Systemnahes Programmieren
Bild und Matrix Rotation

Bilder rotieren
Ein Bild mit Hilfe einer selbstprogrammierten Routine um 90° drehen.

```
private Bitmap rotateImage(Bitmap sourceImage)
{
    Bitmap destinationImage = new Bitmap(sourceImage.Width, sourceImage.Width);
    for (int y = 0; y < sourceImage.Height; ++y)
    {
        for (int x = 0; x < sourceImage.Width; ++x)
        {
            destinationImage.SetPixel(sourceImage.Width - 1 - y, x,
            sourceImage.GetPixel(x, y));
        }
    }
    return destinationImage;
}
```

Bilder gradweise rotieren
Ein Bild mit Hilfe von .Net Methoden um einzelne Winkelgrade drehen.

```
private Bitmap rotateImage(Bitmap b, float angle)
{
    //create a new empty bitmap to hold rotated image
    Bitmap returnBitmap = new Bitmap(b.Width, b.Height);
    //make a graphics object from the empty bitmap
    Graphics g = Graphics.FromImage(returnBitmap);
    //move rotation point to center of image
    g.TranslateTransform((float)b.Width / 2, (float)b.Height / 2);
    //rotate
    g.RotateTransform(angle);
    //move image back
    g.TranslateTransform(-(float)b.Width / 2, -(float)b.Height / 2);
    //draw passed in image onto graphics object
    g.DrawImage(b, new Point(0, 0));
    return returnBitmap;
}
```

Matrix rotieren
Eine generische Matrix um 45° drehen.

```
public static T[][] RotateMatrix<T>(T[,] arr)
{
    if (arr == null)
        throw new ArgumentNullException("arr");
    if (arr.GetLength(0) < 1 || arr.GetLength(1) < 1)
        throw new InvalidOperationException("Matrix can not be rotated because one
dimension is 0");
    if (arr.GetLowerBound(0) != 0 || arr.GetLowerBound(1) != 0)
        throw new NotSupportedException("Arrays which do not start at index 0 are not
supported");
    int smallSide = Math.Min(arr.GetLength(0), arr.GetLength(1));
    int diff = Math.Abs(arr.GetLength(0) - arr.GetLength(1));
    //Create array
    T[][] rotated = new T[smallSide * 2 + diff - 1][];
    int n = 0;
    for (int i = 0; i < smallSide; i++)
```

```
    {
        rotated[n++] = new T[i + 1];
    }
    for (int i = 0; i < diff; i++)
    {
        rotated[n++] = new T[smallSide];
    }
    for (int i = smallSide - 1; i > 0; i--)
    {
        rotated[n++] = new T[i];
    }
    //Fill array
    for (int x = 0; x < arr.GetLength(0); x++)
    {
        for (int y = 0; y < arr.GetLength(1); y++)
        {
            int row = x + (arr.GetLength(1) - y - 1);
            int col = Math.Min(x, y);
            // Opposite direction
            //int row = x + y;
            //int col = Math.Min(x, (arr.GetLength(1) - y - 1));
            rotated[row][col] = arr[x, y];
        }
    }
    return rotated;
}
```