Digital Image Processing

Berlin University of Technology (TUB),
Computer Vision and Remote Sensing Group
Berlin, Germany

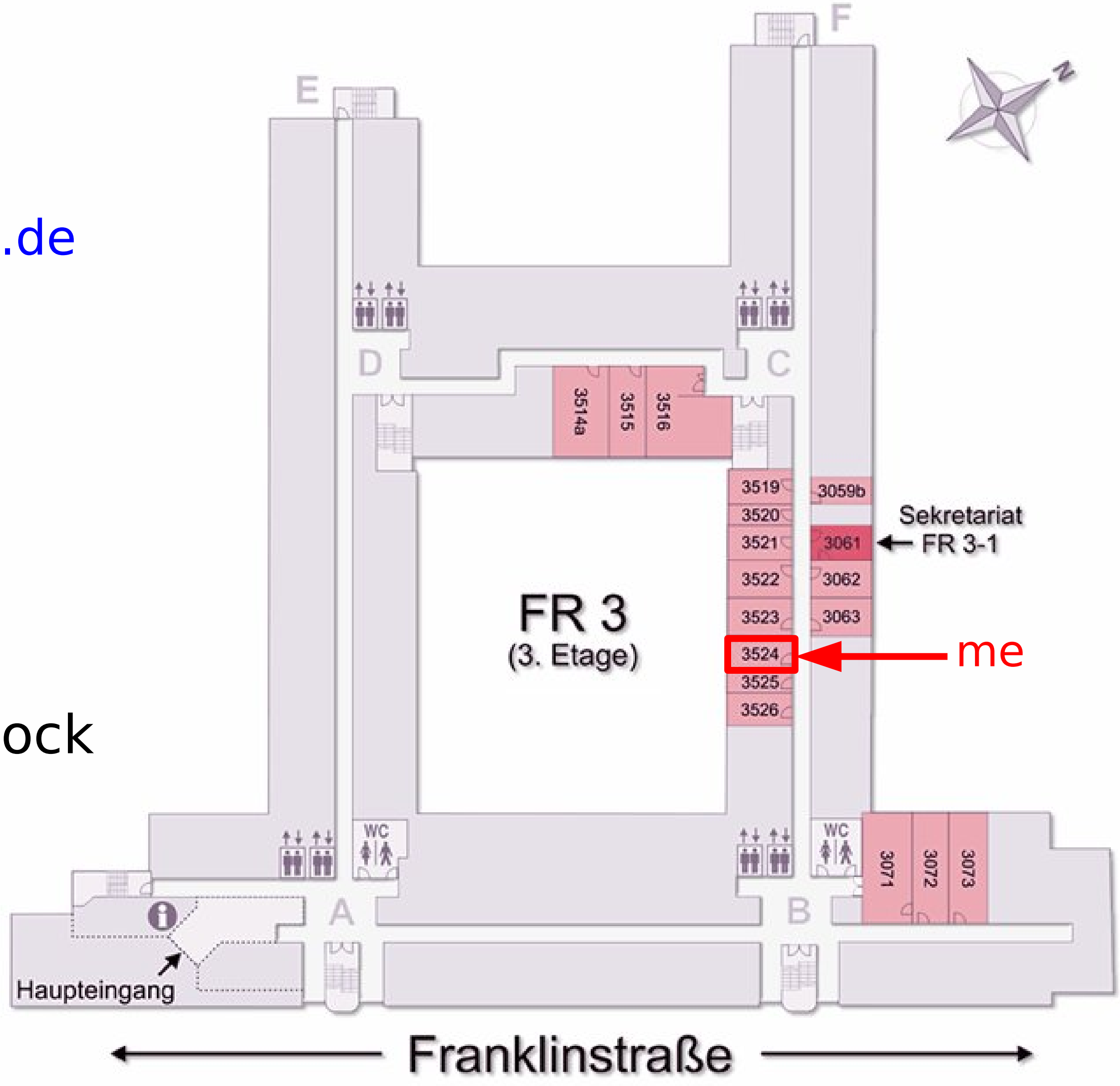
Contact

Ronny Hänsch

- **E-Mail:** r.haensch@tu-berlin.de

- **Office**
  - FR 3524,
    Franklin Building,
    3rd Floor

- **Consultation Time**
  - Monday, 10:00-12:00 o'clock
  - (Or by arrangement)



