Making Software See

FrOSCon 2011

Michael Maclean
mgdm@php.net
http://mgdm.net

You want to do *what*?

We're all used to creating graphics in software

- ImageMagick, GD, Cairo…

You want to do *what*?

Generally we don't spend a lot of time *processing* graphics

You want to do *what*?

apart from the usual resizing, cropping, rotating, etc

Introducing OpenCV

“The Open Computer Vision Library has > 500 algorithms, documentation and sample code for real time computer vision.”

Introducing OpenCV

Originally developed by Intel, released as open source under the BSD licence

Introducing OpenCV

http://opencv.org

Applications

Object recognition

This includes face detection!

Applications

Gesture tracking

Applications

Structure from motion

Creating 3D models of objects from 2D inputs

Applications

Other cool things I've not yet thought of

Bindings

- OpenCV is written in C/C++, and has a C-based API at its core
- In recent versions, the C++ API has been extended

Bindings

There are bindings for many of the major languages

- Python
- Java
- I'm working on a PHP one
- .NET
- …

Bindings

Some are more complete than others

A Brief Disclaimer

It's not my aim for this talk to give a short course on computer vision

A Brief Disclaimer

Rather, I'm giving a tour of an interesting bit of software

A Brief Disclaimer

(I'm certainly not the best qualified, anyway!)

A Brief Disclaimer

I will not explain the theory behind many things
(as I might not understand it myself)

A Brief Disclaimer

I'm not really a maths geek, so if there are any inaccuracies, point them out

Blatant advertisement

- I'm working on a PHP wrapper for OpenCV

Blatant advertisement

It's on Github

http://github.com/mgdm

You need PHP 5.3+

Blatant advertisement

Patches are welcome

☺

Image capture

OpenCV can handle most of the common raster image formats

JPEG, PNG, TIFF...

Image capture

It can also grab images from a camera

Image capture

You can also grab frames from video files, if you use the FFmpeg support

Image capture

There is also now some official support for the X-Box Kinect sensor, on Linux and Windows

Basic usage

Let's start by loading a test image

Loading and Saving Images

- OpenCV speaks the usual popular raster image formats

In PHP:

```
$image = OpenCV\Image::load("test.jpg");
$image->save("test2.jpg");
```

You now have an Image object

The same image has been saved as test2.jpg

Basics of OpenCV

It treats images as being fundamentally matrices of numbers

Basics of OpenCV

So, many of the same operations you'd perform on a matrix can also be performed on an image

Basics of OpenCV

This includes things like transposition, adding, multiplying, adding scalars, etc

Basics of OpenCV

There is a very large list of these basic operators

Not all of which I understand...

Image Processing

OpenCV (predictably) has many, many functions that do various things to images

These range from the fairly mundane to the very powerful

Smoothing

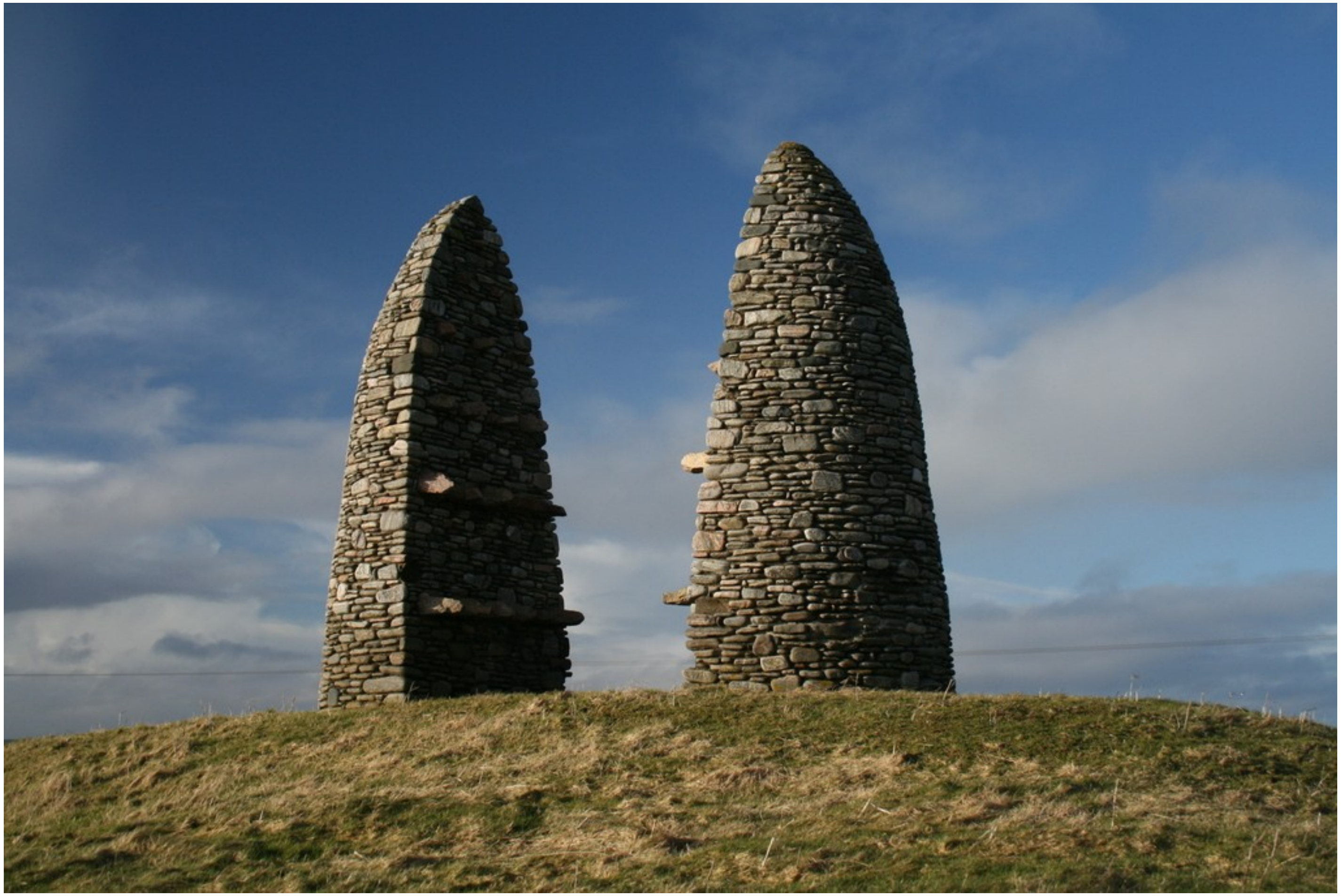
Not very exciting, but a good demonstration – a Gaussian blur

Smoothing

```
$dst = $image->smooth(
      OpenCV\Image::GAUSSIAN, 31, 0, 0, 0);
```

`$dst` will now contain a slightly fuzzy version of the image

The input



The result

Smoothing, continued

Why would you want to do that?

Perhaps to remove noise from an image taken in low light

Smoothing, continued

- Various methods are available, accessible via different parameters to smooth()
  - Median blur
  - Gaussian
  - Bilateral filtering

Image Morphology

(Sounds cool, doesn't it?)

