Torch vision: a machine vision package for Torch

Sébastien Marcel

marcel@idiap.ch

IDIAP Research Institute
Martigny, Switzerland

http://www.idiap.ch

Outline

- ▷ Introduction
- Motivation
- Reminder: Main concepts of Torch
- Main concepts of Torch vision
- Basic features
- Advanced features
- Demonstrations

Introduction

- a common software platform is important
  - for research purposes,
  - to warrant the transfert of knowledge between "generations" of researchers (in addition to publications),
  - to create software deliverables (required in some projects),
  - to facilitate the development of demonstration systems,
  - to move "easily" software developed at IDIAP toward the industry (transfer of technology).
- in computer vision many libraries are available:
  - VXL, University of Manchester
  - RAVL, University of Surrey
  - Intel OpenCV, Intel research
  - LTI-Lib, Gandalf, ...

Introduction

- on one side:
  - computer vision libraries with many algorithms
  - making use of all the advantages of the modern C++ (multiple inheritance, templates, operator overloading, ...)
  - sometimes doing also linear algebra, numerical algorithms, classification or probability modelling but not under a unified and modular framework such as Torch (http://www.torch.ch)
- on the other side:
  - Torch is a machine learning library developped at IDIAP, widely used in the ML community and in all "groups" at IDIAP
  - Torch coding style and philosophy is: simple C++, single inheritance, no templates, no operator overloading, no private, no cin/cout and no external libraries with "millions" of functions you will never use

Motivation

- our first objective was to build a common software platform at IDIAP for researchers working on face/gesture recognition and using Torch
- then other people were more and more interested in features developped (image and video support) they could used directly with Torch
- $\Rightarrow$ Torch vision is a machine vision package for Torch (Release 3)

  Machine Learning + Computer Vision = Machine Vision

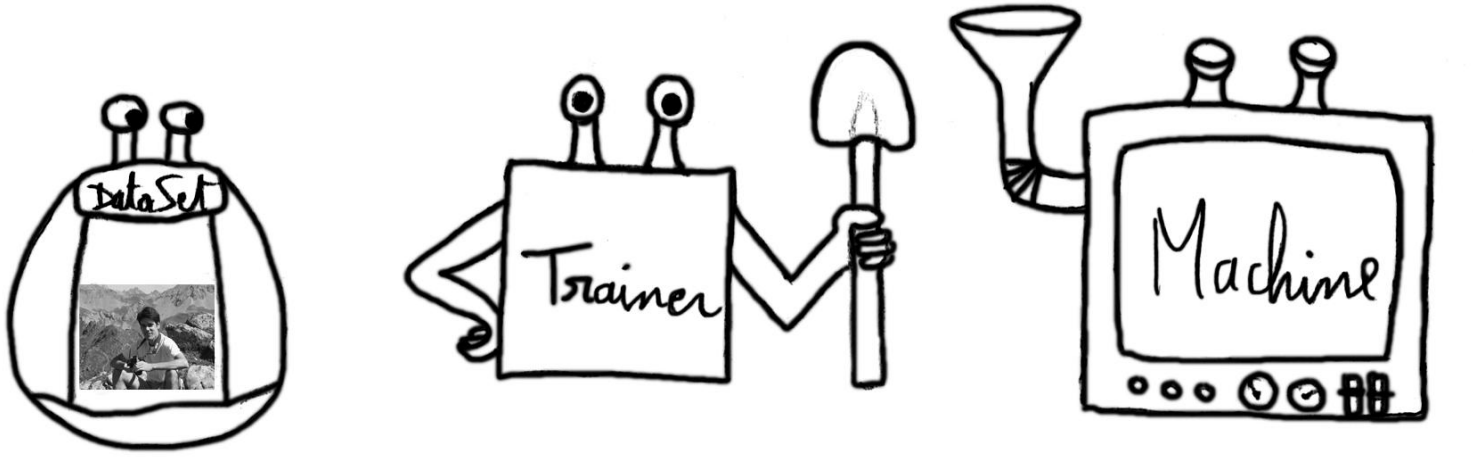
  - fully integrated with Torch (Image $\rightarrow$ MLP or GMM)
  - covers most of IDIAP needs in computer vision and image processing
  - fully documented
  - simple to use and to program
  - plenty of ready-to-use examples
  - internal support

Outline

- × Introduction
- × Motivation
- ▷ Reminder: Main concepts of Torch
- • Main concepts of Torch vision
- • Basic features
- • Advanced features
- • Demonstrations

