

OpenCV

UNIVERSITÄT POTSDAM

SoCar SS 2011

Pierre Babeck

Was ist OpenCV?

- OpenCV eine freie c/c++ Bibliothek für “Computer Vision”
- OpenCV umfasst heute mehr als 500 Algorithmen
aus den Bereichen: Image Processing, Feature Detection,
Object Detection, Video Analysis, Camera Calibration and
Machine Learning
- Verfügbar für: Windows, Linux, Mac OS X
- Wrapper & Interfaces: c#, Python, Ruby, Matlab, Android,
Cuda, Xilinx

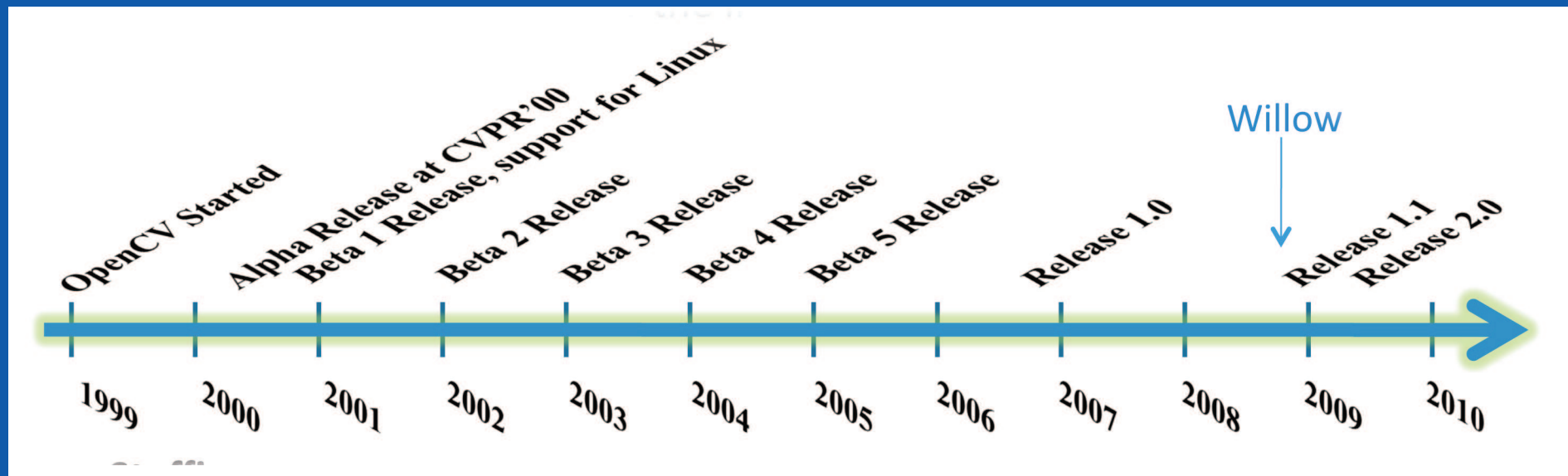
Was ist OpenCV?

The image is a collage of various OpenCV applications and concepts, organized into several categories represented by green boxes:

- General Image Processing Functions:** Shows various image processing results like edge detection and thresholding on different images.
- Image Pyramids:** Illustrates the concept of multi-scale image representations.
- Geometric descriptors:** Shows a face image with key points and a hand with labeled joints (A, B, C, H).
- Segmentation:** Displays image segmentation results, including a hand being segmented from a background.
- Camera calibration, Stereo, 3D:** Shows a checkerboard used for camera calibration and 3D scene reconstruction.
- Features:** Displays feature detection and matching between two images of a car.
- Utilities and Data Structures:** Shows a diagram of a data structure and a small application window.
- Tracking:** Shows a sequence of frames with a person's face being tracked and optical flow vectors.
- Machine Learning: Detection, Recognition:** Shows face detection and recognition results, including a group of people.
- Transforms:** Shows image transformation results like rotation and scaling.
- Fitting:** Shows a person's face with a fitted model and a diagram of a fitted shape.
- Matrix Math:** Shows a diagram of matrix operations.

