Gri Image, Binary Image, Histogram, Birleştirme, Panorama

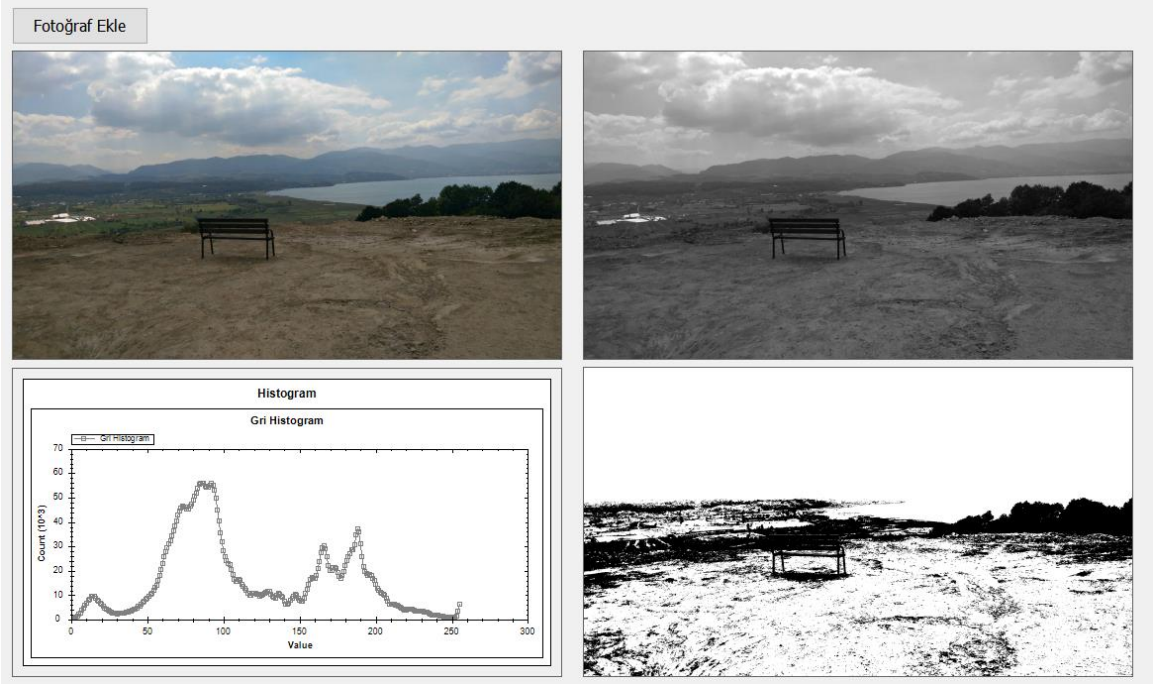
Dosya adını getir fonksiyonu:

```
public static string DosyaAdiGetir()
        {
            OpenFileDialog ofd = new OpenFileDialog();
            ofd.Filter = "Images (*.BMP;*.JPG;*.GIF)|*.BMP;*.JPG;*.GIF|" + "All
files (*.*)|*.*";
            if (ofd.ShowDialog() == DialogResult.OK)
                return ofd.FileName;
            else
                return "";
        }
```

Resim dizisini getir fonksiyonu:

```
  public static Image<Bgr, byte>[] ResimDiziGetir()
        {
            OpenFileDialog ofd = new OpenFileDialog();
            ofd.Filter = "Images (*.BMP;*.JPG;*.GIF)|*.BMP;*.JPG;*.GIF|" + "All
files (*.*)|*.*";
            ofd.Multiselect = true;
            if (ofd.ShowDialog() == DialogResult.OK)
            {
                Image<Bgr, byte>[] images = new Image<Bgr,
byte>[ofd.FileNames.Length];
                for (int i = 0; i < images.Length; i++)
                {
                    images[i] = new Image<Bgr, byte>(ofd.FileNames[i]);
                }
                return images;
            }
            else
            {
                return null;
            }
        }
```

Gri Image, Binary Image, Histogram



Gri resimi elde etme:

Islem sınıfı:

```
public static Image<Gray, byte> GriGetir(Image<Bgr, byte> img)
{
    Image<Gray, byte> gri = img.Convert<Gray, byte>();
    return gri;
}
```

Buton kodu:

```
//gri image
griFoto = Islem.GriGetir(renkliFoto);
griImageBox.Image = griFoto;
```

Histogram:

Islem sınıfı:

```
public static Mat HistogramGetir(Image<Gray, byte> griFoto)
```

```
{
    DenseHistogram hist = new DenseHistogram(256, new RangeF(0, 256));
    hist.Calculate(new Image<Gray, Byte>[] { griFoto }, false, null);
    Mat m = new Mat();
    hist.CopyTo(m);
    return m;
}
```

Buton kodu:

```
//histogram
Mat m = Islem.HistogramGetir(griFoto);
histogramBox.ClearHistogram();
histogramBox.AddHistogram("Gri Histogram", Color.Gray, m, 256, new
float[] { 0.0f, 256.0f });
histogramBox.Refresh();
```