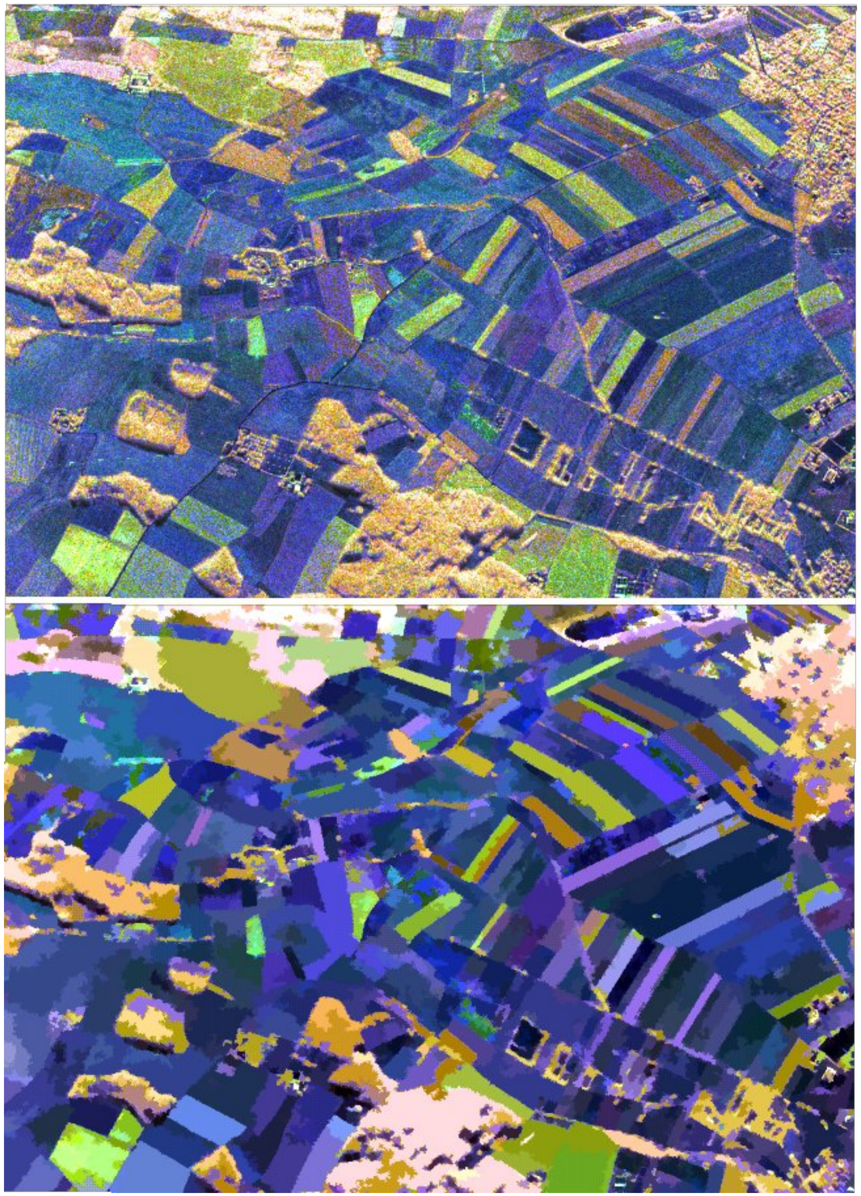
Research Topics

- **Computer Vision**
    - Feature Extraction
    - Segmentation/Clustering
    - Object Categorization

- **Artificial Intelligence**
    - MLP
    - Random Forests
    - Probabilistic Models

