OpenCV Tutorial

Using OpenCV with Microsoft Visual Studio .net 2005

Lecturer: Amir hossein khalili
Sharif university of technology
March 2007

OpenCV

What is OpenCV?
(from the documentation)

OpenCV means Intel® Open Source Computer Vision Library. It is a collection of C functions and a few C++ classes that implement some popular Image Processing and Computer Vision algorithms.

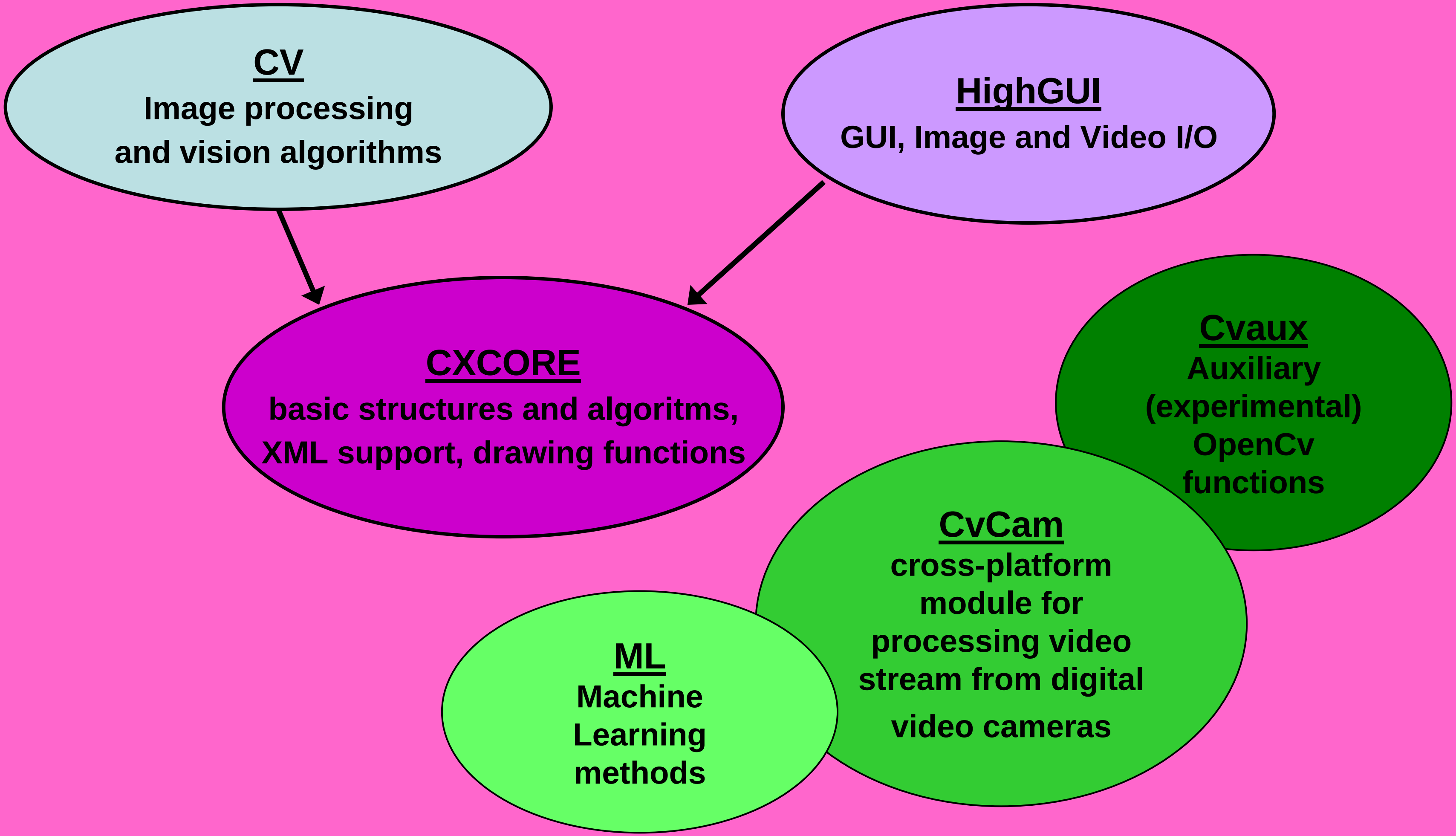
The key features
(from the documentation)

Cross-Platform API of C functions
FREE for commercial and non-commercial uses

What this means

You can take advantage of high speed implementations of functions commonly used in Computer Vision/Image Processing.

Overview of OpenCV



OpenCV

How to obtain the library

Available on Sourceforge

http://sourceforge.net/projects/opencvlibrary/

(Or use your favorite search engine)

How to install the library

(On Windows)

Download and Install the Executable

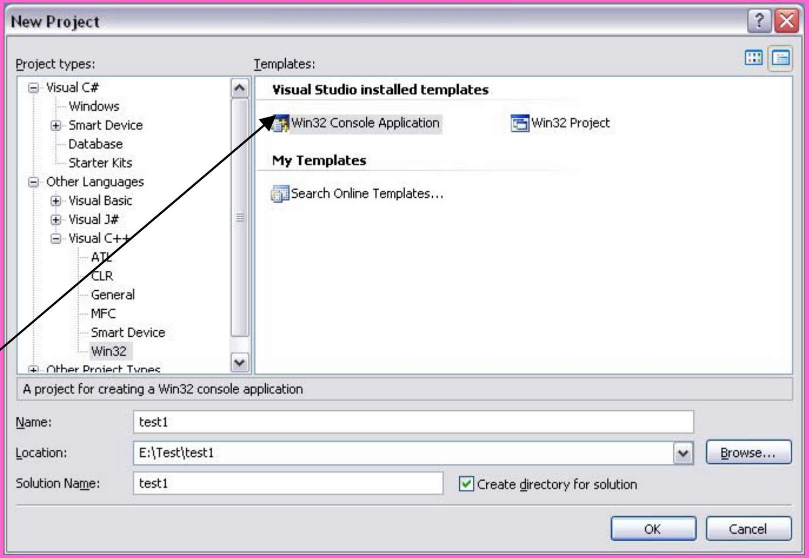
Configuring MSVS .net 2005

Creating the Project

A project is initially created by selecting:
File -> New -> Project

Create a “**Win32 Console Application**”

