A Tutorial on VLFeat

IMAGE PROCESSING LABORATORY

Antonino Furnari

Image Processing Lab
Dipartimento di Matematica e Informatica
Università degli Studi di Catania

`furnari@dmi.unict.it`

17 aprile 2014

MATLAB & Computer Vision

- MATLAB offers some powerful toolboxes to work with images and Computer Vision:
  - **Image Processing Toolbox**, providing basic functions to read, write, show and process images;
  - **Computer Vision Toolbox**, providing functions for feature extraction, object recognition camera calibration and stereo vision;
  - **Image Acquisition Toolbox**, providing utilities for image acquisition (e.g., from webcam).

VLFeat

VLFeat is an open source library which implements popular computer vision algorithms. We will have a look at:

- Harris Corner Detector;
- Scale Invariant Feature Transform (SIFT);
- Dense SIFT.

It is written in C for effiency and have interfaces for both MATLAB and C.

http://www.vlfeat.org/

Installation

MATLAB

- Download the last binary package from: http://www.vlfeat.org/download.html;
- extract the archive to the chosen directory (e.g., `~/MATLAB/vlfeat-0.9.18/`);
- from the MATLAB prompt type: `run('~/MATLAB/vlfeat-0.9.18/toolbox/vl_setup');`
- check that everything is working by entering:
  ```
  vl_setup demo
  vl_demo_sift_basic
  ```
- for permanent setup, add the “run” line to your `startup.m` file.

Installation

C

- Add the appropriate directory `~/MATLAB/vlfeat-0.9.18/bin/ARCH` to the system path (where ARCH is your architecture) or copy the `~/MATLAB/vlfeat-0.9.18/bin/ARCH/libvl.so` file to directory `/usr/local/lib/libvl.so`;
- copy the `~/MATLAB/vlfeat-0.9.18/vl/` dir to `/usr/local/include/vl` or remember to specify `~/MATLAB/vlfeat-0.9.18/` as include path when compiling.

OpenCV Interoperability

It might be useful to use both the OpenCV library and VLFeat in the same application. A grayscale `cv::Mat` can be easilly converted into a `float*` (the image format used by VLFeat) using the following code:

```
Mat toFloat;
cvmat.convertTo(toFloat,CV_32F);
float *vlimage = (float*) tofloat.data;
```

