DESCRIPTION OF CINE FILE FORMAT (*.CIN, *.CCI, *.CHD)

Vision Research Inc.
190 Parish Drive
Wayne, New Jersey 07470
USA

Initial design: November, 1992
Revised November 1, 1997
Revised October 29, 1999
Revised October 3, 2000
Revised December 15, 2000
Revised April 22, 2003
Revised August 28, 2003

"The information contained in this document file includes data that is proprietary to Vision Research, Inc. and shall not be duplicated, used, or disclosed - in whole or in part - for any purpose other than to create file conversion routines from Vision's proprietary "CIN" format to other formats which may be in use by the end user. This restriction does not limit your right to use information contained in this data if it is obtained from another source without restriction. The data subject to this restriction are contained in all pages of this file"

The general structure of the cine file includes:

- the cine file header
- the image header (Windows BITMAPINFO structure)
- the camera setup information structure
- optional: tagged blocks with acquired signals, IRIG time for every image, etc.
- the table with the image positions in file
- a number of image objects (annotations and pixel array)

This description uses the usual notations:

```
BYTE         = 8  bits, unsigned
char         = 8  bits, signed
WORD         = 16 bits, unsigned
INT16, short = 16 bits, signed
BOOL         = 32 bits, logic value(TRUE=1, FALSE=0)
DWORD, UINT  = 32 bits, unsigned
long, int    = 32 bits, signed
```

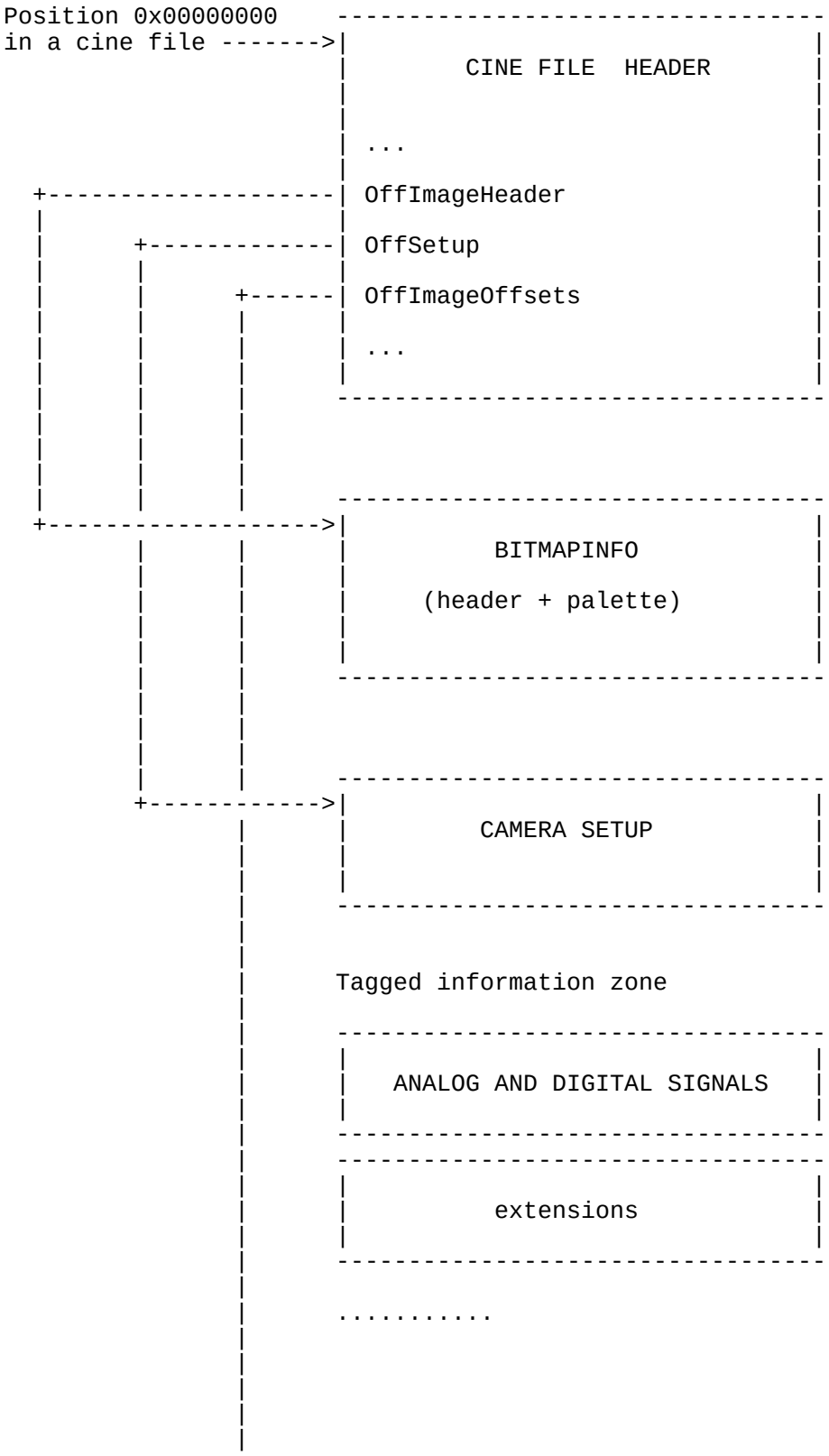
```
//a compact structure for time on 64 bits (32.32 seconds)
typedef struct tagTIME64
    {
    DWORD fractions;          //fractions of seconds  (resolution
                              //1/4Gig i.e. cca.  1/4 ns)
                              //The fractions of the second are stored
                              //here multiplied by 2**32
    time_t seconds;           //seconds from Jan 1 1970
                              //(max year: 2036 signed ; 2102 unsigned)
    }TIME64;
```

