Project presentation

Open Source Programming

RawTherapee

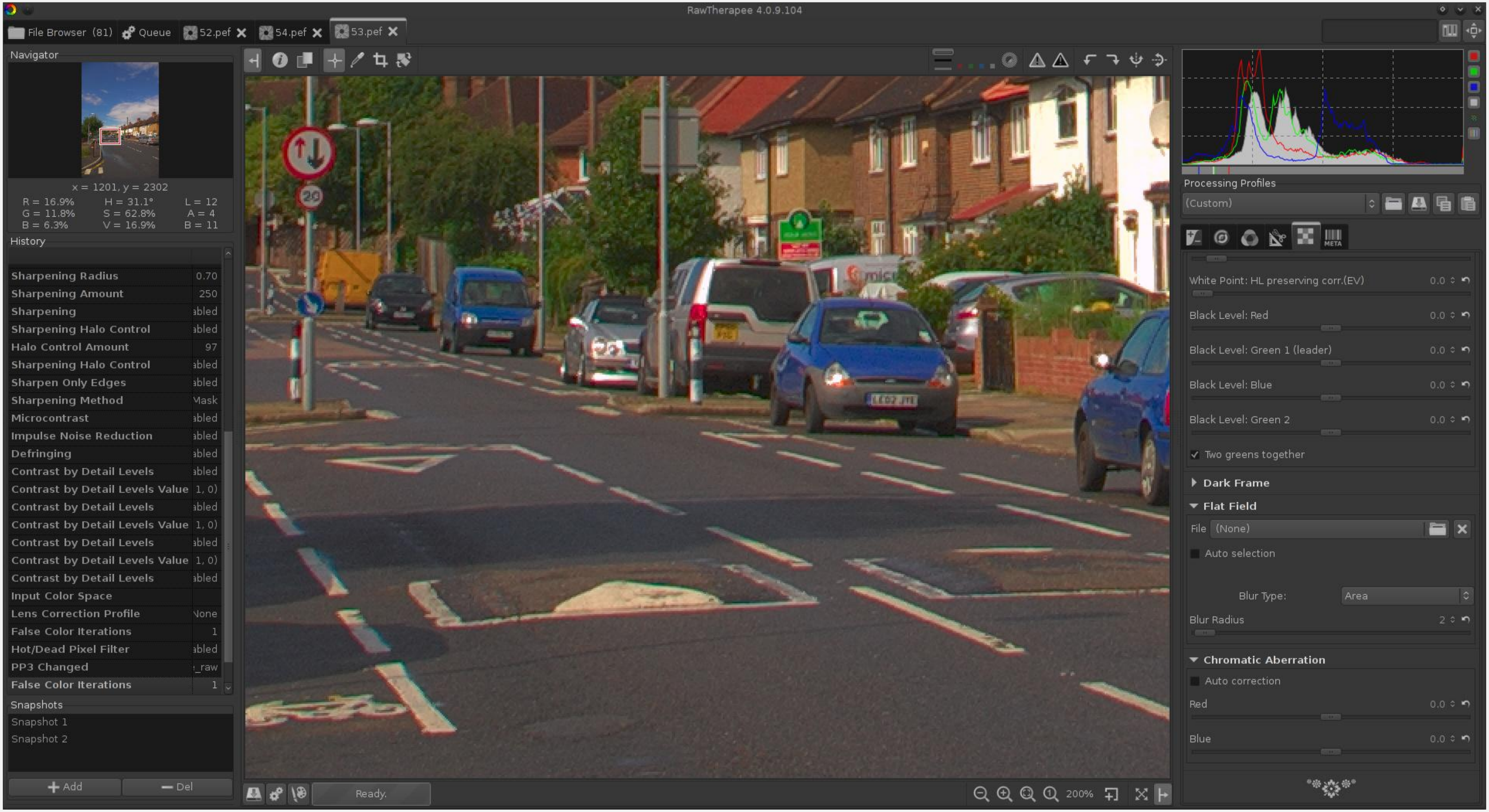
Tomáš Buzek

Project description

- **TH**e **E**xperimental **RA**w **P**hoto **E**ditor
- RAW image processing program
- GNU General Public License Version 3