Make it an “**Empty Project**” by selecting the box under “Application Settings”



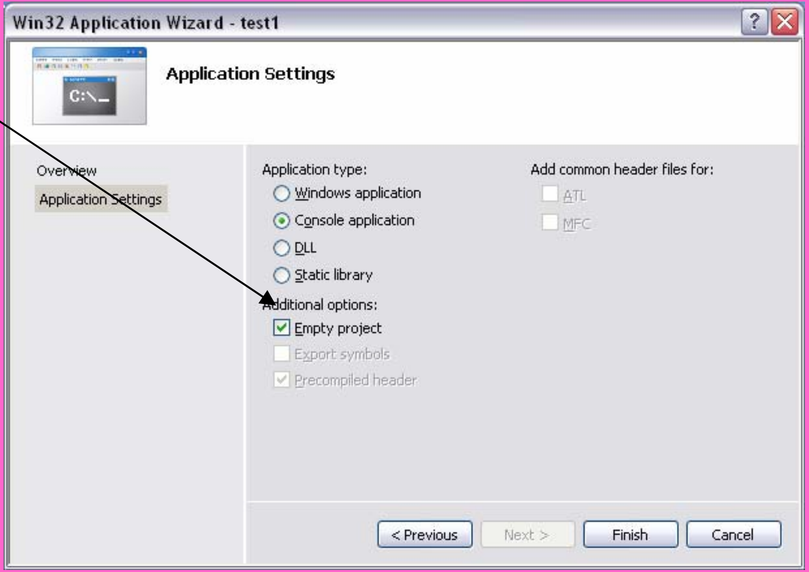
New Project

Project types: Visual C# (Windows, Smart Device, Database, Starter Kits), Other Languages (Visual Basic, Visual J#, Visual C++ (ATL, CLR, General, MFC, Smart Device, Win32)), Other Project Types

Templates: Visual Studio installed templates: Win32 Console Application, Win32 Project; My Templates: Search Online Templates...

A project for creating a Win32 console application

Name: test1
Location: E:\Test\test1
Solution Name: test1
[✓] Create directory for solution



Win32 Application Wizard - test1

Application Settings

Application type:
- Windows application
- [✓ Console application]
- DLL
- Static library

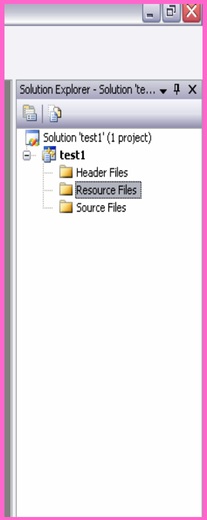
Add common header files for:
- [ ] ATL
- [ ] MFC

Additional options:
- [✓] Empty project
- [ ] Export symbols
- [✓] Precompiled header

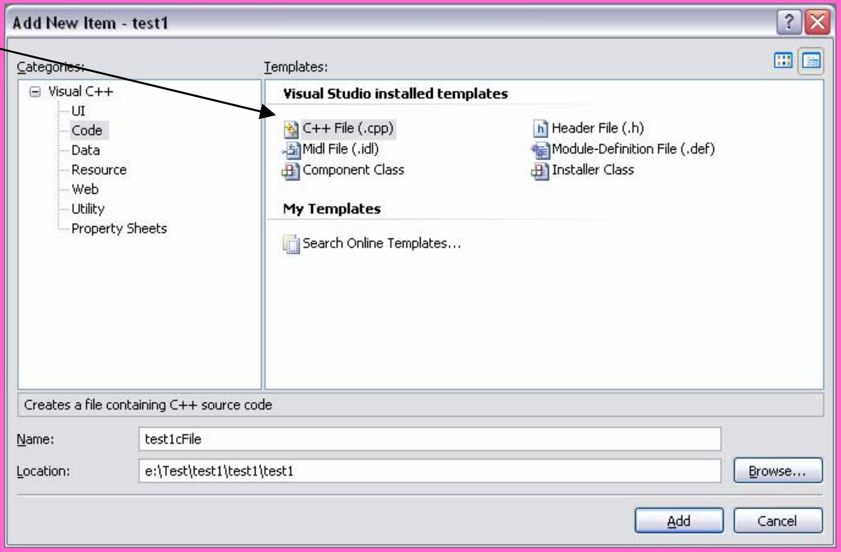
Configuring MSVS .net 2005

Create the First File

Right Click the "**Source Files**" Folder under the project name ("Test1" in this case) **Add -> Add new Item**


Solution Explorer - Solution 'te...
Solution 'test1' (1 project)
test1
Header Files
Resource Files
Source Files

Select "**C++ file(.cpp)**" and give it a name


Add New Item - test1

Categories:
- Visual C++
  - UI
  - Code
  - Data
  - Resource
  - Web
  - Utility
  - Property Sheets

Templates:

Visual Studio installed templates

C++ File (.cpp)	Header File (.h)
Midl File (.idl)	Module-Definition File (.def)
Component Class	Installer Class

My Templates

Search Online Templates...

Creates a file containing C++ source code

Name: test1cFile

Location: e:\Test\test1\test1\test1

Creating a file makes it possible to set "**Additional Include Directives**" in the C/C++ pane under the project properties.

