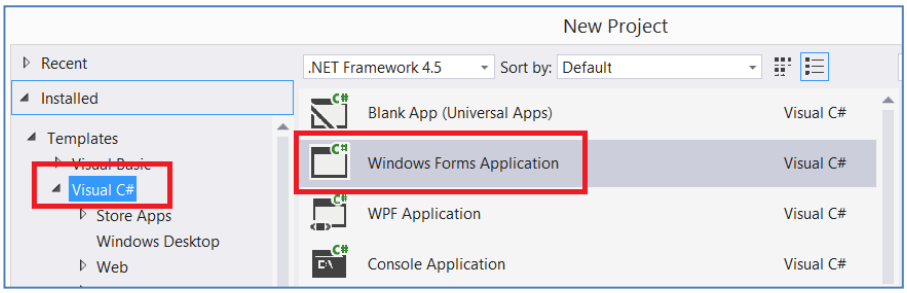
GÖRÜNTÜ İŞLEME - (2.Hafta)

C# PROGRAMLAMA İLE GÖRÜNTÜ İŞLEME

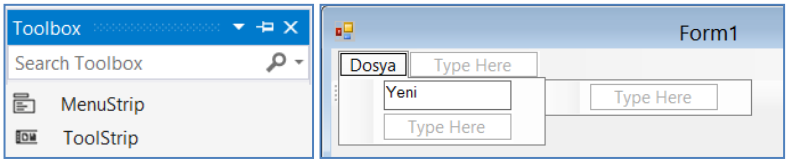
Görüntü işleme kapsamında geliştirilecek algoritmalar C# diliyle yazılacaktır. Bu amaçla bilgisayarımızda Visual Studio programının kurulu olması gerekmektedir. Öncelikle programın altyapının oluşturulup, dersler ilerledikçe içerisine kodlar eklenecektir.

Program Altyapısının Oluşturulması

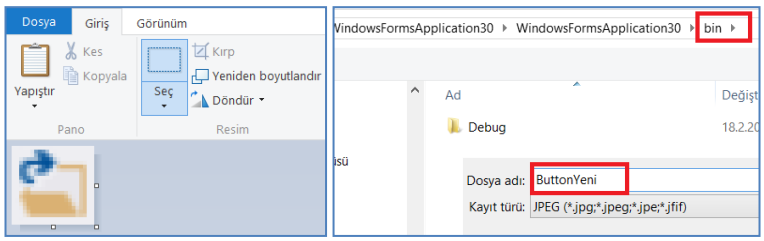
Geliştirilecek program Windows Form Application uygulaması olacaktır.



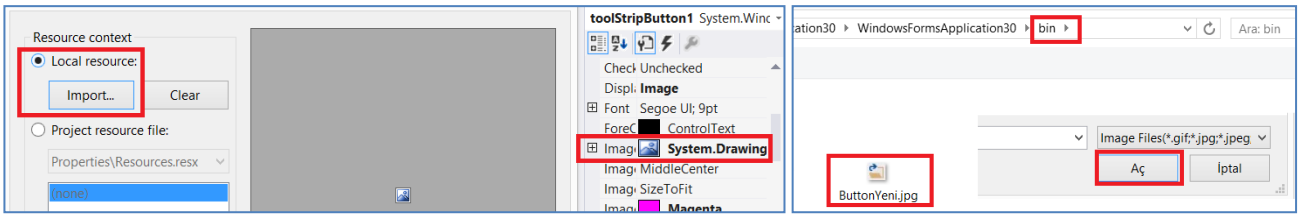
Program içerisinde Menü ve Araç çubuğu kullanılacaktır. Bu amaçla MenuStrip ve ToolStrip nesneleri kullanılacaktır.



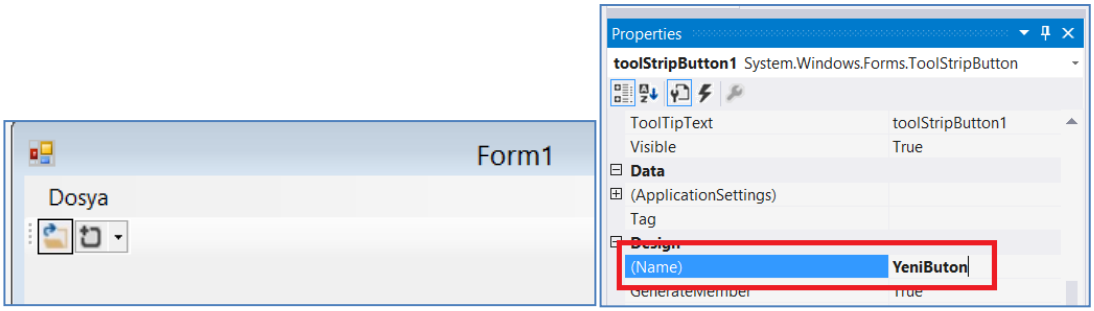
ToolStrip (araç çubuğu) üzerindeki butonların resimlerini atarken, Paintte oluşturulan örnek resimler projenin içerisindeki Bin klasörünün altına kopyalayalım.



Daha sonra ToolStrip üzerindeki butonu seçip Image özelliğinden butonu yükleyelim.



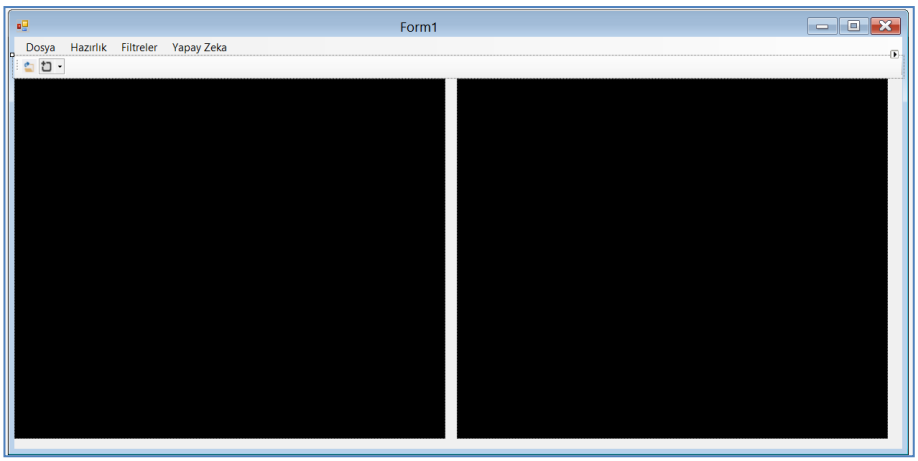
Butonun resmi yüklendikten sonra Properties penceresinden başka gerekli ayarlamalar yapılabilir. Örneğin butonun adı yapacağı işle ilgili olarak değiştirilebilir.



Butona tıkladığımızda ona ait olan fonksiyon kodları açılacaktır.

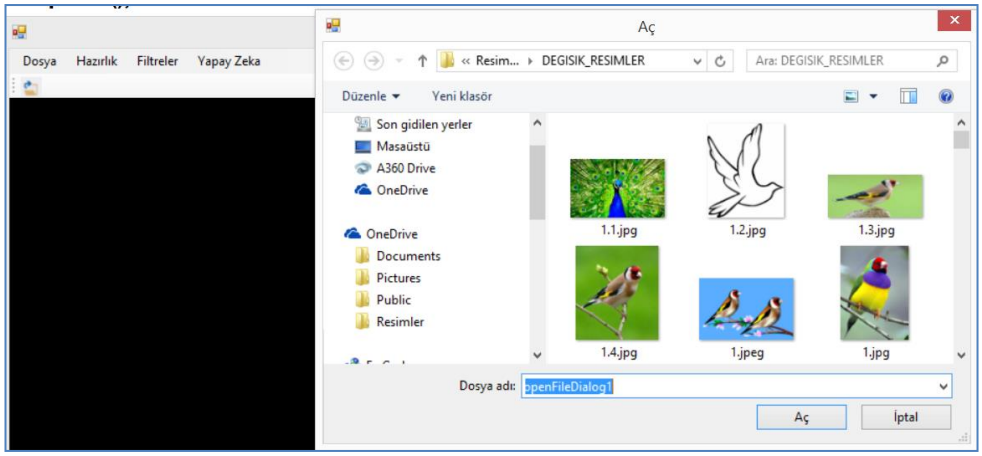
```
public partial class Form1 : Form
{
    1 reference
    public Form1()
    {
        InitializeComponent();
    }

    1 reference
    private void YeniButon_Click(object sender, EventArgs e)
    {

    }
}
}
```

Programımızın tasarımını üstte bir menu çubuğu, altında ise Araç çubuğu olacak şekilde tasarlayalım. Form üzerinde ise iki tane Picturebox olsun. Bunlardan birincisi orjinal resmi, diğeri ise dönüştürülmüş resmi göstersin. Program ilerledikçe ekleme ve çıkarmalar olacaktır. Bu altyapı üzerinde devam edelim.



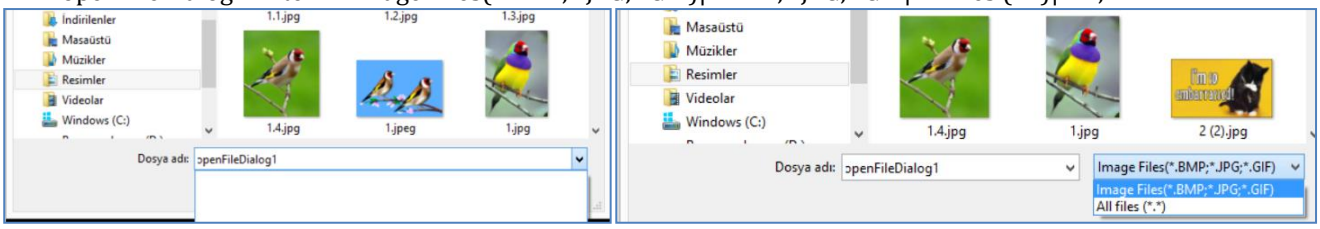
RESİM YÜKLEME

Bilgisayarımızdaki herhangi bir resmi PictureBox1 yüklemek için aşağıdaki kodları kullanabiliriz. Bunu fonksiyon şeklinde yazarsak hem menüden hem de araç çubuğunda aynı kodlar kullanılarak resim yüklenebilir. Burada geliştirilen programda iki tane picturebox kullanılmaktadır. Birincisinde orjinal resim (giriş resmi), ikincisinde ise dönüştürülmüş resim (çıkış resmi) görüntülenecektir.



