Active Palett e Suppo rt in Indeo ™ Video R3.2

Indeo™ Video Engineering
Intel Corporation

September 1994

E

Indeo™ Video Tips and Tricks

Additional copies of this document or other Intel literature may be obtained from:

Intel Corporation
Literature Sales
P.O. Box 7641
Mt. Prospect, IL 60056-7641

or call 1-800-879-4683

Order Number: 242309-001

Active Palette Support in Indeo™ Video R3.2

Introduction

Recently, users of Intel's Indeo™ video technology have raised questions regarding support for Microsoft's Video for Windows* active palette interface: What is the active palette interface? Does Indeo video support it? How does one activate it? This document addresses these questions.

What Is a Palette?

Video displays are often limited in the number of colors they can display at one time. A number of video display adapters can support millions of colors, yet at any given moment, can display only a small subset of those colors. For example, several adapters support 16 million colors and yet can only display 256 at a time in certain display modes. This subset of the possible colors is called a *palette* or *Color LookUp Table* (CLUT). Each palette entry contains a red, green and blue color component. To produce a certain color on the screen, the index of that color is written to video memory. This is illustrated in the diagram below.

For each pixel, the application writes a CLUT index (instead of an RGB value) into video memory. The video hardware then gets the RGB value from the CLUT through the index held in video memory.

Limitations of a Fixed Palette

Many video environments support running multiple applications simultaneously. However, if the video device can only display a single 256 color palette at any given time, some means is needed for determining *which* application's palette will be used, and when. For example, should the video environment decide which palette to use and then force all applications to use that palette?

Video Memory

34	255	202	9
12	26	34	49
2	4	6	8
8	67	53	9

Palette (RGB Values)

(24,24,24)	(25,24,24)	(26,24,24)	(27,24,24)	(28,24,24)	(29,24,24)	(30,24,24)	(31,24,34)
(24,25,24)	(25,25,24)	(26,25,25)	(27,26,25)	(38,24,24)	(39,24,24)	(30,34,24)	(41,34,34)
(26,25,24)	(27,25,24)	(26,25,27)	(27,26,26)	(38,24,25)	(39,24,26)	(40,34,24)	(41,34,35)
(26,26,24)	(27,26,24)	(27,26,27)	(27,26,27)	(38,24,26)	(39,24,27)	(40,35,24)	(41,36,36)
(26,27,24)	(28,27,24)	(28,27,28)	(28,27,28)	(38,24,27)	(39,24,28)	(40,36,24)	(41,36,36)
(26,28,24)	(28,28,24)	(29,30,30)	(28,27,29)	(38,24,28)	(39,24,29)	(40,37,24)	(41,38,37)
(26,29,24)	(28,29,24)	(29,32,32)	(29,30,32)	(38,24,29)	(39,24,30)	(40,38,24)	(41,38,38)
(26,30,24)	(28,30,24)	(29,33,33)	(30,33,34)	(38,24,30)	(39,24,31)	(40,39,24)	(41,39,39)

Video compressor-decompressors (or *codecs*) typically have their own unchanging (or *fixed*) palettes, which have been optimized for playback image quality and in which the order of the colors have been arranged for fast display. However, if the foreground palette changes, several undesirable effects can occur:

1. There may be a "flash" of false color information as the new palette is asserted.
2. The video must be rendered using the new palette. In the Video for Windows environment, this is done by finding a "best match" for each entry in the video palette to an entry in the new palette, then changing the palette indices accordingly before drawing each video frame. This, in turn, produces three undesirable side-effects:
    3. A performance penalty for the "best match" operation.
    4. The loss of direct frame buffer access.
    5. A degradation in the video color quality.

What Is *Active Palette Support*?

Active palette support is designed to overcome these problems. With active palette support, the application supplies the desired palette to the codec (which could be the new foreground palette), and the codec renders imagery in this palette. If the supplied palette is the same as the asserted palette, palette "flash" does not occur. Further, this makes the "best match" operation unnecessary, averting the associated performance and quality losses.

Active palette support does have its own costs:

6. When an application supplies a new palette to the codec, some initialization must occur, which may freeze the video for a moment or two.
7. Rendering to an arbitrary palette will likely be slower than rendering to the codec's default palette.
8. If the palette supplied does not have a proper mix of colors to render the video being played, the video will not look good.