Reminder: Main concepts of Torch

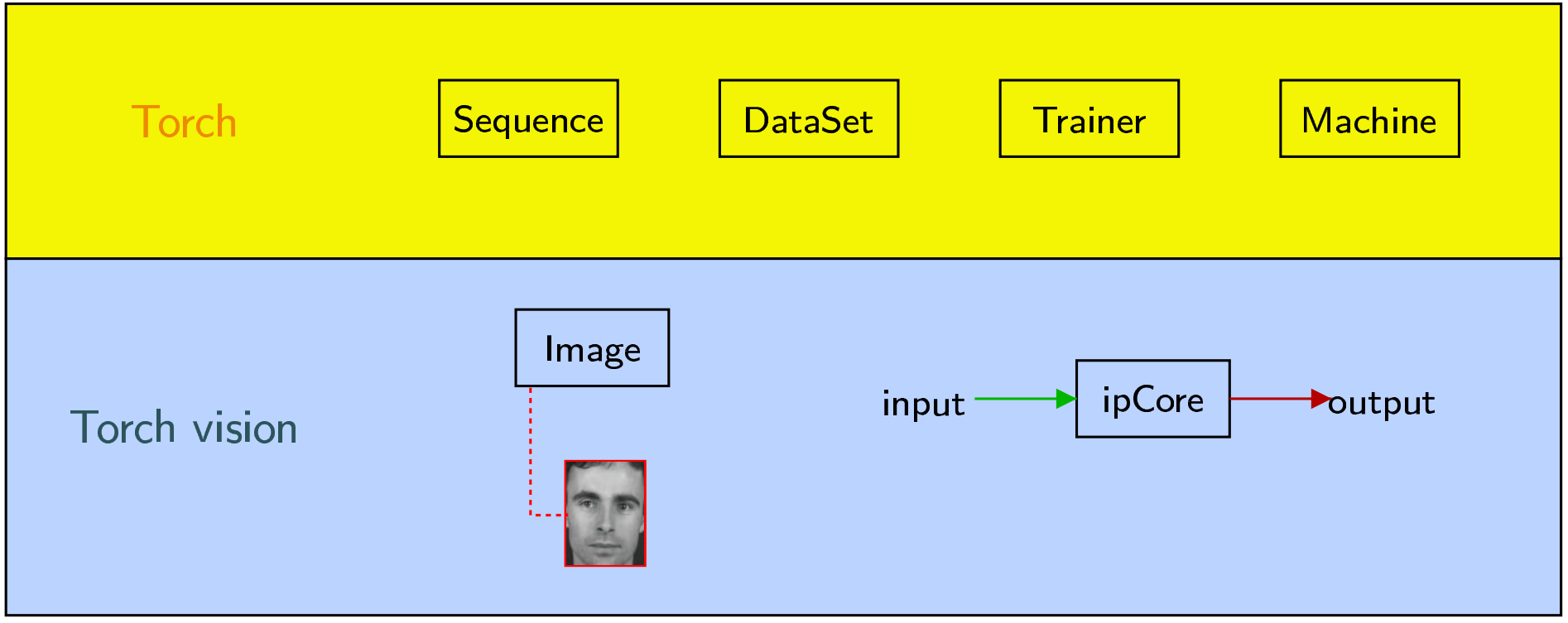
- DataSet: handles data as a set of sequences (of frame features),
- Trainer: trains a Machine using a DataSet,
- Machine: given an input returns an output (classification, likelihood, ...),



© Nicolas Gilardi

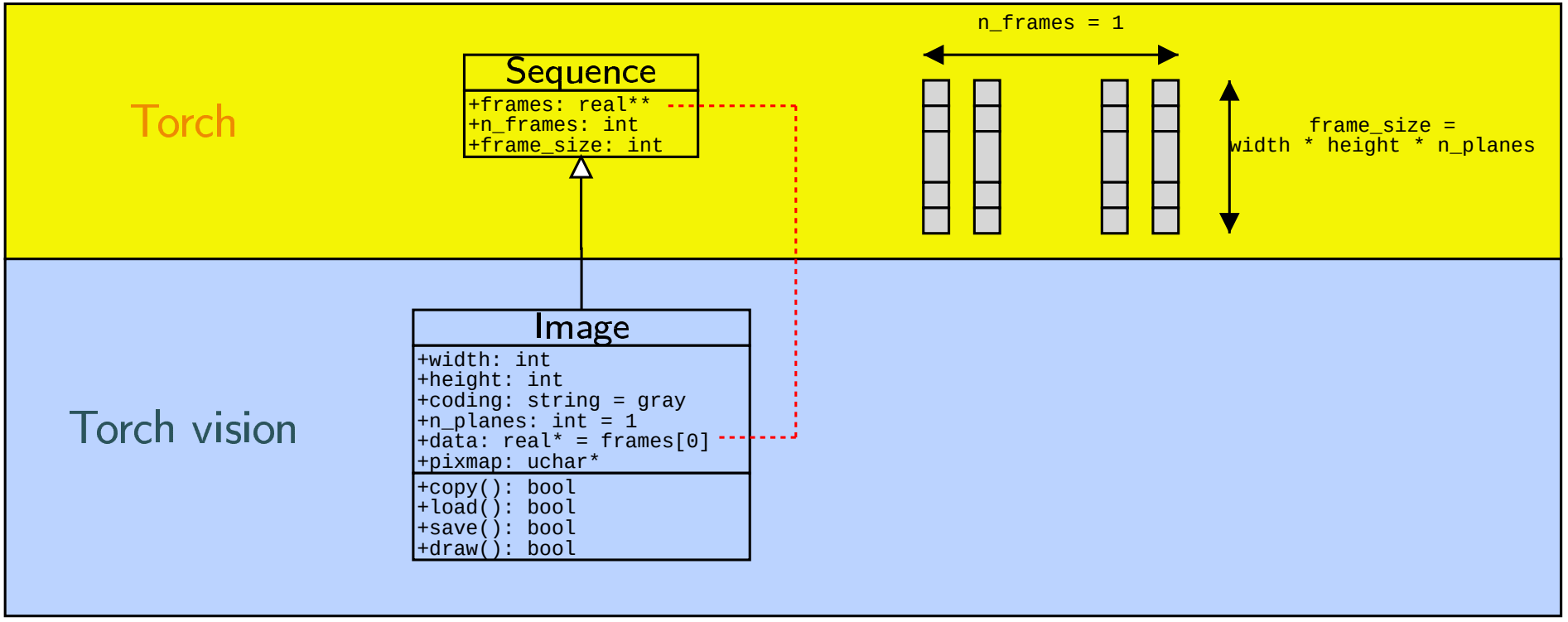
Main concepts of Torch vision

- Image: handles an image (grayscale or color)
- ipCore: given an input Sequence (features or images) returns an output Sequence (features or images)