Dosya açarken uzantıların görüntülenmesi süzmek için uygun olacaktır. Bunun için aşağıdaki kodları kullanabiliriz.

```
openFileDialog1.DefaultExt = ".jpg";
openFileDialog1.Filter = "Image Files(*.BMP;*.JPG;*.GIF)|*.BMP;*.JPG;*.GIF|All files (*.*)|*.*";
```



Eğer resim yükleme çalışırken openFileDalog açıldıktan resim seçilmeyip "iptal" tuşuna basılırsa, resimyolu değişkeni "null" olacağı için program hata verir. Bunu engellemek için Try-Catch kodları içine yazılması gerekir.

```
private void DosyaAc_menu_Click(object sender, EventArgs e)
{
 DosyaAc();
}
private void DosyaAc_toolbar_Click(object sender, EventArgs e)
{
 DosyaAc();
}
 //DOSYA AÇ ----------------------------
 public void DosyaAc()
 {
      try
      {
            openFileDialog1.DefaultExt = ".jpg";
            openFileDialog1.Filter = "Image Files(*.BMP;*.JPG;*.GIF)|*.BMP;*.JPG;*.GIF|All files (*.*)|*.*";

            openFileDialog1.ShowDialog();

            String ResminYolu = openFileDialog1.FileName;
            pictureBox1.Image = Image.FromFile(ResminYolu);
      }
      catch { }
 }
```

Görüntülenen resim çeşitli boyutlarda olabilir. Bu nedenle resmin picturebox içerisinde düzgün görüntülenmesi için SizeMode seçeneğini aşağıdaki seçeneklerden uygun olanla yapmak gerekir. Burada;

a) Normal (resmin sığdığı kadar kısmı normal görüntülenir

b) StrechImage (resim çerçevenin içine tam sığacak şekilde daraltılır yada genişletilir,

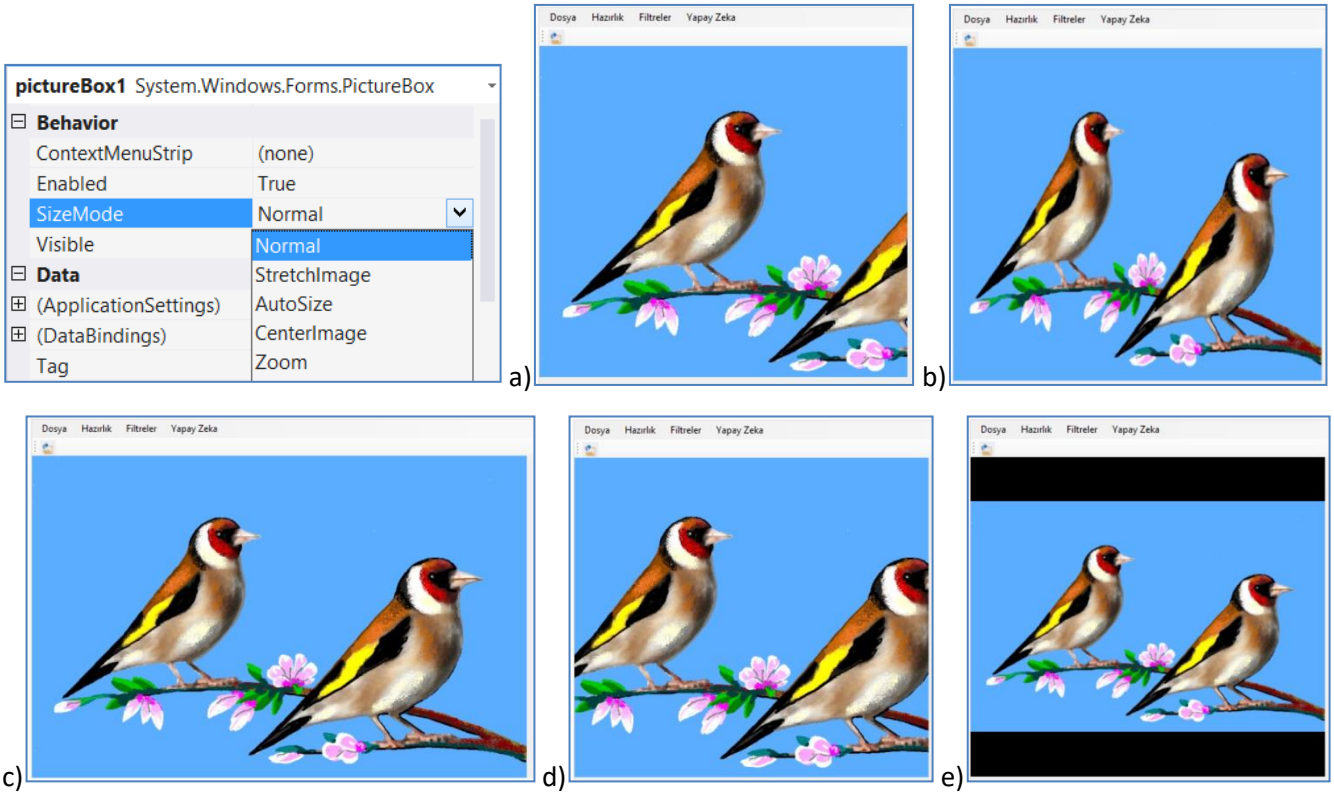
c) AutoSize (çerçeve resmin boyutlarını alır)

d) CenterImage (resmin orta bölgesi çerçeve içinde normal şekilde görüntülenir, sığmayan kısımlar dışarıda kalır,

e) Zoom (resmin orantısı bozulmadan tamamı çerçeve içinde görüntülenir).

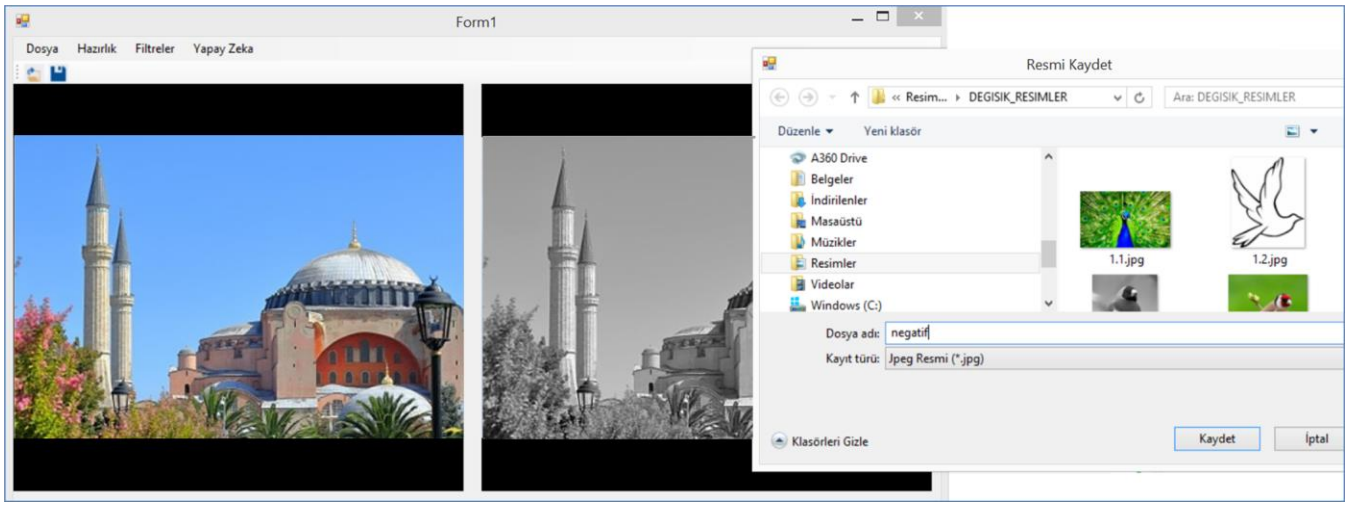
Not: Kodla size ayarlamak için

```
pictureBox1.SizeMode = PictureBoxSizeMode.Zoom;
```



RESMİ KAYDETME

Birinci picturebox da görüntülenen orjinal resim, belli filtrelerden geçirildikten sonra ikinci picturebox da görüntülenecektir. Görüntülenen bu ikinci resim gerektiğinde bilgisayara kaydedilebilmelidir. Bu amaçla aşağıdaki kodları kullanabiliriz. Burada üç tip resim kaydetme yapılmaktadır. Bunlar jpeg, bitmap ve gif resimleridir.

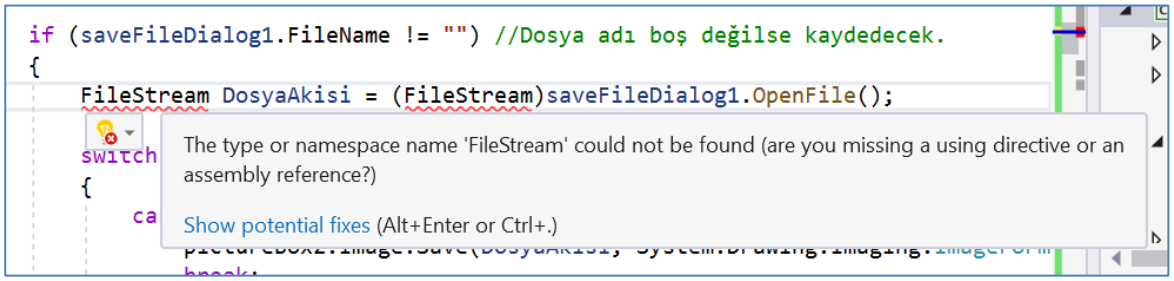


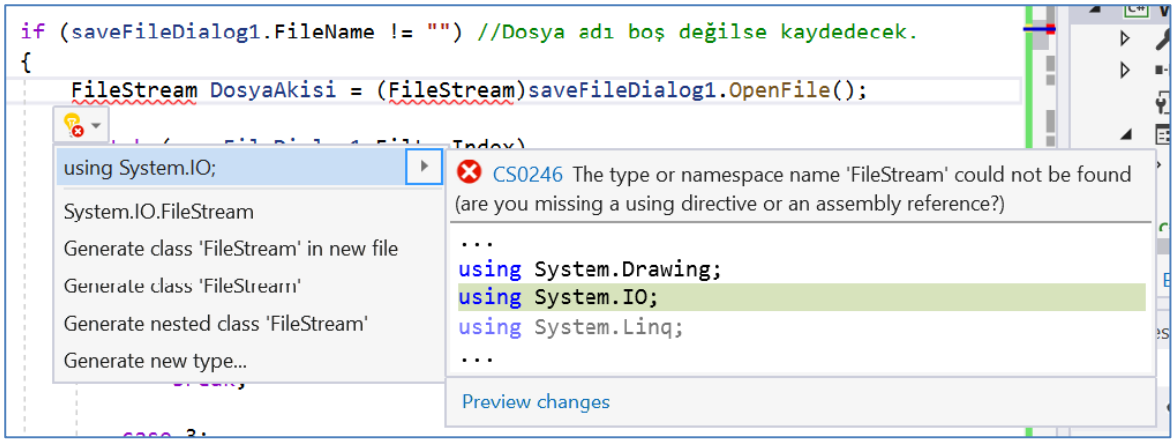

```
//RESMİ KAYDETME---------------------------------------
public void ResmiKaydet()
{
  SaveFileDialog saveFileDialog1 = new SaveFileDialog();
  saveFileDialog1.Filter = "Jpeg Resmi|*.jpg|Bitmap Resmi|*.bmp|Gif Resmi|*.gif";
```

```
        saveFileDialog1.Title = "Resmi Kaydet";
        saveFileDialog1.ShowDialog();

        if (saveFileDialog1.FileName != "") //Dosya adı boş değilse kaydedecek.
        {
            // FileStream nesnesi ile kayıtı gerçekleştirecek.
            FileStream DosyaAkisi = (FileStream)saveFileDialog1.OpenFile();

            switch (saveFileDialog1.FilterIndex)
            {
                case 1:
                    pictureBox2.Image.Save(DosyaAkisi, System.Drawing.Imaging.ImageFormat.Jpeg);
                    break;

                case 2:
                    pictureBox2.Image.Save(DosyaAkisi, System.Drawing.Imaging.ImageFormat.Bmp);
                    break;

                case 3:
                    pictureBox2.Image.Save(DosyaAkisi, System.Drawing.Imaging.ImageFormat.Gif);
                    break;
            }

            DosyaAkisi.Close();
        }
    }
```

Not:

2019 Versiyonunda bir komutun kütüphanesini (reference) eklemek için aşağıdaki gibi mouse ile üzerine gelince potansiyel seçenekleri gösterecektir. Bunun için “Show potential fixes” tıklayınız. Daha sonra çıkan listede using.System..IO; kütüphanesini seçin. Kendisi otomatik olarak programın en üstüne ekleyecektir.





RESMİ PİKSEL OLARAK AKTARMA

```
private void btnResmiPikselOlarak_Click(object sender, EventArgs e)
{
    Color OkunanRenk, DonusenRenk;
    int R = 0, G = 0, B = 0;

    Bitmap GirisResmi, CikisResmi;
    GirisResmi = new Bitmap(pictureBox1.Image);


    int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı. İçerisine görüntü
yüklendi.
    int ResimYuksekligi = GirisResmi.Height;
    CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini oluşturuyor.
Boyutları giriş resmi ile aynı olur. Tanımlaması globalde yapıldı.

    for (int x = 0; x < ResimGenisligi; x++)
    {
        for (int y = 0; y < ResimYuksekligi; y++)
        {
            OkunanRenk = GirisResmi.GetPixel(x, y);

            R = OkunanRenk.R;
            G = OkunanRenk.G;
            B = OkunanRenk.B;

            DonusenRenk = Color.FromArgb(R, G, B);
            CikisResmi.SetPixel(x, y, DonusenRenk);
        }
    }

    pictureBox2.Image = CikisResmi;
}
```

RESMİN NEGATİFİNİ ALMA (Negative)

Bu işlem basit olarak şu şekilde yapılır. *Her piksel renk değeri bir döngü içinde 255 sayısından çıkarıldığında geriye kalan değer, negatif rengi verecektir*. Renkli resimlerde ise 3 renk de aynı işleme tabi tutulur.

