Applicazioni pratiche della visione artificiale

Parte 1.7 – OpenCV

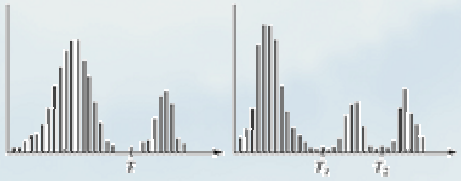
Curato da: Ing. Francesco La Rosa
Università di Messina – Facoltà di Ingegneria

Corso di Calcolatori II – A.A. 2003/2004
Ing. Giancarlo Iannizzotto

Thresholding

- Può essere vista come un'operazione di confronto tra i valori dei pixel dell'immagine ed una funzione T avente la forma: $T = T[x, y, p(x, y), f(x, y)]$

  dove f(x,y) è il livello di grigio di un punto e p(x,y) indica qualche proprietà locale (ad es.: la media dei livelli di grigio di un intorno centrato in (x,y)).

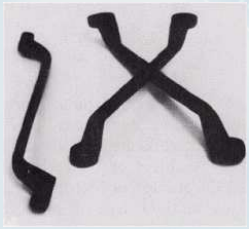


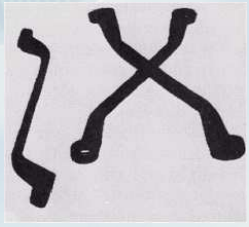
VisiLAB Computer Vision and Image Processing Lab - University of Messina

Thresholding

- L'operazione di sogliatura restituisce un'immagine binaria g(x,y) così definita:

$$g(x,y) = \begin{cases} 1 & if \quad f(x,y) > T \\ 0 & if \quad f(x,y) \le T. \end{cases}$$





Thresholding

- I pixel etichettati con 1 corrispondono ai punti dell'oggetto mentre i pixel etichettati con 0 ai punti di background.
- Quando la funzione T è unica per tutta l'immagine la threshold è chiamata globale, nel caso in cui dipenda anche da p(x,y) locale. Se T dipende dalle coordinate (x,y) è chiamata dinamica.

Thresholding Function (IPL)

void iplThreshold(IplImage* srcImage, IplImage*dstImage, int threshold);

srcImage The source image.

dstImage The resultant image.

threshold The threshold value to use for each pixel. The pixel value in the output is set to the maximum presentable value if it is greater than or equal to the threshold value (for each channel). Otherwise the pixel value in the output is set to the minimum presentable value.

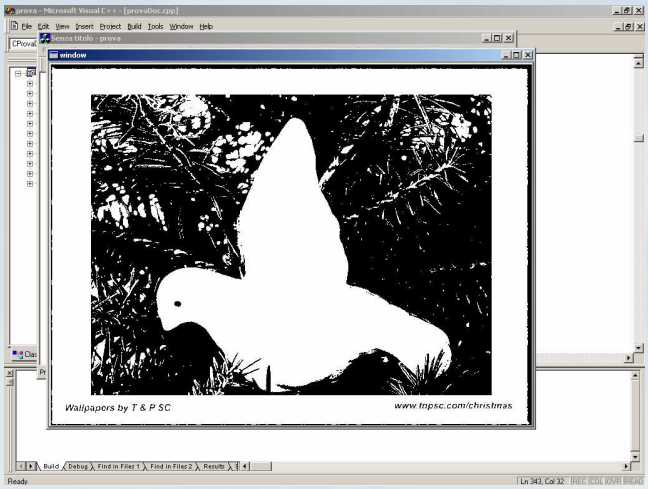
Thresholding Function (IPL)

Discussion

The function iplThreshold() thresholds the source image srcImage using the value threshold to create the resultant image dstImage. The pixel value in the output is set to the maximum presentable value (for example, 255 for an 8-bit-per-channel image) if it is greater than or equal to the threshold value. Otherwise it is set to the minimum presentable value (for example, 0 for an 8-bit-per-channel image). This is done for each channel in the input image.

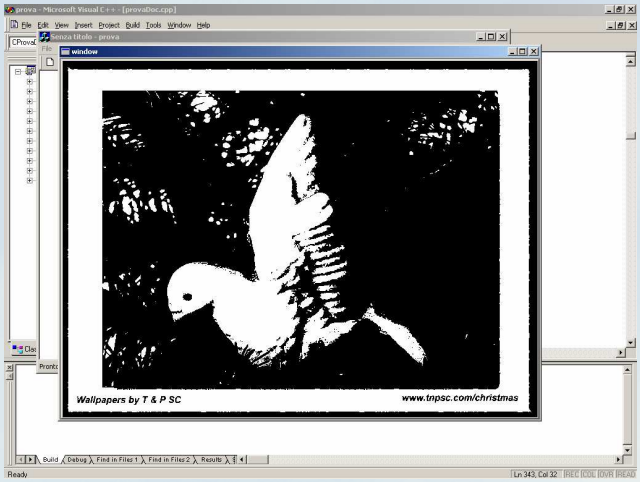
Thresholding di un'immagine

Thresholding di un'immagine



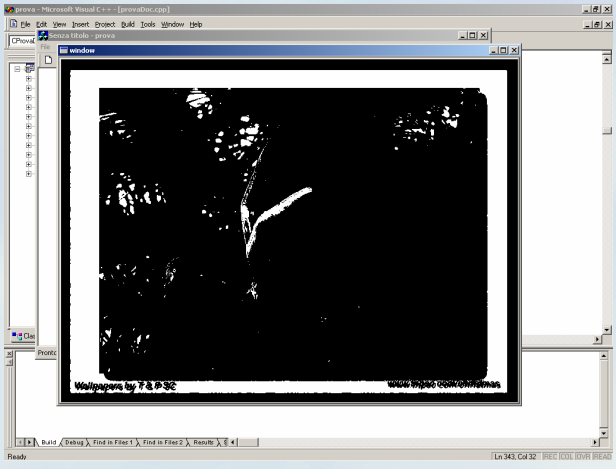
Threshold=50

Thresholding di un'immagine



Threshold=127

Thresholding di un'immagine



Threshold=200

Thresholding Function (openCV)

void cvThreshold (IplImage* *src*, IplImage* *dst*, float *thresh*, float *maxvalue*, CvThreshType *type*);

src Source image.

dst Destination image; can be the same as the parameter *src*.

thresh Threshold parameter.

maxvalue Maximum value; parameter, used with threshold types

CV_THRESH_BINARY, CV_THRESH_BINARY_INV, and

CV_THRESH_TRUNC.

