Ghostscript and MuPDF Status
OpenPrinting Summit May 2018

Michael Vrhel, Ph.D.

Artifex Software Inc.

Novato CA

Outline

GS/MuPDF Overview

Ghostscript or MuPDF?

What is new with Ghostscript

What is new with MuPDF

Fork of LCMS2

Current Work

The Basics of GS

Ghostscript is a document conversion and rendering engine.

Written in C ANSI 1989 standard (ANS X3.159-1989)

Essential component of the Linux printing pipeline.

Dual AGPL/Proprietary licensed. Artifex owns the copyright.

Source and documentation available at www.ghostscript.com

The Basics of MuPDF

- **A Core Set of Libraries Focused on PDF**
  Entirely written in C, very portable, small ROM footprint
  Windows/Linux/MacOS/iOS/Android/BB10/QNX/others

- **Example Tools:**
  Simple viewers for Linux/Android/MacOS/iOS/Windows/WinRT.
  Command line tools:
  - Rendering PDF/XPS/Epub pages
  - Creating PDF pages
  - Merging PDF content
  - Extracting pages
  - Decompressing content streams
  - Extracting resources
  - Repairing files

- **Licensing.** Dual AGPL/Proprietary licensed. Artifex owns the copyright.

Ghostscript or MuPDF?

http://twiki.ghostscript.com/do/view/Ghostscript/GhostscriptOrMuPDF

For most printing applications - use Ghostscript.

PostScript
PCL
Massive range of output devices

Ghostscript or MuPDF?

For screen use or embedded devices - use MuPDF.

Small

- Much smaller ROM footprint.

Simple

- Small set of dependent libraries
- Simpler to port

Interactive features

- More suitable for building viewers
- Searching
- Zooming
- Form filling
- Transitions

New!

- Full managed color support.
- Spot color support

Changes to GS since last meeting

Release 9.22 October 4th 2017

- Ghostscript can now consume and produce (via the pdfwrite device) PDF 2.0 compliant files.
- The main focus of this release was security and code cleanliness. Hence many AddressSanitizer, Valgrind and Coverity issues addressed.
- Usual round of bug fixes, compatibility changes, and incremental improvements.

Changes to GS since last meeting

Release 9.23 October 4th 2017

- Added family of 'pdfimage' devices (pdfimage8, pdfimage24 and pdfimage32) which produce rendered output wrapped up as an image in a PDF.

- Added 'pclm' device which produces PCLm format output.

- Added ColorAccuracy parameter allowing the user to decide between speed or accuracy in ICC color transforms.

Changes to GS since last meeting

Release 9.23 October 4th 2017

- JPEG Passthrough: devices which support it can now receive the 'raw' JPEG stream from the interpreter. (pdfwrite/ps2write)
- PDF transparency performance improvements.
- Fork created of LittleCMS. (more about this later)
- Continued bug fixes, performance improvements.

Changes to MuPDF since last meeting

Release 1.11: (April 4 2017)

- Android and iOS viewers split into separate git repositories:
  mupdf-viewer-ios.git has the iOS viewer.
  mupdf-viewer-android-old.git has the Android viewer.
  mupdf-viewer-android-nui.git has a new advanced Android viewer.
  mupdf-viewer-android-mini.git has a new minimalist Android viewer.

- PDF portfolio support with command line tool "mutool portfolio".

- Add callbacks to load fallback fonts from the system.
  Use system fonts in Android to reduce install size.

- Flag to disable publisher styles in EPUB layout.   Also Improved SVG output.

Changes to MuPDF since last meeting

Release 1.12: (November 23rd 2017)

Git repositories for the SDK projects:

- mupdf-android-fitz.git has the JNI bindings in a library.
- mupdf-android-viewer.git has the viewer as an activity in a library.
- mupdf-android-viewer-mini.git has the minimalist viewer as an activity in a library.

Binary packages for these libraries in our Maven repository:

```
com.artifex.mupdf:fitz:1.12.+
com.artifex.mupdf:viewer:1.12.+
com.artifex.mupdf:mini:1.12.+
```

Changes to MuPDF since last meeting

Release 1.12: (November 23rd 2017)

- Color Management:
  - LCMS2 library for color management.
  - CMYK rendering with overprint simulation.
  - Spot color rendering.
  - Transparency rendering fixes.

- Structured text output improvements:
  - Reworked structured text API.
  - Faster text searching.
  - Highlight and copy text by selecting lines instead of by area.
  - New semantic XHTML output format.
  - New layout preserving HTML output format.

Changes to MuPDF since last meeting

Release 1.12: (November 23rd 2017)

- Improved non-AA rendering with new scan converter.
- Improved LARGEFILE support.
- Improved TIFF support.
- Improved documentation.
- PCLm output.
- PSD output.
- New "mutool trace" tool.

Little CMS

Ghostscript has been using lcms since Release 9.0

LittleCMS just added to MuPDF.

Multi-threaded rendering revealed issues.

Talked with Marti Maria and he suggested that we do a fork at this time.

