2D COMPUTER GRAPHICS

Diego Nehab
Summer 2019

IMPA
INTRODUCTION
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...
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
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-2019-0-2dcg
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
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
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
We will focus on vector graphics

- Layers of colored shapes
- PDF, SVG, AI, EPS, CGM, etc
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
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
Images have a fixed, finite resolution
Images have a fixed, finite resolution
Images have a fixed, finite resolution
Images have a fixed, finite resolution
Resolution and scalability

Images have a fixed, finite resolution

Vector graphics are scalable
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 filled 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 simplifications that come from this key observation lead to the fat-clipping algorithm, which does not require a stack (or recursion).

Flat clipping

The intuition is that, during a union operation, the first 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 first 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 finite-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 first two state variables keep track of the current clipping nesting depth, \( d \), and the number of transitions away from S, \( u \).

The remaining transitions are between P and SA. If the machine finds 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 \( \times \) 3 projective transformation to be applied to the sample coordinates. Experienced users can design arbitrary warping functions in CUDA. Since the pipeline

![State transition diagram for the finite-state machine of the fat-clipping algorithm.](image-url)
Vector graphics are everywhere
VECTOR GRAPHICS ARE EVERYWHERE
Evaluation
Grading

Assignments: 60%
Exams: 30%
Participation: 10%
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 $p$ YUAs with 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.

<table>
<thead>
<tr>
<th>Alg.</th>
<th>Step complexity</th>
<th>Max. # of threads</th>
<th>Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>RP</td>
<td>$\frac{b}{w} \cdot 4p$</td>
<td>$\frac{w}{b}$</td>
<td>$8w$</td>
</tr>
<tr>
<td>2</td>
<td>$\frac{b}{w} \cdot (4r - 1) \cdot \frac{w}{b}$</td>
<td>$\frac{w}{b}$</td>
<td>$(5 - 16 \frac{r}{b})w$</td>
</tr>
<tr>
<td>4</td>
<td>$\frac{b}{w} \cdot (4r - 1) \cdot \frac{w}{b}$</td>
<td>$\frac{w}{b}$</td>
<td>$(3 - 18 \frac{r}{b})w$</td>
</tr>
<tr>
<td>5</td>
<td>$\frac{b}{w} \cdot (4r - 1) \cdot \frac{w}{b}$</td>
<td>$\frac{w}{b}$</td>
<td>$(3 - 22 \frac{r}{b})w$</td>
</tr>
<tr>
<td>SAT</td>
<td>$\frac{b}{w} \cdot (4 - \frac{r}{b})$</td>
<td>$\frac{w}{b}$</td>
<td>$(3 + \frac{r}{b})w$</td>
</tr>
</tbody>
</table>
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

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 confirm 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
Properties preserved by a group of transformations

- Euclidean
- Affine
- Projective
Properties preserved by a group of transformations
• Euclidean
• Affine
• Projective

Representations for points, vectors, and transformations
Properties preserved by a group of transformations

- Euclidean
- Affine
- Projective

Representations for points, vectors, and transformations

Focus on using transformations to solve geometric problems
Seminal work by Warnock and Wyatt [1982]

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

Rational Bézier curves
  - Required for circular arcs
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
Representation of paths
  • Converting other primitives to paths
Representation of paths
  • Converting other primitives to paths

Floating-point representation and properties
  • Numerical issues
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
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
Radiometry
  • Physics of light
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
Moving towards an implicit test for intersections
  • Avoid costly root-finding
Moving towards an implicit test for intersections
  • Avoid costly root-finding

Implicit form of parametric polynomial curves
Moving towards an implicit test for intersections
  • Avoid costly root-finding

Implicit form of parametric polynomial curves

Resultant
  • Sylvester form
  • Cayley-Bezout form
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
Planar parametric curves
Planar parametric curves

Rectification, and arc length
Planar parametric curves
Rectification, and arc length
Arc-length parameterization
Planar parametric curves
Rectification, and arc length
Arc-length parameterization
Curvature, offset, and evolute
Planar parametric curves
Rectification, and arc length
Arc-length parameterization
Curvature, offset, and evolute
Inflections
Planar parametric curves
Rectification, and arc length
Arc-length parameterization
Curvature, offset, and evolute
Inflections
Double-points
Planar parametric curves
Rectification, and arc length
Arc-length parameterization
Curvature, offset, and evolute
Inflections
Double-points
Stroking
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
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
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
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
Proper definition of digital image
Proper definition of digital image
Rendering as an approximation problem
Proper definition of *digital image*