Thresholding Function (openCV)

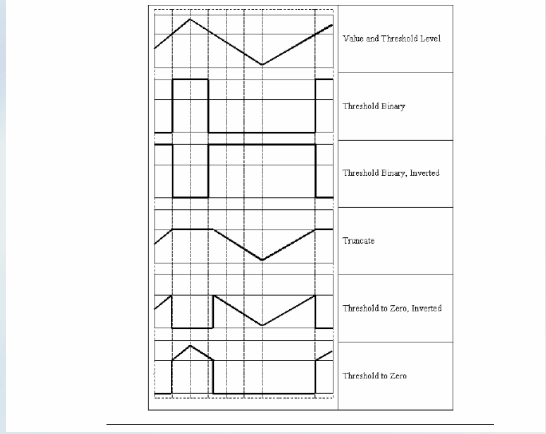
type Thresholding type;

must be one of:

- CV_THRESH_BINARY;
- CV_THRESH_BINARY_INV ;
- CV_THRESH_TRUNC;
- CV_THRESH_TOZERO ;
- CV_THRESH_TOZERO_INV .

Thresholding Function (openCV)

Figure 10-4 Meanings of Threshold Types



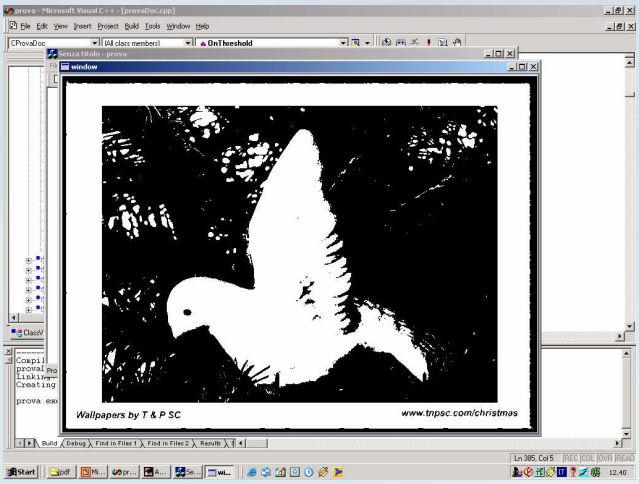
Thresholding Function (openCV)

Discussion

The function Threshold applies fixed-level thresholding to grayscale image. The result is either a grayscale image or a bi-level image.

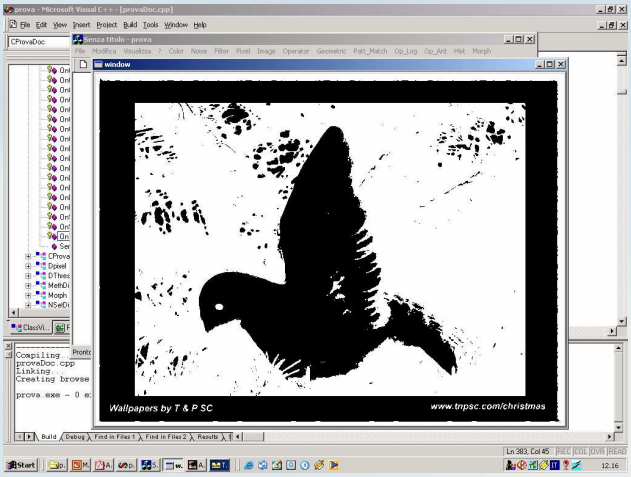
Thresholding
CV_THRESH_BINARY

Tsh=100



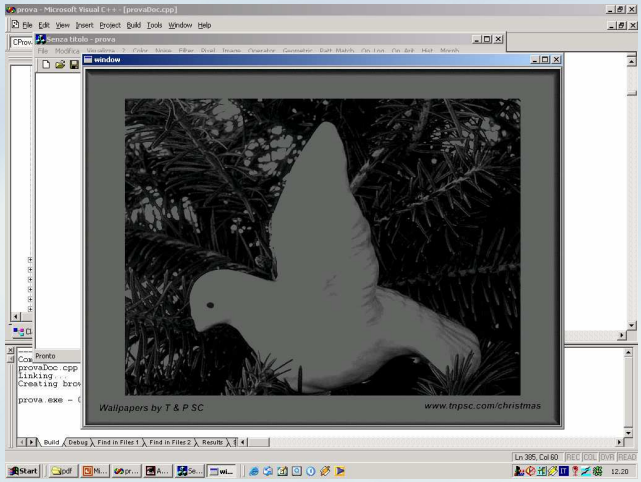
Thresholding
CV_THRESH_BINARY_INV

Tsh=100



Thresholding
CV_THRESH_TRUNC

Tsh=100

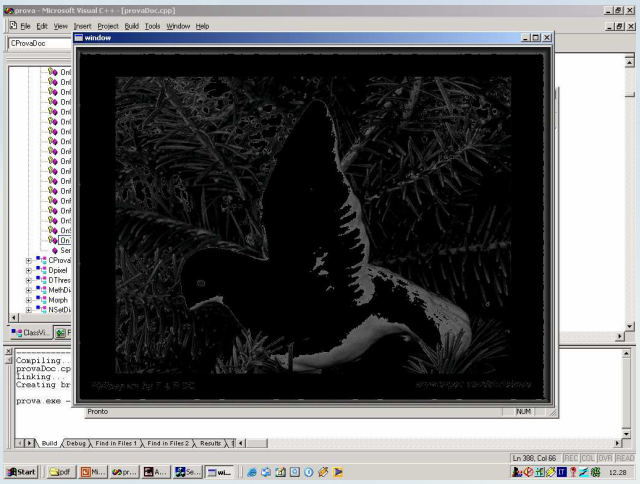


Thresholding
CV_THRESH_TOZERO

Tsh=100

Thresholding
CV_THRESH_TOZERO_INV

Tsh=100



Thresholding

- Se l'illuminazione della scena è sufficientemente uniforme, gli oggetti possono essere considerati flat (riflettività quasi costante) e ben contrastati allora è possibile estrarre l'oggetto dallo sfondo scegliendo come threshold un opportuno valore di intensità.
- L'operazione di thresholding descritta comporta la conversione dell'immagine grey-scale iniziale in un'immagine binaria che presenta un oggetto nero su sfondo bianco (o il viceversa).
- Rimane aperto il problema di trovare una procedura automatica che consenta di trovare un livello di thresholding ottimo.