Image Morphology

These are a class of *operators* which *convolve* an image with a *kernel*

The kernel is like a mask, which is scanned across every point in the image

It has an *anchor point*, which represents the pixel being currently transformed

Kernels

0	1	0
1	2	1
0	1	0

For some operations, each number represents the amount of “influence” that pixel has on the output

Dilation

A kernel consisting of a 3x3 square, with an anchor in the centre, is scanned over the image

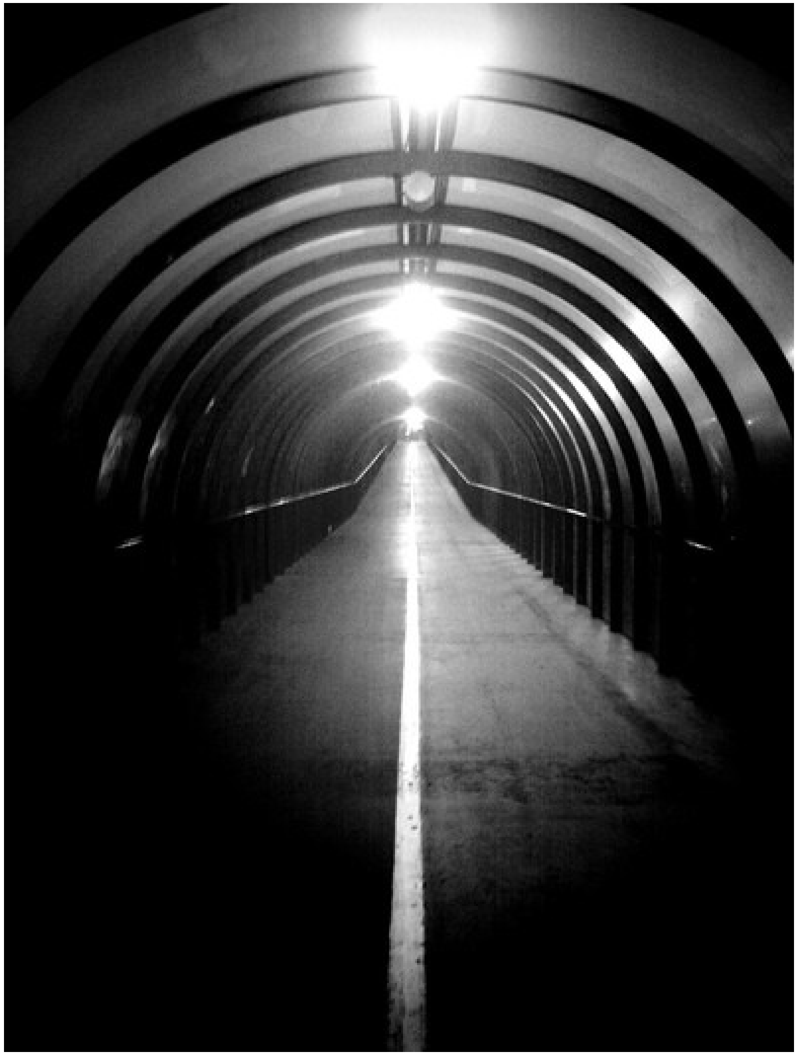
For each point that the anchor goes over, the maximum value covered by the kernel is found

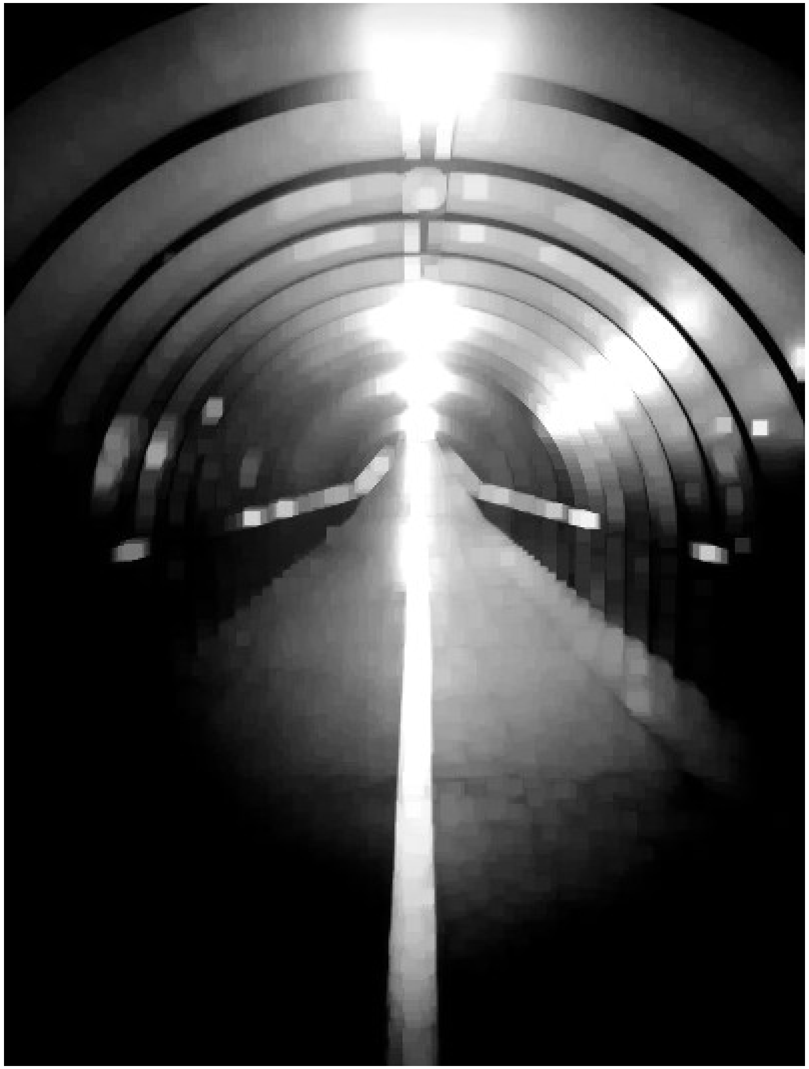
Dilation

This maximum value becomes the value of the pixel at that position in the new image

The upshot of this is, bright areas get larger

Dilation





Dilation

```
import sys, cv
src = cv.LoadImage(sys.argv[1],
cv.CV_LOAD_IMAGE_COLOR)
dest = cv.CloneImage(src)
cv.Dilate(src, dest, None, 3)
cv.SaveImage(sys.argv[2], dest)
```

