Một số hàm xử lý ảnh trong OpenCV

OpenCV là một thư viện xử lý ảnh nguồn mở của Intel. Nó có nhiều tác dụng như : nhận dạng mặt, dò tìm mặt, phát hiện mặt, lọc Kalman, nó thể hiện sự đa dạng của trí tuệ nhân tạo (AI). Thêm vào đó nó cải thiện rất nhiều các thuật toán cở bản của thị giác máy như các hàm API cấp thấp hơn.

Hiểu được phương thức làm việc của chúng như thế nào là cách rất tốt khi sử dụng OpenCV. Trong bài này, tôi sẽ giới thiệu về OpenCV và cách sử dụng chúng để nhận dạng ảnh, dò tìm ảnh … Sau đó tôi sẽ giải thích cách thức làm việc của chúng và đưa ra cho bạn các lời khuyên cũng như những kinh nghiệm để có thể tiếp thu chúng một cách tốt nhất.

Trong bài báo giới thiệu này. Tôi sẽ nói cho bạn biết làm thế nào cài đặt và sử dụng một vài chức năng của OpenCV trên máy của bạn. Bạn sẽ học được cách đọc và viết các file ảnh, bắt ảnh, chuyển đổi giữa các định dạng ảnh và truy nhập vào dữ liệu của phần tử ảnh (fixel) thông qua giao diện OpenCV.

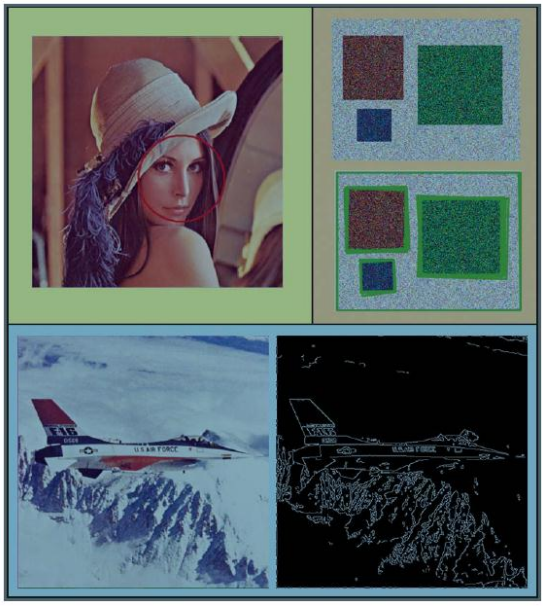

Figure 1:Rất nhiều tác dụng của OpenCV: dò tìm mặt(phía trên, bên trái), dò biên(phía trên, bên phải), dò cạnh (bên dưới)

I.Tổng quan về OpenCV

OpenCV là thư viện xử lý ảnh mã nguồn mở hoàn toàn miễn phí. Bạn có thể tải nó từ địa chỉ ***http://sourceforge.net/projects/opencvlibrary*** .

Intel đưa ra phiên bản OpenCV đầu tiên vào năm 1999. Ban đầu nó yêu cầu phải có thư viện xử lý ảnh của Intel. Sau đó vì sự lệ thuộc này mà họ đã phải gỡ bỏ và bây giờ chúng ta có thể sử dụng thư viện này hoàn toàn độc lập.

OpenCV rất đa dạng, nó hỗ trợ rất nhiều hệ điều hành như: Window, Linux và MacOSX.

II.Những điểm đặc trưng

OpenCV có rất nhiều chức năng. Bạn có thể bất ngờ khi lần đầu tiếp xúc với nó. Tuy nhiên, bạn sẽ chỉ cần một vài lần đầu để khởi động nó.

Sau đây là những tóm tắt cơ bản về hệ thống các về chức năng của các hàm trong OpenCV, mà cụ thể ở đây là ver 1.0 (phiên bản ra cùng thời gian với bài viết này)

2.1/Image and Video I/O

Những giao diện này sẽ giúp bạn đọc được dự liệu ảnh từ file hoặc trực tiếp từ video. Bạn cũng có thể tạo các file ảnh và video với giao diện này

2.2/Thị giác máy và các thuật toán xử lý ảnh (General computer-vision and image-processing algorithms(mid – and low level APIs))

Sử dụng những giao diện này, bạn có thể thực hành với rất nhiều chuẩn thị giác máy mà không cần phải có mã nguồn của chúng.