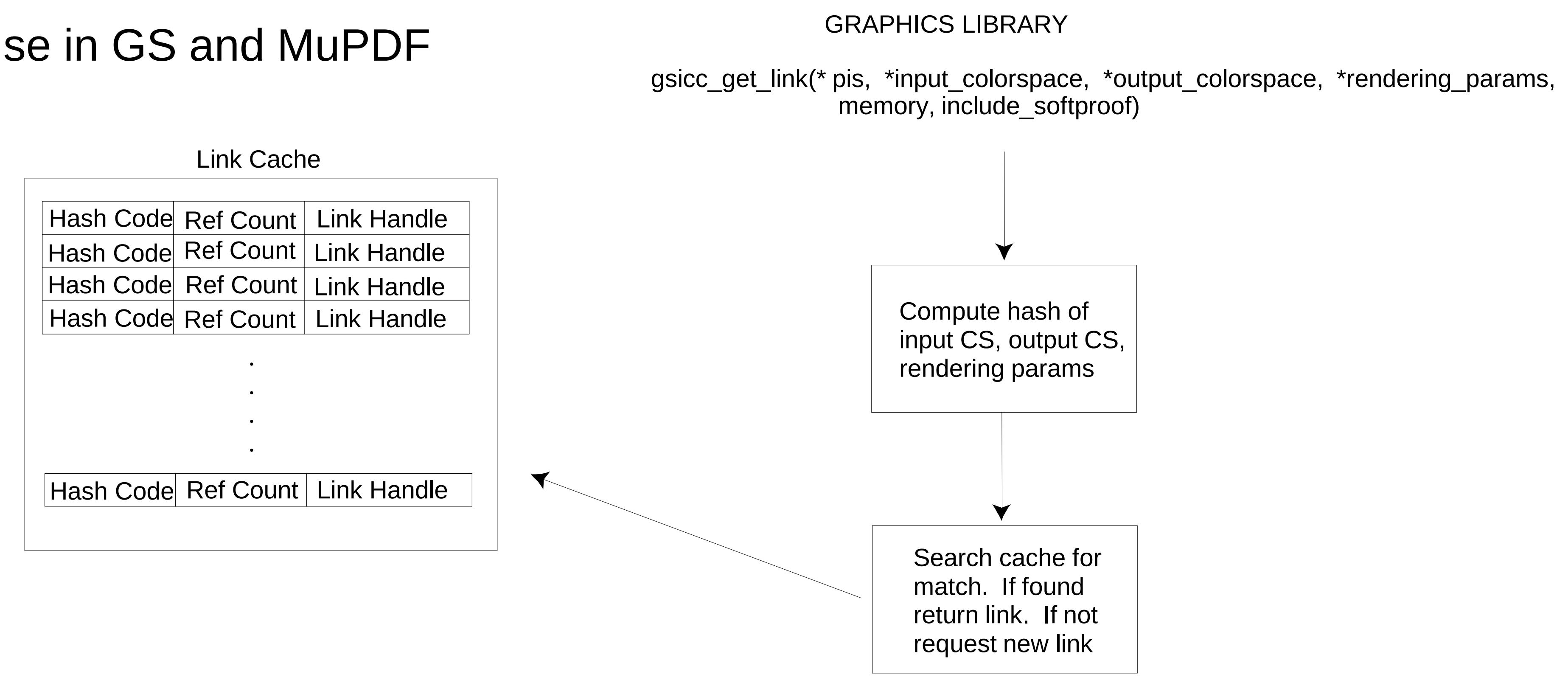
Little CMS

Goals in GS and MuPDF

- Create links between ICC profiles once
- Create hash of the link information
- Store link handle in cache
- Share links amongst rendering threads

Little CMS

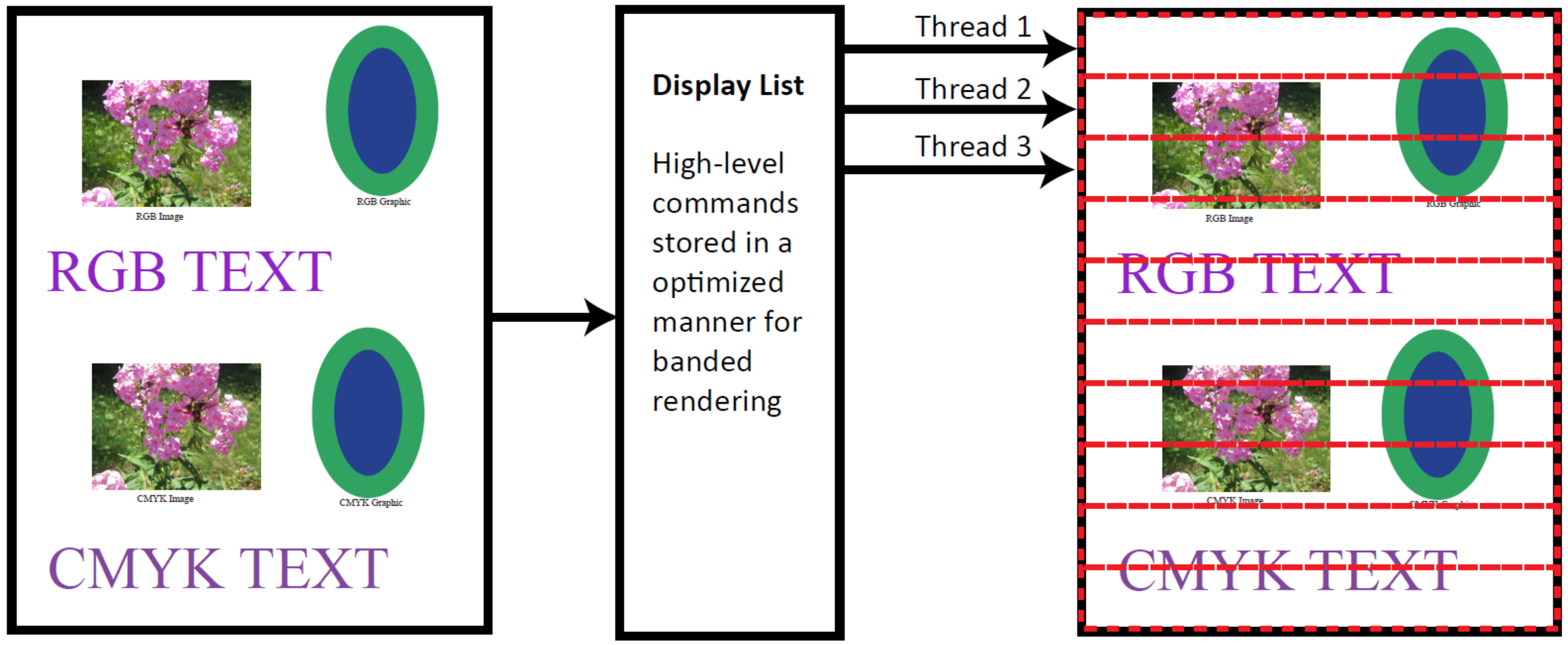
Use in GS and MuPDF



GRAPHICS LIBRARY
gsicc_get_link(* pis, *input_colorspace, *output_colorspace, *rendering_params, memory, include_softproof)
Link Cache
Hash Code
Ref Count
Link Handle
Compute hash of input CS, output CS, rendering params
Search cache for match. If found return link. If not request new link

