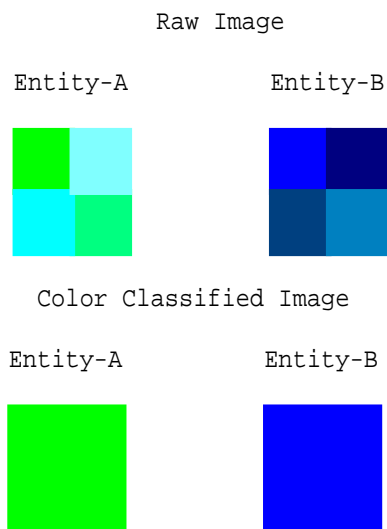


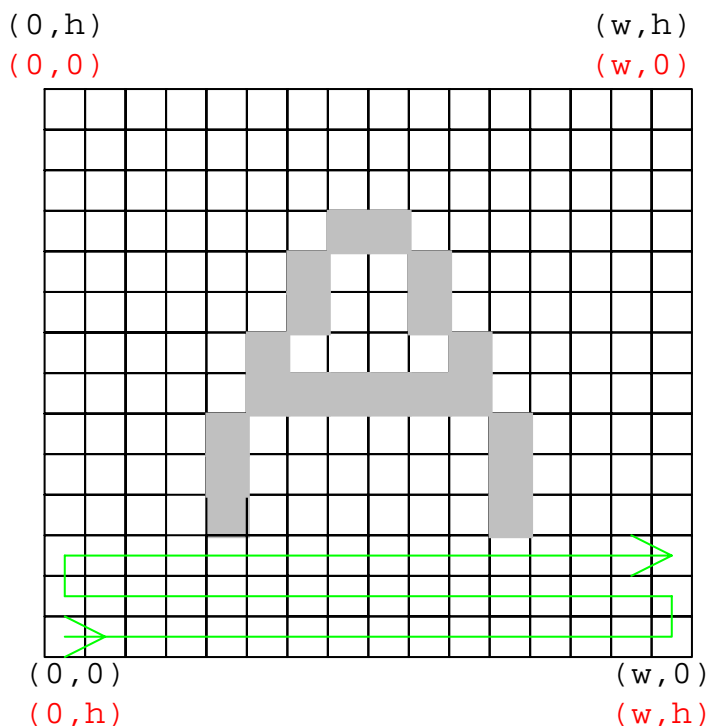
Color Classification – GetColor()



Shown here is an image of a green and blue square grabbed by a camera. Entity-A is supposed to be 'pure' green and Entity-B 'pure' blue. Each square represents a pixel. The pixels of Entity-A fall within a certain range(slice/segment) of H,S,I(color) values and those of Entity-B fall within a different range.

The objective of Color Classification is to convert all pixels within specified color ranges to a 'pure' colors, by **color slicing/segmentation** and store them, as a color code, in a 2D array of size equal to the size(height x width) of the Raw image divided by the Scale Factor.

The Raw image is read into an array and scanned one pixel at a time from the bottom left hand corner(0,0) towards the right and upwards to the top right hand corner. Pixels at the edge of the image are given a color code of zero, for Entity Classification to work correctly. Pixels within the 'Low(background)/Hi(foreground) Intensity' ranges are given color codes defined for these ranges. The color codes selected for these intensities should be different from those selected for segments, If not, they will be converted to Entities by Entity Classification, they may also be given a code of 0(black). Finally Pixels with colors within a color segment are given color codes defined for the segment.

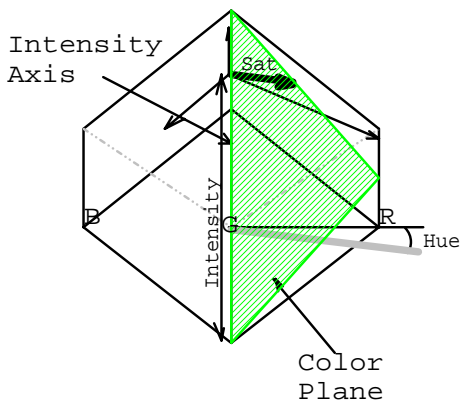


- Scan direction
- (X,X) GetColor() Co-ordinates(cartesian)
- (X,X) MS-Paint Co-ordinates(Spatial)

COLOUR

A complex color is defined by the values of its R,G,B simple color constituents. These constituents are vectors in the color cube, ranging in value from 0 to 255. Hold the color cube with the Intensity axis (connecting the two extreme vertices) perpendicular to the floor and the plane formed by the R,G,B vertices parallel to the floor, with the B and R vertices facing you.