Örneğin; siyah renkli bir pikselin değeri 0 dır. Bu pikselin değeri 255-0=255 olarak ayarlanırsa bu renk de beyaz olacaktır. Yada beyaz yakın bir değer olan 212 değerini dönüştürürsek, 255-212=43 olacaktır. Bu da siyaha yakın bir renktir. Bunu matematiksel olarak ifade edersek, g(x;y) giriş resminin Kırmızı piksel değeri, f[x;y] çıkış resminin kırmızı piksel değeri olacaktır.

$$f(x; y) = 255 - g(x; y)$$

Programlama

```
    //NEGATİFİNİ ALMA ----------------------------------
private void btnNegatif_Click(object sender, EventArgs e)
{
    Color OkunanRenk, DonusenRenk;
    int R = 0, G = 0, B = 0;

    Bitmap GirisResmi, CikisResmi;
    GirisResmi = new Bitmap(pictureBox1.Image);


    int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı. İçerisine görüntü
yüklendi.
    int ResimYuksekligi = GirisResmi.Height;
    CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini oluşturuyor.
Boyutları giriş resmi ile aynı olur. Tanımlaması globalde yapıldı.

    int i = 0, j = 0; //Çıkış resminin x ve y si olacak.
    for (int x = 0; x < ResimGenisligi; x++)
    {
         for (int y = 0; y < ResimYuksekligi; y++)
        {
            OkunanRenk = GirisResmi.GetPixel(x, y);

            R = 255 - OkunanRenk.R;
            G = 255 - OkunanRenk.G;
            B = 255 - OkunanRenk.B;

            DonusenRenk = Color.FromArgb(R, G, B);
            CikisResmi.SetPixel(x, y, DonusenRenk);
        }
    }

    pictureBox2.Image = CikisResmi;

}
```

GRİ TON RESİM (Siyah-Beyaz) (Grayscale)

Griton resimler renkler olmadan sadece ışığın yoğunluğunu gösteren resimlerdir. Eski teknoloji siyah beyaz resimlere benzer.

Bu resimler bilgisayarda 8 bit formatında saklanır. Bunun anlamı her piksel 8 bit ikili kod ile saklanır. Yani 2^8=256 olarak gösterilirse 0-255 arasında değer alır. 0 siyah rengi gösterirken, 255 beyaz rengi gösterecektir.

Bir griton resim 800x600 piksel boyutlarında olursa içerisinde 480.000 piksel olacaktır. Her piksel 8 bit=1 byte hafızada yer alacağına göre resmin tamamı 480 kByte=0,48 MByte yer kaplayacaktır. Aynı resim renkli olsayda,

her renk (RGB-RedGreenBlue) benzer şekilde 256 ton renk alarak $2^8*3=2^{24}$ yani 24 bit yer kaplar. Bu durumda aynı resim bu sefer 3 katı yer kaplar.

Griton resimleri, kenar tespiti yada eşik uygulamaları gibir farklı amaçlar için ileriki uygulamlarımızda kullanacağız.

Renkli resmin Gri-Ton a dönüştürülmesi

Renkli sayısal bir görüntüyü gri-ton bir görüntüye dönüştürme işlemi aslında RGB renk modelinde belirtilen her bir renk bandına karşı düşen gri-ton görüntülerin ölçeklendirilmesinden başka bir şey değildir. Normalde üç rengin değerini toplayıp üçe bölerek gri tonu elde edebiliriz. Fakat bu gözümüzün farklı renkleri farklı algılama hassasiyetini tam yansıtmaz. Bu nedenle aşağıdaki formüllerde verildiği gibi ölçekleme yapılmalıdır. İlk verilen formül en gerçekçi olanıdır. Dördüncü formülü ise basit olarak kullanabileceğimiz akılda kalıcı bir olabilir.

$$GriDegeri = (R + G + B)/3$$

$$GriDegeri = 0.299 \times R + 0.587 \times G + 0.114 \times B$$

$$GriDegeri = 0.21 \times R + 0.71 \times G + 0.071 \times B$$

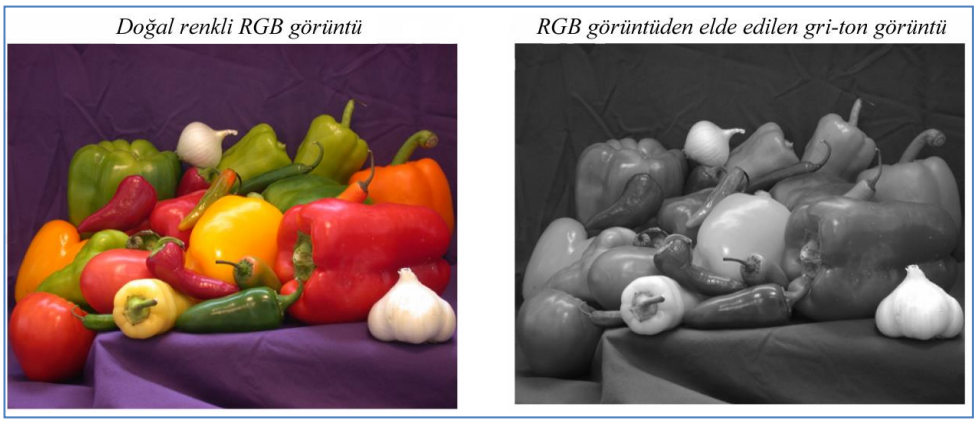
$$GriDeğer=0.3 \times R + 0.6 \times G + 0.1 \times B$$

Gri renk değeri bulunduktan sonra, R,G,B renk değerlerinin hepsi aynı renk koduna sahip olmalıdır. Yani;

$$R=GriDegeri,\ G=GriDegeri,\ B=GriDegeri$$

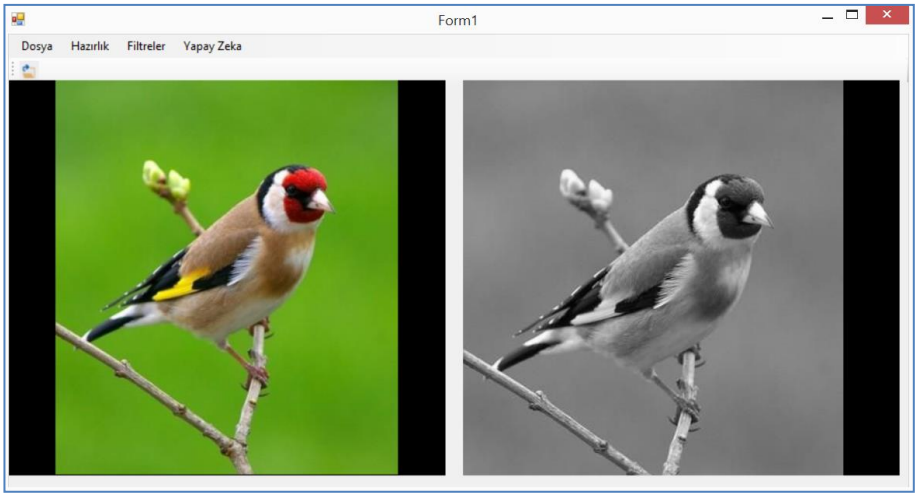
Tabii bu şekliyle resim yine 24 bit olarak saklanmış olur (hafızada renkli resim gibi yer tutar). Bunu 8 bit olarak saklamak gerekir.

Araştırma: *24 bit resim 8 bit olarak nasıl saklanır. Yani renkler üç tane formatta değilde tek renk formatında nasıl saklanır. Konuyu araştırınız. Her iki formatla saklanan gri-ton resmin hafızda kapladığı alanın farklı olduğunu gösteriniz.*



Şekil. Doğal renkli görüntüden Gri ton görüntünün elde edilmesi.

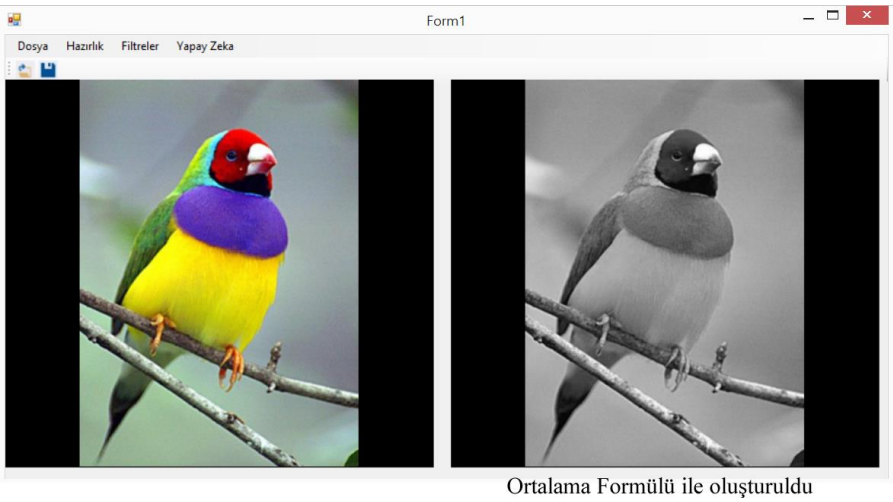
Programlama




```
//GRİ RESİM  ***********************************
private void btnGriResim_Click(object sender, EventArgs e)
{
    Color OkunanRenk, DonusenRenk;


    Bitmap GirisResmi = new Bitmap(pictureBox1.Image);

    int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı. Fonksiyonla gelmedi.
    int ResimYuksekligi = GirisResmi.Height;

    Bitmap CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini
oluşturuyor. Boyutları giriş resmi ile aynı olur.

    int GriDeger = 0;
    for (int x = 0; x < ResimGenisligi; x++)
    {
        for (int y = 0; y < ResimYuksekligi; y++)
        {
            OkunanRenk = GirisResmi.GetPixel(x, y);

            double R = OkunanRenk.R ;
            double G = OkunanRenk.G ;
            double B = OkunanRenk.B ;

            //GriDeger = Convert.ToInt16((R + G + B) / 3);

            GriDeger = Convert.ToInt16(R * 0.3 + G*0.6 + B*0.1);



            DonusenRenk = Color.FromArgb(GriDeger, GriDeger, GriDeger);
            CikisResmi.SetPixel(x, y, DonusenRenk);
        }
    }

    pictureBox2.Image = CikisResmi;
}
```

Gri-ton formülü olarak 3 kanal (R,G,B) renk ortalamasını alarak da gri-ton a dönüştürebiliriz. Ama bu formül daha az gerçekçi bir sonuç verecektir. Bunun için yukarıdaki programda ilgili satırı aşağıdaki kodlarla değiştirmeliyiz.

```
int GriDegeri = (int)(OkunanRenk.R + OkunanRenk.G + OkunanRenk.B)/3; //Ortalama Gri-ton formülü
```



Ortalama Formülü ile oluşturuldu

Gri-ton ölçekleme formülü ile oluşturuldu

Not:

Resmin üzerindeki noktaların renk değerlerini öğrenmek için aşağıdaki kodları kullanabilirsiniz. Böylece hangi rengin dönüşümlerde nasıl değiştiğini daha iyi yorumlarsınız.

```
private void pictureBox1_MouseClick(object sender, MouseEventArgs e)
        {
            Color Renk = ((Bitmap)pictureBox1.Image).GetPixel(e.X, e.Y);
            txtRGB.Text = string.Format("R:{0} G:{1} B:{2}", Renk.R, Renk.G, Renk.B);
        }
```

Araştırma: Renkli resimleri, Gri resme dönüştürmeyi öğrendik. Peki Gri resmi, renkli resme nasıl dönüştürürüz. Üzerinde araştırma yapın. Aşağıdaki linkteki bu uygulamanın başarısını gözlemleyin. Daha iyi bir algoritmayı nasıl geliştirirsiniz fikir geliştirin. (https://demos.algorithmia.com/colorize-photos)