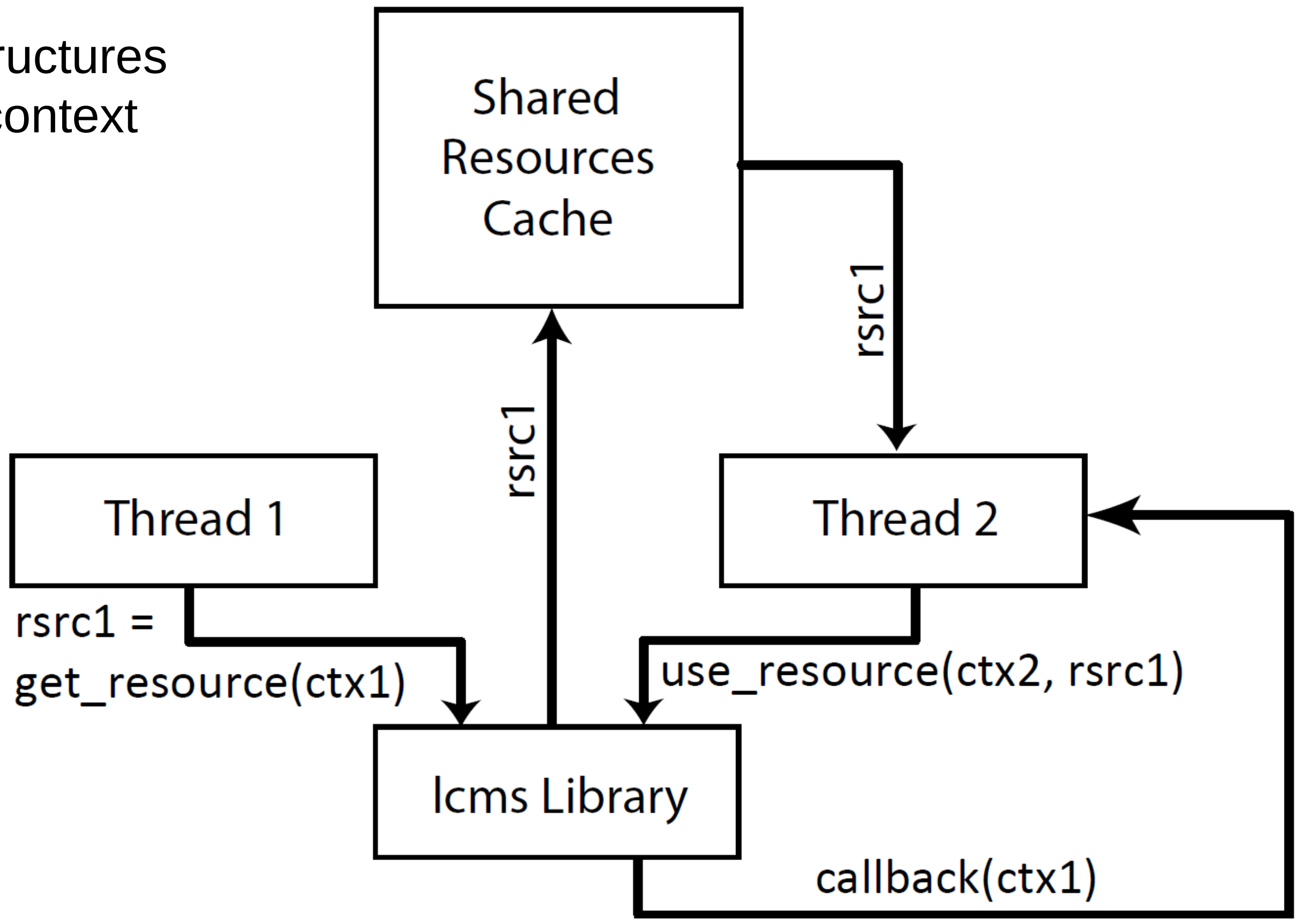
Little CMS

Multi-threaded rendering



Little CMS

Issues
1) Lcms stores context in some structures
2) Not all functions in Lcms pass context



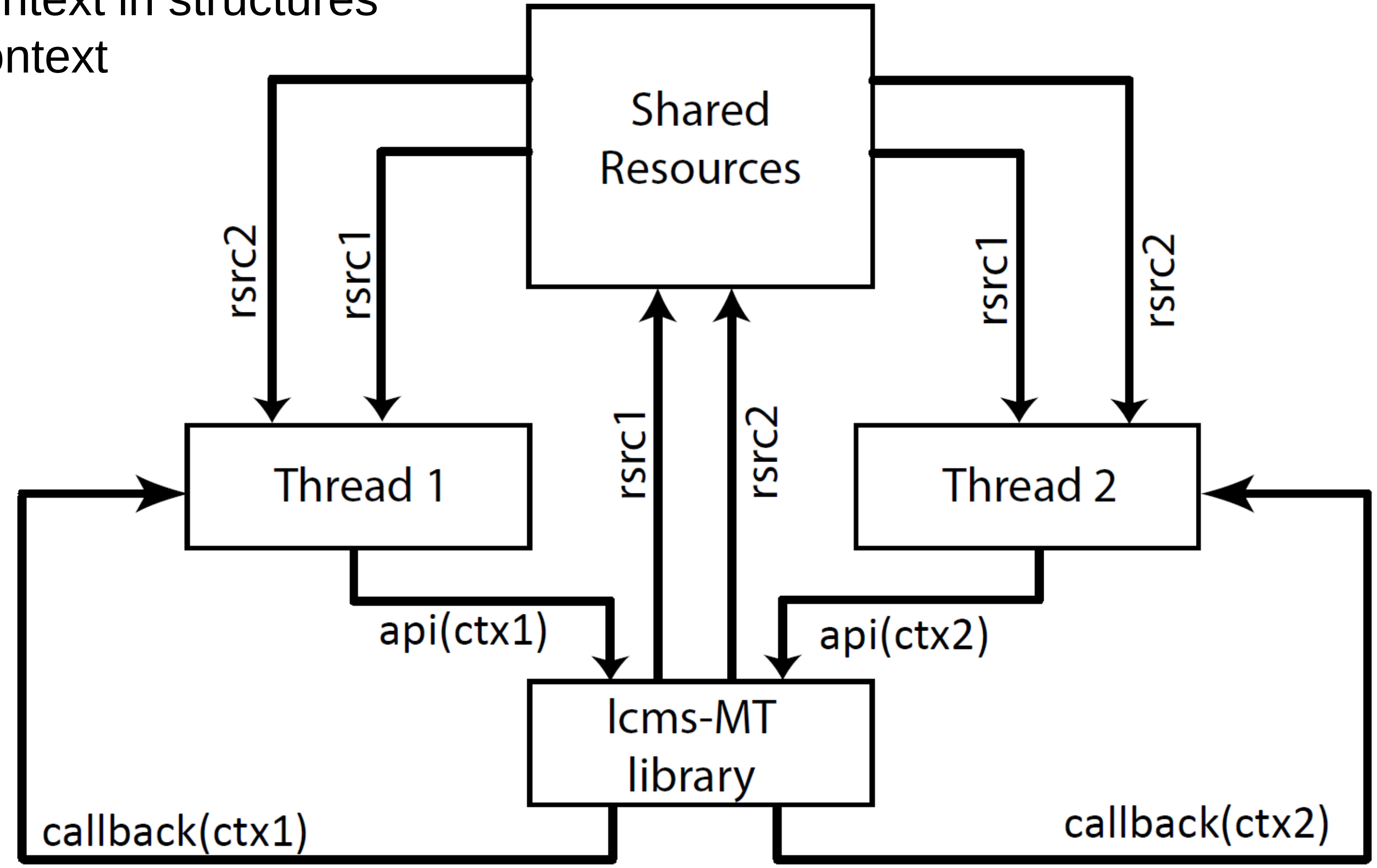
Little CMS

If each bar is in a unique ICC color space and you are using N bands that is either 8 links created or 8 * Number of bands (if the links can't be shared), or place a lock during usage