Rendering as an approximation problem

Ideal sampling theory
  - Introduction to Fourier transforms
  - Whittaker-Nyquist-Kotelnikov-Shannon theorem
  - 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
The anti-aliasing integral

- Analytic solutions are not possible
The anti-aliasing integral
  • Analytic solutions are not possible

Conflation of coverage with opacity
  • Problem with correlated mattes
  • Problem with gamma correction
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
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
Classical acceleration data structures
  • Space partition
    • Quadtree, K-d tree, and BSP
Classical acceleration data structures

- Space partition
  - Quadtree, K-d tree, and BSP
- Bounding volume hierarchy
  - R-tree
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
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
History of typesetting
  • Calligraphy
  • Gutenberg’s printing press
History of typesetting
  • Calligraphy
  • Gutenberg’s printing press

Unicode
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
Definition

- Dashing and decorations
Class 18: Stroked primitives

Definition
  • Dashing and decorations

Two different approaches to rendering
  • Using distance to generator
Definition

- Dashing and decorations

Two different approaches to rendering

- Using distance to generator
- Converting to filled primitives
Definition
  • Dashing and decorations

Two different approaches to rendering
  • Using distance to generator
  • Converting to filled primitives

Two conversion methods
  • Flattening the generator
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
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
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
Blur

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

Clipping
- Per pixel or per sample
- Vatti’s algorithm [1992]
Active edge list algorithm [1967]

NVPR [2012]
Old-school graphics
Computer Graphics

Mathematics

Hardware

Software
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: $80 \times 25$ with $8 \times 8$ characters
• Graphics: $320 \times 200$ 4 bpp, $640 \times 200$ 1 bpp

VGA (Video Graphics Array) (1987)
• 256KB of video memory
• Text mode $80 \times 25$ with $9 \times 16$ characters
• Graphics: $320 \times 240$ 8 bpp, $640 \times 480$ 4 bpp

SVGA (Super Video Graphics Array) (1989)
• Graphics: $800 \times 600$ 4 bpp, $640 \times 480$ 8 bpp
## CODE PAGE 437

<table>
<thead>
<tr>
<th></th>
<th><em>0</em></th>
<th><em>1</em></th>
<th><em>2</em></th>
<th><em>3</em></th>
<th><em>4</em></th>
<th><em>5</em></th>
<th><em>6</em></th>
<th><em>7</em></th>
<th><em>8</em></th>
<th><em>9</em></th>
<th><em>A</em></th>
<th><em>B</em></th>
<th><em>C</em></th>
<th><em>D</em></th>
<th><em>E</em></th>
<th><em>F</em></th>
</tr>
</thead>
<tbody>
<tr>
<td><em>0</em></td>
<td>NUL</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>1</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>2</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>3</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>4</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>5</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>6</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>7</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>8</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>9</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>A</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>B</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>C</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>D</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>E</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td><em>F</em></td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>A:1: 'EMP</td>
<td>Worksheet</td>
<td>Range</td>
<td>Copy</td>
<td>Move</td>
<td>File</td>
<td>Print</td>
<td>Graph</td>
<td>Data</td>
<td>System</td>
<td>Quit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td>-------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>-------</td>
<td>-------</td>
<td>------</td>
<td>--------</td>
<td>------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EMP NAME</td>
<td>DEPNO</td>
<td>JOB</td>
<td>YEARS</td>
<td>SALARY</td>
<td>BONUS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1777 Azibad</td>
<td>4000 Sales</td>
<td>2</td>
<td>40000</td>
<td>10000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>81964 Brown</td>
<td>6000 Sales</td>
<td>3</td>
<td>45000</td>
<td>10000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40370 Burns</td>
<td>6000 Mgr</td>
<td>4</td>
<td>75000</td>
<td>25000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50786 Caeser</td>
<td>7000 Mgr</td>
<td>3</td>
<td>65000</td>
<td>25000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49692 Curly</td>
<td>3000 Mgr</td>
<td>5</td>
<td>65000</td>
<td>20000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>34791 Dabarro</td>
<td>7000 Sales</td>
<td>2</td>
<td>45000</td>
<td>10000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>84984 Daniels</td>
<td>1000 President</td>
<td>8</td>
<td>150000</td>
<td>10000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>59937 Dempsey</td>
<td>3000 Sales</td>
<td>3</td>
<td>40000</td>
<td>10000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>51515 Donovan</td>
<td>3000 Sales</td>
<td>2</td>
<td>30000</td>
<td>5000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48338 Fields</td>
<td>4000 Mgr</td>
<td>5</td>
<td>70000</td>
<td>25000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>91574 Fiklore</td>
<td>1000 Admin</td>
<td>8</td>
<td>35000</td>
<td>---</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>64596 Fine</td>
<td>5000 Mgr</td>
<td>3</td>
<td>75000</td>
<td>25000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13729 Green</td>
<td>1000 Mgr</td>
<td>5</td>
<td>90000</td>
<td>25000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>55957 Hernann</td>
<td>4000 Sales</td>
<td>4</td>
<td>50000</td>
<td>10000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31619 Hodgdon</td>
<td>5000 Sales</td>
<td>2</td>
<td>40000</td>
<td>10000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1773 Howard</td>
<td>2000 Mgr</td>
<td>3</td>
<td>80000</td>
<td>25000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2165 Hugh</td>
<td>1000 Admin</td>
<td>5</td>
<td>30000</td>
<td>---</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23907 Johnson</td>
<td>1000 VP</td>
<td>1</td>
<td>100000</td>
<td>50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7166 Laflare</td>
<td>2000 Sales</td>
<td>2</td>
<td>35000</td>
<td>5000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DATA.WK3
VGA textual user interface
3D MONSTER MAZE (1981)
3D MONSTER MAZE (1981)

