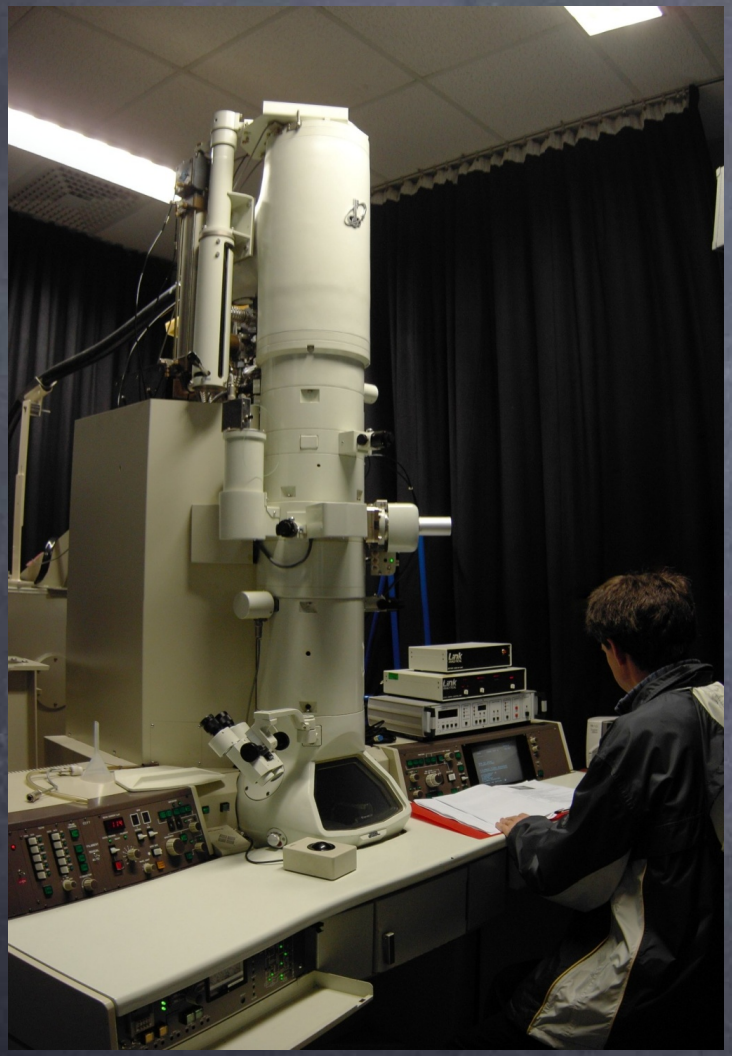
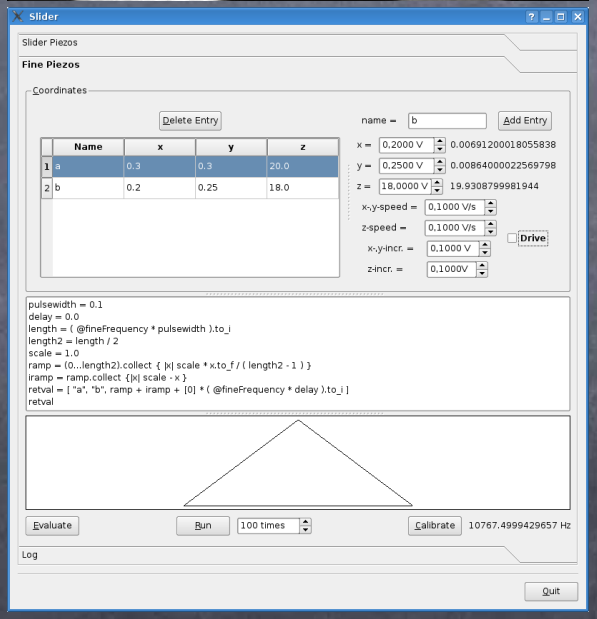
Ruby & Machine Vision

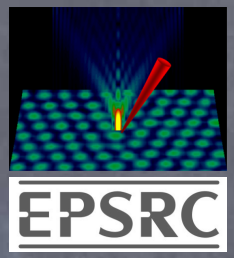
Jan Wedekind

Wednesday, February 4th 2009

UK EPSRC Nanorobotics Project







Microscopy Software

- telemanipulation
- drift compensation
- closed-loop control

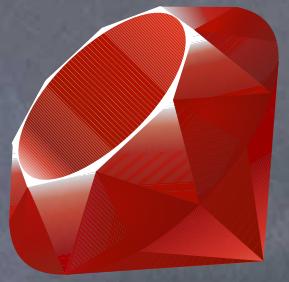
Machine Vision

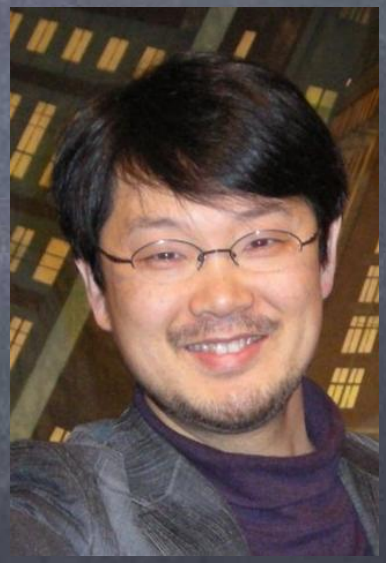
- real-time software
- system integration
- theoretical insights

Ruby Programming Language

Ruby

- created by Yukihiro Matsumoto
- released 1995 (free software(*), Ruby license)
- inspired by Perl, Python, Smalltalk, Eiffel, Ada, Lisp
- "pseudo simplicity": simple syntax $\Leftrightarrow$ multi-paradigm language
- highly portable
- Tiobe Programming Community Index #11
- 1.8.6 being superseded by 1.9.1