- **Remote Sensing**
    - PolSAR
    - Optical Imagery
    - Object Recognition

- **Photogrammetric CV**
    - Robust 3D Reconstruction



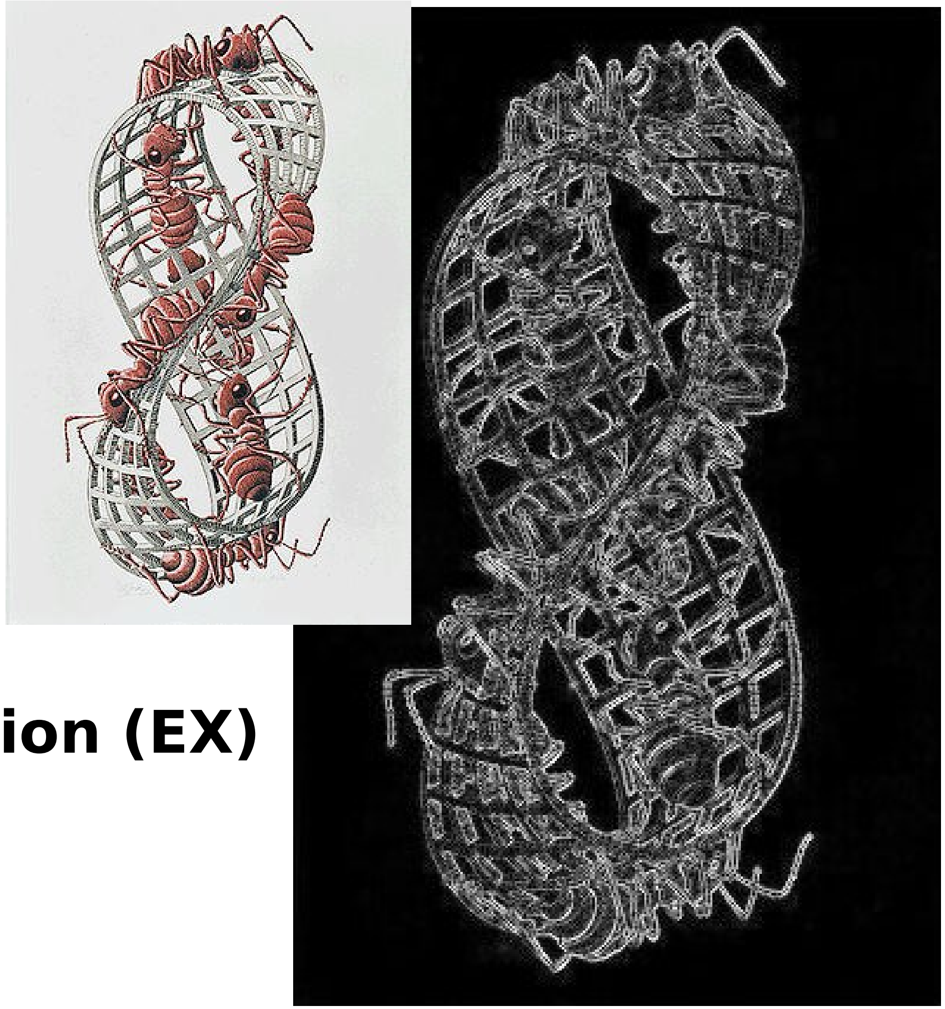
Teaching

- **Digital Image Processing (EX)**
  - Image → Image
  - Image → Description
  - Summer Term

- **Automatic Image Analysis (EX)**
  - Image → Object Model
  - Image → Object Detection
  - Winter Term

- **Photogrammetric Computer Vision (EX)**
  - Image(s) → 3D Model
  - Winter Term