Finding a suitable threshold

- In situazioni simili a quella dell'*optical character recognition* (OCR) per cui la proporzione tra il numero di pixel di background e quelli di foreground è sostanzialmente costante è possibile effettuare un'analisi statistica (preliminare) che consente di dedurre un fattore di proporzionalità (probabilità a priori) utile nella scelta del livello di thresholding.
- Si tratta di una tecnica particolarmente sensibile al rumore.
- Tale tecnica è spesso utile in applicazioni industriali.

Finding a suitable threshold

```
…
char wndname01[] = "Source image";
char wndname02[] = "Histogram";
int dims[1] = {DIMENSION_SIZE};
float thresh[1][2] = { {0, 256} };
float* pthresh[1] = { thresh[0] };

float prop=0.5;
IPLIMAGE image01 = cvLoadImage( path );
//alias per IplImage*
//highgui.h
//typedef IplImage* IPLIMAGE;
…
```

Finding a suitable threshold

```
…
// Create the destination images. HighGUI use.
IplImage * image02 = cvCreateImage( cvSize(image01->width,image01->height),
                         IPL_DEPTH_8U, 1);
// Make one channel image.
cvCvtColor(image01,image02, CV_BGR2GRAY);
//iplColorToGray(image01,image02);
//Created histogram headers. OpenCV use.
CvHistogram* hist01 = cvCreateHist(1,    // Histogram dimension number.
                         dims,  // Dimension size array.
                         CV_HIST_ARRAY,   // Histogram type.
                         pthresh, 1);
…
```


Finding a suitable threshold

```
…
// Create windows. HighGUI use.
cvNamedWindow( wndname01, CV_WINDOW_AUTOSIZE);
cvNamedWindow( wndname02, CV_WINDOW_AUTOSIZE );
// Show the image. HighGUI use.
cvShowImage( wndname01, image02 );
// Calculated the histogram. OpenCV use.
cvCalcHist( &image02,  // Source images.
          hist01,    // Pointer to the histogram.
          0, 0 );      // Clear flag.
float prop_number=(image02->width)*(image02->height)*prop;
float bins; float sum=0;
…
```

Finding a suitable threshold

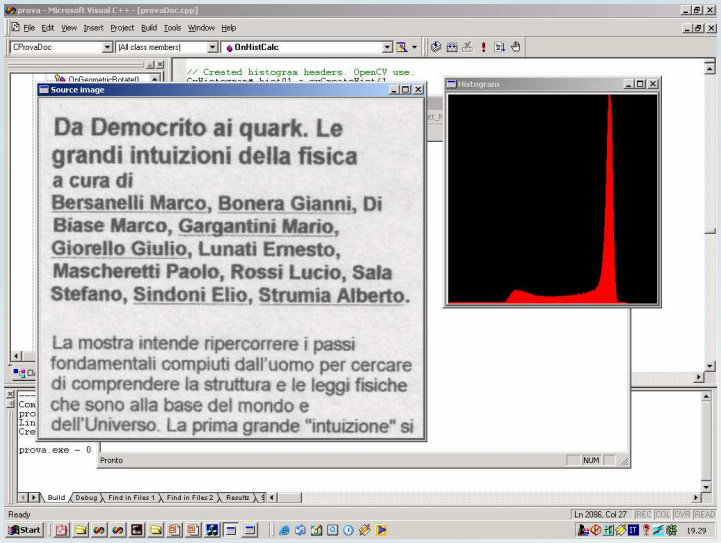
```
…
for(int i=0; i<=DIMENSION_SIZE; i++)
   {
      // The function cvQueryHistValue_1D() returns the
      // value of the specified histogram bin.
      bins=cvQueryHistValue_1D(hist01,i);
      sum+=bins;
      if(sum>=prop_number) break;
   }
int th=i;
char pixelc[10];  CString str="Il valore della soglia è: ";
_itoa(th, pixelc,10); str=str+pixelc; AfxMessageBox(str);
…
```


Finding a suitable threshold

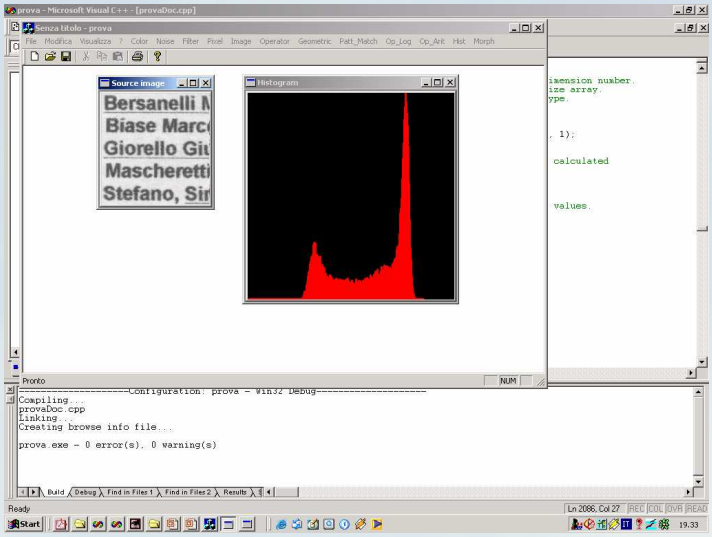
```
…
cvThreshold(image02, image02, th, 255,CV_THRESH_BINARY);
cvShowImage( wndname02, image02 );
cvWaitKey(0);

cvDestroyWindow(wndname01);
cvDestroyWindow(wndname02);
cvReleaseImage(&image01);
cvReleaseImage(&image02);
cvReleaseHist( &hist01 );
…
```