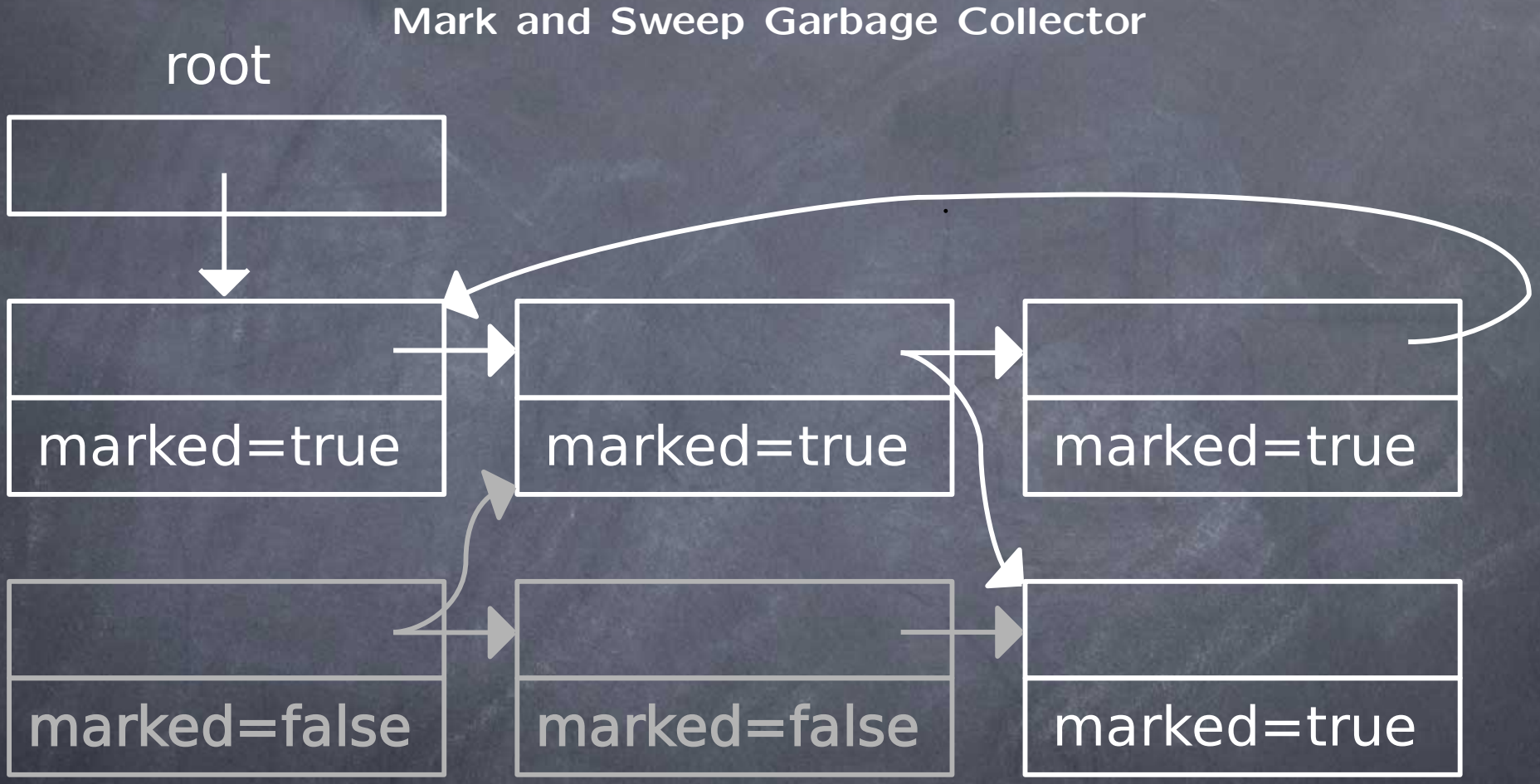
page	url
Ruby Homepage	`http://www.ruby-lang.org/`
Ruby Core-API	`http://www.ruby-doc.org/`
RubyForge	`http://rubyforge.org/`
Ruby Application Archive	`http://raa.ruby-lang.org/`

(*) `http://www.gnu.org/philosophy/free-sw.html`

Dynamic Typing

```ruby
#!/usr/bin/env ruby
def test( a, b )
  a + b
end
x = test( 3, 5 )      # x -> 8
x = test( 'a', 'b' ) # x -> 'ab'
```

Garbage Collector



http://www.brpreiss.com/books/opus5/html/page424.html

(Pure) Object-Oriented, Single-Dispatch

```ruby
#!/usr/bin/env ruby
class Numeric
  def plus(x)
    self.+(x)
  end
end
y = 5.plus 6
# y is now equal to 11
```

http://www.ruby-lang.org/en/about/

Mixins

```ruby
#!/usr/bin/env ruby
module TimesThree
  def three_times
    self + self + self
  end
end
class String
  include TimesThree
end
'abc'.three times # -> 'abcabcabc'
```

Exception Handling

```ruby
#!/usr/bin/env ruby
begin
  print "Enter filename: "
  STDOUT.flush
  file_name = STDIN.readline.delete( "\n\r" )
  file = File.new file_name, 'r'
  # ...
rescue Exception => e
  puts "Error opening file '#{file_name}': #{e.message}"
end
```

Closures

Unifying Concept for Iterators, Function Objects, and Loops

```ruby
#!/usr/bin/env ruby
def inc( i )
  lambda do |v|
    v + i
  end
end
t = inc( 5 )
t.call( 3 ) # -> 8
[ 1, 2, 3 ].each do |x|
  puts x
end
[ 1, 2, 3 ].collect do |x|
  x ** 2
end           # -> [1, 4, 9]
[ 1, 2, 3 ].inject do |v,x|
  v + x
end           # -> 6
```

Continuations

```ruby
#!/usr/bin/env ruby
def test( c2 )
  callcc do |c1|
    return c1
  end
  c2.call
end
callcc do |c2|
  c1 = test( c2 )
  c1.call
end
```

Introspection

Program can "see" itself

```
#!/usr/bin/env ruby
x = 5                          # -> 5
x.class                        # -> Fixnum
x.class.class                  # -> Class
x.class.superclass             # -> Integer
x.is_a?( Fixnum )              # -> true
Fixnum < Integer               # -> true
5.respond_to?( '+' )           # -> true
5.methods.grep( /^f/ ).sort # -> ["floor", "freeze", "frozen?"]
```

Metaprogramming

Interpreter modifies Program

```ruby
#!/usr/bin/env ruby
eval 'x=5'                  # x -> 5
a = [ 1 ]
a.instance_eval do
  push 2
end                         # a -> [ 1, 2 ]
a.send( 'push', 3 )         # a -> [ 1, 2, 3 ]
Object.const_get( 'String' ).class_eval do
  define_method 'test' do
    reverse
  end
end
'abc'.reverse               # -> 'cba'
```

Reification

Program modifies Interpreter

```ruby
#!/usr/bin/env ruby
class Numeric
  def method_missing( name, *args )
    prefix = Regexp.new( "^#{name}" )
    full_name = methods.find { |id| id =~ prefix }
    if full_name
      send( full_name, *args )
    else
      super
    end
  end
end
5.mod 2 # calls 5.modulo 2
```

Ruby Extensions

C Library

```c
// gcc -shared -fPIC -I/usr/lib/ruby/1.8/x86_64-linux \
//   -o myextension.so myextension.c
#include <ruby.h>
#include <math.h>

VALUE wrap_logx( VALUE self, VALUE x )
{
  return rb_float_new( log( NUM2DBL( self ) ) / log( NUM2DBL( x ) ) );
}

void Init_myextension(void) {
  VALUE numeric = rb_const_get( rb_cObject, rb_intern( "Numeric" ) );
  rb_define_method( numeric, "logx", RUBY_METHOD_FUNC( wrap_logx ), 1 );
}
```

