Writing Video Applications

Dr. Lisa Spencer

Last updated 6/23/06

Microsoft Video APIs

- Video For Windows (vfw)
  - Dates back to Windows 3.1
  - No further development
  - Doesn’t support DV compression
- DirectShow
  - Component of Direct X
  - Preferred method for new development

DirectX Components

- Graphics: Direct3D
- Audio: DirectSound and DirectMusic
- Joysticks and force-feedback: DirectInput
- Networked games: DirectPlay
- Multimedia: DirectShow
- DirectSetup

Basic DirectShow

The minimum you need to know to use DirectShow

Getting DirectX

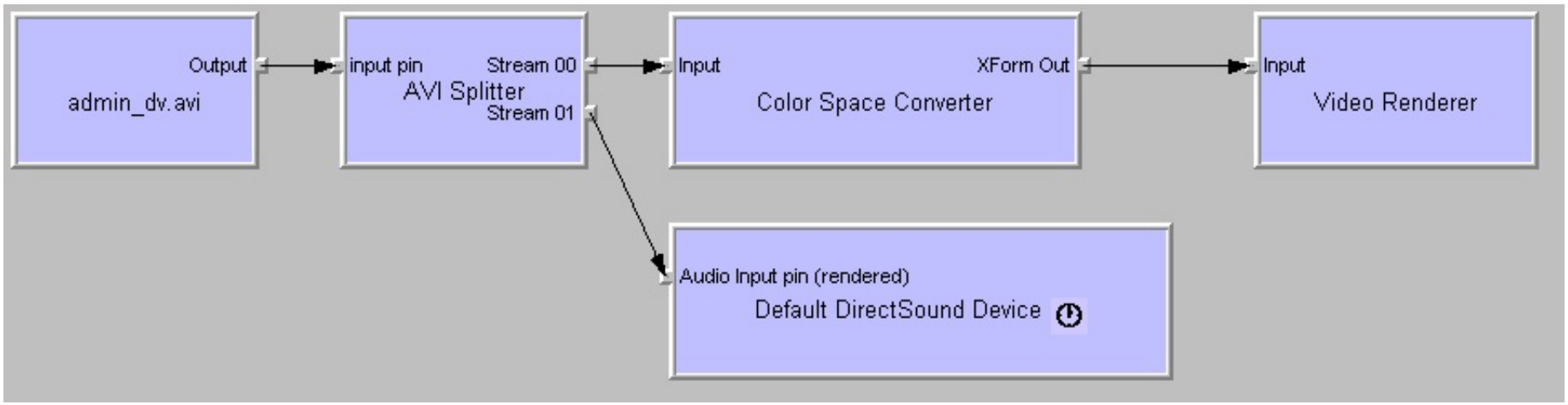
- Uninstall any previous DX SDK versions
- Download the DirectX 9 Software Development kit
  - Go to http://msdn.microsoft.com/directx/directxdownloads.
  - If you are using VC++6.0, dig to find the Summer 2003 update, which still included .dsw files.
  - Download the SDK Developer Kit (June06 version is >400 Mb).
  - Run the downloaded .exe file to unzip and install.
  - If “GraphEdit” isn’t listed in the DX utilities on the Start menu, download and install the most recent “Extras” too.

Set up DirectX for VC++

- Build C:\(DxInstallDir)\samples\C++\DirectShow\BaseClasses. (Load .sln or .dsw and build for Debug and Release libraries)
- In Developer Studio,
  - Add include dir C:\(DxInstallDir)\Samples\C++\DirectShow\Baseclasses so it can find streams.h.
    - VC++6: Tools->Options->Directories->Include Files
    - VC++.NET: Tools->Options->Projects->VC++ Directories
  - Make sure C:\(DxInstallDir)\include is first in the list.

DirectShow

- Based on the “Filter Graph”
- Modules that handle a single function can be mixed and matched.
- This example renders an AVI file.



Filter Types

- Source Filters (output pins only)
  - Capture
  - Video/Audio File
- Transform Filters (input and output pins)
  - Compression/Decompression
  - Format Conversion
- Rendering Filters (input pins only)
  - Video
  - Sound
  - File

How to use DirectShow

- Filter Graph Editor
  - Easy to experiment
  - Programs->Microsoft DirectX 9.0 SDK Update->DirectX Utilities->GraphEdit Application
    - If you installed “extras”, you may have to add this link yourself.
  - Easier to run
  - Better GUI
  - Sample menu at All Programs->Microsoft DirectX SDK->DirectX Sample Browser

Using GraphEdit

- Programs->Microsoft DirectX SDK ->DirectX Utilities->GraphEdit
  - Make a shortcut to it on your desktop for easy access
- To render an AVI file: “File->Render Media File”, then “Play”
- Add filters with Graph->Insert Filters
- To render live video:
  - From filter list, under “Video Capture Sources”, insert filter for camera
  - Right-click on capture pin and select “Render pin”
  - Press “Play”

Using GraphEdit

- You can add filters in the middle for effects, or change the output to a file writer to save the results.
- You can see the graph that an application has built:
  File->Connect To Remote Graph
- If your application is not listed, add the code from the topic *Loading a Graph From an External Process* in the DirectShow documentation (included with “extras”).

Writing a Transform Filter

For VC++ .NET:

1. Unzip http://www.cs.ucf.edu/~lspencer/MYFILTER.zip. (Code was created using wizard on next page.)
2. Rename the directory and all files containing “MYFILTER”, changing “MYFILTER” to your filter name.
3. Replace “MYFILTER” with your filter name inside all the files (in Visual Studio: Edit->Find and Replace->Replace in Files, for *.*, without opening the project.)
4. Load the renamed .dsw file (will be converted to .sln)
5. Fix the DXSDK path to strmbase.lib for both debug and release (Project->Properties->Linker->Input)
6. Update DEFINE_GUID lines in iMYFILTER.h with new strings generated using Tools->Create GUID.