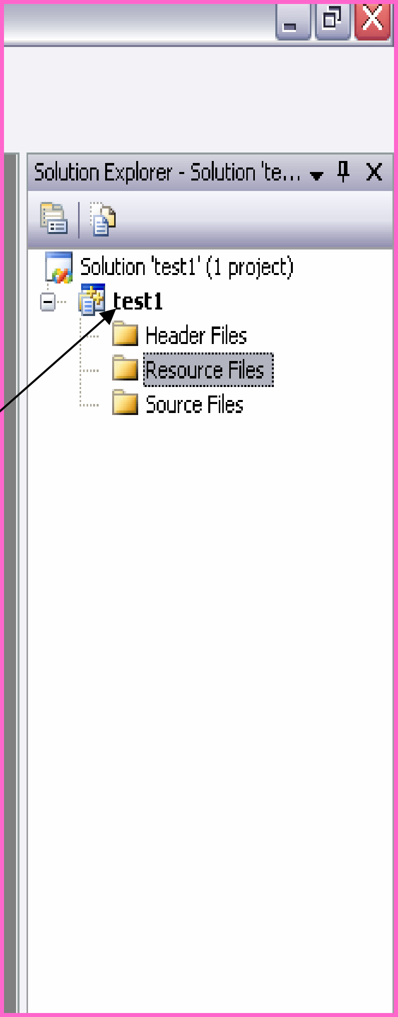
Configuring MSVS .net 2005

In order to build projects using OpenCV the required libraries and directives must be included in the project's properties

Open the Properties Pane

Right Click the name of the project and select **"Properties"**

("Test1" in this case)



Solution Explorer - Solution 'te...
Solution 'test1' (1 project)
test1
Header Files
Resource Files
Source Files

Configuring MSVS .net 2005

Set Additional Include Directives

Under the C/C++ tab select "**General**"

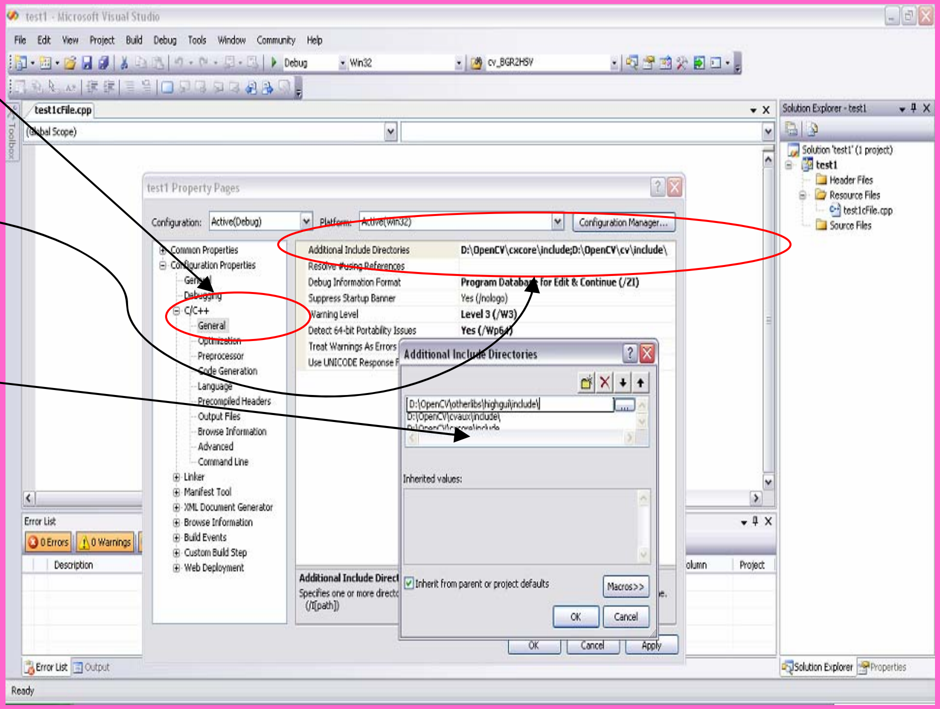
Select the "**Additional Include Directories**"

Add the full path to each of the folders which contain ".h" files required to use OpenCV

Be sure to include trailing "\"

Utilized Directives

D:\OpenCV\cvaux\include\
D:\OpenCV\cxcore\include\
D:\OpenCV\cv\include\
D:\OpenCV\otherlibs\highgui\
D:\OpenCV\otherlibs\cvcam\include\



test1 Property Pages

Configuration: Active(Debug) Platform: Active(Win32)

- Common Properties
- Configuration Properties
  - General
  - Debugging
  - C/C++
    - General
    - Optimization
    - Preprocessor
    - Code Generation
    - Language
    - Precompiled Headers
    - Output Files
    - Browse Information
    - Advanced
    - Command Line
  - Linker
  - Manifest Tool
  - XML Document Generator
  - Browse Information
  - Build Events
  - Custom Build Step
  - Web Deployment

Additional Include Directories	D:\OpenCV\cxcore\include;D:\OpenCV\cv\include\
Resolve #using References	
Debug Information Format	Program Database for Edit & Continue (/ZI)
Suppress Startup Banner	Yes (/nologo)
Warning Level	Level 3 (/W3)
Detect 64-bit Portability Issues	Yes (/Wp64)
Treat Warnings As Errors	
Use UNICODE Response F	

Additional Include Direct
Specifies one or more directo
(/I[path])

Additional Include Directories

D:\OpenCV\otherlibs\highgui\include\
D:\OpenCV\cvaux\include\
D:\OpenCV\cxcore\include

Inherited values:

[✓] Inherit from parent or project defaults

Configuring MSVS .net 2005

Utilized Directives

```
..\..\cvaux\include\
..\..\ cxcore\include\
..\..\cv\include\
..\..\otherlibs\highgui\
..\..\otherlibs\cvcam\include\
```

Configuring MSVS .net 2005

Set Additional Dependencies

Under the Linker tab select "**Input**"