Little CMS

Alter Lcms

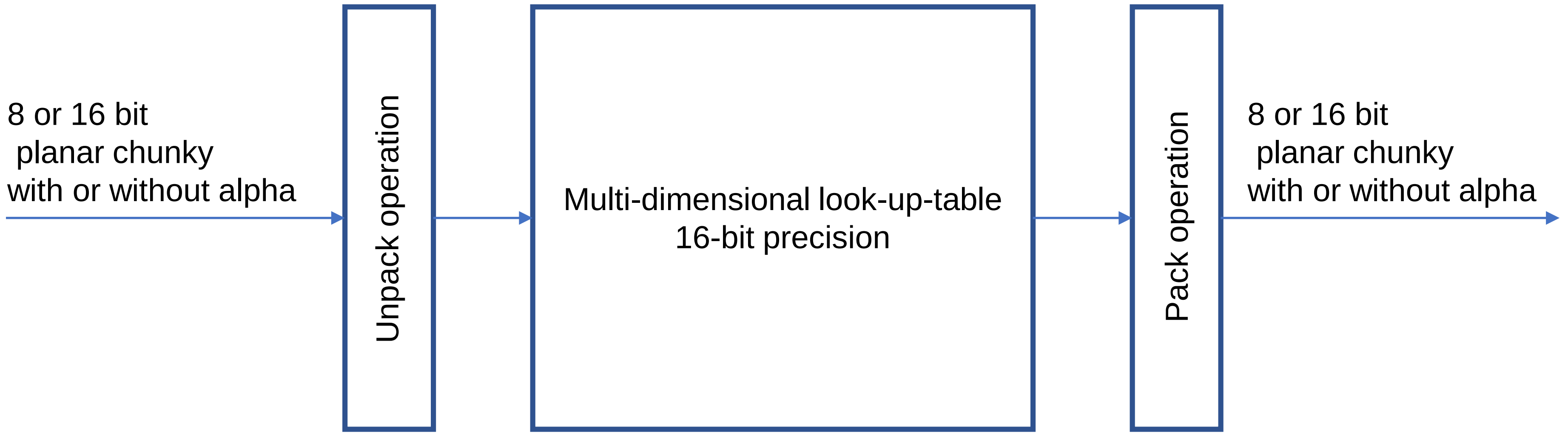
1) Do not allow storage of context in structures
2) Have all functions pass context



Little CMS

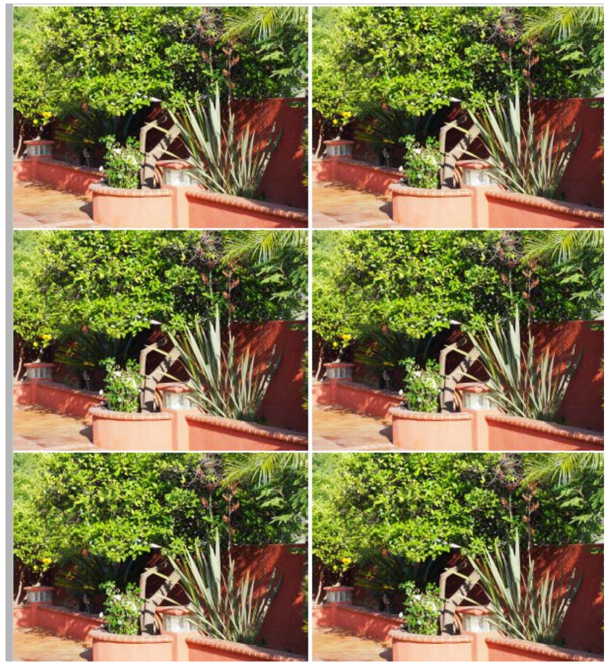
Other Performance Related Changes

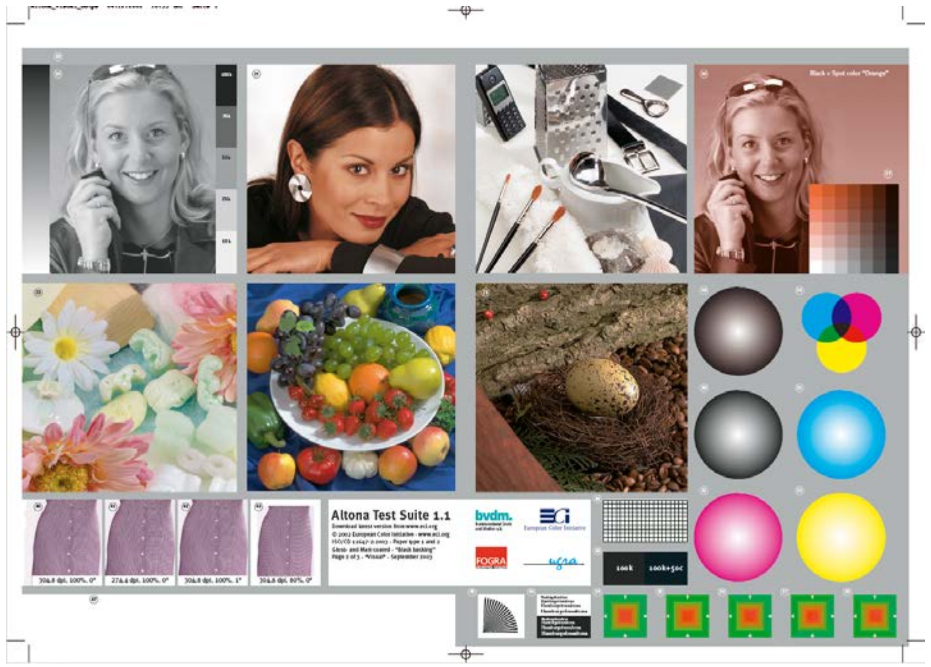
1) Add link cloning operation (for shared usage of table for both 16 and 8 bit data)
2) Avoid function calls on pack/unpack operations