2.3/Modul thị giác máy ở cấp độ cao

OpenCV gồm một vài tác dụng ở cấp độ cao. Thêm vào nhận dạng mặt, dò tìm, theo dõi. Nó bao gồm luồng thị giác (sử dụng camera di động để xác định cấu trúc 3D), kích cỡ camera và âm thanh nổi.

2.4/AI and machine-learning

Các ứng dụng của thị giác máy thường yêu cầu máy móc phải học (machine learning) hoặc các hình thức trí tuệ nhân tạo khác. Một vài trong số chúng là có sẵn trong gói OpenCV

2.5/Lấy mẫu ảnh và phép biến đổi

Nó thường rất tốt cho quá trình xử lý một nhóm phần tử ảnh như là một đơn vị. OpenCV bao gồm lấy tách ra, lấy mẫu ngẫu nhiên, phục chế, xoay ảnh, làm cong ảnh (warping), thay đổi hiệu ứng của ảnh.

2.6/Cách thức tạo và phân tích ảnh nhị phân

Ảnh nhị phân thường xuyên được dùng trong các hệ thống kiểm tra có khuyết điểm hình dạng hoặc các bộ phận quan trọng. Sự biểu diễn ảnh cũng rất thuận tiện khi chúng ta biết rõ vật thể cần bắt.

2.7/Cách thức cho tính toán thông tin 3D (methods for computing 3D information)

Những hàm này rất có ích khi cần sắp xếp và xác định với một khối lập thể (with a stereo rig) hoặc với không gian nhìn phưc tạp (multiple views) từ một camera riêng.

2.8/Các phép toán cho xử lý ảnh, thị giác máy và biểu diễn ảnh(image interpretation)

OpenCV sử dụng các phép toán phổ biến như: đại số học, thống kê và tính toán hình học

2.9 /Đồ họa

Những giao diện này giúp bạn viết chữ và vẽ trên hình ảnh. Thêm vào đó những chức năng này được sử dụng nhiều trong ghi nhãn và đánh dấu. Ví dụ nếu bạn viết một chương trình cần nhận dạng nhiều đối tượng thì nó sẽ rất có ích cho tạo nhãn ảnh (label image) với kích thước và vị trí.

2.10/Phương thức GUI

OpenCV bao gồm cửa sổ giao diện của chính bản thân nó. Trong khi đó những giao diện này được so sánh giới hạn với khả năng có thể thực hiện trong mỗi môi trường. Chúng cung cấp những môi trường API đa phương tiện và đơn giản để hiện thị hình ảnh, cho phép người dùng nhập dữ liệu thông qua chuột , bàn phím và điều khiển quá trính.

2.11 Cấu trúc dữ liệu và giải thuật

Với những giao diện này bạn có thể giữ lại, tìm kiếm, lưu và cách danh mục điều khiển, các tuyển tập(cũng như các tập hợp lệnh được gọi), đồ họa và sơ đồ nhánh một cách hiệu quả.

2.12/*Khả năng tồn tại lâu dài của dữ liệu(*Data *persistence)*

Những phương pháp này cung cấp các giao diện một cách thuận lợi để lưu trữ các dạng khác nhau của dữ liệu vào đĩa để có thể khôi phục khi cần thiết.

Hình 1 cho chúng ta thấy một vài ví dụ về tác dụng của OpenCV như: nhận dạng mặt, nhận dạng đường viền, nhận dạng biên.