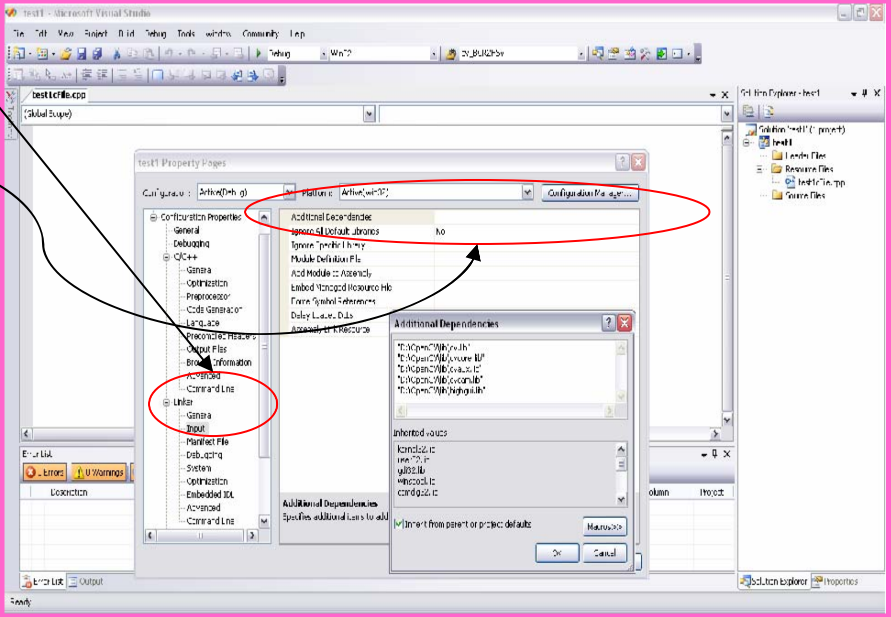
Select the "**Additional Dependencies**"

Add the full path to each of the ".lib" files required to use OpenCV

Be sure to keep the paths in quotes

Utilized Dependencies

"D:\OpenCV\lib\cv.lib"
"D:\OpenCV\lib\cvaux.lib"
"D:\OpenCV\lib\cxcore.lib"
"D:\OpenCV\lib\cvcam.lib"
"D:\OpenCV\lib\highgui.lib"



Additional Dependencies

"D:\OpenCV\lib\cv.lib"
"D:\OpenCV\lib\cxcore.lib"
"D:\OpenCV\lib\cvaux.lib"
"D:\OpenCV\lib\cvcam.lib"
"D:\OpenCV\lib\highgui.lib"

Configuring MSVS .net 2005

Utilized Dependencies

“..\..\lib\cv.lib“
“..\..\lib\cvaux.lib“
“..\..\lib\cxcore.lib“
“..\..\lib\cvcam.lib“
“..\..\lib\highgui.lib"

Testing MSVS .net 2005

Now that the environment is configured it would be a good idea to test it to make sure that a program will correctly build and run.

Testing the First Program

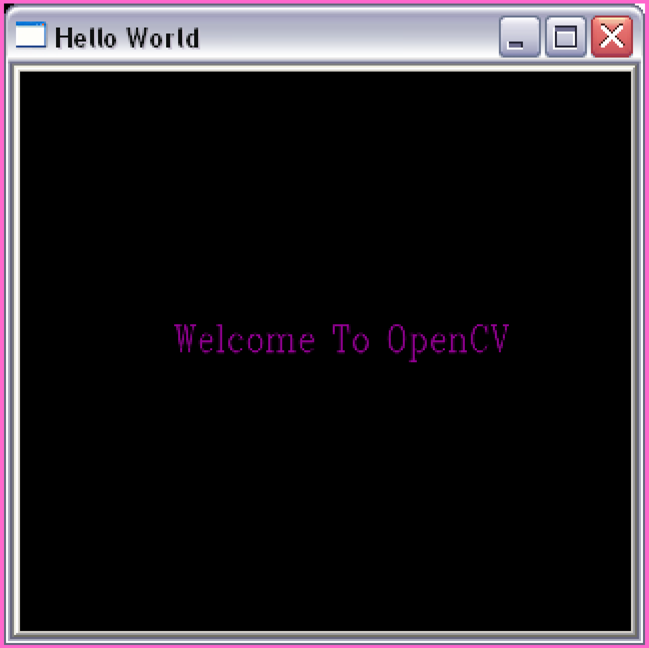
The enclosed code can be cut and pasted into the file created in the project space to test OpenCV

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

/*
			This will pop up a small box with "Welcome to OpenCV"
as the text.
			@author: Amir hossein khalili	a_khalili@ce.sharif.edu
				imitated from Gavin Page, gsp8334@cs.rit.edu
			@date: 1 March 2007
*/
int main( int argc, char** argv ) {
			//declare for the height and width of the image
			int height = 620;
			int width = 440;
			//specify the point to place the text
			CvPoint pt = cvPoint( height/4, width/2 );
			//Create an 8 bit, 3 plane image
			IplImage* hw = cvCreateImage(cvSize(height, width), 8,
3);
			//Clearing the Image
			cvSet(hw,cvScalar(0,0,0));
			//initialize the font
			CvFont font;
			cvInitFont( &font, CV_FONT_HERSHEY_COMPLEX,
						1.0, 1.0, 0, 1, CV_AA);
			//place the text on the image using the font
			cvPutText(hw, "Welcome To OpenCV", pt, &font,
CV_RGB(150, 0, 150) );
			//create the window container
			cvNamedWindow("Hello World", 0);
			//display the image in the container
			cvShowImage("Hello World", hw);
			//hold the output windows
			cvWaitKey(0);
			return 0;
}
```

Testing MSVS .net 2005

Now that the environment is configured it would be a good idea to test it to make sure that a program will correctly build and run.



