

USP - ICMC - SSC
SSC 0715 (SensInt) - 2o. Semestre 2010

Disciplina de Sensores Inteligentes SSC-0715

Prof. Fernando Osório
Email: fosorio [at] { icmc. usp. br , gmail. com }
Estagiário PAE: Daniel Sales
Web: http://www.icmc.usp.br/~fosorio/
Wiki ICMC: http://wiki.icmc.usp.br/index.php/Ssc-715(Fosorio)

Aula 02

Aula 02 – Visão Computacional: OpenCV

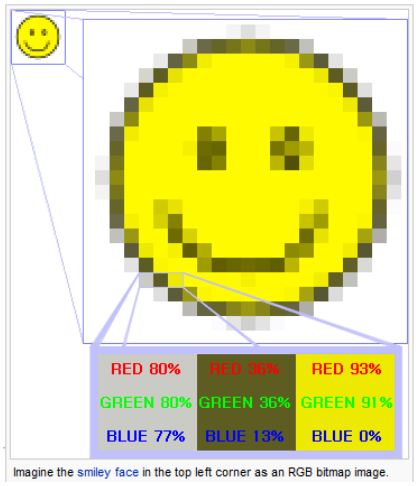
Agenda:

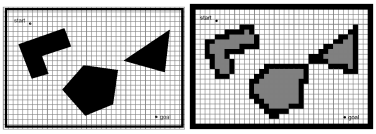
1. Processamento de Imagens
2. Visão Computacional
3. OpenCV
   - Instalação – Linux / Windows
   - Aquisição de Imagens
   - Salvando Imagens
   - Processando Imagens
4. Demos e Aplicações baseadas no OpenCV

Visão Computacional e Robótica Móvel

1. Processamento de Imagens

Mapa de Pixels = Bitmap





Amostragem:

Resolução X-Y

Quantização:

Bits por Pixel

B&W = 1 bit/pixel

Gray Scale (níveis de cinza)= 8 bits/pixel

Palette (falsas cores) = 8 bits/pixel

TrueColor RGB = 24 bits/pixel

RGB + Alfa (Transparência) = 32 bits/pixel

Multi-espectral = *N* Canais de cor

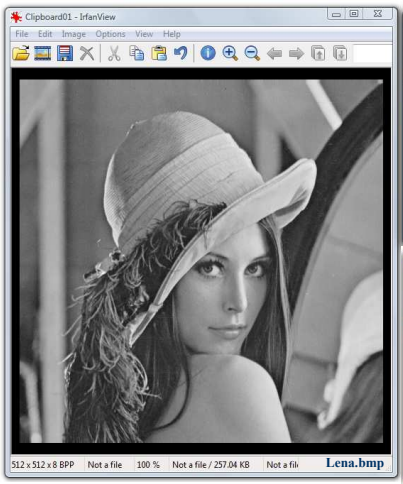
USP – ICMC - SSC0715- Turma 2010/2
Sensores Inteligentes / Enfase: Robótica
Prof. Fernando Osório – PAE Daniel Sales

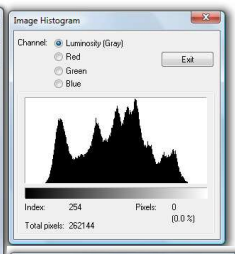
Visão Computacional e Robótica Móvel

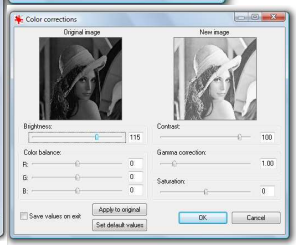
1. Processamento de Imagens

Imagens em Tons de Cinza









- **Histograma**
- **Brilho**
- **Contraste**
- **Formato das Imagens:**
  BMP, GIF, JPG,
  PBM, PGM, PPM,
  PNG, ...