- Since January 2010
- Cross-platform
- http://rawtherapee.com

Project description



x = 1201, y = 2302

R = 16.9% H = 31.1° L = 12
G = 11.8% S = 62.8% A = 4
B = 6.3% V = 16.9% B = 11

History

Sharpening Radius	0.70
Sharpening Amount	250
Sharpening	abled
Sharpening Halo Control	abled
Halo Control Amount	97
Sharpening Halo Control	abled
Sharpen Only Edges	abled
Sharpening Method	Mask
Microcontrast	abled
Impulse Noise Reduction	abled
Defringing	abled
Contrast by Detail Levels	abled
Contrast by Detail Levels Value	1, 0)
Contrast by Detail Levels	abled
Contrast by Detail Levels Value	1, 0)
Contrast by Detail Levels	abled
Contrast by Detail Levels Value	1, 0)
Contrast by Detail Levels	abled
Input Color Space	
Lens Correction Profile	None
False Color Iterations	1
Hot/Dead Pixel Filter	abled
PP3 Changed	_raw
False Color Iterations	1

Snapshots

Snapshot 1

Snapshot 2

Processing Profiles

(Custom)

White Point: HL preserving corr.(EV)	0.0
Black Level: Red	0.0
Black Level: Green 1 (leader)	0.0
Black Level: Blue	0.0
Black Level: Green 2	0.0

☑ Two greens together

Dark Frame

Flat Field

File (None)