```
Welcome To OpenCV
```

Testing the First Program

The enclosed code can be cut and pasted into the file created in the project space to test OpenCV

At this point you should have a working OpenCV project. If the program is not working you should go back and carefully recheck the steps.

From here you can explore the documentation to review the functions available.

There are also a number of tutorials on the web including:
http://www.site.uottawa.ca/~laganier/tutorial/opencv+directshow
or you can just search for them

You should also join the OpenCV Community located at:
http://groups.yahoo.com/group/OpenCV/
As of today there are >15000 members available to answer questions. There is also a searchable message board where you can look up previous queries.

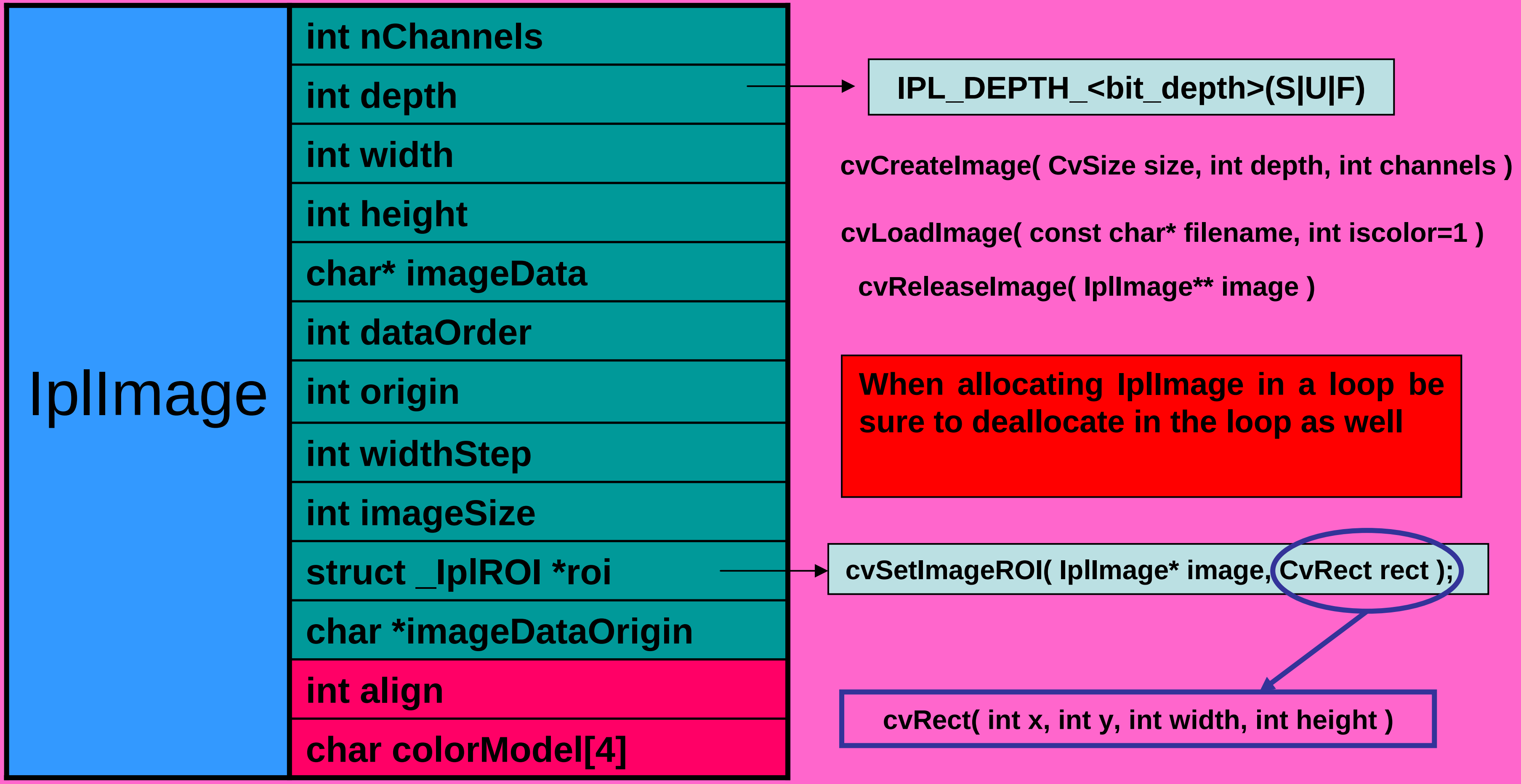
Memory management

- **Why is Managing OpenCV objects Important?**
  - Video, 30 frames per second
  - Each frame is an image
  - Images are arrays of pixels
  - A 640x480 image is 307,200 pixels
  - These must be represented in memory
  - How much memory does your machine have?

void cvResize(const CvArr* src, CvArr* dst, int interpolation)

The metatype CvArr* is used *only* as a function parameter to specify that the function accepts arrays of more than a single type, for example IplImage*, CvMat* or even CvSeq*. The particular array type is determined at runtime by analyzing the first 4 bytes of the header.

Image data structure



IplImage

```
int nChannels
int depth
int width
int height
char* imageData
int dataOrder
int origin
int widthStep
int imageSize
struct _IplROI *roi
char *imageDataOrigin
int align
char colorModel[4]
```

```
IPL_DEPTH_<bit_depth>(S|U|F)
```

```
cvCreateImage( CvSize size, int depth, int channels )
cvLoadImage( const char* filename, int iscolor=1 )
cvReleaseImage( IplImage** image )
```

When allocating IplImage in a loop be sure to deallocate in the loop as well

