Photographic Technology

wiki: PhotoTechEDU

Lecture 21: June 13, 2007

Visualizing via Matlab:

Color Profiles, Ray Tracing, Diffraction

Richard F. Lyon

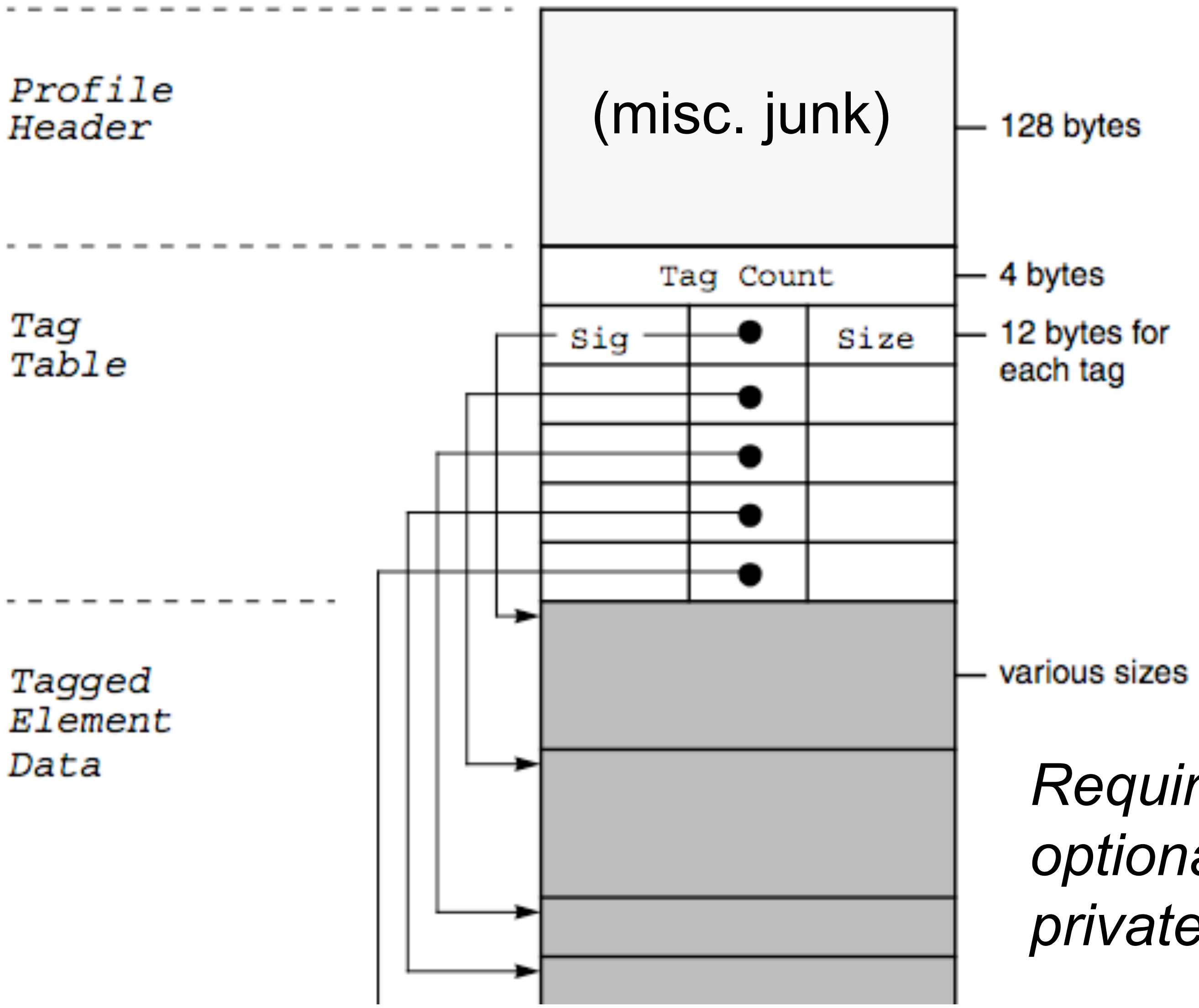
Google Research

dicklyon@google.com

Empirical and Visualization Approaches to Topics in Photography, Using Matlab

- ICC Color Profiles: parsing profiles to see what’s in them; different types
- Ray Tracing: brute-force numerical ray tracing for the non-Gaussian situation; example: aberration caused by a glass plate (filter or prism, a plano optic)
- Diffraction: brute-force vector summing to find diffraction-limited optical spot, including mis-focus effect

ICC profile format



Required, *optional*, and *private* tag data

ICC profile tags: signature + data

```
switch sig
  case 'cprt'
    copyright = char(data)
  case 'desc'
    description = char(data)
  case 'dmdd'
    devicemodel = char(data)
  case 'vued'
    viewingdesc = char(data)
  case 'meas'
    illuminantnumber = data(datasize(i))
    illuminants = ['unk'; 'D50'; 'D65'; 'F2 '; 'D55'; 'A  '; 'E  '; 'F8 '];
    illuminant = illuminants(illuminantnumber+1,:)
  case 'wtpt'
    [whitepointXYZ, wxyz] = extractXYZ(data)
  case ’A2B0'
    table = extractTable(data)
  case ’rTRC'
    redCurve = extractCurve(data)
```

Easy cases: data are characters, byte index, or 32b XYZ values

Also various tables, usually as unsigned 16b

Official sRGB ‘mntr’ profile from HP

```
>> readiccprofile('sRGB.icm')
fn =
sRGB.icm
header =
  HLino  mntrRGB XYZ              1  acspMSFT    IEC sRGB
        -HP
tagcount =
   17

cprt
copyright =
text    Copyright (c) 1998 Hewlett-Packard Company

desc
description =
desc      sRGB IEC61966-2.1          sRGB IEC61966-2.1
```

(copyright is a *REQUIRED* tag)

sRGB white, black, and primaries

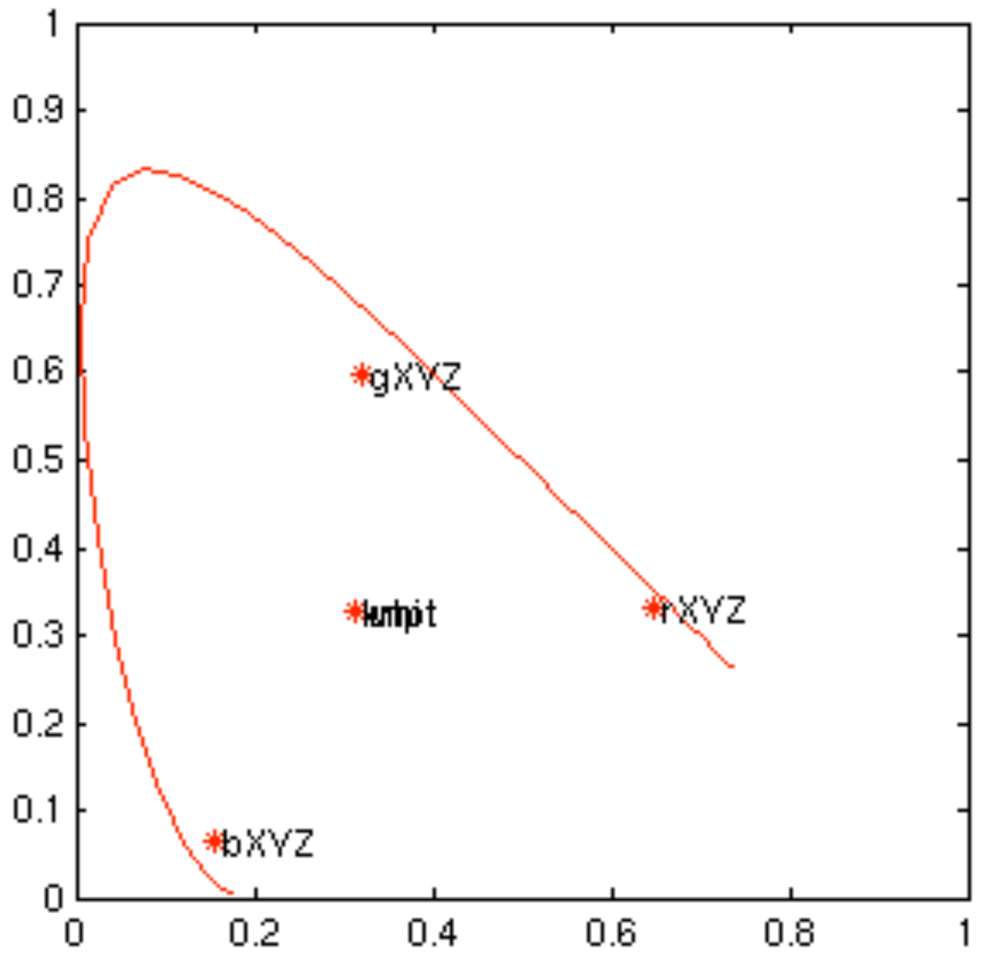
```
meas
illuminant =
    D65

wtpt
whitepointXYZ =
    0.9505    1.0000    1.0891

bkpt
blackpointXYZ =
     0      0      0

rXYZ
rXYZ =
    0.4361    0.2225    0.0139

gXYZ, bXYZ, etc.; plot them . . .
```