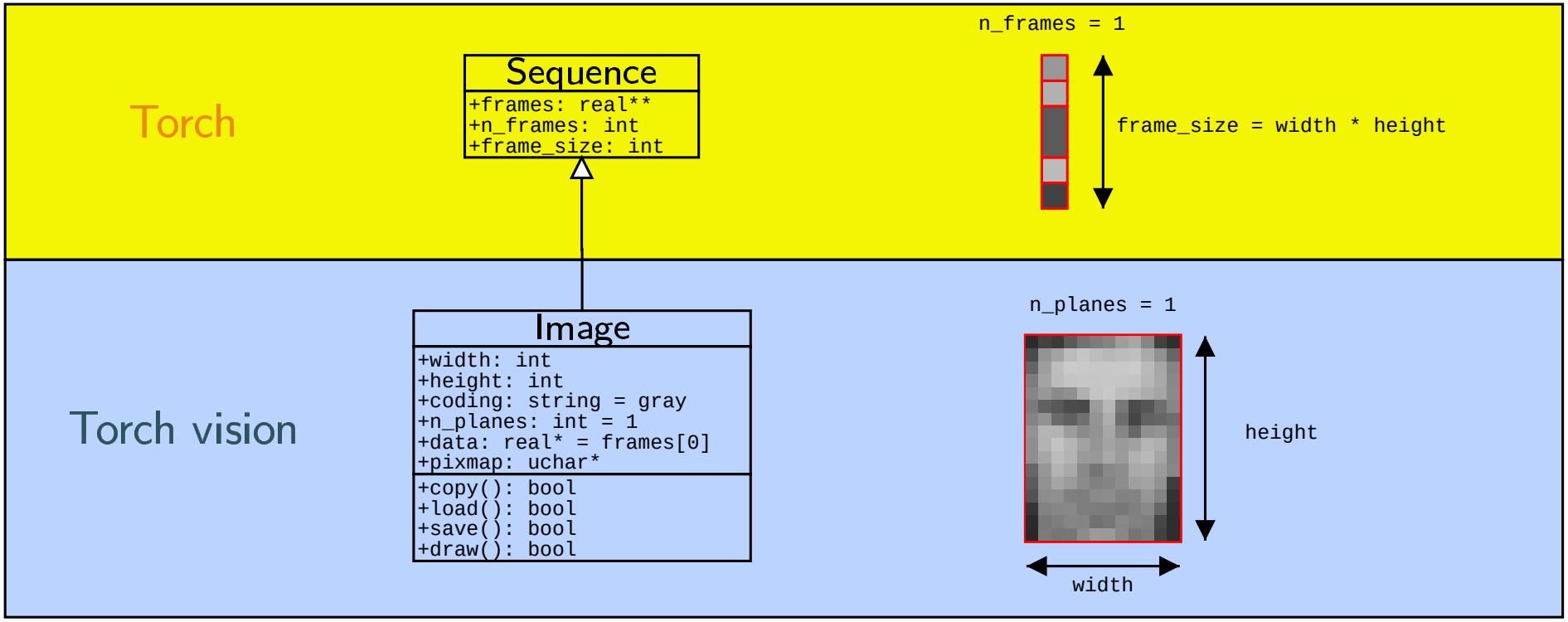
Main concepts of Torch vision: Image

- an image is a sequence of 1 frame
- the size of this frame is width $\times$ height $\times$ n_planes