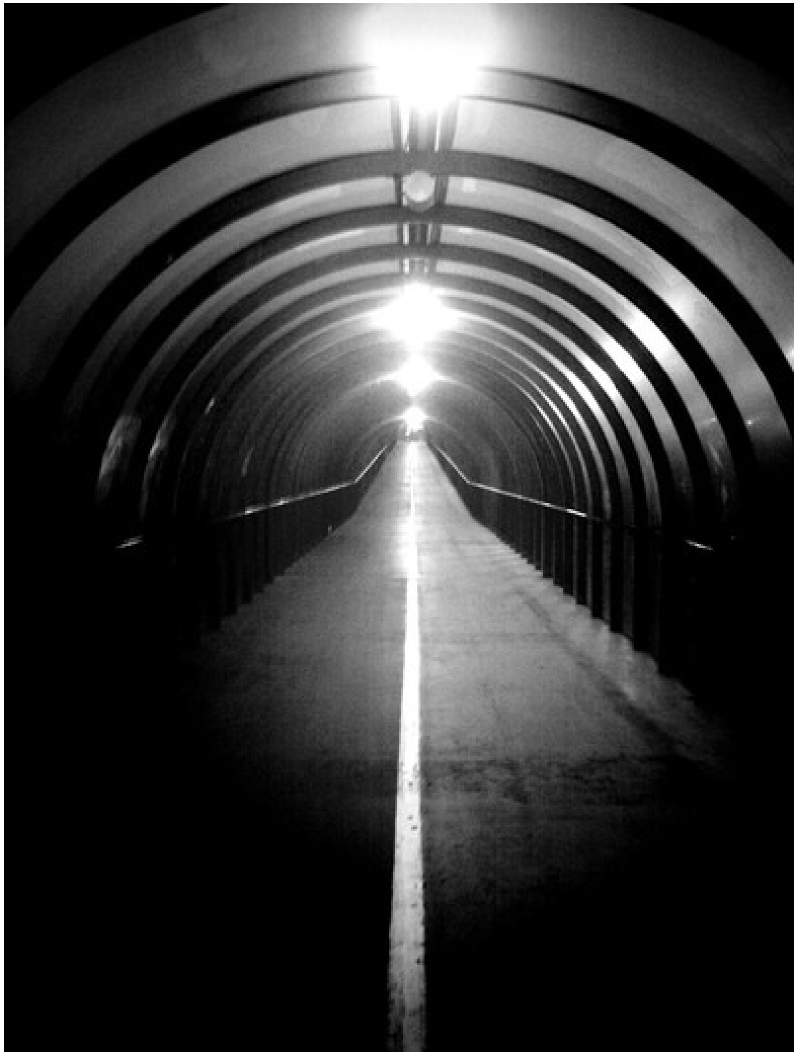
Erosion

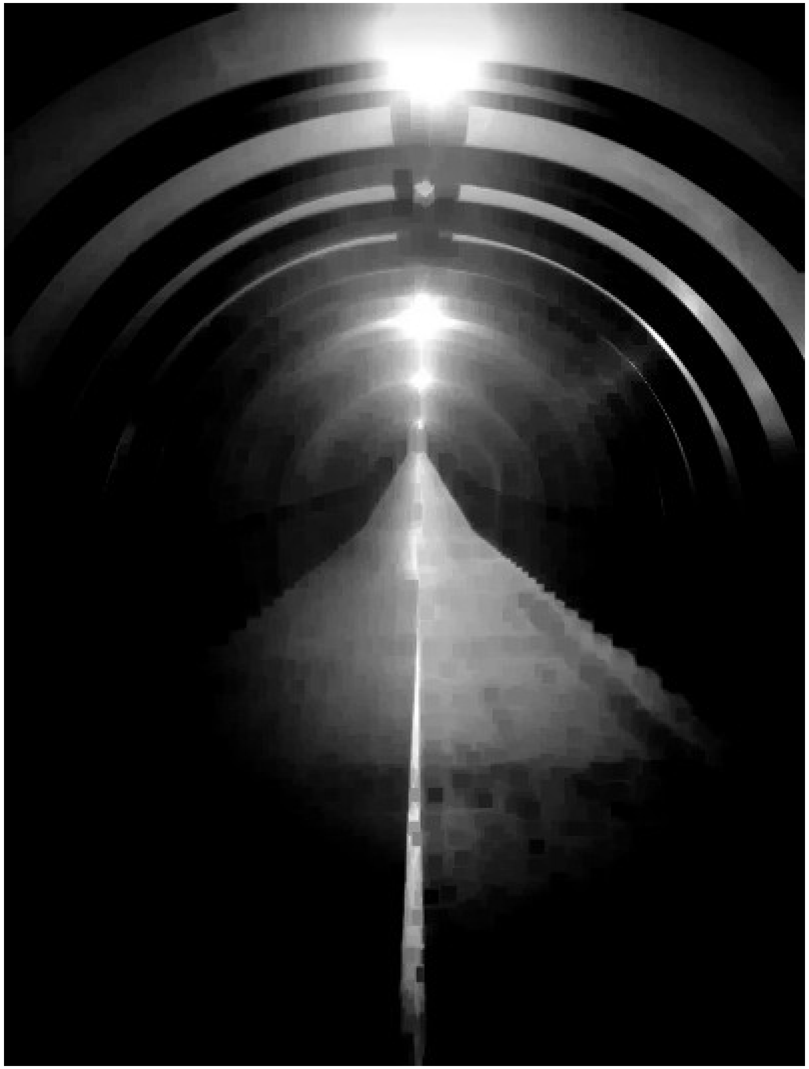
This is fundamentally the opposite of dilation

Instead of the maximum value under the kernel,
we look for the minimum

Bright areas get smaller

Erosion





What's the point?

Bright areas might be part of the same object, but split into several parts in the image

(Shadows, obstructions)

What's the point?

These operations will cause these areas to join up, making them easier to identify

Edge detection

OpenCV features several algorithms for edge detection

You might well have seen these before in GIMP or Photoshop

You can use them to get outlines of objects

Sobel

The Sobel operator is a way to get a derivative of an image

(Maths geeks may point out this is technically incorrect – just run with it for now)

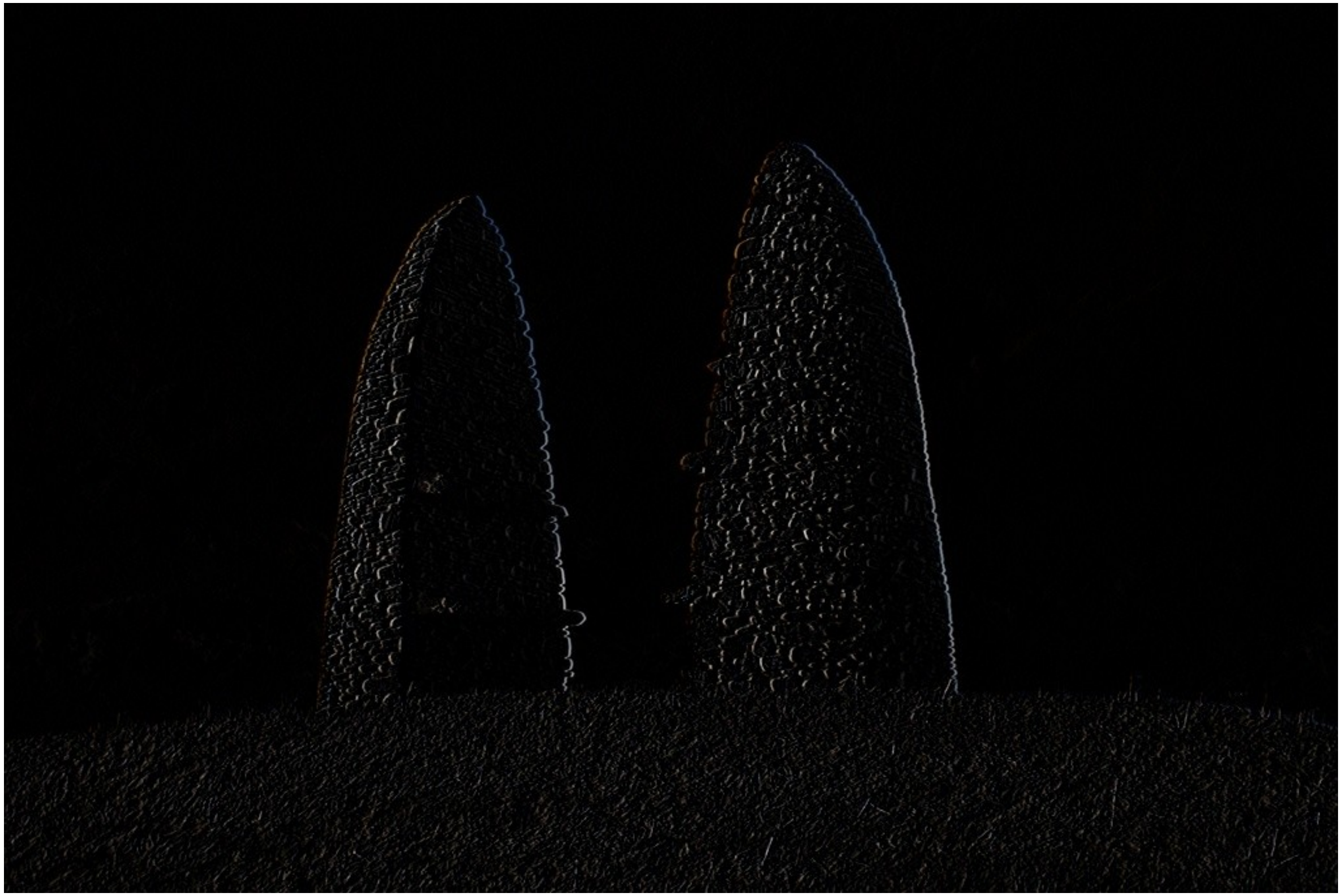
Sobel

```
<?php
use OpenCV\Image as Image;
use OpenCV\Histogram as Histogram;

$i = Image::load("test.jpg",
Image::LOAD_IMAGE_COLOR);
$dst = $i->sobel(1, 0, 1);
$dst->save("test_sobel.jpg");
```

