EPIX®

PXIPL™ LIBRARY

Image Processing and Analysis Library

$$\nabla^2 f\,(x,y) \equiv \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \qquad \frac{d_2\text{-}f}{f}$$

```
for(;;)  {
  capture();
  difference();
  lowpass();
  if (findblobs())
    reportlargest();
}
```

$$\begin{bmatrix} \cos\varnothing & \sin\varnothing & 0 \\ -\sin\varnothing & \cos\varnothing & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Pixel Clock:	45MHz
Pixel Jitter:	1.0 ns
Resolution:	6144 x 4096
Buffers:	253
Pixel Bits:	8
Camera:	Area Scan
Trigger:	TTL
Field Delay:	5

$$\frac{1}{d_1} + \frac{1}{d_2} = \frac{1}{f}$$

Blobs Found:	*259*
Center of Mass:	*(23.65, 78.78)*
Subpixel Edge @:	*(97.3498, 87.5000)*

$$\frac{2\lambda}{\pi n \omega_0}$$

$$\delta = 1.22\frac{\lambda d_i}{l} \qquad s.n.r. = \frac{\bar{x}}{\sqrt{\Sigma(x_i\text{-}\bar{x})^2/n}}$$

FEATURES

- **Image Processing and Analysis.**
- **Image Graphics and Printing.**
- **Image Morphology and Transforms.**
- **Subpixel Accuracy Measurements.**
- **Blob Analysis and Particle Tracking.**
- **Image Correlation.**
- **Image Load, Save, and Print.**
- **For use with XCLIB, SVOBJ, or 4MOBJ.**
- **C/C++ Library for 16 & 32 Bit DOS Programs.**
- **DLL for 16, 32, 64 bit Windows Applications.**
- **C/C++ Library for 32 bit & 64 bit Linux Programs.**

PROCESSING POWER

The PXIPL Library empowers C/C++ and Windows programmers to process and analyze images in conjunction with:

- The PIXCI® imaging boards and XCLIB Library,
- The EPIX® SILICON VIDEO® cameras, their PIXCI® imaging board, and XCLIB Library,
- The 4MEG VIDEO™ imaging boards and 4MOBJ Library, or
- The SV-MUX™ imaging boards and SVOBJ Library.

The PXIPL Library is compatible with all of the hardware and software environments supported by the XCLIB, 4MOBJ, and SVOBJ libraries. PXIPL routines operate directly upon imaging board buffers, upon images in PC memory, or upon images stored on disk.

PXIPL provides a wide selection of imaging routines. The major categories include: processing, enhancements, graphic lines and shapes, text overlay, printing, morphology, filters and edge detectors, transforms, convolutions, sequence integration and averaging, image printing, image copy and resizing, single image and image pair normalizations, blob analysis, histograms and moments, image load and save, calibration, correlation, subpixel accuracy measurements, and particle tracking.

SOPHISTICATED SOLUTIONS

The PXIPL C/C++ Function Library allows embedding image processing and analysis into user-written applications. Under Windows, the PXIPL DLL provides services to existing Windows applications which support “hooks” into DLLs.

Under Windows, PXIPL also provides image display on the S/VGA with integrated (non-blinking) cross-hair cursor overlay and integrated palette modifications. PXIPL also provides “waterfall” display of repeatedly captured image lines on the S/VGA.

PXIPL assists user-written programs in applications such as image enhancement, archival, analysis, and measurement; event and motion study; document capture; particle analysis; visual inspection; machine vision and quality control. Join the scientists and engineers in medical, industrial, and research environments who rely upon EPIX® imaging solutions.

PXIPL™ LIBRARY Image Processing and Analysis Library

FEATURES

Resolution Flexibility - PXIPL functions can process images of almost any size, located either in image board memory, PC memory, or disk files.