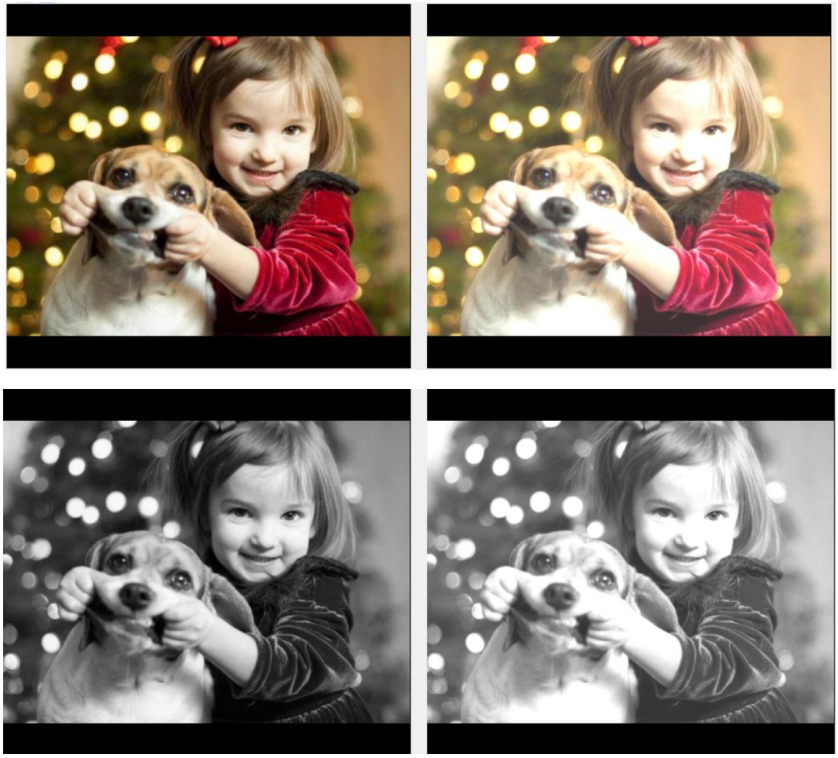
PARLAKLIK (BRIGHTNESS)

Bir resmin parlaklılığını artırma yada rengini açma, beyaz tona doğru ilerlemek için renk kanalları üzerine aynı büyüklükte eşdeğer bir sayı eklemekle gerçekleştirilir. Resmi koyulaştırmak için tam tersi sayı çıkarılır. Burada dikkat edilmesi gereken üzerine eklenen sayılarla sonuç 255 değerini geçmemelidir yada çıkarılan sayılar nedeniyle 0 altına düşmemelidir. Şu şekilde formülüze edebiliriz.

$$f(x,y)= g(x,y)(R\mp b, G\mp b, B\mp b)$$

Program Kodları

Birinci resimler orjinal resim, ikinciler rengi 50 değerinde açılmış resimdir.



```
    //PARLAKLIĞINI DEĞİŞTİRME ---------------------------------------
    public Bitmap ResminParlakliginiDegistir()
    {
      int R = 0, G = 0, B = 0;
      Color OkunanRenk, DonusenRenk;
      Bitmap GirisResmi, CikisResmi;
      GirisResmi = new Bitmap(pictureBox1.Image);

      int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
      int ResimYuksekligi = GirisResmi.Height;
      CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini oluşturuyor. Boyutları giriş resmi
ile aynı olur.

      int i = 0, j = 0; //Çıkış resminin x ve y si olacak.
      for (int x = 0; x < ResimGenisligi; x++)
      {
        j = 0;
        for (int y = 0; y < ResimYuksekligi; y++)
        {
          OkunanRenk = GirisResmi.GetPixel(x, y);
        //Rengini 50 değeri ile açacak.
          R = OkunanRenk.R + 50;
          G = OkunanRenk.G + 50;
          B = OkunanRenk.B + 50;

          //Renkler 255 geçtiyse son sınır olan 255 alınacak.
          if (R > 255) R = 255;
          if (G > 255) G = 255;
          if (B > 255) B = 255;

          DonusenRenk = Color.FromArgb(R, G, B);
          CikisResmi.SetPixel(i, j, DonusenRenk);
          j++;
        }
        i++;
      }
      return CikisResmi;
```

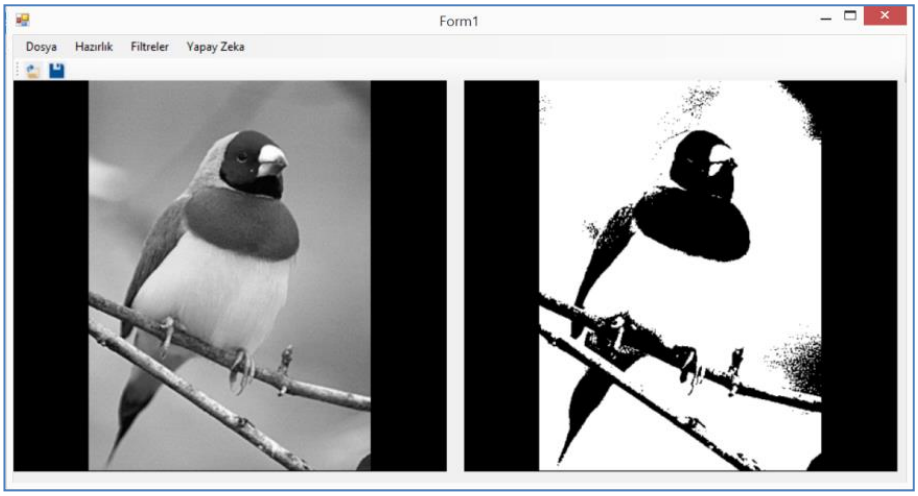
```
    }
```

EŞİKLEME (Thresholding)

Bu teknikte resim içerisindeki her pikselin renk değeri belli bir eşik değeri ile kıyaslanır. Eğer bu eşik değerinin altında ise bir renk, üzerinde ise başka bir renk atanır. Örneğin gri-ton bir resmi bu filtreden geçirirken, renk değeri 128 gibi bir eşik değerin altındaysa 0-siyah olarak, üzerinde ise 255-beyaz olarak atanabilir. Matematiksel olarak aşağıdaki gibi gösterilebilir.

$$f(x,y) = \begin{cases} 0 & eğer\ g(x,y) < T = 128 \\ 255 & diğer \end{cases}$$

Programlama



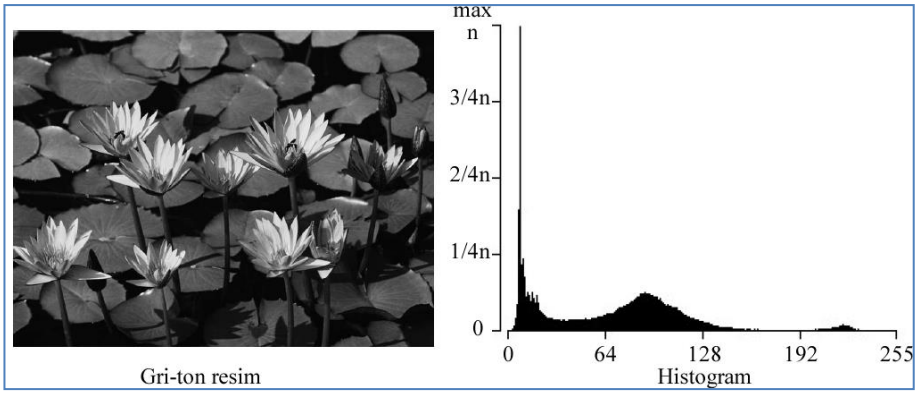
```
//EŞİKLEME ----------------------------------
public Bitmap ResmiEsiklemeYap()
{
int R = 0, G = 0, B = 0;
Color OkunanRenk, DonusenRenk;
Bitmap GirisResmi, CikisResmi;
GirisResmi = new Bitmap(pictureBox1.Image);

int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
int ResimYuksekligi = GirisResmi.Height;
CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini oluşturuyor. Boyutları
giriş resmi ile aynı olur.

int EsiklemeDegeri = Convert.ToInt32(textBox1.Text);

for (int x = 0; x < ResimGenisligi; x++)
{
    for (int y = 0; y < ResimYuksekligi; y++)
    {
        OkunanRenk = GirisResmi.GetPixel(x, y);
```

```
        if (OkunanRenk.R >= EsiklemeDegeri)
            R = 255;
        else
            R = 0;

        if (OkunanRenk.G >= EsiklemeDegeri)
            G = 255;
        else
            G = 0;

        if (OkunanRenk.B >= EsiklemeDegeri)
            B = 255;
        else
            B = 0;

        DonusenRenk = Color.FromArgb(R, G, B);
        CikisResmi.SetPixel(x, y, DonusenRenk);
    }
}

pictureBox2.Image = CikisResmi;
}
```

HİSTOGRAM

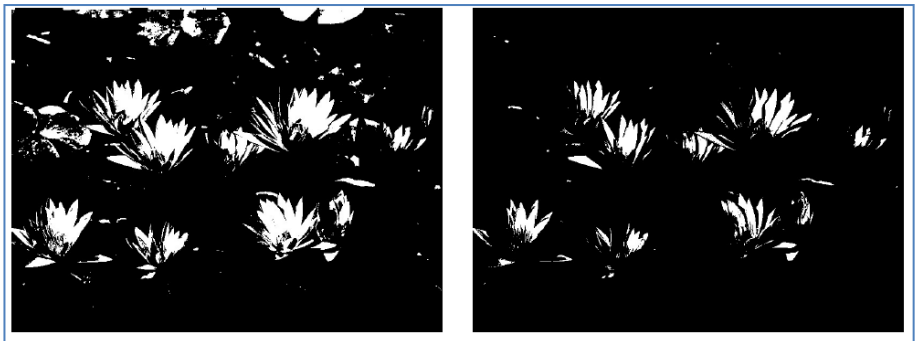
Resim üzerinde eşikleme işlemi yapıldığında belli özellikleri ortaya çıkarmak için, uygun eşik değeri kullanmamız gerekir. Fakat bu eşik değerinin hangi değer olması gerektiğini tam olarak bilemeyiz. İşte tam bu noktada histogram konusu devreye girmektedir.

Görüntü histogramı, gri-ton resmin üzerindeki piksel bilgilerini gösteren bir çubuk grafiktir. Grafiğin yatay x-ekseni 0-255 arasındaki gri ton değerlerini gösterir. y-ekseni ise bu değerlere sahip piksel sayılarını gösterir.



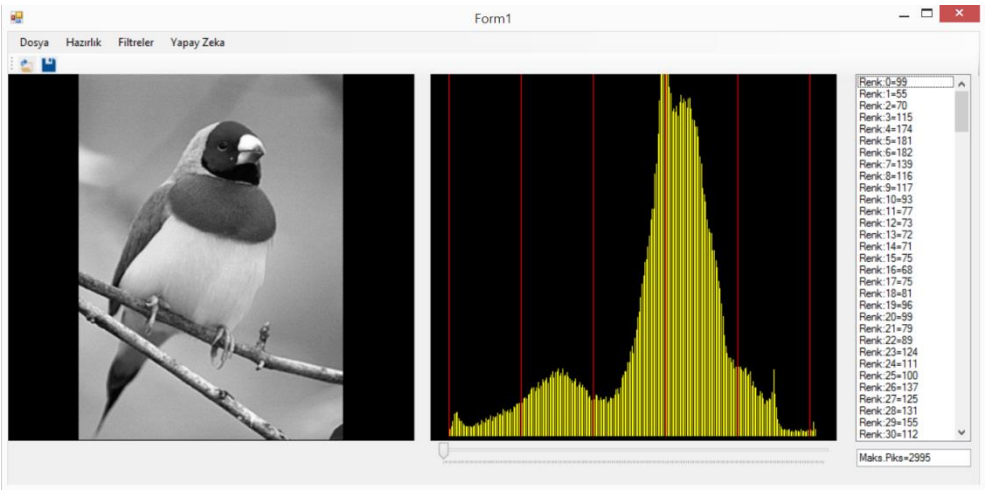
Aşağıdaki grafikte yatay eksen 0-255 arası eşit 4 parçaya bölünmüştür. Dikey eksen ise hangi renk tonundan en fazla adet varsa onun sayısına bağlı olarak (n ile gösterilmiştir) eşit kısımlara bölünmüştür. Buna göre en fazla piksel adedi 10 renk tonu civardında olduğu görülmektedir.