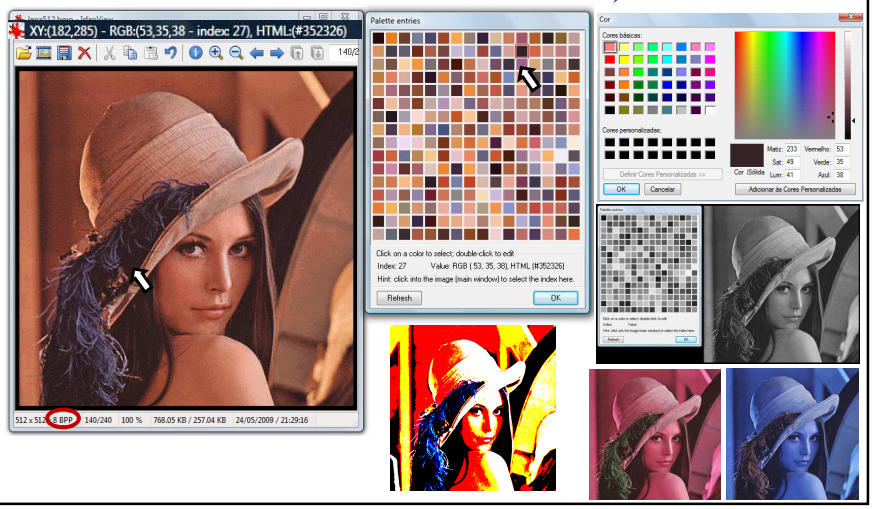
Compactação:

- **LossLess**
- **Lossy**

Visão Computacional e Robótica Móvel

1. Processamento de Imagens

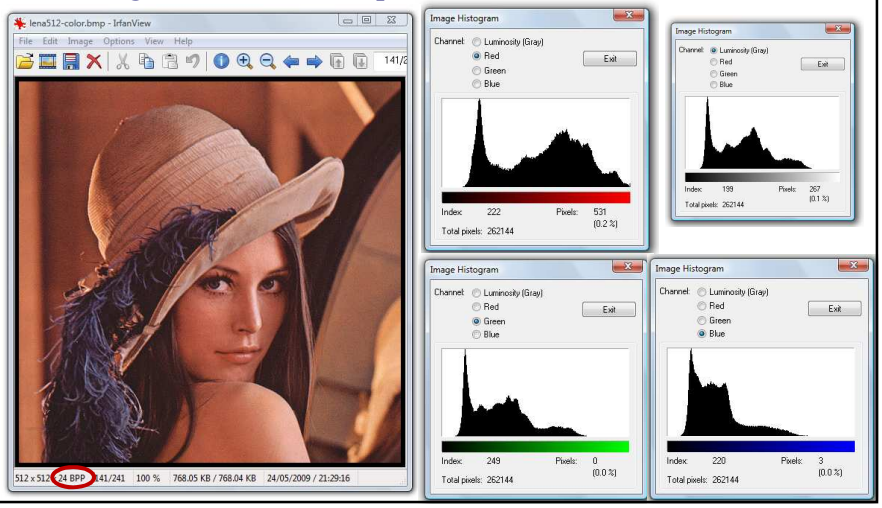
Imagens Coloridas: Falsas Cores (Palette)



Visão Computacional e Robótica Móvel

1. Processamento de Imagens

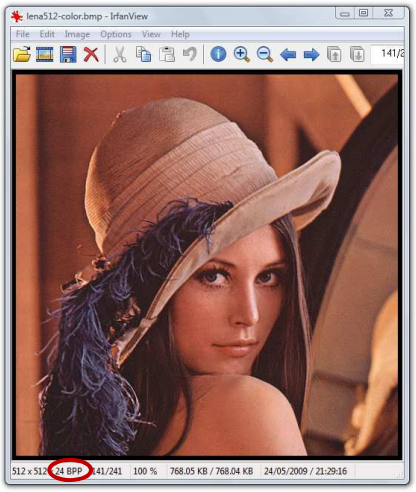
Imagens Coloridas: Espaço de Cores – RGB, YCM, HSV, ...

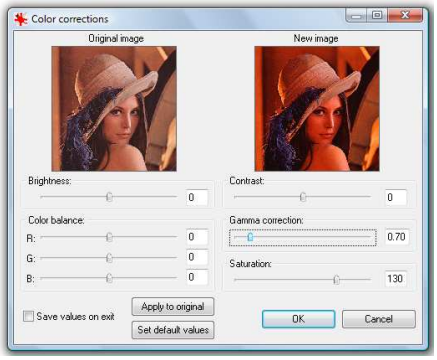


Visão Computacional e Robótica Móvel

1. Processamento de Imagens

Imagens Coloridas: Espaço de Cores – RGB, YCM, HSV, ...





Brightness, Contrast
Gamma Correction, Saturation
Filters, Convolution,
Equalization, ...