Sobel

You end up with an image which has bright areas where there is lots of contrast

The result

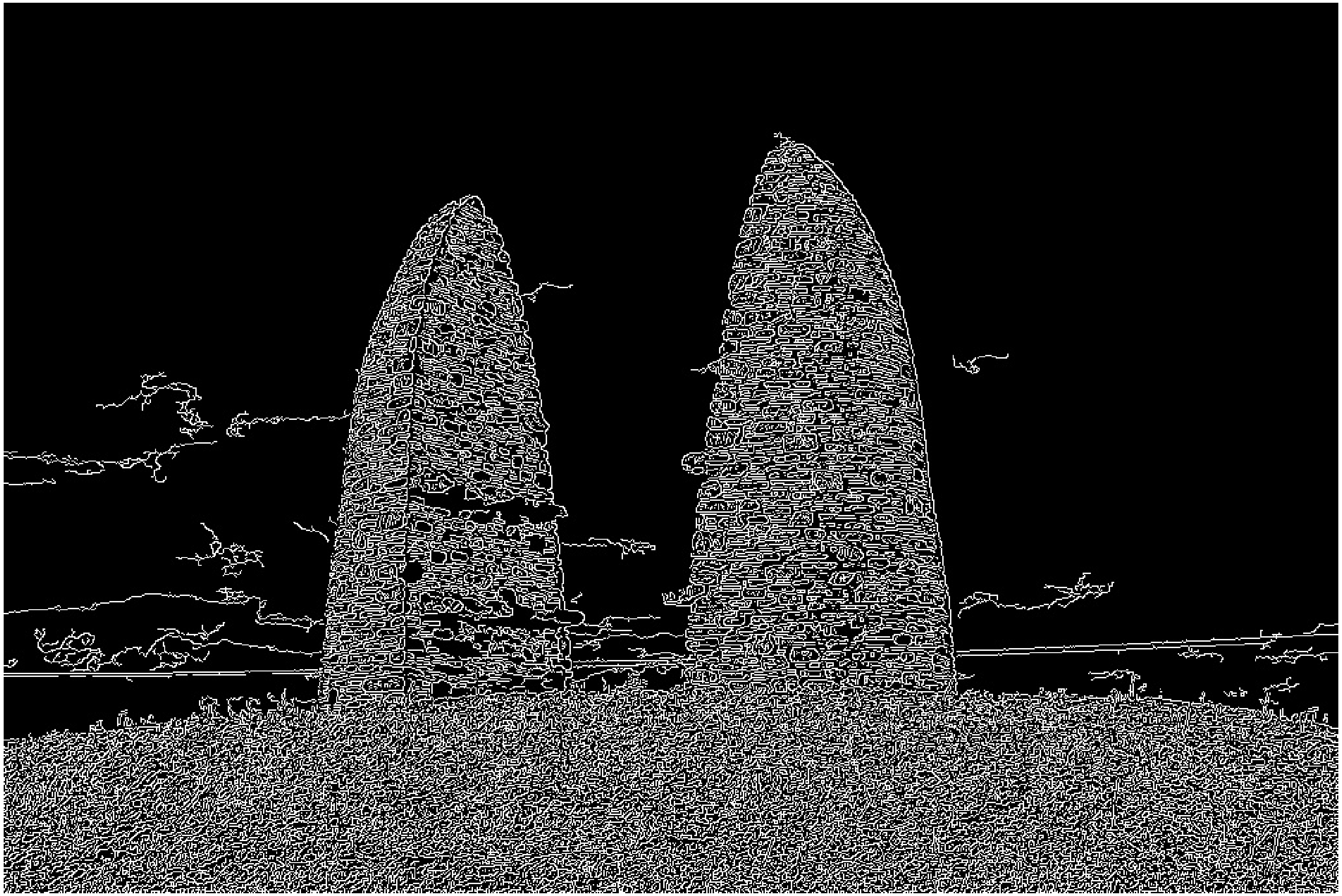


Canny

A more complex algorithm, which takes derivatives in *x* and *y*

Some more processing results in a far clearer line

The result



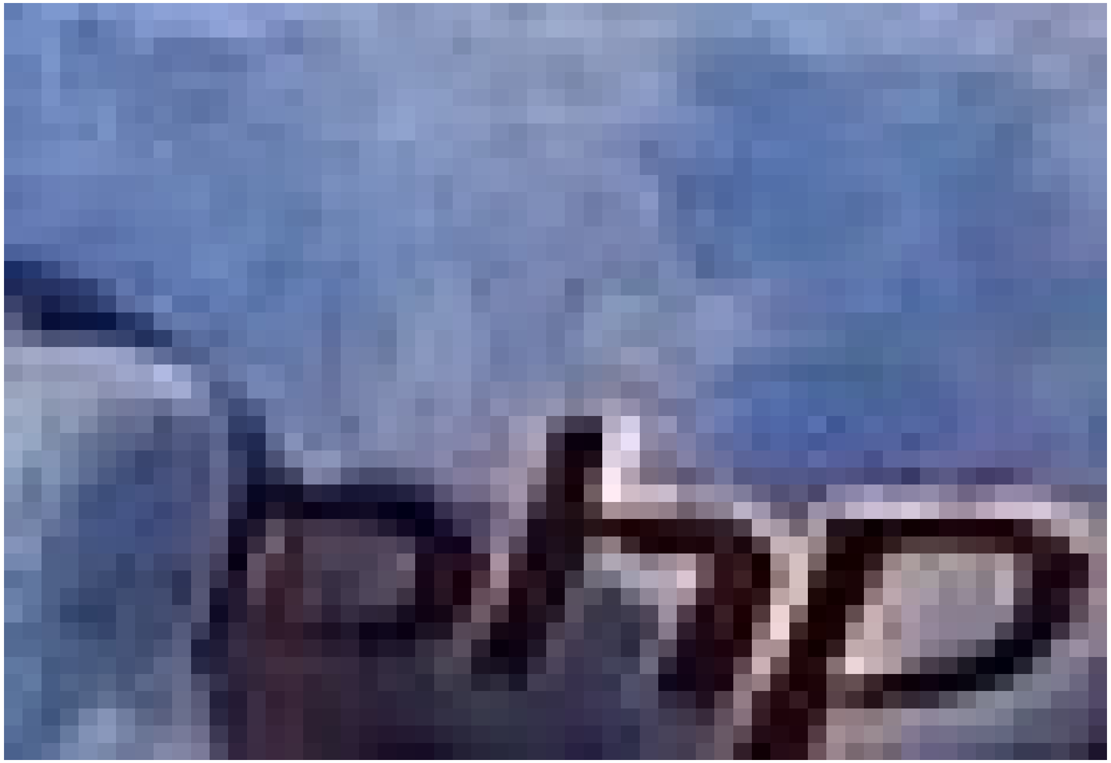
And now for something
~~completely different~~
more interesting