Resme bakıldığında renklerin koyu piksellerden oluştuğu gözükmektedir. Bu durum histogramın sol tarafa siyaha yakın bölgede toplanmasına yol açmıştır. Aynı zamanda histogramın orta bölgeye yakın bir yerde bir zirve yaptığını görmekteyiz. Bu bölgedeki renkler zambakların yapraklarının renginden kaynaklanmaktadır (suyun üstündeki geniş yapraklar). 128 den sonra renk sayılarının azaldığını görmekteyiz. Bunlar açık renkli tonlardır. Aşağıdaki gibi yapılacak iki farklı eşikle (128 veya 192) renklerin hangi nesnelere ait olduğunu görebiliriz.



128 den sonra eşik uygulanmıştır. 192 den sonra eşik uygulanmıştır.

Programlama




```
private void HISTOGRAM_Click(object sender, EventArgs e)
{
    ArrayList DiziPiksel = new ArrayList();

    int Gri = 0;
    Color OkunanRenk;
    int R = 0, G = 0, B = 0;

    Bitmap GirisResmi; //Histogram için giriş resmi gri-ton olmalıdır.
    GirisResmi = new Bitmap(pictureBox1.Image);

    int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
    int ResimYuksekligi = GirisResmi.Height;

    int i = 0; //piksel sayısı tutulacak.

    //GRİ DİZİYE ATMA***************
    for (int x = 0; x < GirisResmi.Width; x++)
    {
        for (int y = 0; y < GirisResmi.Height; y++)
        {
            OkunanRenk = GirisResmi.GetPixel(x, y);
            //Gri = (int)(OkunanRenk.R + OkunanRenk.G + OkunanRenk.B) / 3; //Griton resimde üç
kanal rengi aynı değere sahiptir.

            Gri = OkunanRenk.R;

            DiziPiksel.Add(Gri); //Resimdeki tüm noktaları diziye atıyor.
        }
    }

    //RENKLERİ SAYMA**************
```

```
int[] DiziPikselSayilari = new int[256];

for (int r = 0; r <= 255; r++) //256 tane renk tonu için dönecek.
{
    int PikselSayisi = 0;

    for (int s = 0; s < DiziPiksel.Count; s++) //resimdeki piksel sayısınca dönecek.
    {
        if (r == Convert.ToInt16(DiziPiksel[s]))
        {
            PikselSayisi++;

        }
    }
    DiziPikselSayilari[r] = PikselSayisi;
}


//Maksimum ve Minimum X değerlerini bulma === histogramın iki ucundaki sayıları bulma
int MaksX = 0;
int MinX = 255;

for (int s = 0; s < DiziPiksel.Count; s++) //resimdeki piksel sayısınca dönecek.
{
    int DiziRenkDegeri = Convert.ToInt16(DiziPiksel[s]);

    if (DiziRenkDegeri > MaksX)
    {
        MaksX = DiziRenkDegeri;
    }
    if (DiziRenkDegeri < MinX)
    {
        MinX = DiziRenkDegeri;
    }

}
txt1.Text = MinX.ToString();
txt2.Text = MaksX.ToString();

//Değerleri listbox'a ekliyor.
int RenkMaksPikselSayisi = 0; //Grafikte y eksenini ölçeklerken kullanılacak.
for (int k = 0; k <= 255; k++)
{
    //listBox1.Items.Add("Renk:" + k + "=" + DiziPikselSayilari[k]);
    //Maksimum piksel sayısını bulmaya çalışıyor.
    if (DiziPikselSayilari[k] > RenkMaksPikselSayisi)
    {
        RenkMaksPikselSayisi = DiziPikselSayilari[k];
    }
}

//Grafiği çiziyor. ************
Graphics CizimAlani;
Pen Kalem1 = new Pen(System.Drawing.Color.Yellow, 1);
Pen Kalem2 = new Pen(System.Drawing.Color.Red, 1);
CizimAlani = pictureBox2.CreateGraphics();

pictureBox2.Refresh();

int GrafikYuksekligi = 250;

double OlcekY = RenkMaksPikselSayisi / GrafikYuksekligi;
double OlcekX = 1.5;
int X_kaydirma = 10;
```

```
    for (int x = 0; x <= 255; x++)
    {
        CizimAlani.DrawLine(Kalem1, (int)(X_kaydirma + x * OlcekX), GrafikYuksekligi,
(int)(X_kaydirma + x * OlcekX), (GrafikYuksekligi - (int)(DiziPikselSayilari[x] / OlcekY)));

        if (x % 50 == 0)
        {
            CizimAlani.DrawLine(Kalem2, (int)(X_kaydirma + x * OlcekX), GrafikYuksekligi,
(int)(X_kaydirma + x * OlcekX), 0);
        }
    }
    txt3.Text = RenkMaksPikselSayisi.ToString();
}
```

KARŞITLIK (CONTRAST)

Bir görüntünün ayırt edilebilirliğini ifade eden kontras ifadesini Türkçe olarak da kullanmaktayız. Resim üzerindeki renk değerleri birbirine çok yakın değerlerde olursa (dar bir aralıkta toplandığında) düşük kontraslı resim olur. Bunu görebilmek için resmin Histogramını çizdirmek gerekir. Bu resimlerin renk değerlerini birbirinden uzaklaştırmak için resmin histogramı üzerinde gerdirme işlemi yapmak gerekir. Bunun için burada üç farklı örnek üzerinde durulacaktır.

Resmin kontrasını (karşıtlığını) artırmak ile keskinliğini artırmak (sharpen) aynı şey değildir. Kontrasda resim üzerindeki alanların renk değerleri birbirinden uzaklaştırılır. Keskinleştirmede ise alanların iç kısımları değil kenar kısımları belirgin hale getirilir. Bu konu ileride gelecektir.

Kontras örneği: Resim renk değerlerinin arası açılmış

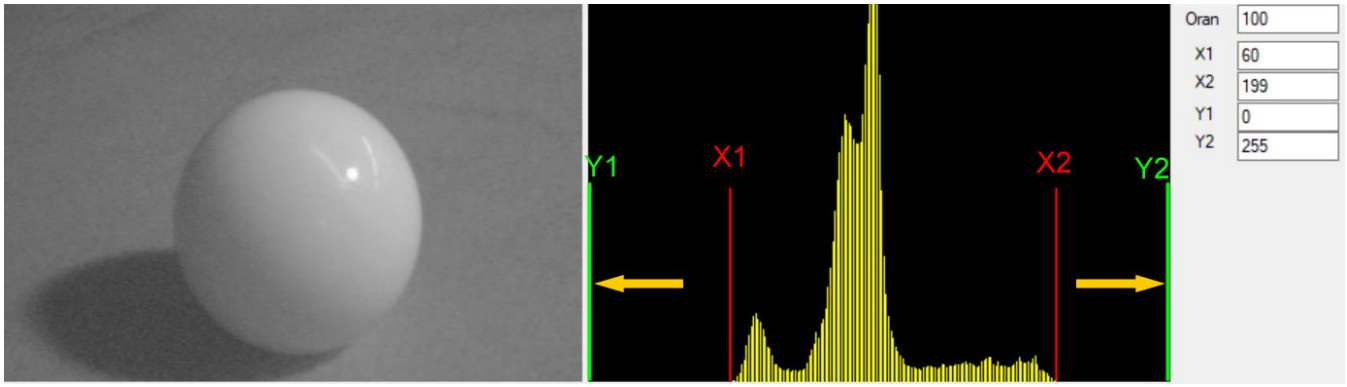
Keskinleştirme: Kenarlar belirgin yapılmış (sharpen)

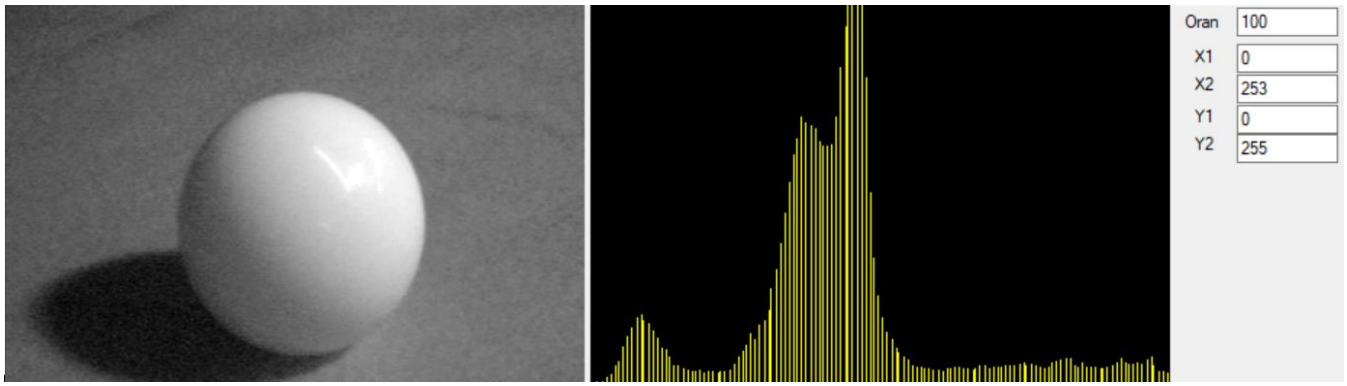
a) Lineer (Doğrusal) Kontras Uygulama (otomatik sınır belirleme)

Eğer histogram sınırları ortalarda bir yerde keskin bir şekilde oluşmuşsa tüm resim Lineer olarak siyah ve beyaz bölgeye doğru gerdirilebilir. Aşağıdaki örnekte spektrumun dış sınırları ortada bir yerlerde direk olarak sonlanmış (X1 ve X2 değerleri). Bu spektrum lineer olarak gerdirildiği zaman (Y1 ve Y2 ye çekildiğinde) herhangi bir renk kaybı oluşmaz. Fakat dar bir aralıktaki renkler (60 ile 199 arası görünüyor) geniş bir aralığa gerdirildiğinde (0-255 arasına) aralarda bazı renk değerleri doğal olarak resim üzerinde oluşmayacaktır. Buda resmi noktalı (gürültülü) gibi gözükmesine neden olmaktadır. İkinci resimde bunu görüyoruz. Kontras uygulanmış bu resmin spektrumuna da baktığımızda histogramı oluşturan çizgilerin aralarının açık olduğunu da görebiliyoruz. Bu tür noktalı oluşan görüntülerde resmi yumuşatmak için ileride gelecek konulardan bulanıklaştırma vs uygulanabilir. Bu durumda kenarlarda da bulanıklaşma olacaktır. Ardından Netleştirme (sharpening) uygulanabilir.

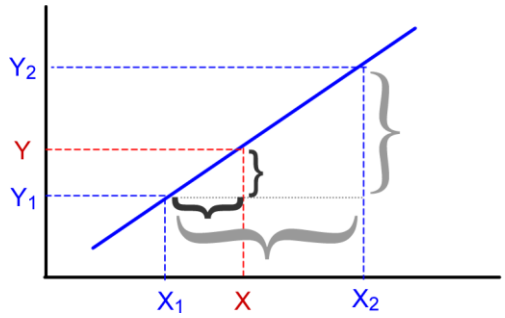
Burada şuna da dikkat çekmek gerekir. Bir resim üzerinde bütün renkler aşağıda olduğu gibi ortada toplanmış iken içerisinde bir kaçtane piksel beyaz yada siyah bölgelerde sınırların dışında olabilir. Oysa çok az sayıda birkaç tane olan bu pikseller sınırları değiştireceğinden resmin kontrasını da etkileyecektir. O yüzden çok az sayıdaki piksellerin buna yol açmaması için en fazla piksel sayısının belli bir oranı üzerinde olan piksellerin değerleri sınırları bulmak için kullanılmalıdır. Aşağıdaki örnekte en fazla piksel sayısının (grafiğin en üst tepesi) 1/100 kadar olan pikseller sınırları bulmak için (X1 ve X2 için) kullanılmıştır. Bu oran istenirse değiştirilebilir. Böylece çok

sayıdaki renkler resmi etkilememiş olur. Çok sayıdaki bu renkler zaten grafiğin ortalarında bir yerdeyse bunlar zaten sınırları etkilemeyecektir. Burada söz konusu olan sınırların dışında bulunan birkaç pikselin renk değeridir.





