Tarea 19

Angel Manrique Pozos Flores; N.C M07211505
Tecnológico Nacional de México,
Blvd. Industrial, Mesa de Otay, 22430 Tijuana, B.C., México.

19 de mayo de 2016

Realizar el algoritmo de segmentación por histogramas visto en clase.

1. Segmentación de imágenes

La segmentación de imágenes se refiere a la partición de imágenes en diferentes regiones que son homogéneas respecto a alguna característica de la imagen.

La técnica mas simple es la de segmentación por thresholding de histograma que asume que la imagen puede ser separada dependiendo de la cantidad de picos de las distintas regiones que están presentes en la imagen.

Lo que hace es dividir estas regiones en segmentos los cuales son pixeles conectados mediante un análisis del criterio de similaridad (Klette, 2014), otra forma de definir la segmentación de acuerdo a (Szeliski, 2010) la cual nos dice que es la tarea que es capaz de encontrar pixeles juntos, donde los pixeles van a compartir propiedades similares.

La segmentación tiene dos objetivos principales:

- Debe ser capaz de descomponer la imagen en n partes para poder realizar su análisis.
- Ser capaz de realizar un cambio en su representación.

Entonces la segmentación puede verse como la partición de una imagen que definimos como ω en un numero de segmentos finitos S_i tales que:

1. Si $\neq$ para cada i $\in \{1,\ldots,n\}$

2. $\cup_{i=1}^{n}, S_i = \omega$

3. $S_i \cap S_j = 0$ para todo $i, j \in \{1, \ldots, n\}$ con $i \neq j$

Existen varias técnicas de segmentación, las cuales pueden clasificarse en:

- Globales - Estas se basan en las propiedades globales de la imagen (división).
- Locales - Estas se basan en agrupar pixeles en base a sus atributos y los de sus vecinos (agrupamiento).
- División y agrupamiento - Estas combinan ambas técnicas, locales y globales.

2. Segmentación por histograma

La segmentación por histograma, también llamada thresholding es una técnica global la cual asume que existe un solo objeto sobre un fondo uniforme como se observa en la figura 1.

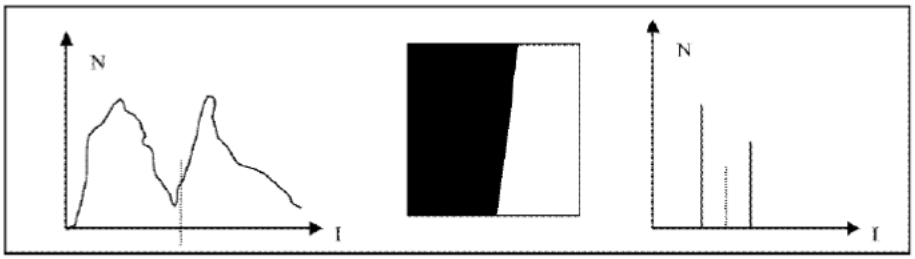

Figura 1: Segmentación por thresholding de dos regiones.

Esta técnica considera la intensidad comúnmente, sin tomar en cuenta la coherencia espacial de la región, por ello dos pixeles separados en la imagen pueden pertenecer a la misma región de la imagen como se observa en la figura 2.

Esta técnica de segmentación por histograma puede extenderse a N cantidad de regiones tomando en consideración cada pico del histograma como una región y los valles o mínimos como la separación entre estas regiones, como se observa en la figura 3.

El análisis en diferentes espacios de escala es necesario debido a que:

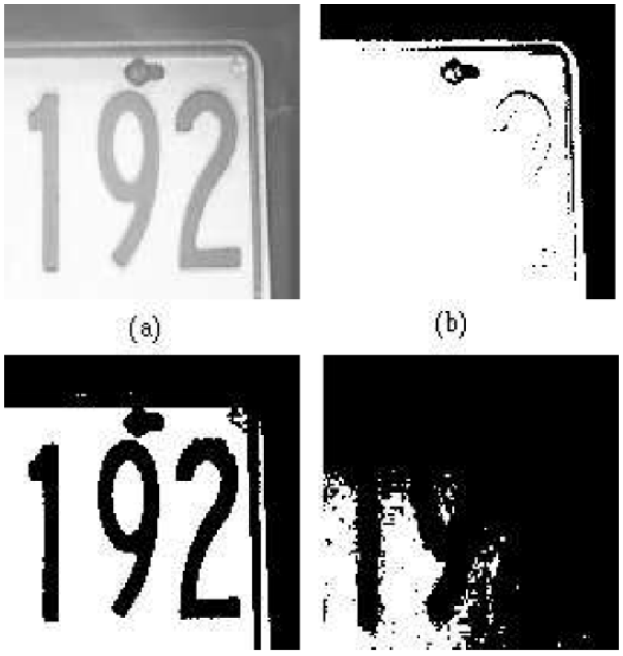

Figura 2: Coherencia espacial de una región las cuales pertenecen a una misma imagen.