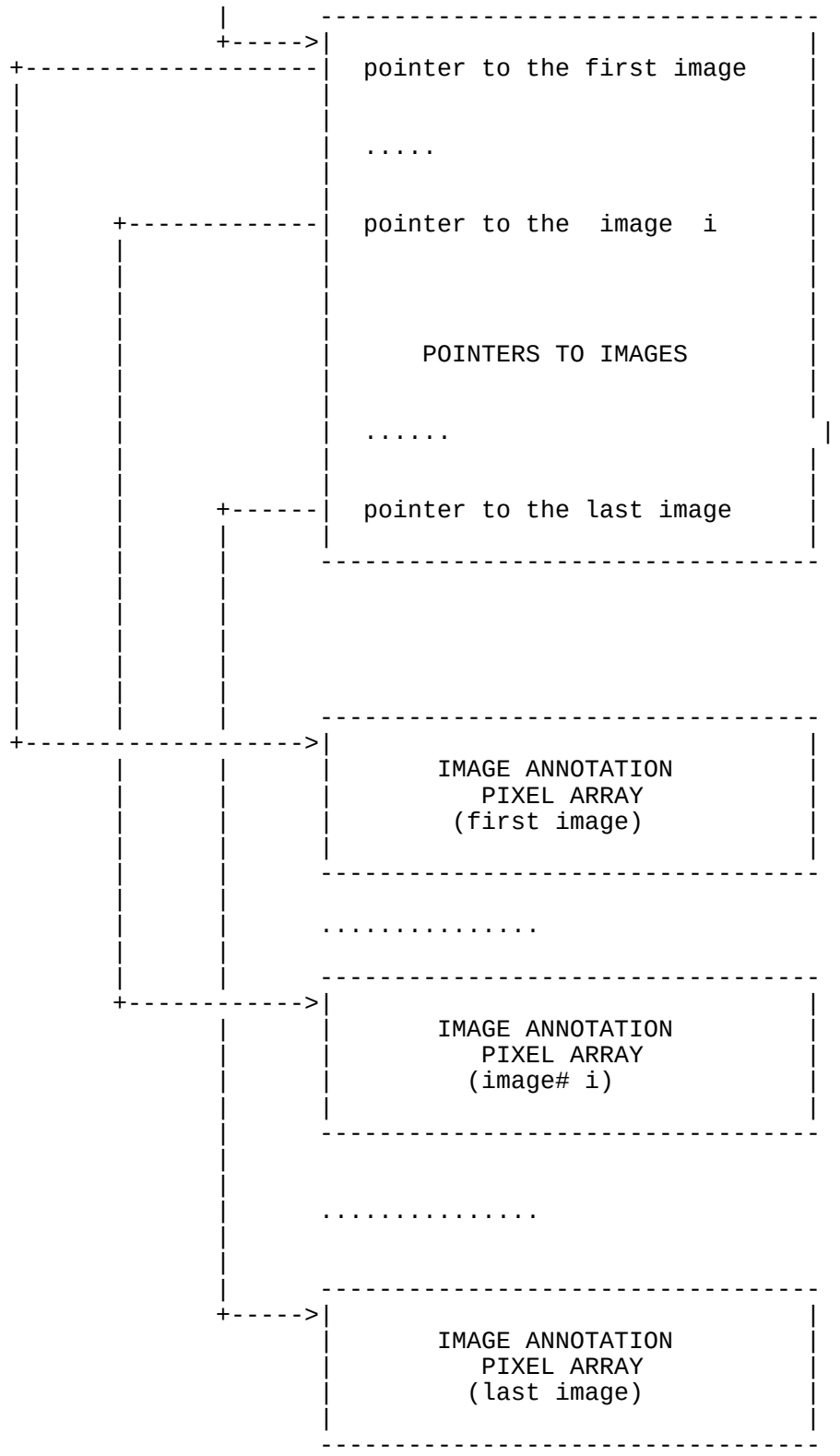
Byte order is with the least significant byte first (Intel)