Enabling Active Palette Support

Understanding what active palette support is and why it is useful, you are ready to learn how an application enables active palette support. This discussion presents the simplest method: using the `mciSendString` functionality. To enable active palette support in this way, the code must implement the following steps:

9. Create a logical palette.
10. Create a palette object from the logical palette and get a handle to the palette.
11. Use the `mciSendString` command to set the video palette for the codec.

Code Sample

The following code excerpt was added to an existing application (in this case `movplay2.c` in `VFWDK\SAMPLES\MOVPLAY`) in order to enable active palette support. (This excerpt contains code related to active palette activation only.)

```
/* hpal is the palette that we will assert.
/*It is initialized to the current palette in initPalette.
/* It is asserted later on in the call to mciSendString.  */

HPALETTE hpal;

/*-------------------------------------------------------------------------+
| initPalette - creates the desired palette for the movie     |
+-------------------------------------------------------------------------*/
void initPalette()
{
       LOGPALETTE *plgpl;
```

```
        int i;
        HANDLE hDC;

        /* Allocate the logical palette.  */
        plgpl = (LOGPALETTE*) LocalAlloc(LPTR,
        sizeof(LOGPALETTE) + 256 * sizeof(PALETTEENTRY));
        plgpl->palVersion = 0x300;

        /* Fill in the desired palette for use by the video.
           We use the current system palette entries in this case,
           but any arbitrary palette could be created here.  */
        hDC = GetDC(NULL);
        GetSystemPaletteEntries(hDC, 10, 236, (PALETTEENTRY FAR *)plgpl->palPalEntry);
        ReleaseDC(NULL, hDC);

        plgpl->palNumEntries = 236;
        for (i = 0; i < 236; ++i)
                plgpl->palPalEntry[i].peFlags = PC_NOCOLLAPSE;

        /* Now create the palette object and get a handle to it using CreatePalette.  */
        hpal = CreatePalette(plgpl);
        LocalFree((HLOCAL) plgpl);
}

main ( )
{
        ...
        initPalette(); /* Probably done in application initialization usually.  */
        ...
        openMovie ();
        playMovie ();
        ...
}

openMovie()
{
        /* other tasks related to opening a movie :          */
        /* open the file                                 */
        /* get the window handle                         */
        /* position the movie in the center of the screen    */

        /* Assert our desired palette - This is the command to enable
        /* active palette support.  */

         wsprintf(achCommand, "setvideo mov palette handle to %d", hpal);
         if (mciSendString((LPSTR)achCommand, NULL, 0, NULL) != 0){
                MessageBox(hWnd, "Unable to set palette handle", NULL,
                        MB_ICONEXCLAMATION|MB_OK);
        ...
        /* Now update the title and menu bars,              */
        /* Update the window (ie.  call InvalidateRect) */

}
```

Testing Active Palette Support

Having sent the appropriate messages to the codec, you will want to assure yourself active palette support is really enabled. Following is an example of a simple test:

12. Have the application (built for debug mode) create an extremely abnormal palette. For example, the palette might contain only shades of gray, or all black.

13. Have the codec assert this palette by performing the steps described in the previous section, and attempt to decompress a video file. If the image is rendered using the abnormal palette, you have successfully enabled active palette support in the Indeo video codec.

Note that the MCIAVI layer will still do best-matching to the palette you asserted, and that the 20 system colors will still be present. Thus, you might not see a completely gray video, even if you have asserted a gray palette. However, with an asserted palette that contains only the system colors and gray, the video playback should be noticeably worse than without active palette support. You can try this approach with an abnormal palette to ensure that active palette is enabled, and then assert a "normal" palette when you are satisfied with your palette assertion code.

Versions

Active palette support is a relatively new feature in both the Microsoft MCIAVI interface and in the Indeo video codec. It is supported in:

14. Version 1.10.0.187 and later of the Microsoft Video for Windows runtime package, msvideo.dll
15. Version R3.2.01.30 and later of the Indeo video codec

For More Information

For more information on the CreatePalette() call, consult the Microsoft Visual C/C++ V1.5 documentation. For more information on the mciSendString read the on-line documentation for the Microsoft Video for Windows Development Kit, Version 1.1. Look in the MCIAVI Command Strings section for the setvideo command. Note that the setvideo command as documented in Video for Windows Development Kit does not discuss the ability to set the palette handle of a movie.

If you have additional questions, please call the Indeo Technology Support hotline at 800-628-8686. (Outside the U.S., call +916-356-3551.