Invoking Ruby Program

```ruby
#!/usr/bin/env ruby
require 'myextension'
e = 1024.logx( 2 )
puts "2 ** #{e} = 1024"
```

http://www.rubyist.net/~nobu/ruby/Ruby_Extension_Manual.html

HornetsEye - Ruby Extension for Machine Vision

Free Software Project

- Real-Time Machine Vision
- Ruby Extension
- released under GNU General Public License
- 2 years development
- 22000 lines of code

http://www.wedesoft.demon.co.uk/hornetseye-api/

http://rubyforge.org/projects/hornetseye/

http://sourceforge.net/projects/hornetseye/

https://launchpad.net/hornetseye/

http://raa.ruby-lang.org/project/hornetseye/

Using existing Free Software

Libraries Integrated

C++ Boost

libdc1394

DotGNU

FFTW

MPlayer

OpenEXR

Qt4-QtRuby

RMagick

NArray

libxine

Xorg, Mesa3D

Tools in Use

Bazaar

GCC

make, automake, autoconf

NaturalDocs

NSIS

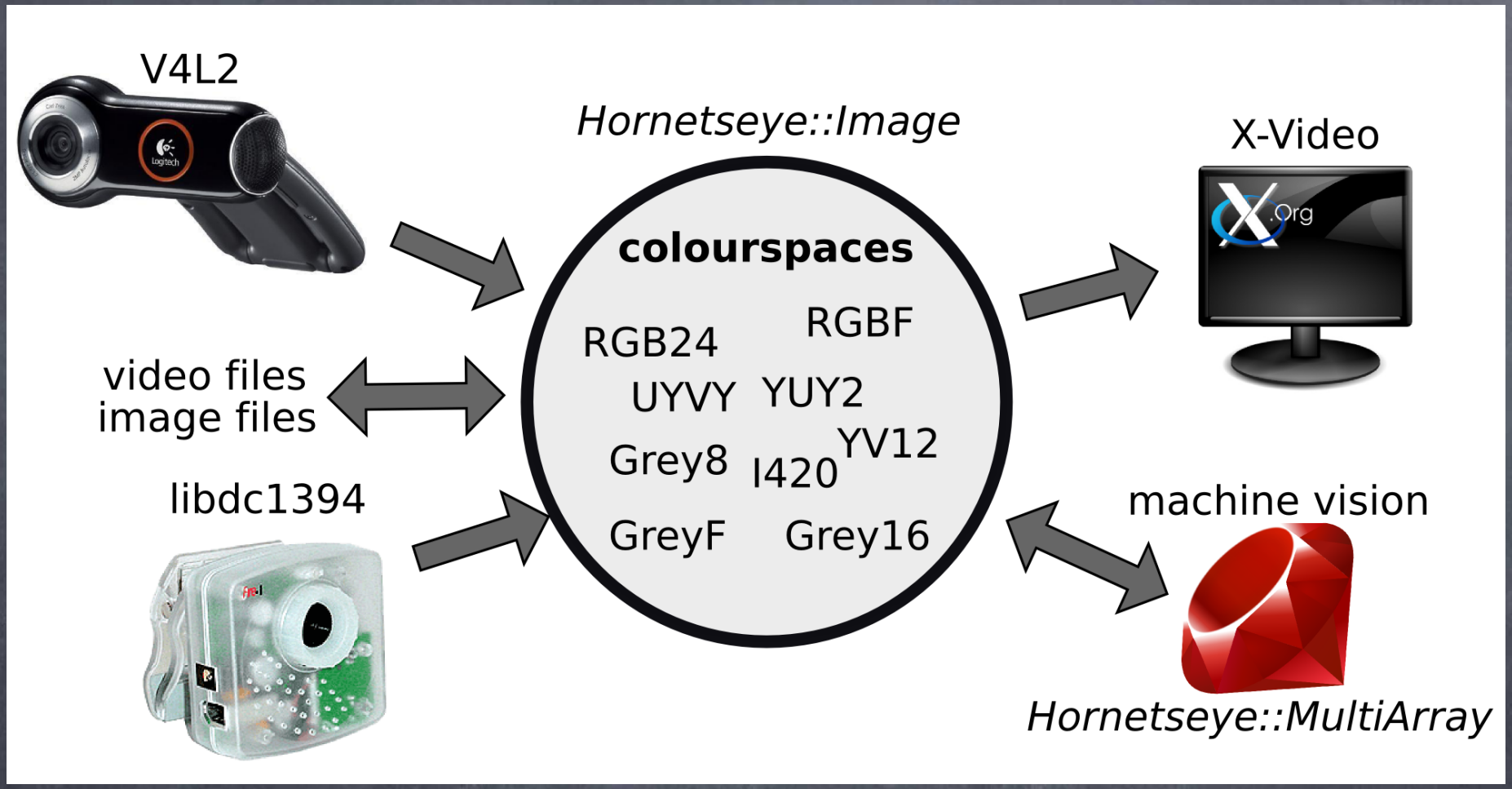
Ruby

Inpput/Output Classes

Input/Output Classes

 / 	
V4LInput	VFWInput
V4L2Input	DShowInput
DC1394Input	—
XineInput	—
MPlayerInput	MPlayerInput
MEncoderOutput	MEncoderOutput
X11Display	W32Display
X11Window	W32Window
XImageOutput	GDIOutput
OpenGLOutput	—
XVideoOutput	—