Writing a Transform Filter

For VC C++ 6.0:

1. Use AppWizard from http://www.ifp.uiuc.edu/~chenyq/research/Utils/DShowFilterWiz/DShowFilterWiz.html. We named ours “myfilter”. Choose 24-bit color, so the image will be 3 channels.
2. In “Additional include directories” change “Multimedia” to “C++”. (The location of streams.h changed in DX9.)

Writing a Transform Filter

After creating the project by either method:

1. Put your code in Transform() in myfilter.cpp
2. Add controls to Property dialog.
3. Add variables to class myfilterParams in imyfilter.h.
4. Search for “param1” and add code for new variables at each occurrence.
5. Install the filter with regsvr32 “C:/…/myfilter.ax”.

Example 1: Filter

Problem: Write a filter that converts a color image to grayscale by copying the green component to red and blue.

1. Create a new filter project named “filter1” by one of the methods described previously.
2. In filter1.cpp, in Transform(), change the *for* loop to:

```
for (iPixel=0; iPixel<numPixels;
     iPixel++, prgb+=iPixelSize) {
    *(prgb  ) = (prgb + 1);     // B channel
    *(prgb+2) = (prgb + 1);     // R channel
}
```

3. Compile it for Release.
4. Start->Run->regsvr32 “C:\…\Release\filter1.ax”

Using a Transform Filter

1. Render a video source with GraphEdit
2. Add the new filter with Graph->Insert Filters ->DirectShow Filters->myfilter Filter
3. Delete the video renderer, then connect the last output pin to the input pin of myfilter Filter
4. Right-click on the output pin of myfilter Filter and choose “Render pin”
5. Right-click on myfilter Filter and choose Filter Properties to set your parameters
6. Run the graph

Regsvr32 Notes

- Regsvr32 is used to register a filter, COM object, ActiveX control, etc.
- Extension for filter is .ax
- File will be in /Debug or /Release
- To register:
  regsvr32 <full path of target>
- To unregister:
  regsvr32 /u <full path of target>
- Only need to register once – can recompile without re-registering.
- Errors may occur if libraries have been updated, so try recompiling old filters if there are problems.

Writing an Application

For VC++.NET,

1. Unzip http://www.cs.ucf.edu/~lspencer/MYPROJECT.zip. (Code was created using wizard on next page.)
2. Rename all files (including res/*) containing “MYPROJECT”, changing “MYPROJECT” to your project name.
3. Replace “MYPROJECT” with your project name inside all the files (in Visual Studio: Edit->Find and Replace->Replace in Files without opening the project.)
4. Load the renamed .dsw file (will be converted to .sln)
5. Newer DX SDKs require changing “wsprintfW(wsz,” to “StringCchPrintfW(wsz, 128” on line 909 of GraphBase.cpp.

Writing an Application

For VC++6.0,

1. Use the AppWizard from http://www.ifp.uiuc.edu/~chenyq/research/Utils/DxVideoAppWiz/DxVideoAppWiz.html. Choose 24-bit color, so the image will be 3 channels.
2. In “Additional include directories” change “Multimedia” to “C++”. (The location of streams.h changed in DX9.)
3. The link warning can be avoided by ignoring library libcmtd.lib in Debug and libcmt.lib in Release.

Writing an Application

- After creating a new project by either method,
  - Put your code in CxxxDoc::SampleCB().

Writing an Application

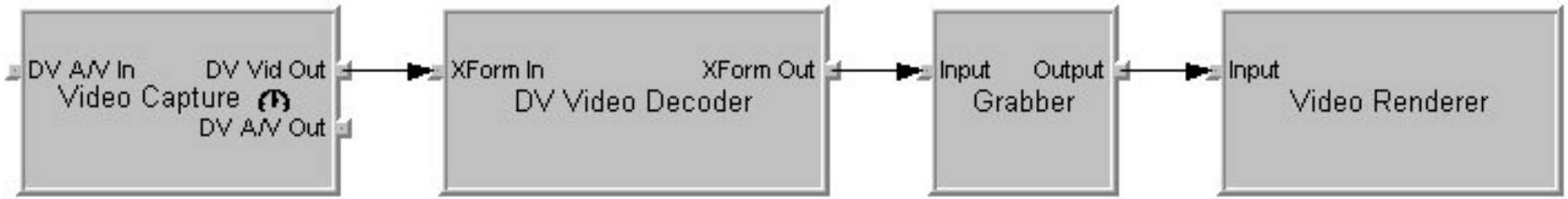
To run movies asynchronously (don’t skip frames; may run faster than original movie), add the following code to RenderFile() in GraphBase.cpp before starting the graph:

```
IMediaFilter *pMediaFilter = 0;
M_pGraphBuilder->QueryInterface(IID_IMediaFilter,
    (void**)&pMediaFilter;
pMediaFilter->SetSyncSource(NULL);
pMediaFilter->Release;
```

- This won’t affect live video.

Application Filter Graph

Here's what the filter graph looks like for the application, using a DV video camera:



The Grabber calls a user function (SampleCB) when it receives a new frame.

Example 2: Application

Problem: Write an application that finds the greenest pixel, and puts an ‘X’ over it.

1. Follow the previous instructions for creating a new video application.
2. Edit xxxDoc::SampleCB (code is on next slides)
3. Run the application. The “X” should follow a green laser pointer.