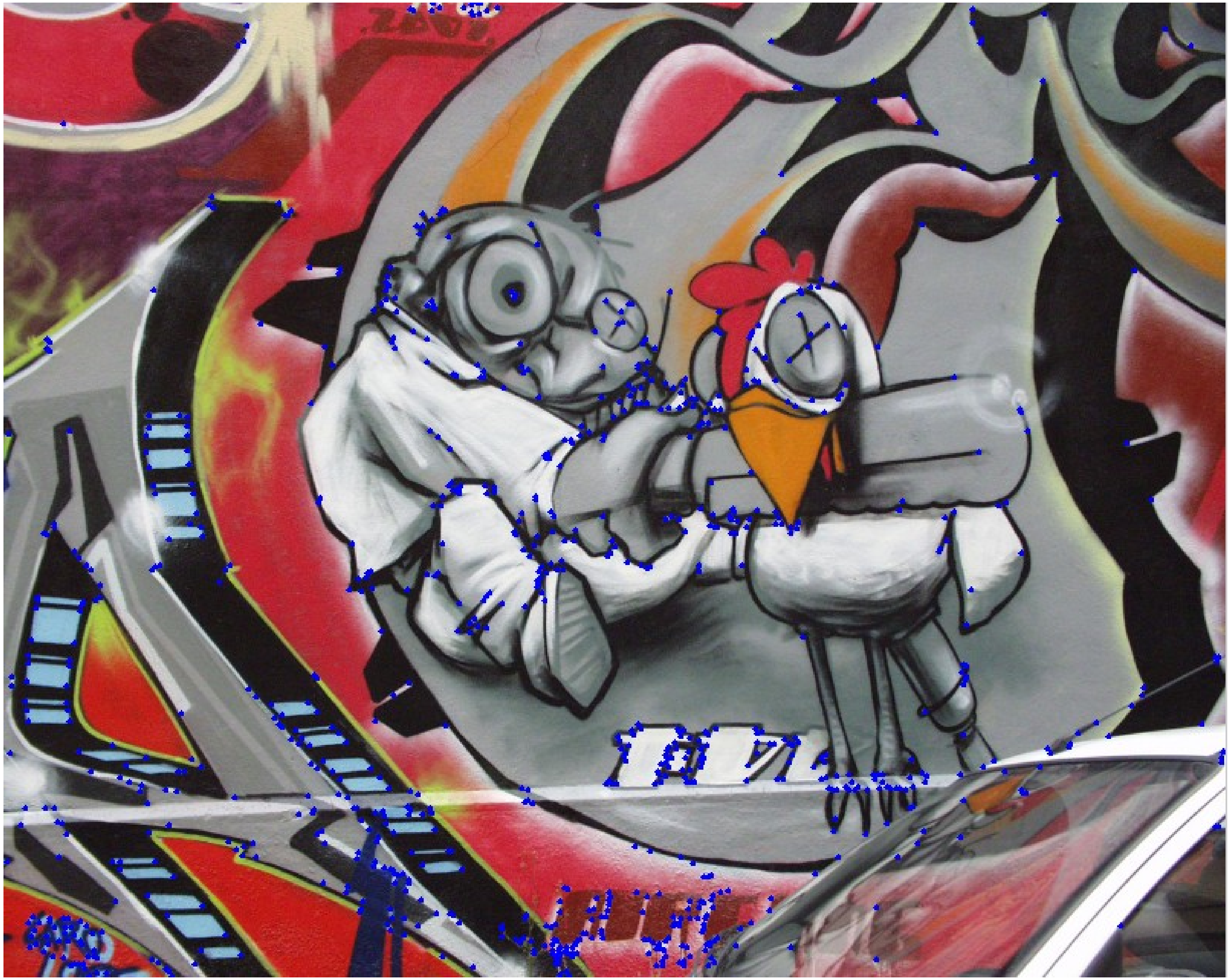
Harris Corner Extraction

- The Harris corners can be extracted computing the local maxima of the Harris corner response.

