Segmentación de color en imágenes digitales usando Visual C#.Net

Bayardo Campuzano Nieto
Ingeniería de Sistemas
UPS-QUITO

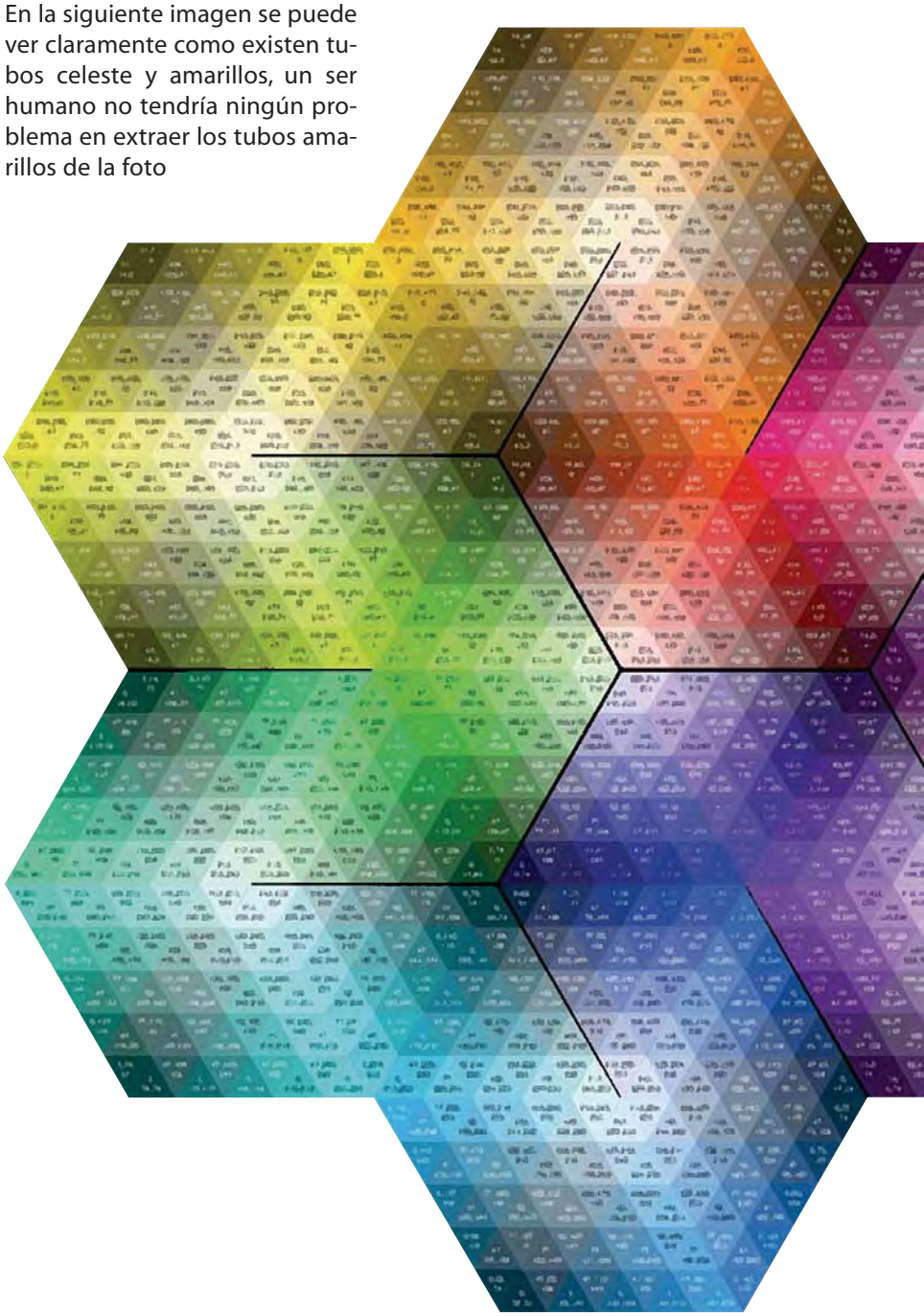
Resumen

En este trabajo se presenta la definición de los histogramas de color de una imagen y se usa dicha definición para el segmentado de objetos cuyo color resalte en dicha imagen además se implementa estos conceptos en Visual C#.Net.