Example 2: Unchanged code

```
STDMETHODIMP CApp2Doc::SampleCB(double SampleTime,
IMediaSample *pSample)
{
    // get current media type
    VIDEOINFOHEADER *pvi = (VIDEOINFOHEADER *)
                                    m_mediaType.pbFormat;

    BYTE *pData;            // Pointer to the actual image buffer
    long lDataLen;          // Holds length of any given sample
    int iPixel;             // Used to loop through the image pixels
    pSample->GetPointer(&pData);
    lDataLen = pSample->GetSize();

    // Get the image properties from the BITMAPINFOHEADER
    int iPixelSize = pvi->bmiHeader.biBitCount / 8;
    int cxImage    = pvi->bmiHeader.biWidth;
    int cyImage    = pvi->bmiHeader.biHeight;
    int cbImage    = cyImage * cxImage * iPixelSize;
    int numPixels  = cxImage * cyImage;
```

Example 2: Find green pixel

```
// Find greenest pixel (note: this demonstrates how to access
// the pixels, not the best way to find a green pixel!)
BYTE *prgb = (BYTE*) pData, *best_pixel;
int dist, best_dist = 9999;
RGBTRIPLE *rgb;

for (iPixel=0; iPixel < numPixels; iPixel++, prgb+=iPixelSize)
{
  rgb = (RGBTRIPLE *)prgb;
  dist = rgb->rgbtBlue + rgb->rgbtRed + (255 - rgb->rgbtGreen);
  if (dist < best_dist)
  {
    best_dist = dist;
    best_pixel = prgb;
  }
}
```

Example 2: Draw X

```
// Draw X around it
   int i, inside = 5, outside = 10;
   RGBTRIPLE color = {0, 0, 255};  // red

   if (best_pixel - outside - outside * cxImage * iPixelSize >= pData &&
       best_pixel + outside + outside * cxImage * iPixelSize <  pData + lDataLen) {
     for (i = inside; i < outside; i++) {
       rgb = (RGBTRIPLE *)(best_pixel + (i + i * cxImage) * iPixelSize);
       *rgb = color;
       rgb = (RGBTRIPLE *)(best_pixel + (i - i * cxImage) * iPixelSize);
       *rgb = color;
       rgb = (RGBTRIPLE *)(best_pixel - (i + i * cxImage) * iPixelSize);
       *rgb = color;
       rgb = (RGBTRIPLE *)(best_pixel - (i - i * cxImage) * iPixelSize);
       *rgb = color;
     }
   }
   return NOERROR;
}
```

Advanced DirectShow

How to write an application
without the AppWizards

Skip section

COM

- COM is the Component Object Model, which is the basis for OLE.
- Call CoInitialize() once at the beginning to initialize COM.
- Call CoUninitialize() at termination to release COM.

COM

- Create each COM object with CoCreateInstance()
  - specify the following:
    - CLSID of the object
    - IID of the desired interface
    - a place to return a pointer to the interface.
  - Use this interface pointer for subsequent calls.

COM

- Use ip->QueryInterface() to get additional interfaces for the same object
  - Specify the following:
    - IID of the interface
    - A place to return a pointer to the interface
- Use ip->Release() to release the interface pointers.

Compiling with DirectShow

- Include <dshow.h>
- Link with
  - strmiids.lib
  - quartz.lib

DirectShow Steps

1. Initialize COM:

   ```
   CoInitialize(NULL);
   ```

2. Create an instance of the Filter Graph Manager:

   ```
   IGraphBuilder *pGraph = NULL;
   CoCreateInstance(CLSID_FilterGraph, NULL,
                    CLSCTX_INPROC_SERVER,
                    IID_IGraphBuilder, (void **)&pGraph);
   ```

DirectShow Steps

3. Get pointers to the other interfaces we need:
   - The **IMediaControl** interface provides methods for controlling the flow of data through the filter graph, including running, pausing and stopping.
     - IMediaControl *pMediaControl = NULL;
       pGraph->QueryInterface(IID_IMediaControl,
       (void **)&pMediaControl);

DirectShow Steps

3. Get pointers to the other interfaces we need:
   - The **IMediaEvent** interface contains methods for retrieving event notifications and for overriding the Filter Graph Manager's default handling of events.
     - IMediaEventEx *pEvent = NULL;
       pGraph->QueryInterface(IID_IMediaEvent,
       (void **)& pEvent);

DirectShow Steps

4. For capturing live video, we need a Capture Graph:
   - Create the Capture Graph
     ICaptureGraphBuilder2 *pCapture = NULL;
     CoCreateInstance(CLSID_CaptureGraphBuilder2, NULL,
     CLSCTX_INPROC,
     IID_ICaptureGraphBuilder2, (void **)&pCapture);
   - Attach the filter graph to the capture graph:
     pCapture->SetFiltergraph(pGraph);
   - Find a capture device
     - copy FindCaptureDevice() from playcap.cpp

DirectShow Steps

5. Use the Filter Graph Manager to build a filter graph.
   - For an AVI file:
     pGraph->RenderFile(L”C:\\Example.avi”, NULL);
   - For live capture:
     pCapture->RenderStream(&PIN_CATEGORY_PREVIEW,
     &MEDIATYPE_Video,
     pSrcFilter, NULL, NULL);

DirectShow Steps

6. (Optional) Send the video to another window.

```
IVideoWindow *pVidWin = NULL;
pGraph->QueryInterface(IID_IVideoWindow, (void **)&pVidWin);
pVidWin->put_Owner((OAHWND)g_hwnd);
                                        // or CWnd::m_hWnd in MFC
pVidWin->put_WindowStyle(WS_CHILD | WS_CLIPSIBLINGS);
RECT grc;
GetClientRect(g_hwnd, &grc);
pVidWin->SetWindowPosition(0, 0, grc.right, grc.bottom);
```

DirectShow Steps

7. Use the Filter Graph Manager to run the filter graph.
   - For an AVI file:
     pMediaControl->Run();
     pEvent->WaitForCompletion(INFINITE, &evCode);
   - For live capture:
     pMediaControl->Run();