Was ist OpenCV?

- ursprünglich von Intel entwickelt auf Grundlage der Image Processing Library (IPL) 1999
- heute unterstützt von Willow (Robot research) und V2.2



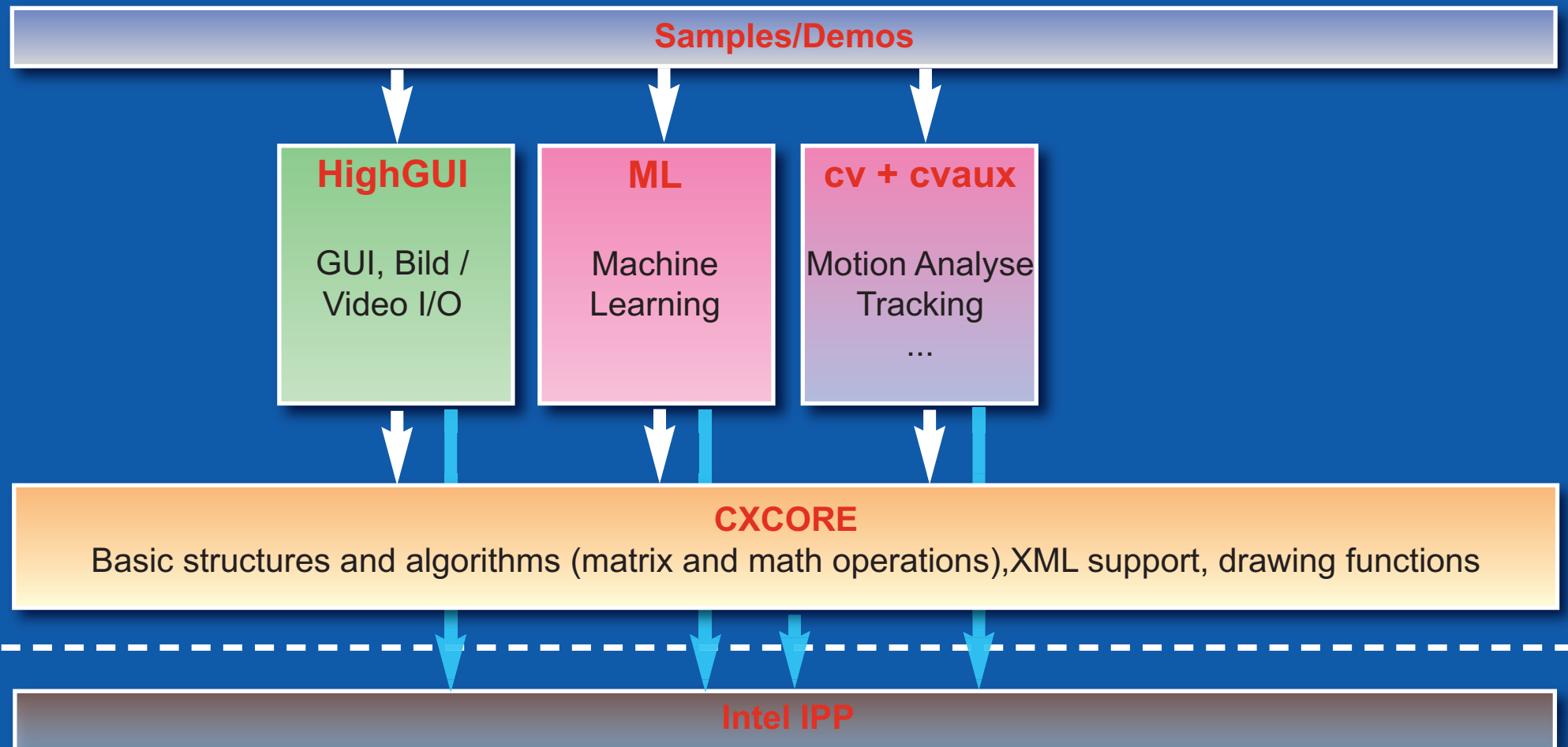
Möglichkeiten von OpenCV

- Echtzeit Bild- bzw. Videobe- und -verarbeitung
- multiprozessorfähig (OpenMP), Intel IPP fähig (bis V2.1),
- Einsetzbar in der Robotik, Überwachungssystem, Unterhaltungsindustrie (Kinect & OpenNI) ...