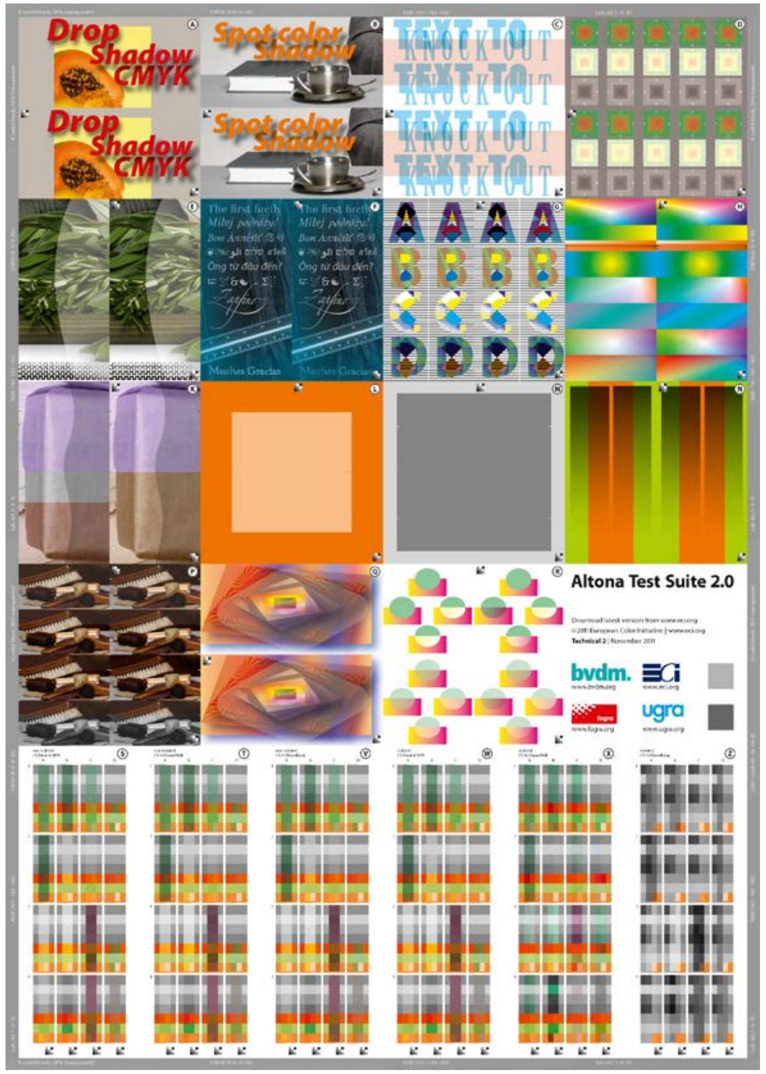


Little CMS

Test Images


95 Mega pixel RGB (Images)


Altona 1.2 Visual


Altona 2.0 Visual

Little CMS

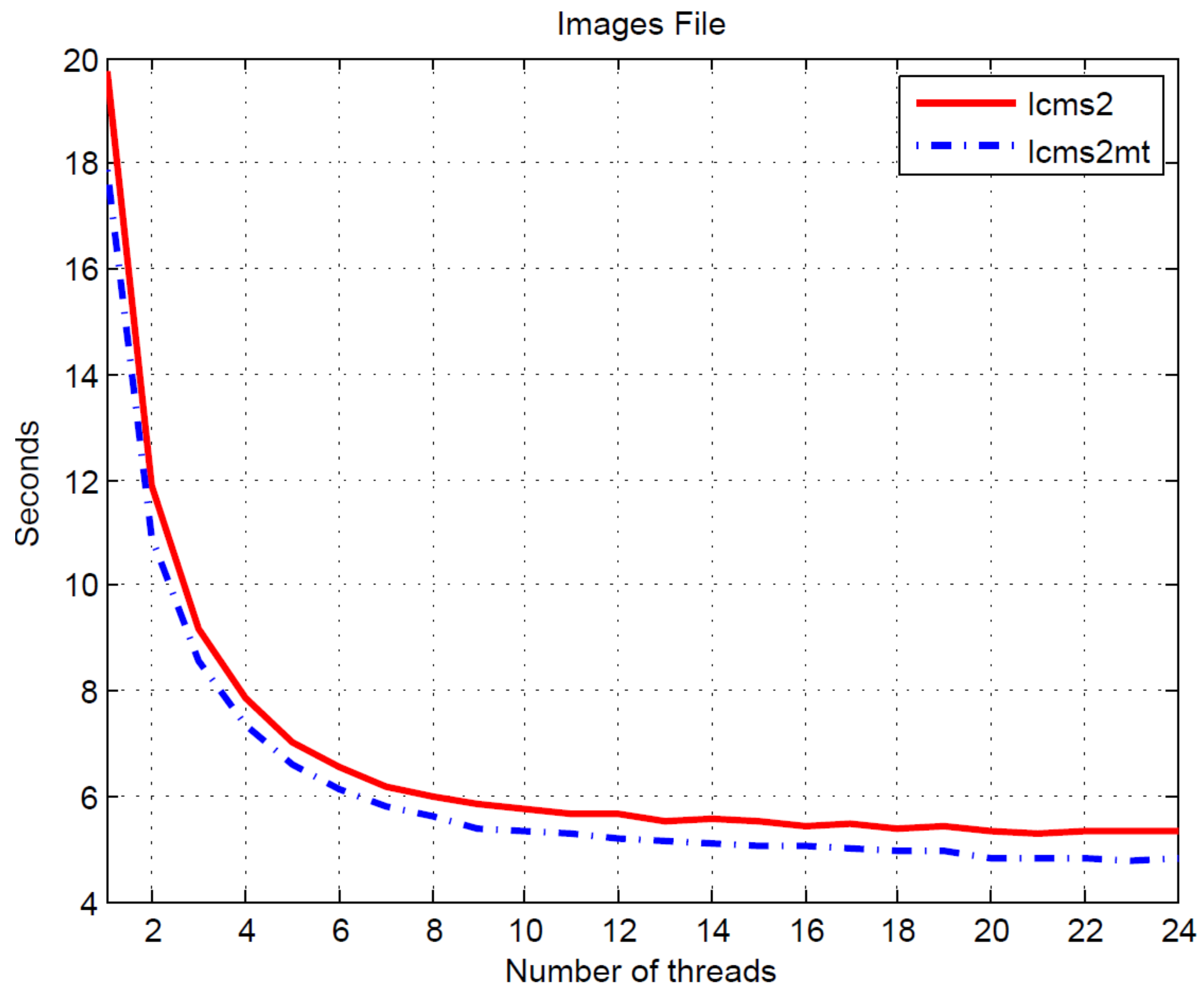
Results

```
./gs -sDEVICE=bitcmyk -Z: -dMaxBitmap=1 -dNumRenderingThreads=#
-r1200 -o /dev/null  -sOutputICCProfile=eciCMYK.icc -dGrayValues=256
-f input file.pdf
```

Run on an Intel 24-core Xeon X5650 CPU 1.6-2.67 GHz
CPU was set into a slower constant speed of 1.6GHz.
CMYK 32bits/pixel at 1200dpi resolution.

