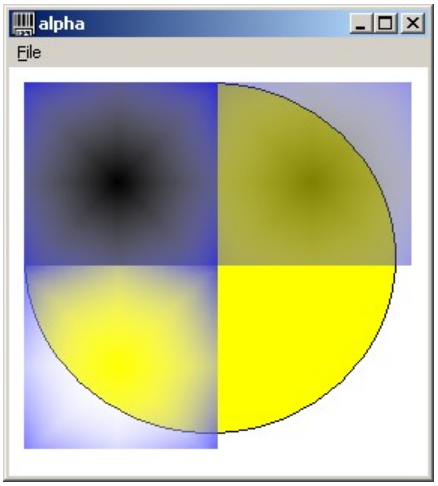
AlphaBlend with Per-pixel Alpha Channel

Version 1.00: Oct 21, 2000. 1.01: Feb 19, 2001 (add sample project link)

Problem

How to use AlphaBlend, with constant alpha, with per-pixel alpha channel, and with external 32-bpp bitmaps.



Calling AlphaBlend with 32-bpp DIB section

Design

As no image processing package is known to be able to generate 32-bpp BMP file, which is the native bitmap format supported by GDI, TGA, the simplest bitmap format supporting alpha channel is used. For example Jasc Paint Shop Pro can generate 32-bpp uncomressed TGA files with a single 8-bit alpha channel.

Using Paint Shop Pro to Edit Bitmap with Alpha Channel (mask)

TGA files can be easily loaded into a Window program, to create a 32-bpp DIB section, which can be used by AlphaBlend. The alpha channel can be pre-multipled to RGB channels to generated pre-multiplied 32-bpp bitmap needed by AlphaBlend to enable per-pixel alpha.

32-bpp uncompressed TGA file generated by Paint Shop Pro

Uncompressed TGA file is extremely simple. For example, 32-bpp uncompressed TGA file generated by Paint Shop Pro consists of 18-byte header, image pixel array, and ignorable footer with creator information.

The header can be defined as:

```
#pragma pack(push, 1)

typedef struct
{
    BYTE    IDLength;        // 0
    BYTE    ColorMapType;    // 0
    BYTE    ImageType;       // 2: Truecolor image data
    WORD    CMapStart;       // 0
    WORD    CMapLength;      // 0
    BYTE    CMapDepth;       // 0
    WORD    XOffset;         // 0
    WORD    YOffset;         // 0
    WORD    Width;           // width
    WORD    Height;          // height
    BYTE    PixelDepth;      // 32 for 32-bpp image
    BYTE    ImageDescriptor; // 8 for 8-bit alpha
}   TGA_Header;

#pragma pack(pop)
```

Loading 32-bpp uncompressed TGA file, including pre-multiplying alpha

```
HBITMAP Load32bppTga(const TCHAR * pFileName, bool bPreMultiply)
{
    HANDLE handle = CreateFile(pFileName, GENERIC_READ, FILE_SHARE_READ,
                        NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);

    if ( handle == INVALID_HANDLE_VALUE )
        return NULL;

    TGA_Header header;

    DWORD dwRead = 0;
    ReadFile(handle, & header, sizeof(header), & dwRead, NULL);

    if ( (header.IDLength!=0) || (header.ColorMapType!=0) || (header.ImageType!=2) ||
         (header.PixelDepth!=32) || (header.ImageDescriptor!=8) )
    {
        CloseHandle(handle);
        return NULL;
    }

    BITMAPINFO bmp = { { sizeof(BITMAPINFOHEADER), header.Width, header.Height, 1, 32 } };

    void * pBits = NULL;

    HBITMAP hBmp = CreateDIBSection(NULL, & bmp, DIB_RGB_COLORS, & pBits, NULL, NULL);

    if ( hBmp==NULL )
    {
        CloseHandle(handle);
        return NULL;
    }

    ReadFile(handle, pBits, header.Width * header.Height * 4, & dwRead, NULL);

    CloseHandle(handle);

    if ( bPreMultiply )
    {
        for (int y=0; y<header.Height; y++)
        {
            BYTE * pPixel = (BYTE *) pBits + header.Width * 4 * y;

            for (int x=0; x<header.Width; x++)
            {
                pPixel[0] = pPixel[0] * pPixel[3] / 255;
                pPixel[1] = pPixel[1] * pPixel[3] / 255;
                pPixel[2] = pPixel[2] * pPixel[3] / 255;

                pPixel += 4;
            }
```

```
        }
    }

    return hBmp;
}
```

Sample Drawing Program

```
void AlphaDraw(HDC hDC, int x, int y, int width, int height, HBITMAP hBmp)
{
    HDC     hMemDC = CreateCompatibleDC(hDC);
    HGDIOBJ hOld   = SelectObject(hMemDC, hBmp);


    HBRUSH hBrush = CreateSolidBrush(RGB(0xFF, 0xFF, 0));
    SelectObject(hDC, hBrush);
    Ellipse(hDC, x, y, width*2, height * 2);                    // a yellow circle in the background
    SelectObject(hDC, GetStockObject(WHITE_BRUSH));
    DeleteObject(hBrush);

    BitBlt(hDC, x, y, width, height, hMemDC, 0, 0, SRCCOPY);  // display the bitmap

    BLENDFUNCTION pixelblend = { AC_SRC_OVER, 0, 255, AC_SRC_ALPHA };

    AlphaBlend(hDC, x, y+height, width, height, hMemDC, 0, 0, width, height, pixelblend); // blend with per-pixel alpha

    BLENDFUNCTION blend50 = { AC_SRC_OVER, 0, 128, 0 };

    AlphaBlend(hDC, x+width, y, width, height, hMemDC, 0, 0, width, height, blend50); // 50% blending

    SelectObject(hMemDC, hOld);
    DeleteObject(hMemDC);
}
```

Sample Project

alphablend.zip