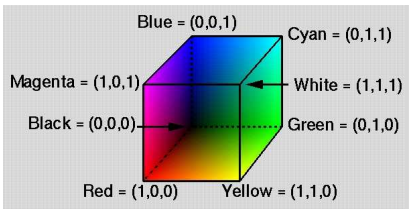


USP – ICMC - SSC0715- Turma 2010/2
Sensores Inteligentes / Enfase: Robótica
Prof. Fernando Osório – PAE Daniel Sales

Visão Computacional e Robótica Móvel

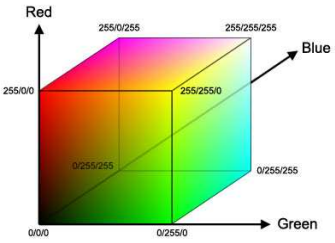
1. Processamento de Imagens

Imagens Coloridas: Espaço de Cores – RGB

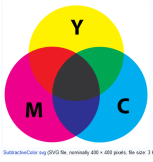


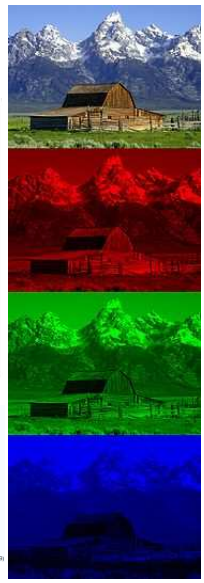


A representation of additive color mixing. Projection of primary color lights on a screen shows secondary colors where two overlap; the combination of all three of red, green, and blue in appropriate intensities makes white.





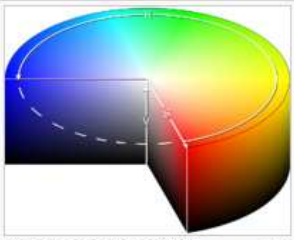




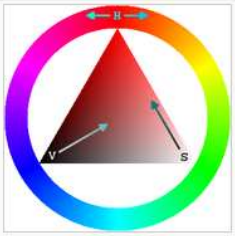
Visão Computacional e Robótica Móvel

1. Processamento de Imagens

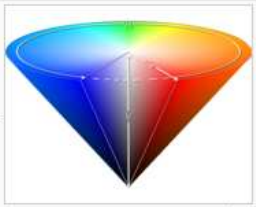
Imagens Coloridas: Espaço de Cores – HSV



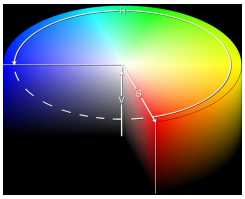
Graphical depiction of HSV



An HSV color wheel allows the user to quickly select a multitude of colors.

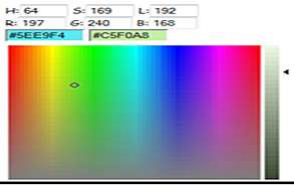


The conical representation of the HSV model is well-suited to visualizing the entire HSV color space in a single object.



H = HUE = MATIZ
S = SATURATION
V = VALUE

RGB = Red, Green, Blue



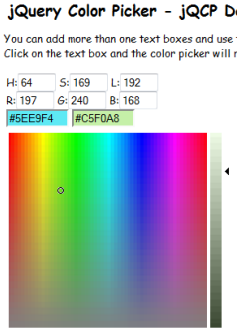
Visão Computacional e Robótica Móvel

1. Processamento de Imagens

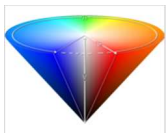
Imagens Coloridas: Espaço de Cores – HSV, HSL, Lab, ...



HSL arranged as a double-cone

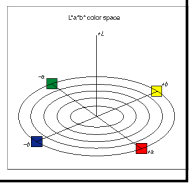


http://jqframework.com/jqcp/



HSV

L a*b*



2. Visão Computacional

Visão Computacional

* Implementação

"Visual Servoing" – Navegação Visual

- Uso de referências visuais para o auxílio a Navegação
  Exemplos:

  Target Point - Ajustar a trajetória em relação a um ponto de referência
  Follow Line - Seguir uma marcação de pista
  Follow Me - Acompanhar um outro robô (com marcação visual)

- Fusão Sensorial:

  Integrar dados do Laser com dados de Imagens
  Laser: Detecção e Desvio de Obstáculo
  Imagens: Direcionamento, Referência e Navegação

2. Visão Computacional

Visão Computacional

* Conceitos

SSC0714 - AULA 09 / Aula10 – Robôs Móveis Autônomos
http://coteia.icmc.usp.br/mostra.php?ident=575.6
http://wiki.icmc.usp.br/index.php/Material_SSC0714(fosorio)

Conceitos:

Imagem: Representação – Amostragem e Quantização
Imagem: Representação – Modelos de Cores RGB, CMY, HSV, ...