3.Cách tổ chức

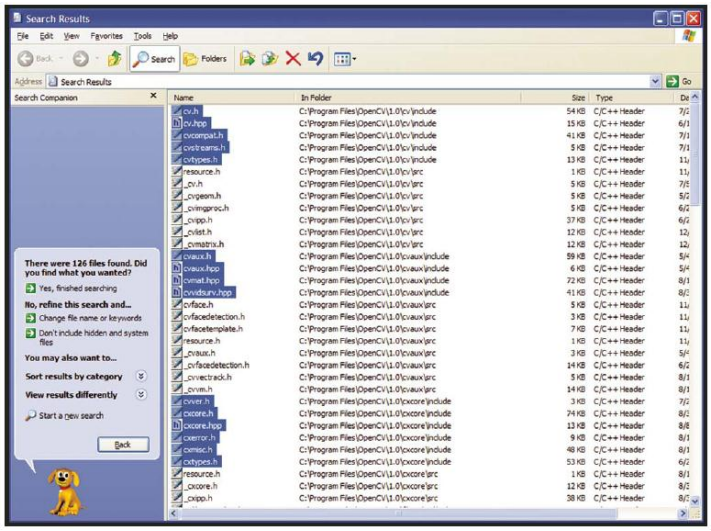

Figure 2:Thêm file header vào thư mục include

Cấu trúc OpenCV gồm nhiều module.

CXCORE bao gồm các dạng dữ liệu cơ bản rõ rang. Ví dụ cấu trúc dự liệu về ảnh, điểm, hình chữ nhật được xác định trong file cxtypes.h. CXCORE chứa các phép toán đại số tuyến tính và thống kê, các hàm lưu trữ lâu dài (persistence fun…) và các lỗi thao tác. Có điều lạ lung thay là các hàm đồ họa được cho việc vẽ ảnh cũng được lưu trữ tại đây.

CV chứa đựng quá trình xử lý ảnh và các phương pháp đánh giá sơ bộ kích thước ảnh. Những hàm tính toán hình học cũng được lưu trữ tại đây.

CVAUX được mô tả trong văn bản của OpenCV như là modul cũ và chỉ dùng để thí nghiệm. Tuy nhiên, giao diện đơn giản nhất cho nhận dạng mặt được nằm trong modul này. Những mã nguồn nằm trong module này rất phù hợp cho việc nhận dạng mặt và chúng được sử dụng rộng rãi cho mục đích này.

3.1/ML chứa đựng giao diện machine- learning

Những hàm còn lại được nằm trong HighGUI và CVCAM. Cả hai đều nằm ở thư mục có tên “otherlibs”, sử dụng chúng rất dễ gặp lỗi. Vì rằng HighGUI chứa các thư viện vào ra cơ bản , bạn sẽ muốn chắc chắn hơn, đừng bỏ sót nó.Nó cũng chứa đựng nhiều cửa sổ đa chức năng.

CVCAM là thư viện chứa các hàm truy nhập video thông qua DirectX trên môi trường Window 32 bit. Tuy nhiên, HighGUI cũng có các giao diện video.Trong bài báo này, tôi chỉ xem xét HighGUI. Chúng đơn giản, làm việc trên nhiều môi trường. Nếu bạn sử dụng Window XP hoặc 2000, bạn có thể làm tăng hiểu quả của nó bằng cách chuyển qua giao diện CVCAM, nhưng học OpenCV sẽ đơn giản hơn khi dùng HighGUI.

4.Khởi tạo OpenCV

4.1/Khởi tạo cơ bản

OpenCV cho Linux và MacOSX được đóng gói như một gói mã nguồn lưu trữ. Bạn sẽ phải xây dựng cả thư viện tĩnh và cả những đối tượng chia sẻ (shared-object). Bạn cần xây dựng RPM trước tiên và cài đặt từ nó, hoặc biên dịch và cài đặt nó ngay.Cấu trúc cho cả hai được nằm trong INSTALL.

Với Window ta khi cài đặt OpenCV,nó sẽ cop file OpenCV vào thư mục mà bạn chọn.Cách thức lựa chọn trong đường dẫn hệ thống của bạn chứa mã nhị phân OpenCV, đăng kí một vài bộ lọc DirectX.Mặc định nó cài đặt đến *C:/Program Files/OpenCV/<version>*

4.2/Những yêu cầu khi khởi tạo trọng window

Với người sử dụng Window, thật dễ dàng để cài đặt OpenCV, những cài đặt mặc định luôn làm việc. Nhưng một chút cải tiến có thể làm bạn ko vui vẻ về kết quả.Và đây là một vài gợi ý nhỏ.