SCORE
15
CGA graphical user interface
SVGA GRAPHICAL USER INTERFACE
ASTEROIDS (1979)
BATTLEZONE (1980)

ENEMY IN RANGE

SCORE 3000

HIGH SCORE 88000
BATTLEZONE (1980)

ENEMY IN RANGE

SCORE 3000
HIGH SCORE 88000
Graphics primitives

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

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

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

Integer endpoints

Integer endpoints

Incremental

• No divisions
• (almost) No multiplications

**Integer endpoints**

**Incremental**
- No divisions
- (almost) No multiplications

**Leave no gaps**
Line drawing

\[(x, y) = (x_0, y_0) - (x_1, y_1)\]

\[\ell(x, y) = 2dy(x - x_0) - 2dx(y - y_0) = 0\]
\[ \ell(x, y) = 2 \frac{dy}{dx} = 0 \]
The line segment between points $(x_0, y_0)$ and $(x_1, y_1)$ can be described by the equation:

$$\ell(x, y) = 2\, dy - 2\, dx = 0$$

This equation represents the line segment in the 2D plane, where $dy$ and $dx$ are the differentials of $y$ and $x$, respectively.
\[
\frac{x - x_0}{y - y_0} = \frac{x_1 - x_0}{y_1 - y_0}
\]
\[
\frac{x - x_0}{y - y_0} = \frac{x_1 - x_0}{y_1 - y_0}
\]

\[
(y_1 - y_0)(x - x_0) - (x_1 - x_0)(y - y_0) = 0
\]
\[
\frac{x - x_0}{y - y_0} = \frac{x_1 - x_0}{y_1 - y_0}
\]

\[
(y_1 - y_0)(x - x_0) - (x_1 - x_0)(y - y_0) = 0
\]

\[
\ell(x, y) = 2dy (x - x_0) - 2dx (y - y_0) = 0
\]
\[ \ell(x, y) = 2dy(x - x_0) - 2dx(y - y_0) = 0 \]
\[ \ell(x, y) = 2dy \ (x - x_0) - 2dx \ (y - y_0) = 0 \]

\[ \ell(x_0, y_0) = \ell(x_1, y_1) = 0 \quad \ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) = dy - dx \]
\[
\ell(x, y) = 2dy (x - x_0) - 2dx (y - y_0) = 0
\]

\[
\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \quad \ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) = dy - dx
\]