Imagem: Análise – Histograma, Região, Vizinhança, Distância
Imagem: Processamento – Limiarização, Equalização de Histograma,
Ajuste de Brilho e Contraste, Suavização, Remoção de Ruído,
Detecção de Bordas, Detecção de Linhas, Erosão/Dilatação,
Transformações (FFT, Wavelets), Deformações,
Filtros: Smoothing, Sharpening, Convolução, ...
Imagem: Segmentação (cor, textura), Extração de Atributos
Imagem: Reconhecimento de Objetos e Classificação

2. Conceitos e Aplicações

Visão Computacional

* **Conceitos**

SSC0714 - AULA 09 / Aula10 – Robôs Móveis Autônomos
http://coteia.icmc.usp.br/mostra.php?ident=575.6
http://wiki.icmc.usp.br/index.php/Material_SSC0714(fosorio)

Referências:

Autor	**Gonzalez, Rafael C / Rafael C. Gonzalez, Richard E. Woods**
Título	**Digital image processing**
Edição	**3rd ed**
Editora	**Upper Saddle River, N.J; Harlow : Pearson/Prentice Hall, 2008**



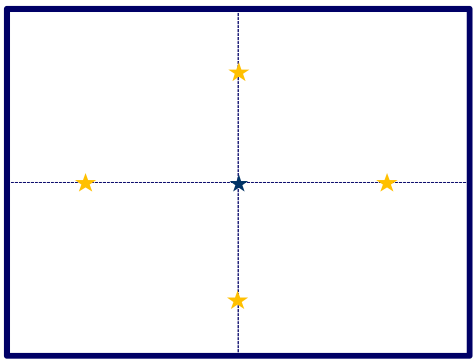
Visão Computacional: **2D**
Visão Computacional: 3D (Stereo)... Não abordado nesta disciplina.

2. Conceitos e Aplicações

Visão Computacional

* **Aplicações: Target (set) Point, Follow Line, Follow Me**

Target Point - Ajustar a trajetória em relação a um ponto de referência



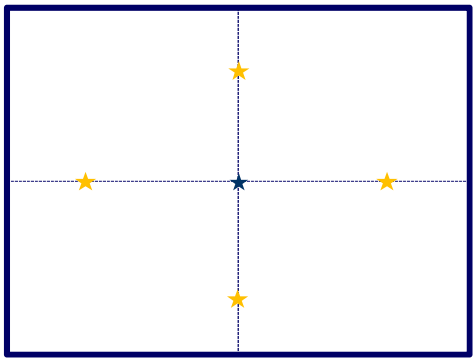


2. Conceitos e Aplicações

Visão Computacional

* **Aplicações: Target (set) Point, Follow Line, Follow Me**

Target Point - Ajustar a trajetória em relação a um ponto de referência





USP – ICMC - SSC0715- Turma 2010/2
Sensores Inteligentes / Enfase: Robótica
Prof. Fernando Osório – PAE Daniel Sales

2. Conceitos e Aplicações

Visão Computacional

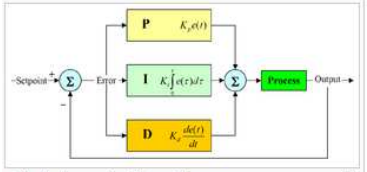
* **Aplicações: Target (set) Point, Follow Line, Follow Me**

Target Point - Ajustar a trajetória em relação a um ponto de referência

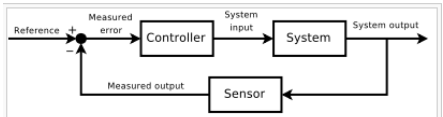
Controle de Malha Fechada:

Set Point *x* Posição Atual => Erro de Posição

Ajuste: Controlador P
Controlador PD
Controlador PID



A block diagram of a PID controller



The concept of the feedback loop to control the dynamic behavior of the system: this is negative feedback, because the sensed value is subtracted from the desired value to create the error signal which is amplified by the controller.

