Alignment Zones

- Use of alignment zones
  - Determined directly from the ADF
  - Define strong vertical and horizontal edges of the glyph and characteristic distances of a typeface
    - Baseline to x-height and baseline to cap-height distances
    - Spacing between repeated vertical stems
    - Left edge of a glyph
  - Used to build the appropriate transformation needed for grid fitting each ADF to the pixel grid or pixel component grid

Alignment Zones

- Determining alignment zones
  - Three step process:
    - Initialize alignment zone detection for a specified typeface
      - Determine x-height and cap-height from particular glyphs of the typeface (i.e., 'z' and 'Z')
    - Detect alignment zones for each required glyph in the specified typeface
      - Use table to specify expected alignment zones for each glyph in Latin fonts
    - Terminate alignment zone detection for the specified typeface
  - Can be done once (offline) and stored with the glyph's ADFPath
    - Requires at most 64 bits per glyph

Rendering

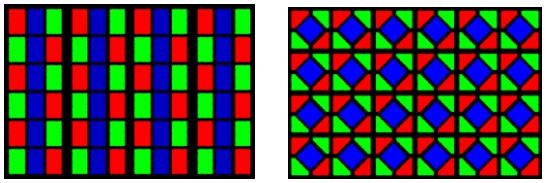
- Density images
  - Glyphs are rendered into density images
    - For each pixel or pixel component in the density image
      - Transform the pixel location to a sample point in the ADF
      - Reconstruct the distance field at the sample point
      - Map the reconstructed distance to a density value
  - Density images can be used by the application to blend a foreground color (e.g., a text color) with a background color or a background image to produce a colored image of the rendered glyph

Rendering

- Rendering Steps
  - Set the render state (e.g., point size, display mode …)
  - Call the render setup function to determine data used for rendering and typesetting
    - Render setup determines a matrix transformation that places the glyph in the display window
    - The matrix transformation represents any affine transformation, i.e., translation, rotation, scale, shear, reflection, etc.
  - Create a density image of the required size
  - Render the glyph into the density image

LCDs and Alternative Pixel Patterns

- Coverage-based anti-aliasing requires multiple samples per pixel component for supersampling
  - TrueType takes advantage of the RGB-striped LCD pixel pattern to reuse samples
  - Each different striping patterns requires different strategy to reuse samples (not hard, just inconvenient)
  - Irregular patterns can't reuse samples as easily



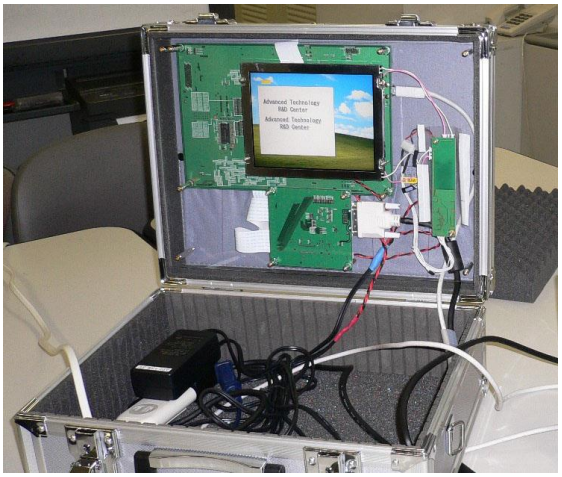
Two alternative pixel patterns used by Clairvoyante

LCDs and Alternative Pixel Patterns

- Distance-based anti-aliasing only requires one sample per pixel component
  - Different patterns do not affect efficiency
    - A simple modification to the transform from image coordinates to ADF coordinates is required
  - Library supports RGBv, BGRv, RGBh, BGRh
  - Effectiveness for alternative pixel patterns already demonstrated
  - CSM makes it easy to adjust stroke weight and contrast for new pixel patterns and to compensate for color fringing

Hardware Implementation

- Saffron – V0 (Biquadratic cells only)