PXIPL functions will process any image captured by an EPIX® imaging board using either 4MOBJ, SVOBJ, or XCLIB software. Typical capture resolutions include 4x1, 32x32, 512x240, 752x480, 768x580, 1024x768, or 2048x2048. Monochrome pixels with a dynamic range from 1 bit (2 grey levels) to as large as 16 bits (2^{16} grey levels) can be processed. Color pixels, in either RGB, YCrCb, or HSB color space, with a range of 1 to 16 bits per color component, are supported. Selected operations also support up to 32 bits per pixel. Image sequence operations, such as sequence average or sequence integration, support up to 2^{23} images (8 bits per pixel).

The PXIPL functions are not restricted to processing images which were captured by EPIX® imaging boards. Images from any source, residing in PC memory, can be any size and any number of colors, limited only by availability of PC memory, and the CPU word size.[1]

Virtual Memory - Should PC memory be insufficient, images may also reside in disk files. All images, whether in an imaging board buffer, in PC memory, or in a disk file, can be enhanced and analyzed by the same functions!

Functional Flexibility - Typical PXIPL functions provide a broad spectrum of operations, allowing a single function to do the work of many functions. For example, a convolution function accepts parameters describing the image buffer, the area of interest within the buffer, the convolution size N, and the NxN kernel coefficients. This single function allows convolving with a 3x3, 9x9, 31x31, or 99x99 kernel size, limited only by available PC memory.

Efficiency - PXIPL functions are coded in optimized C, with selected segments hand-coded in assembler. Many functions internally identify special cases, invoking code optimized for each special case. The NxN convolution, for example, examines the coefficients provided and selects custom routines depending upon the size of N, the multiplication and summation precision needed, and whether division is required.

Proven Performance - The same functions provided with PXIPL also form the backbone of the ready-to-run XCAP, 4MIP, SVIP, and XCIP interactive image analysis programs, and have been proven through daily use in on-line, rigorous, imaging applications.

Image Selection Flexibility - A typical enhancement function operates on any image buffer, on either the full image or selected area of interest, with the result saved to any buffer or area of interest of the same dimensions. Image pair operations allow independent selection of the two source image operands and of the image destination.

PXIPL functions for nonrectangular regions use a common method of region specification, supporting rotated elliptical, rotated rectangular, N-sided polygon, boundary path, and scan list specifications.

Functions can operate on any pixel color component of a color image; selected functions can also operate upon all color components.

```
struct pximage im1, im2, im3;
struct  pxy    xysize = {752, 480};
unsigned char  buffer[752][480];
void           *mallocbuf = NULL;

im1 = *pxd_defineImage(1,1,   // access imaging
       0,0,-1,-1,..,"Grey");  // board’s buffer 1.

pximage_memory(&im, buffer,   // access existing
    &xysize, PXDATUCHAR,      // image in PC malloc memory,
    8, 1,                     // size 752x480, of chars,
    PXHINTGREY, 0);           // 8 bits per pixel,
                              // one color, monochrome

pximage_memmalloc(&im3,       // create & access new image
     &mallocbuf,&xysize,      // in PC memory, 752x480,
     PXDATUCHAR, 8,           // of chars, 8 bits/pixel,
     1, PXHINTGREY);          // one color, monochrome


pxip8_pairsub(&im1, &im2,     // Subtract pixels of
              &im3, 0);       // image 1 from image 2,
                              // put result in image 3.
```

Operating upon imaging board buffers & images in PC memory.

```
struct  pximage *ip1, *ip2, *ip3;
unsigned long   histogram[16], cnt;


ip1 = pxd_defineImage(1,1,  // access image board buffer 1,
         188,120,564,360,   // AOI of center 1/4 (assuming
         .., "BofRGB");     // 752x480), RGB color space,
                            // access color #3, B of RGB.

pxip8_histab2(NULL, ip1,    // compute Blue AOI histogram
         histogram,16);     // binned into 16 ranges.

ip2 = pxd_defineImage(1,1,  // access image board buffer 1,
         0,0,-1,-1,..,      // full image AOI, as HSB,
         "SofBSH");         // access color #2, S of HSB.

pxip8_pixthresholdcnt(NULL, // count Saturation
         ip2, 42, 0,&cnt);  // values >= 42

ip3 = pxd_defineImage(1,2,  // access image board buffer 2,
         0,0,-1,-1,..,      // full image AOI, as HSB
         "SofBSH");         // access color #2, S of HSB

pxip8_copy(NULL,&ip3,&ip2); // set saturation of buffer 1
                            // from buffer 2, leaving
                            // hue & brightness unchanged
```

Operating upon selected colors of selected color space.