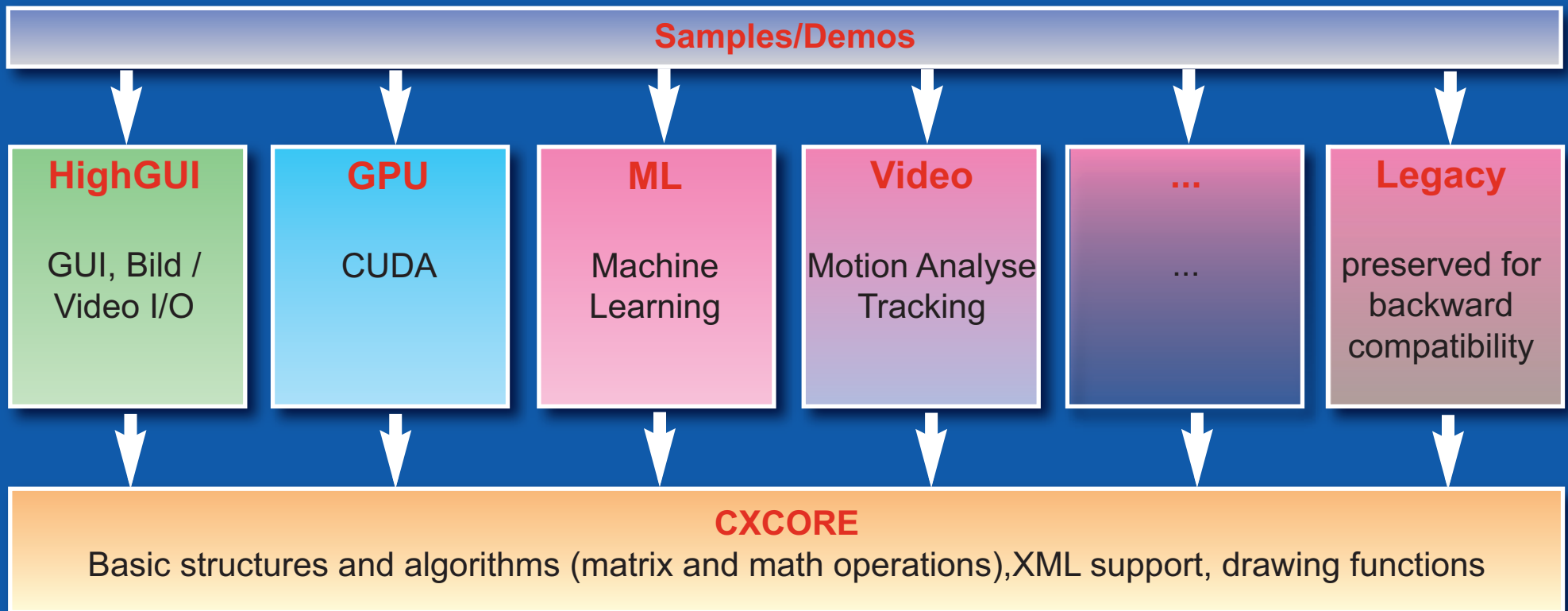
Möglichkeiten von OpenCV

Architektur OpenCV V2.1:



Möglichkeiten von OpenCV

Architektur OpenCV V2.2:



Möglichkeiten von OpenCV

- Kernfunktionen (cxcore):
 - einfache Operation auf Arrays und Matrizen
 - Arithmetische und Logic Op's (xor, div, min, max...)
 - discrete Fourier und Cosine transforms
 - XML I/O
 - Zeichenfunktion (2D)
 - Komplexe Datenstrukturen (z.B. dynamische Speicher)

Möglichkeiten von OpenCV

- GUI Funktionen (highgui - *ab V2.2 Qt*):
 - einfache und “leichte” Benutzeroberfläche
 - für Bild und Video I/O (*PCI, USB, Firewire, GigE*)
 - Trackbars und Mouse callback



Möglichkeiten von OpenCV

- Weitere Funktionen (*cvaux - bis V2.1*):
 - 3D-Vision, Gesichtserkennung, Tracking, Motion...
 - ab V2.2:

opencv_imgproc: image processing (filter, GaussianBlur, erode, resize...)

opencv_ml: statistical machine learning models

opencv_features2d: 2D feature detectors and descriptors

opencv_video: motion analysis and object tracking (optical flow, motion templates, background subtraction)

opencv_objdetect: object detection (face detectors, people detector...)

opencv_calib3d: camera calibration, stereo correspondence

...

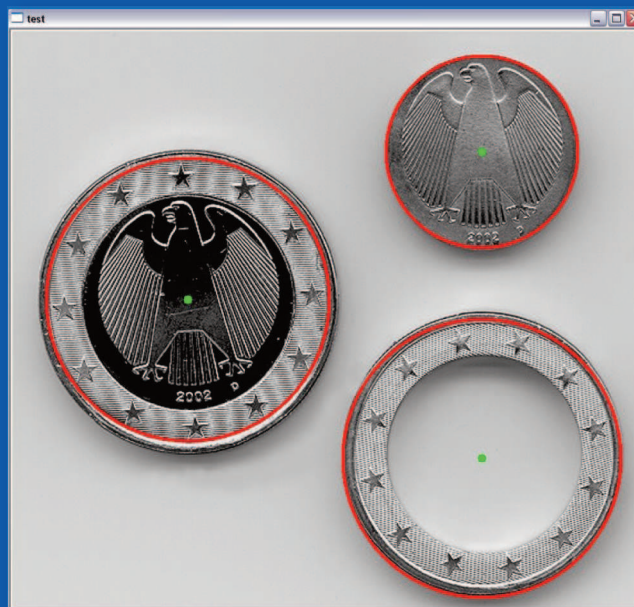
Möglichkeiten von OpenCV

- Was ist also alleine mit der OpenCV Bibliothek machbar?
 - Aufzeichnung und Wiedergabe von Video's
 - Laden und Speichern (bmp, tiff, jpg, XML, AVI)
 - Echtzeitmanipulation & Analyse
 - Stereo Imaging
 - kleine Bild- und Videobearbeitungsprogramme
(für Photoshop, Premiere Clone ist die GUI zu schwach)
- Was ist einzig mit OpenCV noch nicht machbar?
 - Ultraschall, Infrarot, mpeg, 3D-Rekonstruktion ...