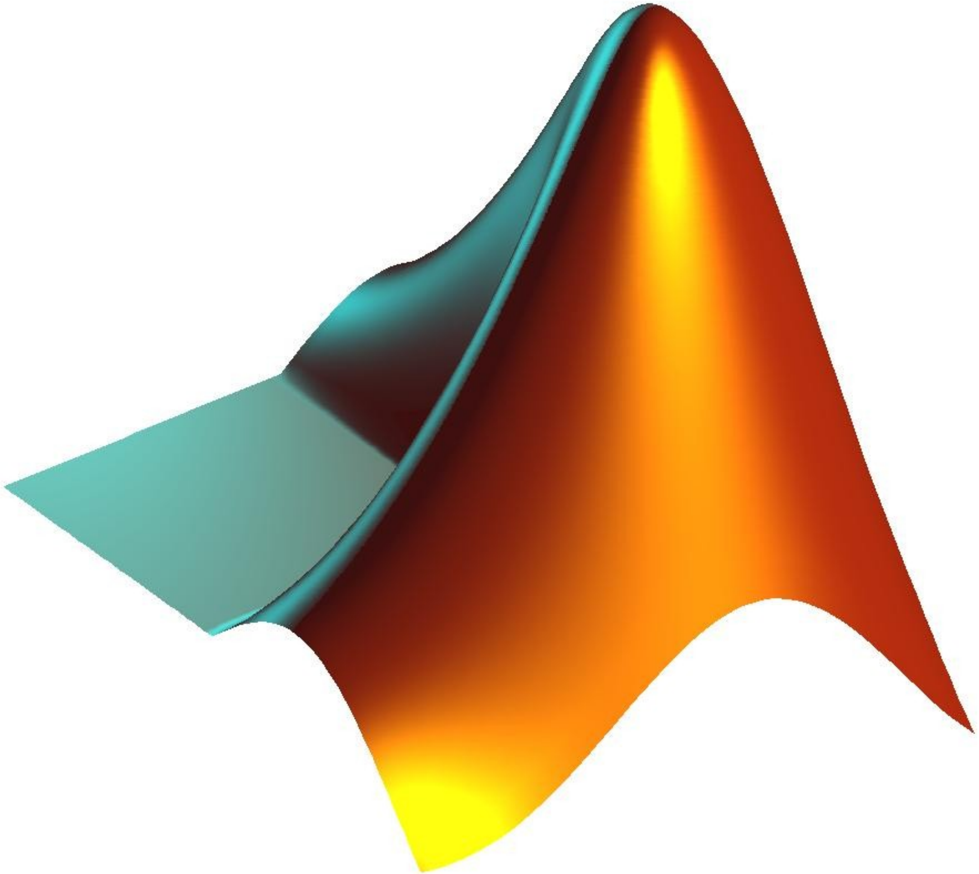
```
response = vl_harris(image,s);
response(response/max(response(:))<0.1) = 0;
idx = vl_localmax(response);
[i,j] = ind2sub(size(image),idx);
```