FEATURES

PXIPL FUNCTIONS

Add Pixels of Image Pair
Add Pseudo-Random Noise
AND Pixels of Image Pair
AND Pixels with Mask in Region
AND Pixels with Mask
Average Image Sequence
Average Pixels of Image Pair
AVI 1.0 File, Save Image Sequence
AVI 1.0 File, Save Image Sequence
AVI 1.0 File, Save Sequence - Init
AVI 2.0 File, Save Image Sequence
AVI 2.0 File, Save Sequence - Init
AVI File, Load Image Sequence
AVI x.0 File, Save Sequence - Add Image
AVI x.0 File, Save Sequence - Done
Binary File, Save Sequence - Add Image
Binary File, Save Sequence - Done
Binary File, Save Sequence - Init
Blend Pixels of Image Pair
BMP File, Load Image
BMP File, Save Image
Calibrate Intensity/Density Mapping
Calibrate Spatial Mapping
Complement Pixel Values in Region
Complement Pixel Values
Compress Region Path
Compute Center of Mass of N'th Power of Region
Compute Center of Mass of N'th Power
Compute Center of Mass of Region
Compute Center of Mass
Compute Center of Mass, Binary Image Region
Compute Center of Mass, Binary Image
Compute Histogram on Region
Compute Histogram on Region
Compute Histogram on Region
Compute Histogram Statistics w. Interpretation
Compute Histogram Statistics w. Interpretation
Compute Histogram Statistics
Compute Histogram Statistics
Compute Histogram
Compute Moments of Region w. Interpretation
Compute Moments of Region
Compute Moments w. Interpretation
Compute Moments
Compute Radial Mass w. Interpretation
Compute Radial Mass
Compute Shape Statistics of Image Region
Compute Tabulated Histogram of Differences on Region
Compute Tabulated Histogram of Differences
Compute Tabulated Histogram
Compute Tabulated Histogram
Construct PXIMAGE: 2-D Slice of 3-D Image
Construct PXIMAGE: 3-D Representation of 2-D Image
Construct PXIMAGE: Access Freq. Domain Complex Image
Construct PXIMAGE: Access Image in File
Construct PXIMAGE: Access Image in File, Done
Construct PXIMAGE: Access Image in Host Memory
Construct PXIMAGE: Access Image in Host Memory
Construct PXIMAGE: Access Imaging Board Buffer
Construct PXIMAGE: Access Imaging Board Buffer
Construct PXIMAGE: Allocate Image in Malloc'ed Memory
Construct PXIMAGE: Allocate Image in Malloc'ed Memory
Construct PXIMAGE: Converted Color Space of Image
Construct PXIMAGE: Release Image in Malloc'ed Memory
Construct PXIMAGE: Release Image in Malloc'ed Memory
Construct PXIMAGE: Slice of Color Image
Construct PXIMAGE3: Access Image Sequence in Host Memory
Construct PXIMAGE3: Access Image Sequence in Host Memory
Construct PXIMAGE3: Access Imaging Board Buffers
Construct PXIMAGE3: Allocate Image Sequence in Malloc'ed Memory
Construct PXIMAGE3: Allocate Image Sequence in Malloc'ed Memory
Construct PXIMAGE3: Release Image Sequence in Malloc'ed Memory
Construct PXIMAGE3: Release Image Sequence in Malloc'ed Memory
Contrast Enhance By Percentile
Contrast Enhance Real Pixels
Contrast Enhance Region By Percentile
Contrast Enhance Region Real Pixels
Contrast Enhance Region
Contrast Enhance
Contrast Match Image Pair
Copy & Area Interpolation
Copy & Bilinear Interpolation w. Orientation
Copy & Convert Data Types
Copy & Exchange Image Buffers
Copy & Nearest Neighbor Interpolation w. Orientation
Copy & Skew Image Left/Right
Copy & Skew Image Up/Down
Copy & Spatial Replicate
Copy Image Buffer Region
Copy Image Buffer with Reversal
Copy Image Buffer with Shift
Copy Image Buffer
Copy Image with Rotation
Copy Image with Warping
Copy Image with Warping
Copy Slice of Image Buffer
Correct Image as per Speckle Mask
Correct Image as per Speckle Mask
Correlation Peak
Correlation Profile
Count Pixels by Threshold
Count Real Pixels by Threshold
Count Region Pixels by Threshold
Count Region Real Pixels by Threshold