```
cvSetImageROI( IplImage* image, CvRect rect );
cvRect( int x, int y, int width, int height )
```

cvLoadImage

Supportted formats:

- Windows bitmaps - BMP, DIB;
- JPEG files - JPEG, JPG, JPE;
- Portable Network Graphics - PNG;
- Portable image format - PBM, PGM, PPM;
- Sun rasters - SR, RAS
- TIFF files - TIFF, TIF.

Functions

Features	1st & 2nd Image Derivatives. Lines: Canny, Hough. Corners: Finding, tracking.
Image Statistics	In region of interest: Count, Mean, STD, Min, Max, Norm, Moments, Hu Moments.
Image Pyramids	Power of 2. Color/texture segmentation.
Morphology	Erode, dilate, open, close. Gradient, top-hat, black-hat.
Distance Transform	Distance Transform
Thresholding	Binary, inverse binary, truncated, to zero, to zero inverse.
Flood Fill	4 and 8 connected
Histogram (recognition	Manipulation, comparison, backprojection
Eigen Objects	Calc Cov Matrix, Calc Eigen objects, decomp. coeffs. Decomposition and projection.

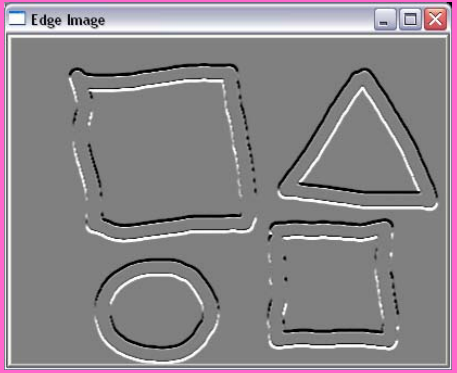
Sample Program

- Extracting edges with sobel

void cvSobel(const CvArr* src, CvArr* dst, int xorder, int yorder, int aperture_size=3);

```
#include "cv.h"
#include "highgui.h"
int main( int argc, char** argv )
{
            char* fileAddress="pic.png";
            IplImage* orginalImage = cvLoadImage(fileAddress,0);
            cvNamedWindow("Orginal Image");
            cvShowImage("Orginal Image", orginalImage);
            IplImage* edgeImage =
             cvCreateImage(cvGetSize(orginalImage),IPL_DEPTH_16S,1);
            cvSobel(orginalImage,edgeImage,0,1);

            cvNamedWindow("Edge Image");
            cvShowImage("Edge Image", absEdgeImage);
            cvWaitKey(0);
            cvReleaseImage(&orginalImage);
            cvReleaseImage(&edgeImage);
            cvDestroyWindow("orginal Image");
            cvDestroyWindow("Edge Image");
}
```



Accessing image elements

- **Assume that you need to access the *K*-th channel of the pixel at the *i*-row and *j*-th column. The row index is in the range *[0-height-1]* . The column index is in the range *[0-width-1]* . The channel index is in the range *[0-nchannel-1]* .**

Indirect access

```
CvScalar s;
s=cvGet2D(img,i,j);
Int value = s.val[k];
s.val[k]=111;
cvSet2D(img,i,j,s);
```

Onother direct access

```
int height = img->height;
int width = img->width;
int step = img->widthStep/sizeof(float);
int channels = img->nChannels;
TYPE * data = (TYPE *)img->imageData;
data[i*step+j*channels+k] = 111;
```

Direct access

```
Value =((TYPE *)(img->imageData + i*img->widthStep))[j*img->nChannels + 0]=111
```

Some other useful data structures

CvMat

OpenCV uses the CvMat* as its general purpose matrix structure. It is managed in an equivalent style toIplImage*

```
cvCreateMat( int rows, int cols, int type );
cvReleaseMat( CvMat** mat );
```

CvMatND

Multi Dimentional version of CvMat

CvSparseMat

SPARSE N-dimensional array

CvScalar

4D vector :double val[4]

```
CvScalar s = cvScalar(double val0, double val1,
double val2, double val3)
```

void cvSet(CvArr* arr, CvScalar value, const CvArr* mask=NULL)

Some other useful data structures

CvSeq

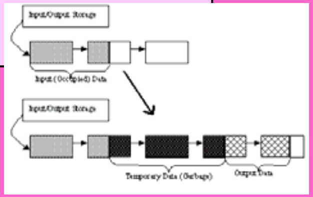
Growable 1-D array, Link list, Queue, Stack

```
cvCreateSeq( int seq_flags, int header_size,
int elem_size, CvMemStorage* storage );
```

CvMemStorage

Growing memory storage

```
cvCreateMemStorage( int block_size=0 );
cvClearMemStorage( CvMemStorage* storage )
```



Points