Hardware Implementation

- Saffron – V0 (Biquadratic cells only)
  - 156K gates on an FPGA
    - Includes memory controllers for accessing fonts and frame buffers
    - Includes compositing (triple buffered)
  - Renders 100,000 glyphs per second at 100 MHz for 10 point, 96 dpi, LCD rendering (3 samples per pixel)
  - Current implementation has a single component pipeline
  - Designed for a 3 component pipeline (3x as fast at a cost of 20K-30K additional gates per pipeline)

Patents

- Detail-Directed Hierarchical Distance Fields: **U.S. Patent No. 6,396,492**
- Method for Antialiasing an Object Represented as a Two-Dimensional Distance Field in Image-Order
- Method for Antialiasing an Object Represented as a Two-Dimensional Distance Field in Object-Order: **Allowed**
- Method for Animating Two-Dimensional Objects
- Method for Converting Two-Dimensional Objects to Distance Fields
- Method for Converting a Two-Dimensional Distance Field to a Set of Boundary Descriptors
- Method for Converting Two-Dimensional Pen Strokes to Distance Fields
- Method for Generating a Two-Dimensional Distance Field within a Cell Associated with a Corner of a Two-Dimensional Object

Patents

- Method and Apparatus for Antialiasing a Set of Objects Represented as a Set of Two-Dimensional Distance Fields in Image-Order
- Method and Apparatus for Antialiasing a Set of Objects Represented as a Set of Two-Dimensional Distance Fields in Object-Order: **Allowed**
- Method for Generating a Composite Glyph and Rendering a Region of the Composite Glyph in Image-Order
- Method for Generating a Composite Glyph and Rendering a Region of the Composite Glyph in Object-Order
- Methods for Generating an Adaptively Sampled Distance Field of an Object with Specialized Cells
- Method and Apparatus for Rendering Cell-Based Distance Fields Using Texture Mapping: **Allowed**
- Method for Typesetting a Set of Glyphs Represented as a Set of Two-Dimensional Distance Fields

Patents

- Method, Apparatus, and System for Rendering Using a Progressive Cache
- Pipeline and Cache for Processing Data Progressively
- Modeling and Combining Multiple Graphics Objects
- System and Method for Generating Adaptively Sampled Distance Fields with Bounded Distance Trees
- Method for Traversing Quadtrees, Octrees, and N-Dimensional Bi-trees: **U.S. Patent No. 6,868,420**

Patents

- Distinction from prior art
  - Distance fields vs. images
  - Distance-based anti-aliasing vs. coverage-based anti-aliasing
  - Sampling discrete space vs. sampling continuous space
  - Derive alignment zones for grid fitting from distance field, not from hinted outlines
  - For typesetting, adjustments are made to advance widths using distance fields after iso-contour selection

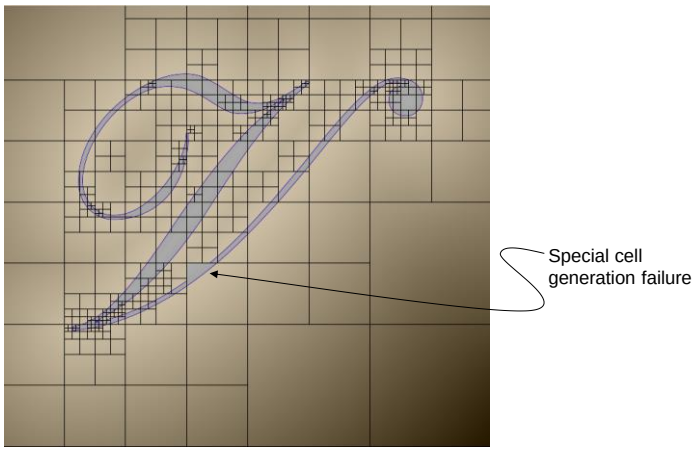
Limitations of Saffron – V1

Alignment Zones

- Current alignment zones include baseline, x-height or cap-height, and two vertical stems
- Some Latin glyphs would benefit from more alignment zones
  - Glyphs with 2 vertical or horizontal zones (e.g., e, F, m)
  - Glyphs with descenders (e.g., p, q, y)
- Need multiple alignment zones or new approach for best results in complex characters
  - e.g., CJK fonts