http://en.wikipedia.org/wiki/Control_theory

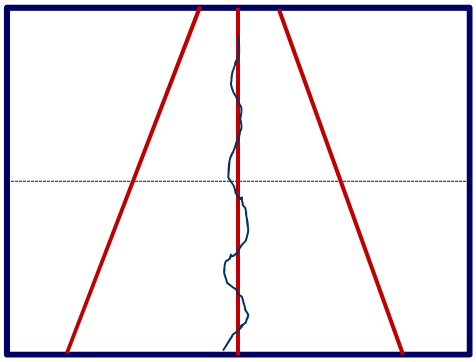
http://en.wikipedia.org/wiki/PID_controller

2. Conceitos e Aplicações

Visão Computacional

* Aplicações: Target (set) Point, Follow Line, Follow Me

Follow Line - Seguir uma marcação de pista



Objetivos:

-Seguir a marcação central da pista

-Típica aplicação em robótica... AGVs, Desafios Robóticos

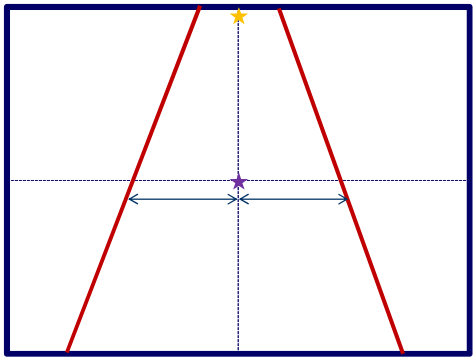


2. Conceitos e Aplicações

Visão Computacional

* Aplicações: Target (set) Point, Follow Line, Follow Me

Follow Line - Seguir uma marcação de pista



Objetivos:

- Se manter no centro da pista ★

- Direção Alvo ★



2. Conceitos e Aplicações

Visão Computacional

* Aplicações: Target (set) Point, Follow Line, Follow Me

Follow Me - Acompanhar um outro robô (com marcação visual)





3. Ferramentas

Visão Computacional

* OPENCV

OpenCV is a computer vision library originally developed by Intel.
It is free for commercial and research use under the open source BSD license.

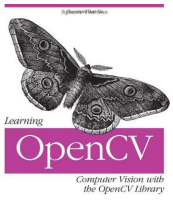
The library is cross-platform, and runs on Windows, Mac OS X, Linux, PSP, VCRT (Real-Time OS on Smart camera) and other embedded devices.

It focuses mainly on real-time image processing, as such, if it finds Intel's Integrated Performance Primitives on the system, it will use these commercial optimized routines to accelerate itself.

Refs: http://en.wikipedia.org/wiki/Opencv
http://opencv.willowgarage.com/wiki/FullOpenCVWiki
http://opencvlibrary.sourceforge.net/

http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/index.html
http://www.tecgraf.puc-rio.br/~malf/opencv/index.htm
http://www.comp.leeds.ac.uk/vision/opencv/

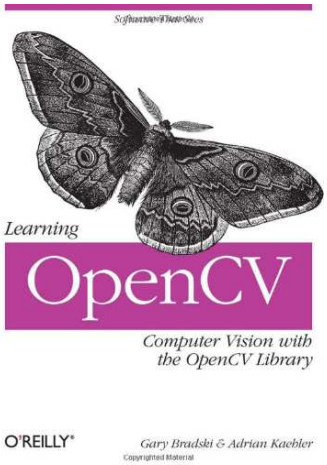




3. Ferramentas

Visão Computacional

* **OPENCV – Referência...**



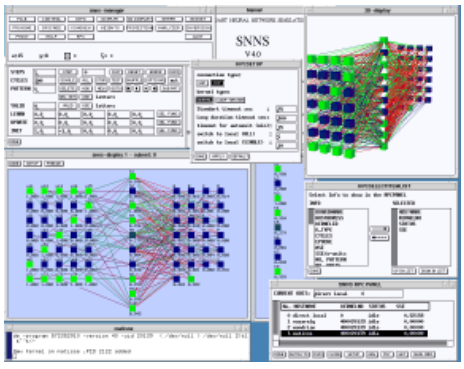
Conteúdo:

- Conceitos de Processamento de Imagens
- Funcionamento do OpenCV
- Exemplos de programas