Template matching

You can use OpenCV to try and locate an image within another image

You'd usually do this with a small image of an object, and a larger search image

My test image

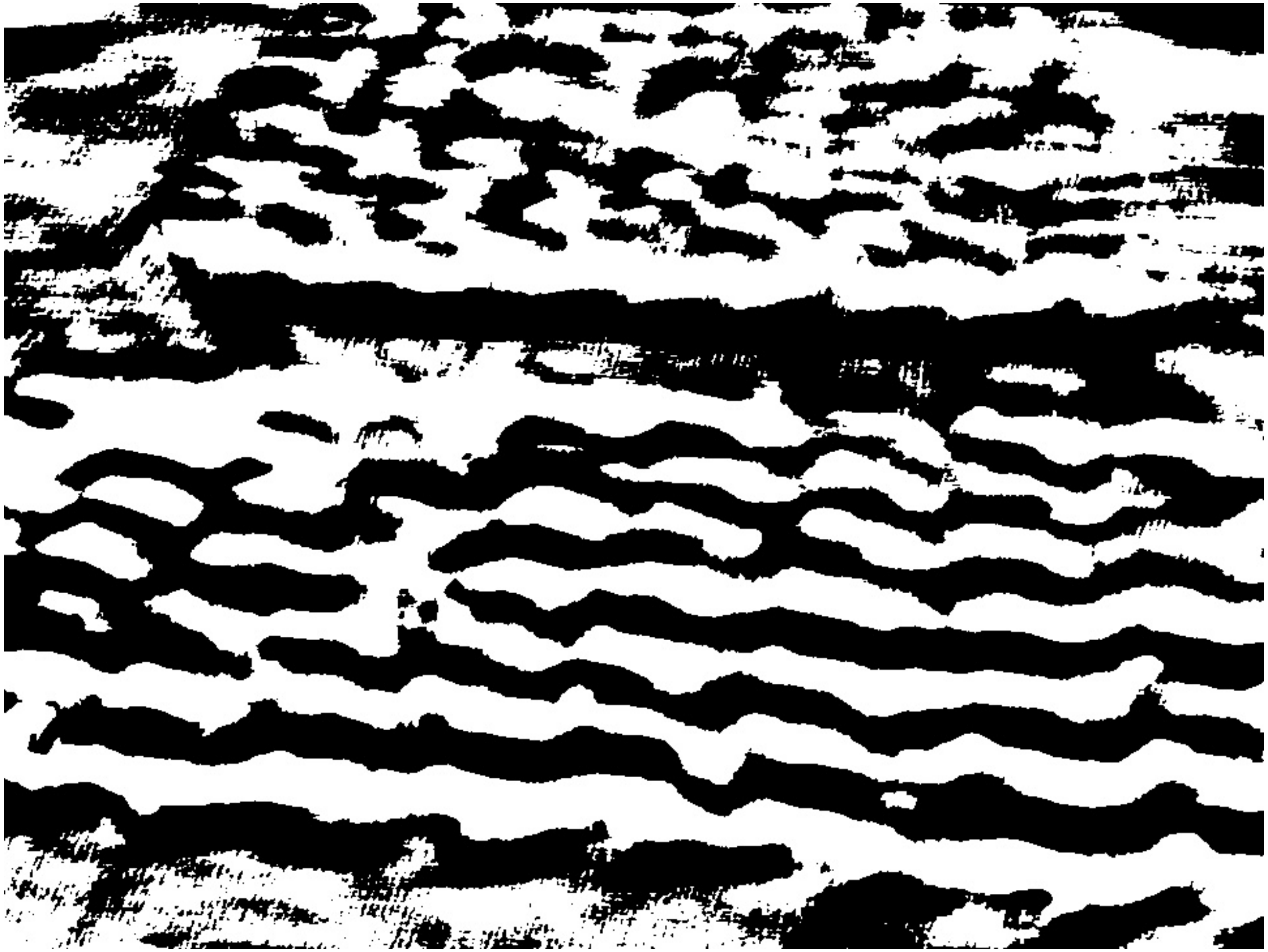


(It is only 59 pixels wide, excuse the fuzziness)

My other image

Thanks to Michaelangelo van Dam for the Creative Commons picture
http://www.flickr.com/photos/dragonbe/3411273755/

The result



The result

It's not perfect, but it's a start

The result

The reason it matches most of the ElePHPants is because the small image is, well, small, and has been re-saved as a JPEG

The result

Thus, the pixels it is searching with are no longer identical to the originals

Histograms

Histograms are a way of fitting values into slots, or *bins*, as OpenCV calls them

They are fundamentally the same as an array, with a count of values, and can be graphed as a bar chart

I'm not going to do anything quite that dull

Histograms

OpenCV uses histograms to record a signature of features in an image

Histograms

For example, you can use them to record the colours in a section of an image

(Stay awake at the back)