Kontras için kullanacağımız formülü aşağıdaki gibi elde edebiliriz. Sonuçta bu formül lineer olarak grafik üzerinde bir değeri elde etmek istediğimiz genel bir formüldür. Formülün çıkarılışı için Lineer bir grafik üzerindeki orantı kullanılabilir.



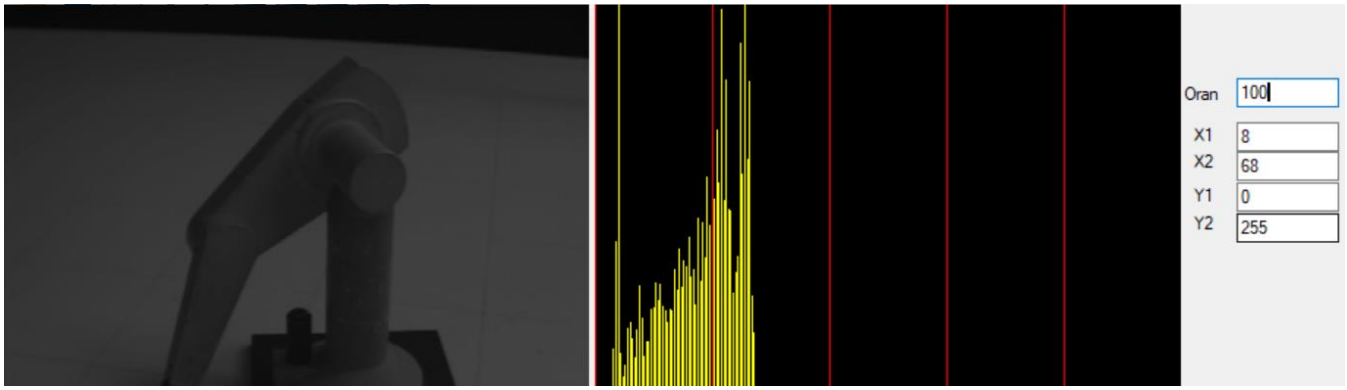
$$\frac{(Y_2 - Y_1)}{(X_2 - X_1)} = \frac{(Y - Y_1)}{(X - X_1)}$$

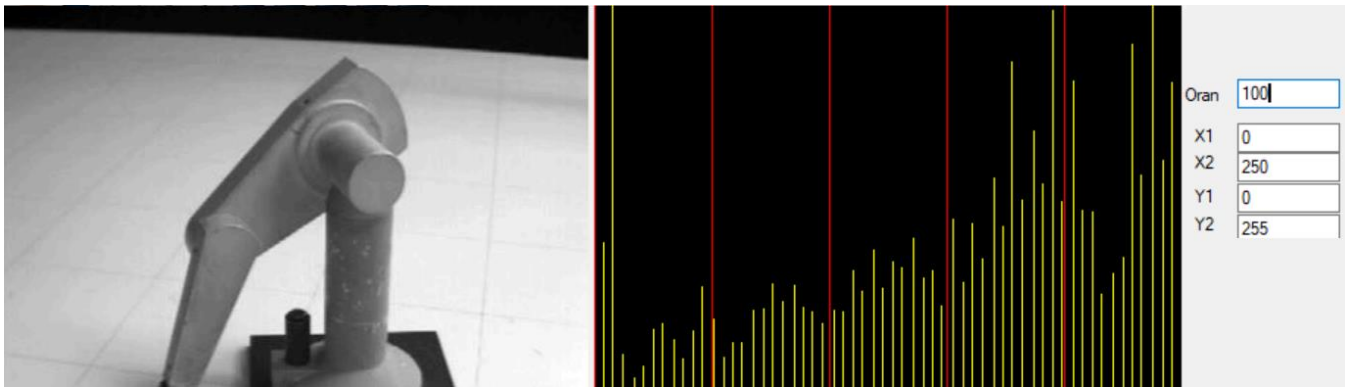
Not: Bu formüldeki Y değerlerini A,B,C olarak değiştirmek iyi olacaktır. Resmin Y eksenine ait değerler gibi algılanıyor. Y1,, A yap , Y2 -B yap, Y –C yap

$$Y = \frac{(X - X1)}{(X2 - X1)}(Y2 - Y1) + Y1$$

```
private void KONTRAS_histogram_Click(object sender, EventArgs e)
{
    Color OkunanRenk, DonusenRenk;
    //int R = 0, G = 0, B = 0;
    int Red;

    Bitmap GirisResmi, CikisResmi;
    GirisResmi = new Bitmap(pictureBox1.Image);


    int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı. İçerisine görüntü
yüklendi.
    int ResimYuksekligi = GirisResmi.Height;
```

```
    CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini oluşturuyor.
Boyutları giriş resmi ile aynı olur. Tanımlaması globalde yapıldı.

    int X1 = Convert.ToInt16(txt1.Text);
    int X2 = Convert.ToInt16(txt2.Text);
    int A = 0;  //Convert.ToInt16(textBox3.Text);
    int B = 255; //Convert.ToInt16(textBox4.Text);


    for (int x = 0; x < ResimGenisligi; x++)
    {
        for (int y = 0; y < ResimYuksekligi; y++)
        {
            OkunanRenk = GirisResmi.GetPixel(x, y);

            Red = OkunanRenk.R;
            //G = OkunanRenk.G;
            //B = OkunanRenk.B;

            //int Gri = (R + G + B) / 3;

            int Gri = Red;

            //*********** İnterpolasyon işlemi***************

            int X = Gri;

            int C = (((X - X1) * (B - A)) / (X2 - X1)) + A;

            if (C > 255) C = 255;
            if (C < 0) C = 0;

            DonusenRenk = Color.FromArgb(C, C, C);
            CikisResmi.SetPixel(x, y, DonusenRenk);
        }
    }
    pictureBox3.Refresh();
    pictureBox3.Image = null;
    pictureBox3.Image = CikisResmi;

}
```



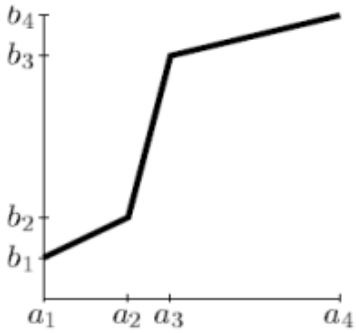


b) Lineer (Doğrusal) Kontras Uygulama-(Manuel Sınır Belirleme)

Yukarıdaki örnekte sınırlar (X1,X2) tüm spektrumu sonlara dayanmışsa (0-255 lere) bu resmi nasıl gerdireceğiz? Bu durumda resmin daha gerdirirsek sınırlar spektrumun dışına çıkacak demektir. Böyle bir durumda sınırı aşan değerler 0 ve 255 ile düzeltileceğinden resmin doğallığı kaybolacaktır. Yani resimde hiç bulunmasa bile siyah ve beyaz alanlar artacaktır. Bunun çözümü için bir sonraki çoklu Lineer Grafik uygulanabilir. Fakat burada direk kişi çekmek istediği sınırları kendisi belirlesin ve bunun sonucunun nasıl olacağını görelim.

c) Çoklu Lineer (doğrusal) Kontras Uygulama-(Manuel Sınır Belirleme)

Bir önceki maddede anlatıldığı şekilde resmin sınırları spektrumun sınırlarına dayanmış olabilir. Bu durumda tüm resme aynı genişletmeyi uygulamak spetkrumun dışına çıkmaya neden oluyordu. Bunun yerine resim üzerinde bazı bölgeleri daraltarak, bazı bölgeleri genişleterek farklı kontras bölgeleri oluşturarak uygun bir çözüm bulabiliriz.



Şekil. Lineer germe fonksiyonu

$$y = \frac{b_{i+1} - b_i}{a_{i+1} - a_i}(x - a_i) + b_i$$

2006.]

Not: Yukarıda verilen histogram örnekleri renkli resimlere de uygulanabilir. Histogram hesabı gri renk üzerinden yapılırsa ve bulunan değerler üç kanal içinde uygulandığında sonuçlar gözlemlenebilir. Bu denemeler ödevler içinde istenecektir.

----------***----------

Renkli Resim Üzerinde Kontras Artırma-(Histogramsız uygulama)

Aşağıda hazır kullanım için geliştirilmiş formüller kullanılarak renkli resim üzerinde kontras yapabiliriz. Burada temel mantık resmi 128 değerinin üzerindeki renkleri yukarı 255 doğru, aşağıdaki renkleri 0 doğru ötelemekten ibarettir. F sayısı öteleme miktarını belirlerken, bunu kullanarak her üç kanalın renk değerini belirleyebiliriz. Bu formül kullanıldığında renk değerleri 255 üzerine ve 0 altına inebileceğinden bu sınırlarda kesmek resmin aslında doğallığını kaybettirir. Yani aşırı kontrasta resim üzerinde beyaz ve siyah bölgeler belirgin olarak artabilir. Onun yerine yukarıda kullanılan histogram yöntemi ile kontras uygulamak daha iyi sonuç verir. Yani hangi sınır değerlerin nereye çekileceğine grafikle görerek karar vermek daha iyi sonuçlar verir. Yine de bu formül pratik kullanım için tercih edilebilir.

Görüntünün kontrasını ayarlama da İlk adım, aşağıdaki formülle verilen bir kontrast düzeltme faktörünü hesaplamaktır:

$$F = \frac{259(C + 255)}{255(259 - C)}$$

F: Kontras düzeltme faktörü (double)

C=Kontras seviye katsayısı

Algoritma'nın doğru çalışması için kontrast düzeltme faktörü (F) değerinin bir tamsayı değil, ondalık sayı (double) olması gerekir. Formüldeki C değeri, istenilen kontrast seviyesini belirtir.

Bir sonraki adım, gerçek kontrast ayarını yapmaktır. Aşağıdaki formüller kırmızı (R), yeşil (G), mavi (M) renkler için verilmiştir.

$$R' = F(R - 128) + 128$$

$$G' = F(G - 128) + 128$$

$$B' = F(B - 128) + 128$$

Bu formüllerden çıkan R,G,B renk değerlerinin 0-255 geçerli renk aralığında olması gerekir.

Kırmızı, yeşil ve mavi değerlerin 0'dan 255'e kadar geçerli bir aralıkta olmasını sağlar. Kontrast seviyesi (C) ise +255 ile -255 arasında olmalıdır. Pozitif değerler kontrast miktarını artırırken, Negatif değerler kontrast miktarını düşürecektir.

----------***----------

Program Kodları

Aşağıdaki resimler Orijinal resim, +128 Kontrast ve -128 kontrast miktarlarını göstermektedir.

