Matrox Imaging Library 开发介绍

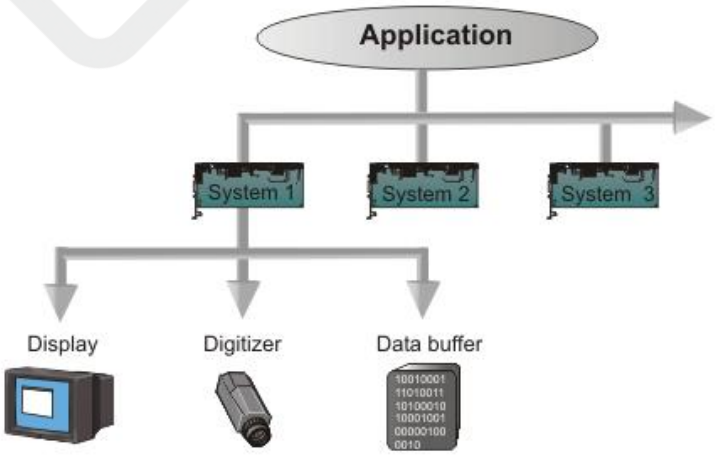
目录

Matrox Imaging Library

MIL 即为 Matrox Imaging Library 的缩写，是加拿大 Matrox 公司提供的图像处理函数库，主要是针对其公司生产的 Matrox 系列图像采集卡。我使用的是一个 Matrox RapixoCXP 采集卡，需要使用 MIL 开发自己的视觉系统。

一、MIL 中有五大基本对象

(这里的对象和 C++的对象不一样，不过也可以看成一样的，也可以看成一个结构体)：Application、System、Display、Digtizer、Buffer。如下图是这 5 大基本对象的一个简单关系图。



1、Application：

Application 指的是你自己开发的一个应用程序，一般应用程序同一时刻只存在一个 Application 对象。主要用它来提供一个用于控制和执行 MIL 应用程序的基本环境。

2、System：

System 代表为一个包含 CPU 或 GPU、内存或显存和图像控制器的单元分配的一个虚拟访问对象，例如一块 Matrox 图像板卡，一个电脑主机都可以被分配为一个 System。System 能够通过加上相机和显示器来采集、保存和显示。每个 Application 下可以包含多个 System，这就好比一台电脑可以插上多块 Matrox 图像板卡。

3、Buffer：

Buffer 对应一块内存，可以对它赋予不同的属性用来对图像作相应处理，如存储、显示、采集、处理，只有赋予了对应的属性的 Buffer 才能用于对应操作，只赋予了保存属性的 Buffer 是不能用于显示的。

4、Digtizer：

Digitizer 对应相机,它用于相机的采集和相机属性的调整等，和相机有关的操作都是靠它来完成。

5、Display：

Display 对应显示器,所有和显示的操作都是靠它来完成。这个在手册中提到了两种 Display:一种是 MIL 内建的用于演示的 Display 叫 Auxiliary Display,它不适用于 Windows Desktop，主要用于和 Matrox 显卡配套使用的 Screen，一般用不上，不予讨论；另一种是叫 Windowed Display，一种是 MIL 用于演示的 Display，不需要选择要显示的窗口句柄，默认分配的 Display 对象是此种 display，另一种是用户选择要显示的窗口句柄，这个需要你自己选择在哪个 windows 窗体上显示对应的 Buffer 图像内容。

二、功能浏览(函数方式)

1、Mapp：应用级

带有 Mapp 前缀的函数构成应用程序模块。应用程序模块可以初始化和控制 MIL 应用程序的执行环境。这个模块提供集成的调试服务，高精度计时器，方便的线程和事件控制，和对处理和内存操作的系统资源的自动补偿。

MIL_ID MappAlloc(long InitFlag, MIL_ID *ApplicationIdPtr)

void MappTimer (long ControlValue, double *TimePtr)

2、Msys：系统级

系统分配和查询模块支持分配和查询系统。系统通常指图像采集卡，也可以是显卡和主机 CPU。该模块用来指定程序中要用的物理设备，并且进行一些系统级的设置。只有当系统资源已经设置完成，才可以添加相关的其他部分，如 Digitizer 来控制设备的细节方面。

MIL_ID MsysAlloc(MIL_TEXT_PTR SystemDescriptor,long SystemNum, long

InitFlag, MIL_ID *SystemIdPtr)

3、Mdig:数字化级

Digitizer 模块管理和控制板卡的数字化部分。无论是简单还是复杂采集，都可以使用 Digitizer 模块设置需要的状态，并且可以将命令发送到图像采集卡。

MIL_ID MdigAlloc(MIL_ID SystemId, long DigNum, MIL_TEXT_PTR DataFormat, long InitFlag, MIL_ID *DigIdPtr)

void MdigControl(MIL_ID DigId, long ControlType,double ControlValue)

void MdigGrab(MIL_ID DigId,MIL_ID BufID);

void MdigFocus();