Beispiele zu OpenCV

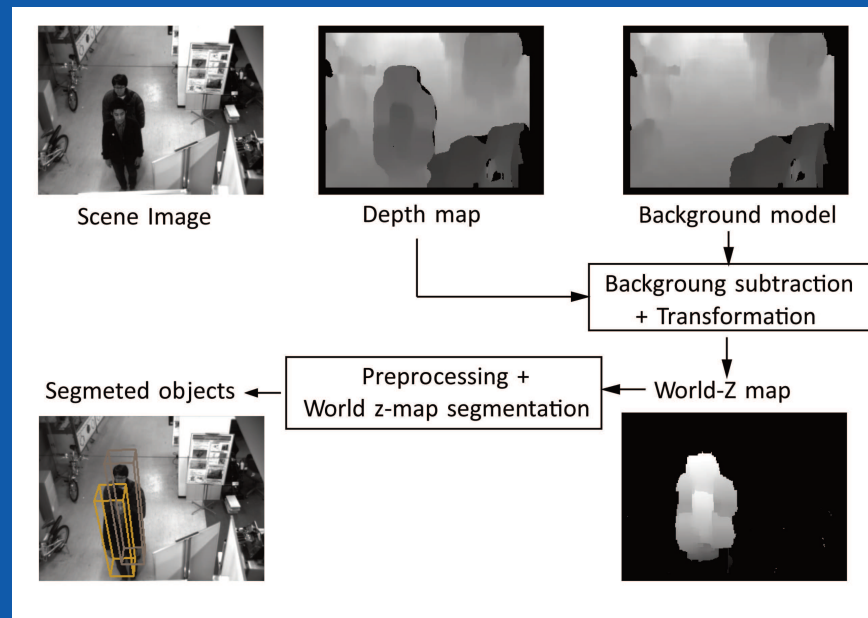
- Kreiserkennung:

`cvHoughCircles` (image, centers&radiuses)



Beispiele zu OpenCV

- Personenerkennung:



Beispiele zu OpenCV

- Urban Challenge - Bird View:

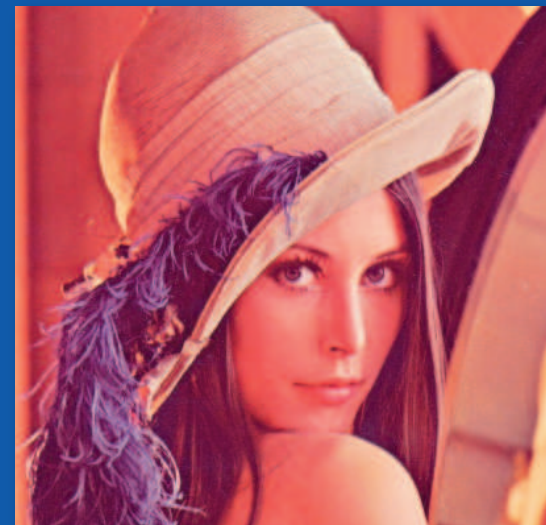


Arbeiten mit OpenCV

- Bild Laden:

```
#include <cv.h>
#include <highgui.h>

int main(int argc, char** argv)
{
    IplImage* image;
    if( argc != 2 ) return -1;
    image = cvLoadImage( argv[1] );
    if( !image ) return -1;
    cvNamedWindow( "Sample", 1 );
    cvShowImage( "Sample", image );
    cvWaitKey();
    return 0;
}
```



Arbeiten mit OpenCV

```
int main(int argc, char *argv[])  
{  
    IplImage* img = cvLoadImage( "Koala.jpg", 1 ); ← Bild laden  
  
    cvNamedWindow("Bild1", CV_WINDOW_AUTOSIZE); ← GUI Fenster erstellen  
    cvShowImage("Bild1", img); ← und anzeigen  
}
```


Arbeiten mit OpenCV

```
int main(int argc, char *argv[])
{
    IplImage* img = cvLoadImage( "Koala.jpg", 1 );

    cvNamedWindow("Bild1", CV_WINDOW_AUTOSIZE);
    cvShowImage("Bild1", img);

    IplImage* clone = cvCloneImage(img);
    cvSmooth( img, clone, CV_GAUSSIAN, 19, 19 );

    cvNamedWindow("Bild2", CV_WINDOW_AUTOSIZE);
    cvShowImage("Bild2", clone );
}
```