Back projection

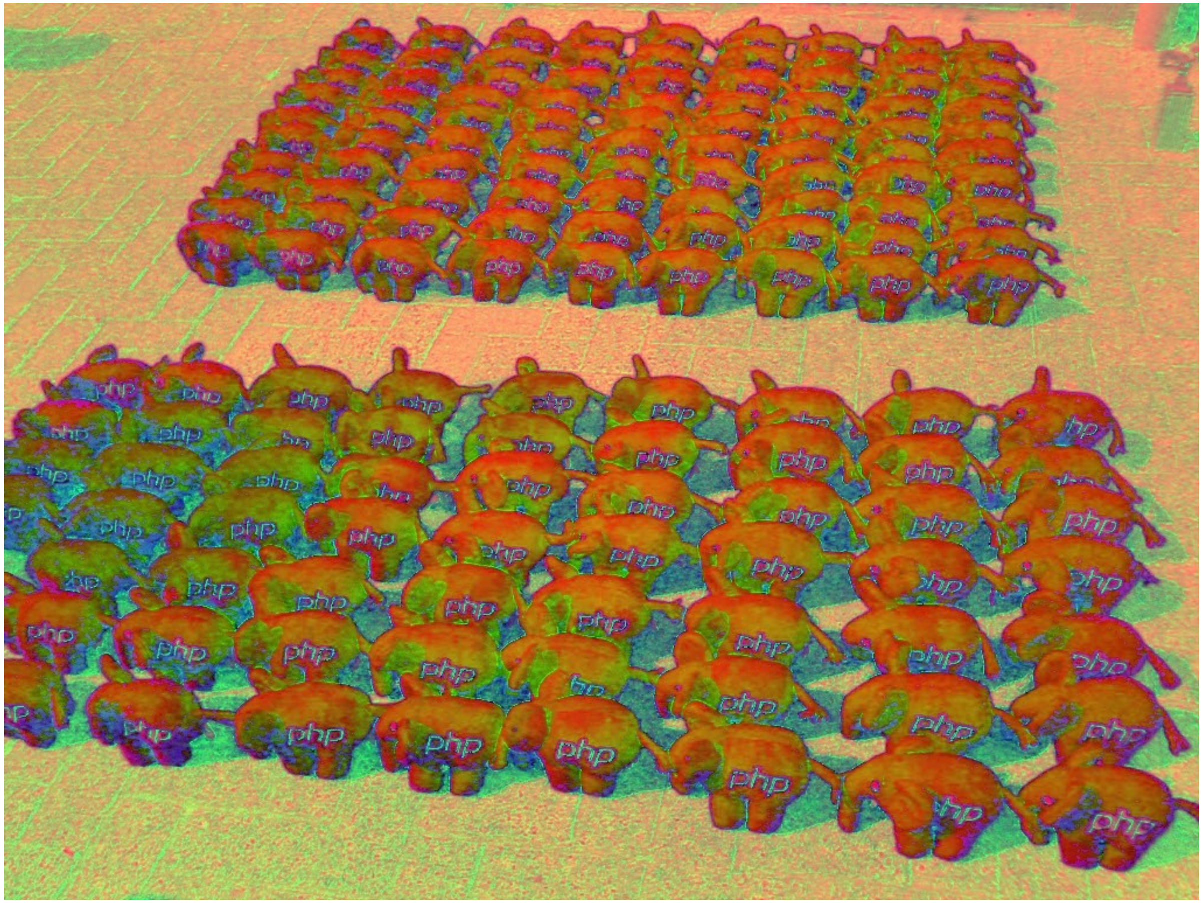
If I convert this image to HSV format, I can get a version of the image which encodes the hue as one channel

Back projection

Then I can convert this into a histogram, to get a hash of sorts of the proportion of colours in the image

Then, OpenCV will take this histogram and look for it in another image

An aside...

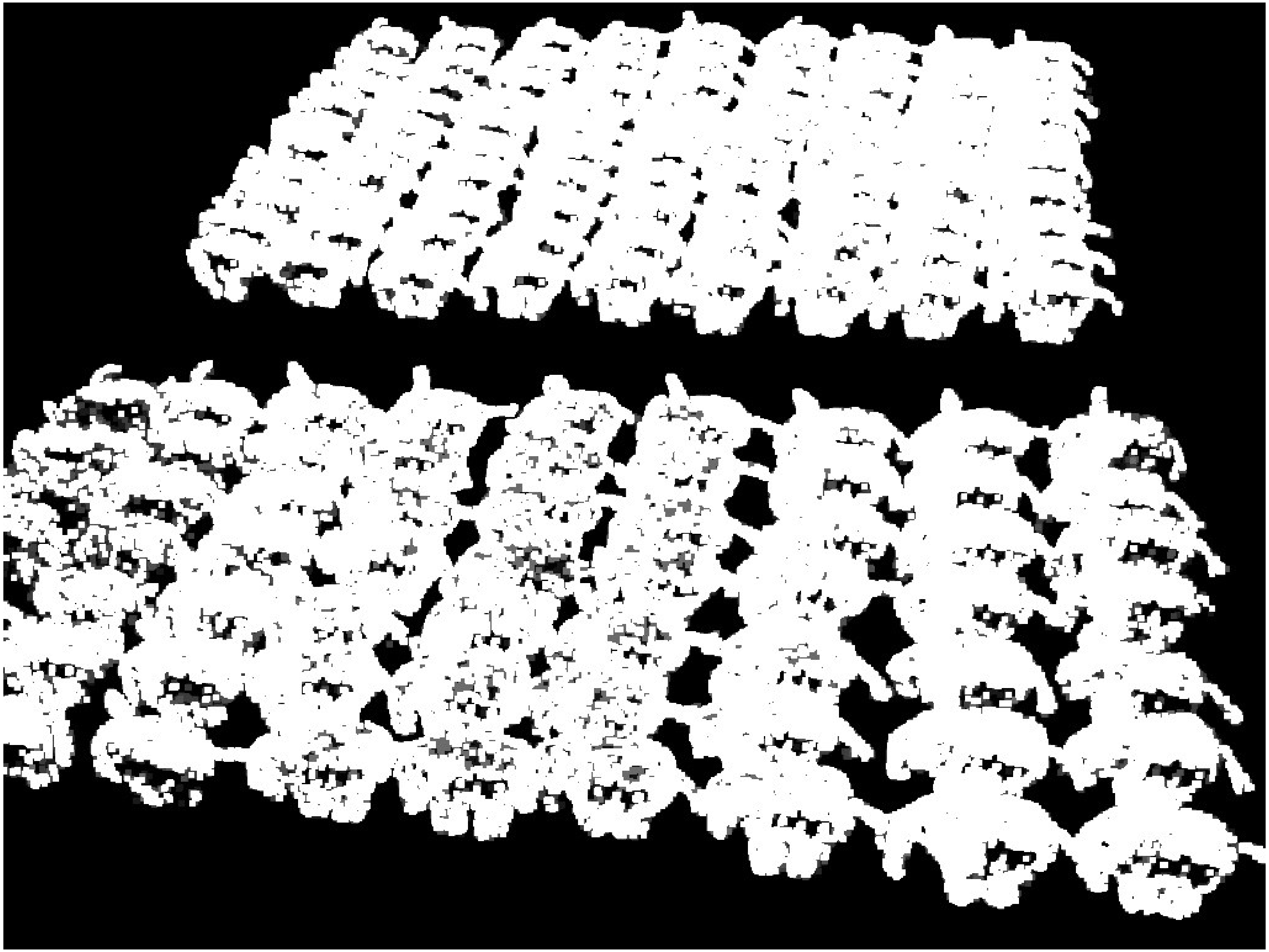


This is what an image looks like when it's converted to HSV and saved as if it was still RGB

The code

```php
<?php
use OpenCV\Image as Image;
use OpenCV\Histogram as Histogram;

echo "Loading sample\n";
$i = Image::load("elephpant_sample.jpg",
   Image::LOAD_IMAGE_COLOR);
$hsv = $i->convertColor(Image::RGB2HSV);
$planes = $hsv->split();
$hist = new Histogram(1, 32,
   Histogram::TYPE_ARRAY);
$hist ->calc($planes[0]);

echo "Loading main image\n";
$i2 = Image::load("dragonbe_elephpants.jpg",
   Image::LOAD_IMAGE_COLOR);
$hsv2 = $i2->convertColor(Image::RGB2HSV);
$planes2 = $hsv2->split();
$result = $planes2[0]->backProject($hist);
$result ->save("bp_output.jpg");
```

The result



The result

As you can see, the result is a binary image showing the places where the histogram matched the colours in the image

The result

You could probably get a better match with a better sample

The result

You can then go on and do other things with the morphology operators, to find the extents of each match

The result

Or you can use that image as a mask for other operations

Other uses for histograms

In combination with the edge detection stuff from earlier, you can make a histogram of the edge gradients