3. Ferramentas

Visão Computacional e Machine Learning

* **SNNS e JavaNNS**



Aprendizado de Máquina:
Rede Neural Artificial

Aplicação:
-Reconhecimento de Padrões

Vantagens:

> Batchman
Linguagem de Scripts

> SNNS2C
Compilador que transforma Redes Neurais em Programas em "C" (Inserção em programas)

Refs: http://www.ra.cs.uni-tuebingen.de/SNNS/
http://www.ra.cs.uni-tuebingen.de/downloads/JavaNNS/

http://www.ra.cs.uni-tuebingen.de/software/welcome_e.html

4. Usando o OpenCV

OpenCV

* **Exemplo: Exibição de Arquivos de Imagens**

```
//  Carrega e exibe uma imagem em uma janela na tela
//  O nome do arquivo é passado como parâmetro ao programa
//  através da linha de comando

#include <highgui.h>

int main( int argc, char** argv )
{
    IplImage* img = cvLoadImage( argv[1] );
    cvNamedWindow( "Example1", CV_WINDOW_AUTOSIZE );
    cvShowImage( "Example1", img );
    cvWaitKey(0);
    cvReleaseImage( &img );
    cvDestroyWindow( "Example1" );
}
```


4. Usando o OpenCV

OpenCV

* **Exemplo: Exibição de Filmes**

```
//  Carrega e exibe uma filme AVI na tela
//  O nome do arquivo é passado como parâmetro
#include <highgui.h>
int main( int argc, char** argv )
{
    IplImage* frame;
    cvNamedWindow( "Example2", CV_WINDOW_AUTOSIZE );
    CvCapture* capture = cvCreateFileCapture( argv[1] );
    while(1) {
        frame = cvQueryFrame( capture );
        if( !frame ) break;
        cvShowImage( "Example2", frame );
        char c = cvWaitKey(33);
        if( c == 27 ) break;
    }
    cvReleaseCapture( &capture );
    cvDestroyWindow( "Example2" );
}
```

4. Usando o OpenCV

OpenCV

* **Exemplo:** Exibição de imagens capturadas da câmera

```
// ...
int main( int argc, char** argv )
{
  IplImage* frame, *frame_copy = 0;
  CvCapture* capture = 0;
  capture = cvCaptureFromCAM(0);
  cvNamedWindow( "result", 1 );
  for(;;)
  {
     if( !cvGrabFrame( capture ))
          break;
     frame = cvRetrieveFrame( capture );
     if( !frame )
          break;
   //...
  }
  // ...
}
```


4. Usando o OpenCV

OpenCV

* **Exemplo:** Alterando uma Imagem (exemplo 1)

```
// ...

// get the image data
    height    = frame->height;
    width     = frame->width;
    step      = frame->widthStep;
    channels  = frame->nChannels;
    data = (uchar *)frame->imageData;
    printf("Processing a %dx%d image with %d channels\n",
           height,width,channels);

//  invert the image
    for(i=0;i<height;i++) for(j=0;j<width;j++) for(k=0;k<channels;k++)
        data[i*step+j*channels+k]=255-data[i*step+j*channels+k];

// ...

//  Nota: Channels na ordem BGR
```

4. Usando o OpenCV

OpenCV

* **Exemplo: Alterando uma Imagem (exemplo 2)**

```
// ...
 for(i=0;i<height;i++) for(j=0;j<width;j++)
        {
                if ((data[i*step+j*channels+0] >= 240) &&
                    (data[i*step+j*channels+1] >= 240) &&
                        (data[i*step+j*channels+2] >= 240))
                {
                         data[i*step+j*channels+0]=255;
                         data[i*step+j*channels+1]=0;
                         data[i*step+j*channels+2]=0;
                }
                if ((data[i*step+j*channels+0] <= 50) &&
                    (data[i*step+j*channels+1] <= 50) &&
                        (data[i*step+j*channels+2] <= 50))
                {
                         data[i*step+j*channels+0]=0;
                         data[i*step+j*channels+1]=0;
                         data[i*step+j*channels+2]=255;
                }
        }
```