sRGB “device” and “viewing” tags

sig =
dmnd
devicemanufacturer =
desc IEC http://www.iec.ch IEC http://www.iec.ch

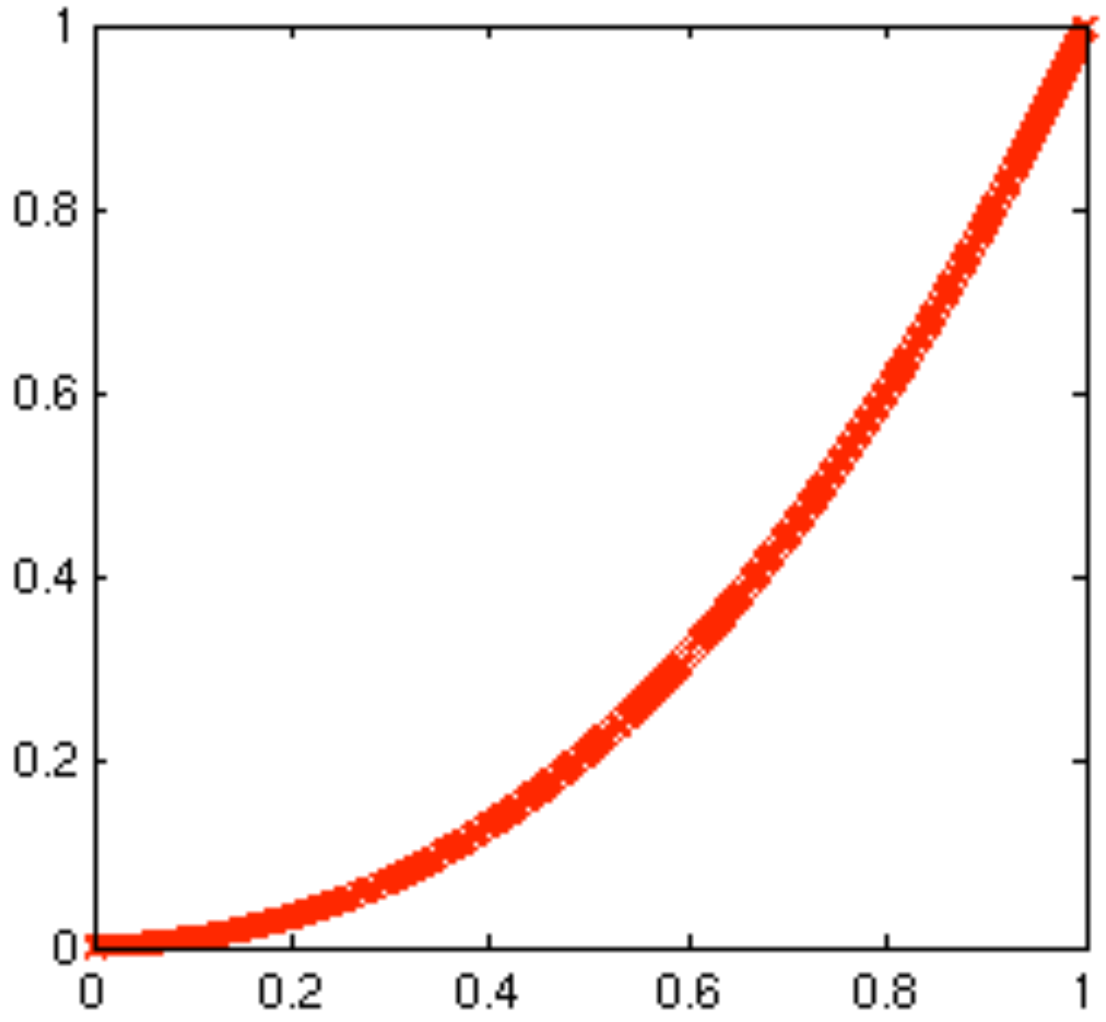
sig =
dmdd
devicemodel =
desc .IEC 61966-2.1 Default RGB colour space - sRGB .IEC 61966-2.1 Default RGB colour space - sRGB

sig =
vued
viewingdesc =
desc ,Reference Viewing Condition in IEC61966-2.1 ,Reference Viewing Condition in IEC61966-2.1

sRGB Tone Reproduction Curves (1D LUTs of 16b values)

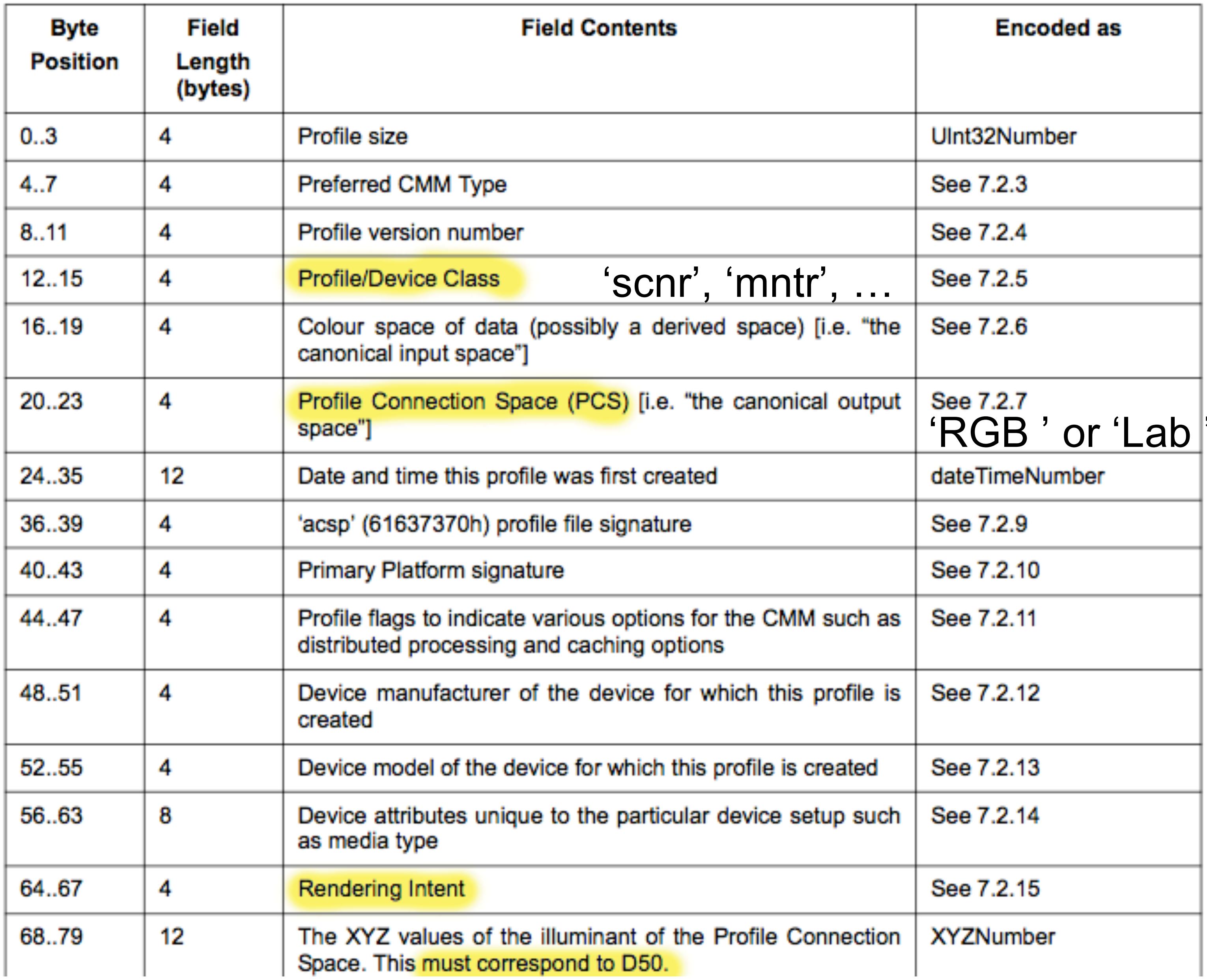
sig =
rTRC
curvecount =
1024
rtoeslope =
12.8123
rtopslope =
0.4388
rtopgamma =
2.2791

(same for green and blue)



Profile header

Table 13 — Profile header fields



Byte Position	Field Length (bytes)	Field Contents	Encoded as
0..3	4	Profile size	UInt32Number
4..7	4	Preferred CMM Type	See 7.2.3
8..11	4	Profile version number	See 7.2.4
12..15	4	Profile/Device Class ‘scnr’, ‘mntr’, …	See 7.2.5
16..19	4	Colour space of data (possibly a derived space) [i.e. “the canonical input space”]	See 7.2.6
20..23	4	Profile Connection Space (PCS) [i.e. “the canonical output space”]	See 7.2.7 ‘RGB ’ or ‘Lab ’
24..35	12	Date and time this profile was first created	dateTimeNumber
36..39	4	‘acsp’ (61637370h) profile file signature	See 7.2.9
40..43	4	Primary Platform signature	See 7.2.10
44..47	4	Profile flags to indicate various options for the CMM such as distributed processing and caching options	See 7.2.11
48..51	4	Device manufacturer of the device for which this profile is created	See 7.2.12
52..55	4	Device model of the device for which this profile is created	See 7.2.13
56..63	8	Device attributes unique to the particular device setup such as media type	See 7.2.14
64..67	4	Rendering Intent	See 7.2.15
68..79	12	The XYZ values of the illuminant of the Profile Connection Space. This must correspond to D50.	XYZNumber

A ‘scnr’ profile – there is no ‘camera’ class

```
>> readiccprofile('epsn1p04.icm')
header =
  *KCMS   scnrRGB Lab          # acspMSFT    EPSOnone
tagcount =
    23


cprt
text    COPYRIGHT (c) 1992=1994 Eastman Kodak Company. All rights reserved


dmdd
desc      ES-800C                    E S - 8 0 0 C   ES-800C 0<Q     -0T -0


K019
UNKNOWN tag signature
datastring =
desc       Ektacolor Plus         E k t a c o l o r   P l u s    Ektacolor Plus 256 levels/Color
      Scanning: 1 pass/Halftone: none/Drp
```

“Private data tags allow CMM developers to add proprietary value to their profiles.”

Lookup table (LUT) tags

Quoting `ICC1v42_2006-05.pdf`:

9.2.1 AToB0Tag

Tag signature A2B0 (41324230h)

Allowed tag types: lut8Type or lut16Type or lutAtoBType

This tag defines a colour transform from Device to PCS using lookup table tag element structures. For most profile classes it defines the transform to achieve perceptual rendering (see table 21). The processing mechanisms are described in lut8Type or lut16Type or lutAtoBType (see 10.8, 10.9 and 10.10).

