Matrox Imaging Library (MIL) X Update 85
Release Notes (GigE Vision with MIL)
January 2021
(c) Copyright Matrox Electronic Systems Ltd., 1992-2021.

This document outlines what is new with MIL for GigE Vision and explains the current limitations and particularities when using MIL with GigE Vision devices.

It also presents last-minute information that did not make it into the manual or on-line help. Note that this help file serves to complement your manual. The information found in this file overrides your formally documented material.

Contents

1. GigE Vision with MIL

1.1 Differences between MIL X Update 85 and MIL 10 Update 82

- New Matrox Capture Works utility.

- Support for grab containers.

- Support for Matrox AltiZ 3D profile sensors.

- Support for GigE Vision 2.1 and GigE Vision 2.2 payload types.

- Support for GenICam 3.2.

- Various bug fixes and improvements.

1.2 Differences between MIL 10 Update 82 and MIL 10 Update 54

- Support for the Concord PoE with TOE (Trigger over Ethernet).

- Support for new M_DYNAMIC grab buffers.

- Various bug fixes and improvements.

1.3 Differences between MIL 10 Update 54 and MIL 10 Update 25

- Support for new GigE Vision 2.0 payload types.

- Support for GigE Vision 2.1 devices.

- Support of GenICam 3.1.

-    MIL's GigE Vision system does not require MIL's non-paged memory for image acquisition.

- Removed Windows® XP support.

- New set of inquires for GenICam feature enumeration.

- Various bug fixes and improvements.

1.4 Differences between MIL 10 Update 25 and MIL 10

- Support for GigE Vision 2.0 devices.

- Support for GigE Vision Action and Scheduled Action command.

- Support for GenICam feature change notification.

- Capture Assistant re-design to support USB3 Vision and GigE Vision 2.0 devices.

- Support of GenICam 3.0.

1.5 Differences between MIL 10 and MIL 9 Update 52

- Support for Microsoft® Windows® 8.

- Support for GigE Vision multicast mode in MIL.

- Support for GigE Vision multicast mode in the GigE Vision Configuration Assistant.

- Support for Microsoft® Network Driver Interface Specification 6.20.

- Additions to MdigControlFeature and MdigInquireFeature.

1.6 Differences between MIL 9 Update 52 and MIL 9 Update 34

- Support of GenICam 2.3.1.

- Support of GenICam chunk mode.

- Support of new pixel formats.

- New modes for MdigControlFeature/MdigInquireFeature.

- New InitFlags for MdigAlloc().

- GenICam features present in a DCF can now be marked as optional.

- Seamless support for Link Aggregated devices.

- Improved automatic packet size negotiation.

- Re-engineered the GigE Vision Configuration Assistant.

- Performance optimizations in the Matrox GigE Vision filter driver.

- Updated device feature browser's interface.

1.7 Differences between MIL 9 Update 34 and MIL 9 Update 9

- Support of GenICam 2.1.1

- Performance optimizations done to the Matrox GigE Vision filter driver.

- Support of more than 16 GigE Vision cameras. The default is now 32, the maximum is 128.

- Support of non-streaming GigE Vision devices (must use empty DCF file).

1.8 Differences between MIL 9 Update 9 and MIL 9 Update 8

- The tool used to install Matrox Imaging Library drivers under 32-bit Microsoft Windows XP (DrvInst.exe) has been updated to fix two issues:
  - Registry key corruption

  - MtxGigeFilter.sys installation that could lead to an error pop-up at MsysAlloc(): "System allocation error: Could not open handle to filter driver."

- Resolves compatibility issues with MdigControlFeature and MdigInquireFeature when MIL 9 Processing Pack 2 is installed.

1.9 What's new in MIL X Update 85 for GigE Vision

Note that MIL X Update 85 (U85) is a cumulative update, including all content from MIL 10 Update 82 (U82). MIL X Update 85 (U85) requires MIL X Service Pack 4.

For MIL X Update 85, information has either been integrated into the actual MILHelp, or is contained in these release notes.

1.9.1 Standards compliance

- Support for GigE Vision 2.2 RC2.

- Support for GenICam 3.2.

- Support for GenICam GenDC 1.0.

- Support for Standard Feature Naming Convention (SFNC) version 2.5.

1.9.2 New features summary

- Matrox Capture Works utility.

- Support for Matrox AltiZ 3D profile sensors.

- Support for GigE Vision 2.2 GenICam GenDC payload type.

- Support for GigE Vision 2.1 multi-part payload type.

- Support for grab containers.

- Support for device enumeration API.

- Support for the redesigned Matrox Feature Browser.