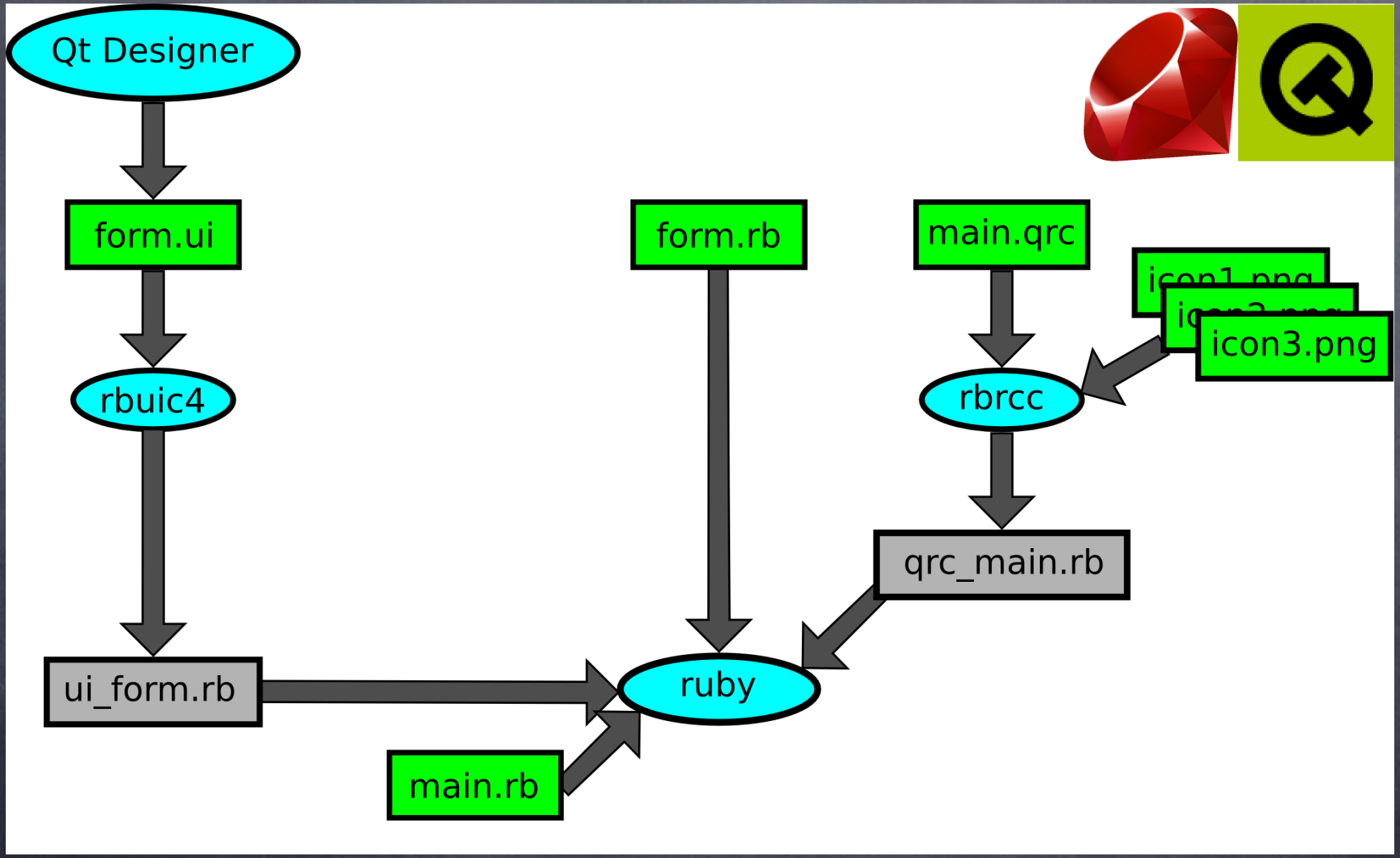
Colourspace Conversions



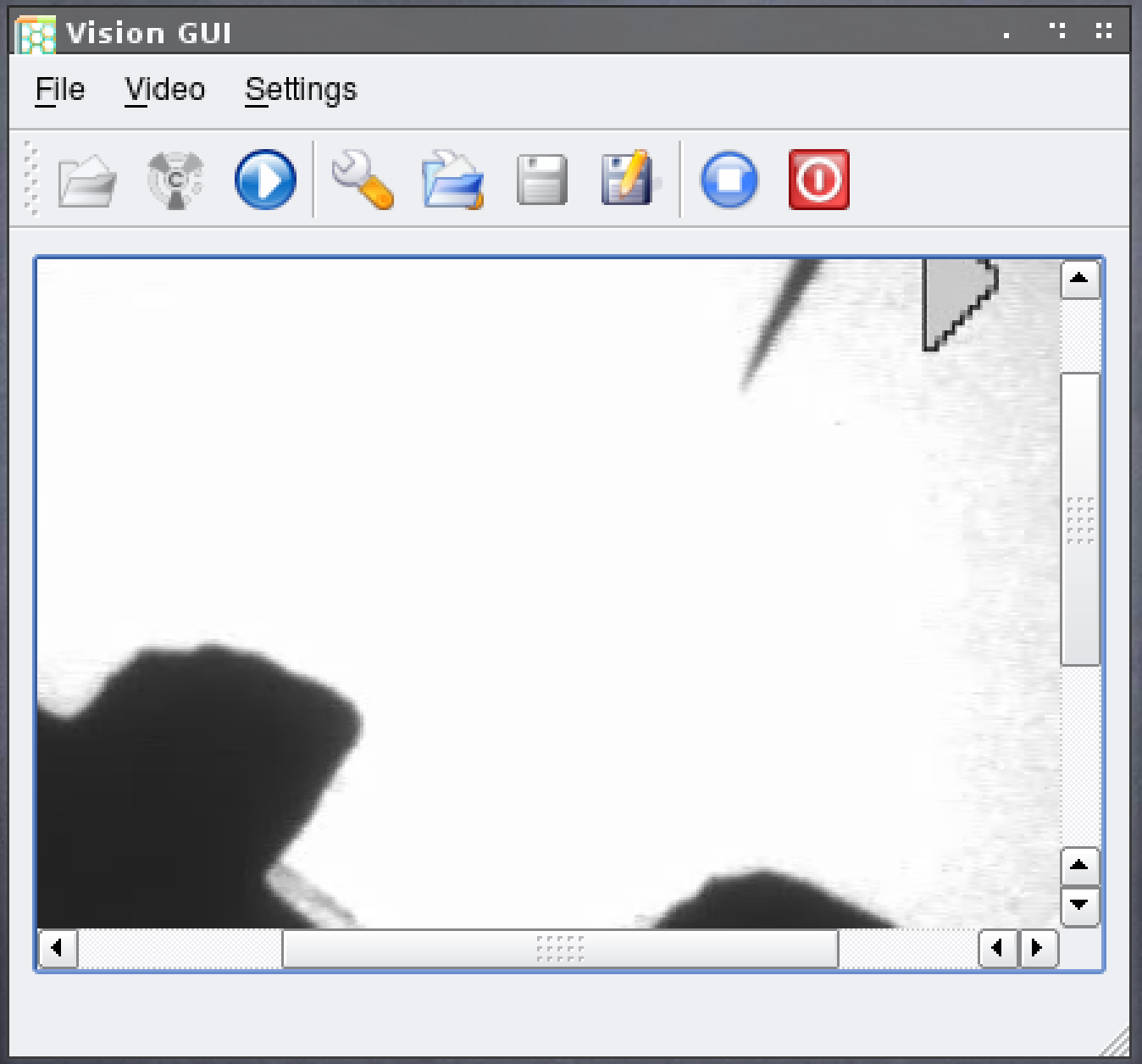
$$\begin{pmatrix} Y \\ C_b \\ C_r \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.168736 & -0.331264 & 0.500 \\ 0.500 & -0.418688 & -0.081312 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 0 \\ 128 \\ 128 \end{pmatrix}$$

also see: `http://fourcc.org/`

GUI Integration I/II



GUI Integration II/II



Just-In-Time Compiler

```
#!/usr/bin/env
require 'hornetseye'
include Hornetseye
fun = JITFunction.compile( JITType::DFLOAT,
                           JITType::DFLOAT,
                           JITType::DFLOAT ) do |f,a,b|
  Math.log( a ) / Math.log( b )
end
fun.call( 1024, 2 ) # -> 10.0
```