So what’s a “Profile Connection Space” (PCS)?

- A profile tells you how to convert colors of a device or an abstract colorspace, to or from a standard “profile connection space”.
- PCS is usually 16b (u1Fixed15Number) linear CIEXYZ with D50 white point (unlike sRGB and most monitor spaces which use D656 white point)
- PCS can also be 8b or 16b CIELAB (L*a*b*) with D50 white point, which is almost 1:1 with CIEXYZ but with different nonlinear encoding.

PCS includes standard colorimetry and measurement methods

“So, in summary, the PCS is based on XYZ (or CIELAB) determined for a specific observer (CIE Standard 1931 Colorimetric Observer – often known colloquially as the 2 degree observer), relative to a specific illuminant (D50 – a chromatic adaptation transform is used if necessary), and measured with a specified measurement geometry (0º/45º or 45º/0º), for reflecting media. Measurement procedures are also defined for transmitting media.”

Profile classes and color types

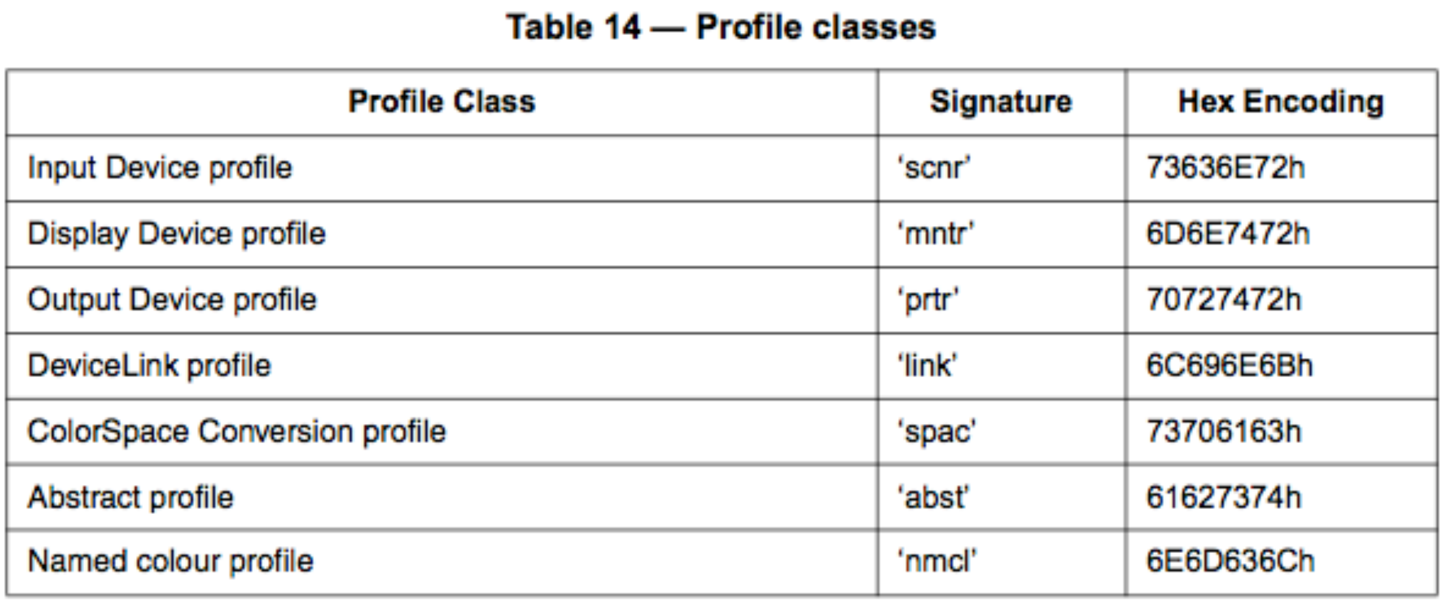


Table 14 — Profile classes

Profile Class	Signature	Hex Encoding
Input Device profile	'scnr'	73636E72h
Display Device profile	'mntr'	6D6E7472h
Output Device profile	'prtr'	70727472h
DeviceLink profile	'link'	6C696E6Bh
ColorSpace Conversion profile	'spac'	73706163h
Abstract profile	'abst'	61627374h
Named colour profile	'nmcl'	6E6D636Ch

No camera class!

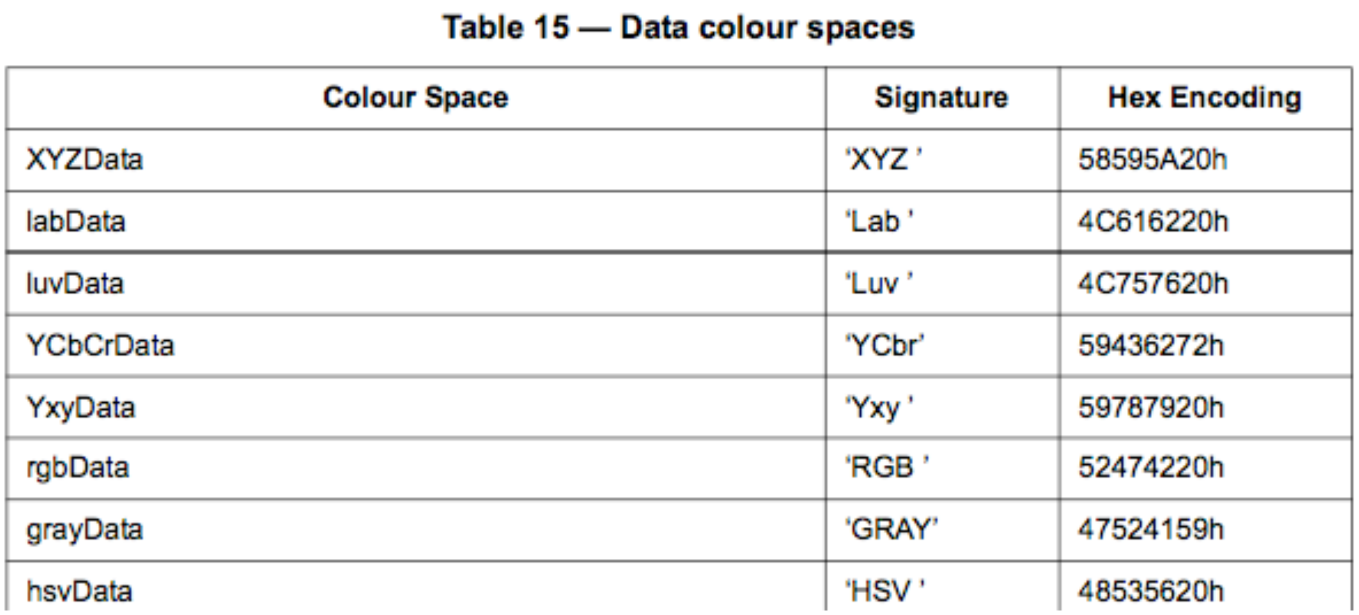
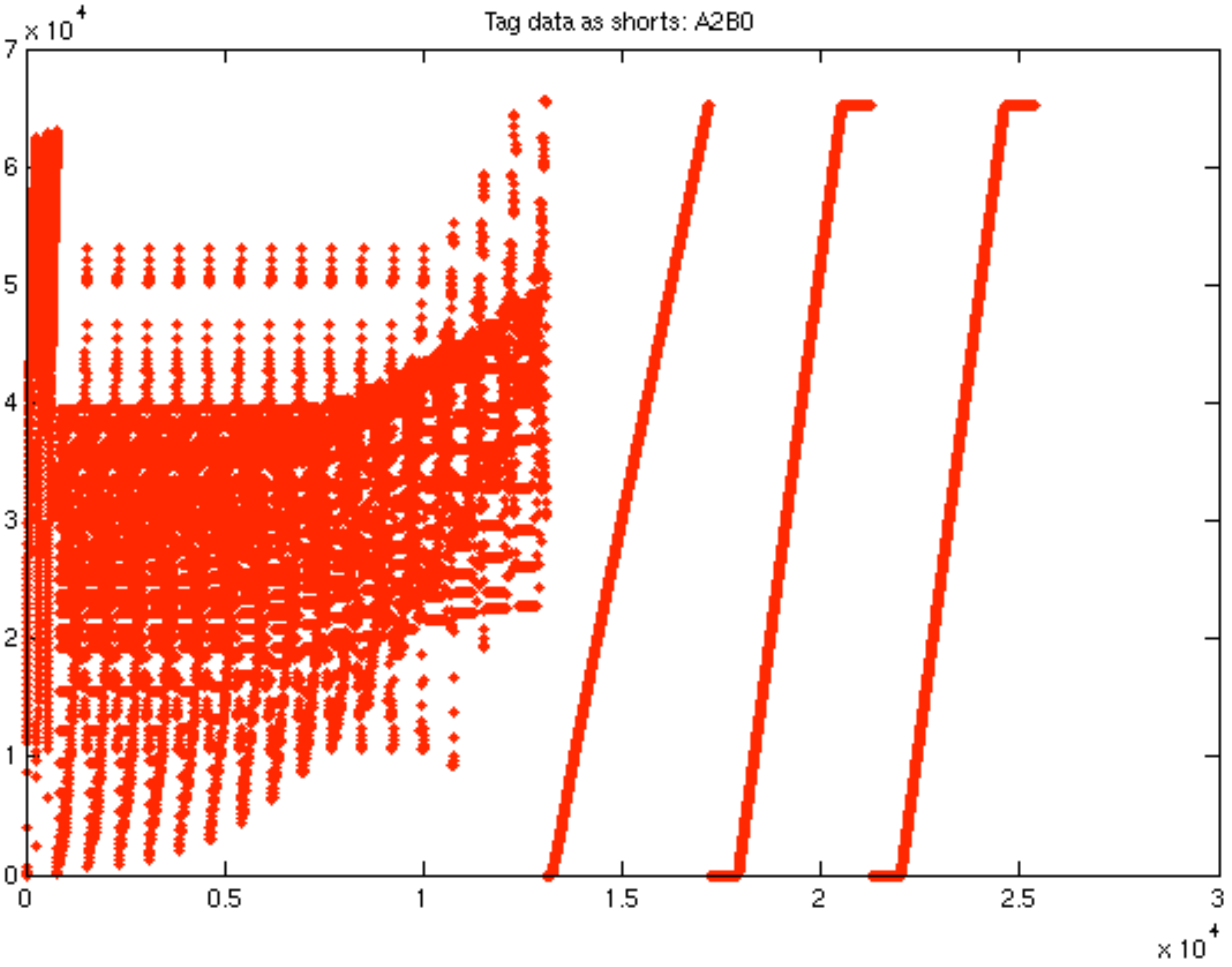


