```
int[] histogramObjek = new int[256]; //histogram grayscale
Bitmap bmp5 = new Bitmap(pictureBox2.Image);//image grayscale
Bitmap bmp6 = new Bitmap(pictureBox3.Image);//image biner
int height = bmp5.Height;
int width = bmp5.Width;
Color h;
Color h2;

// GLCM skala 16 x 16 supaya matrik GLCM tidak terlalu besar dimensinya
// akan mempengaruhi waktu eksekusi
#region
int intensitas5 = 0;
int[,] matrikImage = new int[height, width];//matrik image yang sudah ternormalisasi, isi intensitas gray 0-15
double[,] matrikCooccurrence = new double[16, 16]; // matriks co-occurrence
for (int y = 0; y < height; y++)
{
    for (int x = 0; x < width; x++)
    {
        h = bmp6.GetPixel(x, y); //biner
        h2 = bmp5.GetPixel(x, y); //grayscale
        if (h.R == 255)
        {
            intensitas5 = h2.B;

            if (intensitas5 >= 0 && intensitas5 <= 15)
            {
                intensitas5 = 0;
            }
            if (intensitas5 >= 16 && intensitas5 <= 31)
            {
                intensitas5 = 1;
            }
            if (intensitas5 >= 32 && intensitas5 <= 47)
            {
                intensitas5 = 2;
            }
            if (intensitas5 >= 48 && intensitas5 <= 63)
            {
                intensitas5 = 3;
            }
            if (intensitas5 >= 64 && intensitas5 <= 79)
            {
                intensitas5 = 4;
            }
            if (intensitas5 >= 80 && intensitas5 <= 95)
            {
                intensitas5 = 5;
            }
            if (intensitas5 >= 96 && intensitas5 <= 111)
```

```
            {
                intensitas5 = 6;
            }
            if (intensitas5 >= 112 && intensitas5 <= 127)
            {
                intensitas5 = 7;
            }
            if (intensitas5 >= 128 && intensitas5 <= 143)
            {
                intensitas5 = 8;
            }
            if (intensitas5 >= 144 && intensitas5 <= 159)
            {
                intensitas5 = 9;
            }
            if (intensitas5 >= 160 && intensitas5 <= 175)
            {
                intensitas5 = 10;
            }
            if (intensitas5 >= 176 && intensitas5 <= 191)
            {
                intensitas5 = 11;
            }
            if (intensitas5 >= 192 && intensitas5 <= 207)
            {
                intensitas5 = 12;
            }
            if (intensitas5 >= 208 && intensitas5 <= 223)
            {
                intensitas5 = 13;
            }
            if (intensitas5 >= 224 && intensitas5 <= 239)
            {
                intensitas5 = 14;
            }
            if (intensitas5 >= 240 && intensitas5 <= 255)
            {
                intensitas5 = 15;
            }

            matrikImage[y, x] = intensitas5;
        }
        //Console.WriteLine("matrikImage[" + y + "," + x + "] : " + matrikImage[y, x]);
    }
}

int c1 = 0, c2 = 0, c3 = 0, c4 = 0;
int jumlah;
for (int y = 0; y < height; y++)
```

```
{
    for (int x = 0; x < width - 1; x++)
    {
        h = bmp6.GetPixel(x, y);
        if (matrikImage[y, x] != 0 && matrikImage[y, x + 1] != 0)
            if (h.R == 255)
            {
                c1 = matrikImage[y, x];
                c2 = matrikImage[y, x + 1];
                jumlah = 0;
                if (matrikCooccurrence[c1, c2] == 0)
                {
                    for (int z = 0; z < height; z++)
                    {
                        for (int z1 = 0; z1 < width - 2; z1++)
                        {
                            c3 = matrikImage[z, z1 + 1];
                            c4 = matrikImage[z, z1 + 2];

                            if (c3 == c1 && c4 == c2)
                            {
                                jumlah = jumlah + 1;