Some other Vision Libraries I/II

feature	Blepo	Camellia	CMVision	libCVD	EasyVision	Filters	Framewave	Gamera	Gandalf
Camera Input	✔	✖	✔	✔	✔	✖	✖	✖	✖
Image Files	✔	✔	✖	✔	✔	✔	✔	✔	✔
Video Files	✖	✖	✖	✔	✖	✖	✔	✖	✖
Display	✔	✖	✔	✔	✔	✖	✖	✔	✔
Scripting	✖	✔	✖	✖	✔	✖	✖	✔	✖
Warps	✖	✖	✖	✔	✖	✖	✔	✖	✔
Histograms	✖	✖	✔	✖	✖	✔	✖	✔	✔
Custom Filters	✔	✖	✖	✔	✖	✔	✔	✔	✔
Fourier Transforms	✔	✖	✖	✖	✖	✖	✖	✖	✔
Feature Extraction	✔	✖	✖	✔	✔	✔	✔	✔	✔
Feature Matching	✔	✖	✖	✖	✔	✖	✖	✖	✖
GPL compatible	✔	✔	✔	✔	?	✔	✔	✔	✔

Also see `http://www.wedesoft.demon.co.uk/hornetseye-api/files/Links-txt.html`

Some other Vision Libraries II/II

feature	HornetsEye	ITK/VTK	IVT	LTIlib	Lush	Mimas	NASA V. W.	OpenCV	SceneLib	VIGRA
Camera Input	✔	✖	✔	✔	✔	✔	✖	✔	✔	✖
Image Files	✔	✔	✔	✔	✔	✔	✔	✔	✖	✔
Video Files	✔	✖	✔	✖	✖	✔	✖	✔	✖	✖
Display	✔	✔	✔	✔	✔	✔	✖	✔	✔	✖
Scripting	✔	✖	✖	✖	✔	✖	✖	✔	✖	✖
Warps	✔	✔	✖	✖	✔	✔	✔	✔	✖	✖
Histograms	✔	✔	✔	✔	✔	✖	✖	✔	✖	✖
Custom Filters	✔	✔	✔	✔	✔	✔	✔	✖	✖	✔
Fourier Transforms	✔	✔	✖	✖	✔	✔	✖	✔	✖	✔
Feature Extraction	✔	✔	✔	✔	✔	✔	✖	✔	✔	✔
Feature Matching	✖	✖	✔	✔	✖	✖	✖	✔	✔	✖
GPL compatible	✔	✔	✔	✔	✔	✔	✖	✔	✔	✔

Also see `http://www.wedesoft.demon.co.uk/hornetseye-api/files/Links-txt.html`

Dense Scripts

OpenCV + Python

```
#! /usr/bin/env python
import sys
from opencv import cv
from opencv import highgui
highgui.cvNamedWindow( 'Camera' )
capture = highgui.cvCreateCameraCapture( -1 )
while 1:
    frame = highgui.cvQueryFrame( capture )
    gray = cv.cvCreateImage( cv.cvSize( frame.width, frame.height), 8, 1 )
    cv.cvCvtColor( frame, gray, cv.CV_BGR2GRAY )
    highgui.cvShowImage( 'Camera', gray )
    if highgui.cvWaitKey( 5 ) > 0:
        break
```

HornetsEye + Ruby

```
#!/usr/bin/env ruby
require 'hornetseye'
include Hornetseye
capture = V4L2Input.new
X11Display.show { capture.read.to_grey8 }
```

Interactive Ruby (IRB)

```
require 'hornetseye'
include Hornetseye
img = MultiArray.load_rgb24 '/home/engjw/test/hornetseye/data/images/world.jpg'
# MultiArrayubytergb(640,320):
# [ [ RGB( 0, 20, 55 ), RGB( 0, 20, 55 ), RGB( 0, 20, 55 ), .... ],
#   [ RGB( 17, 36, 69 ), RGB( 17, 36, 69 ), RGB( 18, 37, 70 ), .... ],
#   [ RGB( 9, 24, 55 ), RGB( 9, 24, 55 ), RGB( 8, 23, 54 ), .... ],
#   [ RGB( 8, 22, 51 ), RGB( 8, 22, 51 ), RGB( 7, 21, 50 ), .... ],
#   [ RGB( 8, 19, 49 ), RGB( 8, 19, 49 ), RGB( 8, 19, 49 ), .... ],
# ....
filter = MultiArray.to_multiarray( [ [ 1, 1, 1 ], [ 1, 1, 1 ], [ 1, 1, 1 ] ] ).to_usint
# MultiArrayusint(3,3):
# [ [ 1, 1, 1 ],
#   [ 1, 1, 1 ],
#   [ 1, 1, 1 ] ]
img.correlate( filter )
# MultiArrayusintrgb(640,320):
# [ [ RGB( 34, 112, 248 ), RGB( 52, 169, 373 ), RGB( 54, 171, 375 ), .... ],
#   [ RGB( 52, 160, 358 ), RGB( 78, 240, 537 ), RGB( 79, 241, 538 ), .... ],
#   [ RGB( 68, 164, 350 ), RGB( 101, 245, 524 ), .... ],
#   [ RGB( 50, 130, 310 ), RGB( 73, 193, 463 ), RGB( 72, 192, 462 ), .... ],
#   [ RGB( 45, 123, 306 ), RGB( 66, 182, 458 ), RGB( 64, 182, 457 ), .... ],
# ....
```

Opening Webcam/Framegrabber

```
#!/usr/bin/env ruby
require 'hornetseye'
include Hornetseye
input = V4L2Input.new
img = input.read
img.show
```

Capture Image

```
#!/usr/bin/env ruby
require 'hornetseye'
include Hornetseye
input = V4L2Input.new
img = nil
X11Display.show { img = input.read_rgb24 }
img.save_rgb24 'test.jpg'
```