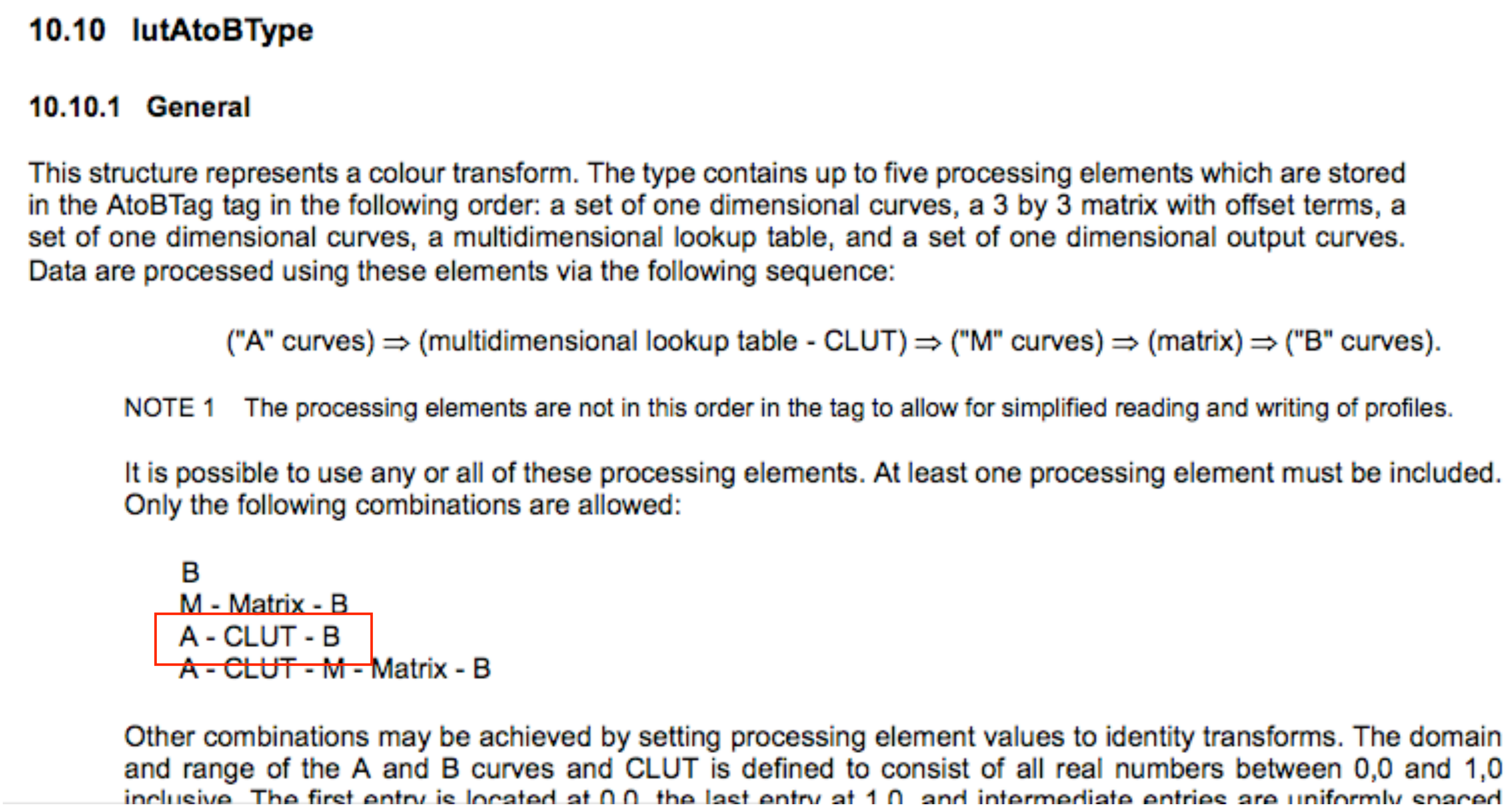
Table 15 — Data colour spaces

Colour Space	Signature	Hex Encoding
XYZData	'XYZ '	58595A20h
labData	'Lab '	4C616220h
luvData	'Luv '	4C757620h
YCbCrData	'YCbr'	59436272h
YxyData	'Yxy '	59787920h
rgbData	'RGB '	52474220h
grayData	'GRAY'	47524159h
hsvData	'HSV '	48535620h

Looking at data block tells you a lot



What does it all mean?



10.10 lutAtoBType

10.10.1 General

This structure represents a colour transform. The type contains up to five processing elements which are stored in the AtoBTag tag in the following order: a set of one dimensional curves, a 3 by 3 matrix with offset terms, a set of one dimensional curves, a multidimensional lookup table, and a set of one dimensional output curves. Data are processed using these elements via the following sequence:

("A" curves) ⇒ (multidimensional lookup table - CLUT) ⇒ ("M" curves) ⇒ (matrix) ⇒ ("B" curves).

NOTE 1 The processing elements are not in this order in the tag to allow for simplified reading and writing of profiles.

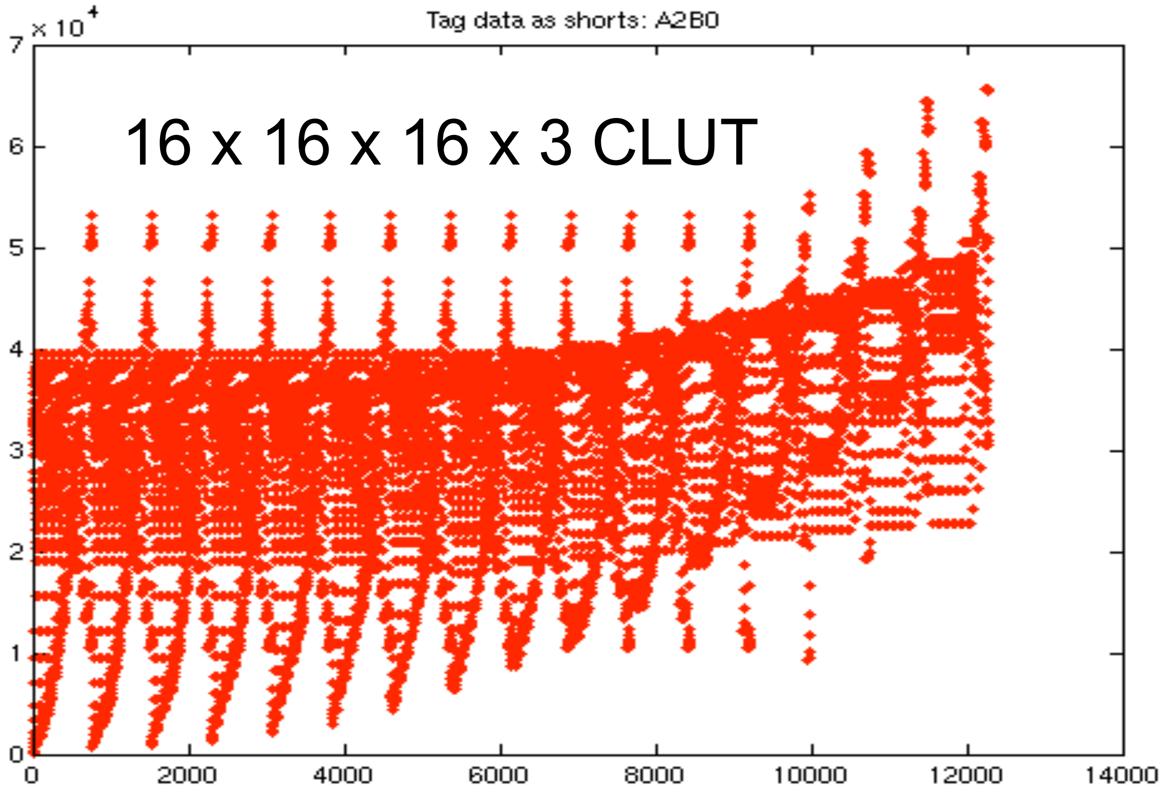
It is possible to use any or all of these processing elements. At least one processing element must be included. Only the following combinations are allowed:

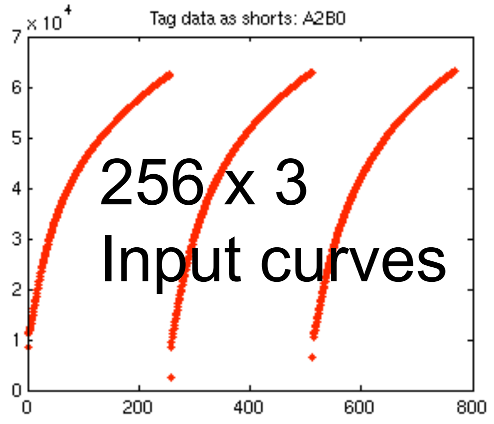
B
M - Matrix - B
A - CLUT - B
A - CLUT - M - Matrix - B