Threshold & Histogram

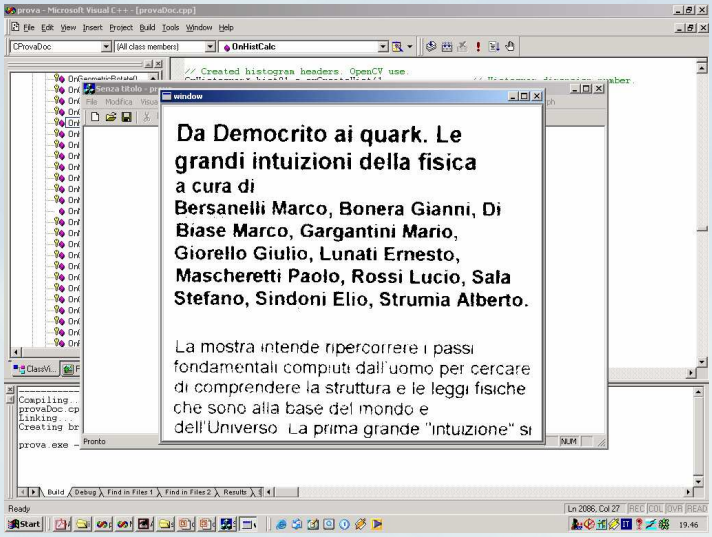


Threshold & Histogram



Threshold & Histogram

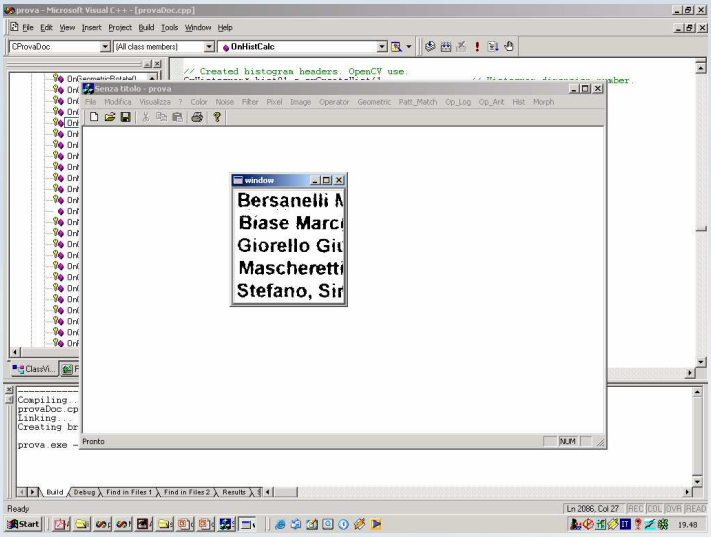
prop=0.5



parte 1.7- OpenCV

Threshold & Histogram

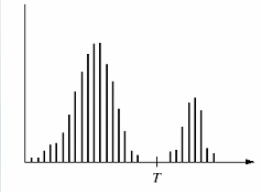
prop=0.5



VisiLAB Computer Vision and Image Processing Lab - University of Messina

Finding a suitable threshold

- La tecnica più usata per la scelta del livello di threshold ricorre all'analisi dell'istogramma dei livelli d'intensità dell'immagine.
- Se viene trovato un minimo significativo questi si assume essere il livello di thresholding cercato.
- Si assume che l'istogramma presenti una distribuzione bimodale.



Finding a suitable threshold

- Il metodo descritto presenta le seguenti difficoltà:
  - La valle tra le due campane può essere così larga da rendere difficile individuare un minimo significativo;
  - Possono esserci più minimi a causa dei dettagli presenti nell'immagine;
  - Rumore localizzato nella valle può inibire il minimo;
  - Possono non esserci minimi apprezzabili nella valle a causa di rumore eccessivo o illuminazione del background particolarmente variabile;
  - Il picco maggiore dell'istogramma (di solito il background) può essere più largo dell'altro e questo determina un bias della posizione del minimo;
  - L'istogramma può essere per sua natura multimodale.

Threshold & Segmentazione

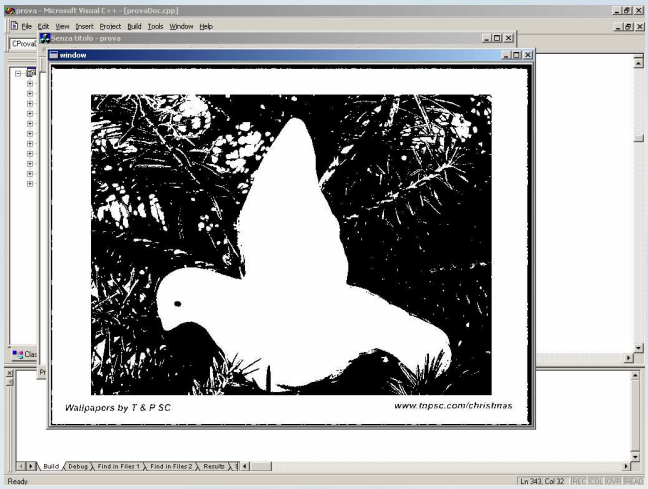
- Thresholding non significa segmentazione!!!
  - Il risultato dell'operazione di thresholding è ancora un'immagine binaria in cui i pixel non sono aggregati in regioni.
- E' quindi necessario operare una ricerca dei connected components nell'immagine binaria in modo da ottenere regioni connesse, che sono il prodotto della segmentazione.

Segmentazione & Thresholding

```
…
iplColorToGray(src,src_gray);
cvThreshold( src_gray,src_gray, 50, 255,
                    CV_THRESH_BINARY);
// find contours and store them all as a list
cvFindContours( src_gray, storage, &contours, sizeof(CvContour),
CV_RETR_LIST, CV_CHAIN_APPROX_SIMPLE );
iplSet(dst,255);
// disegno i contorni di primo livello trovati
cvDrawContours(dst,contours,0,0,1);
…
```