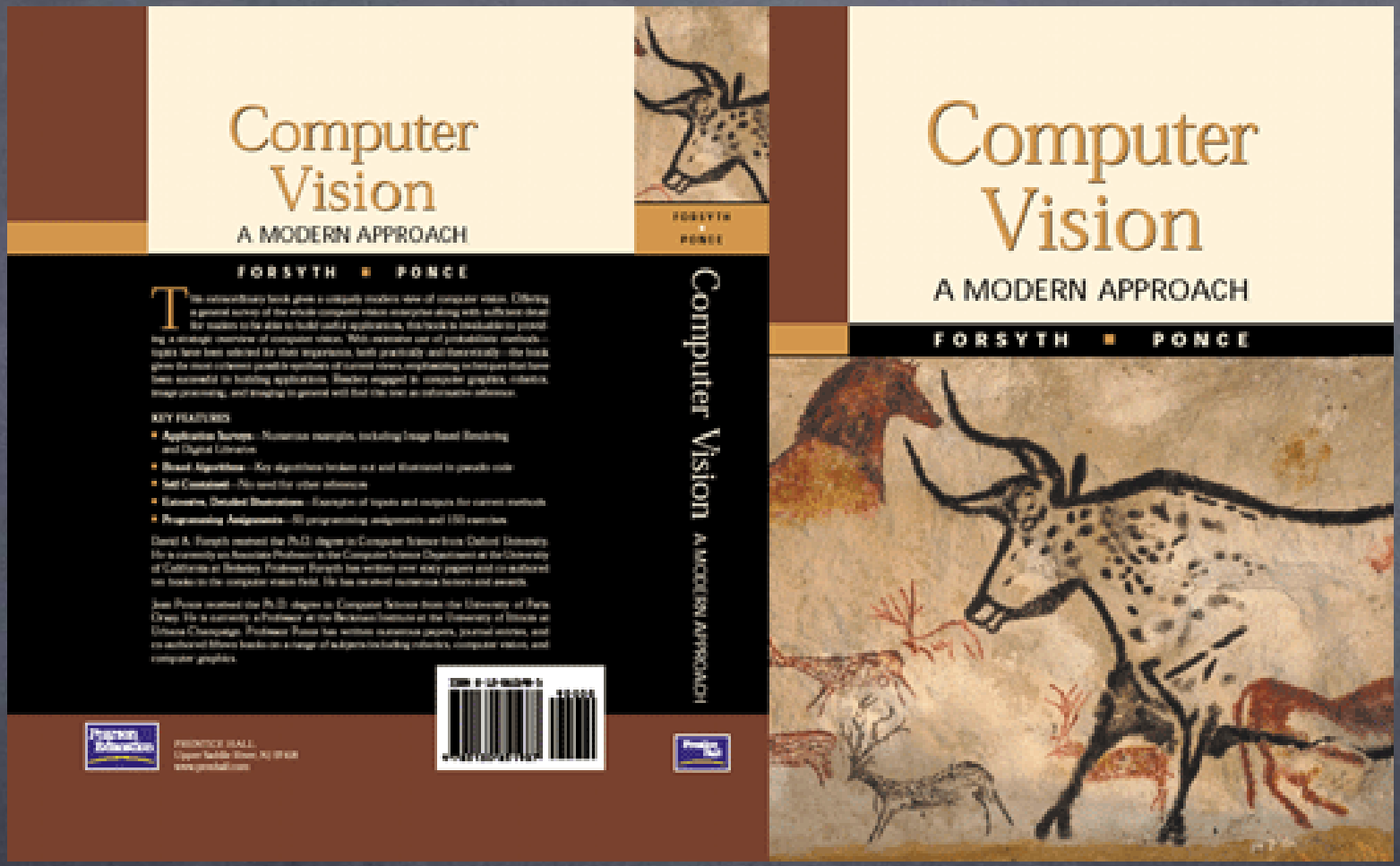
Capture Video

```ruby
#!/usr/bin/env ruby
require 'hornetseye'
include Hornetseye
input = V4L2Input.new( '/dev/video0', 640, 480 )
output = MEncoderOutput.new( 'test.avi', 10,
  '-ovc lavc -lavcopts vcodec=msmpeg4:vhq:vbitrate=4000' )
X11Display.show do
  img = input.read
  output.write( img )
  img
end
output.close
```

Center of Gravity

```ruby
#!/usr/bin/env ruby
require 'hornetseye'
include Hornetseye
input = V4L2Input.new '/dev/video0', 640, 480
idx = MultiArray.lint( input.width, input.height ).indgen!
x = idx % idx.shape[0]
y = idx / idx.shape[0]
img = nil
X11Display.show { img = input.read_rgb24 }
ref = img[ 0, 0 ]
X11Display.show do
  img = input.read_rgb24.to_sintrgb
  cdiff = img - ref
  diff = cdiff.r.abs + cdiff.g.abs + cdiff.b.abs
  mask = ( diff < 40 ).to_ubyte
  n = mask.sum
  puts "x = #{( mask * x ).sum / n}, y = #{(mask * y ).sum / n}" if n > 0
  ( img / 2 ) * ( mask + 1 )
end
```

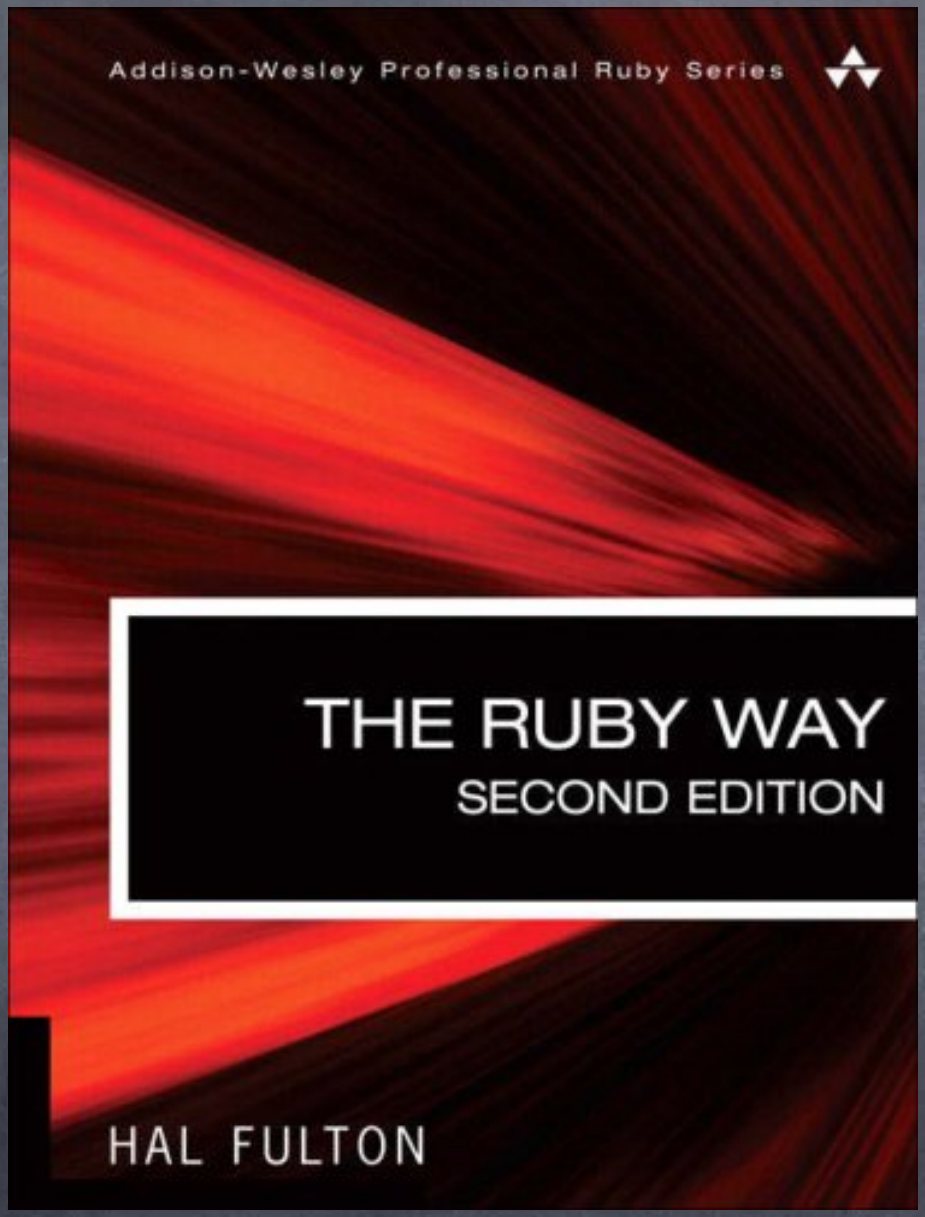
David A. Forsyth, Jean Ponce - Computer Vision: A modern Approach



