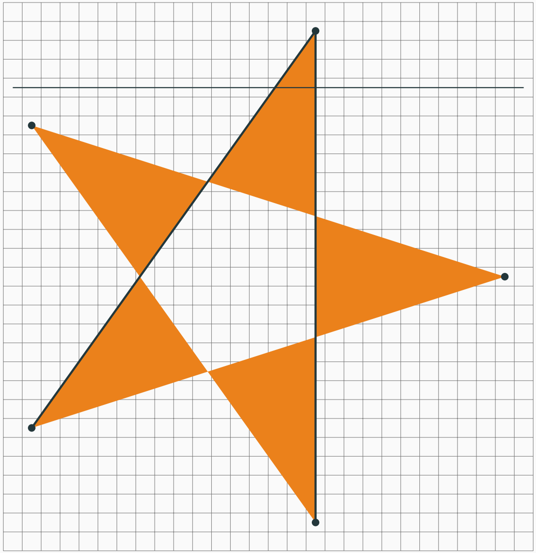
Polygon filling



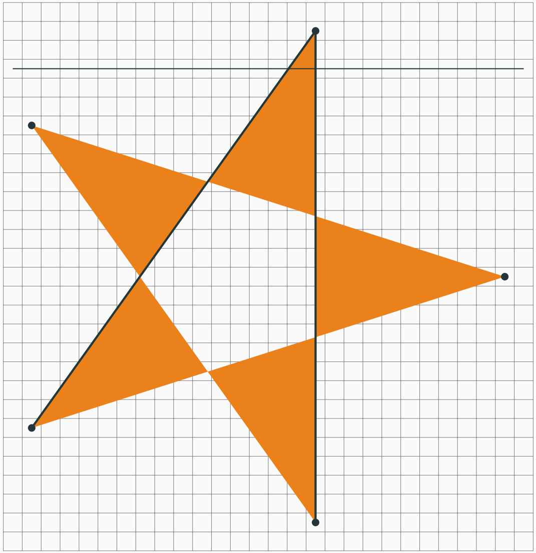
Polygon filling



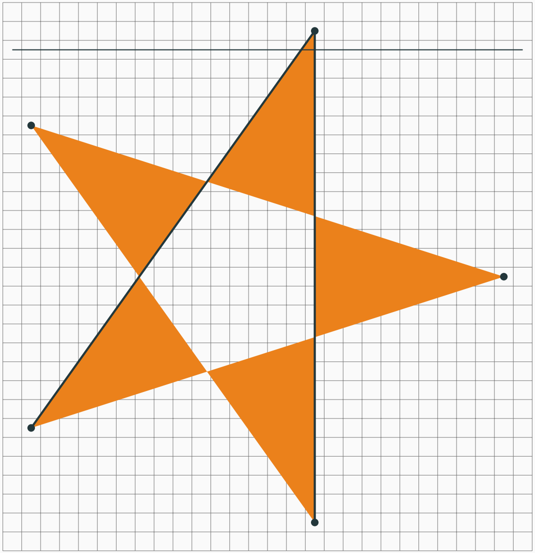
Polygon filling



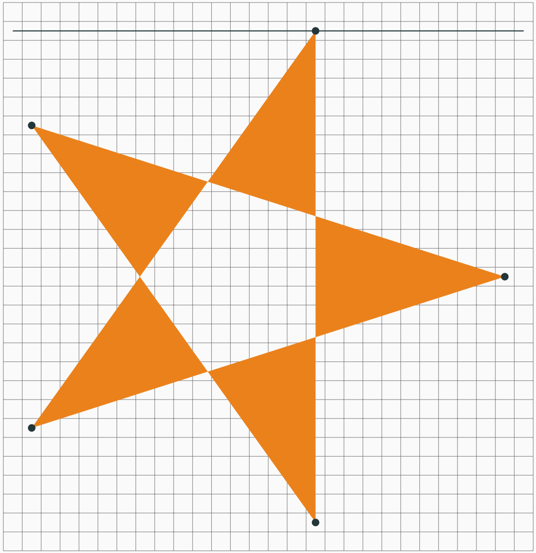
Polygon filling



Polygon filling



Polygon filling



Polygon filling