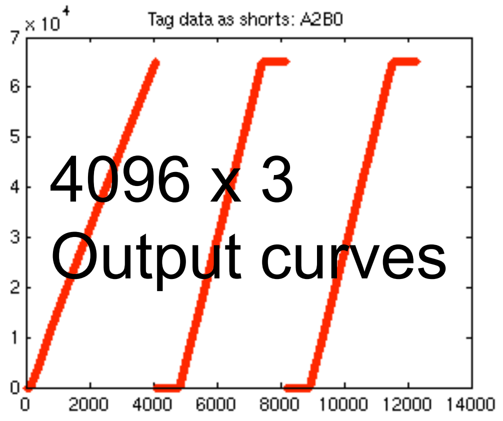
Other combinations may be achieved by setting processing element values to identity transforms. The domain and range of the A and B curves and CLUT is defined to consist of all real numbers between 0,0 and 1,0 inclusive. The first entry is located at 0,0, the last entry at 1,0, and intermediate entries are uniformly spaced

A2B0 multi-function tables: curves and CLUT

sig =
A2B0
ninputs =
3
noutputs =
3
nCLUTgridpoints =
16
inputentries =
256
outputentries =
4096

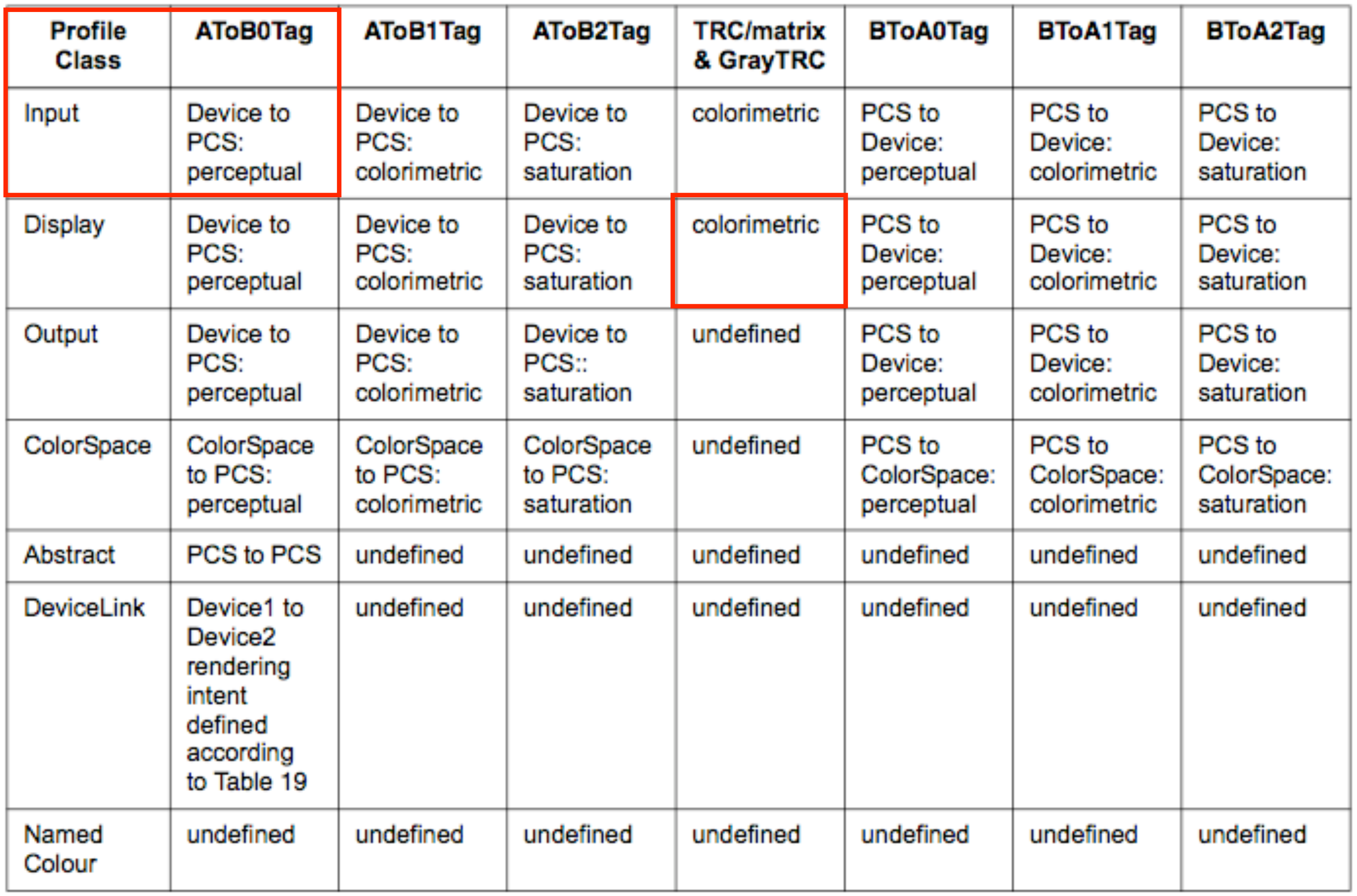






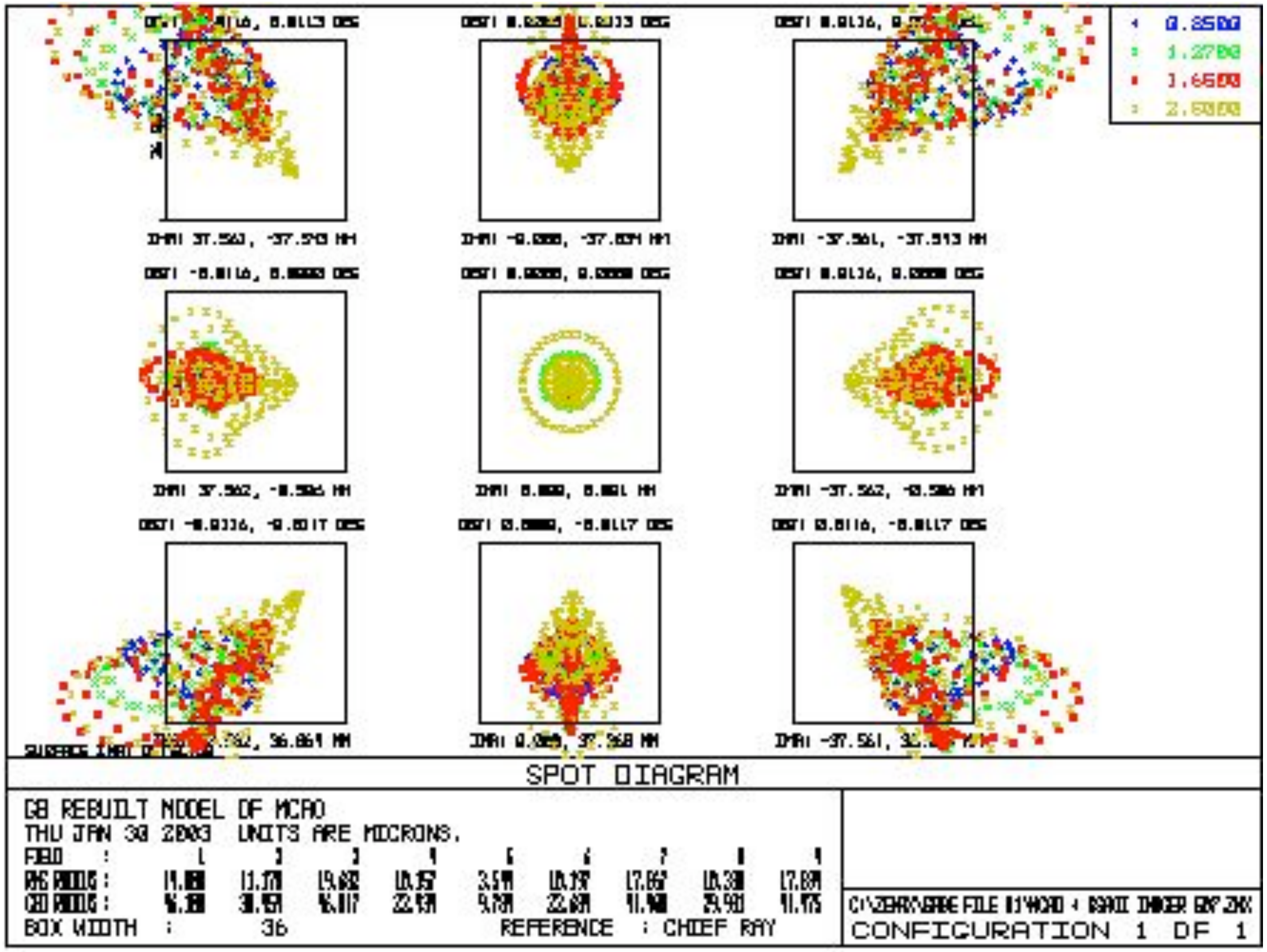
A2B and B2A tags

Table 21 — Profile type/profile tag and defined rendering intents



Profile Class	AToB0Tag	AToB1Tag	AToB2Tag	TRC/matrix & GrayTRC	BToA0Tag	BToA1Tag	BToA2Tag
Input	Device to PCS: perceptual	Device to PCS: colorimetric	Device to PCS: saturation	colorimetric	PCS to Device: perceptual	PCS to Device: colorimetric	PCS to Device: saturation
Display	Device to PCS: perceptual	Device to PCS: colorimetric	Device to PCS: saturation	colorimetric	PCS to Device: perceptual	PCS to Device: colorimetric	PCS to Device: saturation
Output	Device to PCS: perceptual	Device to PCS: colorimetric	Device to PCS:: saturation	undefined	PCS to Device: perceptual	PCS to Device: colorimetric	PCS to Device: saturation
ColorSpace	ColorSpace to PCS: perceptual	ColorSpace to PCS: colorimetric	ColorSpace to PCS: saturation	undefined	PCS to ColorSpace: perceptual	PCS to ColorSpace: colorimetric	PCS to ColorSpace: saturation
Abstract	PCS to PCS	undefined	undefined	undefined	undefined	undefined	undefined
DeviceLink	Device1 to Device2 rendering intent defined according to Table 19	undefined	undefined	undefined	undefined	undefined	undefined
Named Colour	undefined	undefined	undefined	undefined	undefined	undefined	undefined