De-Flicker Interlace: Line Pair Average
De-Flicker Interlace: Line Pair Duplicate
De-Flicker Interlace: Modify Singularities
Decode SMPTE Vertical Interval Time Code
Difference for Insert of Image Pair
Dither Pixels, Uniform
DOS Mouse: Get Clicks
DOS Mouse: Get Motion
DOS Mouse: Get Status
DOS Mouse: Initialize Access
DOS Mouse: Terminate Access
DOS S/VGA: Set Mode and Initialize Access
DOS S/VGA: Terminate Access
Draw 2-D Cosine Product Pattern
Draw 2-D Fiducial Pattern
Draw 2-D Gaussian Pattern
Draw 2-D Separable Patterns
Draw Alignment Pattern
Draw Arrow
Draw Box
Draw Characters
Draw Curved Line defined as Bezier Polynomial
Draw Ellipse
Draw Icon or Cursor
Draw Icon Primitive, Free Resources
Draw Icon Primitive, Initialize
Draw Icon Primitive, Modify Pixels
Draw Icon Primitive, Test Completion
Draw Line Segment
Draw Region Boundary
Draw Region Path
Draw Test Pattern
Draw Text from Font Map
Draw Text
Edge Detection, Kirsch
Edge Detection, Roberts
Edge Detection, Sobel Absolute
Edge Detection, Sobel
Edge Gradient, Thin
Ellipse Fitting Measurement
Errors: Translate Error Code to String
Exclusive OR Pixels of Image Pair
Export Region to File
Extend Region Path
Extend Region Path
FFT: Filter Frequency Domain
FFT: Get Dimensions of Freq. Domain Representation
FFT: Inverse Transform Image
FFT: Log Magnitude Plot of Freq. Domain
FFT: Scale Freq. Domain by Log Magnitude Plot
FFT: Transform Image
Field Interlaced Image Line Shuffle
Field Interlaced Image Line UnShuffle
FIFO Average
Filter, Low Pass, Fixed
Filter, Low Pass, Low Smear
Filter, Low Pass, Weighted
Filter, Median
Filter, Median, Binary Images
Filter, Median, Weighted
Filter, Rank High (Dilate)
Filter, Rank Low (Erode)
Filter, Sharpen, Laplacian
Find Blobs and List
Find Blobs and List
Find Blobs, Analyze and List
Find Region's Enclosed Area
Find Region's Enclosing Window
FITS File, Load Image
FITS File, Save Image
Follow and Collect Region Boundary by Value
Free Region
Gamma Correction
Get PXIMAGE: Access Imaging Board Buffer
Get PXIMAGE: Access Imaging Board Color Buffer
Get PXIMAGE: Access Imaging Board Frame Buffer
Get PXIMAGE: Access Imaging Board Frame Buffer
Get PXIMAGE: Release Access to Imaging Board Frame Buffers
Get PXIMAGE3: Access Imaging Board Buffers
Get PXIMAGE3: Access Imaging Board Color Buffers
Get PXIMAGE3: Access Imaging Board Frame Buffers
Get PXIMAGE3: Access Imaging Board Frame Buffers
Get PXIMAGE3: Release Access to Imaging Board Frame Buffers
H-P PCL Font: Draw Line of Characters
H-P PCL Font: Load
H-P PCL Font: Obtain Character Info
H-P PCL Font: Obtain Information
H-P PCL Font: Unload
Halftone by Black/White Sum
Histogram Equalization
Image File, Obtain Information on Subfiles
Image File, Obtain Information
Image File, Release Information
Import Region from File
Initialize Region Path
Insert of Differences of Image Pair
Integrate Image Sequence
JPEG File, Load Image
JPEG File, Save Image
Left Shift Pixel Values in Region
Left Shift Pixel Values
Line Pair Pixel Shuffle
Line Pair Pixel UnShuffle
Linux: Display Cursor via XWindows/X11
Linux: Display Image via XWindows/X11
Load Image from File, Hex ASCII
Load Image from File, Packed Binary
Load Image from File, Unpacked Binary
Load Image Sequence from File, Packed Binary
Load Image Sequence from File, Unpacked Binary
Map Pixel Values in Region
Map Pixel Values
Map Uchar Pixel Values in Region
Map Uchar Pixel Values
Map uint16 Pixel Values in Region
Map uint16 Pixel Values
Map uint32 Pixel Values in Region
Map uint32 Pixel Values
Maximum of Pixels of Image Pair
Medial Axis Thinning
Minimum of Pixels of Image Pair
Modify Region Definition: Rectangle to Polygon