DirectShow Steps

8. Clean up

```
if (pVidWin) pVidWin->Release();
pMediaControl->Release();
pEvent->Release();
pCapture->Release();
pGraph->Release();
CoUninitialize();
```

Play an AVI file

```
#include <dshow.h>
void main(void)
{
    IGraphBuilder *pGraph;
    IMediaControl *pMediaControl;
    IMediaEvent   *pEvent;
    CoInitialize(NULL);

    // Create the filter graph manager and query
       for interfaces.
    CoCreateInstance(CLSID_FilterGraph,
       NULL, CLSCTX_INPROC_SERVER,
       IID_IGraphBuilder, (void **)&pGraph);
    pGraph->QueryInterface(IID_IMediaControl,
       (void**)&pMediaControl);
    pGraph->QueryInterface(IID_IMediaEvent,
       (void **)&pEvent);

    // Build the graph. IMPORTANT: Change
       string to a file on your system.
    pGraph->RenderFile(L“C:\\movie.avi",
       NULL);
    // Run the graph.
    pMediaControl->Run();

    // Wait for completion.
    long evCode;
    pEvent->WaitForCompletion(INFINITE,
       &evCode);

    // Clean up.
    pMediaControl->Release();
    pEvent->Release();
    pGraph->Release();
    CoUninitialize();
}
```

Using a Custom Filter in an Application

- Use CamShiftDemo in OpenCV as an example.
- Include header file that defines CLSID and IID of custom filter and the GUID macros:

```
#include “objbase.h”
#include “initguid.h”
#include “imyfilter.h”
```

Using a Custom Filter in an Application

- Create instance of custom filter and property page:

```
CoCreateInstance(CLSID_CamShift, NULL,
                 CLSCTX_INPROC_SERVER,
                 IID_IBaseFilter,
                 (void**)&m_CamShift);
m_CamShift->QueryInterface(IID_ICamShift,
                           (void **)&m_CamShiftProp);
SafeRelease(m_CamShift);
```

Using a Custom Filter in an Application

- Add the filter to the graph:
  m_FilterGraph->AddFilter(m_CamShift, L”CamShift”);
- For a media file, just render it and it will be connected automatically:
  m_GraphBuilder->RenderFile(wname, 0);
- Use the property page to manipulate the filter.
- For live capture, we have to connect the pins ourselves…

Using a Custom Filter in an Application

```
m_FilterGraph->AddFilter( m_SourceFilter, L"Video Source" );

IPin* pSourceOut    = get_pin( m_SourceFilter, PINDIR_OUTPUT );
IPin* pCamShiftIn   = get_pin( m_CamShift, PINDIR_INPUT );
IPin* pCamShiftOut  = get_pin( m_CamShift, PINDIR_OUTPUT );

m_GraphBuilder->Connect( pSourceOut, pCamShiftIn );

m_GraphBuilder->Render( pCamShiftOut );

SafeRelease( pSourceOut );
SafeRelease( pCamShiftIn );
SafeRelease( pCamShiftOut );
```

Open CV

Intel Open Source Computer Vision Library

OpenCV

- Intel’s Open Source Computer Vision Library
- Available for anyone, including commercial use
- Source and examples are available.
- Built on top of Intel’s Image Processing Library (IPL) (if present), optimized for speed.

Some OpenCV Functions

- Motion Analysis and Object Tracking
  - Background Subtraction
  - Mean Shift
  - Optical Flow
- Image Analysis
  - Feature Detection
  - Pyramids
  - Moments
  - Histogram
  - Contour Retrieving
- Structural Analysis
  - Contour Processing
  - Shape Fitting
- Object Recognition
  - Eigen Objects
  - Hidden Markov Models
- 3D Reconstruction
  - Camera Calibration
  - Fundamental Matrix
  - View Morphing

OpenCV Resources

- Web page is http://www.intel.com/technology/computing/opencv/index.htm
- Newsgroup is at http://groups.yahoo.com/group/OpenCV/
  - Subscribe to the newsgroup with a free Yahoo ID.
- Downloads at http://sourceforge.net/projects/opencvlibrary
  - Version Beta 5 was released July 2005
- For help, read documentation, search sample apps or search newsgroup. If all else fails, post to the newsgroup.

Compiling with OpenCV

- After installing, do the following once (per installation):
  - Add opencv\bin to the system path (use the full path).
  - In DeveloperStudio:
    - In Tools->Options->(Projects->VC++)Directories:
      - Add absolute paths to the following to “include files”:
        - » opencv\cv\include
        - » opencv\cxcore\include
        - » opencv\otherlibs\highgui
        - » opencv\cvaux\include
      - Add absolute path to the following to “library files”:
        - » opencv\lib
    - Compile the debug libraries:
      - Start->OpenCV->OpenCV Workspace (.NET or MSVC6)
      - Select non-MIL debug version, and compile all.
        (MIL = Matrox Image Library)

Compiling with OpenCV

- Do the following for each project:
  - Include header file **cv.h**
  - Link debug version with **CVd.lib and cxcored.lib**
  - Link release version with **CV.lib and cxcore.lib**.
  - If loading/saving images, link **highguid.lib** (debug) and **highgui.lib** (release)

Using OpenCV

- Many of the functions only work with limited image formats (like 8-bit grayscale). In the old pdf manual, this was listed in Appendix A.
- There's no way to load a grayscale image or input a BW stream in DirectShow. Use CvtColor() to convert.

Open CV Image Formats

- Image formats are coded as a number and a letter, like 8u or 32f.
- The number is the number of bits per channel: 8, 16, 32, or 64.
- The letter is the type:
  - u = integer unsigned
  - s = integer signed
  - f = floating point
- The number of channels (1 for gray, 3 for RGB) is specified separately.

Example 3: Open CV Filter

Problem: Create a DirectShow filter using OpenCV that does Canny edge detection.