Optical CAD programs make spot diagrams like these by ray tracing



```
G8 REBUILT MODEL OF MCAO
THU JAN 30 2003   UNITS ARE MICRONS.
FIELD     :      1        2        3        4        5        6        7        8        9
RMS RADIUS :  [illegible]  11.178  19.682  10.157   3.599  10.197  17.667  10.38   17.89
GEO RADIUS :  [illegible]  30.459  46.117  22.939   9.789  22.69   41.9    29.981  41.975
BOX WIDTH :         36                      REFERENCE  : CHIEF RAY
```

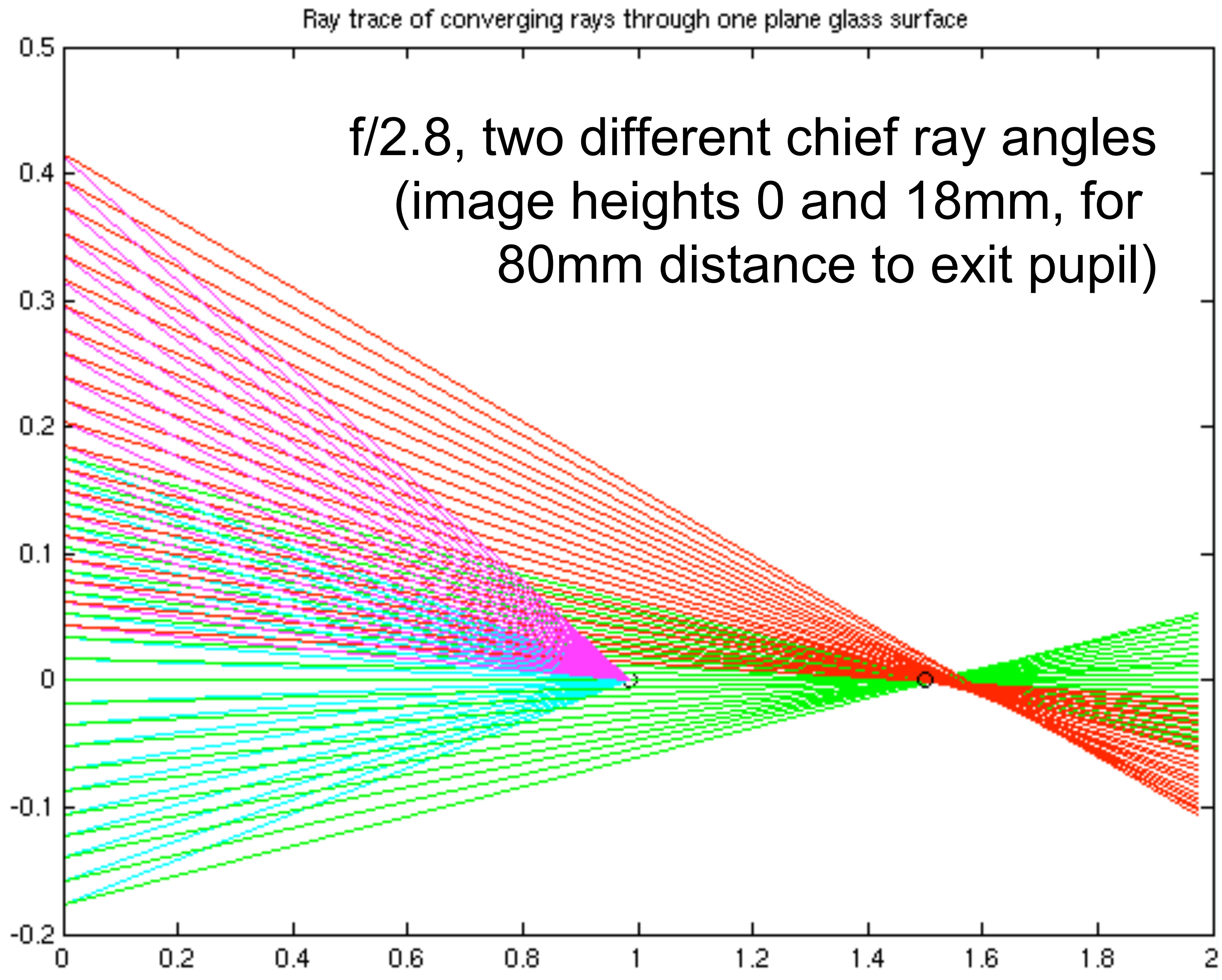
```
C:\ZEMAX\SAMPLE FILE [illegible] [illegible] [illegible]
CONFIGURATION  1  OF  1
```

http://www.mso.anu.edu.au/gsaoi/documentation/sdns/sdn02.07.htm

Ray tracing a plano optic

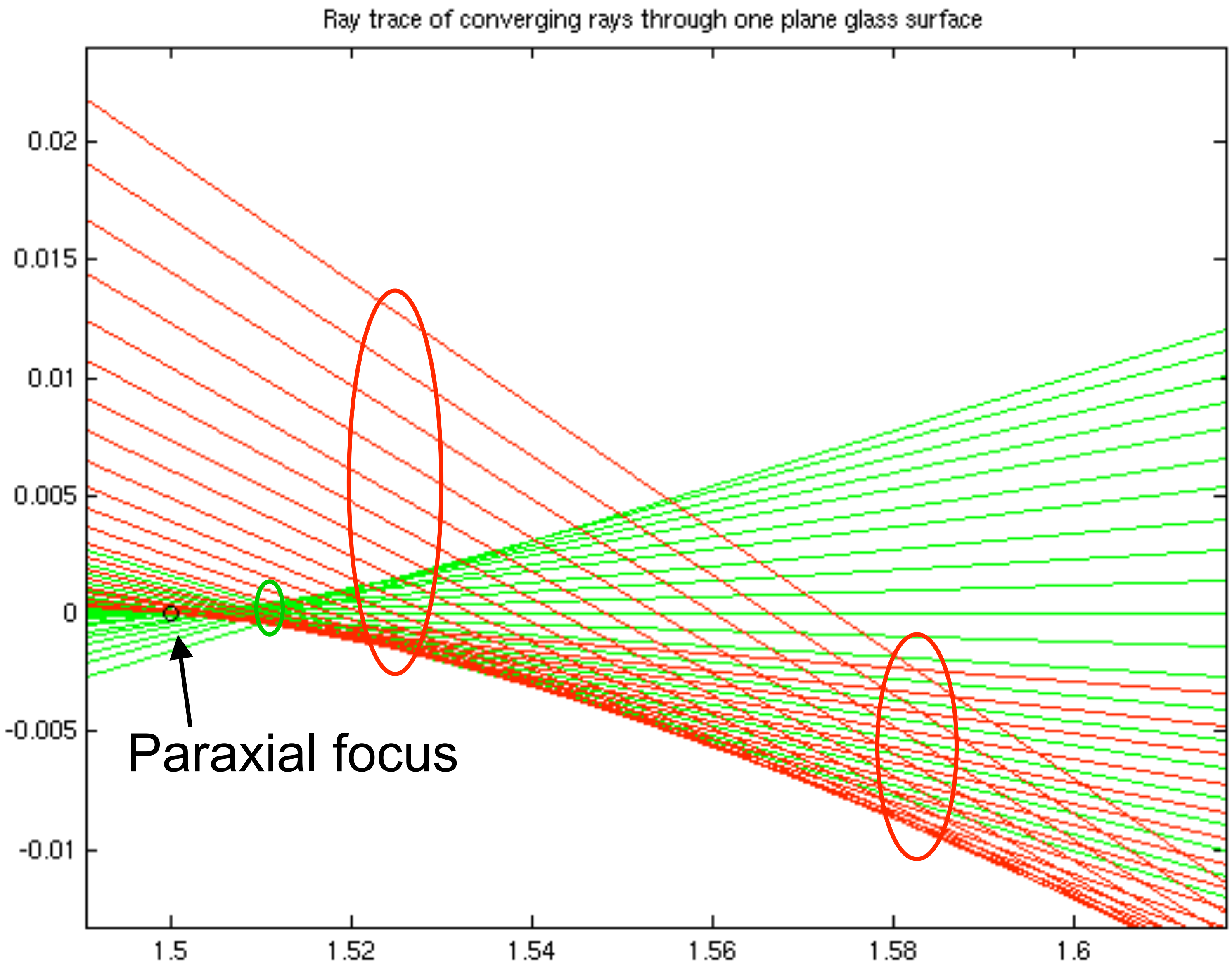
- A flat filter, a ‘plano’ optic, or two, is usually in the path between lens and sensor. What does it do to an otherwise ideal “stigmatic” optical system?
- Trace rays numerically. Send a bunch of rays toward a focal point, but recompute their paths where they enter (and leave?) the plane glass layer.