Morphology Close
Morphology Dilation w. 3x3 Element
Morphology Dilation
Morphology Erosion w. 3x3 Element
Morphology Erosion
Morphology Hit-Miss
Morphology Open
MSB Extend Pixel Values in Region
MSB Extend Pixel Values
Normalize Columns' Mean
Normalize Image as per Background Image
Normalize Lines' Mean
NxN Convolution, Integer
NxN Convolution, Real
NxN Dynamic Threshold
NxN Inverse Contrast Ratio Mapping
Obtain Filtered pximage Access into Imaging Board Memory
Obtain Filtered pximage3 Access into Imaging Board Memory
Obtain pximage Access into Imaging Board Memory
Obtain pximage3 Access into Imaging Board Memory
Offset Pixel Values in Region
Offset Pixel Values in Region
Offset Pixel Values
Offset Pixel Values
OR Pixels of Image Pair
OR Pixels with Mask in Region
OR Pixels with Mask
Overlay Pixels of Image Pair
Overlay Pixels of Image Pair
Paint within Region
PCX File, Save Image
Perform Intensity/Density Mapping
Perform Inverse Spatial Mapping
Perform Spatial Mapping
Print Image
Product of Pixels of Image Pair
Product of Pixels of Image Pair
PXIMREGION: NonRectangular Image Region Specification
Ratio of Pixels of Image Pair
Ratio of Pixels of Image Pair
Recursive Average
Release Intensity/Density Mapping State
Release Spatial Mapping State
Right Shift Pixel Values in Region
Right Shift Pixel Values
S/VGA: Display Cursor
S/VGA: Display Image
S/VGA: Translate Image to Screen Coordinates
S/VGA: Translate Screen to Image Coordinates
S/VGA: Waterfall Line Display
Save Image Sequence to File, Packed Binary
Save Image Sequence to File, Unpacked Binary
Save Image to File, Hex ASCII
Save Image to File, Packed Binary
Save Image to File, Unpacked Binary
Scale Pixel Values in Region
Scale Pixel Values in Region
Scale Pixel Values
Scale Pixel Values
Scan, Connect, Collect Region by Table
Scan, Connect, Collect Region by Value
Search for Largest Pixel Value
Search for Pixel by Table
Search for Pixel by Value
Search for Smallest Pixel Value
Set Color Pixel Values in Region
Set Color Pixel Values
Set Pixel Components to Maximum in Region
Set Pixel Components to Maximum
Set Pixel Components to Median in Region
Set Pixel Components to Median
Set Pixel Components to Minimum in Region
Set Pixel Components to Minimum
Set Pixel Values in Region
Set Pixel Values
Set PXIMAGE: Set 2-D Area of Interest Window
Set PXIMAGE3: Set 3-D Area of Interest Window
Set Real Pixel Values
Shift Image One-Half Line Up or Down
Shuffle Column Order to Even-Odd Halves
Shuffle Even-Odd Halves to Column Order
Spatial Intensity Normalization
Spatial Quantization & Shrink
Subpixel Edge Measurement
Subtract Pixels of Image Pair
Swap Line or Column Pairs
Targa File, Save Image
Threshold Pixel Values in Region
Threshold Pixel Values in Region
Threshold Pixel Values in Region
Threshold Pixel Values
Threshold Pixel Values
Threshold Pixel Values
TIFF File, Load Image Sequence
TIFF File, Load Image
TIFF File, Save Image Sequence
TIFF File, Save Image
TIFF File, Save Sequence - Add Image
TIFF File, Save Sequence - Done
TIFF File, Save Sequence - Init
Tile Image Sequence
Track Particle Motion
Translate Region Definition to Path
Translate Region Definition to Scan List
User-Defined Premature Termination Functions
Windows: Create Device Independent Bitmap (DIB)
Windows: Display Cursor via GDI
Windows: Display Image via DirectDraw
Windows: Display Image via GDI
Windows: Display Image via Video for Windows
Windows: Draw Text using FONT
Windows: Release Device Independent Bitmap (DIB)
Windows: Translate Device to Image Coordinates
Windows: Translate Image to Device Coordinates
Windows: Waterfall Line Display via GDI
XOR Pixels with Mask in Region
XOR Pixels with Mask