1.9.3 Matrox Capture Works

Matrox Capture Works is a new interactive utility to configure and test devices that make use of a GenICam-based interface standard. The version included in this update adds support for the GigE Vision interface standard. Matrox Capture Works will completely replace the Matrox Capture Assistant interactive utility in time and is available for Windows and eventually Linux. Matrox Capture Works adds, in particular, support for container payload types from the GenICam GenDC and multi-part additions to interface standards, support for 3D display, and specific support for the Matrox AltiZ 3D profile sensors.

1.9.4 Working with a 3D device

Grabbing from a 3D device

Data from a 3D device is grabbed using functions from the Digitizer module such as MdigGrabContinuous(), MdigGrab() and MdigProcess(). In most respects, grabbing data from a 3D device is similar to grabbing images from a 2D camera. However, to work with 3D data in MIL, you need to use a container. The reason is that 3D devices can transmit multiple components (range, confidence, etc.) during a grab.

If your 3D device transmits data using the GenICam GenDC or the GigE Vision multi-part payload types, you can grab directly into a container that was previously allocated using MbufAllocContainer().

For demonstrations on how to grab from such a 3D device with MIL, see the MdigProcess3D and MultiComponentGrab examples. These examples make extensive use of containers for grabbing and show the great flexibility they provide.

Grabbing into a MIL container

When grabbing into a container for the first time, MIL will free all existing components and allocate the

components required to store the current frame of data. MIL will typically reuse these automatically allocated components during subsequent grabs, unless you add or remove components from the container. If the latter is the case, all components will be freed and new components will again be allocated for the grabbed data.

MIL will not grab into components of the container that you have allocated manually using MbufAllocComponent(), MbufCopyComponent(), or MbufCreateComponent(). Rather, MIL will create components, as required.

Grabbing into a MIL buffer

If you grab into a traditional buffer from a device that outputs data in a format suitable for a container, the first intensity component (i.e., a standard image) in the frame of data will be grabbed. If there is no intensity component, the first component that is not metadata will be grabbed.

Grabbing using older utilities and certain examples

Limitations are to be expected when using older utilities (such as Matrox Intellicam and Matrox Capture Assistant), and certain MIL examples (such as MdigGrab, MdigGrabSequence and MdigProcess), to grab from particular 3D devices. These utilities and examples do not support grabbing multiple components. That is, they are not able to grab all of the components at once. If present, the first intensity component is always used. Otherwise, the first component that can be stored into a MIL image buffer is used.

1.9.5 Additions to the command reference.

- Additions to MsysControl() / MsysInquire()
  - Support for M_DISCOVER_DEVICE.
    - MsysControl(): Performs a device discovery. ControlValue must be M_DEFAULT. Note: executing this control can trigger an M_CAMERA_PRESENT hook.

  - Support for M_DISCOVER_DEVICE_COUNT.
    - MsysInquire(): Inquires the number of devices present. Inquire returns a MIL_INT.

  - Support for M_DISCOVER_DEVICE_DIGITIZER_NUMBER + N.
    - MsysInquire(): Inquires the MIL digitizer device number associated to the nth device. N must be between 0 and M_DISCOVER_DEVICE_COUNT – 1. Inquire returns a MIL_INT.

  - Support for M_DISCOVER_DEVICE_MANUFACTURER_NAME + N.
    - MsysInquire(): Inquires the device's manufacturer name, for the nth device. N must be between 0 and M_DISCOVER_DEVICE_COUNT – 1. Inquire returns a MIL_STRING.

  - Support for M_DISCOVER_DEVICE_MODEL_NAME + N.
    - MsysInquire(): Inquires the device's model name, for the nth device. N must be between 0 and M_DISCOVER_DEVICE_COUNT – 1. Inquire returns a MIL_STRING.

  - Support for M_DISCOVER_DEVICE_MANUFACTURER_INFO + N.
    - MsysInquire(): Inquires the device's manufacturer-specific information, for the nth device. N must be between 0 and M_DISCOVER_DEVICE_COUNT – 1. Inquire returns a MIL_STRING.

  - Support for M_DISCOVER_DEVICE_USER_NAME + N.
    - MsysInquire(): Inquires the device's user programmable device name, for the nth device. N must be between 0 and M_DISCOVER_DEVICE_COUNT – 1. Inquire returns a MIL_STRING.

  - Support for M_DISCOVER_DEVICE_VERSION + N.
    - MsysInquire(): Inquires the device version of the nth device. N must be between 0 and M_DISCOVER_DEVICE_COUNT – 1. Inquire returns a MIL_STRING.

  - Support for M_DISCOVER_DEVICE_SERIAL_NUMBER + N.
    - MsysInquire(): Inquires the device serial number of the nth device. N must be between 0 and M_DISCOVER_DEVICE_COUNT – 1. Inquire returns a MIL_STRING.

  - Support for M_DISCOVER_DEVICE_INTERFACE_NAME + N.
    - MsysInquire(): Inquires the name of the host interface associated to the nth device. N must be between 0 and M_DISCOVER_DEVICE_COUNT – 1. Inquire returns a MIL_STRING.