- Where
  - `s` is the variance of the gaussian window (integration scale);
  - the Harris response is thresholded removing the 10% of the values;
  - `vl_localmax` computes the local maxima;
  - `ind2sub` is used to retrieve the (row, column) coordinates.

Harris Corner Extraction (2)



Ex01: Harris Corner Extraction



SIFT Features extraction

The SIFT features can be extracted using the `vl_sift` procedure. It operates on single precision grayscale images. A conversion to this format is therefore required:

```
img1 = imread('img1.ppm');
img1_gray = single(rgb2gray(img1));
[f1,d1] = vl_sift(img1_gray);
```

Where `f1 [4 x n]` contains the feature points (including scale and orientation), `d1 [128 x n]` contains the descriptors corresponding to the points in f and `n` is the number of extracted keypoints.

SIFT Features extraction (2)

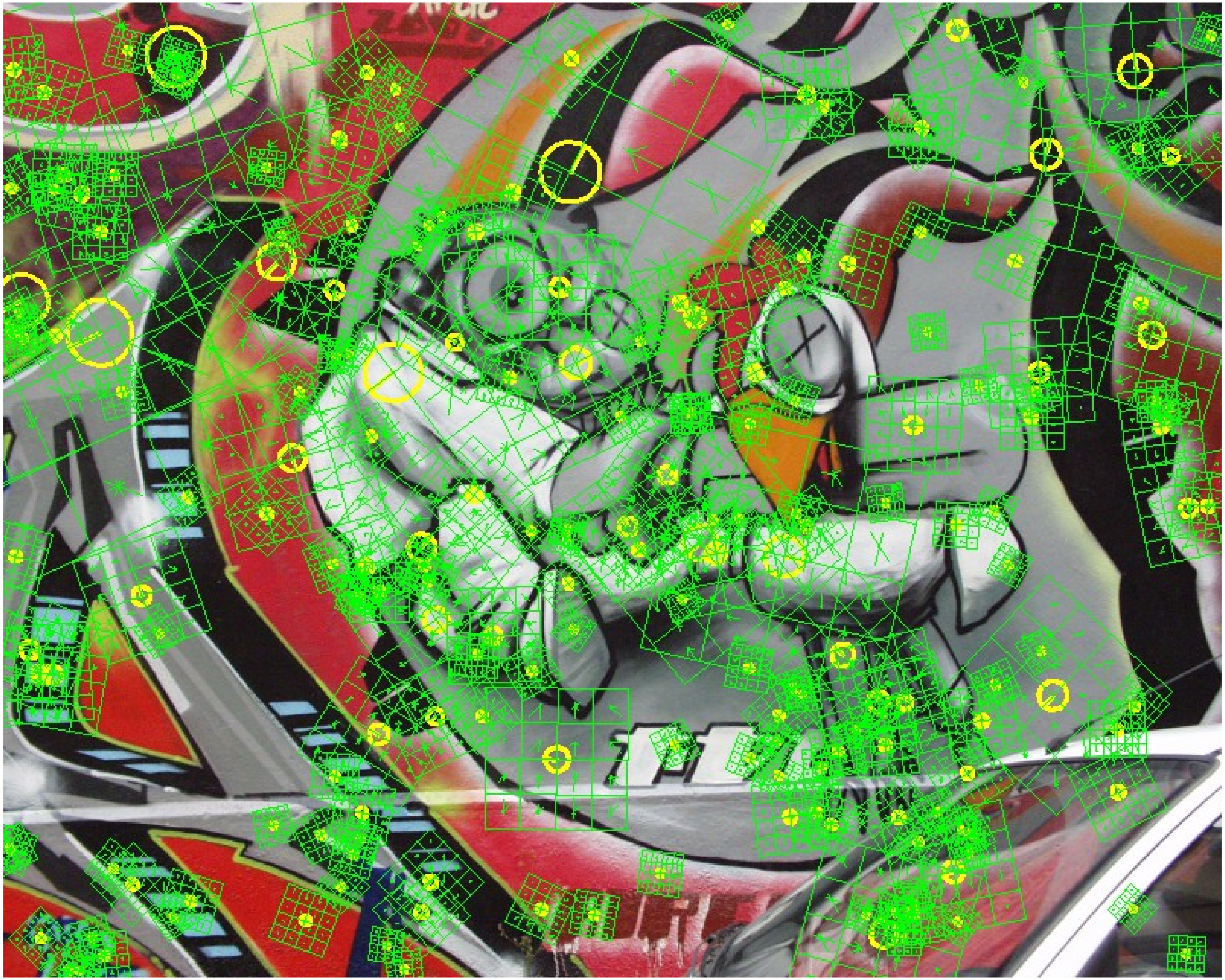
Each column `f` of `f1` contains:

- The point coordinates `f(1:2)`;
- The point scale `f(3)`;
- The point orientation `f(4)`;
- The functions:

  ```
  vl_plotframe(f1);
  vl_plotsiftdescriptor(d1,f1);
  ```

  can be used to plot both the keypoints and the representations of the descriptors.

SIFT Features extraction (3)



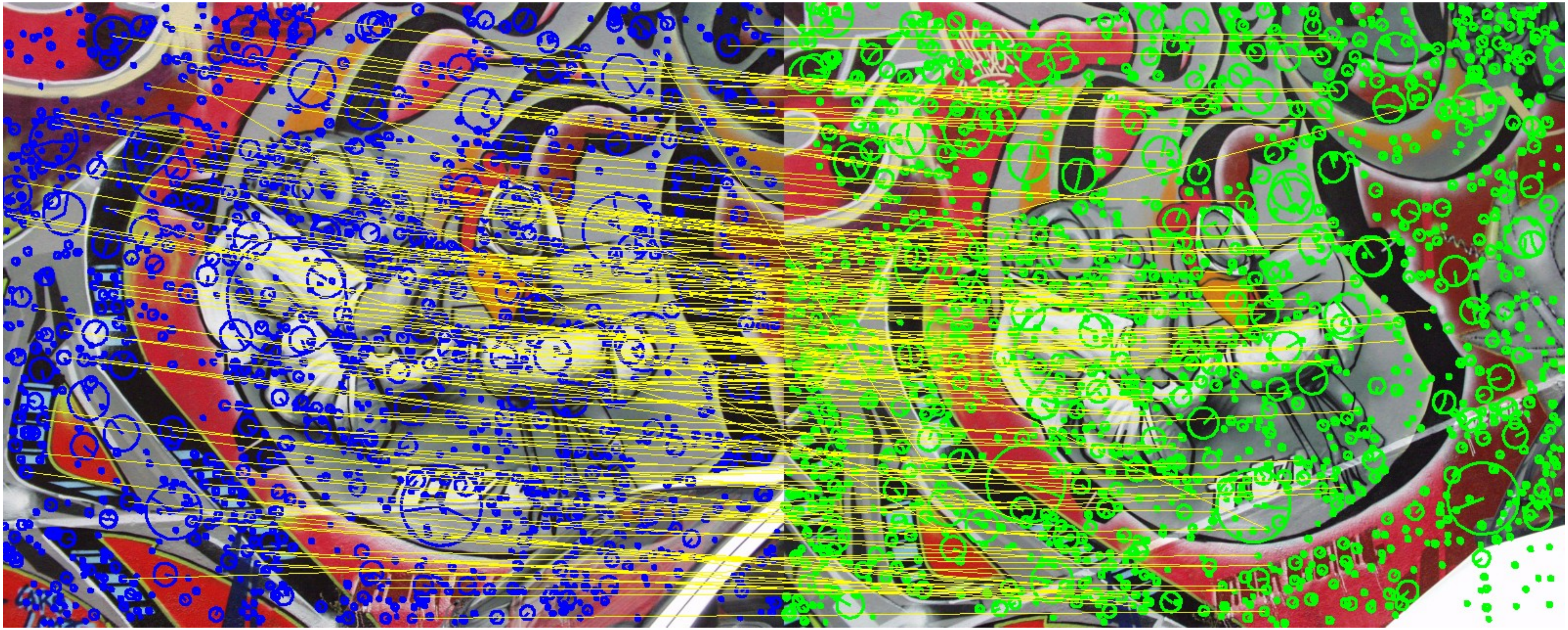
SIFT Features Matching

The matching can be performed using vl_ubcmatch, which implements the Lowe's algorithm to reject matches which are too ambiguous

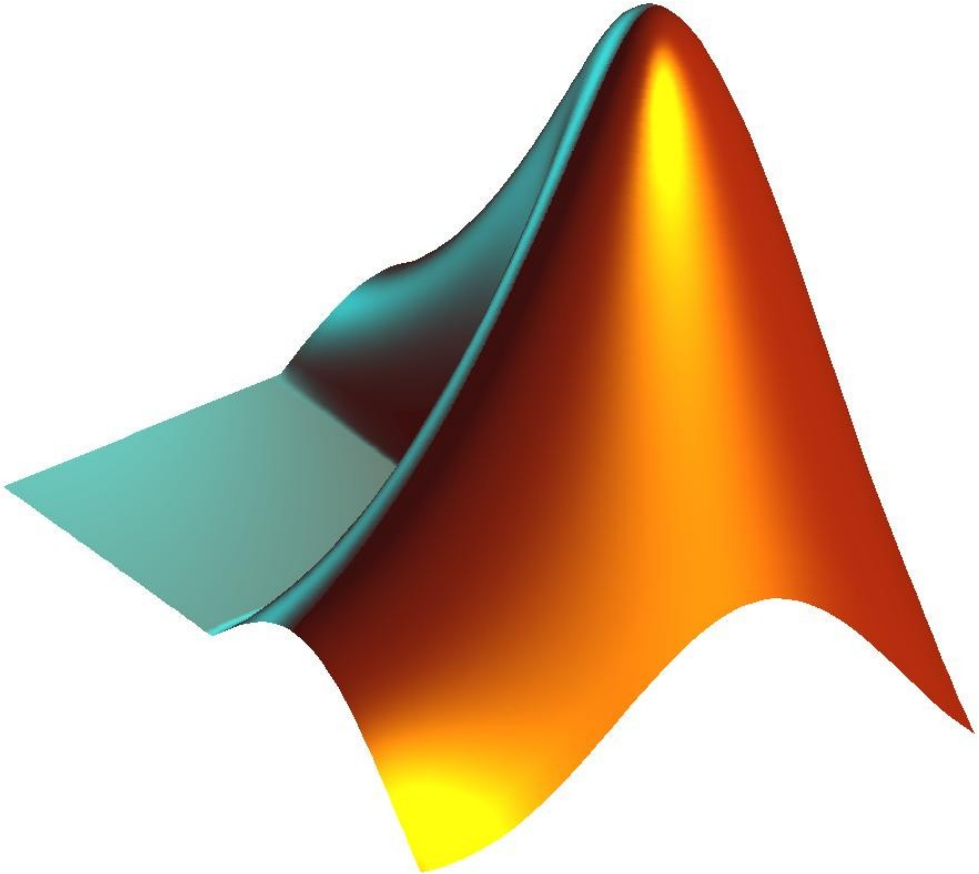
```
[matches,scores] = vl_ubcmatch(d1,d2);
```

where `d1` and `d2` are the descriptors corresponding to the keypoints `f1` and `f2` extracted from different images.