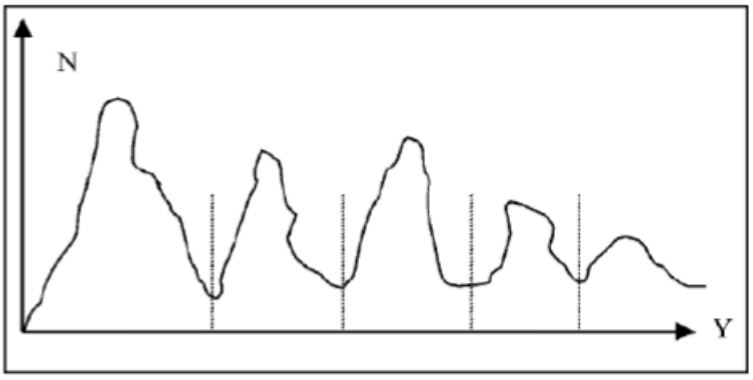

Figura 3: Segmentación de N regiones en base a los picos y valles de un histograma.

- Las estructuras y atributos presentes en una imagen existen a lo largo de un rango continuo de tamaños.
- El tamaño de los atributos específicos en una imagen no es conocido con anterioridad.
- Es posible seguir el surgimiento de estructuras a lo largo de las escalas y

utilizar estas técnicas para así obtener, un procesamiento independiente a la escala que sea ademas computacionalmente eficiente.