4、Mbuf:内存级

以 Mbuf 为前缀的函数构成 Buffer 模块。 Buffer 模块可以分配和控制其他 MIL 模块函数要操作的数据内存（存储空间）。内存包括图像内存和查找表内存。Buffer 模块可以使用子 buffer 从一个内存区域中分离一部分，也可以将一个 buffer 中的连续，不连续或某些比特位拷贝到另外一个 buffer。这个模块的拷贝函数是经过优化的，来实现在最有效利用系统资源的可那下达到最快的传输速度。这个模块还可以以通用的存储格式（如 TIFF 和 JPEG）存储和读取图像数据。也可以创建，保存和读取 AVI 格式的文件。还可以对单幅和序列图像进行压缩和解压缩。压缩格式支持有损和无损 JPEG，以及 JPEG2000。此外，该模块还可以将从带有 Bayer 滤波器的相机采集到的图像转换为彩色图像。

5、Mdisp:显示级类

显示分配和控制模块可以显示已经分配的 buffer 中的图像。并且可以对显示进行一些处理和控制。显示控制提供很多显示效果，如添加注释，LUTs，移动和缩放。

MIL_ID MdispAlloc(MIL_ID SystemId, long DispNum, MIL_TEXT_PTR DispFormat, long InitFlag, MIL_ID *DisplayIdPtr)

void MdispZoom(MIL_ID DisplayId, long XFactor, long YFactor)

void MdispSelectWindow(MIL_ID DisplayId, MIL_ID ImageBufId,HWND ClientWindowHandle)

三、常用的相关的函数含义

1.Alloc	分配一个资源 ID
2.Free	释放资源 ID
3.Control	参数设置
4.Inquire	查询资源 ID 的参数设置
5.Save	保存

6.Restore 读取保存的内容

四、常用的函数(具体函数定义请查看 Milhelp 文档)

1.Mapp 类

●MappAllocDefault()//一个默认的配置，Application、System、Digitizer、Display、Buffer 都是按照默认的情况进行配置。

例：MappAllocDefault(M_DEFAULT, &MilApplication, &MilSystem, &MilDisplay, &MilDigitizer, M_NULL);

●MappAlloc()//这个函数的主要功能是：为你的应用创建一个控制和执行环境。使用完后记得 MapFree

例：MappAlloc(M_NULL, M_DEFAULT, &MilApplication);

●MappControl()//改变指定的 MIL 应用的属性。

例：MappControl(M_DEFAULT, M_ERROR, M_PRINT_DISABLE);

●MappFree()//释放默认应用分配的资源

例：MappFree(MilApplication);

●MappFreeDefault()

例：MappFreeDefault(MilApplication, MilSystem, MilDisplay, M_NULL, MilImage);

●MappTimer()//获取函数运行时间

例：MappTimer(M_DEFAULT, M_TIMER_RESET + M_SYNCHRONOUS, M_NULL);

```
/* Perform a first image transformation with the calibration grid. */
MappAllocDefault(M_DEFAULT, &MilApplication, &MilSystem, &MilDisplay,
    &MilDigitizer, M_NULL);
MappTimer(M_DEFAULT, M_TIMER_READ + M_SYNCHRONOUS, &Time);
MosPrintf(MIL_TEXT("The process time is \t\t\t%.3f ms\n\n"), Time*1000.0);
```

//输出 MappAllocDefault 函数的运行时间单位是秒。

●MappInquire()//获取取指定的应用配置情况

2.Msys 类

●MsysAlloc()//配置一个硬件环境，指定使用得板卡类型，使用板卡序号。该函数要在分配 buffer，display，digitizer 前使用。

例：MsysAlloc(M_SYSTEM_RAPIXOCXP, M_DEV0, M_COMPLETE, &MilSystem);

//开发时将 MsysAlloc 的第一项参数改成您所使用的板卡类型。当前使用的是 Rapixo cxp 的采集卡

●MsysFree()//释放分配的系统资源

例：MsysFree(MilSystem);//释放分配的系统资源

●MsysControl()//改变指定的系统的属性

例：MsysControl(MilSystem, M_ALLOCATION_OVERSCAN, M_ENABLE);

●MsysInquire()//获取系统设置。

例：MsysInquire(MilSystem, M_ALLOCATION_OVERSCAN, &AllocationOverscan);