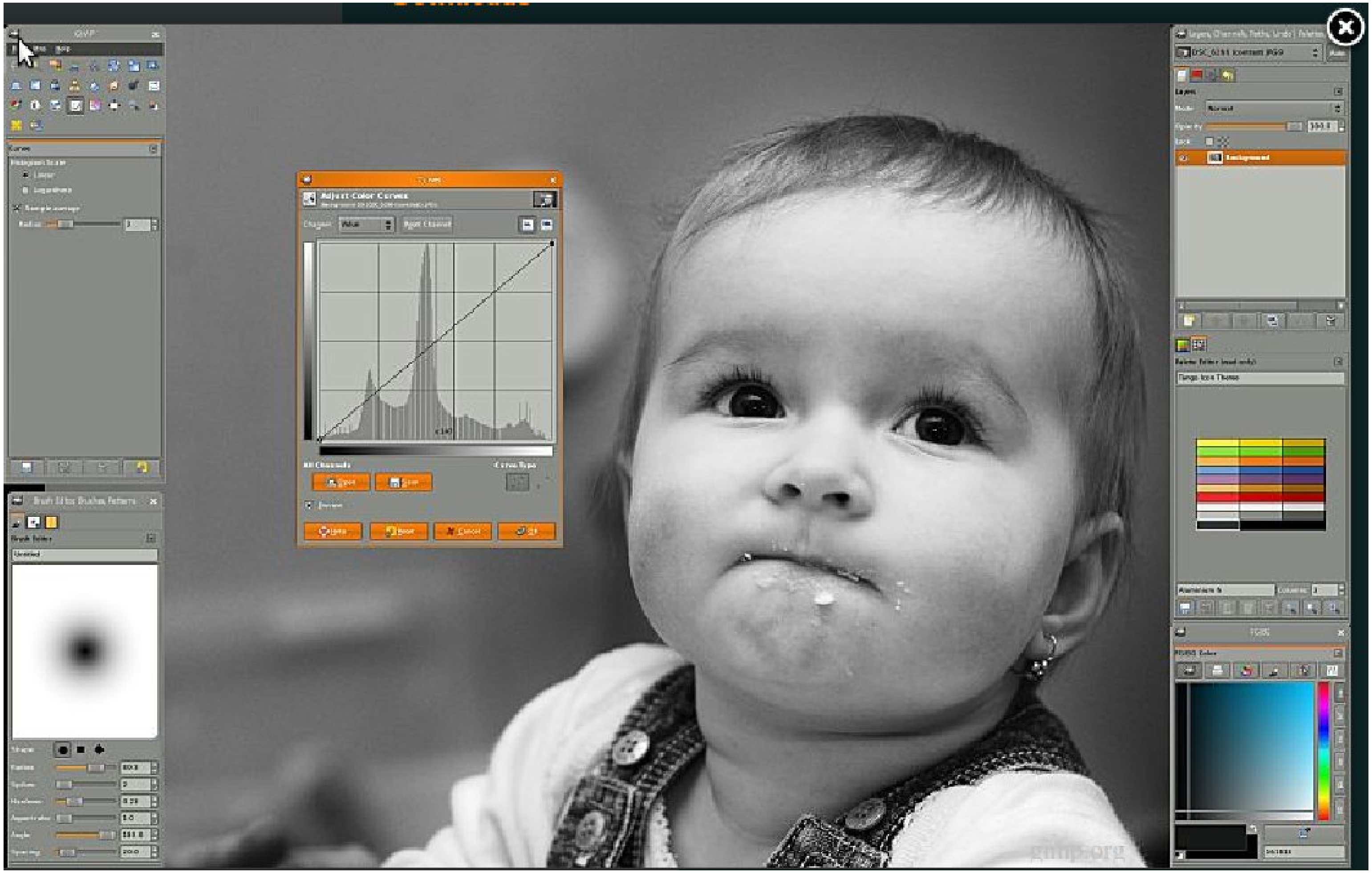
- **GPU-Project**
  - Summer Term 2012



Supervision of Bachelor-/Master-Thesis

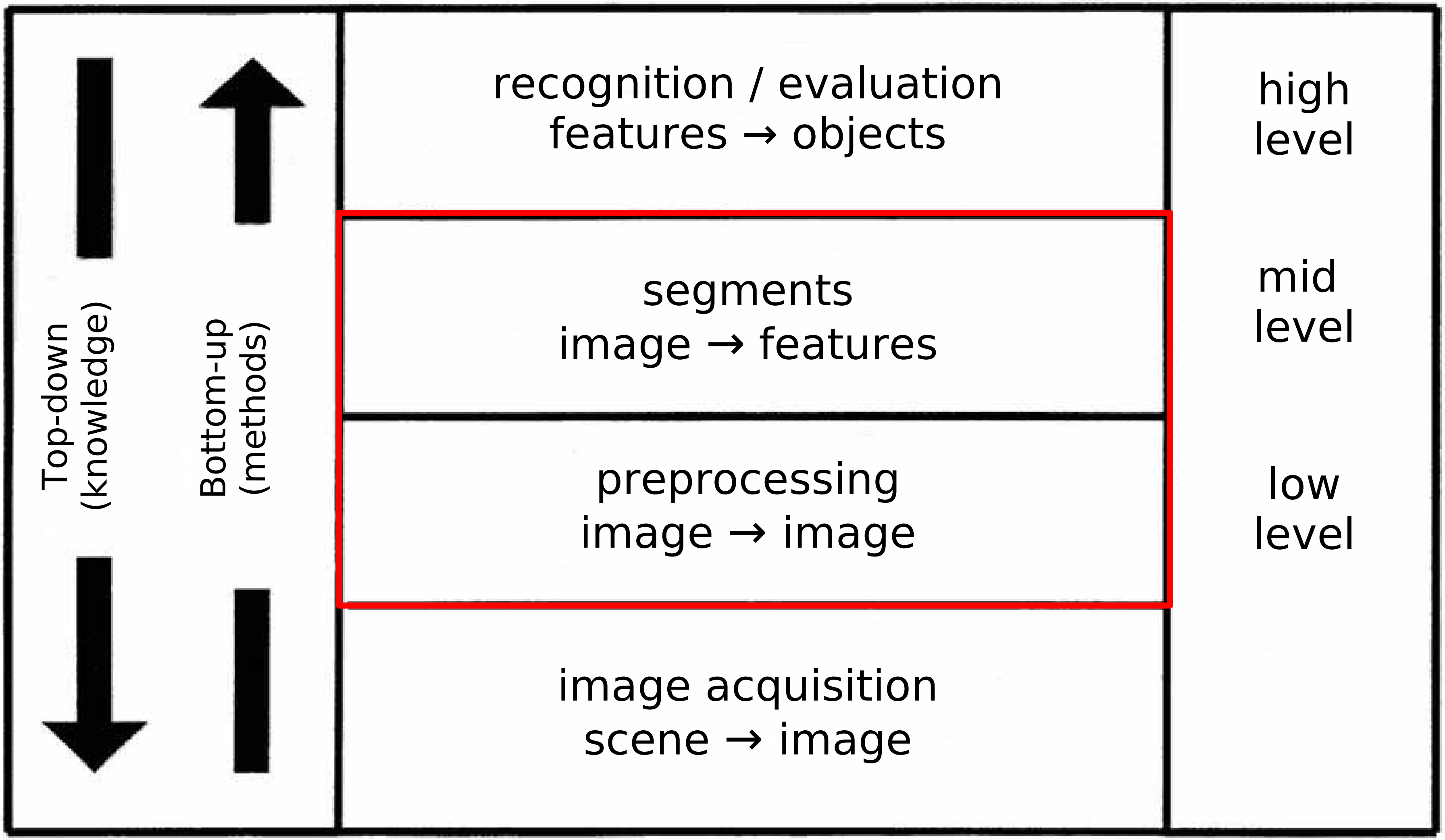
What are you gonna learn?