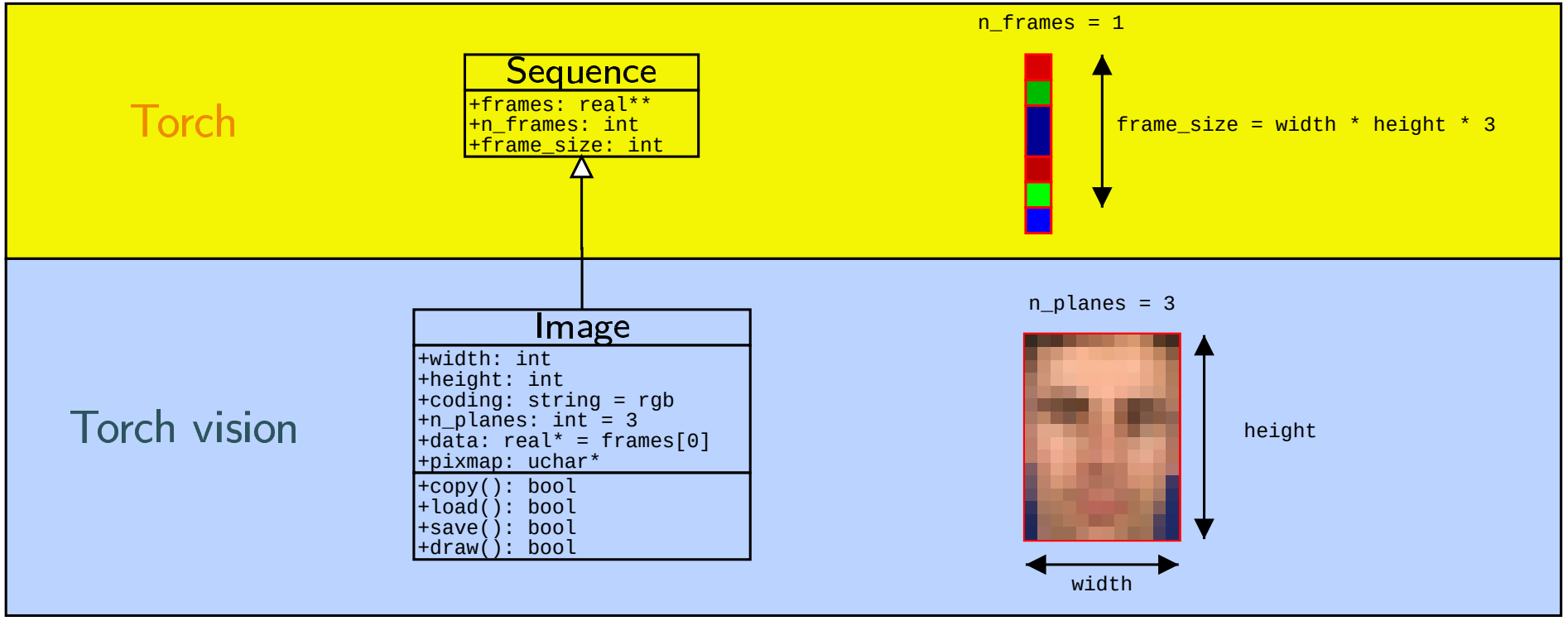
Main concepts of Torch vision: Image

- example: grayscale image



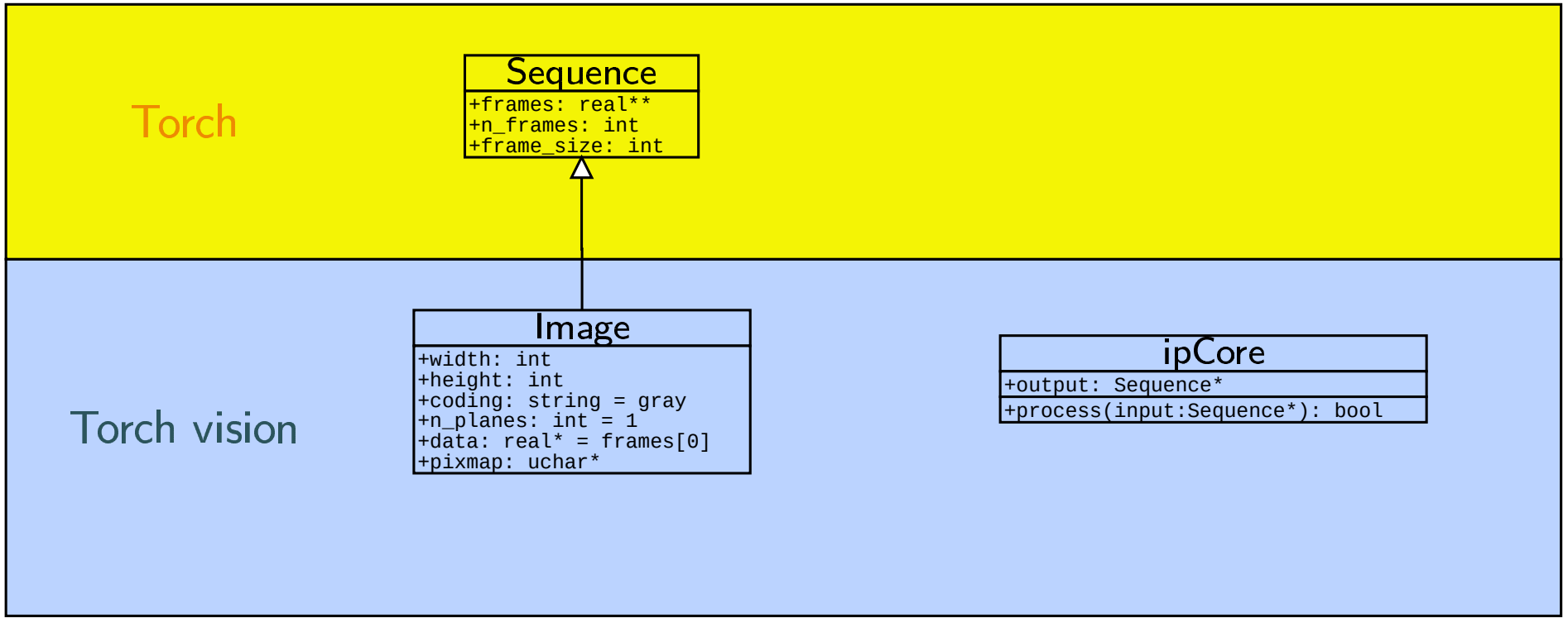
Main concepts of Torch vision: Image

- example: color image (RGB)



Main concepts of Torch vision: ipCore

- input: a Sequence or an Image
- output: a Sequence (sequence of frame features or several images)



Outline

- × Introduction
- × Motivation
- × Reminder: Main concepts of Torch
- × Main concepts of Torch vision
- ▷ Basic features
  - Image
  - ipCore
- Advanced features
- Demonstrations

Basic features: handle an image

- grayscale and color (RGB) image:

```
Image *imagegray = new ImageGray(200, 100);
Image *imagergb = new ImageRgb(200, 100);
```

- image attributes:

```
print("width = %d\n", image->width);
print("height = %d\n", image->height);
print("format = %s\n", image->coding);
print("n_planes = %d\n", image->n_planes);
```

- image data:

```
// direct access pointer
imagegray->pixmap[i] = gray_value;
// classical pixel address
imagegray->get(y,x)[0] = gray_value;
imagecolor->get(y,x)[0] = red_value;
imagecolor->get(y,x)[1] = green_value;
imagecolor->get(y,x)[2] = blue_value;
```

Basic features: image I/O

- default formats (pgm for gray and ppm for color):

```
// Loading a grayscale image
Image *imagegray = new ImageGray();
DiskXFile *image_file = new DiskXFile("lena.pgm", "r");
imagegray->loadXFile(image_file);

// Saving a color RGB image
Image *imagecolor = new ImageRgb();
DiskXFile *image_file = new DiskXFile("lena.ppm", "w");
imagecolor->saveXFile(image_file);
```

- other formats using a specific image loader (jpeg, gif, tiff):

```
gifDiskXFile loader("color.gif", "r");
imagecolor->loadImageXFile(&loader);
jpegDiskXFile writer("color.jpeg", "w");
imagecolor->saveImageXFile(&writer);
```