Image numbering:

The images are numbered in a growing order using 32 bits signed numbers. The images before trigger are negative. The images after the trigger are 0 or positive.

The cine file contains fixed structures and a few optional tagged blocks.




pointer to the first image
.....
pointer to the image i
POINTERS TO IMAGES
.....
pointer to the last image
IMAGE ANNOTATION
PIXEL ARRAY
(first image)
.....
IMAGE ANNOTATION
PIXEL ARRAY
(image# i)
.....
IMAGE ANNOTATION
PIXEL ARRAY
(last image)

Detailed structure of each block:

Cine File Header (the CINEFILEHEADER structure):

All offsets (pointers, addresses) in file are related to the file begin

```
WORD   Type;            //Marker, must be "CI"
WORD   HeaderSize;      //Header size in bytes
WORD   Compression;     //CC_RGB=0 - uncompressed BMP
                        //CC_JPEG=1 - JPEG Compressed
                        //CC_UNINT=2 - Uninterpolated color image for
                        //             the gbrg/rggb CFA
WORD   Version;         //upgrades, now 1, supports files > 4GB
long   FirstMovieImage; //First recorded image number, relative to
                        //trigger
DWORD  TotalImageCount; //Total count of recorded movie images
long   FirstImageNo;    //First image saved to this file
                        //(relative to trigger)
DWORD  ImageCount;      //count of images saved to this file
DWORD  OffImageHeader;  //offset in the file of the
                        //BITMAPINFO structure for all images
DWORD  OffSetup;        //offset in file of the
                        //SETUP structure
DWORD  OffImageOffsets; //offset in file of an array with position
                        //of each annotated image in file

TIME64 TriggerTime;     //Trigger time 32.32 in second and fraction of
                        //second from Jan 1 1970 (resolution: cca 1/4
                        //nanosecond)
```

Windows structure for image header (BITMAPINFO):

```
DWORD   biSize;         //header size (without palette)
LONG    biWidth;        //image width (pixels)
LONG    biHeight;       //image height (pixels)
WORD    biPlanes;       //plane of colors
WORD    biBitCount;     //bits per pixel
DWORD   biCompression;  //=0 means no compression
                        //not used when the file is JPEG compressed
DWORD   biSizeImage;    //the size in bytes of the image
LONG    biXPelsPerMeter;//horizontal resolution in pixels per meter
LONG    biYPelsPerMeter;//vertical resolution in pixels per meter
DWORD   biClrUsed;      //the number of color indexes in the
                        //actually used by the bitmap
DWORD   biClrImportant; //the number of color indexes in the color
                        //table considered important
```

Note: the palette is no more saved in the cine file.