Bild kopieren

Bild Effekt - Blur

GUI Fenster erstellen
und anzeigen

Arbeiten mit OpenCV

```
int main(int argc, char *argv[])
{
    IplImage* img = cvLoadImage( "Koala.jpg", 1 );

    cvNamedWindow("Bild1", CV_WINDOW_AUTOSIZE);
    cvShowImage("Bild1", img);

    IplImage* clone = cvCloneImage(img);
    cvSmooth( img, clone, CV_GAUSSIAN, 19, 19 );

    cvNamedWindow("Bild2", CV_WINDOW_AUTOSIZE);
    cvShowImage("Bild2", clone );

    IplImage* Erode = cvCloneImage(img);
    cvErode( img, Erode, NULL , 10);

    cvNamedWindow("Bild3", CV_WINDOW_AUTOSIZE);
    cvShowImage("Bild3", Erode );
}
```

Bild kopieren

Bild Effekt - Erode

GUI Fenster erstellen
und anzeigen

Arbeiten mit OpenCV

```
int main(int argc, char *argv[])
{
    IplImage* img = cvLoadImage( "Koala.jpg", 1 );

    cvNamedWindow("Bild1", CV_WINDOW_AUTOSIZE);
    cvShowImage("Bild1", img);

    IplImage* clone = cvCloneImage(img);
    cvSmooth( img, clone, CV_GAUSSIAN, 19, 19 );

    cvNamedWindow("Bild2", CV_WINDOW_AUTOSIZE);
    cvShowImage("Bild2", clone );

    IplImage* Erode = cvCloneImage(img);
    cvErode( img, Erode, NULL , 10);

    cvNamedWindow("Bild3", CV_WINDOW_AUTOSIZE);
    cvShowImage("Bild3", Erode );

    cvWaitKey(0);
    cvReleaseImage( &img);
    cvReleaseImage( &clone);
    cvReleaseImage( &Erode);
    cvDestroyWindow("Bild1");
    cvDestroyWindow("Bild2");
    cvDestroyWindow("Bild3");}
```