Kể từ lúc OpenCV được phát triển như một bộ công cụ - không phải một chương trình- bạn muốn lưu nó vào một nơi nào khác nữa mà không phải thư mục **Program Files**. Nếu bạn thực sự muốn lưu ở một nơi nào khác thì trước tiên bạn cần chạy file cài đặt và điền cái thư mục bạn muốn lưu khi được hỏi.

Bạn cũng sẽ quyết định-trước khi khởi tạo – bạn muốn Window tìm kiếm những file dlls của OpenCV như thế nào?Bạn có thể lựa chọn chỉnh sửa đường dẫn hệ thống có sẵn của bạn thay cho đường dẫn của họ hoặc bạn thay đổi thư mục "bin" của OpenCV bằng thư mục gốc của bạn.

Nếu bạn thích chuyển những file dlls, nhưng bạn không chắc chắc thư mục gốc của bạn nằm ở đâu, bạn có thể lưu chúng bằng cách chạy chương trình **sysinfo utility** có sẵn ở địa chỉ **www.cognotics.com/utilities**

Nếu bạn thích chỉnh sửa đường dẫn hơn là thay đổi file dlls, bạn có thể khởi cài đặt và lựa chọn vào hộp thoại "Add bin directory to PATH".

4.3/Sau khi khởi tạo

Trong thư mục OpenCV có chứ một vài thư mục khác. Ta cùng xem xét những thư mục đó. Thư mục **docs** chứa file văn bản html cho toàn bộ các hàm OpenCV và kiểu dữ liệu.Từ file văn bản này bạn có thể làm các ví dụ, bạn cũng có thể muốn xem thư mục "samples".

Những file header sẽ cần thiết khi bạn dịch chương trình sử dụng OpenCV.

Cả Linux và Window bạn có thể xác định file header bằng cách tìm kiếm trong thư mục cài đặt và những thư mục khác những file có dạng *.h, *.hpp. Sẽ có rất nhiều tuy nhiều, tuy nhiên bạn không cần quá nhiều. Header dùng cho tất cả các module trừ HighGUI được tách riêng trong "include". Bạn có thể bỏ header trong thư mục "src". Khi dùng HighGUI bạn cần **highgui.h**. Nó nằm trong **otherlibs/highgui**.

5.Một vài cơ bản về chương trình của OpenCV

5.1/Những thứ cần biết về Header và Library

Hầu hết chương trình OpenCV cần cv.h và highgui.h.Sau đó để nhận dạng mặt chúng ta cần thêm cvaux.h. Phần còn lại của file header được thêm vào bởi những header cao nhất (high-level headers).

Nếu bạn có quên những file header trong nhánh thư mục (cài đặt mặc định) , bạn phải chắc chắn đường dẫn thêm vào khi dịch phải chứa những thư mục này. Nếu bạn từng tập hợp các header vào một thư mục , thì chắc chắn cái thư mục này đã được thêm vào khi dịch.

Mối liên kết của bạn sẽ cần cả đường dẫn thư viện lẫn tên của thư viện tĩnh được sử dụng. Thu viện tĩnh bạn cần liên kết tới cxcore.lib, cv.lib và highgui.lib. Sau đó để phục vụ cho quá trình nhận dạng mặt bạn cần liên kết tới cvaux.lib. Tất cả thư viện này đều nằm trong thư mục "lib" của OpenCV.

5.2/Đọc và ghi hình ảnh *(Sử dụng thư viện: highgui.h)*

Nhập và xuất ảnh là rất đơn giản với OpenCV. Hình 3 cho chúng ta thấy một chương trình hoàn chỉnh để đọc một ảnh từ file và ghi nó file thứ 2, trong một đinh dạng khác.

Việc đọc một file ảnh, ta gọi hàm cvLoadImage(), ở trong ví dụ nó nằm ở dòng thứ 12. OpenCV hỗ trợ hầu hết các định dạng phổ biến như JPEG, PNG và BMP. Bạn không cần cung cấp các thông tin định dạng. cvLoadImage() nhận diện định dạng file bằng cách đọc file header.