- o Support for M_DISCOVER_DEVICE_ADDRESS + N.
  - MsysInquire(): Inquires the address of the nth device. N must be between 0 and M_DISCOVER_DEVICE_COUNT – 1. Inquire returns a MIL_STRING.

- Additions to Mbuf…()
  - o Support for grab containers. See container support in the Mbuf module of the MIL Help for details.

  - o Support for M_GRAB_TIME_STAMP.
    - MbufInquire(): Inquires the time when MIL finished grabbing into the buffer, expressed in seconds. Inquire returns a MIL_DOUBLE.

  - o Support for M_GRAB_TIME_STAMP_NS.
    - MbufInquire(): Inquires the time when MIL finished grabbing into the buffer, expressed in nanoseconds. Inquire returns a MIL_INT64.

  - o Support for M_CAMERA_TIME_STAMP.
    - MbufInquire(): Inquires the time at witch the device generated the frame of data, expressed in seconds. Inquire returns a MIL_DOUBLE.

  - o Support for M_CAMERA_TIME_STAMP_NS.
    - MbufInquire(): Inquires the time at witch the device generated the frame of data expressed in nanoseconds. Inquire returns a MIL_INT64.

  - o Support for M_COMPONENT_TIME_STAMP.
    - MbufInquire(), MbufInquireContainer(): When using MbufInquire(), the BufId must be that of a component. When using MbufInquireContainer(), the BufId must be that of a container and the Component parameter must not be set to M_CONTAINER. When grabbing into a container, from a device transmitting data in GenDC format, this inquire gets the time at witch the device generated the data component expressed in seconds. Zero is returned if there is no component time stamp available. Inquire returns a MIL_DOUBLE. Note: components belonging to the same container can have different component time stamps.

  - o Support for M_COMPONENT_TIME_STAMP_NS.
    - MbufInquire(), MbufInquireContainer(): When using MbufInquire(), the BufId must be that of a component. When using MbufInquireContainer(), the BufId must be that of a container and the Component parameter must not be set to M_CONTAINER. When grabbing into a container, from a device transmitting data in GenDC format, this inquire gets the time at witch the device generated the data component expressed in nanoseconds. Zero is returned if there is no component time stamp available. Inquire returns a MIL_INT64. Note: components belonging to the same container can have different component time stamps.

  - o Support for M_COMPONENT_TIME_STAMP_IS_PTP.
    - MbufInquire(), MbufInquireContainer(): When using MbufInquire(), the BufId must be that of a component. When using MbufInquireContainer(), the Component parameter must not be set to M_CONTAINER. If M_COMPONENT_TIME_STAMP_IS_PTP returns M_TRUE, then M_COMPONENT_TIME_STAMP_NS is relative to the epoch January 1st 1970 0:00:00.

  - o Support for M_LAYOUT_MODIFICATION_COUNT.
    - MbufInquire(), MbufInquireContainer(): When using MbufInquire(), the BufId must be that of a container. When using MbufInquireContainer(), the BufId must be that of a container and the Component parameter must be set to M_CONTAINER. This value is modified every time a component is added or removed from the container. Inquire returns a MIL_INT.

1.9.6 Improvements

- M_CONTAINER grab buffer support was added to the MIL GigE Vision driver. A container is a data object that stores related buffers, referred to as the container's buffer components. Components are automatically added to a container, when you grab into it or use it as a destination of a supported function. See the MultiComponentGrab and MdigProcess3D examples located in examples\boardspecific\multicomponentgrab\c++ and examples\core\mdigprocess3d\c++, for details.

- We are now performing an explicit AcquisitionStop of the device in MdigFree() and MdigAlloc(). Some GigE

Vision devices do not stop the acquisition engine when the device is closed; opening the device results in immediate transmission of stream packets.

- GigE Vision device enumeration order is based on the MAC addresses of the device and its host NIC. Sorting was performed on the addresses in big Endian byte order resulting in a predictable but counter intuitive order. Proper sorting can be restored using "Resolve MAC address sort order" in Capture Assistant's system configuration page. Changing this parameter can change MIL digitizer device numbers associated to your GigE Vision devices.

- The Matrox Feature Browser was completely redesigned and now supports both Windows and Linux.

- Values of unavailable features are now kept blank in the Feature Browser.

- Ethernet flow control PAUSE frames are now enabled in GigE Vision devices, when supported. Note that flow control must also be enabled in the host Ethernet controller.

- The MtxGigeService now uses a dynamic port number. This resolves a potential communication failure issue between the service and MIL.

- Resolved a device discovery performance issue on networks with a DHCP server where GigE Vision devices are present and use an invalid IP address. The discovery routine execution time would become unusually long.