Photoshop, Gimp, …



- NOT how to USE it (image editing)
- BUT how it WORKS (image processing)

What are you gonna learn?



Top-down (knowledge)

Bottom-up (methods)

recognition / evaluation
features → objects

segments
image → features

preprocessing
image → image

image acquisition
scene → image

high level

mid level

low level

How are you gonna learn it?

1. Visit lectures
   - Every week (HFT-FT 101, Tuesday, 10-12o'clock)

2. Visit exercises
   - Every two weeks (E20, Friday, 10-12o'clock)

3. Doing homework
   - Every two weeks (Consultation time: FR3524, Monday, 10:00-12:00 o'clock)

4. ASK QUESTIONS!
   - Always! But: Ask me, not your neighbour

5. (Read further material)
   - As often as possible

Books

- Petrou, Maria: Image Processing – The Fundamentals, 2nd edition, Wiley 2011
- Gonzalez, Woods: Digital Image Processing, 2nd edition. Prentice Hall, Upper Saddle River 2002.
- Jähne: Digital Image Processing. Springer Verlag, Berlin 2005.
- Brigham: The Fast Fourier Transform and Its Applications. Englewood Cliffs, 1988.

(See course web-page for more)

- Scientific paper: www.ieeexplore.com (free download from TU-network)

Infos and Exam

WWW

- Information, important announcements:
  - ➔ http://www.cv.tu-berlin.de Announcements:'Lectures'

- Slides and other material: ISIS

Exam

- Mid-term:
  - Near the middle of the term, in place of an exercise
  - Duration: ca. 30 min
  - No grade, but pass is necessary to take part at the final exam

- Final:
  - At the end of the term
  - Duration: ca. 60 min

- Questions in English, answers in English or German

Homework

What to do?

Programming methods for processing digital images

How?

In groups of 3-4 people

Programming Language: C++ & OpenCV

Completion of provided software packages

- Class descriptions (header files): given
- Includes: given
- Basic functionality: given
- Specific functions: Your task!

Goal?

Practise, Learning. No grades!

But pass necessary to take part at the exam

Homework

- “Grades”

+++	more than just a correct solution (efficient, clever, cool, ...)
++	correct solution
+	some minor errors, but still acceptable
-	not acceptable → re-work
- -	failed: you are not allowed to write the exam!

Homework

- Next meeting in **ONE** week

- **BEFORE**: send per email at r.haensch@tu-berlin.de:
  - Your code (as .cpp-file)
  - Input images (if not provided)
  - Output images

- **DURING**: hand in your print-out (one per group)

- Algorithms more important than code (but try!)