                                matrikCooccurrence[c1, c2] = jumlah;
                            }
                        }
                    }
                }
            }
    }
}

#endregion

/*for (int y = 0; y < 16; y++)
{
    for (int x = 0; x < 16; x++)
    {
        Console.WriteLine(y + "," + x + " : " + matrikCooccurrence[y, x]);
    }
}*/

double jumlahIntensitasDalamOccurrence = 0;
for (int y = 0; y < 16; y++)
{
    for (int x = 0; x < 16; x++)
    {
        jumlahIntensitasDalamOccurrence = jumlahIntensitasDalamOccurrence + matrikCooccurrence[y, x];
    }
}
```

```
Console.WriteLine("jumlahIntensitasDalamOccurrence : " + jumlahIntensitasDalamOccurrence);

//Mean Matrik Cooccurence Untuk ngitung fitur variance
double MeanMatrikCo = 0;
MeanMatrikCo = jumlahIntensitasDalamOccurrence / (16 * 16);
Console.WriteLine("Mean Matrik Co Occurence : " + MeanMatrikCo);

double jumlahnormalisasi = 0;
for (int y = 0; y < 16; y++)
{
    for (int x = 0; x < 16; x++)
    {
        //Console.WriteLine(y + "," + x + " : " + matrikCooccurrence[y, x] / jumlahIntensitasDalamOccurrence);
        jumlahnormalisasi = jumlahnormalisasi + (matrikCooccurrence[y, x] / jumlahIntensitasDalamOccurrence);
    }
}
//jumlah normalisasi pasti 1
Console.WriteLine("jumlahnormalisasi :" + jumlahnormalisasi);

//ENERGI /ASM (ANGULAR SECOND MOMENT)
double energi1 = 0;
for (int y = 0; y < 16; y++)
{
    for (int x = 0; x < 16; x++)
    {
        energi1 = energi1 + Math.Pow((matrikCooccurrence[y, x] / jumlahIntensitasDalamOccurrence), 2);
    }
}
Console.WriteLine("energi 1 : " + energi1);

//ENTROPI
double entropi1 = 0;
for (int y = 0; y < 16; y++)
{
    for (int x = 0; x < 16; x++)
    {
        if (matrikCooccurrence[y, x] != 0)
        {
            entropi1 = (entropi1 + (matrikCooccurrence[y, x] / jumlahIntensitasDalamOccurrence) *
                Math.Log(matrikCooccurrence[y, x] / jumlahIntensitasDalamOccurrence));
        }
    }
}
Console.WriteLine("entropi 1 : " + -entropi1);

//INVERS DIFFERENCE MOMENT (IDM)
double IDM = 0;
for (int y = 0; y < 16; y++)
{
```

```
    for (int x = 0; x < 16; x++)
    {
        if (matrikCooccurrence[y, x] != 0)
        {
            IDM = (IDM + (1 / (1 + Math.Pow(x, 2))) * (matrikCooccurrence[y, x] / jumlahIntensitasDalamOccurrence));
        }
    }
}
Console.WriteLine("IDM : " + IDM);

//SUM OF SQUARES, VARIANCE
double variance = 0;
for (int y = 0; y < 16; y++)
{
    for (int x = 0; x < 16; x++)
    {
        if (matrikCooccurrence[y, x] != 0)
        {
            variance = (variance + Math.Pow((x - MeanMatrikCo), 2) * (matrikCooccurrence[y, x] / jumlahIntensitasDalamOccurrence));
        }
    }
}
Console.WriteLine("variance : " + variance);

//DISSIMILARITY
double Dissimilarity = 0;
for (int y = 0; y < 16; y++)
{
    for (int x = 0; x < 16; x++)
    {
        if (matrikCooccurrence[y, x] != 0)
        {
            Dissimilarity = (Dissimilarity + Math.Abs(x - y) * (matrikCooccurrence[y, x] / jumlahIntensitasDalamOccurrence));
        }
    }
}
Console.WriteLine("Dissimilarity : " + Dissimilarity);

//KONTRAS
double kontras = 0;
for (int y = 0; y < 16; y++)
{
    for (int x = 0; x < 16; x++)
    {
        if (matrikCooccurrence[y, x] != 0)
        {
            kontras = kontras + Math.Pow((y - x), 2) * (matrikCooccurrence[y, x] / jumlahIntensitasDalamOccurrence);
        }
    }
```

```
}
Console.WriteLine("kontras : " + kontras);

//HOMOGENITAS
double homogenitas = 0;
for (int y = 0; y < 16; y++)
{
    for (int x = 0; x < 16; x++)
    {
        if (matrikCooccurrence[y, x] != 0)
        {
            homogenitas = homogenitas + (matrikCooccurrence[y, x] / jumlahIntensitasDalamOccurrence) / (1 + Math.Abs(y - x));
        }
    }
}
Console.WriteLine("homogenitas : " + homogenitas);
```