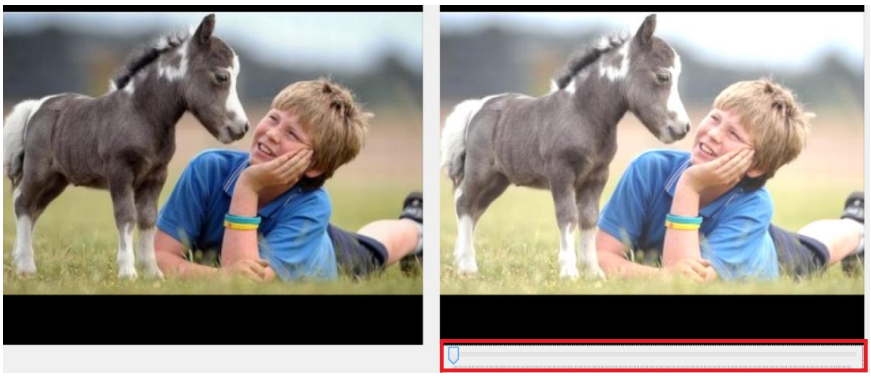
```
public Bitmap Karsitlik()
{
    int R = 0, G = 0, B = 0;
    Color OkunanRenk, DonusenRenk;
    Bitmap GirisResmi, CikisResmi;
    GirisResmi = new Bitmap(pictureBox1.Image);
```

```
int ResimGenisligi = GirisResmi.Width; //GirisResmi global tanımlandı.
int ResimYuksekligi = GirisResmi.Height;
CikisResmi = new Bitmap(ResimGenisligi, ResimYuksekligi); //Cikis resmini oluşturuyor. Boyutları
giriş resmi ile aynı olur.

double C_KontrastSeviyesi = Convert.ToInt32(textBox1.Text);

double F_KontrastFaktoru = (259 * (C_KontrastSeviyesi + 255)) / (255 * (259 -
C_KontrastSeviyesi));


for (int x = 0; x < ResimGenisligi; x++)
{

    for (int y = 0; y < ResimYuksekligi; y++)
    {
        OkunanRenk = GirisResmi.GetPixel(x, y);
        R = OkunanRenk.R;
        G = OkunanRenk.G;
        B = OkunanRenk.B;

        R = (int)((F_KontrastFaktoru * (R - 128)) + 128);
        G = (int)((F_KontrastFaktoru * (G - 128)) + 128);
        B = (int)((F_KontrastFaktoru * (B - 128)) + 128);

        //Renkler sınırların dışına çıktıysa, sınır değer alınacak.
        if (R > 255) R = 255;
        if (G > 255) G = 255;
        if (B > 255) B = 255;

        if (R < 0) R = 0;
        if (G < 0) G = 0;
        if (B < 0) B = 0;

        DonusenRenk = Color.FromArgb(R, G, B);
        CikisResmi.SetPixel(x, y, DonusenRenk);
    }

    }
    pictureBox2.Image = CikisResmi;
}
```

ÖDEVLER

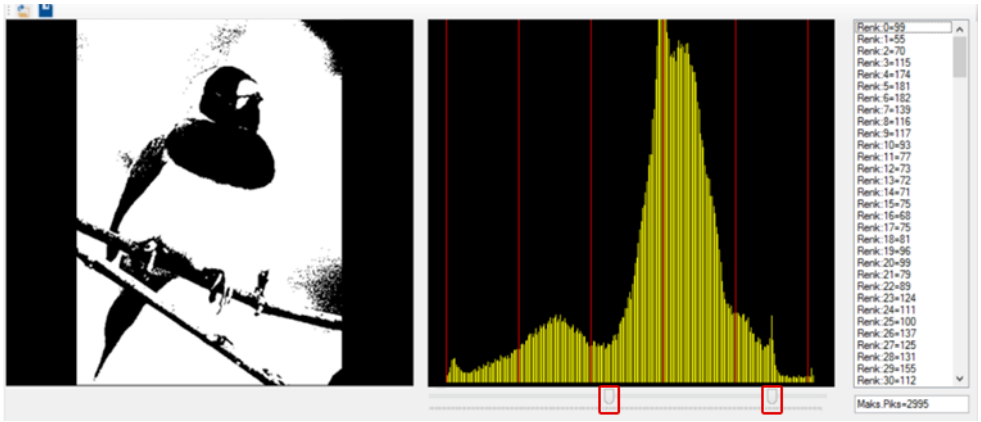
Ödev 1: (Parlaklık)

Sürgü kullanarak bir resmin parlaklığını artırma yada düşürme işlemini dinamik olarak gerçekleştirin.



Ödev 2: (Eşikleme)

Form üzerinde iki tane sürgü kullanarak Eşikleme yapan bir program yazınız. Sürgüler grafik altında x ekseni boyunca olsun. Sürgü ilerletildikçe resmin nasıl değiştiği gözlemlenebilsin. Olay şöyle çalışsın. Sürgülerin birisi 50, diğeri 200 gösteriyorsa, Gri renk değeri 50 ile 200 da ise o pikseli direk göstersin. Bu iki sınırın dışında ise göstermesin yani siyah yapsın.



Aynısını Renkli Resim içinde yapın. Yani Kırmızı, Yeşil ve Mavi kanallara göre 3 tane ayrı Histogram grafiği çizdirin. Her grafiğin altında iki tane sürgü bulunsun. Okunan renk Kanal değeri (örneğin kırmızı) Bu sürgülerin değerleri arasında ise kırmızıyı kendi renginde göstersin. Dışında ise hiç göstermesin (yani siyah yapsın). Aynı işlemi diğer üç kanal içinde yapmanız gerekir.

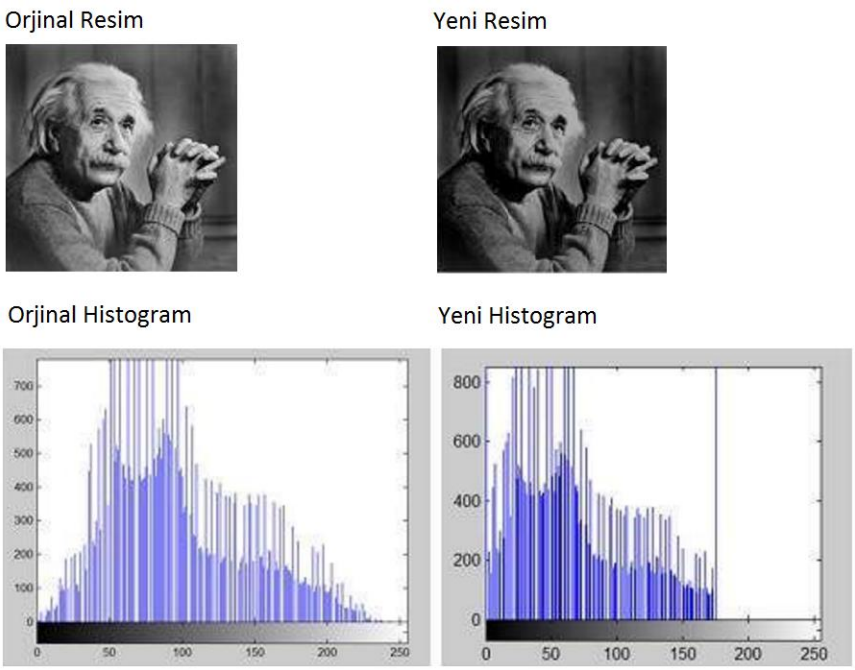
Histogram grafikleri bize hangi sınırı dışarıda tutacağımız daha iyi gösterir.

Ödev 3: (Histogram ile Resmi Sıkıştırma)

a) Aşağıdaki gibi bir siyah beyaz resmin belli bir sınırın ötesindeki beyaz renkleri sıfırlayan ve o bölgelerin koyulaşmasını sağlayan ve bunu histogram üzerinden ayarlayan programı yazınız. Histogramın altındaki sürgü sürüklendiğinde resim değişsin ve resmin histogramı tekrar yenilensin. Aşağıdaki histogramda tek bir sürgü kullanılmıştır. Beyaz bölge geriye çekildiğinde tüm 0-255 arasındaki sayılar Ölçeklenerek yenilenmesi gerekir. Diyelim sürgü 180 çekilirse, önceden 255 olan renk artık 180 değerine gelmiş olması gerekir. Tüm diğer renk değerleri de 0-180 arası ölçeklenmelidir.

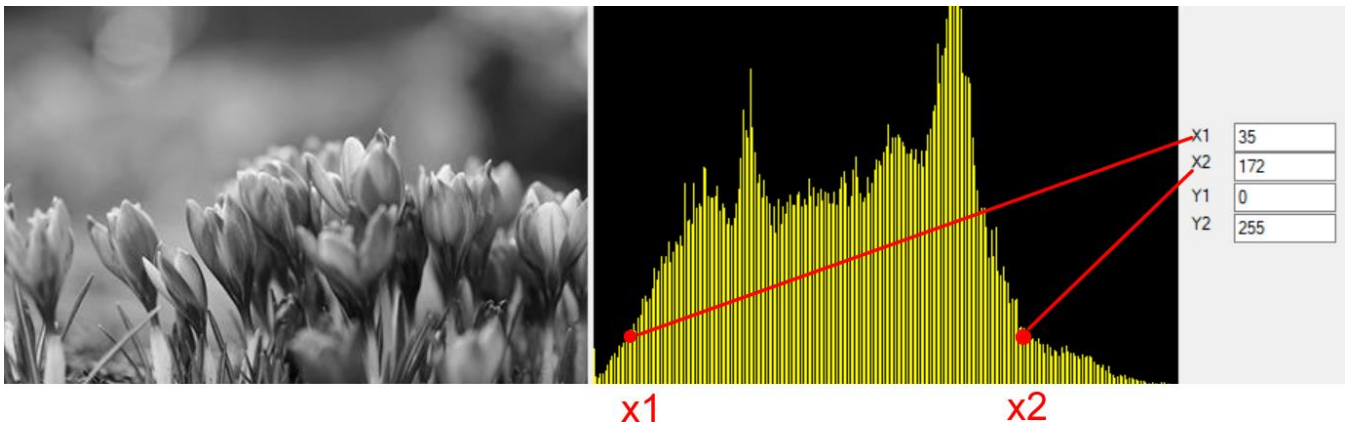
b) Aynı uygulamayı çift sürgü ile yapın. Yani resmi hem aşağıdan yukarı doğru sıkıştırın, hemde üstten aşağı doğru sıkıştırın. Böyle bir resimde renk değerleri belli bir aralık içine gelmiş olacaktır.

c) Yeni oluşan resim üzerinde istenen noktaya Mouse ile tıklandığında Textbox içinde o noktanın X ve Y koordinatını göstersin. Ayrıca R, G, B şeklinde üç renk değerinide göstersin. Böylece resmin hangi kısmı nasıl bir renk değeri almış kontrol edilsin. Bunun için MouseMove ve MouseDown olaylarına bakmalısınız. Örnek kodlar İnt.Tab.Prg dersinin sayfalarında vardır.



Ödev 4: (Gri Resim üzerinde Histogram ile Resmin Kontrasını Artırma)

Aşağıdaki gibi bir resmin histogramını çizdirdikten sonra, Kontras uygulayalım. Kontrasın uygulanacak sınır değerlerini Grafiğin bulunduğu picturebox üzerinden Mouse ile tıklayarak belirleyelim. Mouse'un sol tuşuna Picturebox2 üzerinde tıklayınca okuduğu x koordinatını X1 kabul edilsin Textbox1 de gösterilsin. Daha sonra mouse'un Sağ tuşuna Picturebox2 üzerinde tıklayınca okunan koordinatın x değeri X2 olarak kabul edilsin ve bu da Textbox2 de gösterilsin. Bu sınırlar 0 ve 255 çekilecek şekilde bir başka düğmeye tıklayınca kontras uygulasın. (Dikkat: Pixturebox 2 nin boyu 400 ise, x=400 koordinatından alınan renk değeri 400 olamaz. Bu nedenle alınan X koordinatlarını ölçeklemelisiniz. Ölçek = 255/400 olur. Kendi uygulamanıza göre ayarlayın).



Ödev 5: (Renkli resimler üzerinde histogramla Kontras artırma)

a) Kontras uygulamasını her bir Renk kanalı için ayrı ayrı uygulayın. Önce her renk kanalının Histogramını çizdirin. Grafik üzerinden 0 ve 255 çekilecek sınır değerlerini üstteki örnekde olduğu gibi mouse ile tıklayarak bulun. Yada tahmini olarak renk için sınırları belirleyin (50 lik kızmızı dikey çizgileri çizdirirseniz gözle daha iyi görürsünüz). Bu değerleri (6 tane değer olur) textboxlara yazın. Ardından başka bir butona basınca bu her bir renk değeri için kendi sınır değeri kullanılarak yeni kontras uygulanmış renk değerleri ile resmi yeniden oluşturun. Bu yeni oluşan resmi, orijinal resimle karşılaştırın. Renkli resimde başarılı bir Kontras yapabildinizmi görün. Renkli resimde iyi bir kontras oluşturmak için ne yapılabilir yazarak yorumlayın.

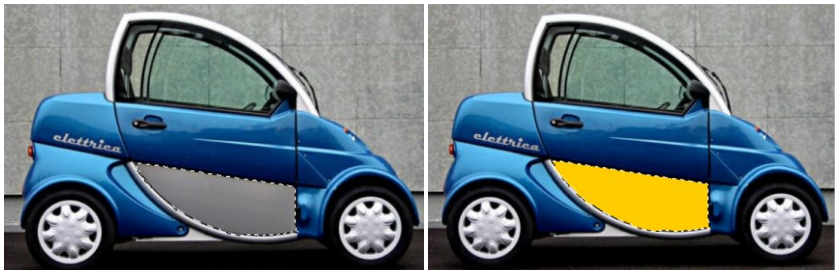
b) Ders notları içinde renkli resim için verilen hazır bir formül vardır. Bu formülü kullanarak elde ettiğiniz kontras sonucunu kendi bulduğunuz çözüm ile karşılaştırın. Ödev içinde yorumlayarak gösterin.