Images	0.57GB
Altona Visual 1.2	1.25GB
Altona 2.0	1.11GB

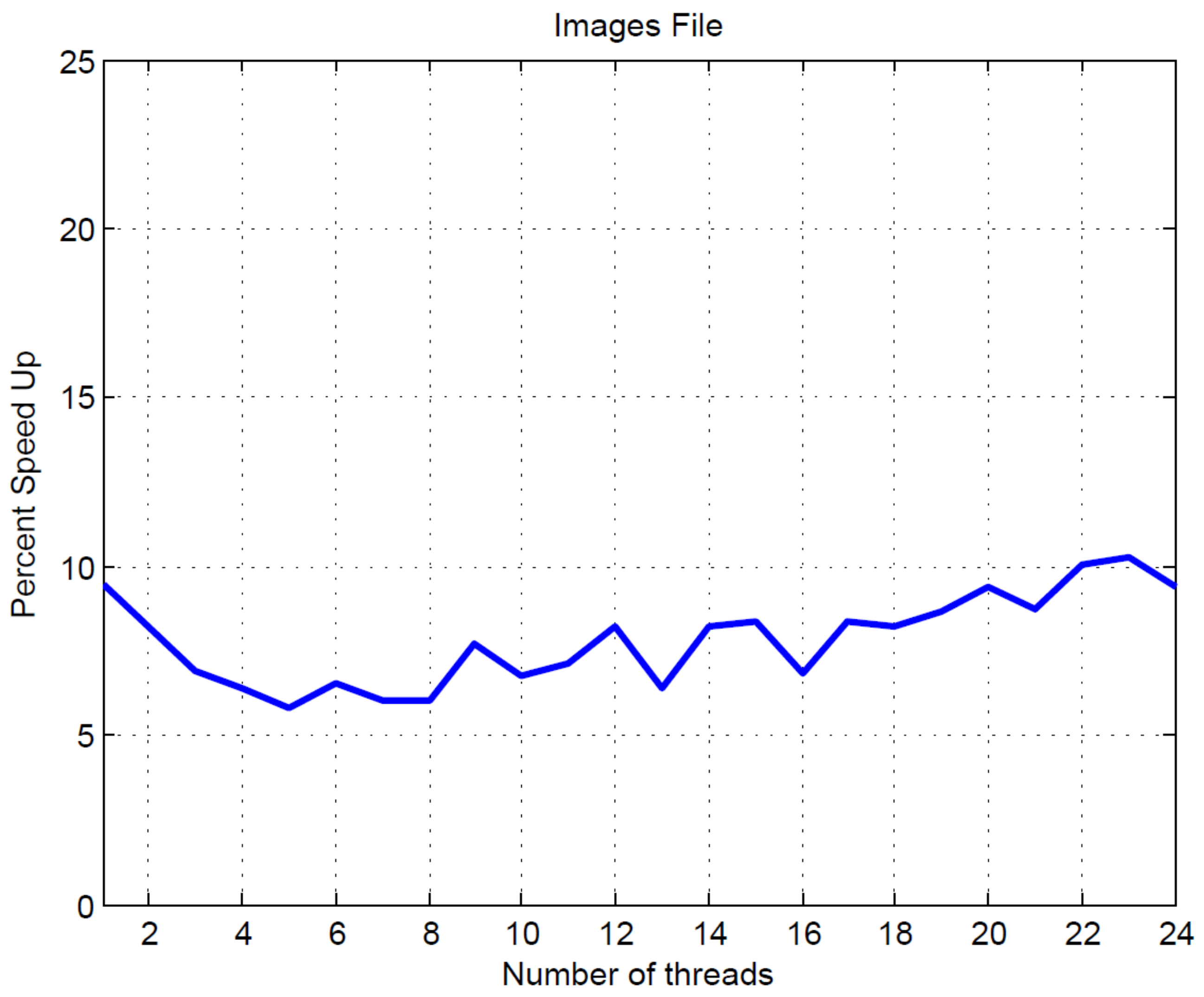
Little CMS



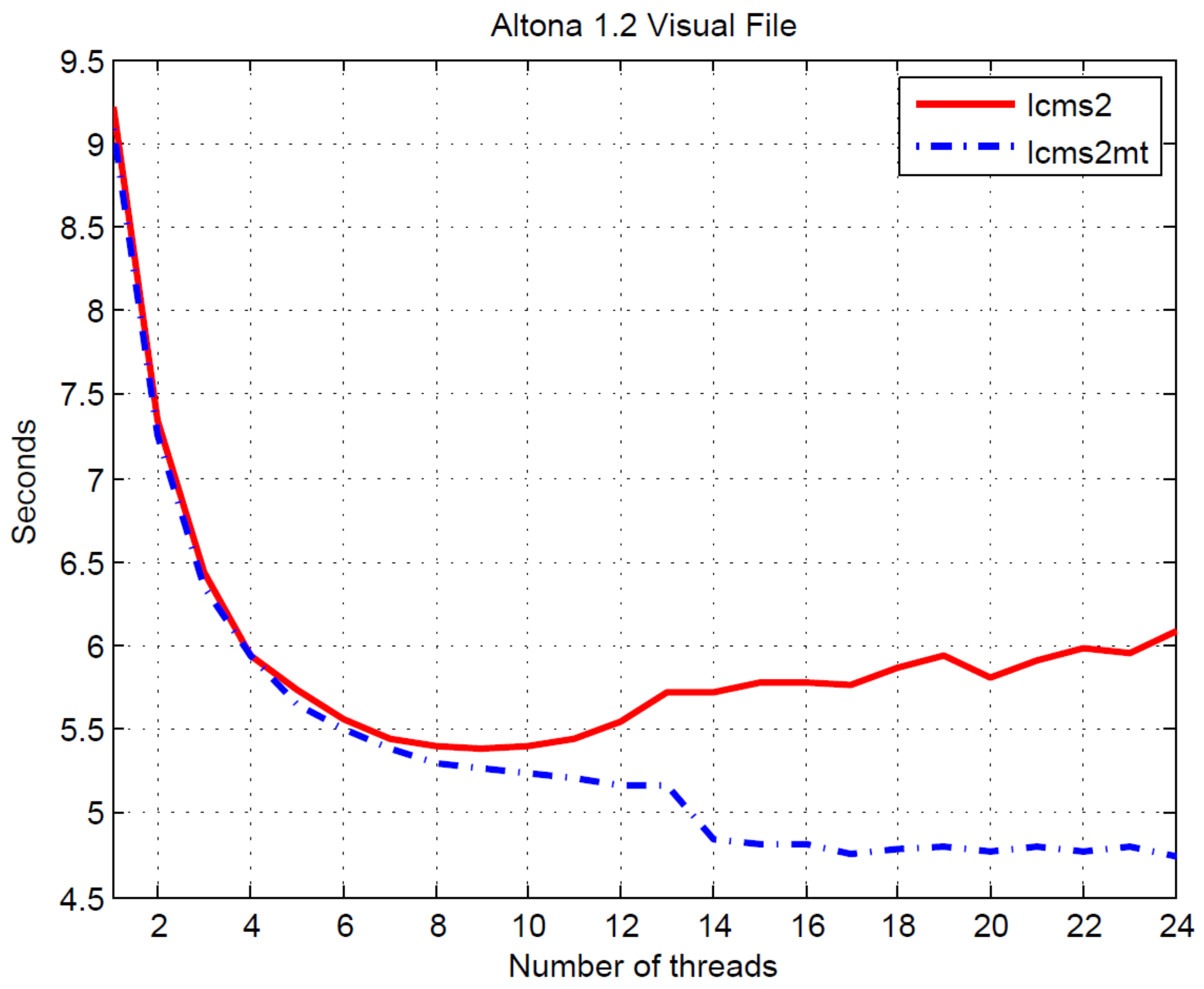
Images File
Seconds
Number of threads
20
18
16
14
12
10
8
6
4
2
4
6
8
10
12
14
16
18
20
22
24
lcms2
lcms2mt