Para ello se desarrollo el siguiente código en C++ utilizando las librerías de visión de OpenCV.

```
#include <opencv2/opencv.hpp>
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include <iostream>
#include <stdio.h>
#include <vector>


using namespace std;
using namespace cv;



int main(int argc, char** argv)
{
        Mat src, hist, dst, dst2, dst3, color1;
        double min, max, maxloc, maxloc2, maxloc3;
        int minloc, minloc2;
        Point miniLoc, maxiLoc;

        src = imread("v.jpg", 1);
        cvtColor(src, src, CV_BGR2GRAY);
        imshow("Gray Scale", src);


        int histSize = 256;


        float range[] = { 0, 256 };
        const float* histRange = { range };
        bool uniform = true; bool accumulate = false;

        calcHist(&src, 1, 0, Mat(), hist, 1, &histSize
           , &histRange, uniform, accumulate);
```

```
//Localizar Picos
for (int i = 2; i < hist.rows-1; i++)
{
        if (((int)hist.at<float>(i)) > ((int)
           hist.at<float>(i-2)) && ((int)hist.
           at<float>(i)) > ((int)hist.at<float
           >(i - 1)) && ((int)hist.at<float>(i
           )) > ((int)hist.at<float>(i+1)) &&
           ((int)hist.at<float>(i)) > 1000)
        {
                maxloc = i;
                i = (hist.rows - 1);
        }
}
for (int i = maxloc+1; i < hist.rows - 1; i++)
{
        if (((int)hist.at<float>(i)) >((int)
           hist.at<float>(i - 2)) && ((int)
           hist.at<float>(i)) > ((int)hist.at<
           float>(i - 1)) && ((int)hist.at<
           float>(i)) > ((int)hist.at<float>(i
            + 1)) && ((int)hist.at<float>(i))
           > 1000)
        {
                maxloc2 = i;
                i = (hist.rows - 1);
        }
}
for (int i = maxloc2 + 1; i < hist.rows - 1; i
   ++)
{
        if (((int)hist.at<float>(i)) >((int)
           hist.at<float>(i - 2)) && ((int)
           hist.at<float>(i)) > ((int)hist.at<
           float>(i - 1)) && ((int)hist.at<
           float>(i)) > ((int)hist.at<float>(i
            + 1)) && ((int)hist.at<float>(i))
           > 1000)
        {
                maxloc3 = i;
```

```
                i = (hist.rows - 1);
        }
}

//Definir Valles
minloc = (int)(maxloc + ((maxloc2 - maxloc)/2)
    );
minloc2 = (int)(maxloc2 + ((maxloc3 - maxloc2)
    /2));

cout << "Pico 1: " << maxloc << endl << "Valle
     1: " << minloc << endl << "Pico 2: " <<
    maxloc2 << endl;
cout << "Valle 2: " << minloc2 << endl << "
    Pico 3: " << maxloc3 << endl;

// Seccion 1
threshold(src, dst, minloc, 255, 1);
imshow("Seccion 1", dst);

// Seccion 2
threshold(src, dst2, minloc2, 255, 4);
threshold(dst2, dst2, minloc, 255, 0);
imshow("Seccion 2", dst2);

// Seccion 3
threshold(src, dst3, minloc2, 255, 0);
imshow("Seccion 3", dst3);

//Convertir imagenes para agregar color
dst.convertTo(dst, CV_8UC1);
dst2.convertTo(dst2, CV_8UC1);
dst3.convertTo(dst3, CV_8UC1);

//Unir 3 canales para mostrar las imagenes a
    color
vector<Mat> channels;
channels.push_back(dst);
channels.push_back(dst2);
channels.push_back(dst3);
```

```
        merge(channels, color1);
        imshow("Imagen Segmentada", color1);

        // Dibujar el histograma
        int hist_w = 512; int hist_h = 400;
        int bin_w = cvRound((double)hist_w / histSize)
            ;

        Mat histImage(hist_h, hist_w, CV_8UC3, Scalar
            (0, 0, 0));

        /// Normalizar el resultado de [ 0, histImage.
            rows ]
        normalize(hist, hist, 0, histImage.rows,
            NORM_MINMAX, -1, Mat());

        // Dibujar el unico canal
        for (int i = 1; i < histSize; i++)
        {
                line(histImage, Point(bin_w*(i - 1),
                    hist_h - cvRound(hist.at<float>(i -
                     1))),
                        Point(bin_w*(i), hist_h -
                            cvRound(hist.at<float>(i)))
                            ,
                        Scalar(255, 255, 255), 1, 8,
                            0);
        }

        // Mostrar Histograma
        namedWindow("Histograma", CV_WINDOW_AUTOSIZE);
        imshow("Histograma", histImage);

        waitKey(0);
        return 0;
}
```

Se utilizo la imagen de la figura 4 para la realización de las pruebas.

Figura 4: Imagen utilizada para las pruebas de segmentación por histograma.

En la figura 5 se muestra el histograma obtenido de la imagen de prueba.

Figura 5: Histograma correspondiente a la imagen de prueba.

Tambien se obtienen las salidas para cada pico, las cuales se muestran en la figura 6.

(a) Mascara para la sección 1

(b) Mascara para la sección 2

(c) Mascara para la sección 3

Figura 6: Salidas correspondientes a las secciones de acuerdo a la segmentación de pico-valle en relación al valor máximo de cada pico.

Finalmente obtenemos la salida segmentada de nuestra imagen en RGB, la cual se muestra en la imagen 7.

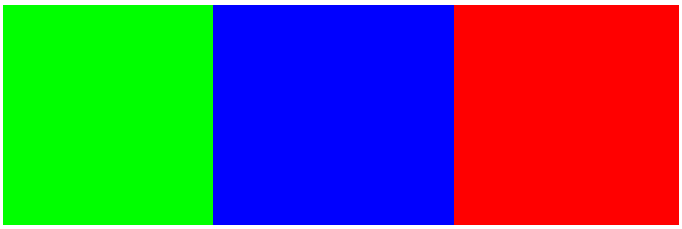

Figura 7: Imagen segmentada por histograma.

3. Bibliografía

1 Michael J. Dana H., “Indexing via color histogram”, IEEE Third International Conference on Computer Visión, 1990. Proceedings.

1 Castleman, Kenneth R. 1979. Digital Image Processing. Prentice-Hall.

2 Phillips, Dwayne. August 1991. “Image Processing, Part 4: Histograms and Histogram Equalization,”The C Users Journal.

3 J. Sporring, et.al. (eds), Gaussian Scale-Space Theory, Kluwer Academic Publishers, Netherlands, 1997.

4 Opencv, Histogram Backprojection, $http://tinyurl.com/zeoyb6p$