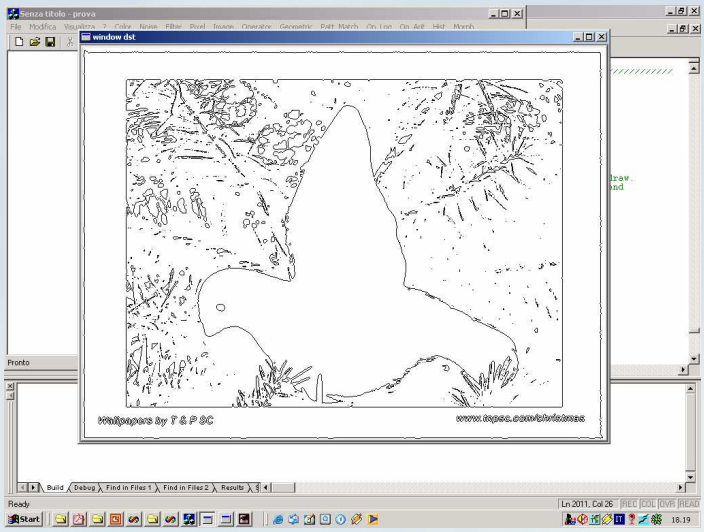
Thresholding di un'immagine

Thresholding di un'immagine



Threshold=50

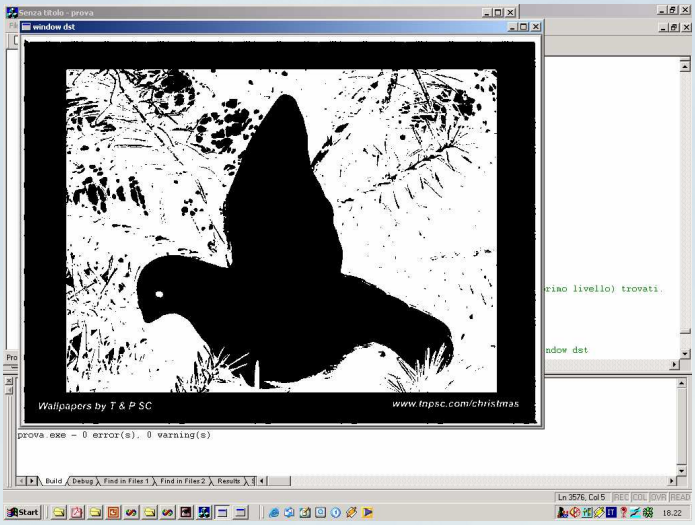
cvFindContours



Segmentazione & Thresholding

```
…
iplColorToGray(src,src_gray);
cvThreshold( src_gray,src_gray, 50, 255,
                    CV_THRESH_BINARY);
// find contours and store them all as a list
cvFindContours( src_gray, storage, &contours, sizeof(CvContour),
CV_RETR_LIST, CV_CHAIN_APPROX_SIMPLE );
iplSet(dst,255);
//effettuo il filling delle regioni delimitate dai contorni trovati.
cvDrawContours(dst,contours,0,0,1,-1);
…
```

cvFindContours & Filling



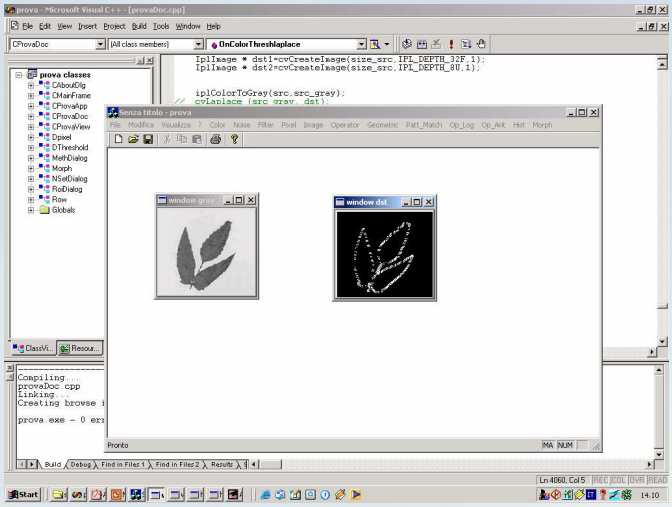
Threshold & caratteristiche di bordo

- La scelta della threshold è favorita da istogrammi aventi picchi alti, stretti, simmetrici e ben separati.
- Un modo per migliorare la forma degli istogrammi è quello di considerare solo i pixel che stanno, o sono vicini, al confine tra oggetto e background.

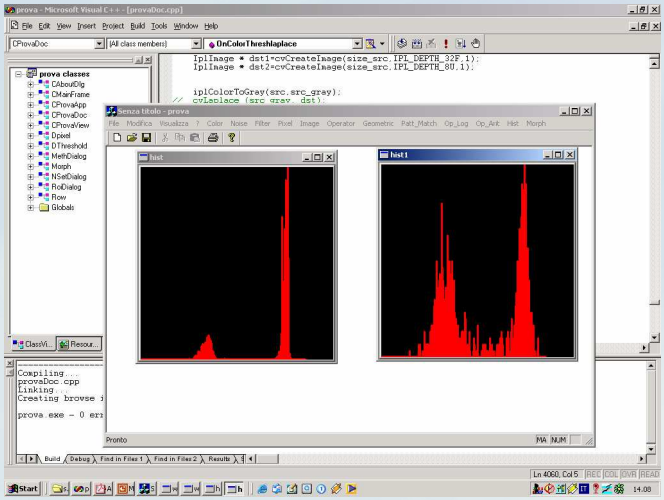
Threshold & caratteristiche di bordo

I pixel che stanno, o sono vicini, al confine tra oggetto e background possono essere individuati scegliendo pixel dell'immagine aventi un valore elevato di gradiente.

Threshold & caratteristiche di bordo



Threshold & caratteristiche di bordo



Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
src=cvLoadImage(path);
CvSize size_src=cvSize(src->width,src->height);
src_gray=cvCreateImage(size_src,src->depth,1);
dst=cvCreateImage(size_src,IPL_DEPTH_16S,1);
IplImage * dst1;
dst1=cvCreateImage(size_src,IPL_DEPTH_32F,1);
IplImage * dst2;
dst2=cvCreateImage(size_src,IPL_DEPTH_8U,1);
…
```

Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
iplColorToGray(src,src_gray);
cvSobel (src_gray, dst, 1, 1);
iplAbs(dst, dst);
cvConvertScale (dst, dst1, 1.0,0.0);
float minVal1;
float maxVal1;
iplMinMaxFP (dst1, &minVal1, &maxVal1);
float value=255/maxVal1;
…
```


Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
iplMultiplySFP(dst1, dst1, value);
cvConvertScale (dst1, dst2, 1.0,0.0);
int threshold;
//acquisisco il valore di threshold da una dialog box …
cvThreshold(dst2, dst2, threshold, 255,  CV_THRESH_BINARY);
cvNamedWindow("window dst", CV_WINDOW_AUTOSIZE);
cvShowImage("window dst", dst2);
…
```

Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
// apro una finestra denominata window gray
cvNamedWindow("window gray",
                        CV_WINDOW_AUTOSIZE);
cvShowImage("window gray", src_gray);
int dims[1] = {DIMENSION_SIZE};
float thresh[1][2] = { {0, 255} };
float* pthresh[1] = { thresh[0] };
IplImage * image03= cvCreateImage(cvSize(300,300),
                                        IPL_DEPTH_8U, 3);
…
```


Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
//Created histogram headers. OpenCV use.
CvHistogram* hist01 = cvCreateHist(1, dims,
CV_HIST_ARRAY, pthresh, 1);
cvNamedWindow( "hist", CV_WINDOW_AUTOSIZE );
//Calculated the histogram. OpenCV use.
cvCalcHist( &src_gray, hist01, 0, 0 );
//Normalize and Draw histogram.
int i; CvPoint pt1, pt2; pt2.x=0;
…
```

Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
pt2.y=(image03->height -1);
pt1.x=0; pt1.y=0;
cvSetZero(image03);
float minVal[1], maxVal[1];
int minIdx[1], maxIdx[1];
cvGetMinMaxHistValue (hist01, minVal, maxVal, minIdx,
                      maxIdx);
float bins;
…
```


Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
// Draw histogram.
 for(i=0; i<DIMENSION_SIZE; i++)
 {
    pt1.x=pt1.x+1;
    //The function cvQueryHistValue_1D() returns the
    //value of the specified histogram bin.
    bins=cvQueryHistValue_1D(hist01,i);
    int valore;
…
```

Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
    valore=int((image03>height)*(bins/maxVal[0]));
     pt1.y=image03->height - valore;
     pt2.x=pt2.x+1;
     cvLine(image03, pt1, pt2, CV_RGB(255,0,0), 2,
                              8);
  }
 unsigned char pixel[1], pixel_mask[1];
 int istogramma[256];
 …
```


Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
 for(i=0; i<256; i++)  istogramma[i]=0;
 for(i=0; i<src_gray->height; i++)
  {     for(int j=0; j<src_gray->width; j++)
        {
          iplGetPixel(dst2,j,i,pixel_mask);
          if(pixel_mask[0]>127)
          {
            iplGetPixel(src_gray,j,i,pixel);
…
```

Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
istogramma[pixel[0]]++;
        }
      }
 }
 int maximum=istogramma[0];
 for(i=1;i<256;i++)
 if(istogramma[i]>maximum)     maximum=istogramma[i];
 …
```


Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
cvNamedWindow( "hist1", CV_WINDOW_AUTOSIZE);
IplImage * image04 = cvCreateImage(    cvSize(300,300),
                                IPL_DEPTH_8U, 3);
cvSetZero(image04);
pt2.x=0;   pt2.y=(image04->height -1);
pt1.x=0; pt1.y=0;
…
```

Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
for(i=0; i<DIMENSION_SIZE; i++)
{  pt1.x=pt1.x+1;
 // The function cvQueryHistValue_1D() returns the
 // value of the specified histogram bin.
    bins=istogramma[i];
   int valore;
   valore=int((image04>height)*(bins/(float)maximum));
   pt1.y=image04->height - valore;
 …
```


Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
pt2.x=pt2.x+1;
cvLine(image04, pt1, pt2, CV_RGB(255,0,0), 2, 8);
}
// Show the image. HighGUI use.
cvShowImage( "hist", image03 );
cvShowImage( "hist1", image04 );
cvWaitKey(0);
…
```

Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
cvDestroyWindow("window dst");
cvDestroyWindow("window gray");
cvDestroyWindow("hist");
cvDestroyWindow("hist1");
…
```


Threshold & caratteristiche di bordo

Uso del filtro di Sobel

```
…
cvReleaseImage (&src);
cvReleaseImage (&src_gray);
cvReleaseImage (&dst);
cvReleaseImage (&dst1);
cvReleaseImage (&dst2);
cvReleaseImage (&image03);
cvReleaseImage (&image04);
…
```

Threshold & caratteristiche di bordo

- Questo tipo di istogrammi è influenzato meno dalla diversa dimensione di oggetto e background: i picchi hanno ampiezza simile.
- La probabilità che un pixel appartenga all'oggetto è prossima a quella che appartenga al background: i picchi sono simmetrici.
- Per la rivelazione dei pixel prossimi al contorno degli oggetti si può usare una tecnica che fa uso del gradiente e del Laplaciano.

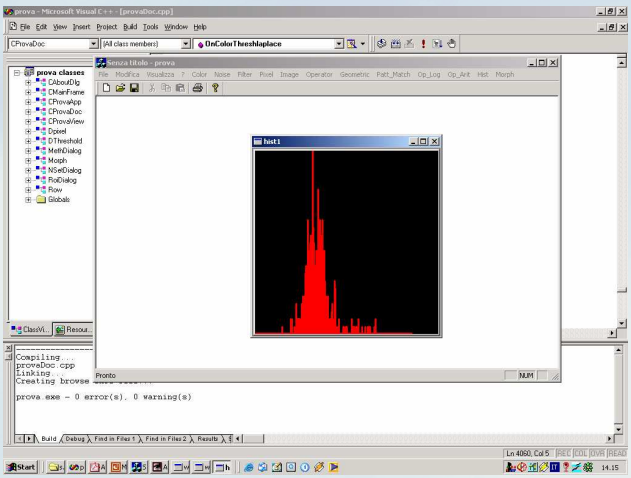
Threshold & caratteristiche di bordo

- L'applicazione dell'operatore gradiente o del Laplaciano su un'immagine ha la tendenza ad accentuare la separazione tra i picchi dell'istogramma.
- In particolare, i pixel di edge possono essere rivelati tramite l'uso del gradiente, mentre lo studio del segno del Laplaciano consente di rivelare se i pixel restituiti dallo step precedente appartengono ad un oggetto o al background.