Location: Adsetts Centre, Shelfmark: 006.37 FO (LEVEL 2)

http://catalogue.shu.ac.uk/search~S1/t?Computer%20vision:%20a%20modern%20approach

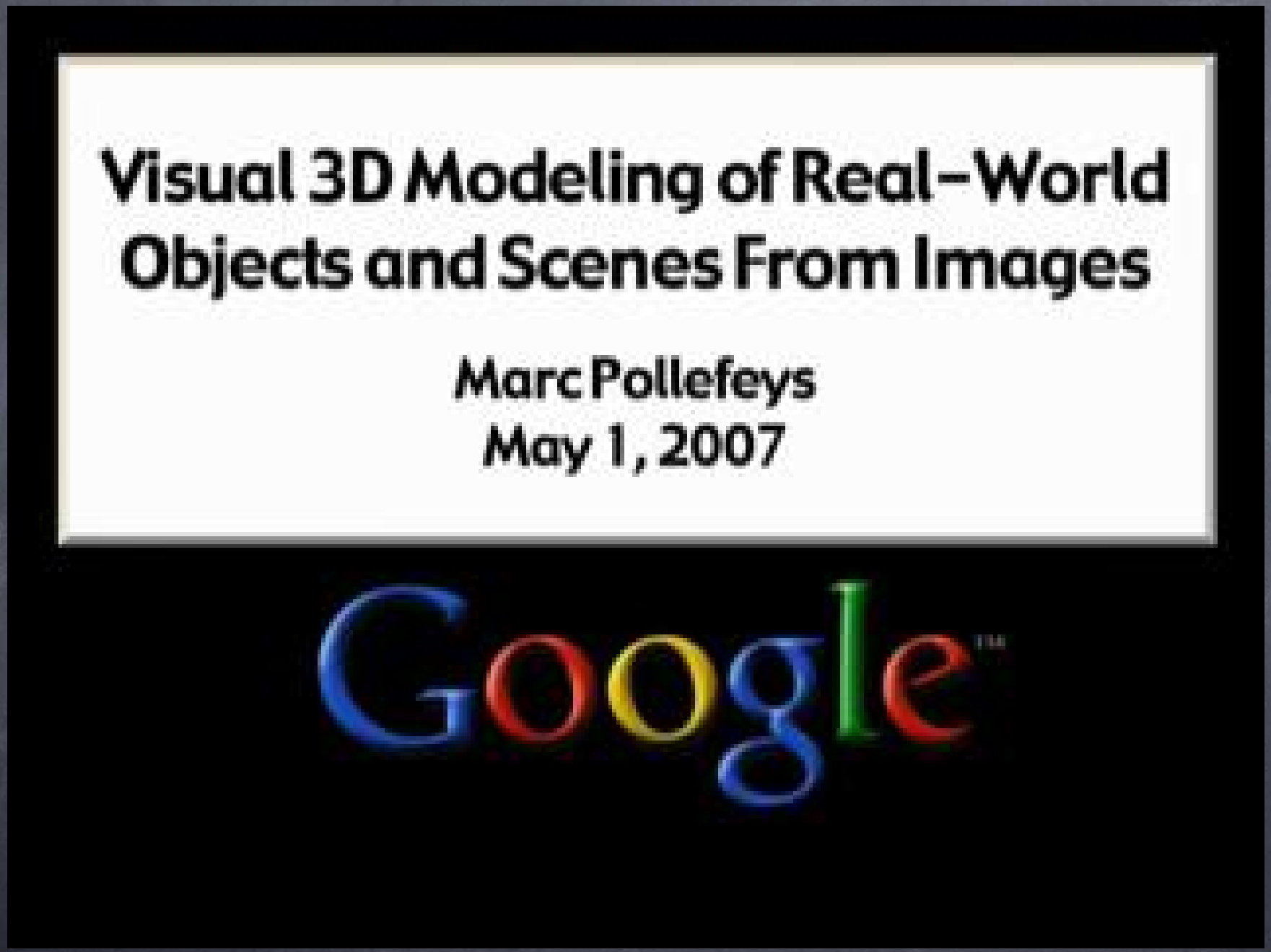
Hal Fulton - The Ruby Way



Location: Adsetts Centre, Shelfmark: 005.133 RUB FU (LEVEL 2)