3.Mdig 类

●MdigAlloc()//配置一个采集卡对象，指定可以使用的采集卡通道数，然后才能使用图像采集卡的函数。

例：MdigAlloc(MilSystem, M_DEV0, "1.dcf", M_DEFAULT, &MilDigitizer);

//请将 MdigAlloc 的第三项参数改成您所使用的 dcf 文件

●MdigFree()//释放分配的相机资源

例：MdigFree(MilDigitizer);

●MdigGrab()//这是最常用的采集函数，但每次只能采集一帧，存储在 Millmage 之中

例：MdigGrab(MilDigitizer,MilImage);

●MdigGrabContinuous()//:这是一个连续采集的采集函数，但注意，它并不会存储在 PCbuffer 之中，在最后一帧之前都是从相机 Buffer 直接总到显存 Buffer 的。直到最后一帧，才会储存在 PCbuffer 之中。

例：MdigGrabContinuous(MilDigitizer, MilImage);

●MdigProcess()//这个函数也是采图函数，不过，它在每一帧采集到的时候，都会调用它参数里面的回调函数，每来一帧调一次。相当于是帧事件驱动了这个回调函数。

例：MdigProcess(MilDigitizer, MilGrabBufferList, MilGrabBufferListSize,

M_STOP, M_DEFAULT, ProcessingFunction, &UserHookData);

●MdigControl()//修改 MilDigitizer 的参数

例：MdigControl(MilDigitizer, M_GRAB_TRIGGER, M_DISABLE);

●MdigInquire()//这个函数是用来读取 MIL 这个采集卡的参数的

例 : MdigInquire(MilDigitizer, M_GRAB_CONTINUOUS_END_TRIGGER, &GrabContinuousEndTrigger);

●MdigHookFunction()//回调函数

例： MdigHookFunction(MilDigitizer, M_GRAB_FRAME_START, ProcessingFunction, &UserHookData);

4.Mdisp 类

●MdispAlloc()//配置一个 display，把相机采集的图像使用该抽象进行显示。

例：MdispAlloc(MilSystem, M_DEFAULT, M_DEFAULT, M_WINDOWED, &MilDisplay);

●MdispFree()//释放分配的显示资源

例：MdispFree(MilDisplay);

●MdispControl()//改变指定的显示属性

例：MdispControl(m_MilDisplay, M_MOUSE_USE, M_ENABLE);//允许鼠标实现缩放和拖动

●MdispInquire()//获取当前显示的相关属性

●MdispSelect()//图像 Buffer 内容显示到相应 Display 上,此后修改 Buffer,Display 自动刷新

例：MdispSelect(MilDisplay, MilImage);

5.Mbuf 类

●MbufAlloc2d()//分配显示的图像 Buffer

例 ： MbufAlloc2d(MilSystem, 500, 500, M_DEF_IMAGE_TYPE, M_IMAGE+M_DISP, &MilImage);

●MbufFree()//释放分配的 Buffer 资源

例：MbufFree(MilImage);

●MbufControl()//可控制修改分配的 Buffer 资源的属性

●MbufInquire()//获取指定内存块的配置情况

例：MbufInquire(MilImage, M_SIZE_BYTE, &SrcImageDataSize);//数据大小

●MbufClear()//这个函数将指定的缓冲区清除为指定的颜色。

例：MbufClear(MilImage, 0L);

●MbufExport()//保存为指定格式的图像文件

例：MbufExport("1.bmp", M_BMP, MilBufferDraw);

●MbufExportSequence()//保存为视频

例 ：

MbufExportSequence("D:\\file1.avi",M_AVI_DIB,&ModifiedBufferId,1,20,M_WRITE);

●MbufLoad//先分配固定大小的 MIL Buffer，再将图像文件(.bmp，.jpg 格式文件)数据读到新建 MIL Buffer,图像文件分辨率不能比新建的 Buffer 大

//分配 Buffer

例：MbufAlloc2d(MilSystem,

MbufDiskInquire(".\\image\\1.JPG", M_SIZE_X, M_NULL),

MbufDiskInquire(".\\image\\1.JPG", M_SIZE_Y, M_NULL),

32+M_UNSIGNED,

M_IMAGE + M_DISP,

&MilBufferLoad);

MbufLoad(".\\image\\1.JPG", MilBufferLoad);

●MbufRestore()//从已有的图像文件(.bmp，.jpg 格式文件)读入图像数据新建 MIL Buffer,新建的 Buffer 内存大小由图像大小来确定

例：MbufRestore(".\\image\\1.JPG", MilSystem, &MilBufferRestore);

●MbufImport()//图像导入函数 MbufImport：可以使用其自身的自动识别及分配能力，不必再单独定义导入图像类型和需要分配的内存大小.