← Speicher freigeben für
Bilder und GUI

Arbeiten mit OpenCV

Bild 2 - Blur

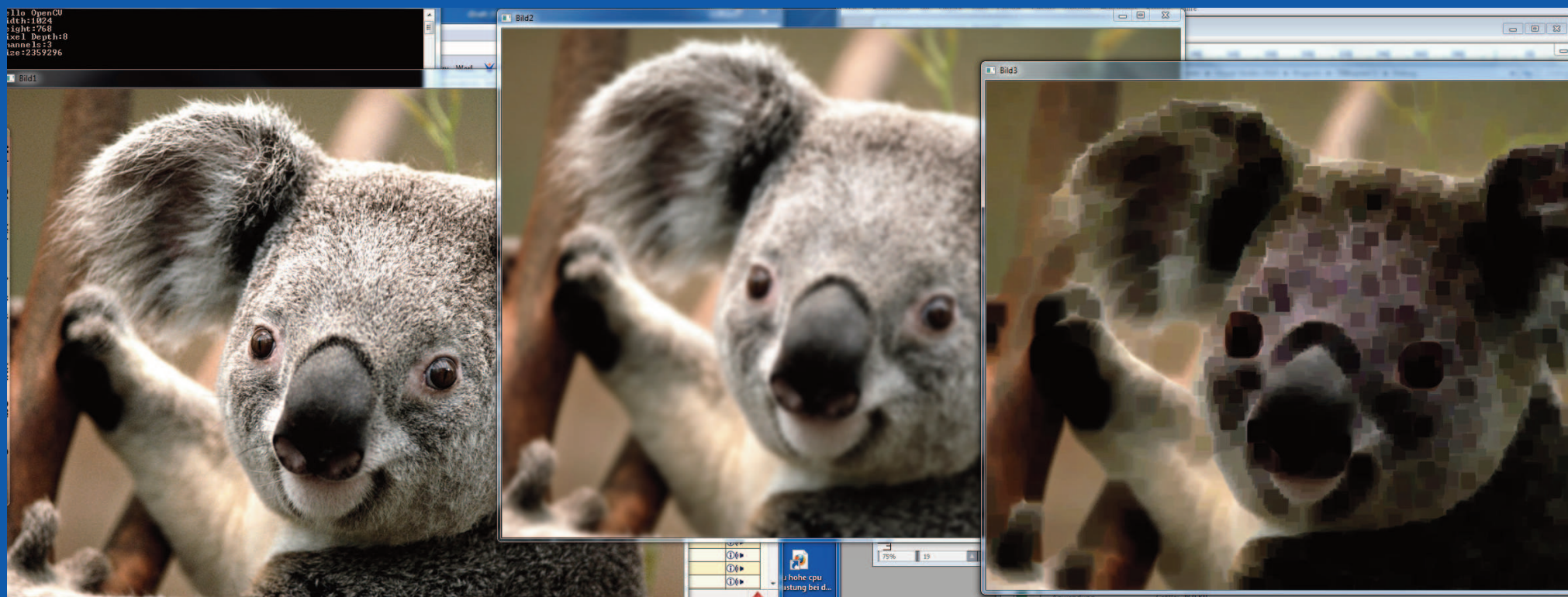


Bild 1

Bild 3 - Erode

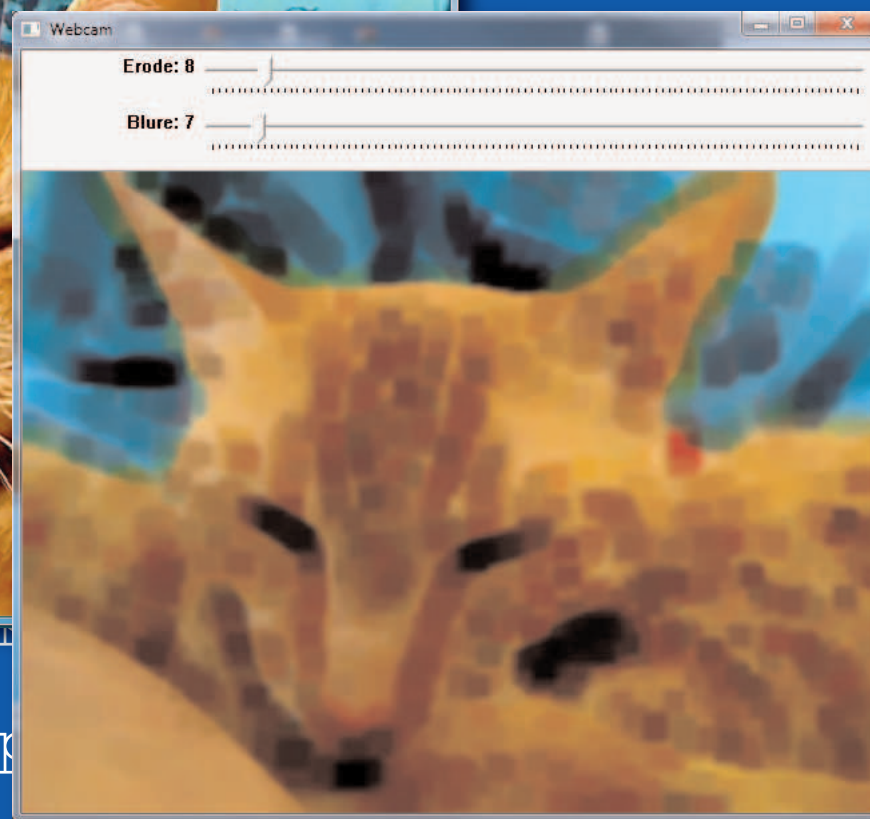
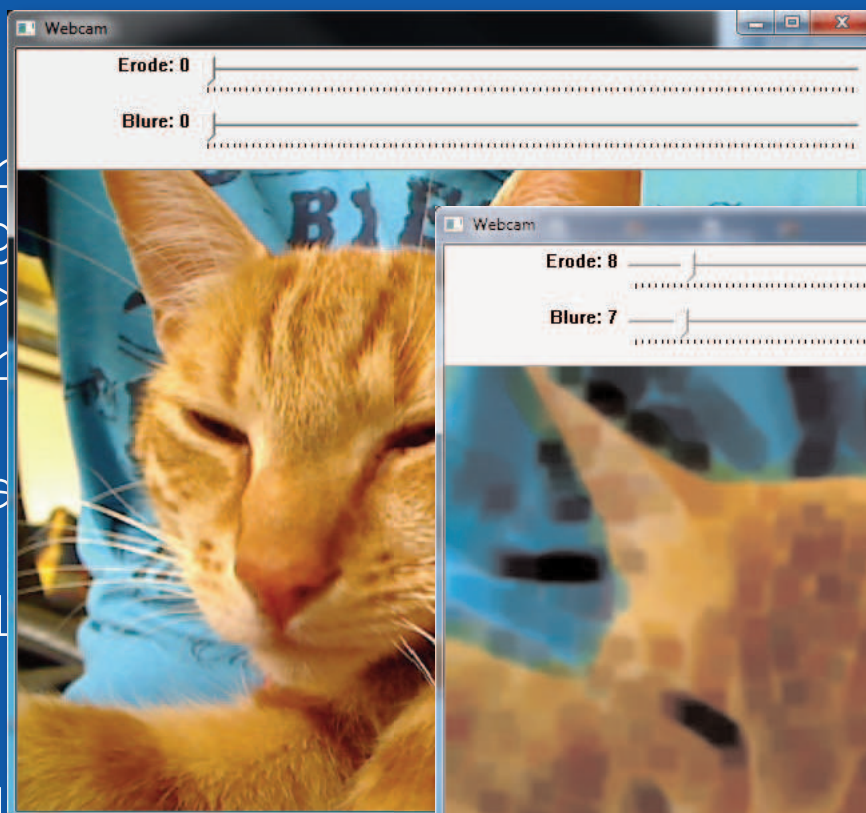
Arbeiten mit OpenCV