4. Usando o OpenCV

OpenCV

* **Exemplo: Desenhando na Tela - Line**

```
#include <highgui.h>

int main( int argc, char** argv )
{
    int i;
    IplImage* img = cvLoadImage( argv[1] );

    cvNamedWindow( "Example1", CV_WINDOW_AUTOSIZE );

    for (i=0; i<=300; i=i+3)
        cvLine(img, cvPoint(0,0),cvPoint(100,i),CV_RGB(255,0,0),1,8,0);

    cvShowImage( "Example1", img );

    cvWaitKey(0);
    cvReleaseImage( &img );
    cvDestroyWindow( "Example1" );
}
```

4. Usando o OpenCV

OpenCV

* **Exemplo:** Desenhando na Tela - Line

```
PRIMITIVAS:

Ponto:
  step     = Imagem->widthStep;
  Channels = Imagem->nChannels;
  Imagem [CoordY*step+CoordX*channels]  = Blue_Value;
  Imagem [CoordY*step+CoordX*channels+1]= Green_Value;
  Imagem [CoordY*step+CoordX*channels+2]= Red_Value;

Linha:
cvLine(img, cvPoint(xi,yi),cvPoint(xf,yf),CV_RGB(VRed,VGreen,VBlue),1,8,0);

Retângulo:
cvRectangle(img, cvPoint(xi,yi), cvPoint(xf,yf), CV_RGB(VR,VG,VB),1, 8, 0);

Circle:
cvCircle(img, cvPoint(xc,yc),Raio_int, CV_RGB(VR,VG,VB),1, 8, 0);

Outras primitivas:
cvEllipse, cvFillPoly, cvPolyLine, cvPutText, ...
```


4. Usando o OpenCV

OpenCV

* **Exemplo:** ROI – *Region of Interest*

```
#include <highgui.h>

int main( int argc, char** argv )
{
    int i;
    IplImage* img;

    img = cvLoadImage( argv[1] );
    cvSetImageROI(img,cvRect(150,150,200,200));

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

    cvResetImageROI(img);
    cvNamedWindow( "Example1", CV_WINDOW_AUTOSIZE );
    cvShowImage( "Example1", img );

    cvWaitKey(0);
    cvReleaseImage( &img );
    cvDestroyWindow( "Example1" );
    cvDestroyWindow( "ROI");
}
```

4. Usando o OpenCV

OpenCV

* **Exemplo: Salva em disco a ROI – *Region of Interest***

```
#include <highgui.h>

int main( int argc, char** argv )
{
    int i;
    IplImage* img;

    img = cvLoadImage( argv[1] );
    cvSetImageROI(img,cvRect(150,150,200,200));

    cvNamedWindow( "ROI", CV_WINDOW_AUTOSIZE );
    cvShowImage( "ROI", img);
    cvSaveImage("file.jpg",img);

    cvResetImageROI(img);
    cvNamedWindow( "Example1", CV_WINDOW_AUTOSIZE );
    cvShowImage( "Example1", img );

    cvWaitKey(0);
    cvReleaseImage( &img );
    cvDestroyWindow( "Example1" );
    cvDestroyWindow( "ROI");
}
```


5. Prática de uso do OpenCV

OpenCV

* **Exercícios...**

1. **Crie um programa para segmentar uma cor específica**
   Exemplo: identificar uma marca de uma cor específica (e.g. azul)

2. **Crie um programa que identifique a região onde se encontra esta cor**
   Exemplo: identificar os extremos xi, yi e xf, yf da região que contém uma determinada cor (envelope/bounding box)

3. **Crie um programa que identifique o ponto central desta região**
   Exemplo: x médio, y médio

4. **Crie um programa que indique para o usuário o quanto o ponto médio da região segmentada está deslocado do centro da tela.**





INFORMAÇÕES SOBRE A DISCIPLINA

USP - Universidade de São Paulo - São Carlos, SP
ICMC - Instituto de Ciências Matemáticas e de Computação
SSC - Departamento de Sistemas de Computação

LRM - Profs. DENIS Wolf, Eduardo SIMÕES, Fernando OSÓRIO
Web LRM: Http://lrm.icmc.usp.br/
Página pessoal: Http://www.icmc.usp.br/ ~denis/ ~simoes/ ~osorio/
E-mail: { denis, simoes, fosorio } [at] icmc. usp. br

Disciplina de Sensores Inteligentes
Web Disciplinas: Http://www.icmc.usp.br/~fosorio/
Web Wiki: Http://wiki.icmc.usp.br/index.php/Ssc-715(Fosorio)
> Programa, Material de Aulas, Critérios de Avaliação,
> Material de Apoio, Trabalhos Práticos