PXIPL™ LIBRARY Image Processing and Analysis Library

FEATURES

```
int kernel[15][15], i, j;               // Define 15x15 kernel as low pass
for (i = 15; i--; )                     // filter with all coefficients 1.
    for (j = 15; j--; )
        kernel[i][j] = 1;

pxip8_NxNconvolve(                      // Do 15x15 convolution on 100x100
    pxd_defineImage(1,1,0,0,100,100,..,"Grey"),
                                        // AOI of buffer 1, result into
    pxd_defineImage(1,2,0,0,100,100,..,"Grey"),
    15, kernel, 0, 0, 0);               // buffer 2.
```

Performing 15x15 convolution on AOI.

```
struct  pxywindow   bounds;
struct  pxip8blob   blob[100];          // results
struct  pxy  search = {-1, 0};          // init search coordinates
int     n, i;
bounds.nw.x = 3;                        // min blob width
bounds.nw.y = 3;                        // min blob height
bounds.se.x = 100;                      // max blob width
bounds.se.y = 100;                      // max blob height
n = pxip8_bloblist(NULL,                // search image buffer 5
        pxd_defineImage(1,5,0,0,-1,-1,..,"Grey"),
        &search, ’g’^’t’, 123, 0,      // for up to 10 blobs
        &bounds,0,NULL,100,blob,NULL);  // identified by pixel
                                        // values >= 123

printf("Blobs found: %d\n", n);
for (i = 0; i < n; i++)                 // report blobs
    printf("Blob: %d, Center of Mass: (%g,%g), Area: %ld\n",
           i, blob[i].ucom.xd, blob[i].ucom.yd, blob[i].xyarea);
```

Searching for blobs.

```
struct pxio8tiffparm tiffparm;
memset(&tiffparm, 0, sizeof tiffparm);
tiffparm.bits = 8;                      // set number of bits to be saved
tiffparm.description = "Test Run #4";   // and a short description.

for (int i = 0; i < 100; i++)           // save sequence of 100 buffers
    pxio8_tiffwrite(NULL,
        pxd_defineImage(1,i+1,0,0,-1,-1,..,"Default"),
        "RUN4.TIF", i,                 // from image buffer i & full AOI
        NULL, &tiffparm, 0);            // to file name & subimage number
```

Saving image sequence to single TIFF file.

```
double  mass, xcenter, ycenter;
pxip8_masscenter(NULL,                  // use AOI of buffer 3
    pxd_defineImage(1,3,0,0,100,100,..,"Grey"),
    &mass, &xcenter, &ycenter);         // returned results
printf("Mass Center @ (%g,%g)\n",
        xcenter, ycenter);              // report results
```

Computing Center of Mass of AOI.