Threshold & caratteristiche di bordo

Sobel & Laplaciano

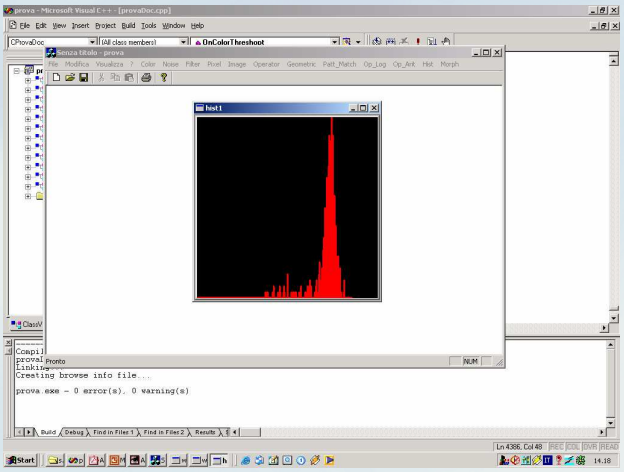
Istogramma dei pixel di oggetto



Threshold & caratteristiche di bordo

Sobel & Laplaciano

Istogramma dei pixel di sfondo



Threshold & caratteristiche di bordo

Sobel & Laplaciano

```
…
src=cvLoadImage(path);
CvSize size_src=cvSize(src->width,src->height);
src_gray=cvCreateImage(size_src,src->depth,1);
dst=cvCreateImage(size_src,IPL_DEPTH_16S,1);
IplImage * dst1, * dst2, * dsts, * dst32;
dst1=cvCreateImage(size_src,IPL_DEPTH_32F,1);
dst2=cvCreateImage(size_src,IPL_DEPTH_8U,1);
dsts=cvCreateImage(size_src,IPL_DEPTH_16S,1);
dst32=cvCreateImage(size_src,IPL_DEPTH_32F,1);
iplColorToGray(src,src_gray);
cvLaplace (src_gray, dsts); …
```


Threshold & caratteristiche di bordo

Sobel & Laplaciano

```
…
cvConvertScale (dsts, dst32, 1.0,0.0);
cvSobel (src_gray, dst, 1, 1);
iplAbs(dst, dst);
cvConvertScale (dst, dst1, 1.0,0.0);
float minVal1, maxVal1;
iplMinMaxFP (dst1, &minVal1, &maxVal1);
float value=255/maxVal1;
iplMultiplySFP(dst1, dst1, value);
cvConvertScale (dst1, dst2, 1.0,0.0);
int threshold;
…
```

Threshold & caratteristiche di bordo

Sobel & Laplaciano

```
…
cvThreshold(dst2, dst2,threshold,255,CV_THRESH_BINARY);
cvNamedWindow("window dst", CV_WINDOW_AUTOSIZE);
cvShowImage("window dst",dst2);
cvNamedWindow("window gray", CV_WINDOW_AUTOSIZE);
cvShowImage("window gray",src_gray);
int bins,i; CvPoint pt1, pt2;
unsigned char pixel[1], pixel_mask[1];
float pixel_s[1];
int istogramma[256];
for(i=0; i<256; i++)  istogramma[i]=0; …
```


Threshold & caratteristiche di bordo

Sobel & Laplaciano

```
…
for(i=0; i<src_gray->height; i++)
{  for(int j=0; j<src_gray->width; j++)
      {
        iplGetPixel(dst2,j,i,pixel_mask);
        iplGetPixel(dst32,j,i,pixel_s);
        if((pixel_mask[0]>127)&&(pixel_s[0]<0))
        { //sfondo: pixel_s[0]<0, oggetto: pixel_s[0]>0
          iplGetPixel(src_gray,j,i,pixel);
          istogramma[pixel[0]]++;
        }
      }
 }
 …
```

Threshold & caratteristiche di bordo

Sobel & Laplaciano

```
…
int maximum=istogramma[0];
for(i=1;i<256;i++)
if(istogramma[i]>maximum) maximum=istogramma[i];
cvNamedWindow( "hist1", CV_WINDOW_AUTOSIZE );
IplImage * image04 = cvCreateImage( cvSize(300,300),
                        IPL_DEPTH_8U, 3);
cvSetZero(image04);
pt2.x=0; pt2.y=(image04->height -1);
pt1.x=0; pt1.y=0;
…
```


Threshold & caratteristiche di bordo

Sobel & Laplaciano

```
…
for(i=0; i<DIMENSION_SIZE; i++)
{
   pt1.x=pt1.x+1; bins=istogramma[i];
   int valore=int((image04->height)*(bins/(float)maximum));
   pt1.y=image04->height - valore;
   pt2.x=pt2.x+1;
   cvLine(image04, pt1, pt2, CV_RGB(255,0,0), 2, 8);
}
cvShowImage( "hist1", image04 );
cvWaitKey(0);
//chiude le window aperte e libera la memoria allocata
…
```

Threshold basata su più variabili

- In alcuni casi i sensori a disposizione forniscono immagini in cui i pixel sono caratterizzati da più variabili (ad es.: immagini a colori di tipo RGB).
- Nel caso delle immagini a colori ogni pixel è caratterizzato da 3 variabili rendendo possibile la costruzione di istogrammi 3-D.
- In questi casi i pixel sono suddivisi in sottoinsiemi che prendono il nome di cluster. Pixel che hanno colore simile appartengono allo stesso cluster.

Threshold basata su più variabili

- L'operazione di segmentazione restituisce un'immagine in cui viene assegnata la stessa intensità a pixel appartenenti al medesimo cluster.
- Intensità diverse sono associate a cluster diversi.

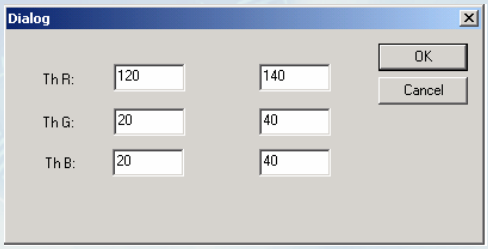
Threshold basata su più variabili

- In alcuni casi i sensori a disposizione forniscono immagini in cui i pixel sono caratterizzati da più variabili (ad es.: immagini a colori di tipo RGB).
- Nel caso delle immagini a colori ogni pixel è caratterizzato da 3 variabili rendendo possibile la costruzione di istogrammi 3-D.
- In questi casi i pixel sono suddivisi in sottoinsiemi che prendono il nome di cluster. Pixel che hanno colore simile appartengono allo stesso cluster.