Việc ghi một ảnh vào file ta gọi hàm cvSaveImage(). Hàm này quyết định định dạng file ta sử dụng tử file ban đầu. Trong ví dụ này, file ban đầu có định dạng "png", vì vậy nó sẽ ghi dữ liệu ảnh dạng PNG.

```
1 // ImageIO.c
2 //
3 // Example showing how to read and write images
4
5 #include "cv.h"
6 #include "highgui.h"
7 #include <stdio.h>
8
9 int main(int argc, char** argv)
10 {
11    IplImage * pInpImg = 0;
12
13    // Load an image from file
14    cvLoadImage("my_image.jpg", CV_LOAD_IMAGE_UNCHANGED);
15    if(!pInpImg)
16    {
17       fprintf(stderr, "failed to load input image\n");
18       return -1;
19    }
20
21    // Write the image to a file with a different name,
22    // using a different image format -- .png instead of .jpg
23    if( !cvSaveImage("my_image_copy.png", pInpImg) )
24    {
25       fprintf(stderr, "failed to write image file\n");
26    }
27
28    // Remember to free image memory after using it!
29    cvReleaseImage(&pInpImg);
30
31    return 0;
32 }
```

Figure 3:Ví dụ chương trình đọc một ảnh từ file và ghi nó vào một file khác với định dạng khác

Cả cvLoadImage() và cvSaveImage() đều nằm trong modul HighGUI.

Khi chúng ta đã kết thúc quá trình nhập ảnh và sử dụng xong hàm cvLoadImage(), chúng ta cần gọi hàm cvReleaseImage() như dòng 29. Hàm này trả về địa chỉ của điểm khi mà ta nhập vào bởi vì nó đảm bảo an toàn dữ liệu. Nó trả tự do cho cấu trúc ảnh nếu nó là không rỗng. Sau khi giải phóng nó, nó tạo một điểm ảnh bằng 0 (image pointer)

Trước khi hiển thị ảnh lên màn hình thì chúng ta phải biết làm như thế nào để tải một đọc từ ổ đĩa. Chúng ta sẽ sử dụng hàm

cvLoadImage();

IplImage* cvLoadImage(

const char* filename,

int iscolor = CV_LOAD_IMAGE_COLOR

);

filename: Tên đường dẫn ảnh iscolor: Chế độ hiển thị ảnh
Một số giá trị của iscolor:
CV_LOAD_IMAGE_GRAYSCALE
CV_LOAD_IMAGE_ANYDEPTH
CV_LOAD_IMAGE_UNCHANGED

Sau khi đã đọc được ảnh từ ổ cứng chúng ta sẽ hiển thị lên màn hình bằng hàm cvShowImage(); Đồng thời chúng ta phải tạo một cửa sổ hiển thị ảnh (Dùng hàm cvNamedWindow();)

cvNamedWindow("Tên cửa sổ hiển thị");

```
void cvShowImage (
        const char* name,
        const cvArr* image
);
```

Name: tên cửa sổ hiển thị ảnh

Image: tên ảnh

Ví dụ:

```
#include "highgui.h" int
main()
{
    IplImage* img = cvLoadImage(imag); // Tạo biến chứa ảnh
    cvNamedWindow("ShowIMG",CV_WINDOW_AUTOSIZE); // Tạo một cửa sổ hiển thị ảnh  cvShowImage("ShowIMG",img); // hiển thị ảnh

    cvWaitKey(0); // Đợi người dùng nhập vào một phím bất kì
    cvReleaseImage(&img); // Giải phóng bộ nhớ chứa ảnh
    cvDestroyWindow("ShowIMG"); // Đóng cửa sổ chứa ảnh

}
```

5.3/ Lưu ảnh

```
void cvSaveImage (
const char* name,
const cvArr* image,
const int* params CV DEFAULT(0)
);
```

OpenCV lưu trữ ảnh giống như trong C bơi IplImage. Chuẩn IPL cho thư viện xử lý ảnh là một sự thừa kế từ phiên bản OpenCV ban đầu.