```
CvPoint p = cvPoint(int x, int y);
CvPoint2D32f p =
     cvPoint2D32f(float x, float y);
CvPoint3D32f p =
   cvPoint3D32f(float x, float y, float z);
```

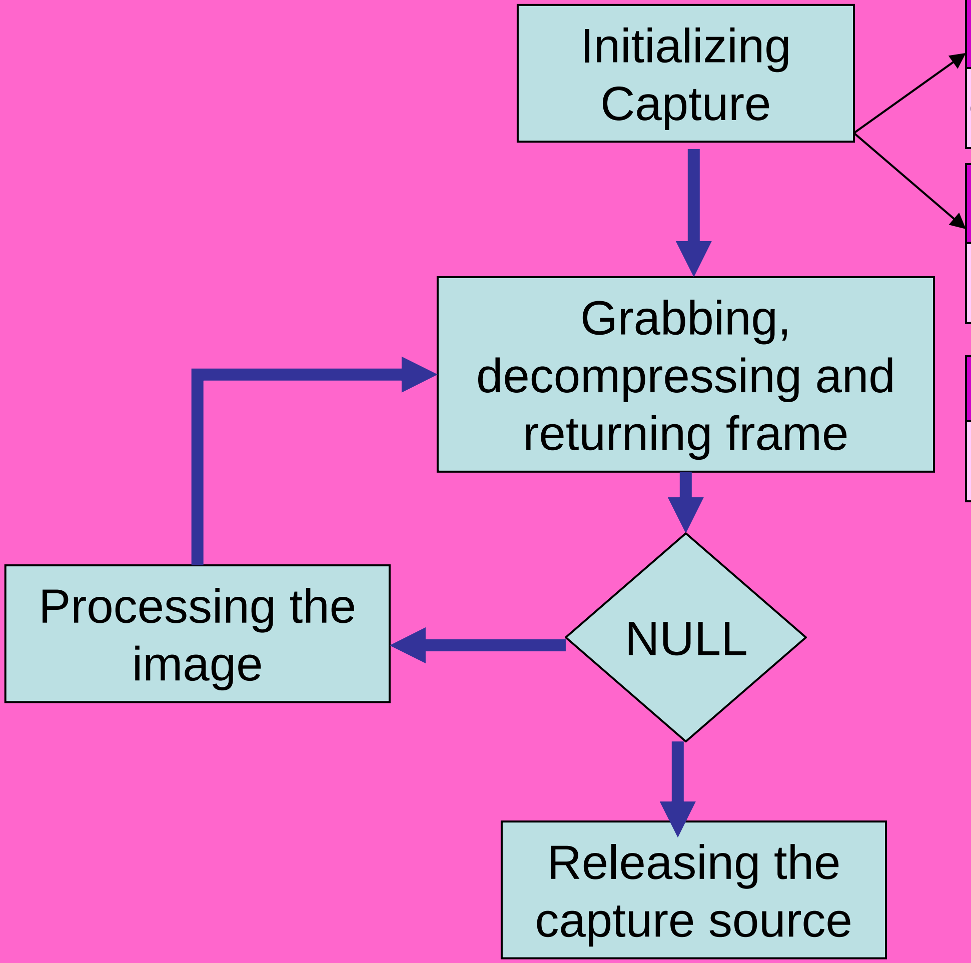
Rectangular dimensions

```
CvSize r = cvSize(int width, int height);
CvSize2D32f r =
cvSize2D32f(float width, float height);
```

Rectangular dimensions with offset

```
CvRect r = cvRect(int x, int y, int width,
int height);
```

Working with video sequences


Initializing Capture
Grabbing, decompressing and returning frame
NULL
Processing the image
Releasing the capture source

From a camera

```
CvCapture* capture = cvCaptureFromCAM(0);
```

From a file

```
CvCapture* capture = cvCaptureFromAVI("infile.avi");
```

cvQueryFrame

```
IplImage* cvQueryFrame( CvCapture* capture );
```

The returned image should not be released or modified by user

cvReleaseCapture

the image captured by the device is allocated /released by the capture function. There is no need to release it explicitly