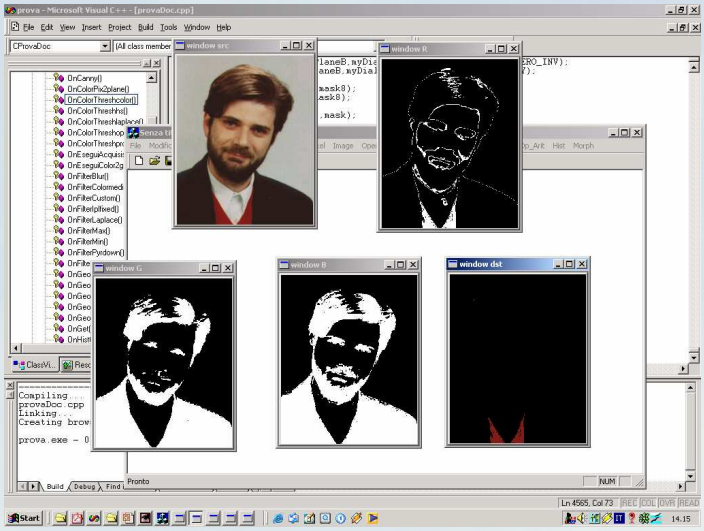
Threshold basata su più variabili

- L'operazione di segmentazione restituisce un'immagine in cui viene assegnata la stessa intensità a pixel appartenenti al medesimo cluster.
- Intensità diverse sono associate a cluster diversi.

Threshold basata su più variabili



Threshold basata su più variabili



Threshold basata su più variabili

```
…
IplImage *src=cvLoadImage(path);
CvSize size_src=cvSize(src->width,src->height);
IplImage *PlaneB=cvCreateImage(size_src,
                    src->depth,1);
IplImage *PlaneG=cvCreateImage(size_src,
                    src->depth,1);
IplImage *PlaneR=cvCreateImage(size_src,
                    src->depth,1);
IplImage *mask8=cvCreateImage(size_src,src->depth,1);
IplImage *mask;
mask=cvCreateImage(size_src,IPL_DEPTH_1U,1);
…
```

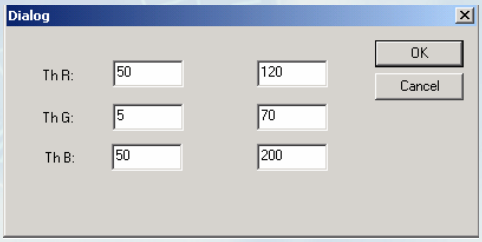

Threshold basata su più variabili

```
…
dst=cvCreateImage(size_src, IPL_DEPTH_8U, 3);
// codice per inserire da una dialog box i valori soglia
cvCvtPixToPlane(src, PlaneB, PlaneG, PlaneR,0);
cvThreshold(PlaneR, PlaneR, myDialog.m_HH, 255,
                CV_THRESH_TOZERO_INV);
cvThreshold(PlaneR, PlaneR, myDialog.m_H, 255,
                CV_THRESH_BINARY);
cvThreshold(PlaneG, PlaneG, myDialog.m_SS, 255,
                CV_THRESH_TOZERO_INV);
cvThreshold(PlaneG, PlaneG, myDialog.m_S, 255,
                CV_THRESH_BINARY);
…
```

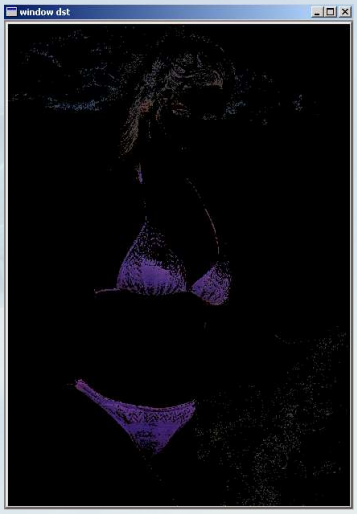
Threshold basata su più variabili

```
…
cvThreshold(PlaneB, PlaneB, myDialog.m_VV, 255,
                CV_THRESH_TOZERO_INV);
cvThreshold(PlaneB, PlaneB, myDialog.m_V, 255,
                CV_THRESH_BINARY);
iplAnd(PlaneR,PlaneG,mask8);
iplAnd(PlaneB,mask8,mask8);
iplGreaterS(mask8,127,mask);
src->maskROI=mask;
iplCopy(src,dst);
…
```


Threshold basata su più variabili



Threshold basata su più variabili



Threshold della tinta

40< HUE <60

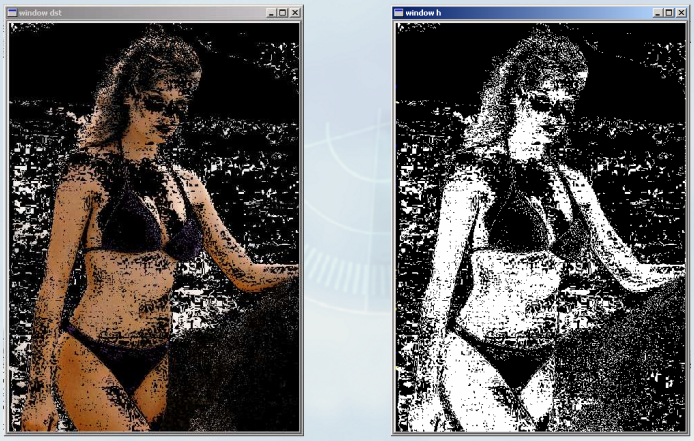
Threshold della tinta

```
…
//acquisisco un'immagine a colori
src=cvLoadImage(path);
//determino la size dell'immagine acquisita
CvSize size_src=cvSize(src->width,src->height);
dst=cvCreateImage(size_src, IPL_DEPTH_8U, 3);
IplImage * h=cvCreateImage(size_src,IPL_DEPTH_8U,1);
IplImage * s=cvCreateImage(size_src,IPL_DEPTH_8U,1);
IplImage * v=cvCreateImage(size_src,IPL_DEPTH_8U,1);
IplImage * mask;
mask=cvCreateImage(size_src,IPL_DEPTH_1U,1);
…
```


Threshold della tinta

```
…
dst=cvCreateImage(size_src,IPL_DEPTH_8U,3);
IplImage *dst1;
dst1=cvCreateImage(size_src,IPL_DEPTH_8U,3);
cvCvtColor(src,dst, CV_BGR2HSV);
//ricavo i piani H, S e V
cvCvtPixToPlane (dst, h, s, v, 0);
cvThreshold(h, h, myDialog.m_HH, 255,
            CV_THRESH_TOZERO_INV);
cvThreshold(h, h, myDialog.m_H, 255,
              CV_THRESH_BINARY);
iplGreaterS(h, 127, mask);  dst1->maskROI=mask;
iplCopy(src,dst1);  …
```

Threshold della tinta



40< HUE <60