Issues and Unproven Features

- The predicate test used for special cell generation is not fail proof



Issues and Unproven Features

- Non-uniform scale in x and y distorts the filter region
- Some markets may require additional hint-level fine tuning to provide ultimate control
- Rendering bitmaps from ADFs untested/unproven
  - 2-bit rendering for eInk was successful

Memory and Processing

- Memory constrained devices
  - ADFs may be too big
  - 2D Distance field representation will always be bigger than 1D outline representation
- Processor constrained devices
  - Generation may be too slow

Saffron Roadmap

Saffron Roadmap

- Direct rendering
  - Solution for memory and processor constrained systems
- Hardware implementations
  - Special cell rendering
  - Direct rendering
  - Low power requirements?
  - Market potential? (Mitsubishi internal vs. broader markets)

Saffron Roadmap

- High quality small CJK
  - High quality stroke-based fonts with
    - Variable stroke weight
    - Various endcaps
      - Rounded, square, mitered, beveled, brushed
  - Alignment zones and grid fitting
  - Direct rendering
  - Automatic generation from outlines
  - Design tool

Saffron Roadmap

- Enhancements and extensions
  - CSM profile builder
  - Alignment zones for non-latin fonts
    - Hindu, Arabic
    - More general approach (e.g., multiple alignment zones) for CJK
- GPU implementation of direct rendering
- Extend library to support alternative pixel patterns
- Performance and size enhancements for Saffron – V1

Direct Rendering

Direct Rendering

- See separate presentation

Advantages Over Traditional Approaches

Advantages

- Sub-pixel rendering patent issues
  - Saffron's IP is comprehensive and patently-distinct from Microsoft's ClearType
  - Traditional, coverage-based anti-aliasing approaches encroach on Microsoft's extensive patent portfolio

Advantages

- Hinting
  - Saffron achieves high quality without hinting
  - Some markets are encumbered by hinting (e.g., Flash and SVG)
    - Markets with size constraints (bandwidth, disk space, memory space)
    - Dynamic environments where applications have no control over font quality (e.g., hints may not be available)
  - Hinting new fonts is labor intensive and expensive
- Sub-pixel hinting
  - Hinting fonts for sub-pixel rendering is even worse
  - Legacy bug in hinting of certain glyphs (e.g., Arial 'x')

Advantages

- Hardware market
  - Computationally clean rendering pipeline is straightforward to implement in silicon
  - No special casing of type vs. graphics
  - Hardware prototypes
    - Saffron – V0 hardware prototype exists
    - Hardware implementations of Saffron – V1 and direct rendering approach are under development

Advantages

- GPU implementation is straightforward
  - Potential markets
    - Desktop/laptop market (e.g., GPU-centric Longhorn)
    - Console and handheld game market
    - Embedded systems market
      - GPUs will be increasingly incorporated into next generation devices
      - GPUs on today's devices could support Saffron using Gouraud shaded texture mapped triangles

Advantages

- High quality stroke-based CJK
  - Can provide higher quality stroke-based fonts with a small memory footprint
  - Can provide much higher quality than current market offerings
    - Variable stroke weight
    - Designed endcaps

Advantages

- Alternative pixel patterns and displays
  - Saffron adapts easily to different sampling patterns
    - Saffron requires only one sample per pixel component for anti-aliasing
    - Saffron doesn't depend on the topology of the pattern
      - Kodak OLEDs
      - ClairVoyante Pentile
  - Continuous Stroke Modulation (CSM) provides OEMS with unprecedented control to tune type for their specific displays

Advantages

- Special effects
  - CSM control
  - Inexpensive thin and bold typefaces
  - Temporal anti-aliasing
  - Motion blur
  - Using the distance field for special effects such as soft-body deformation and particle systems
  - 3D type