Binary Image:

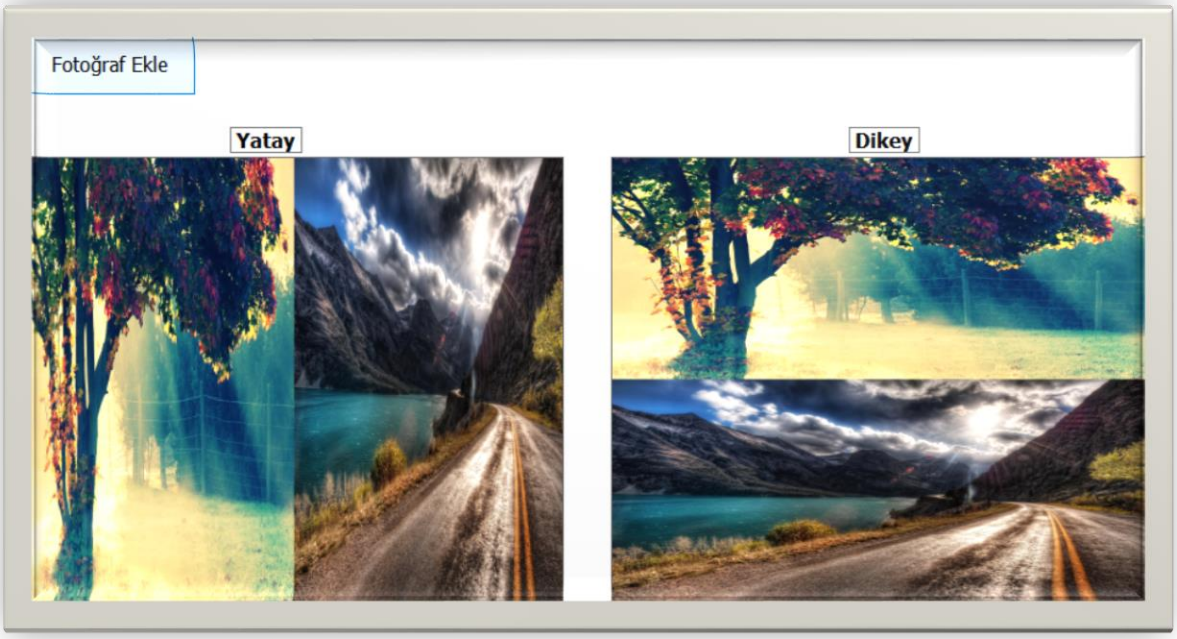
Islem sınıfı:

```
public static Image<Gray, byte> BinaryGetir(Image<Gray, byte> gri)
{
    int threshold = 70;
      Image<Gray,byte> binary = gri.ThresholdBinary(new Gray(threshold),
new    Gray(255));
    return binary;
}
```

Buton kodu:

```
//binary image
binaryFoto = Islem.BinaryGetir(griFoto);
binaryImageBox.Image = binaryFoto;
```

Birleştirme



Islem sınıfı:

```
public static Image<Bgr, byte> YatayBirlestir(Image<Bgr, byte>[] images)
        {

            int ImageWidth = 0;
            int ImageHeight = 0;

            //get max width and max height
            for (int i = 0; i < images.Length; i++)
            {
                //calculate max height
                if (images[i].Height > ImageHeight)
                    ImageHeight = images[i].Height;
                //calculate new width
                ImageWidth += images[i].Width;
            }


            //declare new image (large image).
            Image<Bgr, Byte> imageResult = new Image<Bgr, Byte>(ImageWidth,
ImageHeight);
```