\[
\ell(x + 1, y) - \ell(x, y) = 2dy \quad \ell(x, y + 1) - \ell(x, y) = -2dx
\]
\[ \ell(x, y) = 2dy (x - x_0) - 2dx (y - y_0) = 0 \]

\[ \ell(x_0, y_0) = \ell(x_1, y_1) = 0 \quad \ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) = dy - dx \]

\[ \ell(x + 1, y) - \ell(x, y) = 2dy \quad \ell(x, y + 1) - \ell(x, y) = -2dx \]
\[ \ell(x, y) = 2dy (x - x_0) - 2dx (y - y_0) = 0 \]

\[ \ell(x_0, y_0) = \ell(x_1, y_1) = 0 \quad \ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) = dy - dx \]

\[ \ell(x + 1, y) - \ell(x, y) = 2dy \quad \ell(x, y + 1) - \ell(x, y) = -2dx \]
\[ \ell(x, y) = 2dy \ (x - x_0) - 2dx \ (y - y_0) = 0 \]

\[ \ell(x_0, y_0) = \ell(x_1, y_1) = 0 \quad \ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) = dy - dx \]

\[ \ell(x + 1, y) - \ell(x, y) = 2dy \quad \ell(x, y + 1) - \ell(x, y) = -2dx \]
\[
\ell(x, y) = 2dy \ (x - x_0) - 2dx \ (y - y_0) = 0
\]

\[
\ell(x_0, y_0) = \ell(x_1, y_1) = 0 \quad \ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) = dy - dx
\]

\[
\ell(x + 1, y) - \ell(x, y) = 2dy \quad \ell(x, y + 1) - \ell(x, y) = -2dx
\]
\[ \ell(x, y) = 2dy (x - x_0) - 2dx (y - y_0) = 0 \]

\[ \ell(x_0, y_0) = \ell(x_1, y_1) = 0 \quad \ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) = dy - dx \]

\[ \ell(x + 1, y) - \ell(x, y) = 2dy \quad \ell(x, y + 1) - \ell(x, y) = -2dx \]
\[ \ell(x, y) = 2dy \ (x - x_0) - 2dx \ (y - y_0) = 0 \]

\[ \ell(x_0, y_0) = \ell(x_1, y_1) = 0 \quad \ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) = dy - dx \]

\[ \ell(x + 1, y) - \ell(x, y) = 2dy \quad \ell(x, y + 1) - \ell(x, y) = -2dx \]
\[ \ell(x, y) = 2dy (x - x_0) - 2dx (y - y_0) = 0 \]

\[ \ell(x_0, y_0) = \ell(x_1, y_1) = 0 \quad \ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) = dy - dx \]

\[ \ell(x + 1, y) - \ell(x, y) = 2dy \quad \ell(x, y + 1) - \ell(x, y) = -2dx \]
\[ \ell(x, y) = 2dy \ (x - x_0) - 2dx \ (y - y_0) = 0 \]

\[ \ell(x_0, y_0) = \ell(x_1, y_1) = 0 \quad \ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) = dy - dx \]

\[ \ell(x + 1, y) - \ell(x, y) = 2dy \quad \ell(x, y + 1) - \ell(x, y) = -2dx \]
\( \ell(x, y) = 2dy(x - x_0) - 2dx(y - y_0) = 0 \)

\[
\begin{align*}
\ell(x_0, y_0) &= \ell(x_1, y_1) = 0 \\
\ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) &= dy - dx \\
\ell(x + 1, y) - \ell(x, y) &= 2dy \\
\ell(x, y + 1) - \ell(x, y) &= -2dx
\end{align*}
\]
\[ \ell(x, y) = 2dy (x - x_0) - 2dx (y - y_0) = 0 \]

\[
\begin{align*}
\ell(x_0, y_0) &= \ell(x_1, y_1) = 0 \\
\ell(x_0 + \frac{1}{2}, y_0 + \frac{1}{2}) &= dy - dx \\
\ell(x + 1, y) - \ell(x, y) &= 2dy \\
\ell(x, y + 1) - \ell(x, y) &= -2dx
\end{align*}
\]
**Line Drawing**

```
local function line(x 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


Integer endpoints

Incremental

• No divisions
• (almost) No multiplications

Leave no gaps
Polygon filling


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