Ödev 6: (Mouse ile tıklayarak belirlenen bölgenin içerisindeki renkleri değiştirme)

Picturebox üzerinde mouse ile tıklayarak belli bir kapalı bölge çizin. Alanı gözle de görmek için kesikli olarak bir çizgi ile çerçeve oluşturun. Daha sonra bu bölgenin içerisindeki renk değerleri için Sürgü ile

i) Parlatma (Brightness)
ii) Kontras Uygulayın. Kontrası hem histogram çizdirerek, hemde hazır formül ile uygulayın.
iii) Renk değişimi yaptırın. R, G, B nin her bir değeri için ayrıca üç sürgü kullanabilirsiniz.

a) Çizgilerle belli bir alanı çizmek için İnt.Tab.Prg dersinin sayfasında örnek kodlar vardır. Oradaki bilgilerle bu çizimi yaptırın.
b) Tıklanan noktanın Kapalı bir poligonun içerisinde olup olmadığını bulmak için Analik Geometri konuları içinde bir araştırma yapmalısınız. Yani tıklanan nokta bir poligonun (çokgenin) içinde mi kalır yoksa dışında mı kalır bunu araştırın. Bulduğunuz formül ve yöntemleri anlatın. Ezbere olmasın, mantığını anlayın.



Ödev 7:

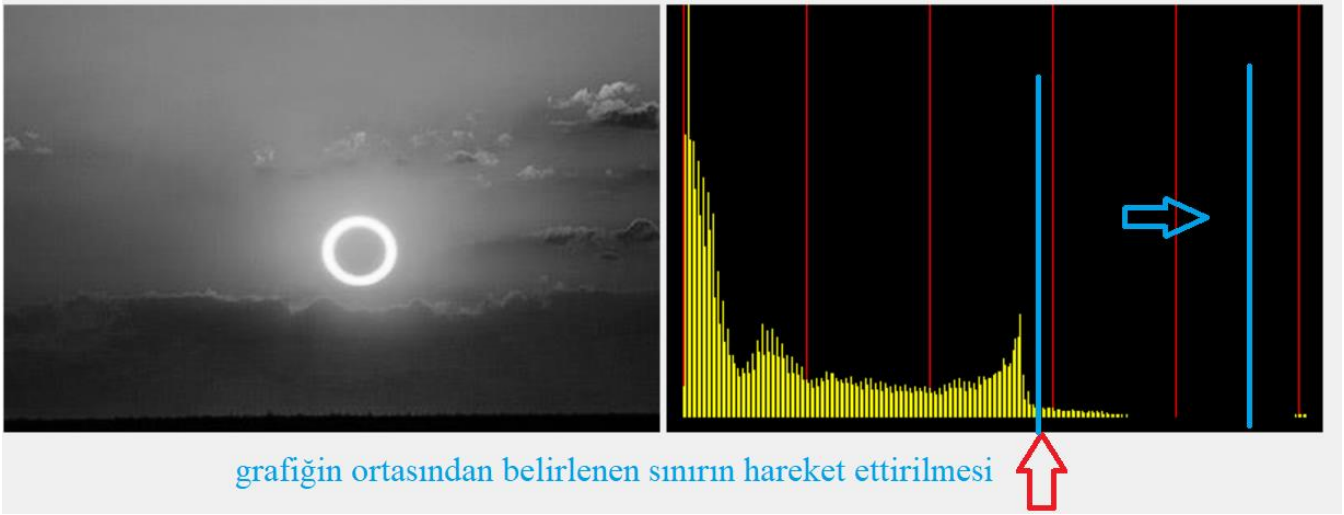
Renkli Resimler üzerinde farklı kanalların histogramlarını çizdirme yapınız. Sonuçları yorumlayarak bu bilgiler nelerde kullanılabilir yazınız.

Ödev 8:

Eski tarihi bir siyah resmi alıp bunun üzerinde belirleyeceğiniz 10 tane noktanın rengini siz verin. Bu renk tahminlerinden yola çıkarak tüm resmi renklendirme işlemini kendi yazacağınız algoritma yapsın…

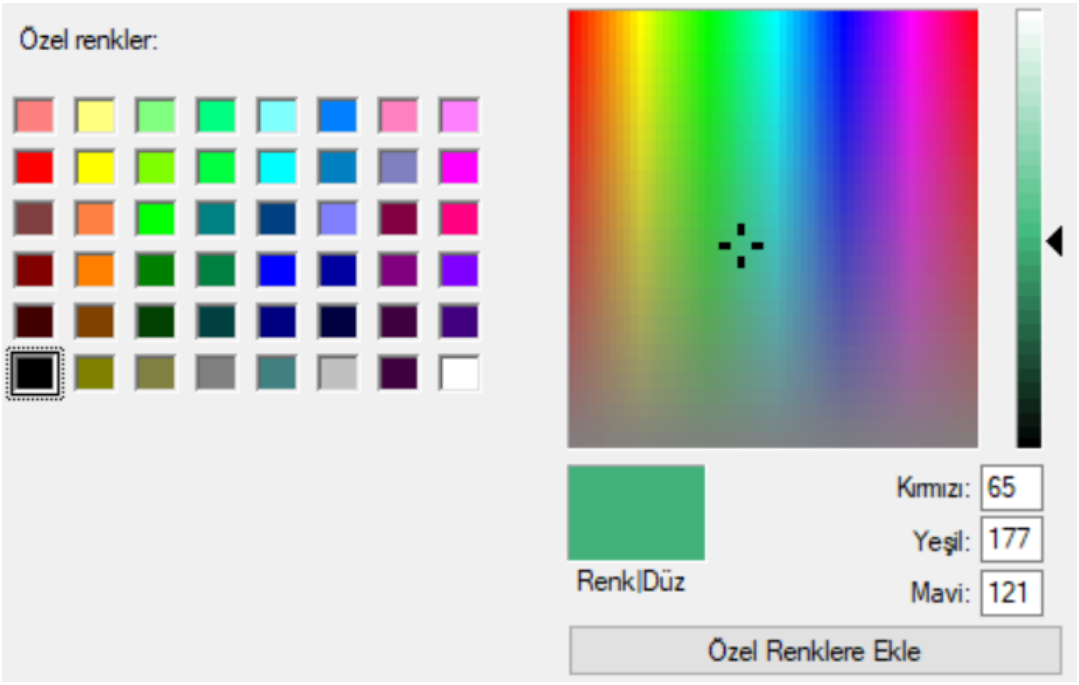
Ödev 9:

İyi bir gece görüş dürbünü yapabilmek için sadece ortamın ışığını artırmak yetmez. Bunun için renklerinde birbirinden uzaklaştırılması (kontrası artırmak) gerekir. Bu işlem için kendiniz bir gece görüş kamerası algoritması geliştirin. Gece karanlığında çok az ışıkta çekilmiş bir çok fotografı en iyi gündüz çekimine dönüştüren algoritmayı yazın. Çekimler geceleyin bir ay ışığı görüntüsü olabilir, karanlık bir odada camdan az bir ışık geliyor olabilir. İyi bir algoritma geliştirmek için kontras sınırlarını kendiniz sürükleyerek ayarlamaya çalışın. Ayrıca belli bölgelerde ışık artırma işlemide yapın. Birde Histogram grafiğinin ortasında bir noktadan kaydırma yaparak onun altındaki renkleri açarak, yukarısındaki renkleri ise bir birine yaklaştırarak daha iyi görüntüler elde etmeye çalışın.



Ödev 10:

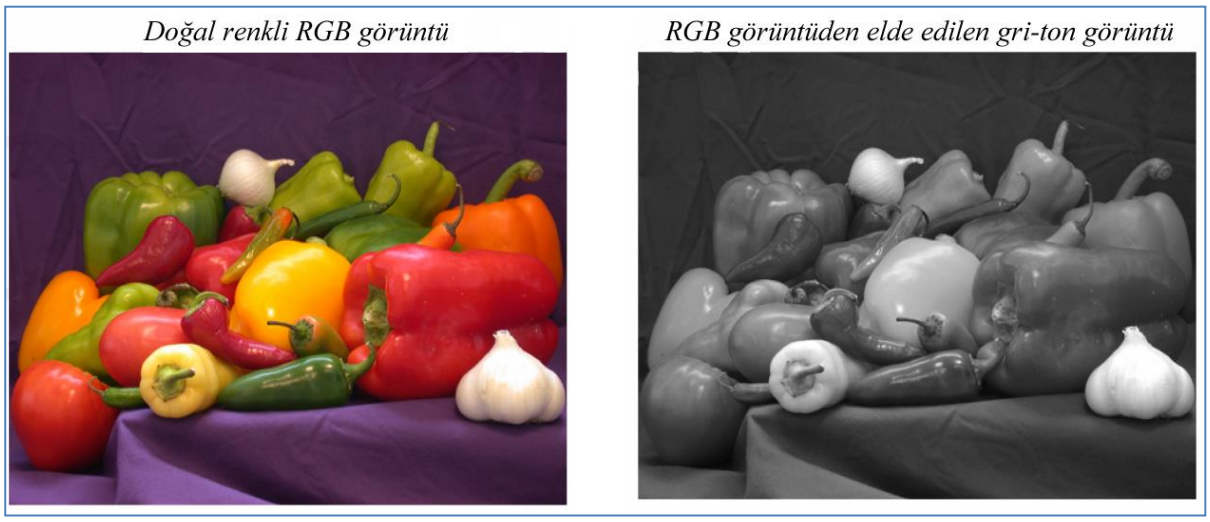
Aşağıdaki gibi kişiye özel renk paleti oluşturmak için bir uygulama yapın. Bunun için Pikturebox içinde 16 milyon rengin her birini oluşturun. (R*G*B=255*255*255 = 16.581.375). Mantığını kendiniz bulacaksınız. Ardından yan tarafa bir skala daha oluşturun. Ortadan bir renk seçildiğinde o rengin tonlarını açıktan koyuya doğru bir şerit halinde göstersin. Böylece kişi ortadan hangi rengi atamak istediğini belirlesin, yandan da bu rengin tonlarını oluştursun. En son karar verdiği rengin RGB değerlerini aşağıda göstersin ve Bu rengi kaydet dediği zaman yanda oluşturulan küçük pencerelerin içine rengi kaydetsin (yani orda göstersin). Böylece kişiye özgü bir renk paleti oluşmuş olsun.



Ödev 11:

Gri Resmi, renkli resme dönüştüren bir program yazın. Bu dönüşüm normal algoritma ile zordur. İçerisinde yapay zekaya ihtiyaç vardır. Yapay zeka kısmını kendimiz şu şekilde oluşturabiliriz. Resmin üzerinden yaklaşık 10 tane nokta seçin. Bu noktalardaki renkleri yan tarafta bir Textboxlarda gösterin. Aldığınız bu örnek rengin tahmini olması gereken renk değerini atayın. Bu tahmini renkler üzerinde yukarı aşağıya tolerans belirleyelim. Bu şekilde

10 tane rengin tahmini üzerinde resmi tekrar renkli resme dönüştürün. Yada kendiniz farklı bir algoritma geliştirin. 3 tane resmi aynı ekranda gösterin (Orijinal, Gri, Dönüşmüş Resim)



Ödev 12

Aşağıda resimlerde görüldüğü gibi bir resmi sulu boyama efekti verecek şekilde renklendirme algoritması geliştirin. Burada kırmızı bölgedeki detaylar kırmızı tek renkle boyanıyor muş gibi olsun. Mavilerde tek mavi renkle boyanıyor muş gibi olsun. Bunun **Eşikleme mantığını** kullanabilirsiniz. Tabii bu mantığı daha da geliştirmeniz gerekir. Algoritmanın çalıştığını farklı resimler gösterin. (Not: Aşağıdaki örnek hatalıdır. Çünkü sarı çiçekte oranın hepsini sarı görmek isteriz. Mavini olduğu bölge hepsi mavi olmalıdır. Kırmızı da aynı şekilde)