Basic features: handle a video

- uncompressed RGB video (avi format):
```
video = new rgbRawVideoFile();
```
- compressed video (most of codecs, including DivX)
```
video = new ffmpegVideoFile();
```
- video attributes:
```
video->open("meeting.avi");
print("width = %d\n", video->getwidth());
print("height = %d\n", video->getheight());
print("nframes = %d\n", video->getnframes());
print("fps = %g\n", video->getframerate());
print("codec = [%s]\n", video->getcodec());
video->close();
```

Basic features: reading a video

```
video->open("meeting.avi");

width = video->getwidth();
height = video->getheight();

image = new ImageRgb(width, height);

// loop on all video frames
for(int i = 0 ; i < video->getnframes() ; i++)
{
    video->read();
    image->copyFrom(width, height, video->pixmap, "rgb");
    image->save("image.ppm");
}

video->close();
```

Basic features: writing a video

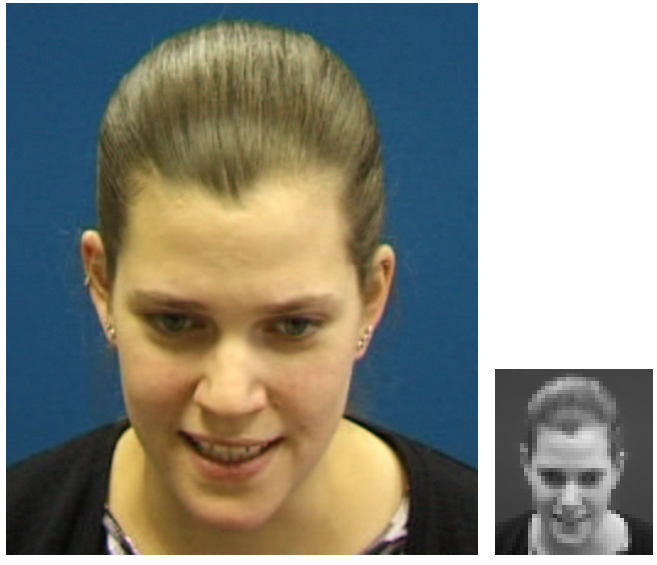
```
video_in->open("meeting.avi");
width = video_in->getwidth();
height = video_in->getheight();

video_out = new ffmpegVideoFile();
video_out->setIOption("width", width);
video_out->setIOption("height", height);
video_out->setROption("framerate", video_in->getframerate());
video_out->open("output.avi", "w");

// loop on all video frames
for(int i = 0 ; i < video_in->getnframes() ; i++)
{
    video_in->read();
    video_out->write(video_in->pixmap);
}

video_out->close();
video_in->close();
```

Outline

- × Introduction
- × Motivation
- × Reminder: Main concepts of Torch
- × Main concepts of Torch vision
- ▷ Basic features
  - × Image
  - – ipCore
- • Advanced features
- • Demonstrations

Basic features

- scaling:



```
ipCore *ip = new ipSubsample(width, height, "gray", 32, 40);
ip->process(imagegray);
image_out = new ImageGray();
image_out->copyFrom(32, 40, ip->getOutput(0), "gray");
image_out->save("scale.pgm");
```

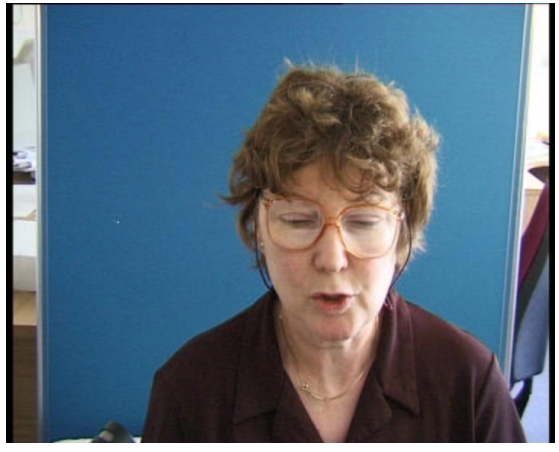
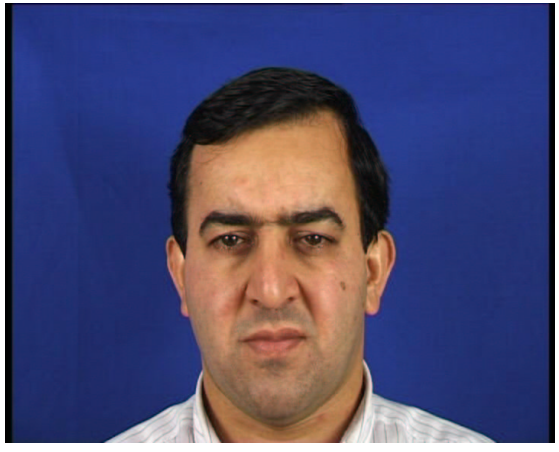
Basic features

- rotation by shear (Paeth rotation):

```
ipCore *ip = new ipRotate(-45, width, height, "gray");
ip->process(imagegray);
image_out = new ImageGray();
int width_out = ip->getWidthOut();
int height_out = ip->getHeightOut();
image_out->copyFrom(width_out, height_out, ip->getOutput(0), "gray");
image_out->save("rotate.pgm");
```