Other uses for histograms

This means that you can then try and match particular shapes using a histogram

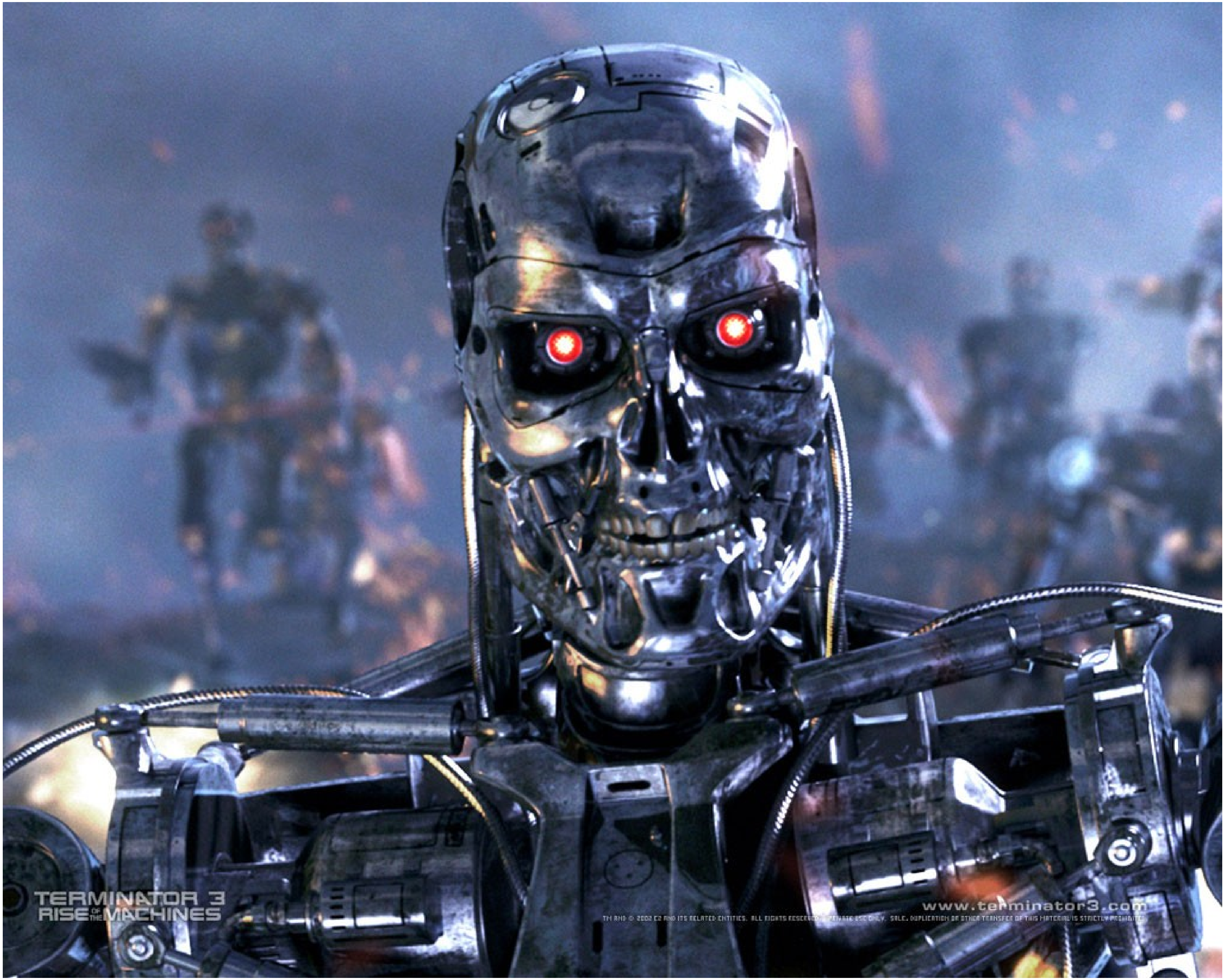
This is used by some for gesture recognition

Other uses for histograms

You can take a histogram of two images and compare them to identify identical images, even at different sizes

Machine learning

In addition to the image processing functions, OpenCV includes some machine learning capability


TERMINATOR 3
RISE OF THE MACHINES
www.terminator3.com
TM AND © 2002 C2 AND ITS RELATED ENTITIES. ALL RIGHTS RESERVED. PRIVATE USE ONLY. SALE, DUPLICATION OR OTHER TRANSFER OF THIS MATERIAL IS STRICTLY PROHIBITED.

Machine learning

The algorithms are general, and not really specific to computer vision

Machine learning

Generally, the algorithms are trained using a large data set, and then tested against another

Machine learning

There are two main ways of implementing this

- *Supervised learning*, in which the input data is "labelled"
- *Unsupervised learning*, where the data has no labels

Supervised learning

In this method, the algorithm knows that the feature it is being trained on is either present or not present

Supervised learning

It can then learn that certain characteristics are distinctive among those where the feature is present

Supervised learning

For example, you can have a dataset of 10,000 images, in which 8,000 contain faces and 2,000 do not

Unsupervised learning

These algorithms are referred to as *clustering algorithms*

They are used to match up similar images, without knowledge of what is within

Training

OpenCV comes with tools for training the various algorithms it supplies

I'm not going to explain them here

Haar classifier

This is a fairly specialized implementation of a learning algorithm

Haar classifier

It is used to detect “mostly rigid” objects

Like faces!