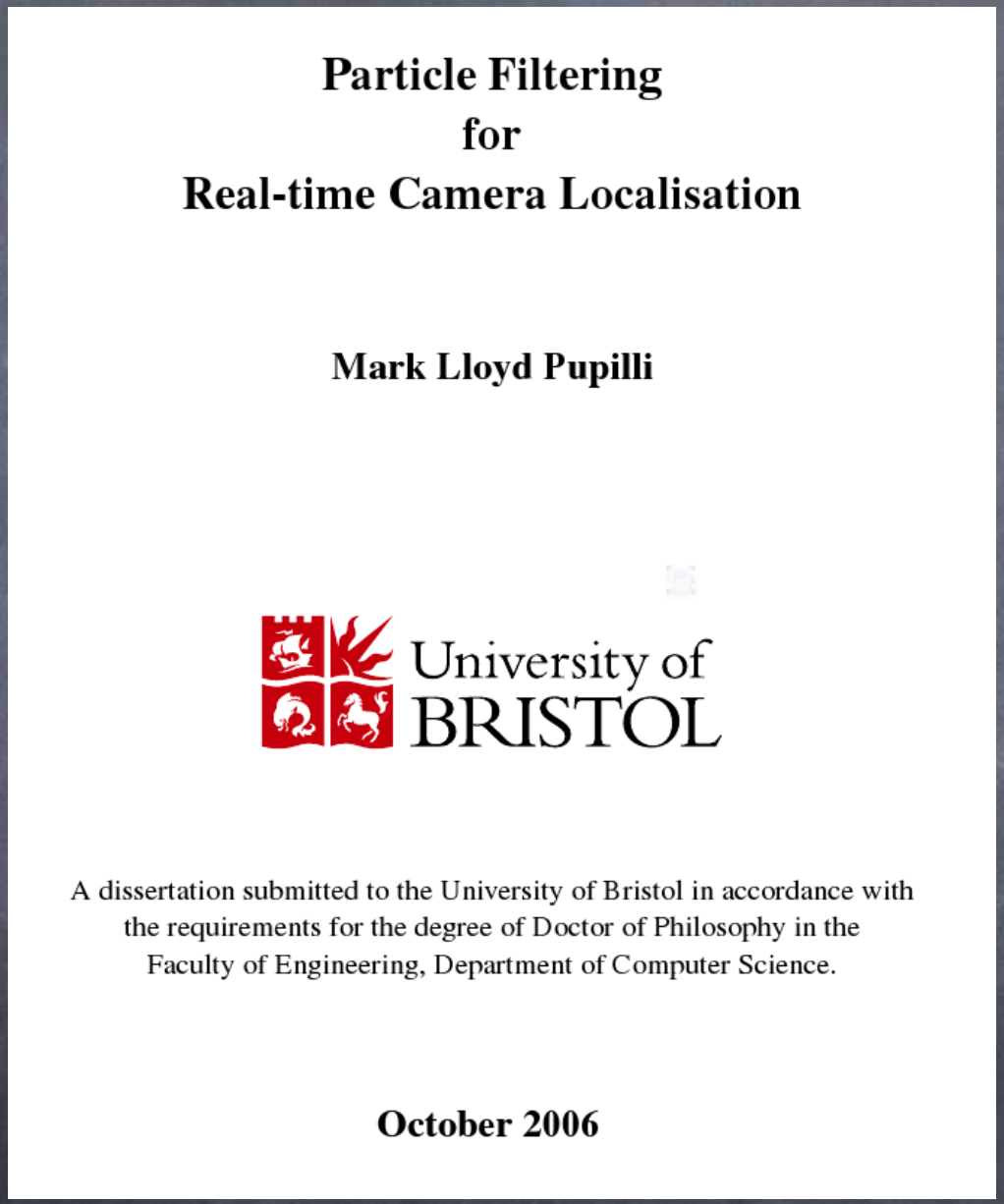
http://catalogue.shu.ac.uk/search~S1/t?The%20Ruby%20way

Mark Pollefeys - Visual 3D modeling of real-world objects and scenes from images



http://video.google.com/videoplay?docid=-1315387152400313941

Mark Pupilli - Particle Filtering for Real-time Camera Localisation



Particle Filtering
for
Real-time Camera Localisation

Mark Lloyd Pupilli

University of BRISTOL

A dissertation submitted to the University of Bristol in accordance with the requirements for the degree of Doctor of Philosophy in the Faculty of Engineering, Department of Computer Science.

October 2006

http://www.cs.bris.ac.uk/home/pupilli/publications/thesis.pdf

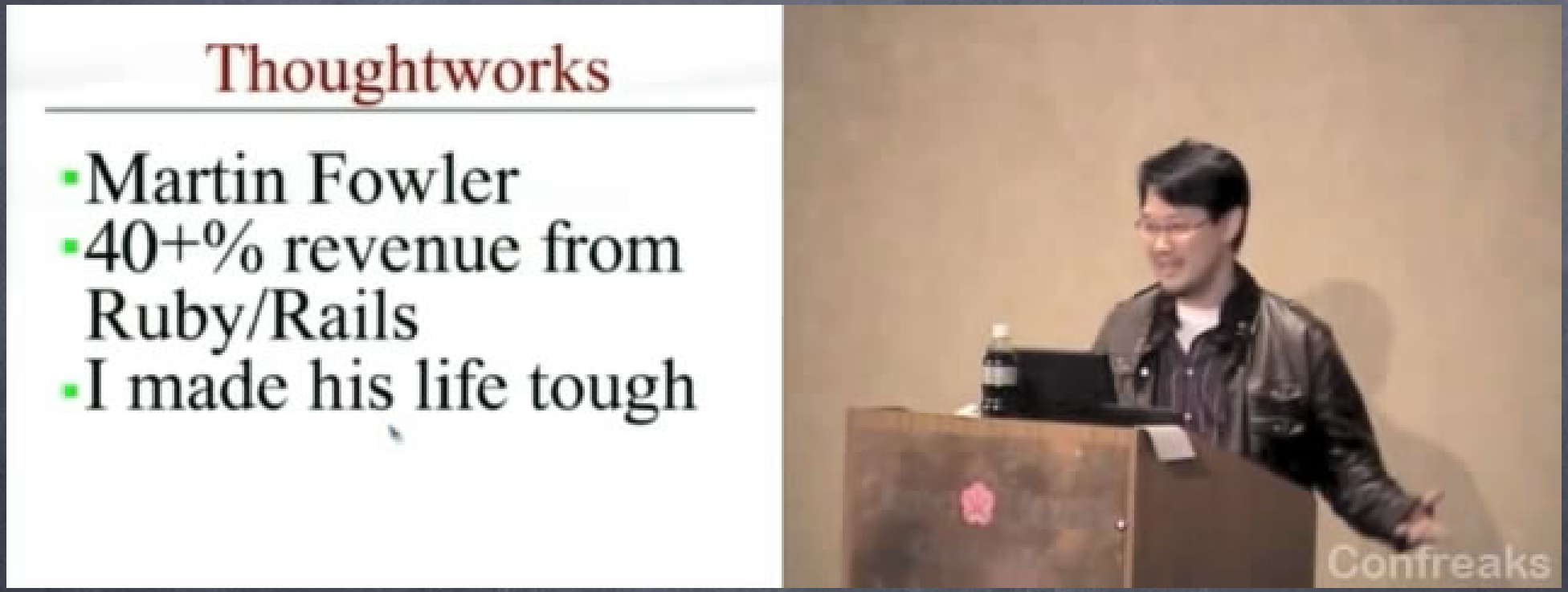
Ben Bleything - Controlling Electronics with Ruby

http://rubyconf2007.confreaks.com/d1t2p1_ruby_and_electronics.html

Patrick Farley - Ruby Internals

http://mtnwestrubyconf2008.confreaks.com/11farley.html

Yukihiro Matsumoto - Does Language Matter?



http://rubyconf2007.confreaks.com/d2t1p8_keynote.html

Willow Garage, Inc.

http://www.willowgarage.com/

Thank You

This presentation was made with LaTeX, TeXPower, InkScape, Ruby, and other free software.

http://vision.eng.shu.ac.uk/jan/demfeb09.pdf