Basic features

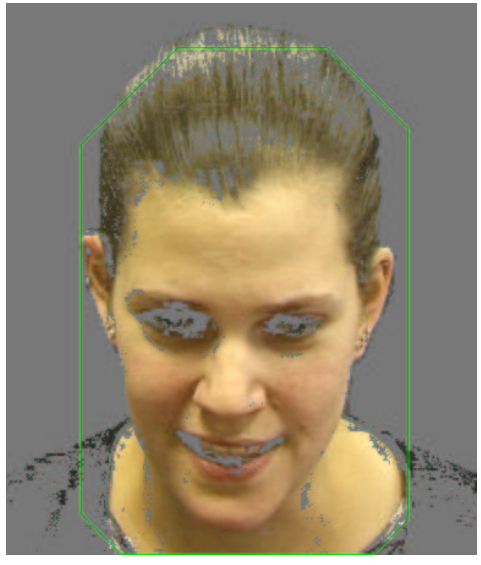
- image warping:

```
// Facial landmarks
sPoint2D *ldm_in = new sPoint2D [n_ldm];
sPoint2D *ldm_out = new sPoint2D [n_ldm];
// ...
ipCore *ip = new ipWarp(width, height, "gray", n_ldm, ldm_in, ldm_out
ip->process(imagegray);
image_out = new ImageGray();
image_out->copyFrom(width, height, ip->getOutput(0), "gray");
image_out->save("warp.pgm");
```

Basic features

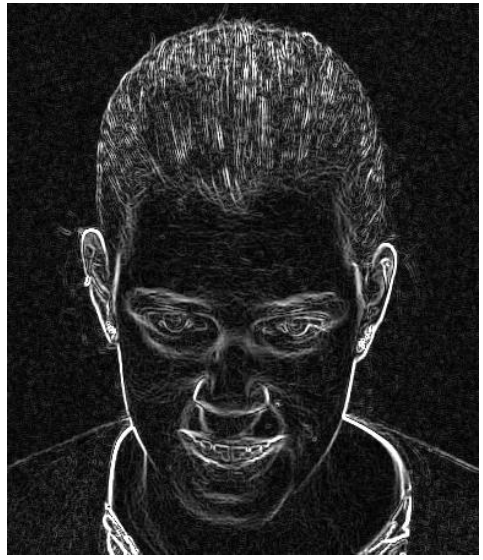
- skin color filtering and connected components:



```
ipCore *ip = new ipColorCube(width, height, "rgb");
ip->load("skin.colorcube");
ip->process(imagecolor);
image_out = new ImageRgb();
image_out->copyFrom(width, height, ip->getOutput(0), "rgb");
image_out->save("skin.ppm");
```

Basic features

- edge detection (Sobel, Canny):



```
ipCore *ip = new ipSobel(width, height, "gray");
ip->setROption("threshold", threshold);
ip->init();
ip->process(imagegray);
image_out = new ImageGray();
image_out->copyFrom(width, height, ip->getOutput(2), "gray");
image_out->save("gradient.pgm");
```

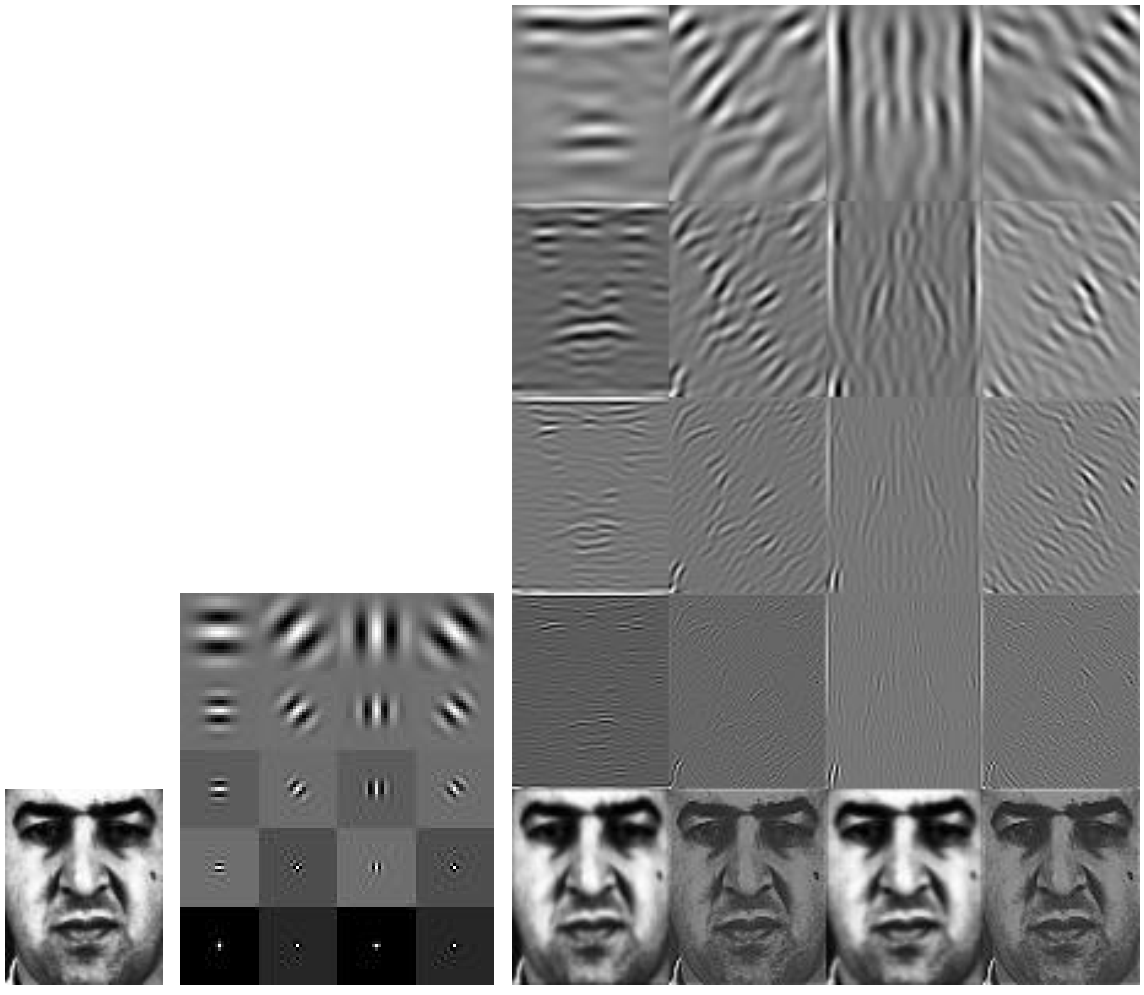
Basic features

- FFT/iFFT and DCT/iDCT:

```
ipCore *fft = new ipFFT2D(width, height);
fft->process(imagegray);
// Loop on coefficients
for(int i = 0; i < fft->seq_out->frame_size; i++)
        print("%g\n", fft->seq_out->frames[0][i]);

ipCore *ifft = new ipFFT2D(width, height, true);
ifft->process(fft->seq_out);
image_out = new ImageGray();
image_out->copyFrom(width, height, ifft->getOutput(0), "gray");
image_out->save("inverse.pgm");
```

