В этой статье мы рассмотрим сглаживание растрового изображения, как это делается в различных графических программах. Код не оптимизирован, работает медленно, но функционален.

Код выглядит следующим образом: рассчитывается значение каждого пикселя по сравнению с окружающими пикселями для нового значения цвета. Для достижения большего сглаживания, мы должны добавить только те пиксели, которые нужно включить в процесс.

```
var x, y: integer;  r, b, g: byte;  begin  with Image3.picture.Bitmap.canvas do  begin
 for x:=1 to Image3.picture.Bitmap.Width-1 do  for Y:=1 to Image3.picture.Bitmap.height-1
do  begin  r:=(GetRValue(Pixels[x-1,y-1])+  GetRValue(Pixels[x,y-1])+
GetRValue(Pixels[x+1,y-1])+  GetRValue(Pixels[x-1,y])+
GetRValue(Pixels[x+1,y])+  GetRValue(Pixels[x-1,y+1])+
GetRValue(Pixels[x,y+1])+  GetRValue(Pixels[x+1,y+1])+  GetRValue(Pixels[x,y]))
div 9;  g:=(GetGValue(Pixels[x-1,y-1])+  GetGValue(Pixels[x,y-1])+
GetGValue(Pixels[x+1,y-1])+  GetGValue(Pixels[x-1,y])+
GetGValue(Pixels[x+1,y])+  GetGValue(Pixels[x-1,y+1])+
GetGValue(Pixels[x,y+1])+  GetGValue(Pixels[x+1,y+1])+
GetGValue(Pixels[x,y])) div 9;  b:=(GetBValue(Pixels[x-1,y-1])+
GetBValue(Pixels[x,y-1])+  GetBValue(Pixels[x+1,y-1])+
GetBValue(Pixels[x-1,y])+  GetBValue(Pixels[x+1,y])+
GetBValue(Pixels[x-1,y+1])+  GetBValue(Pixels[x,y+1])+
GetBValue(Pixels[x+1,y+1])+  GetBValue(Pixels[x,y])) div 9;
Pixels[x,y]:=RGB(r,g,b);  end;  end;  end;
```

Следующий быстрый метод, с помощью **ScanLine**:

```
 type  TRGBTripleArray = array[0..32768] of TRGBTriple;  // 32768 = максимальное
количество пикселей по ширине(ScanLine)  pRGBTripleArray = ^TRGBTripleArray; //
указатель на TRGBTripleArray  ...  procedure Antialiasing(const DC: TCanvas; const
Rectangle: TRect);  var  cx, cy: Smallint;  r, g, b: Byte;  Row1: pRGBTripleArray;  Row2:
pRGBTripleArray;  Row3: pRGBTripleArray;  TEMP: TBitmap;  CurRect: TRect;  begin
TEMP := TBitmap.Create;  try  with TEMP do begin  Width := Rectangle.Right -
Rectangle.Left;  Height := Rectangle.Bottom - Rectangle.Top;  CurRect := Rect(0, 0,
Width, Height);  PixelFormat := pf24Bit;  Canvas.CopyRect(CurRect, DC, Rectangle);
  with Canvas do begin  for cy := 1 to (Height - 2) do begin  Row1 := ScanLine[cy
- 1];  Row2 := ScanLine[cy];  Row3 := ScanLine[cy + 1];  for cx := 1 to
(Width - 2) do begin  r := (Row1[cx - 1].rgbtRed+Row1[cx].rgbtRed+  Row1[cx +
1].rgbtRed+  Row2[cx - 1].rgbtRed+  Row2[cx + 1].rgbtRed+  Row2[cx -
1].rgbtRed+  Row3[cx].rgbtRed+  Row3[cx + 1].rgbtRed+
Row3[cx].rgbtRed) div 9;  g := (Row1[cx - 1].rgbtGreen+
```

```
Row1[cx].rgbtGreen+ Row1[cx + 1].rgbtGreen+ Row2[cx - 1].rgbtGreen+
    Row2[cx + 1].rgbtGreen+ Row2[cx - 1].rgbtGreen+ Row3[cx].rgbtGreen+
        Row3[cx + 1].rgbtGreen+ Row3[cx].rgbtGreen) div 9; b := (Row1[cx -
1].rgbtBlue+ Row1[cx].rgbtBlue+ Row1[cx + 1].rgbtBlue+ Row2[cx -
1].rgbtBlue+ Row2[cx + 1].rgbtBlue+ Row2[cx - 1].rgbtBlue+
Row3[cx].rgbtBlue+ Row3[cx + 1].rgbtBlue+ Row3[cx].rgbtBlue) div 9;
Row2[cx].rgbtBlue := b; Row2[cx].rgbtGreen := g; Row2[cx].rgbtRed := r;
  end; end; end; DC.CopyRect(Rectangle, Canvas, CurRect); end; finally
    TEMP.Free; end; end;
```