I. Antecedentes

En robótica es costumbre sacar a un determinado objeto de su entorno usando para ello la técnica de umbralizado de imágenes grises, pero esta técnica si bien es rápida no me permite extraer objetos cuyo color resalta del entorno, y que una ves convertida en imagen gris se perdía dicho en el entorno, en la siguiente imagen se puede observar esto.

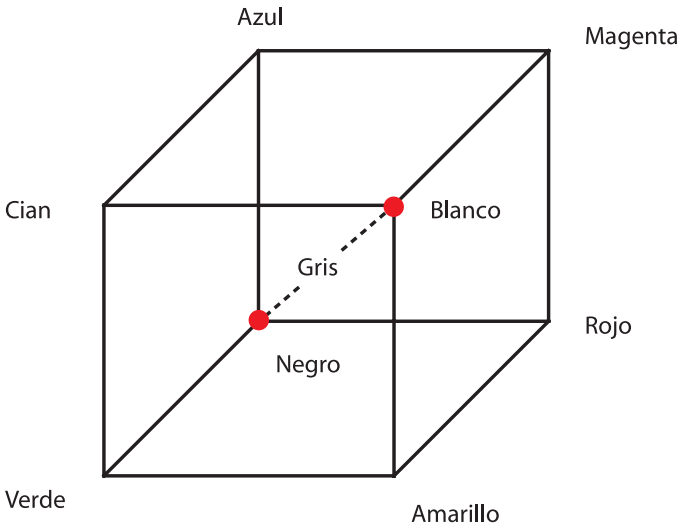
En la siguiente imagen se puede ver claramente como existen tubos celeste y amarillos, un ser humano no tendría ningún problema en extraer los tubos amarillos de la foto



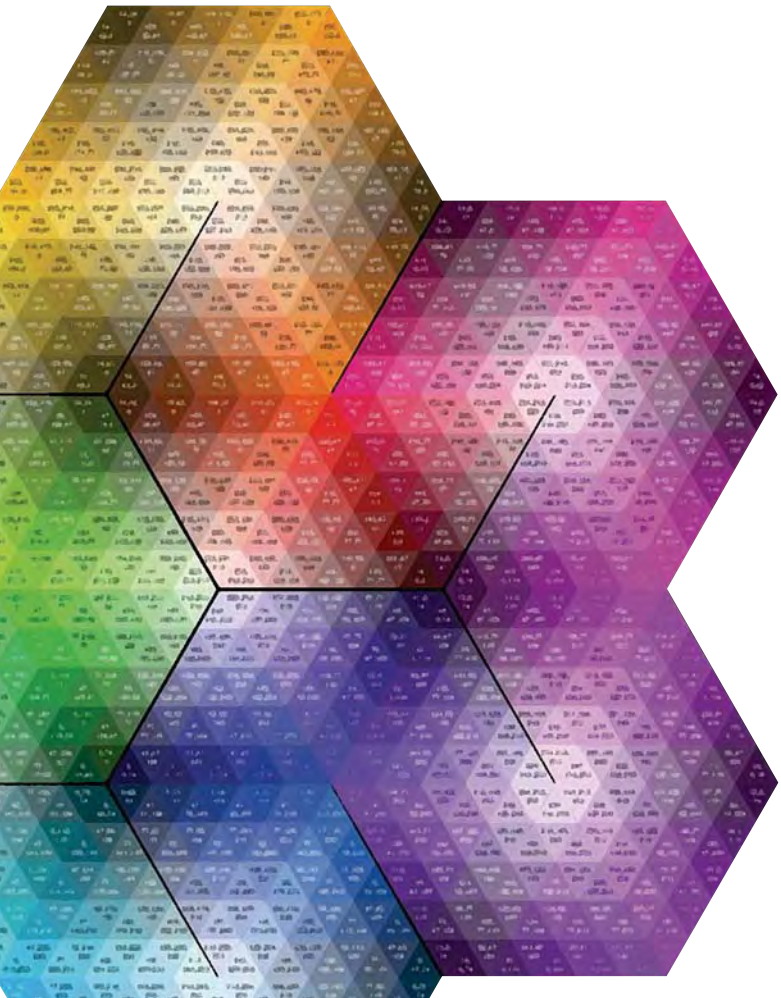
Pero si convertimos esta en imagen gris tendríamos lo siguiente:

En esta imagen todos los juegos infantiles poseen el mismo tono por lo que nos resulta imposible diferenciar los unos de los otros, esto se observa claramente si observamos el histograma de intensidades grises (que es una función del numero de píxeles que poseen una determinada intensidad) de la imagen anterior:

En donde no se ven los picos que nos permitan separar un objeto del otro y además de su fondo por lo que esta imagen seria de difícil segmentado de los objetos en ella contenidos.