例: MbufImport(".\\image\\bird.mim", M_MIL, M_RESTORE, MilSystem, &MilBufferImportR);

●MbufImportSequence()//加载视频

例：MbufImportSequence("1.avi", M_DEFAULT, M_NULL, M_NULL, M_NULL, M_NULL, M_NULL, M_OPEN);

●MbufSave()//MbufSave 默认保存成 TIFF 6.0 兼容的 MIL 格式图像文件

例：MbufSave("1.tif", MilBufferDraw);

●MbufBayer()//拜耳化功能参数，在转换图像并计算适当的系数之后将系数写入指定的 Mil 阵列缓冲器中，然后使用这些自动计算的系数矫正转换后的图像。

使用支持板载拜耳转换的采集卡时，可以使用板载缓冲区或板外主机缓冲区将单波拜耳转换编码图像转换为 1 波段或 3 波段图像，使用源和目标板载图像缓冲区执行操作时采集卡必须能够在板载执行拜耳转换；否则操作将在主机上进行。

此外只有当有足够的内存来完成 Bayer 操作（例如，保存内部缓冲区）时，才能再板上执行 Bayer 操作，否则将在主机上执行该操作，并将结果复制到目标缓冲区，这将导致 Bayer 操作的速度降低。

一般新建和显示流程

```
MIL_ID MilApplication,   /* Application identifier.  */
       MilSystem,        /* System identifier.       */
       MilDisplay,       /* Display identifier.      */
       MilImage;         /* Image buffer identifier. */
```

```
//1.分配默认的应用和系统
MappAllocDefault(M_SETUP,  &MilApplication,  &MilSystem,  M_NULL,  M_NULL,
M_NULL);
```

```
//2.分配显示的图像 Buffer
MbufAlloc2d(MilSystem,   500,   500,   M_DEF_IMAGE_TYPE,   M_IMAGE+M_DISP,
&MilImage);
```

```
// 初始化 Buffer,内存中绘制相应图像
MbufClear(MilImage, 0L);
MgraText(M_DEFAULT, MilImage, 0L, 0L, " MIL ");
```

```
//3.分配显示 Display
MdispAlloc(MilSystem, M_DEFAULT, "M_DEFAULT",  M_WINDOWED, &MilDisplay);
```

```
//4.图像 Buffer 内容显示到相应 Display 上,此后修改 Buffer,Display 自动刷新
MdispSelect(MilDisplay, MilImage);
```

```
// 取消显示
```

```
MessageBox(TEXT("取消显示"));
MdispSelect(MilDisplay, M_NULL);

//5.释放分配的资源.
MessageBox(TEXT("释放所有"));
    MbufFree(MilImage);
MdispFree(MilDisplay);
MappFreeDefault(MilApplication, MilSystem, M_NULL, M_NULL, M_NULL);
```

显示到 Windows 窗体

注意唯一的不同在于使用 MdispSelectWindow

```
MIL_ID MilApplication,  /* Application identifier.  */
       MilSystem,       /* System identifier.       */
       MilDisplay,          /* Display identifier.      */
       MilImage;        /* Image buffer identifier. */

//1.分配默认的应用和系统
MappAllocDefault(M_SETUP,  &MilApplication,  &MilSystem,  M_NULL,  M_NULL,
M_NULL);

//2.分配显示的图像 Buffer
MbufAlloc2d(MilSystem,  500,  500,  M_DEF_IMAGE_TYPE,  M_IMAGE+M_DISP,
&MilImage);

// 初始化 Buffer,内存中绘制相应图像
MbufClear(MilImage, 0L);
MgraText(M_DEFAULT, MilImage, 0L, 0L, " MIL ");

//3.分配显示 Display
MdispAlloc(MilSystem, M_DEFAULT, "M_DEFAULT",  M_WINDOWED, &MilDisplay);

//4.图像 Buffer 内容显示到相应 Display 上,此后修改 Buffer,Display 自动刷新
```

```
    MdispSelectWindow(MilDisplay,                                    MilImage,
GetDlgItem(IDS_DISPALYAREA)->GetSafeHwnd());

    // 取消显示
    Sleep(5000);
    MessageBox(TEXT("取消显示"));
    MdispSelect(MilDisplay, M_NULL);

    //5.释放分配的资源.
    MessageBox(TEXT("释放所有"));
    MbufFree(MilImage);
    MdispFree(MilDisplay);
    MappFreeDefault(MilApplication, MilSystem, M_NULL, M_NULL, M_NULL);
```

联系我们：
北京志强视觉科技发展有限公司
电话：+86 (010) 80482120
传真：+86 (010) 80483130
邮箱：51camera@51camera.com.cn
网址：www.51camera.com.cn