☐ Auto selection

Blur Type: Area

Blur Radius 2

Chromatic Aberration

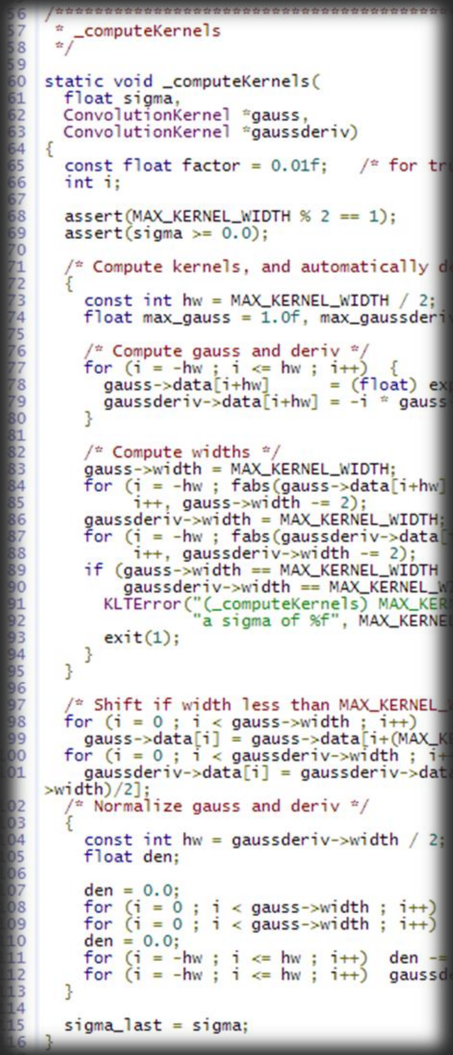
☐ Auto correction

Red 0.0

Blue 0.0

Technical specifications

- C++
- GTK+ front-end
- dcraw library (RAW files reading)
- Multi-threaded processing algorithms
- OpenCL in future releases



```
/*********************************************
 * _computeKernels
 */

static void _computeKernels(
  float sigma,
  ConvolutionKernel *gauss,
  ConvolutionKernel *gaussderiv)
{
  const float factor = 0.01f;   /* for tr
  int i;

  assert(MAX_KERNEL_WIDTH % 2 == 1);
  assert(sigma >= 0.0);

  /* Compute kernels, and automatically d
  {
    const int hw = MAX_KERNEL_WIDTH / 2;
    float max_gauss = 1.0f, max_gaussderi

    /* Compute gauss and deriv */
    for (i = -hw ; i <= hw ; i++)  {
      gauss->data[i+hw]      = (float) ex
      gaussderiv->data[i+hw] = -i * gauss
    }

    /* Compute widths */
    gauss->width = MAX_KERNEL_WIDTH;
    for (i = -hw ; fabs(gauss->data[i+hw]
         i++, gauss->width -= 2);
    gaussderiv->width = MAX_KERNEL_WIDTH;
    for (i = -hw ; fabs(gaussderiv->data[
         i++, gaussderiv->width -= 2);
    if (gauss->width == MAX_KERNEL_WIDTH
        gaussderiv->width == MAX_KERNEL_W
      KLTError("(_computeKernels) MAX_KER
               "a sigma of %f", MAX_KERNE
      exit(1);
    }
  }

  /* Shift if width less than MAX_KERNEL_
  for (i = 0 ; i < gauss->width ; i++)
    gauss->data[i] = gauss->data[i+(MAX_K
  for (i = 0 ; i < gaussderiv->width ; i+
    gaussderiv->data[i] = gaussderiv->dat
>width)/2];
  /* Normalize gauss and deriv */
  {
    const int hw = gaussderiv->width / 2;
    float den;

    den = 0.0;
    for (i = 0 ; i < gauss->width ; i++)
    for (i = 0 ; i < gauss->width ; i++)
    den = 0.0;
    for (i = -hw ; i <= hw ; i++)  den -=
    for (i = -hw ; i <= hw ; i++)  gaussd
  }

  sigma_last = sigma;
}
```

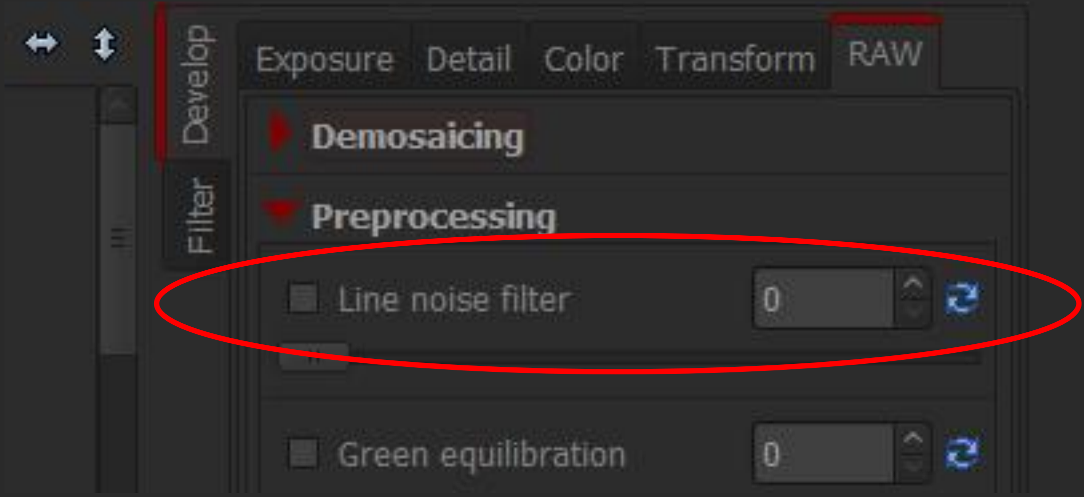
Project organization

- Large development team
- New versions released irregularly (last 8.3.2013)
- Developer Forum: http://rawtherapee.com/forum/
- Mercurial source control management tool
- Google Project Hosting

2,166 commits
61 contributors
127,108 lines of code

My task

- **Include Fixed Pattern Noise Correction algorithm**
- Algorithm already implemented in libraw
- Less sophisticated algorithm currently in application



□ Line noise filter 0

□ Green equilibration 0

My task