- Your solution should include:
  - Cover stating your group ID and all group members
  - All files that are necessary to compile and run your program
  - Well documented code
  - Input and output images (maybe even intermediate results)
  - If necessary, a brief discussion/explanation of your results

1. Exercise

C++ and OpenCV

Given:

- Main function
  - Variable declaration

Todo:

- [Install C++-compiler]
- [Install OpenCV]
- Main function
  - Load image
  - Do something (reasonable)
  - Save image

Deadline:

- Next meeting at **20.04.2012**, 10am

Very brief introduction to C++

Variable decleration:

```
    <typ> <name>;
    double  numberOfSomething;
```

Allowed characters for names: a-z, A-Z, 0-9, _

```
#include <iostream>

using namespace std;

int main(){

    int i = 100;
    double d = 3.12;

}
```

Very brief introduction to C++

Array decleration:

```
    <typ> <name>[numberOfElements];
    double  someArray[5];
```

NOTE: Never write more elements than size of array!

```
#include <iostream>

using namespace std;

int main(){

    int arr[10];

    arr[0] = 1;
    arr[10] = 2;        // BAD IDEA!

}
```

Very brief introduction to C++

Structures:

```
#include <iostream>
using namespace std

struct person{
        int age;
        char fstName[20];
        char lastName[20];
}

int main(){

        Person me;
        me.age = 28;
        strcpy(me.fstName, “Ronny”);
        strcpy(me.lastName, “Haensch”);

        cout  << me.age << “ “ << me.fstName << endl;
}
```

Very brief introduction to C++

Program flow:

```
if (condition){
    // do something
}else{
    // do something else
}

switch(c){
    case 'a':
        cout << 'a' << endl;
        break;
    case 'b':
        cout << 'b << endl;
        break;
    default:
        cout << “neither a nor b” << endl;
}
```

Very brief introduction to C++

Loops:

```
int i;
for(i=0; i<10; i++){
	cout << i << endl;
}
```

```
int i;
while(i<10){
	cout << i++ << endl;
}
```

Very brief introduction to C++

Functions:

```
#include <iostream>
using namespace std;

void hello(void);

void hello(void){
    cout << “hello”  << endl;
}

int main(){
    hello();
    return 0;
}
```

Very brief introduction to C++

Pointer:

```
#include <iostream>
using namespace std;

struct person{   int age; }

int main(){

        person* me;
        (*me).age = 28;
        me->age = 28;


        cout  << me.age << endl;
}
```

Very brief introduction to C++

Functions:

```
#include <iostream>
using namespace std;

int func1(int a, int b){     return  a +   b;   }
int func2(int* a, int* b){   return *a + *b;   }
int func3(int& a, int& b){return a + b;        }

int main(){
        int a=3; int b=4;

        // Call by value
        cout << func1(a, b) << endl;

        // Call by pointer
        cout << func2(&a, &b) << endl;

        // Call by reference
        cout << func3(a, b) << endl;
}
```

Brief introduction to OpenCV

```
#include <iostream>
#include <opencv2/highgui/highgui.hpp>
using namespace std;

int main(int argc, char** argv){

    cv::Mat* img = imread( argv[1], 0 );          // load image as gray-scale

    // show image
    cv::namedWindow( “example”);
    cv::imshow( “example”, img);

    Mat newImg( img.cols, img.rows, CV_8U, cv::Scalar(0) );
    // do something fancy
    fancyFunction(img, newImg);

    cv::imwrite(“coolResult.png”, newImg);

    cv::waitKey(0);
}
```

Brief introduction to OpenCV

Matrix generation, an example:

```
// C/C++
float vals[] = {1,1,1,1,1,1,1,1,1};

// OpenCV
cv::Mat kernel(3, 3, CV_32FC1, vals);  // creates 3x3 matrix of floats “1”
```

Accessing matrix data (the easy way)

```
kernel.at<float>(row, column)

colorImage.at<cv::Vec3b>(row, column)[channel]
```

Brief introduction to OpenCV

Accessing Image data – The hard way

```
float sum( cv::Mat& img ){

    float s = 0.0;

    for(int y=0; y < img.rows; y++){

        uchar* data = img.ptr<uchar>(y);

        for(int x=0; x < img.cols; x++ ) {
            s += ptr[x];
        }
    }
    return s;
}
```

Brief introduction to OpenCV

Compilation

```
user@comp:~/path$ g++ -o dip dip.cpp -lopencv_core -lopencv_imgproc -lopencv_highgui
user@comp:~/path$ g++ -o my_example my_example.cpp `pkg-config opencv --cflags –libs`
```

Or using *cmake* and *make*

Further information:

- http://opencv.willowgarage.com
  - Install guides
  - Documentation
  - FAQ
- OpenCV 2 Computer Vision – Application Programming Cookbook