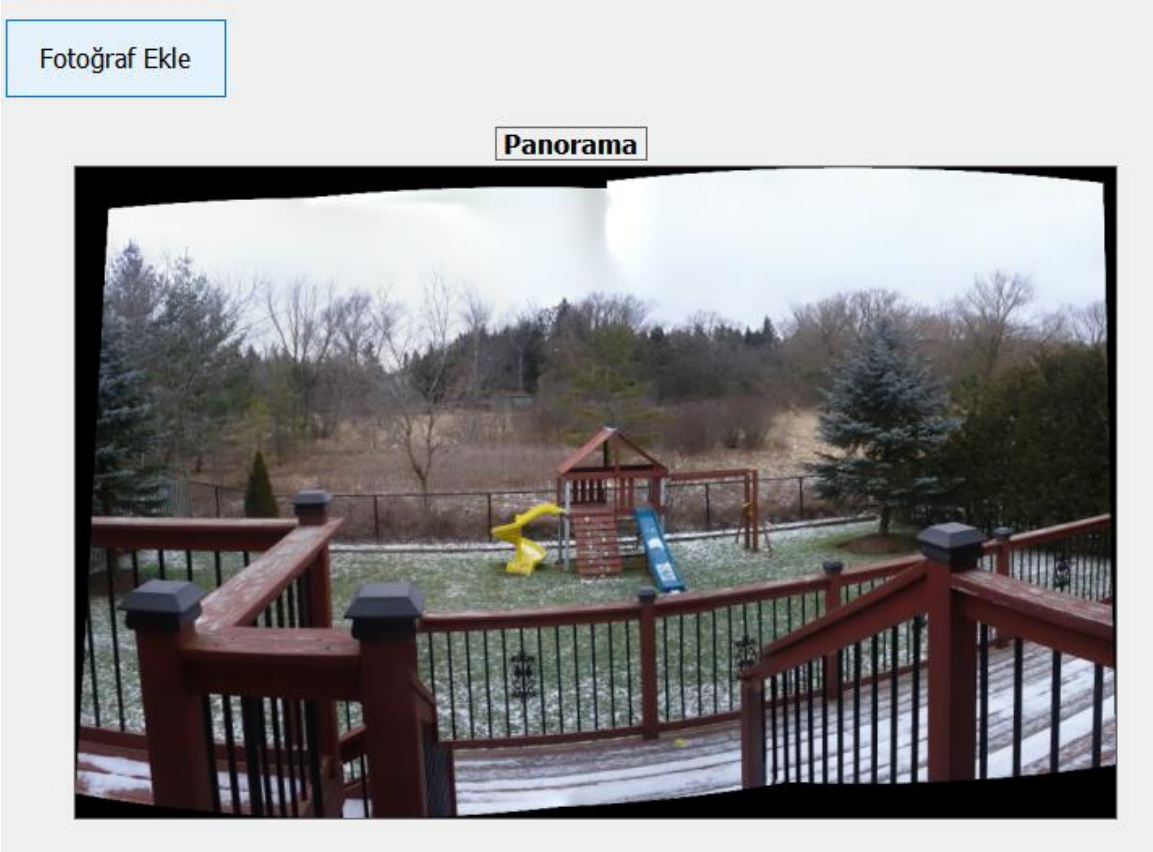
```
            int width = 0;
            for (int i = 0; i < images.Length; i++)
            {
                imageResult.ROI = new Rectangle(width, 0, images[i].Width,
images[i].Height);
                images[i].CopyTo(imageResult);
                width += images[i].Width;
            }

            imageResult.ROI = Rectangle.Empty;
            return imageResult;
        }

        public static Image<Bgr, byte> DikeyBirlestir(Image<Bgr, byte>[] images)
        {

            int ImageWidth = 0;
            int ImageHeight = 0;

            //get max height and max width
            for (int i = 0; i < images.Length; i++)
            {
                //calculate max width
                if (images[i].Width > ImageWidth)
                    ImageWidth = images[i].Width;
                //calculate new height
                ImageHeight += images[i].Height;
            }


            //declare new image (large image).
            Image<Bgr, Byte> imageResult = new Image<Bgr, Byte>(ImageWidth,
ImageHeight);

            int height = 0;
            for (int i = 0; i < images.Length; i++)
            {
                imageResult.ROI = new Rectangle(0, height, images[i].Width,
images[i].Height);
                images[i].CopyTo(imageResult);
                height += images[i].Height;
            }

            imageResult.ROI = Rectangle.Empty;
            return imageResult;
        }
    }
```

Buton kodu:

```
            Image<Bgr, byte>[] fotograflar = Islem.ResimDiziGetir();
            if (fotograflar!=null)
            {
                yatayImageBox.Image = Islem.YatayBirlestir(fotograflar);
                dikeyImageBox.Image = Islem.DikeyBirlestir(fotograflar);
```

```
        }
        else
        {
            MessageBox.Show("Lütfen bir resim seçiniz..");
        }
```

Panorama (Stitcher)



Islem sınıfı:

```
public static Mat Panorama(Image<Bgr, byte>[] fotograflar)
 {

     VectorOfMat matDizi = new VectorOfMat();
     for (int i = 0; i < fotograflar.Length; i++)
     {
         matDizi.Push(fotograflar[i].Mat);
     }
```

```
            Mat result = new Mat();
            Stitcher stitcher = new Stitcher(false);

            var sonnuc = stitcher.Stitch(matDizi, result);
            if (sonnuc)
            {
                return result;
            }
            else
            {
                return null;
            }

        }
```

Buton kodu:

```
              panoramaLabel.Text = "Panorama";
            Image<Bgr, byte>[] fotograflar = Islem.ResimDiziGetir();
            if (fotograflar != null)
            {
                Mat sonuc = Islem.Panorama(fotograflar);
                if (sonuc != null)
                {
                    panoImageBox.Image = Islem.Panorama(fotograflar);
                }
                else
                {
                    panoramaLabel.Text = "Panorama Değildir";
                }
            }
            else
            {
                MessageBox.Show("Lütfen bir resim seçiniz..");
            }
```

SON