Camera setup information (the SETUP structure):

```
WORD FrameRate16;       //frame rate in pictures per second
WORD Shutter16;         //shutter duration in microseconds
WORD PostTrigger16;     //the count of the post trigger frames
WORD FrameDelay16;      //frame delay in microseconds (Synch Frame
                        //mode)
WORD AspectRatio;       //aspect ratio (width/height)
WORD Contrast;          //the position of the contrast adjustment
WORD Bright;            //the position of the brightness adjustment
BYTE Rotate;            //enable the image rotation (90 degrees)
                        //(BOOL)
BYTE TimeAnnotation;    //source of time information
BYTE TrigCine;          //triggered cine  (BOOL)
BYTE TrigFrame;         //Synch imaging  (BOOL)
BYTE ShutterOn;         //enable the shutter (BOOL)
char Description[121];  //event description text

WORD Mark;              //will be  "ST"  -  maker for setup file
WORD Length;            //length of the current version of setup
WORD Binning;           //binning factor - reduce horizontal slice
                        //dimension
WORD BinEnable;         //enable the acqui of the binary sig from
                        //print port
INT16 BinChannels;      //number of multiplexed bytes read from
                        //paralel port
                        //multiple of 8
BYTE BinSamples;        //number of samples acquired per image; now:1
char BinName[8][11];    //8 binary signals names with max 10
                        //chars/name ended each by a NULL byte

WORD AnaEnable;         //enable the acqui of the analog signals
INT16 AnaChannels;      //number of analog channels used
BYTE AnaSamples;        //number of samples acquired per image; now: 1
BYTE AnaBoard;          //board type 0=none, 1=dsk (DSP system kit)
                        //2 = DSP+8 channels ADC (12 bit)
INT16 AnaOffset[8];     //electronic offset correction, per channel
float AnaGain[8];       //electronic gain correction, and conversion
                        //to real units, per channel
char AnaUnit[8][6];     //8 analog signals unit strings with max 5
                        //chars/name ended each by a NULL byte
char AnaName[8][11];    //8 analog signals names with max 10
                        //chars/name ended each by a NULL byte

long lFirstImage;       //range of images for continuous recording
DWORD dwImageCount;
INT16 nQFactor;         //Quality  -  for continuous recording; range
                        //2...255
WORD wCineFileType;     //Cine file type  -  for continuous recording
char szCinePath[4][65]; //4 paths to save cine files - for continuous
                        //recording
WORD bMainsFreq;        //TRUE = 60Hz USA,  FALSE = 50Hz Europe,
                        //for signal view in DSP

//Time board
BYTE bTimeCode;         //Time code (IRIG-B, NASA36, IRIG-A ...
BYTE bPriority;         //Time code has priority over PPS
WORD wLeapSecDY;        //Next day of year with leap second
double dDelayTC;        //Propagation delay for time code
double dDelayPPS;       //Propagation delay for PPS
```

```
//General use bits
//Bit 0 = Flip vertical (only for v3, not used in v4)
//      invert image upside down  -  used to
//      allow the invert of the 256x256. For example
//      channels 8...11 may be directed to the FBM
//      memory SIMMs 0...3 but the resulting image
//      will be upside-down. The effect is inverting
//      bit 3 of the slice address. 0..7 <=> 8..15
//Bit 1 = Flip horizontal (only for v3, not used in v4)
//      mirror the image, left-right
//      added for the color camera where certain components
//      are mirrored
//      Effect: mirror every row during transfer from FBM to
//      memory buffer.
//Bit 2 = Separate channels v4 . Move the pixels to produce
//      rectangular  areas coming out from the same video channel
//      The video channels are interlaced based on a 8x2 kernel
//      in the quarter (512x512) camera
//

WORD GenBits;

//color adjustment:
short ContrastR;        //values for adjustment in analog
short BrightR;          //part of the RGB color system; reused for
                        //digital corrections on V4
short ContrastG;
short BrightG;
short ContrastB;
short BrightB;

WORD ImWidth;           //image dimensions in v4
WORD ImHeight;

WORD EDRShutter16;      //extended dynamic range exposure (v4)
UINT Serial;            //camera serial number -  will be stored in
                        //every cine file

short Saturation;       //Color saturation [-100, 100]
BYTE Reserved[3];       //align to dword
BOOL AutoExp;           //autoexposure
BOOL bFlipH;            //Flip horizontal, vertical in v4
BOOL bFlipV;            //For color images flips are postponed after
                        //interpolation
BOOL bCrossHair;        //display a crosshair in setup

//upgrade from 16 to 32 bits of a few old variables (July2000)
UINT FrameRate;
UINT Shutter;
UINT EDRShutter;
UINT PostTrigger;
UINT FrameDelay;

BOOL bEnableColor;      //available to user: when 0 force gray images

UINT CameraVersion;     //4, 5 ....
UINT FirmwareVersion;   //Firmware version
UINT SoftwareVersion;   //Phantom version
int RecordingTimeZone;  //the time zone active during the recording of
                        //the cine
```

```
UINT CFA;               //code for the color filter array (for late
                        //interpolate or uninterpolate):
                        //CFA_NONE=0,(gray) CFA_VRI=1(gbrg/rggb),
                        //CFA_VRIV6=2(bggr/grbg), CFA_BAYER=3(gb/rg)
                        //high byte carry info about color/gray heads at
                        //v6
                        //Masks: 0x80000000: TLgray 0x40000000: TRgray
                        //0x20000000: BLgray 0x10000000: BRgray
                        //Final adjustments after image processing:
int Bright;             //Brightness -100...100 neutral:0
int Contrast;           //Contrast -100...100 neutral:0
int Gamma;              //Gamma -100...100 neutral:0

UINT Reserved1;         //BOOL LockToIRIG;//camera mode : lock to IRIG –
                        //- removed

UINT AutoExpLevel;      //level for autoexposure control
UINT AutoExpSpeed;      //speed for autoexposure control
RECT AutoExpRect;       //rectangle for autoexposure control

WBGAIN WBGain[4];       //Gain adjust on R,B components, for white
                        //balance,
                        //1.0 = do nothing,
                        //index 0: all image for v4,5,7...TL head for v6
                        //index 1, 2, 3 :   TR, BL, BR for v6
int Rotate;             //0=do nothing +90=counterclockwise
                        //-90=clockwise

WBGAIN WBView;          //White balance to apply on images from cine
                        //file

UINT RealBPP;           //real number of camera bits per pixel
                        //e.g 8 on old cameras and 12 on v7 with 12 bit
                        //converters; pixel stored on 16 bit

UINT Conv8Min;          //Minimum value when convert to 8 bits
UINT Conv8Max;          //Max value when convert to 8 bits

int FilterCode;         //ImageProcessing: area processing code
int FilterParam;
IMFILTER UF;            //user filter, see PhInt.h

UINT BlackCalSVer;      //Black Calibration SoftwareVersion
UINT WhiteCalSVer;      //White Calibration SoftwareVersion
UINT GrayCalSVer;       //Gray Calibration SoftwareVersion
BOOL bStampTime;        //Stamp time (in continuous recording)
```