Kiểu dữ liệu IplImage được xác định trong CXCORE. Thêm vào đó là dữ liệu phần tử ảnh thô.Nó chứa một vùng được miêu tả, cái mà ta gọi là đầu đề ảnh (Image Header). Chúng gồm:

-Width: chiều rộng của phần tử ảnh

-heght: chiều dài của phần tử ảnh.

-Depth:Một trong số vài giá trị không đổi đã xác định, nó đòi hỏi số bit trên mỗi phần tử ảnh của kênh. Ví dụ, nếu depth =IPL_DEPTH_8U , thì dữ liệu cho mỗi kênh phần tử ảnh được lưu bởi 8 bit, và kiểu unsigned.

-nChanel: Số kênh dữ liệu (Từ 1 đến 4) . Mỗi kênh chứa một kiểu của dữ liệu phần tử ảnh. Ví dụ, ảnh RGB có ba kênh: đỏ, xanh lá cây và xanh sẫm (thỉnh thoảng gọi là màu BGR, bởi vì dữ liệu điểm ảnh được lưu trữ bởi màu xanh sẫm, xanh lá cây và sau đó là màu đỏ).Ảnh đen trắng chỉ chứa một kênh duy nhất là độ sáng của điểm ảnh.

Ví dụ:

```
#include "highgui.h"
int main()
{
        IplImage* img = cvLoadImage(imag); // Tạo biến chứa ảnh
        cvNamedWindow("ShowIMG",CV_WINDOW_AUTOSIZE); // Tạo một cửa
        số hiển thị ảnh  cvShowImage("ShowIMG",img); // hiển thị ảnh

        IplImage* out = cvCreateImage(cvGetSize(img),img->depth,img-
        >nChannels);  cvConvertImage(img,out,CV_CVTIMG_FLIP); //
        Chuyển đổi ảnh  cvSaveImage("D:/test.bmp",img); // Lưu ảnh với
        một tên khác cvWaitKey(0); // Đợi người dùng nhập vào một phím
        bất kì cvReleaseImage(&img); // Giải phóng bộ nhớ chứa ảnh
        cvDestroyWindow("ShowIMG"); // Đóng cửa sổ chứa ảnh
}
```

5.4/Nhập video trực tiếp

Chụp ảnh từ một webcam hoặc từ một thiệt bị hinh ảnh số khác thật dễ dàng giống như chạy từ một file.Hình 4 cho chúng ta một danh sách chương trình đầy đủ có tác dụng chụp ảnh, và lưu trữ một vài trạng thái video cũng như đóng giao diện chụp lại.

Giao diện chụp được khởi tạo ở dòng thứ 19 trong ví dụ. Nó được gọi ra bởi hàm cvCaptureFromCAM(). Hàm này trả về một điểm đến một cấu trúc cvCapture. Ta không thể truy nhập và cấu trúc này ngay được. Thay vào đó, ta sẽ lưu trữ điểm này thông qua hàm cvQueryFrame().

Sau khi kết thúc quá trình nhập video, gọi hàm cvReleaseCapture() để giải phóng mã nguồn video. Giống như cvReleaseImage(), ta trả địa chỉ địa chỉ của điểm Cvcapture về hàm cvReleaseImage().

Đừng giải phóng hoặc thay đổi IplImage mà bạn nhận được từ cvQueryFrame()! Nếu bạn muốn chỉnh sửa dữ liệu ảnh, hãy tạo một bản copy để làm việc:

```
// Copy the video frame
IplImage *pImgToChange =
cvCloneImage(pVideoFrame);
// Insert your image-processing code here ...
// Free the copy after using it
cvReleaseImage(&pImgToChange);
```

5.4/Chuyển đổi màu

Hình 5 trình bày mã nguồn chuyển đổi một ảnh màu qua ảnh đen trắng. OpenCV hỗ trợ chuyển đổi từ nhiều chuẩn mẫu màu khác nhau, gồm RGB, HSV, YCrCb và CIELAB.

Có một chú ý là hàm cvCvtColor() yêu cầu hai ảnh trong danh sách nhập vào của nó. Cái đầu tiên pRGBImg là ảnh nguồn.Còn lại là pGrayImg là ảnh gửi tới.