Converging rays, refracted at one plane glass surface

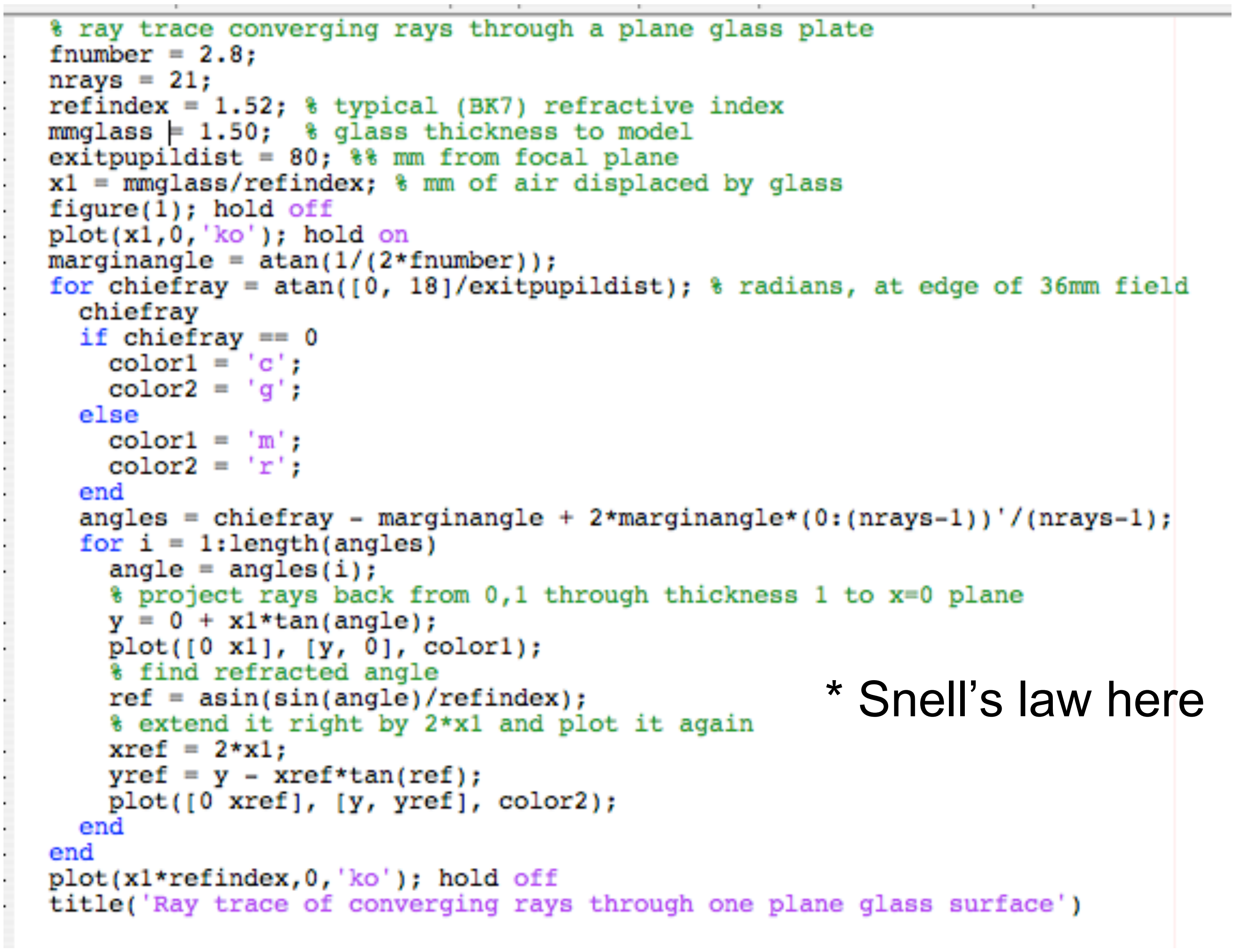


f/2.8, two different chief ray angles (image heights 0 and 18mm, for 80mm distance to exit pupil)

Close-up showing spherical, coma, and curvature of field



Ray tracing is very easy, for easy problems



```
% ray trace converging rays through a plane glass plate
fnumber = 2.8;
nrays = 21;
refindex = 1.52; % typical (BK7) refractive index
mmglass = 1.50;  % glass thickness to model
exitpupildist = 80; %% mm from focal plane
x1 = mmglass/refindex; % mm of air displaced by glass
figure(1); hold off
plot(x1,0,'ko'); hold on
marginangle = atan(1/(2*fnumber));
for chiefray = atan([0, 18]/exitpupildist); % radians, at edge of 36mm field
  chiefray
  if chiefray == 0
    color1 = 'c';
    color2 = 'g';
  else
    color1 = 'm';
    color2 = 'r';
  end
  angles = chiefray - marginangle + 2*marginangle*(0:(nrays-1))'/(nrays-1);
  for i = 1:length(angles)
    angle = angles(i);
    % project rays back from 0,1 through thickness 1 to x=0 plane
    y = 0 + x1*tan(angle);
    plot([0 x1], [y, 0], color1);
    % find refracted angle
    ref = asin(sin(angle)/refindex);
    % extend it right by 2*x1 and plot it again
    xref = 2*x1;
    yref = y - xref*tan(ref);
    plot([0 xref], [y, yref], color2);
  end
end
plot(x1*refindex,0,'ko'); hold off
title('Ray trace of converging rays through one plane glass surface')
```

* Snell's law here

It gets a little harder outside the plane of the optical axis, like for astigmatism

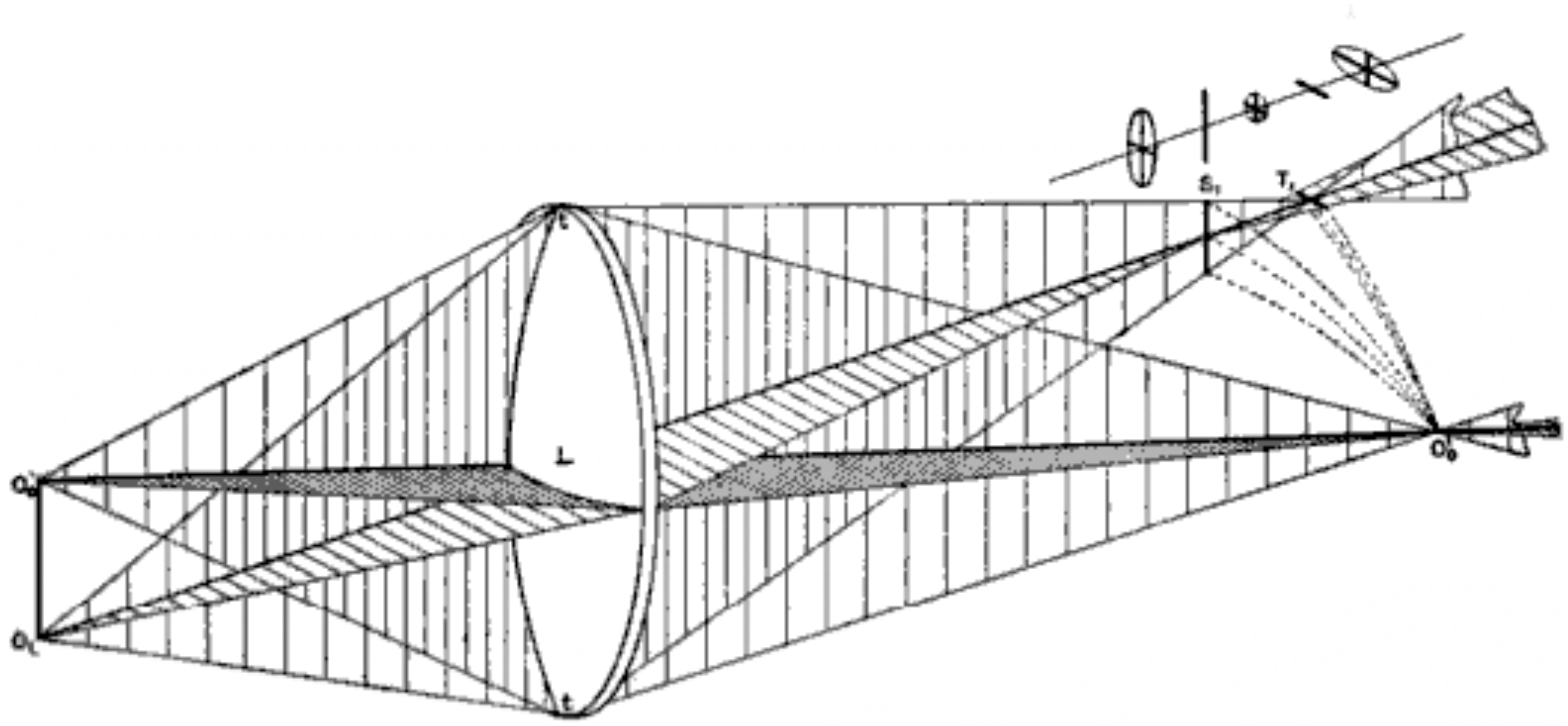
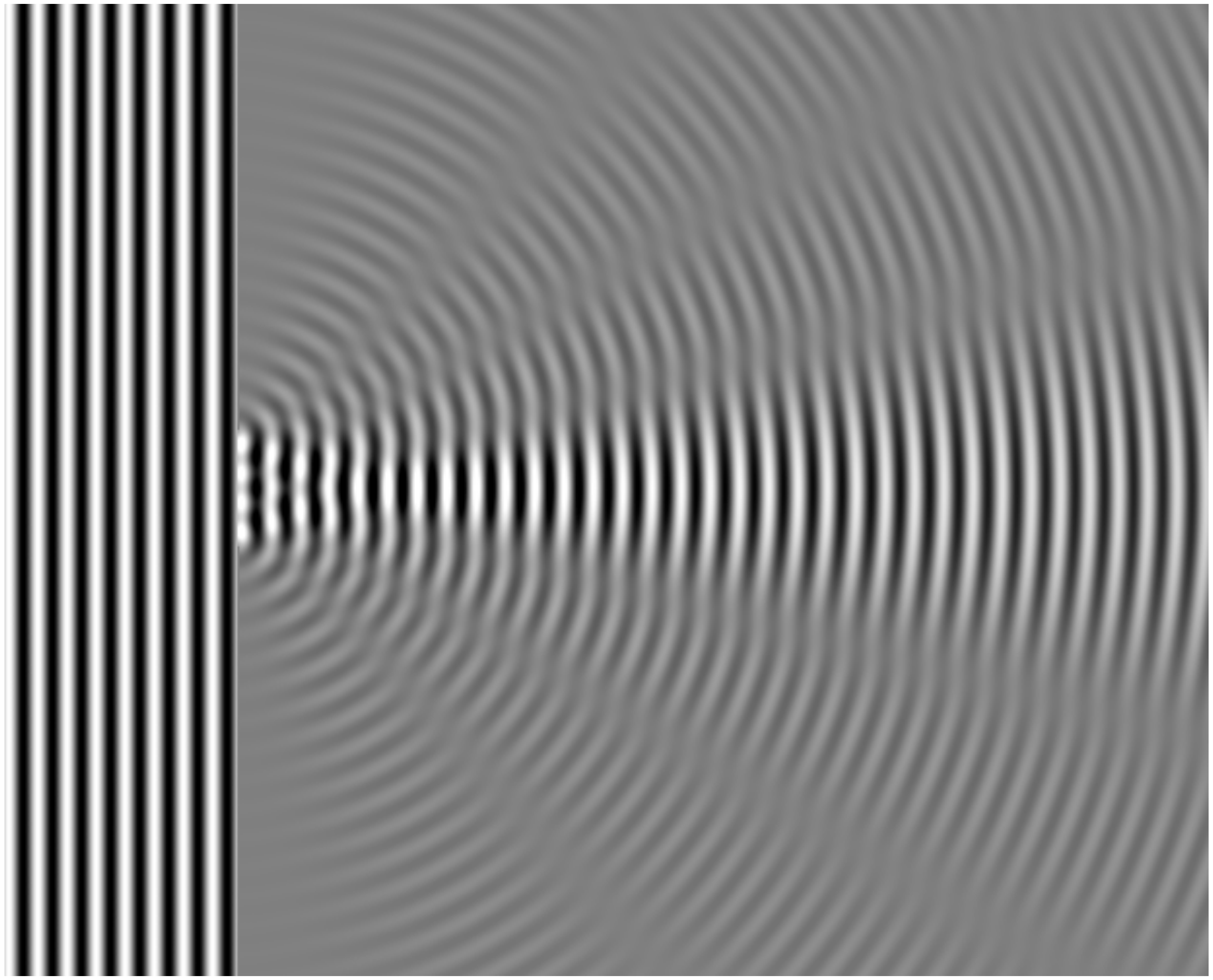