Example 3: Open CV Filter

Here's the prototype for the function:

void cvCanny(IplImage* *img*,
IplImage* *edges*, double *lowThresh*,
double *highThresh*, int *apertureSize*=3);

From Appendix A:

Input Format: 8u # chan: 1
Output Format: 8u

Example 3 Steps

1. Create a DirectShow filter using the method shown previously
2. In Transform()
   a. Convert the input bitmap to an IplImage
   b. Convert the input image to grayscale
   c. Run the Canny algorithm
   d. Convert back to RGB
3. Hook up the property page
4. Run it!

Example 3: Convert Bitmap

a. Convert the input bitmap to an IplImage

```
// Get the image properties from BITMAPINFOHEADER
CvSize size = cvSize(pvi->bmiHeader.biWidth,
                     pvi->bmiHeader.biHeight);
int stride = (size.width * 3 + 3) & -4;
                            // byte offset between rows
// convert the input RGB bitmap into an ipl image
IplImage *src_rgb=cvCreateImageHeader(size,IPL_DEPTH_8U, 3);
src_rgb->origin = 1;// bottom left (needed for text)
cvSetData(src_rgb, pData, stride);
…
cvReleaseImageHeader(&src_rgb);   // release memory when done
```

Example 3: Convert to Grayscale

b. Convert input image to grayscale

```
// convert input image to grayscale
IplImage *src_gray = cvCreateImage(size,
  IPL_DEPTH_8U, 1);
cvCvtColor(src_rgb, src_gray, CV_BGR2GRAY);
…
// clean up when we're done
cvReleaseImage(&src_gray);
```

Example 3: Run Canny

c. Run the Canny algorithm

```
// run the Canny edge detect algorithm
IplImage *dst_gray = cvCreateImage(size,
                                   IPL_DEPTH_8U, 1);
cvCanny(src_gray, dst_gray, low_thresh,
high_thresh, win_size);
…
// clean up
cvReleaseImage(&dst_gray);
```

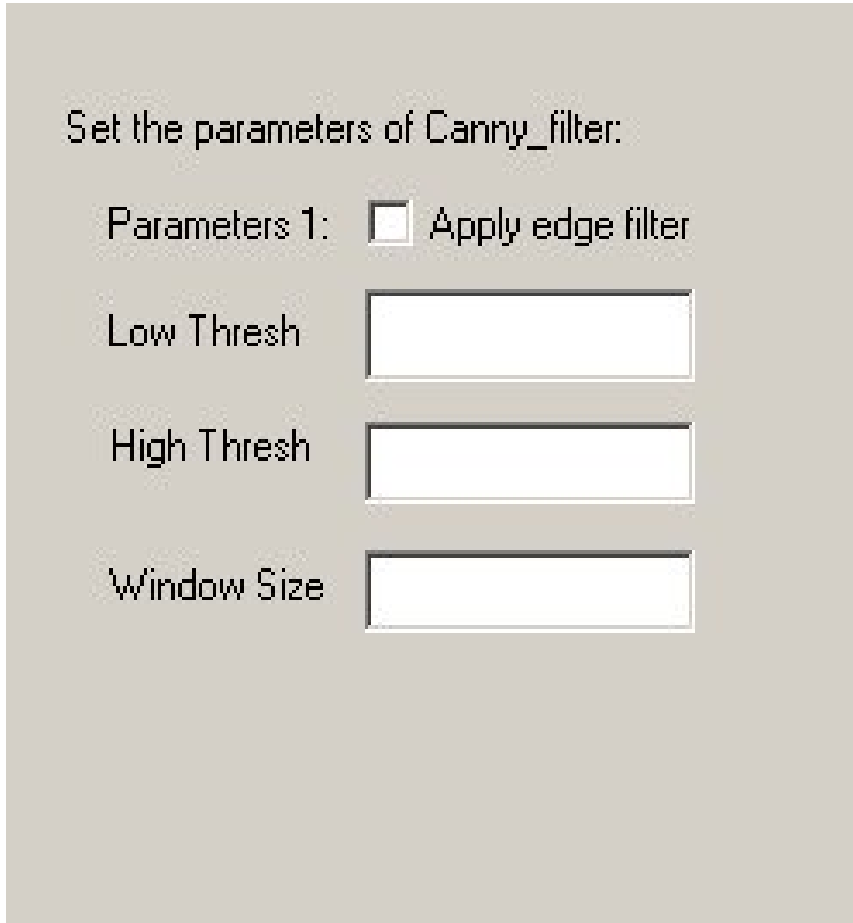
d. Convert back to RGB

```
// convert back to RGB and overwrite input image
cvCvtColor(dst_gray, src_rgb, CV_GRAY2BGR);
```

Example 3: Properties

Step 3: Hook up the property page

- Create the GUI controls
- Add variables
- Use these variables in the Canny function



Step 4: Run it!

Example 4: Open CV Application

Problem: Show the running average of a video stream.
(Used in background subtraction.)

Example 4: Open CV Application

Function Prototype:

void cvRunningAvg(IplImage* *src*, IplImage* *accum*, double *alpha*, IplImage* *mask*=0)

From Appendix A:

src can be 8u, 8s, or 32f; 1 or 3 channels.

accum must be 32f; same number of channels as *src.*

Example 4: Steps

1. Convert the input bitmap to an IplImage
2. Update accumulated image
   a. First time, create and initialize it
   b. If it already exists, call RunningAvg()
3. Convert accumulated image back to 8u, overwriting the input bitmap
4. Hook up the property page
5. Run it!