The tagged information blocks:

This field is present if
(OffSetup + sizeof(SETUP)) < OffImageOffsets

The size of the SETUP structure is the Length field.
The structure of the tagged blocks is:

```
DWORD BlockSize;
WORD  Type;
WORD  Reserved;
BYTE  Data[BlockSize-8];     //sizeof(DWORD) + 2*sizeof(WORD) = 8
```

Allocated tags:

Signals Type=1000 (0x3e8) The signals are stored for each recorded image.

IRIG time Type=1001 (0x3e9). Every element of the array is a TIME64 structure (32.32). The time is stored for each recorded image, the count of time items is TotalImageCount (even if you saved only a smaller range of images: ImageCount). If BlockSize is bigger than the size of this time array (Phantom version >=477) it contains also the exposure length for every image, stored as an array of DWORDs of fractions of second (similar to fractions field of the TIME64 structure).

The array of pointers to images:

```
DWORD pImage[ImageCount];   //the position in file of the every saved image
```

The image object:

The Annotation field

```
DWORD AnnotationSize;          //total length of the bloc
                               //(AnnotationLength  included)
BYTE Annotation[AnnotationSize - 8];
DWORD ImageSize;               //Pixel array size
```

Example of void annotation bytes: 08 00 00 00 00 00 04 00 (uncompressed 512x512 image size = 0x00040000 = 256 kB). The Annotation array
is absent; the AnnotationSize and the ImageSize are always present

Pixel array:

Uncompressed gray images contain the actual gray level as pixel value
Rows are padded to a multiple of 4 (32 bits)

```
BYTE pixels [biWidth * biHeight];//Uncompressed
```

If biBitCount = 24 (color image) the array is three times larger.

```
BYTE pixels [3 * biWidth * biHeight];
```

The order of components for the color pixels (interpolated or from a color
system with 3 cameras) is BGR.

Compressed images contain a complex data structure instead of pixel value. This structure is not described here.

The simple way to access the gray images in the current versions of Phantom camera cine files:

1. Check the file type (the "CI" marker)
2. Use OffImageHeader and biWidth, biHeight to get the image dimensions

3 Use OffImageOffsets to access the table with pointers to the images
4 The first pointer (DWORD) in the table correspond to the first image stored in this file (FirstImageNo). Select the pointer to the image you want, pImage[YourImage-FirstImageNo]
5 Access the image object and skip the annotation using their length stored in the first DWORD.
6 You are now at the beginning of the pixel array for the image i.

Repeat the steps 3-6 to access other images in the file

Revision notes:

1. 1992...2003
New fields were added to the SETUP structure but we have compatibility, both forward and backward between different version of Phantom. The Version field of the header remained 0.