- Webcam:

```
#include <iostream>
#include <highgui.h>
#include <cv.h>
#include <cxcv.h>

int g_switch_val=0;
int erode=0;
int g_blure_val=0;
int blure=0;

void erode_callback( int position )
{ erode = position; }
void blure_callback( int position )
{ blure = position; }
```



Arbeiten mit OpenCV

```
int g_switch_value = 0;
int erode=0;
int g_blure_value = 0;
int blure=0;
```

Variablen für die GUI
Blur & Erode FX

```
void erode_callback( int position )
{   erode = position;}
void blure_callback( int position )
{   blure = position;}
```

Callback für Switches

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

Erstellung der Kamera

GUI - Window, Switches

```
    IplImage* frame;
    int key = 0;
    CvCapture* capture = cvCreateCameraCapture( 0 );
    cvNamedWindow("Webcam", CV_WINDOW_AUTOSIZE);
    cvCreateTrackbar( "Erode", "Webcam", &g_switch_value, 99, erode_callback );
    cvCreateTrackbar( "Blure", "Webcam", &g_blure_value, 99, blure_callback );
    while( key != 'q' ) {
        frame = cvQueryFrame( capture );
        if( !frame ) break;
        {
            cvErode( frame, frame, NULL , erode);
            cvSmooth( frame, frame, CV_BLUR, blure, blure );
        }
        cvShowImage( "Webcam", frame );
```

Schleife für wiederhol-
tes Processing

Quellen und Tutorials

- OpenCV (Main): <http://opencv.willowgarage.com>
- EmguCV (Wrapper c#): <http://www.emgu.com>

Book's

- OpenCV 2 (PacktLib): <http://www.laganiere.name/>
- Learning OpenCV (O'Reilly):
<http://www.willowgarage.com/pages/people/gary-bradski-senior-researcher>

Tutorials

- Xilinx XUP (Example): <http://code.google.com/p/liscv-and-find-tunnel-entrance/>
- Noah Kuntz: <http://www.pages.drexel.edu/~nk752/>

Fragen?