FIG. 22.

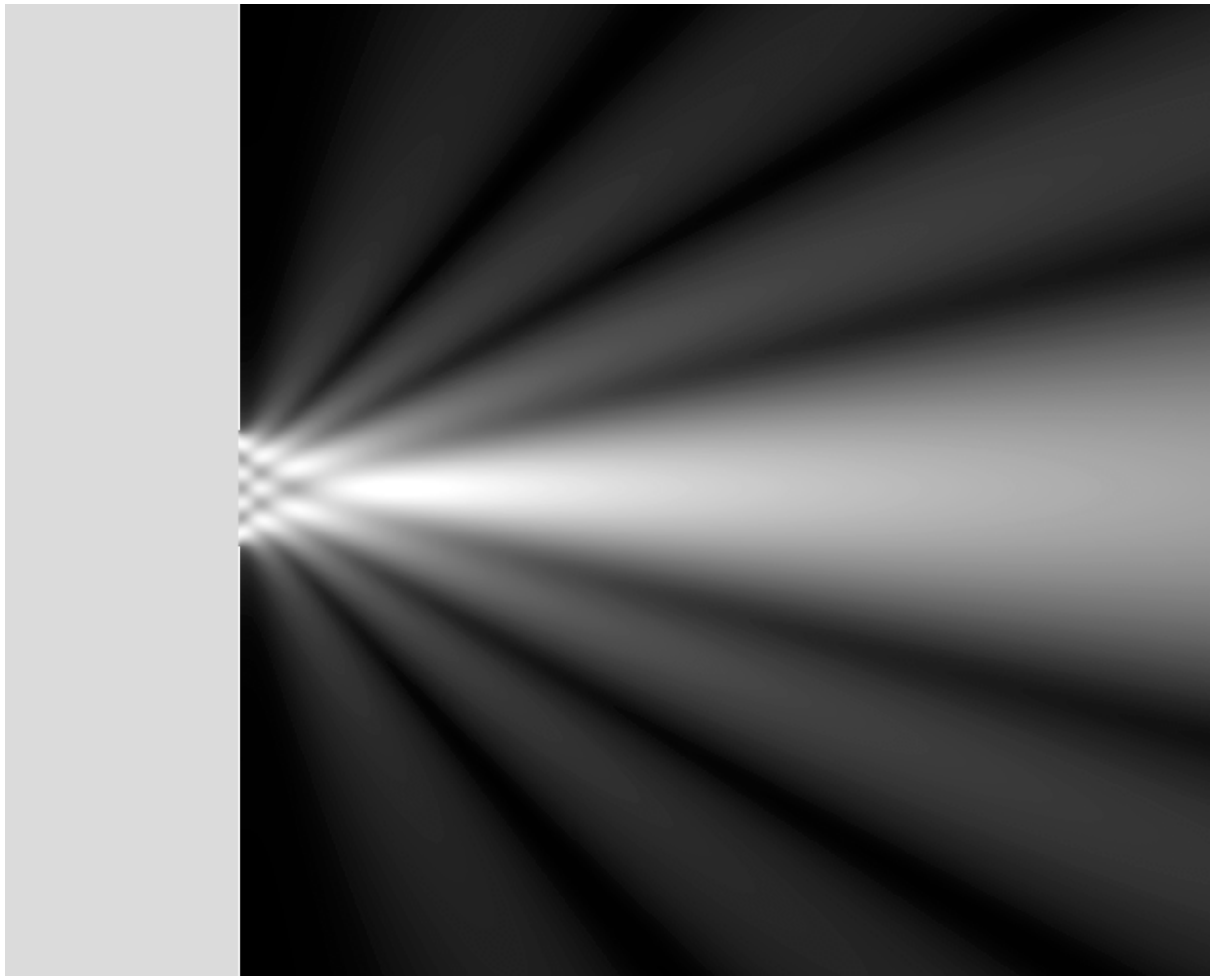
Diffraction Simulation

- Sum complex waves from many source or aperture points to every point where you care about the result
- Take the absolute value to get light intensity
- Points you care about can be anywhere, not necessarily in a focal plane

Slit with no focus (1D source array, equal phase across slit, 2D problem; showing real part, not magnitude)



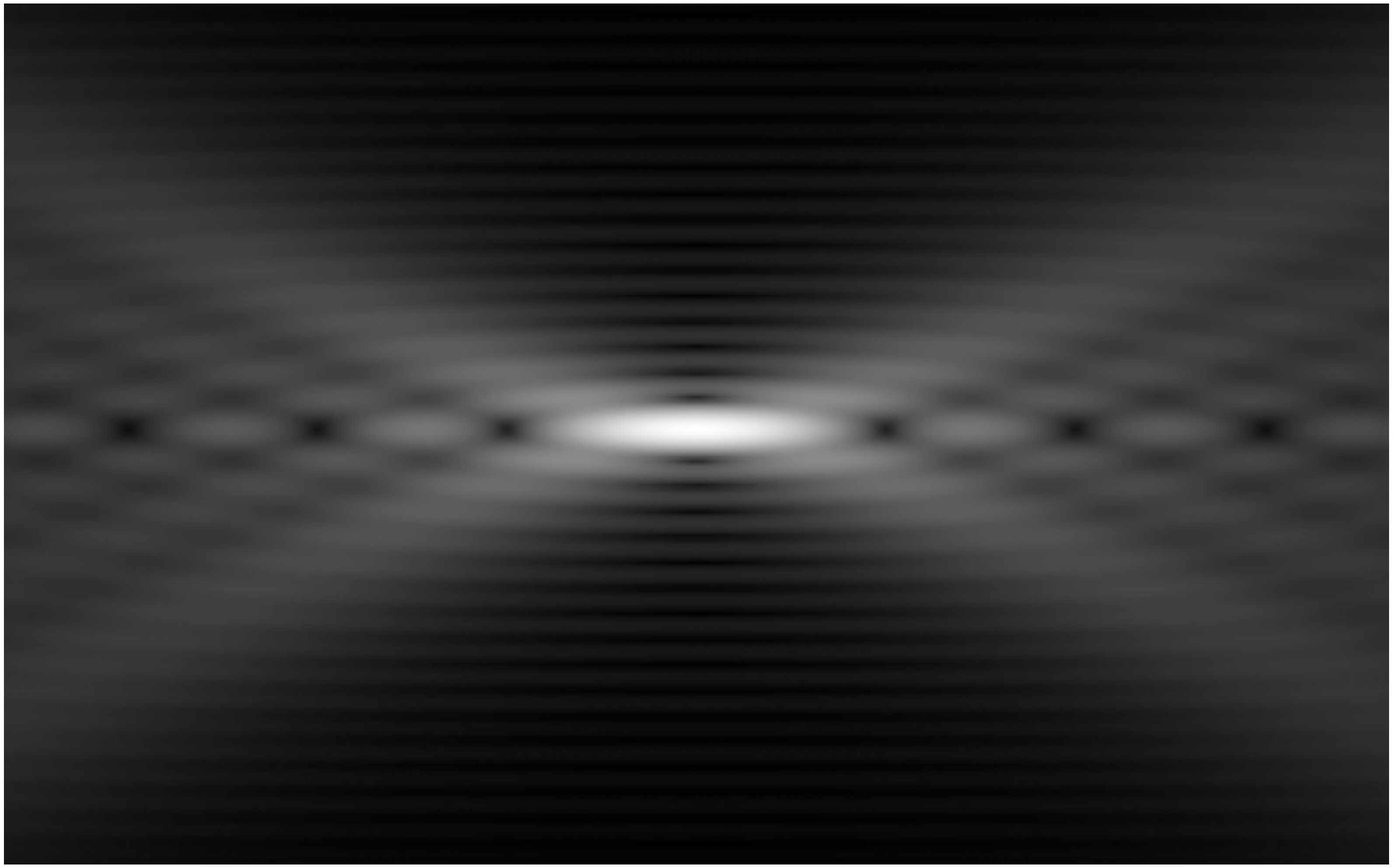
Absolute value of complex wave to show intensity



Diffraction-limited spot

- Model the wavefront as a discrete set of sources distributed across the aperture (exit pupil), phased to have equal phases at a point in the focal plane.
- Or use points on a spherical cap, all of identical phase.
- Generate points in focal plane, de-focused plane, or a cut the other way

The real deal: 2D array of sources, phased to converge; 2D slice of 3D result space; Airy null spacing in focal plane is N*lambda



(in case you couldn't see it, here it is brighter)
These patterns can then be reduced to MTF curves, filters, etc. to simulate optical systems