Basic features

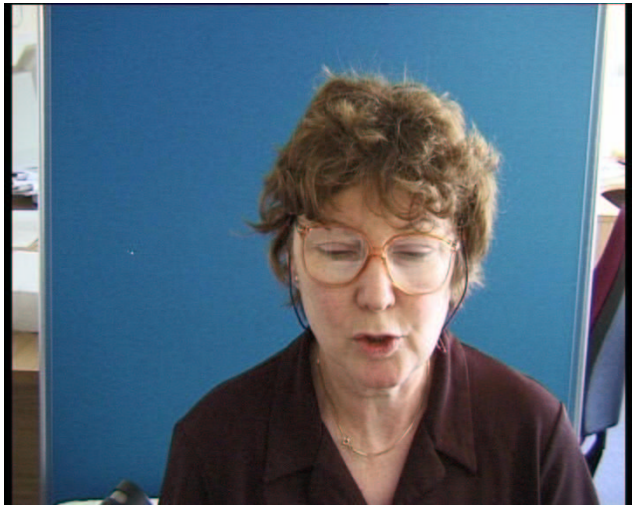
- gabor filters:



```
ipCore *ip = new ipGabor2D(size, size, scale, orientation, ...);
...
```

Basic features: images with Torch

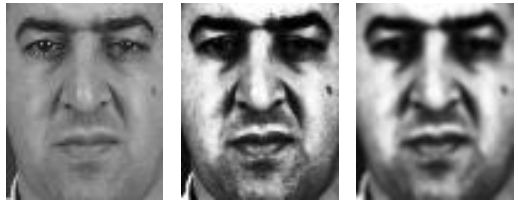
- image segmentation using GMMs and EM:



```
dataset = new MemoryDataSet;
sequence = new Sequence(width * height, 3);
dataset->setInputs(&sequence, 1);
gmm = new DiagonalGMM(3, 4, NULL);
trainer = new EMTrainer(gmm);
for(int i = 0 ; i < width*height ; i++)
    for(int j = 0 ; j < 3 ; j++)
        sequence->frames[i][j] = image->frames[0][i*3+j];
trainer->train(dataset, NULL);
```

Basic features: images with Torch

- face detection/recognition using MLPs:



```
// Image normalisation
ipCore *enhancing = new ipHistoEqual(width, height, "gray");
ipCore *smoothing = new ipSmoothGaussian3(width, height, "gray", 0.25

// the MLP and the image to forward
MyMLP *mlp = new MyMLP("face.mlp");
imagegray->load("face.pgm");

// let's do the job
enhancing->process(imagegray);
smoothing->process(enhancing->seq_out);
mlp->forward(smoothing->seq_out);
print("%g\n", mlp->outputs->frames[0][0]);
```

Outline

- × Introduction
- × Motivation
- × Reminder: Main concepts of Torch
- × Main concepts of Torch vision
- × Basic features
  - × Image
  - × ipCore
- ▷ Advanced features
- • Demonstrations

Advanced features: illumination normalization

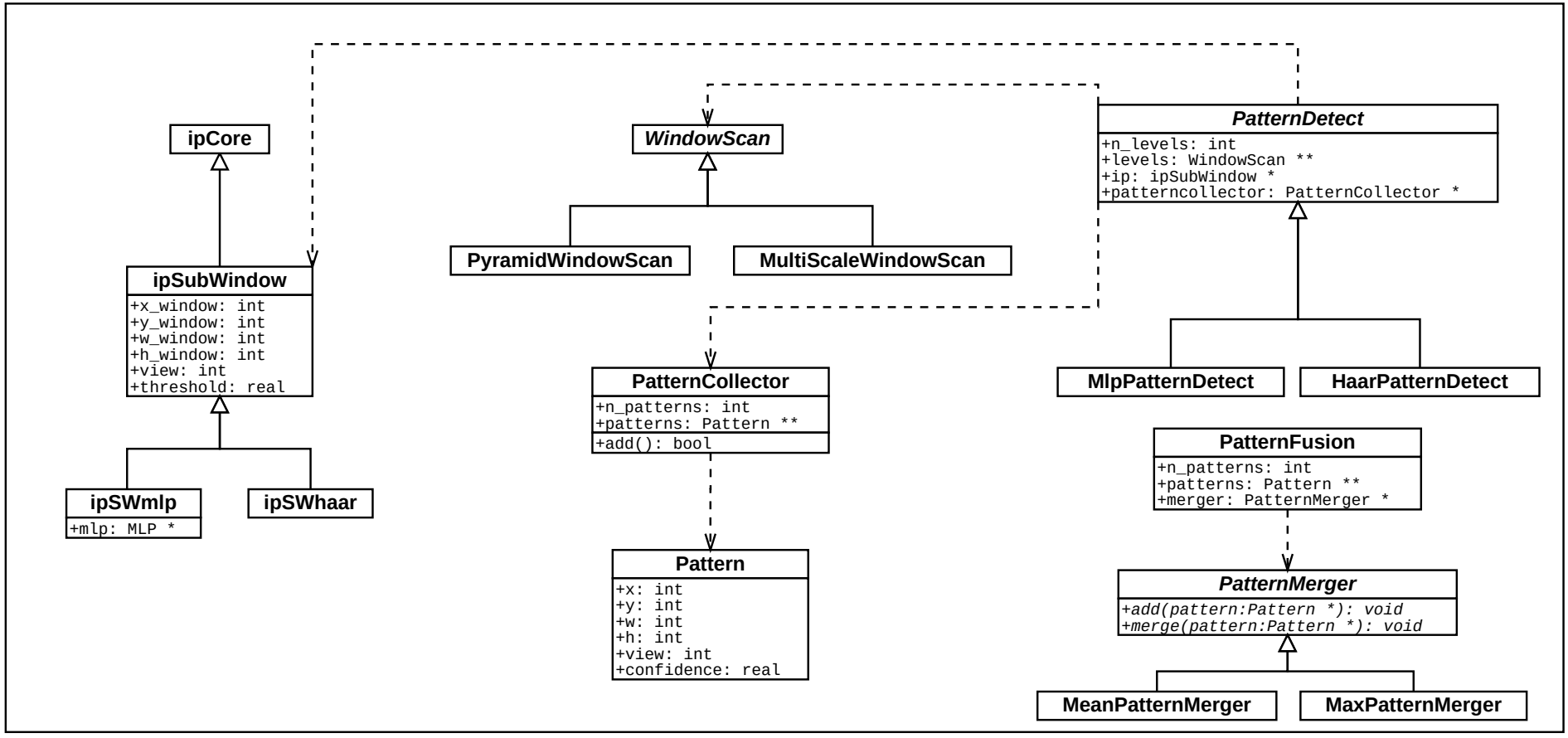
- relaxation/isotropic:

- multigrid/isotropic:

Advanced features: face detection

Advanced features: face detection

- generic pattern detection framework:



Advanced features: face detection/recognition

- features:
  - EigenMachine (PCA, LDA, QZ-LDA, CS-LDA, MMC)

  - DCT/DCT-mod2

  - MCT, LBP

- classifiers: metrics, MLP, SVM, GMM, HMM, HMM2, Stump classifiers for Boosting, Cascade training and Boostrapping from large image datasets

Overview of Torch vision

- features availables:
  - image I/O: reads/writes basic image formats (pgm, ppm, gif, tif, jpeg)
  - video I/O: video acquisition (frame grabber or USB) and decoding/encoding of video files (avi, mpeg1/2).
  - image transformation: rotation, flip, crop, warp, ...
  - image processing: edge detection, photometric normalisations, ...
  - segmentation: morphological operators, connected components, ...
  - feature extraction: histograms, PCA, LDA, 2D-DCT, 2D-FFT, ...
  - geometry: creates, manipulates and draws 2D objects
  - pattern detection framework (applied to face detection)
  - machines and trainers for face detection/recognition

Development Team

- main contributors: S. Marcel and Y. Rodriguez
- other contributors: J. Tiphaigne, O. Masson, A. Just. C. Sanderson, F. Cardinaux, G. Heusch, T. Sauquet
- You are welcome to contribute !!
- Reminder: http://www.idiap.ch/~marcel/en/torch3/torch3vision.php

Outline

- × Introduction
- × Motivation
- × Reminder: Main concepts of Torch
- × Main concepts of Torch vision
- × Basic features
  - × Image
  - × ipCore
- × Advanced features
- ▷ Demonstrations

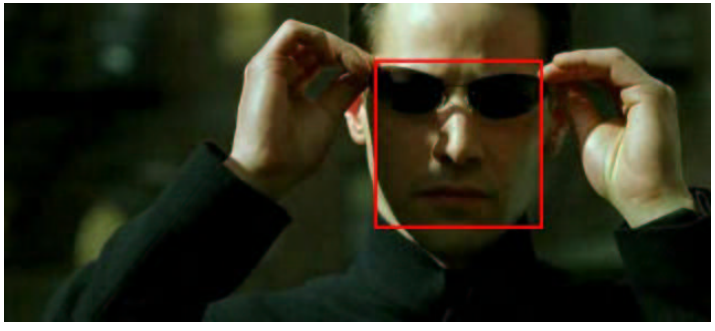
Demonstrations

- Demos powered by Torch vision:
  - face detection (http://www.idiap.ch/~marcel/en/facedemos.php)
  - biometric authentication (http://www.idiap.ch/~marcel/en/biometry.php)
- Face detection in videos (frontal faces only):
  - Meeting room scenario (IM2 and AMI projects) [1] [2]
    - controlled illumination conditions (skin color to speed-up search),
    - priors on face location and size,
    - complex but fixed background,
    - not so fun !
  - MI2 and not IM2 !
    - uncontrolled illumination conditions (skin color is useless),
    - no priors on face location and size,
    - complex and changing background,
    - that's more fun !
  - do you want more ?

Computer Vision Libraries

- VXL (http://vxl.sourceforge.net)
- RAVL (http://ravl.sourceforge.net)
- Intel OpenCV (http://www.intel.com/research/mrl/research/opencv)
- LTI-Lib (http://ltilib.sourceforge.net/doc/homepage/index.shtml)
- Gandalf (http://gandalf-library.sourceforge.net)

Video files to download

- Meeting room scenario (IM2 and AMI projects)
  - Full (ftp://ftp.idiap.ch/pub/marcel/demos/video/facemeeting2.avi)
  - Close up (ftp://ftp.idiap.ch/pub/marcel/demos/video/closeup3.avi)
- MI2 (ftp://ftp.idiap.ch/pub/marcel/demos/video/mi2.avi)

- Matrix 2 (ftp://ftp.idiap.ch/pub/marcel/demos/video/matrix2.avi)