2. November 1, 1997 (Phantom version 235):
This version include the first interface to an IRIG board. The TriggerTime and TriggerTimeExt from the CINEFILEHEADER were replaced by a TIME64 (presented above) structure. The cines recorded with Phantom < v235 have an inverse order of time components (first: seconds, 32 bits than fractions of second 32 bits) but the TriggerTimeExt was always 0 before v235. If you read TriggerTime.seconds == 0 from an old cine file you have to read TriggerTime.fractions as the number of seconds from Jan 1 1970.

IRIG time for every image of the cine is stored in a tagged block if the selected time annotation was an IRIG board.

3. 1997
Color cines were created with the same structure.
The BITMAPINFOHEADER field biBitCount = 24 bits per pixel and a pallete is not present.

4.November 18, 1998(Phantom version 301)
Phantom application upgraded to 32 bits. SETUP fields remains unchanged except a few data types: int become INT16, BOOL become WORD etc. Struct member alignment should be set to 1, at least for the SETUP structure.

4. February 3, 2000 (Phantom version 424):

For the color v4 camera a new format is available: uninterpolated color cine file. The Compression field in the CINEFILEHEADER is 2, biBitCount = 8, and the palette is absent. A dll library is available to interpolate the color.

5. July 3, 2000 (Phantom version 459)
A few of the 16 bits fields were upgraded to 32 bits. The old fields were renamed, getting a "16" termination (e.g. FrameRate16) and they still carry the information if possible (value < 65536). If the SETUP.Length is greater than FIELD_OFFSET(FrameRate) you can use the new fields. This means the cine was saved by a version of Phantom that wrote the new fields upgraded to 32 bits.

6. December 15, 2000 (Phantom version 477)
When save or convert to a set of image files or to avi format a cine header file having the extension .chd is automatically written. It contains an exact copy of the cine file header, bitmapinfoheader, setup and tagged blocks described above.

The time block contains exposure length information for each image. You can recognize whether the exposure information is present from the size of the time block.

Version 1 of CINE file format

7. April 22, 2003 (Phantom version 600)

Starting from version Ph600 Phantom software is able to write and read files bigger than 4 GigaBytes. The operating system has to support the files > 4GB - in the case of Windows the file system must be NTFS.

The main change is the enlargement of the pointers to images to 64 bits. All other file pointers remain 32 bit. This means **the array of the pointers to images** should be declared as:

```
__int64 pImage[ImageCount];   //the position in file of the every image.
```

The Version field in the CINEFILEHEADER is 1. (It was 0 before Ph600). When an old cine file having Version=0 is read, the image pointers have to be expanded to 64 bits.

The palette of the BITMAPINFO is not stored in the header of the cine v1 file. Only BITMAPINFOHEADER is stored even for gray images. The cine reader has to add itself a gray palette if needed.

In the cine v1 the information from tagged blocks is stored only for the range of images that are saved in the file.

New tags were added:

Time only block Type=1002 (0x3ea). Every element of the array is a TIME64 structure (32.32). The time is stored only for the images saved in this file; the count of time items is ImageCount (even if you recorded in the camera a larger range - TotalImageCount)

Exposure only block Type=1003 (0x3eb). Every element of the array is a DWORD that represents a fixed point 0.32 number. You have to divide it by 2**32 to get the real exposure in seconds. The exposures are stored only for the images saved in this file; the count of exposure items is ImageCount.

8. August 28 , 2003 (Phantom version 603)
Support for 16 bpp monochrome images and 48 bpp color images. The values 16 and 48 of the biBitCount field from image header describe these types of cine files. The real bit depth of the camera can be between 8 and 16 bits, e.g Phantom v7 has 12 bits. The pixel value is stored “as it is”; it is not left aligned to 16 bits. This means the pixels from a v7 camera configured to record on 16 bits are stored as 16 bit integers having values from 0 to 4095. The RealBPP field of the SETUP structure has to be used to find the real bit depth and the maximum value of the pixels. If the Length field of SETUP is smaller than the offset of this field the value of RealBPP should be considered 8. The byte order is little endian (Intel) and the color order is BGR.
The value biBitCount=16 has other meaning in Windows (color image 5:6:5) but this is not usually a problem since the bitmap has anyway to be converted to 8 or 24 bits before display. PhInt provide the support for color interpolation and image processing for all bit depths.

Please, send any question regarding the CINE file format to peter.pop@visionresearch.com

- EOD -