Example 4: Convert Input

Step 1: Convert the input bitmap to an IplImage

```
// Get the image properties from BITMAPINFOHEADER
CvSize size = cvSize(pvi->bmiHeader.biWidth,
                     pvi->bmiHeader.biHeight);
int stride = (size.width * 3 + 3) & -4;
                   // byte offset between rows
// convert the input bitmap into an ipl image
IplImage *src_rgb = cvCreateImageHeader(size,
                                 IPL_DEPTH_8U, 3);
src_rgb->origin = 1;    // bottom left
cvSetData(src_rgb, pData, stride);
…
cvReleaseImageHeader(&src_rgb);   // release memory
```

Example 4: Init Mean Image

Step 2: Update the accumulated image

a. First time, create and initialize it

In xxxDoc.h, add a variable for the image:

```
IplImage *m_meanImage;
```

In xxxDoc constructor, initialize it:

```
m_meanImage = NULL;
```

In xxxDoc destructor, destroy it:

```
cvReleaseImage(&m_meanImage);
```

Example 4: Update Mean Image

Step 2: Update the accumulated image

a. First time, create and initialize it
b. Otherwise call **`RunningAvg()`**

In xxxDoc::SampleCB()

```
// update the mean image
If (!m_meanImage) {
   m_meanImage = cvCreateImage(size, IPL_DEPTH_32F, 3);
   cvConvertScale(src_rgb, m_meanImage, 1.0, 0.0);
} else {
   cvRunningAvg(src_rgb, m_meanImage, 0.05, NULL);
}
```

Example 4: Convert Result

Step 3: Convert accumulated image back to 8u, overwriting the input bitmap.

```
// Convert floating point image back to 8 bit
cvConvertScale(m_meanImage, src_rgb, 1.0, 0.0);
```

Example 4: Properties

Step 5: Hook up the property page

a. Add a menu item
b. Make a new dialog box with variables assigned to the controls
c. Add corresponding variables to xxxDoc and initialize them in the constructor.
d. Add a handler for the new menu item:

```
CXxxDlg dlg;
dlg.m_* = m_*;
if (dlg.DoModal() == IDOK) {
    m_* = dlg.m_*;
}
```

Example 4: Output

Step 5: Run it!

Open CV HighGUI

- HighGUI is an additional OpenCV library with some user interface functions.
- Documentation was added in Beta 3 version.

HighGUI Image Functions

- Read and Write images
  - Handles these types:
    - Windows bitmap (.bmp, .dib)
    - Sun Raster (.sr, .ras)
    - JPEG (.jpeg, .jpg, .jpe)
    - Portable Network Graphics (.png)
    - Portable image (.pbm, .pgm, .ppm)
    - TIFF (.tiff, .tif)
  - IplImage* cvLoadImage(const char* *filename, int iscolor*);
    - Iscolor >0: 3 channel (default); 0: 1 channel; <0: matches input file
  - int cvSaveImage(const char* *filename*, const CvArr* *image*);

HighGUI Window Functions

- Display images in windows
  - int cvNamedWindow(const char* *name*, unsigned long *flags*);
    - Returns 0 on success, otherwise error code
    - *flags* can be CV_WINDOW_AUTOSIZE or 0
  - void cvShowImage(const char* *name*, const CvArr* *image*);
  - void cvDestroyWindow(const char* *name*);

HighGUI Key Functions

- Get keystrokes
  - int cvWaitKey(int delay = 0);
    - Argument specifies how long (ms.) to wait.
    - Delay <= 0 means wait forever.
    - Returns the key. Key = -1 if operation timed out.

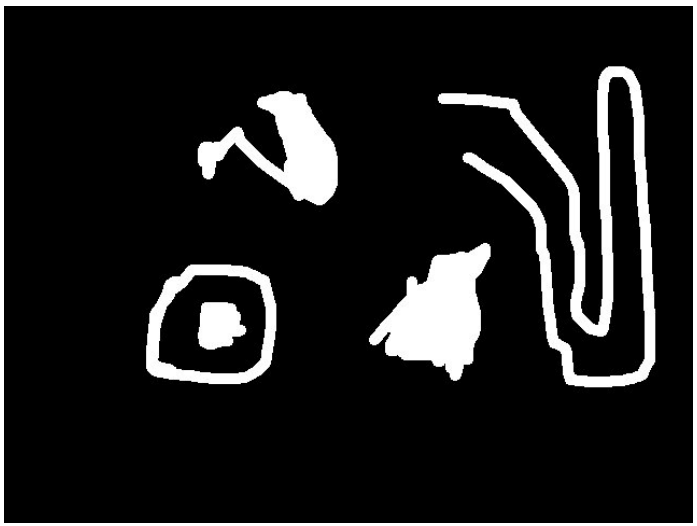
Compiling with HighGUI

- Include **cv.h** and **highgui.h**
- Add opencv\otherlibs\highgui to additional include directories
- Link with **cvd.lib, cxcored.lib** and **highguid.lib** for Debug
- Link with **cv.lib, cxcore.lib** and **highgui.lib** for Release

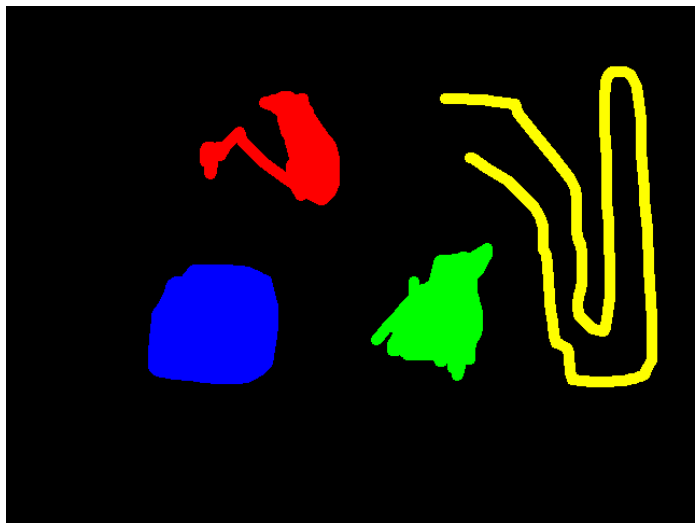
Example 5: HighGUI

Problem: Given a binary image, fill each set of connected components with a different color, ignoring holes.