SPECIFICATIONS

IMAGING BOARD:

For use with XCLIB: Any PIXCI® A, CL1, CL2, CL3SD, D, D24, D32, D2X, D3X, E1, E1DB, E4, E4DB, EB1, EB1-PoCL, EC1, ECB1, ECB1-34, EL1, EL1DB, SI, SI1, SI4 SV2, SV3, SV4, SV5, SV5A, or SV5B imaging board. Any PIXCI® imaging board with an EPIX® SILICON VIDEO® camera.

For use with 4MOBJ: Any 4MEG VIDEO™ Model 5, Model 10, or Model 12, imaging board. Also supports the IMAGE MEMORY EXPANSION and the COC40 series[2] for use with the Model 12.

For use with SVOBJ: Any SILICON VIDEO® MUX™ imaging board.

ENVIRONMENT:

Standard versions support:

- Microsoft C/C++ V7.0, V8.0 (Visual C/C++ V1/V2) 16 bit in M or L models. For DOS V3.0 or later, 8088 or better.
- Borland C/C++ V4.0, V5.0 16 bit in M or L models. For DOS V3.0 or later, 8088 or better.
- Watcom C/C++ V11.0 32 bit in F model. For Tenberry (Rational) DOS extender, 80386 or better.
- Windows 3.x 16 bit DLLs, for any compiler or Windows application. For Windows V3.x, Standard or Enhanced mode, 80286 or better.
- Windows 95, 98, ME 32 bit DLLs, for any compiler or Windows application.
- Windows NT, 2000, XP, Vista 32 bit DLLs, for any compiler or Windows application.
- Windows XP(x64), Vista(x64) 64 bit DLLs, for any compiler or Windows application.
- Linux V2.4.8 or later kernel on Intel 80x86.
- Linux V2.6 or later kernel on Intel x86-64.

Other environments available on request.

Memory requirements: Approximately 16 to 1024 Kbytes, dependent upon selection of library routines.

PXIPL is optionally provided with, and must be used with, the 4MOBJ, SVOBJ, XCLIB version with which it is packaged. PXIPL routines require the presence of a supported imaging board.

LICENSING:

Licensing permits royalty free inclusion of library routines into programs using the 4MEG VIDEO™, the SILICON VIDEO® MUX™, or the PIXCI® imaging boards.

SOFTWARE INCLUDES:

As required by chosen environment: Object code libraries (.lib), Dynamic Link Library (.dll), and/or Object code archive (.a).

C prototype files (.h).

Printed manual(s).

[1] In 16 bit programming environments: the product of the number of pixels per line and color components per pixel may not exceed 32767, the number of lines may not exceed 32767. In 16 and 32 programming environments: the size of an image and image sequence may not exceed 2^{32}-1 bytes. In 64 programming environments: the size of an image may not exceed 2^{32}-1 bytes, the size of an image sequence may not exceed 2^{64}-1 bytes.

[2] PXIPL for the COC40 supports Native and Bound routines for the TMS320C40, as well as PC routines. A detailed description is provided in the PXIPL-COC40 brochure.

The Waterfall Display feature requires digitization and field counts, and is intended only for use with imaging board buffers. Specifications subject to change without notice.

EPIX® imaging products are made in the USA.

EPIX, 4MEG VIDEO, COC40, SILICON VIDEO, SILICON VIDEO MUX, PIXCI, QUICK SET VIDEO, 4MOBJ, SVOBJ, XCLIB, XCOBJ, and PXIPL are trademarks or registered trademarks of EPIX, Inc. Other brand, product, and company names are trademarks or registered trademarks of their respective owners.

EPIX®

EPIX, Incorporated
381 Lexington Drive
Buffalo Grove, IL 60089 USA
Tel - 847 465 1818
Fax - 847 465 1919
epix@epixinc.com
www.epixinc.com