Bởi vì mô hình này của mã nguồn ngẫu nhiên(passing source) và ảnh gửi tới là một hàm quá trình phổ biến trong OpenCV.Bạn sẽ thường xuyên phải tạo một ảnh gửi tới. Trên dòng 25,

gọi hàm CvCreateImage() để tạo ảnh có cùng kích cỡ như ban đầu, với dữ liệu ảnh ko có giá trị bạn đầu (with uninitialized pixel data)

5.6/Truy nhập giá trị điểm ảnh

Ta có thể tạo nhiều kiểu của hàm sử dụng OpenCV mà không cần truy nhập đến dữ liệu điểm ảnh ban đầu.Ví dụ, các chương trình nhận dạng mặt, theo dõi sẽ được trình bày sau trong loạt bài này không cần phải điều khiển dữ liệu điểm ảnh ngay. Thay vào đó, chúng làm việc với con trỏ ảnh và các cấu trúc cao cấp khác. Tất cả các cấp độ điểm ảnh được tính toàn và trình bày trong các hàm của OpenCV. Tuy nhiên, nếu bạn tự viết thuật toán xử lý ảnh của mình, bạn có thể cần truy nhập đến giá trị của điểm ảnh ban đầu. Đây là hai cách làm điều đó:

```
// Capture.c
//
// Example showing how to connect to a webcam and capture
// video frames

#include "stdio.h"
#include "string.h"
#include "cv.h"
#include "highgui.h"

int main(int argc, char ** argv)
{
   CvCapture * pCapture    = 0;
   IplImage *  pVideoFrame = 0;
   int         i;
   char        filename[50];

   // Initialize video capture
   pCapture = cvCaptureFromCAM( CV_CAP_ANY );
   if( !pCapture )
   {
      fprintf(stderr, "failed to initialize video capture\n");
      return -1;
   }

   // Capture three video frames and write them as files
   for(i=0; i<3; i++)
   {
      pVideoFrame = cvQueryFrame( pCapture );
      if( !pVideoFrame )
      {
         fprintf(stderr, "failed to get a video frame\n");
      }

      // Write the captured video frame as an image file
      sprintf(filename, "VideoFrame%d.jpg", i+1);
      if( !cvSaveImage(filename, pVideoFrame) )
      {
         fprintf(stderr, "failed to write image file %s\n", filename);
      }

      // IMPORTANT: Don't release or modify the image returned
      // from cvQueryFrame() !
   }

   // Terminate video capture and free capture resources
   cvReleaseCapture( &pCapture );

   return 0;
}
```

Figure 4: Ví dụ chương trình chụp ảnh video và lưu chúng vào một file

5.6.1/Truy nhập điểm ảnh một cách đơn giản

Cách dễ nhất để đọc điểm ảnh đơn là dùng hàm cvGet2D().

CvScalar cvGet2D(const CvArr*, int row, int col);

Hàm này có 3 thông số là :

-Một con trỏ chứa dữ liệu (CvArr*)

-Một mảng được xếp theo hang và theo cột

Dữ liệu chứa đựng có thể có cấu trúc IplImage.Cái hàng cao nhất của điểm ảnh có row =0 và hàng thấp nhất có row = chiều cao – 1.

Hàm cvGet2D() trả về cấu trúc dạng C là CvScalar được xác định như sau:

typedef struct CvScalar
{
double val[4];
}
CvScalar;

Giá trị của mỗi điểm ảnh cho mỗi kênh được nằm trong val[i]. Với ảnh đen trắng, val[0] chứa độ sáng của điểm ảnh.Ba giá trị khác được đặt ở giá trị 0. Với ảnh BGR thì blue = val[0], green = val[1] và red = val[2].