Una imagen digital de color para el computador es una función vectorial bidimensional cuyo dominio es la posición espacial y el rango esta en el espacio de color que para el caso posee tres componentes (La componente Roja, Verde y Azul)



I. Imágenes digitales de color

Una imagen digital de color para el computador es una función vectorial bidimensional cuyo dominio es la posición espacial y el rango esta en el espacio de color que para el caso posee tres componentes (La componente Roja, Verde y Azul)

$C(R,V,A) = f(x,y)$

Por tanto es almacenada como una matriz NxM de elementos. El color es también almacenada en el sistema RGVA por cuatro números enteros dando un registro de 24 bits.

Color = (8 Bits de Transparencia,8 Bits de Rojo,8 Bits de Verde,8 Bits de Azul)

Donde la transparencia describe el grado de color de fondo que pasa a través del color que se coloca en el píxel x,y.

Esta puede ser representada gráficamente por un cubo cuyo lado tiene 256 niveles la recta que une el valor máximo con el mínimo se encontraran los grises.

III. Histograma de color

El Histograma de una imágenes es una función discreta que representa el numero píxeles de la imagen en función de su nivel de color RGV, es decir que, si en una determinada imagen tiene 150 píxeles con una determinada cantidad r de Rojo, v de Verde y a de Azul entonces el valor del histograma de la imagen para este valor Histo[r,v,a] será igual a 150.

El siguiente código nos permite el calculo del histograma de una imagen de color (en este caso la Imaln.jpg)

- se usa la clase Drawing para edición de Bitmap.
  using System.Drawing;

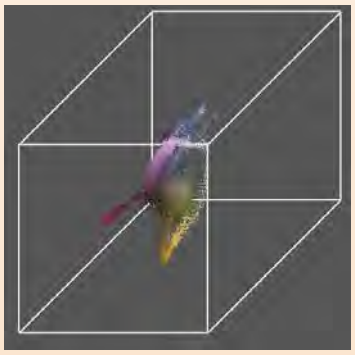
.....

- Aquí va el código que el Visual C# introduce para el manejo de formularios

.....

```
        Bitmap Ima = new Bitmap(@"c:\ImaIn.jpg");
int Alto = Ima.Height;
                int Ancho = Ima.Width;
                int x;    int y;
                Color CAux;
                MaxHisColor=0;
                for ( x = 0 ; x < Alto ; x++ )
                        for ( y = 0 ; y < Ancho ; y++ )
                        {       CAux = Ima.GetPixel(x,y);
                                HisColor[CAux.R,CAux.G,CAux.B]++;
if (HisColor[CAux.R,CAux.G,CAux.B]>MaxHisColor)
                        {
                        MaxHisColor=HisColor[CAux.R,CAux.G,CAux.B];
                        }
                }
```

Para poder visualizar esta función tenemos algunos problemas pues se trata de una función de dimensión 4 (tres dimensiones del dominio mas la dimensión del rango) no es posible mostrar las cantidades de píxeles en función del color por lo que hemos decidido utilizar la transparencia como cuarta dimensión, con ello las intensidades de mayor frecuencia se muestran con el color respectivo y conforme tenemos intensidades menos probables se harán mas transparentes, de esta forma no marcamos los valores que no poseen ningún píxel. El resultado para la imagen en estudio se muestra en la siguiente grafica.



IV. Segmentación de color

En el histograma de color de la imagen en estudio se puede apreciar como se diferencian los intensidades de los diferentes colores, con ello es posible trabajando con estos valores el segmentar los tubos celestes, o si lo queremos los tubos amarillos del fondo de la imagen.