```
cvReleaseCapture(&capture);
```

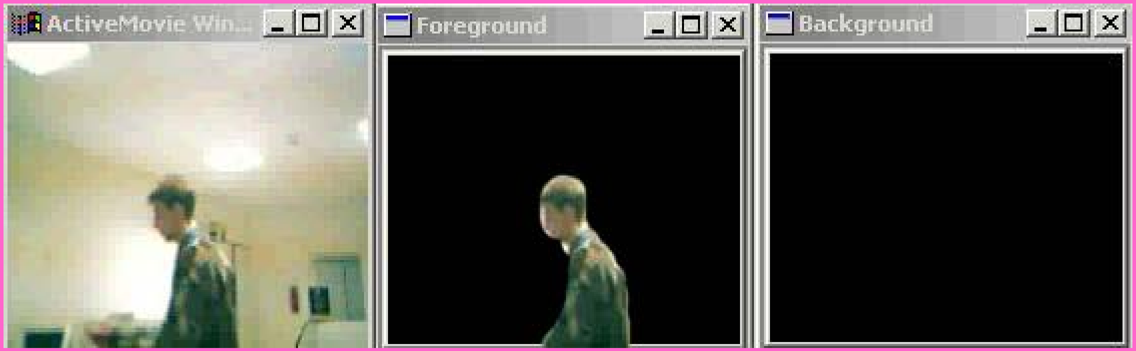
Motion Analysis and Object Tracking

- Background subtraction
- Motion templates
- Optical flow
- Active contours
- Estimators

Background subtraction

- describes basic functions that enable building statistical model of background for its further subtraction.
- Background statistics functions:
  - ✓ Average
  - ✓ Standard deviation
  - ✓ Running average

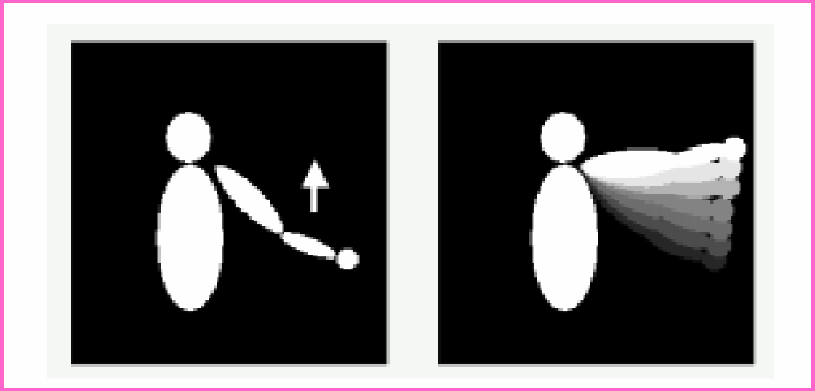
$$\mu_{ij}^{t} = \alpha \cdot I_{ij}^{t} + (1-\alpha) \cdot \mu_{ij}^{t-1},\ 0 \le \alpha \le 1$$



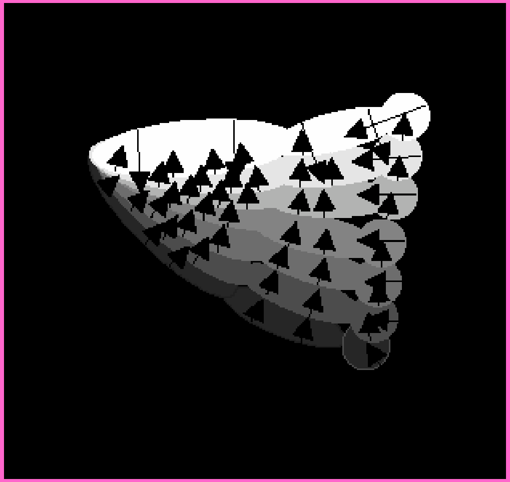
Motion templates

- **To generate motion template images that can be used to rapidly determine where a motion occurred, how it occurred, and in which direction it occurred.**
- Object silhouette
- Motion history images
- Motion history gradients
- Motion segmentation algorithm

silhouette MHI



MHG



Optical Flow

- Block matching technique
- Horn & Schunck technique
- Lucas & Kanade technique
- Pyramidal LK algorithm
- 6DOF (6 degree of freedom) algorithm

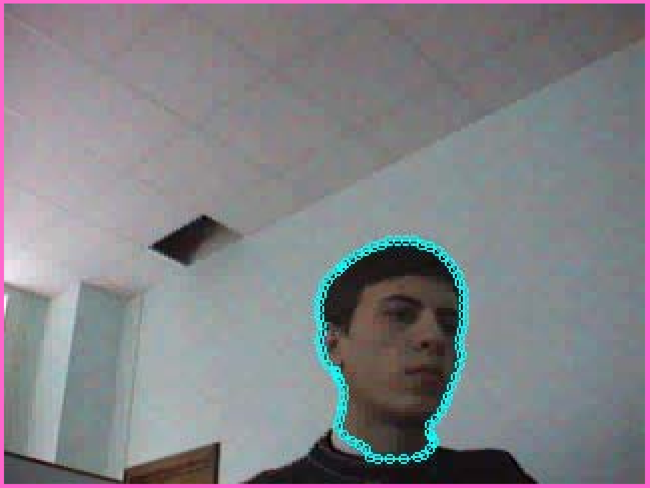
Active Contours

- Snake energy: $E = E_{\text{int}} + E_{ext}$
- Internal energy: $E_{\text{int}} = E_{cont} + E_{curv}$
- External energy: $E_{ext} = E_{img} + E_{con}$
- Two external energy types:

$$E_{img} = -I,$$

$$E_{img} = -\|grad(I)\|,$$

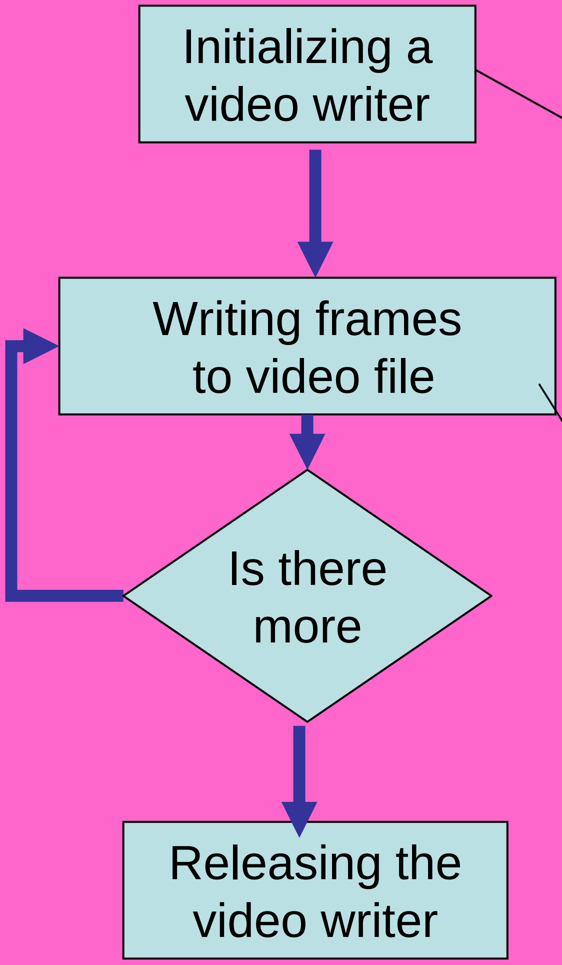
$$E = \alpha \cdot E_{cont} + \beta \cdot E_{curv} + \gamma \cdot E_{img} \Rightarrow \min$$



Estimators

- Kalman filter
- ConDensation filter

Saving a video file


Initializing a video writer
Writing frames to video file
Is there more
Releasing the video writer

```
CvVideoWriter *writer = 0;
 int isColor = 1;
int fps = 25; // or 30
int frameW = 640; // 744 for firewire cameras
int frameH = 480; // 480 for firewire cameras
writer=cvCreateVideoWriter("out.avi",
                CV_FOURCC('P','I','M','1'),
               fps,cvSize(frameW,frameH),isColor);
```

```
IplImage* img = 0;
int nFrames = 50;
for(i=0;i<nFrames;i++)
{
    Img=cvQueryFrame(capture);
     cvWriteFrame(writer,img);
}
```

```
cvReleaseVideoWriter(&writer);
```

Possible Codecs for saving

Codec	fourcc
MPEG-1	CV_FOURCC('P','I','M','1')
motion-jpeg	CV_FOURCC('M','J','P','G')
MPEG-4.2	CV_FOURCC('M', 'P', '4', '2')
MPEG-4.3	CV_FOURCC('D', 'I', 'V', '3')
MPEG-4	CV_FOURCC('D', 'I', 'V', 'X')
H263	CV_FOURCC('I', '2', '6', '3')
FLV1	CV_FOURCC('F', 'L', 'V', '1')
A codec code of -1 will open a codec selection window (in windows).	

Thank You! Questions?