Little CMS

43 percent of rendering time spent transforming colors

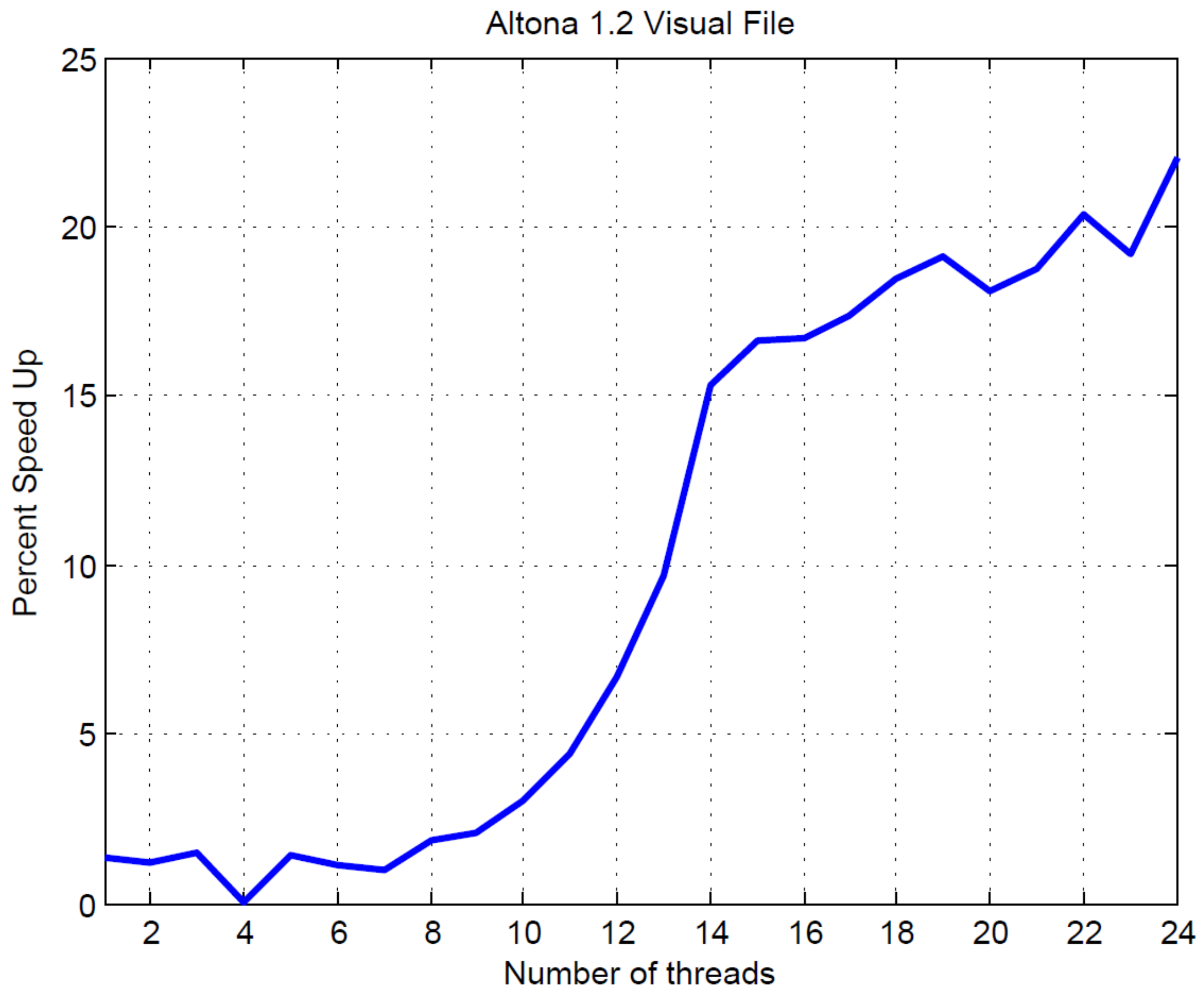


Little CMS

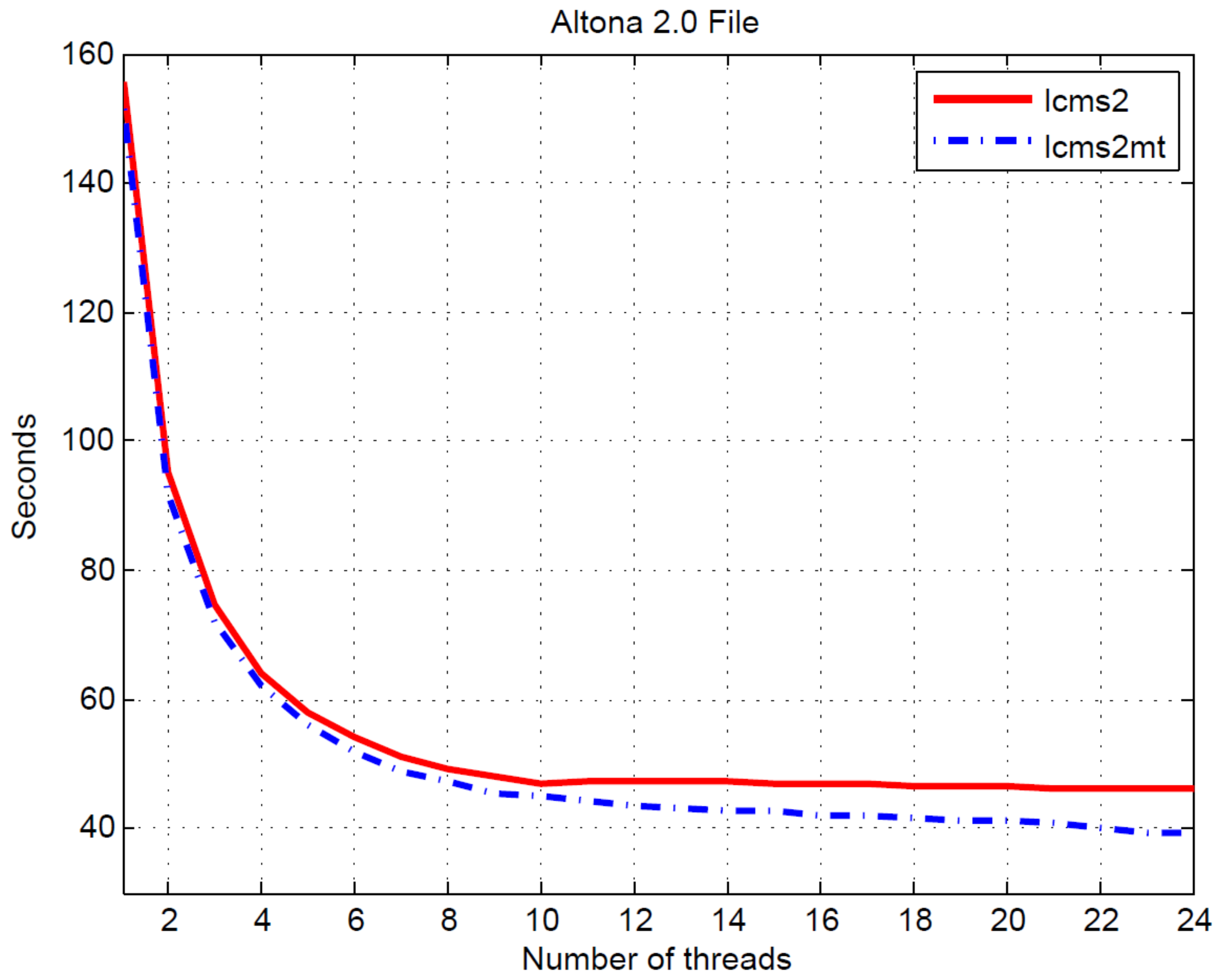

Altona 1.2 Visual File
Seconds
Number of threads
lcms2
lcms2mt
9.5
9
8.5
8
7.5
7
6.5
6
5.5
5
4.5
2
4
6
8
10
12
14
16
18
20
22
24

Little CMS

29 percent of the rendering time spent creating links
12.5 percent spent transforming colors

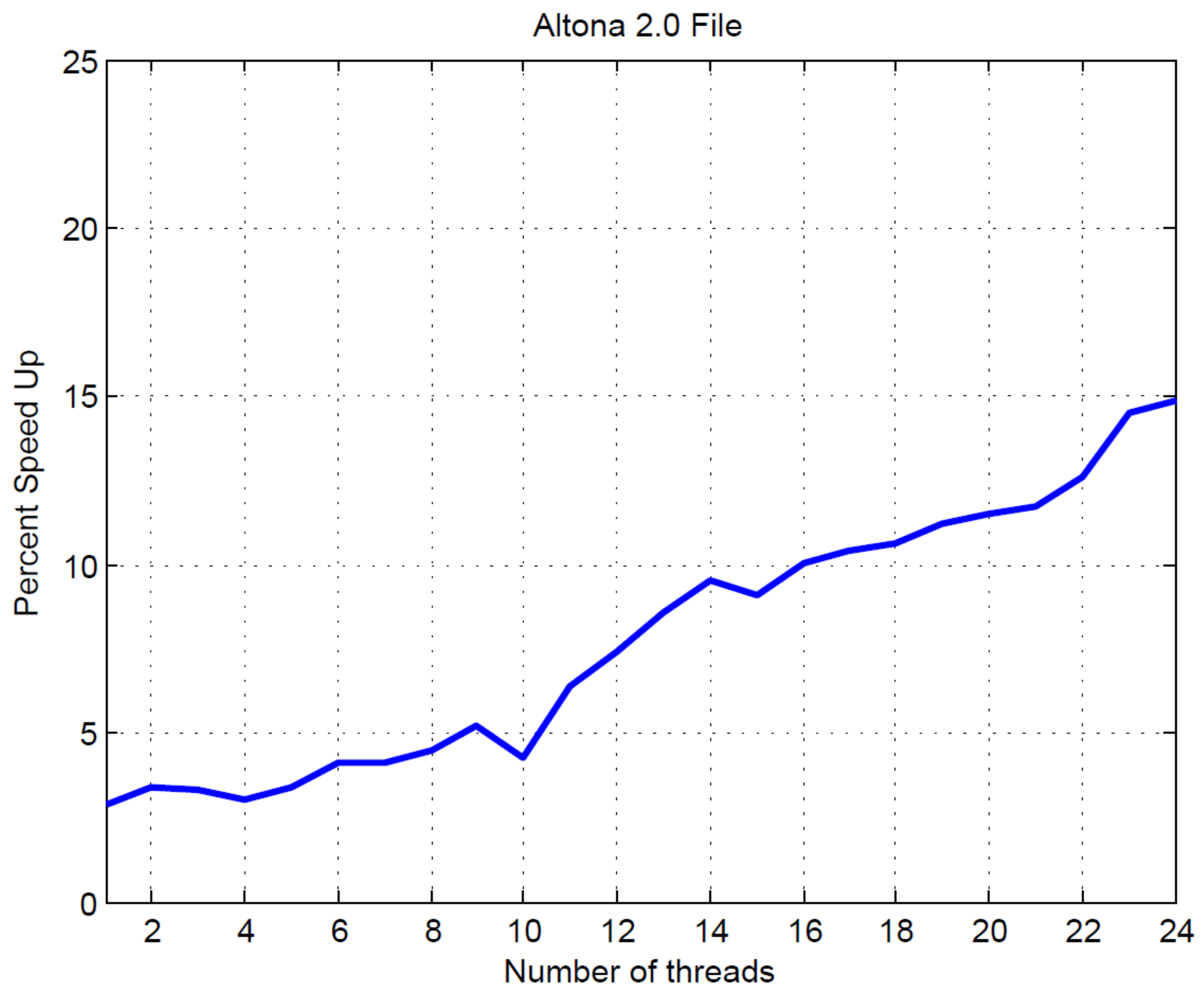


Little CMS


Altona 2.0 File
160
140
120
100
80
60
40
Seconds
2
4
6
8
10
12
14
16
18
20
22
24
Number of threads
lcms2
lcms2mt

Little CMS

7 percent of the rendering time was spent creating ICC links
16 percent time was spent transforming colors.



Little CMS

Fork is currently available with git checkout of ghostscript.

Will likely be added to github.

We will bring in any bug fixes applied to lcms2.

Work underway to provide additional acceleration.

Current Work

Language Switching/Detection (gs)

Device API improvements (passing of graphic state, gs)

Improved Java bindings for MuPDF

Signature and Forms improvements in MuPDF

Javascript manipulations for MuPDF (using mujs)

Significant performance improvements

More Information

Repositories located at
git://git.ghostscript.com

Ghostscript discussions on IRC freenode #ghostscript channel
MuPDF discussions on IRC freenode #mupdf channel

Bug reports
bugs.ghostscript.com

Additional information at www.mupdf.com www.ghostscript.com