SIFT Features Matching



Ex02: SIFT Feature Extraction



Dense SIFT Extraction

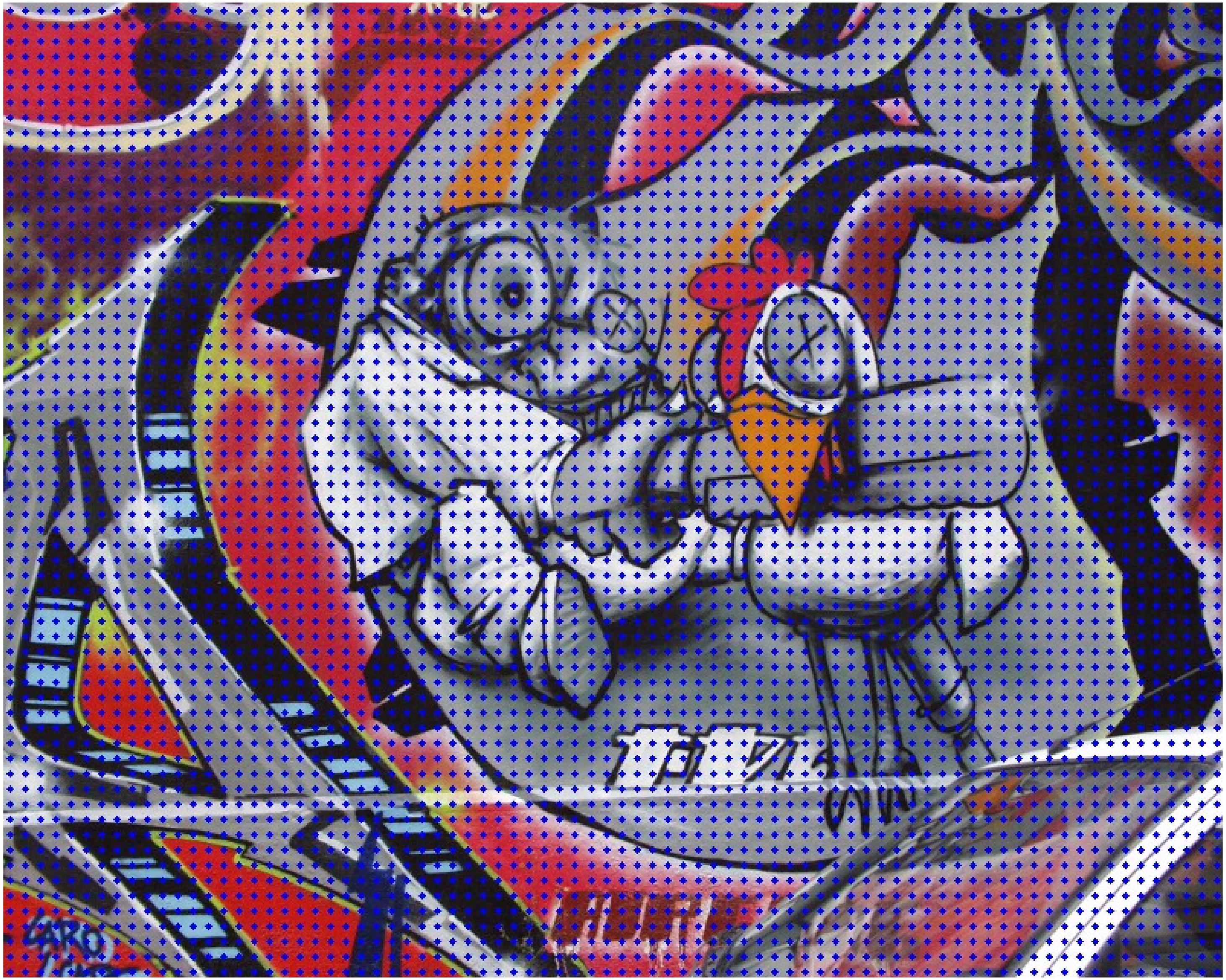
A dense variant of SIFT is included in VLFeat. It skips the feature detection step and performs the description of a number of points selected at a unifrom step:

```
[f,d] = vl_dsift(image,'STEP',10);
```

In this case the feature points are considered at a step of 10 pixel.