Input:



Output:



Example 5: HighGUI

- We need some tools to deal with contours:
  - int cvFindContours(IplImage* *img*, CvMemStorage* *storage*, CvSeq***firstContour*, int *headerSize*=sizeof(CvContour), CvContourRetrievalMode *mode*=CV_RETR_LIST, CvChainApproxMethod *method*=CV_CHAIN_APPROX_SIMPLE);
  - void cvDrawContours(IplImage **img*, CvSeq* *contour*, int *externalColor*, int *holeColor*, int *maxLevel*, int *thickness*=1);

FindContours

int cvFindContours(IplImage* *img*, CvMemStorage* *storage*,
CvSeq***firstContour*,int *headerSize*=sizeof(CvContour),
CvContourRetrievalMode *mode*=CV_RETR_LIST,
CvChainApproxMethod *method*=CV_CHAIN_APPROX_SIMPLE);

img Single channel image of IPL_DEPTH_8U type. Non-zero pixels are treated as 1-pixels. The function modifies the content of the input parameter.

storage Contour storage location.

firstContour Output parameter. Pointer to the first contour on the highest level.

headerSize Size of the sequence header; must be equal to or greater than *sizeof(CvChain)* when the method CV_CHAIN_CODE is used, and equal to or greater than *sizeof(CvContour)* otherwise.

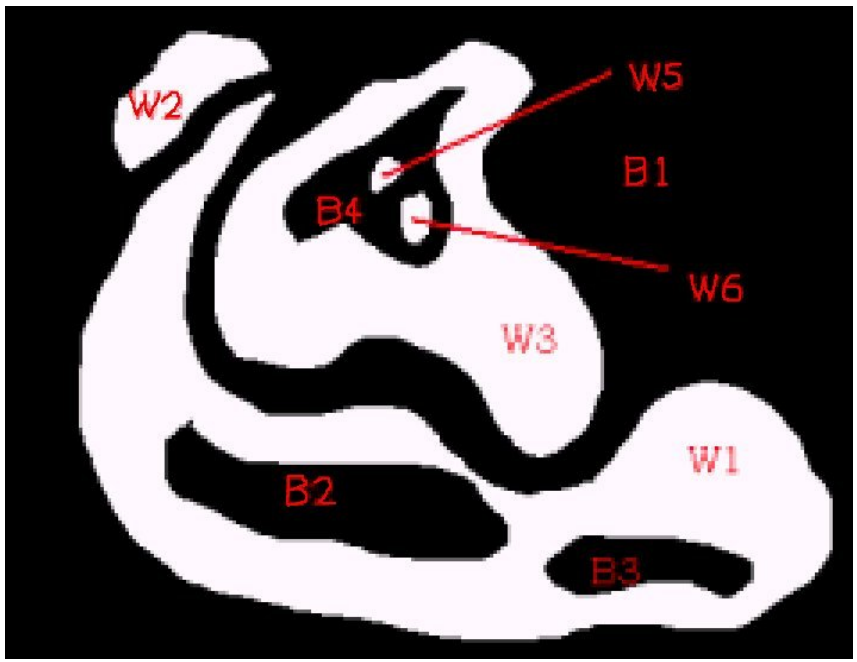
FindContours

int cvFindContours(IplImage* *img*, CvMemStorage* *storage*,
CvSeq***firstContour*,int *headerSize*=sizeof(CvContour),
CvContourRetrievalMode *mode*=CV_RETR_LIST,
CvChainApproxMethod *method*=CV_CHAIN_APPROX_SIMPLE);

mode Retrieval mode.

- CV_RETR_EXTERNAL retrieves only the extreme outer contours (list);
- CV_RETR_LIST retrieves all the contours (list);
- CV_RETR_CCOMP retrieves the two-level hierarchy (list of connected components);
- CV_RETR_TREE retrieves the complete hierarchy (tree).

FindContours



- CV_RETR_EXTERNAL returns W1, W2, W3
- CV_RETR_LIST returns W1, W2, W3, W5, W6, B2, B3, B4
- CV_RETR_CCOMP returns W1->B2, B3; W2->null; W3->B4; W5->null; W6->null
- CV_RETR_TREE returns
  W1 W2 W3
  W5 W6

FindContours

int cvFindContours(IplImage* *img*, CvMemStorage* *storage*,
CvSeq***firstContour*,int *headerSize*=sizeof(CvContour),
CvContourRetrievalMode *mode*=CV_RETR_LIST,
CvChainApproxMethod *method*=CV_CHAIN_APPROX_SIMPLE);

method Approximation method.

- CV_CHAIN_CODE outputs contours in the Freeman chain code.
- CV_CHAIN_APPROX_NONE translates all the points from the chain code into points.
- CV_CHAIN_APPROX_SIMPLE compresses horizontal, vertical, and diagonal segments, that is, it leaves only their ending points;
- CV_CHAIN_APPROX_TC89_L1, CV_CHAIN_APPROX_TC89_KCOS are two versions of the Teh-Chin approximation algorithm.

DrawContours

void cvDrawContours(IplImage **img*, CvSeq* *contour*, int *externalColor*, int *holeColor*, int *maxLevel*,int *thickness*=1);

img Image where the contours are to be drawn. Like in any other drawing function, every output is clipped with the ROI.

contour Pointer to the first contour.

externalColor Color to draw external contours with.

holeColor Color to draw holes with.

maxLevel Maximal level for drawn contours. If 0, only the contour is drawn. If 1, the contour and all contours after it on the same level are drawn. If 2, all contours after and all contours one level below the contours are drawn, etc.

thickness if positive or zero, the thickness of lines the contours are drawn with. If negative (CV_FILLED), the contour is filled.