Ta có hàm bổ sung cvSetd() cho phép bạn chỉnh sửa giá trị điểm ảnh. Nó được khai báo như sau:

void cvSet2D(CvArr*, int row, int col,
CvScalar);

5.6.2/Truy nhập nhanh điểm ảnh

Mặc dù cvGet2D() và cvSet2D() rất dễ sử dụng. Nếu bạn muốn truy nhập nhiều hơn vào một vài giá trị điểm ảnh và xem những vấn đề của nó. Bạn sẽ muốn đọc giá trị này một cách tức thời từ bộ đệm dữ liệu ban đầu **IplImage.imageData**.

```
// ConvertToGray.c
//
// Example showing how to convert an image from color
// to grayscale

#include "stdio.h"
#include "string.h"
#include "cv.h"
#include "highgui.h"

int main(int argc, char** argv)
{
    IplImage * pRGBImg  = 0;
    IplImage * pGrayImg = 0;

    // Load the RGB image from file
    pRGBImg = cvLoadImage("my_image.jpg", CV_LOAD_IMAGE_UNCHANGED);
    if(!pRGBImg)
    {
       fprintf(stderr, "failed to load input image\n");
       return -1;
    }

    // Allocate the grayscale image
    pGrayImg = cvCreateImage
       ( cvSize(pRGBImg->width, pRGBImg->height), pRGBImg->depth, 1 );

    // Convert it to grayscale
    cvCvtColor(pRGBImg, pGrayImg, CV_RGB2GRAY);

    // Write the grayscale image to a file
    if( !cvSaveImage("my_image_gray.jpg", pGrayImg) )
    {
       fprintf(stderr, "failed to write image file\n");
    }

    // Free image memory
    cvReleaseImage(&pRGBImg);
    cvReleaseImage(&pGrayImg);

    return 0;
}
```

Figure 5: Ví dụ chương trình chuyển đổi ảnh màu qua ảnh đen trắng

Dữ liệu ảnh trong bộ đệm được lưu trữ dưới dạng mảng 1D, trong chủ yếu một hàng. Tất cả các giá trị điểm ảnh nằm trong hàng đầu được liệt kê đầu tiên. Tiếp đó là những dữ liệu điểm ảnh nằm trong hàng thứ hai ….v.v

Những lý do để làm việc này là: Dữ liệu điểm ảnh đã xếp thẳng hàng và chèn đầy nếu cần thiết, đến mức mỗi một hàng bắt đầu đều là bội chẵn của bốn byte. Ngoài ra, IplImage.widthStep biểu thị số byte dữ liệu điểm ảnh của mỗi dòng. Đó là dòng thứ i bắt đầu ở IplImage.imagedata+i*IplImage.widthStep.

IplImage.imagedata được định nghĩa kiểu char*, vì vậy bạn cần phải bạn tính chất của kiểu dữ liệu này. Ví dụ , dữ liệu ảnh có kiểu unsigned (loại phổ biến nhất), bạn sẽ cần hiểu về unsigned char* trước khi dùng nó hoặc sử dụng cái khác nó.

Nếu bạn truy nhật dữ liệu từ ảnh đen trắng (kênh đơn) và chiều sâu ảnh là 8 bit (1 byte trên 1 điểm ảnh), bạn sẽ cần truy nhập pixel[row][col] với cấu trúc:

```
pixel[row][col] = ((uchar*)
        (pImg->imageData +
        row*pImg->widthStep + col));
```

Trong ảnh đa kênh, giá trị kênh được kết hợp chặt chẽ với nhau. Đây là một trich đoạn mã nguồn truy nhập tới giá trị điểm ảnh blue, green và red:

```
step = pImg->widthStep;
nChan = pImg->nChannels;
// = 3 for a BGR image
buf = pImg->imageData;
```

```
blue[row][col] =
                ((uchar*)(buf + row*widthStep +
                nChan*col);
green[row][col] =
                ((uchar*)(buf + row*widthStep +
                nChan*col + 1);
red[row][col] =
                ((uchar*)(buf + row*widthStep +
                nChan*col + 2);
```

Cuối cùng, nếu chiều sâu ảnh lớn hơn 8 bit (ví dụ IPL_DEPTH_32S), bạn sẽ cần chuyển đổi byte bội số cho mỗi giá trị và nhận nó với bộ đệm song song để được số byte dữ liệu cho chiều sâu ảnh của bạn. Nó rất phi lý, tuy nhiên bạn sẽ gặp tình huống mà bạn cần truy nhập đến byte bội số của nó ngay tức thì.