Note that `f` contains only the point coordinates (no scale or orientation is contained).

Dense SIFT Extraction (2)

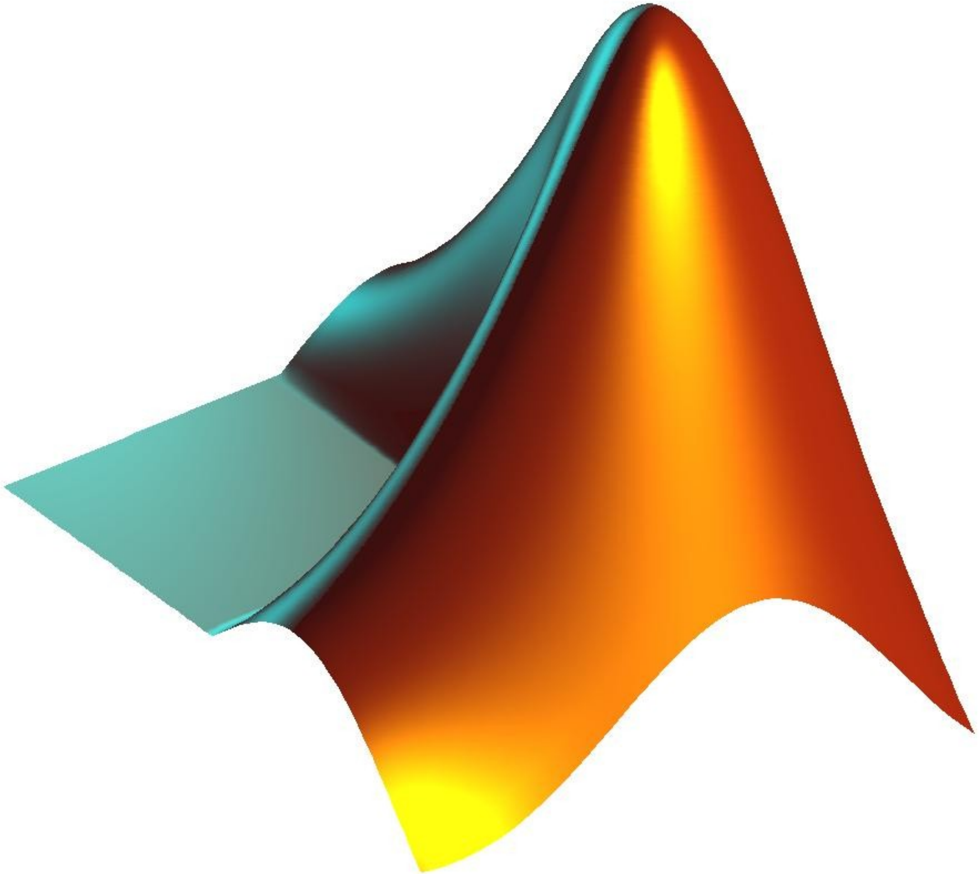


Dense SIFT Extraction (3)

Some parameters (and default values) which can be passed to `vl_dsift`:

- `Step` (1): extracts a SIFT descriptor each `step` pixels;
- `Size` (3): the dimension of the spatial bins used for the computation of the descriptor;
- `Bounds` (whole image): a rectangular area which should be considered for the extraction in the format [x1, y1, x2, y2];
- `Fast`: if specified a fast version of the algorithm is used.

Ex03: Dense SIFT Feature Extraction



Other VLFeat Capabilities

- Operations on the images (e.g., color space conversion, gaussian smoothing);
- Maximally Stable Extremal Regions (MSER);
- Covariant Feature Detectors (e.g., DoG, Harris-Affine, Harris-Laplace);
- Histogram of Oriented Gradient (HOG);
- Support Vector Machine (SVM);
- K-means.

Question Time

Contacts

- For any doubts please contact me:
  - `furnari@dmi.unict.it;`
  - `Room 30;`
- Slides available at:
  - Studium course page:
    - http://studium.unict.it/dokeos/2014/courses/73072C2/