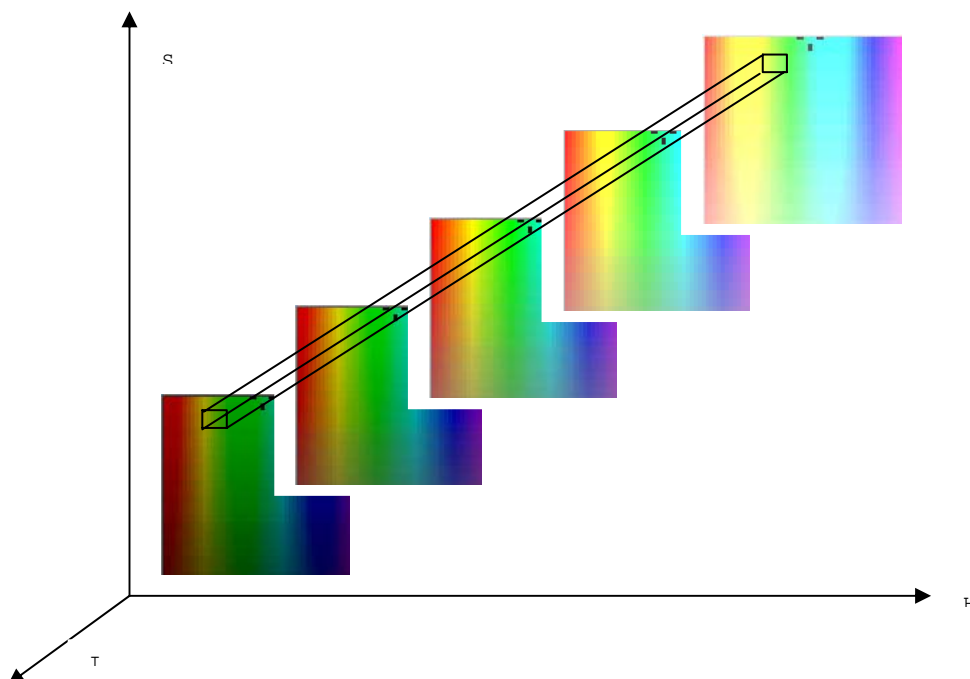
The R,G and B vectors for a given complex color, start at a point on the intensity axis (length ranging from 0 to 255) known as the **Intensity(I)** (brightness of a pixel's colour. Analogous to the 'Brightness' of a TV picture). The resultant of the R,G,B vectors is known as **Saturation(S)** (white content or 'purity', analogous to the 'Contrast' of a TV picture) and the angle it makes w/ the 'R' vertex, the **Hue(H)** (Color). The **Color Plane** is formed by extending the intensity axis outwards to the edge of the cube, at the Hue angle. A color can be specified by its R,G,B or H,S,I constituents and one can be converted to the other.



COLOR SLICING/SEGMENTATION

For each complex color eg 'pure' green of Entity-A, upper and lower threshold values are set for Hue(H), Relative Saturation(S) and Intensity(I). The R, G & B values of every pixel in the image are then converted to **H, S** and **I** values and compared with the threshold values for each complex color. Pixels within the upper and lower thresholds are given the R, G and B values of the pure color.

Pictorial representation of Color Slicing for Entity 'A'



Inputs - Setting

Pass the address of a structure variable of type FLTR, with the following members.

- eclr[2] - Color code for low[0]/hi[1] intensity.
- eint[2] - Limit for low[0]/hi[1] intensity.
- ncc - Number of pure colors in filtered o/p, number of color segments, number of color codes defined for Entities(color codes are 1 to ncc).
- fnam - Pointer to a file name string to save Object details to the disk
- fprn - Save(1)/No Save(0) filtered o/p to the disk
- rfile - Pointer to a file name string of raw image to be filtered.
- scale - Scale factor to divide size of o/p image wrt i/p image(≥ 1)
- H[i*] - 'Hue' of Color Segment 'j', at boundary 'm' at index 'i'
- S[i*] - 'Saturation' of Color Segment 'j', at boundary 'm' at index 'i'
- I[i*] - 'Intensity' of Color Segment 'j' at boundary 'm' at index 'i'
- clsc[j*]- Array of 'COLOR' structures with r,g,b values of color segment 'j'.
Required only when 'fprn'=1

* Given:-

Color Segment 'j' (0 to ncc-1) and segment boundary 'm' (Lower(0)/Upper(1))

$i = (m * ncc) + j$.

Inputs - bmphdr() Outputs

Pass the address of a structure variable of type IMGF, with the following members.

- hdr - A structure variable of type BMPH. Members of 'hdr' are header fields of the raw (unscaled) image file(24 bit, uncompressed, windows bitmap).
- ndum - Number of dummy bytes per pixel row of raw image.
- ht - Depth in pixels of scaled & filtered o/p image= $\text{hdr.h}/\text{scale}$.
- wd - Width in pixels of scaled & filtered o/p image= $\text{hdr.w}/\text{scale}$.
- clrf[] - Empty 'int' array. Number of elements= $(\text{hdr.h} * \text{hdr.w})/\text{scale}^2$.
- data[] - Empty array of 'COLOR' structures into which the raw image is read. Number of elements= $\text{hdr.h} * \text{hdr.w}$

Return Values

The following are returned in the input structure variable of type IMGF

- clrf[j*] - Color Coded and scaled filter o/p
- data[i*]- Array of 'COLOR' structures with r,g and b values of an image. The image is :-
If fprn=0 - Raw unscaled image to be filtered.
If fprn=1 - Scaled filtered image.

* Given that the raw image:-

Column scan count(x) increments from 0 to $\text{hdr.w}-1$ and

Row scan count(y) increments from 0 to $\text{hdr.h}-1$. Then

$i = (x * \text{hdr.h}) + y$

$j = (x * \text{hdr.h})/\text{scale}^2 + y/\text{scale}$