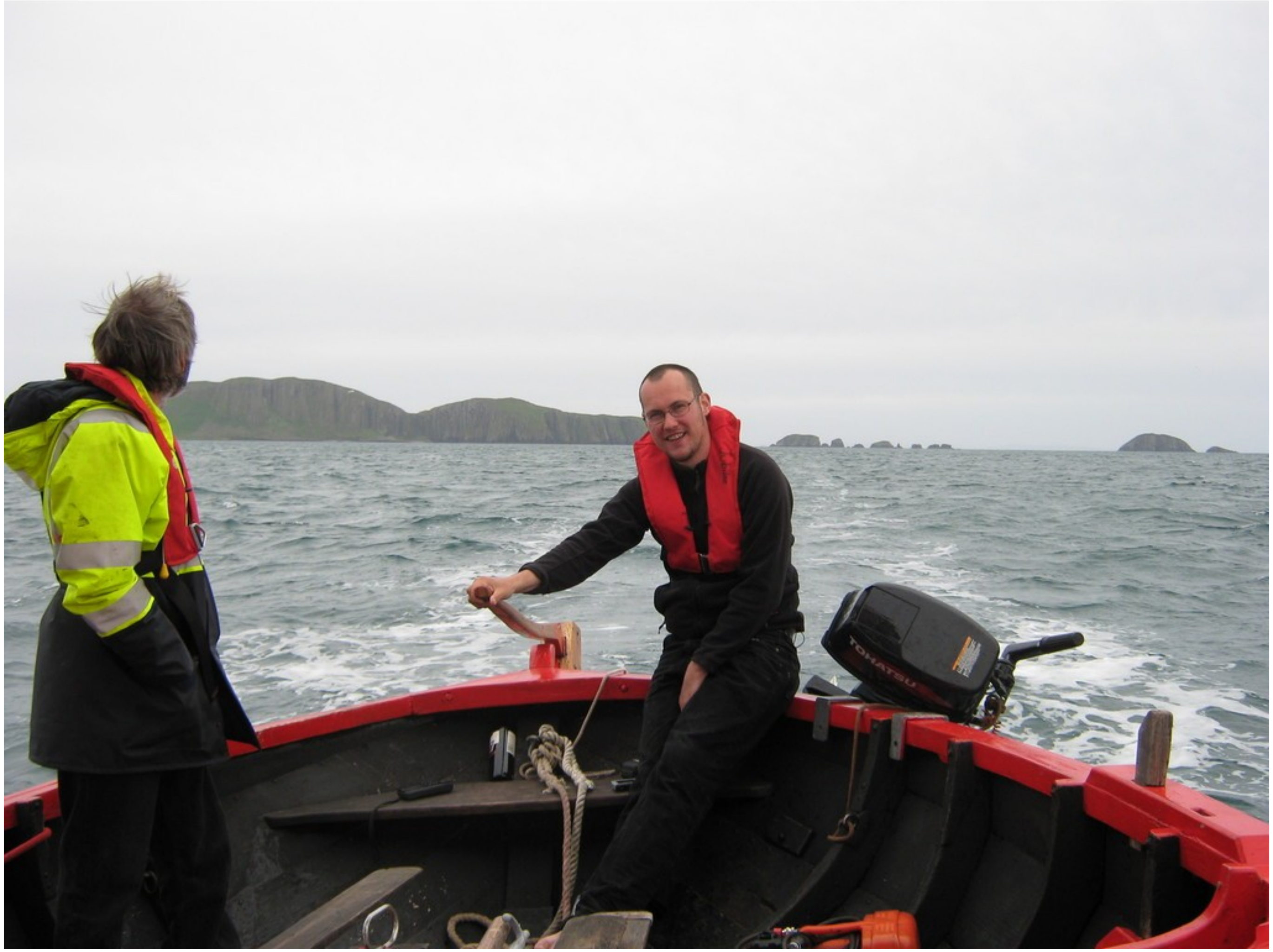
Haar classifier

OpenCV comes with a pre-trained Haar classifier capable of recognising faces within an image

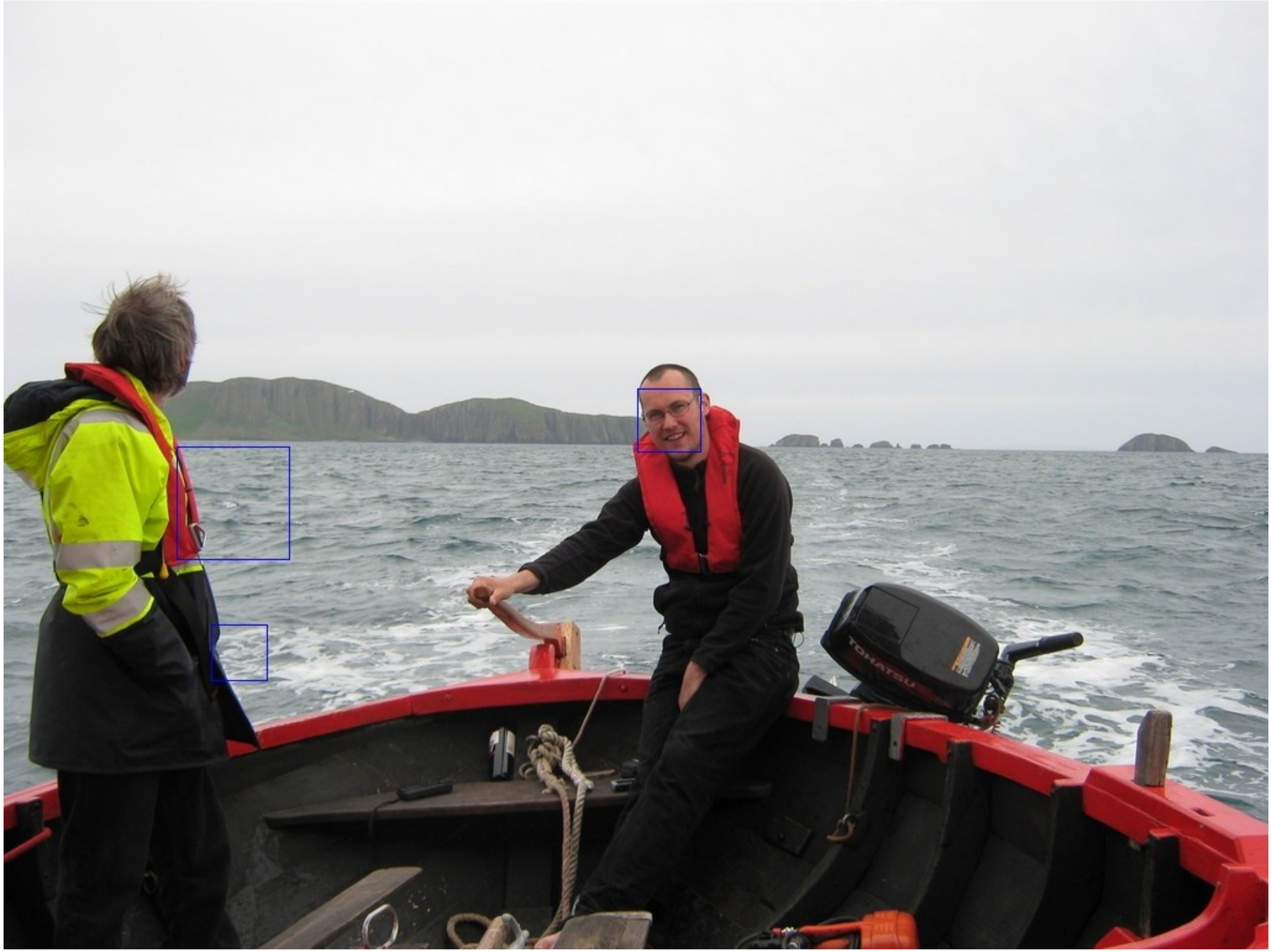
Haar classifier

```php
<?php
use OpenCV\Image as Image;
use OpenCV\Histogram as Histogram;

$i = Image::load("sailing.jpg",
  Image::LOAD_IMAGE_COLOR);

$result = $i->haarDetectObjects("
  haarcascade_frontalface_default.xml");
foreach ($result as $r) {
  $i->rectangle($r['x'], $r['y'],
     $r['width'], $r['height']);
}
$i ->save("sailing_detected.jpg");
```

The input image



The result



Inpainting

Repairing damage to images – for example

- stripping watermarks
- filling in texture after the image has been rotated

To do this properly, you have to create a mask

Contour detection

OpenCV can use its edge detection algorithms to find the contours of an image

These are stored in a sequence, which can be iterated over

Feature detection

OpenCV also includes algorithms to find “interesting” points in an image

Feature detection

Algorithms for this include SURF

SURF feature detection

```
CvSeq* objectKeypoints = 0, *objectDescriptors = 0;

CvSeq* imageKeypoints = 0, *imageDescriptors = 0;

CvSURFParams params = cvSURFParams(500, 1);

IplImage* object_color =
cvCreateImage(cvGetSize(object), 8, 3);

cvCvtColor( object, object_color, CV_GRAY2BGR);

cvExtractSURF( object, 0, &objectKeypoints,
&objectDescriptors, storage, params );
```

Feature detection

This is quite interesting as it can be used as the beginning of object detection algorithms

Feature detection

It's more interesting when used to compare sequential frames of video

Doing this can give you what is known as "optical flow"

Optical flow

The optical flow gives you a list of vectors describing the movement of points in the image from one frame to the next

Optical flow

There are two types of output

- Dense – find for every pixel
- Sparse – find for only some points

Optical flow demo

Thanks for listening

Get in touch

mgdm@php.net

http://mgdm.net

@mgdm

Image capture

There is also now some official support for the X-Box Kinect sensor, on Linux and Windows

Thanks for listening

Get in touch

mgdm@php.net

http://mgdm.net

@mgdm