Para ello solo haremos pasar los píxeles cuyas componentes de color [x,y,z] estén en el interior de un cierto volumen de segmentación de forma de un paralelepípedo de color dado por las siguientes condiciones:

$$Ro < x < Rf$$
$$Vo < y < Vf$$
$$Ao < z < Af$$

El siguiente código nos permite realizar dicha segmentación:

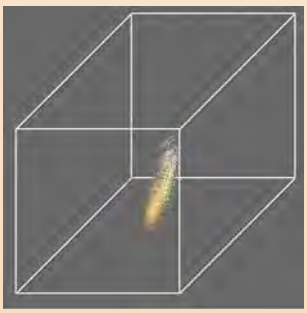
```
int x; int y;
Color colorIn;
        int UoR = 200;
        int UoV = 155;
        int UoA = 0;
        int UfR = 255;
        int UfV = 255;
        int UfA = 200;
        int Ancho = ImaIn.Height;
        int Alto = ImaIn.Width;
        ImaOut = new Bitmap(Alto,Ancho);
        for(x=0;x<Alto;x++)
                for(y=0;y<Ancho;y++)
                {
                        colorIn = ImaIn.GetPixel(x,y);
                        //      Aquí se realiza la segmentación
```

```
                    if ((UoR<colorIn.R)&&(colorIn.R<UfR))
                    if ((UoV<colorIn.G)&&(colorIn.G<UfV))
                    if ((UoA<colorIn.B)&&(colorIn.B<UfA))
                    {
                            ImaOut.SetPixel(x,y,colorIn);
                    }
            }

ImaOut.Save(@"c:\ImaOut.jpg",System.Drawing.Imaging.ImageFormat.Jpeg);
```

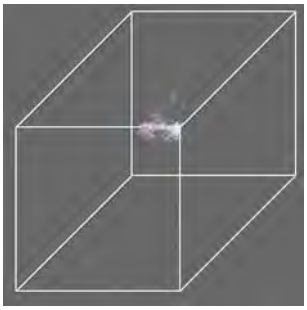
Para esta imagen hemos buscado que el valor de color que pase posea componentes cuyos valores de Rojo este entre 200 y 255, de Verde este entre 155 y 255, y finalmente el valor de la componente Azul estará entre 0 y 200. El resultado de este proceso de segmentación se lo guarda en el archivo ImaOut.jpg el cual los mostramos a continuación..

Y su respectivo Histograma de color será:



Como podrán observar hemos segmentado no solo los tubos amarillos sino todo objeto con tonos amarillos, para poder segmentar los tubos celestes los valores de las componentes de color estarán entre Rojo $0 < x < 256$, verde $0 < y < 256$ y Azul $200 < z < 256$, con el siguiente resultado:

Siendo su histograma:



Como podemos observar se nos han pasado algunos amarillos, y como en los tubos celestes poseen mucho brillo blanco no nos es posible eliminarlo usando un volumen de segmentación de forma de paralelepípedo.

U. Conclusiones

- Hemos definido y obtenido para algunos ejemplos el histograma de color de una imagen.
- Se ha podido segmentar objetos cuyo color resalte usando para ello volúmenes de segmentación de forma de paralelepípedos de color con resultados bastante aceptables.
- Para casos especiales deberá usarse volúmenes de formas especificas para cada caso para mejorar la segmentación.

UI. Bibliografía

- K. S. Fu, R. C. Gonzalez y C.S.G. Lee, *Robotica: Control detección, visión e inteligencia,* ED McGraw-Hill, 1990.
- D. Marvall Gómez-Allende, *Reconocimiento de Formas y Visión Computacional*.
- M: Faúnde Zanuy, *Tratamiento Digital de Voz e Imagen y aplicaciones a la Multimedia,* Ed Alfaomega, Madrid 2001.
- G. Pajares y J. M. De la Cruz, *Visión Por Computadora,* Ed Alfaomega, Madrid, 2002.
- A. de la Escalera, *Visión Por Computadora,* Ed Prentice-Hall, Madrid, 2001.
- *Revista Electrónica de Visión por Computadora,* REVC**,** http://revc.uab.es
- VIDERE, *A Journal of Computer Vision Research*, http://mitpress.mit.edu/e-journals/Videre/