Example 5: Steps

Now we're ready to list the steps to solve the problem:

1. Load the input image
2. Display the input image
3. Convert the input to grayscale
4. Find the contours
5. Make an output image and draw the contours
6. Save and display the output image

Example 5: Load Input

Step 1: Load the input image

```
#include <cv.h>
#include "highgui.h"
void main()
{
    IplImage *image = cvLoadImage("contour_in.bmp");
    CvSize size = cvSize(image->width, image->height);
                                                    // need this later
…
    cvReleaseImage(&image);
}
```

Example 5: Display Input

Step 2: Display the input image

```
char *orig_window_name = "Original image";
cvNamedWindow(orig_window_name, 0);  // not autosized
cvShowImage(orig_window_name, image);
…
cvWaitKey();
…
cvDestroyWindow(orig_window_name);
```

Example 5: Convert Input

Step 3: Convert the input to grayscale

```
 // convert to grayscale
  IplImage *src_gray = cvCreateImage(size, IPL_DEPTH_8U, 1);
  cvCvtColor(image, src_gray, CV_BGR2GRAY);
…
cvReleaseImage(&src_gray);
```

Example 5: Find Contours

Step 4: Find the contours

```
// find contours
CvMemStorage *contour_storage = cvCreateMemStorage(0);
CvSeq* cont;
int num_contours = cvFindContours(src_gray, contour_storage, &cont,
   sizeof(CvContour), CV_RETR_EXTERNAL,
   CV_CHAIN_APPROX_NONE);
...
cvReleaseMemStorage(&contour_storage);
```

Example 5: Draw Contours

Step 5: Make an output image and draw the contours

```
IplImage *contours = cvCreateImage(size, IPL_DEPTH_8U, 3);
cvSetZero(contours);

// draw filled contours
for (int i = 0 ; cont; cont = cont->h_next, i++) {
  i = i % (sizeof(color_list) / sizeof(color_list[0]));
  cvDrawContours(contours, cont, color_list[i], RGB(0, 0, 0),
   0 /* maxlevel */, CV_FILLED);
}
…
cvReleaseImage(&contours);
```

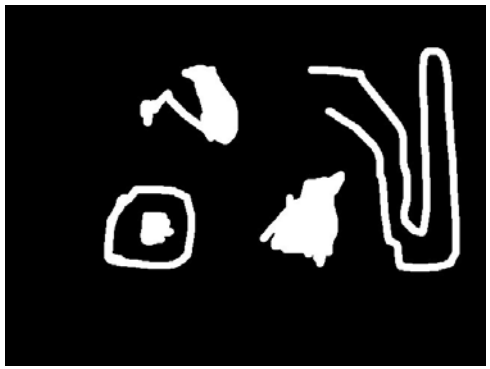
Example 5: Save Output

Step 6: Save and display the output image

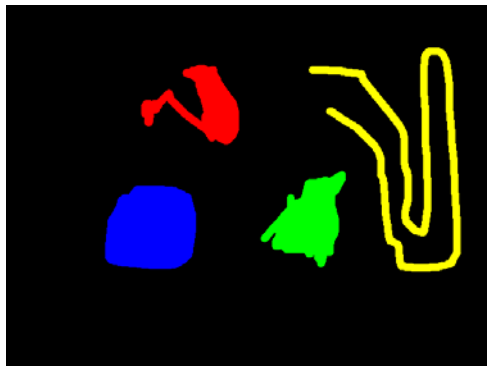
```
  // save output image
  cvSaveImage("contour_out.bmp", contours);

  // display output image
  char *contour_window_name = "Contour image";
  cvNamedWindow(contour_window_name,
                              CV_WINDOW_AUTOSIZE);
  cvShowImage(contour_window_name, contours);
…
  cvDestroyWindow(contour_window_name);
```

Example 5: Results

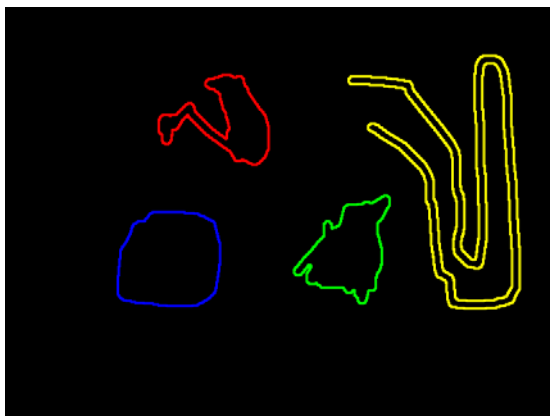
Input Image:



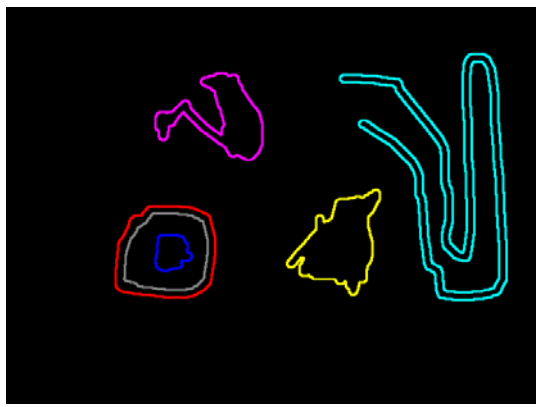
CV_RETR_EXTERNAL, filled



CV_RETR_EXT, width=2



CV_RETR_LIST, width=2



Contact

- Drop me an email if you found this useful, or have suggestions.
  - LSpencer@cs.ucf.edu
- The latest version is at http://www.cs.ucf.edu/~lspencer/vid_app.pdf