- It is now possible to specify the device packet size from the digitizer's configuration file (DCF). To do so, include the device packet size feature in the DCF.

1.9.7 Behavioral changes

- An error message is now generated if MdigControl() is executed with M_WHITE_BALANCE and M_CALCULATE while a grab is in progress. In previous versions the call would fail without error.

- The device's stream will now be immediately stopped after a grab completes when M_GC_STREAMING_STOP_DELAY is set to 0. On previous releases the stream would be stopped from the stream control thread during its next time slice which resulted in the stream continuing for a bit longer.

- The 64-bit version of the GigE Vision driver now allocates 10 auxiliary buffers at digitizer allocation time (up from 5).

1.9.8 Bug fixes

- Fixed bug in the GigE Vision device discovery routine when the subnet mask of the host NIC was not a multiple of 8 bytes (255.0.0.0, 255.255.0.0 or 255.255.255.0). In this case, the discovery could fail.

- Fixed rare timeout issue in device discovery routine that resulted in a crash of the MtxGigeService.

- Fixed crash that occurred in ChunkMode with the FeatureBrowser open and when the grab buffers were deleted immediately after the last grabbed frame was completed.

- Fixed issue with MdigInquire(M_CAMERA_PRESENT) returning incorrect information with some devices.

- Fixed variable frame size issue. MdigGetHookInfo(), when used with M_GC_FRAME_SIZE_Y, would return the device's frame height instead of the actual number of valid lines in the buffer.

- Fixed issue when starting Capture Assistant where the main window would be rendered off screen. Typically occurred when the display resolution was changed.

- Fixed MbufChild…() failure when allocating a child buffer from a parent with the M_GRAB attribute.

- Fixed issue with DCF loader where digitizer allocation could fail if the DCF contained floating point type device features.

- Fixed wrong error message issued from MdigInquireFeature() if low-level register access error was returned from the device.

- Fixed device selection issue in Capture Assistant's Tree - Detailed view.

- Fixed crash caused by a race condition that occurred while a memory access was made to the GigE Vision device and an asynchronous command was queued to the same digitizer.

- Fixed discovery issue that could occur right after OS boot-up when Ethernet controller has not yet been assigned an IP address.

- Fixed crash caused by invalid stream packet. Occurred when a stream packet transmitted by the device was smaller than the absolute minimum packet size allowed for a stream packet.

- Fixed crash caused by a race condition between MdigFree() and a device register access.

- Fixed packet size negotiation issue when jumbo frames (packets) are not enabled or supported.

- Fixed issue where an old frame would be returned when resuming the grab. Occurred when the GVSP block contains 5 packets or less.

- Fixed issue in initial release of MIL 10 Update 85 (MXU085B45) where low-level transport layer errors were dropped. This resulted in these errors not being signalled to the user.

1.10 What's new in MIL 10 Update 82 for GigE Vision

Note that MIL 10 Update 82 (U82) is a cumulative update, including all content from MIL 10 Update 54 (U54).

For MIL 10 Update 82, information has either been integrated into the actual MILHelp, or is contained in this release note. MIL 10 Update 82 updates the official documentation in the MIL Reference for the dig and sys modules.

1.10.1 Standards compliance

- Support for GigE Vision 2.1.

- Support for GenICam 3.1.

- Support for Standard Feature Naming Convention (SFNC) version 2.3.

1.10.2 New features summary

- Support for M_DYNAMIC grab buffers.

1.10.3 Additions to the command reference

- Additions to MbufAlloc…():
  - Support for M_IMAGE + M_GRAB + M_DYNAMIC buffer attribute.

- Additions to MdigInquire():
  - Support for M_GC_PIXEL_FORMAT_STRING: Returns the camera's PixelFormat as a string.

  - Support for M_GC_LOCAL_CONTROL_PORT: Returns the host UDP port number for the control channel

  - Support for M_GC_LOCAL_MESSAGE_PORT: Returns the host UDP port number for the control channel

  - Support for M_GC_LOCAL_STREAM_PORT: Returns the host UDP port number for the control channel

  - Support for M_GC_REMOTE_CONTROL_PORT: Returns the camera's UDP port number for the control channel

  - Support for M_GC_REMOTE_MESSAGE_PORT: Returns the camera's UDP port number for the

control channel

- o Support for M_GC_REMOTE_STREAM_PORT: Returns the camera's UDP port number for the control channel

- o Support for M_GC_LOCAL_IP_ADDRESS: Returns the IP address of the host NIC as a MIL_UINT64.

- o Support for M_GC_LOCAL_IP_ADDRESS_STRING: Returns the IP address of the host NIC as a string.

- o Support for M_GC_LOCAL_MAC_ADDRESS: Returns the MAC address of the host NIC as a MIL_UINT64.

- o Support for M_GC_LOCAL_MAC_ADDRESS_STRING: Returns the MAC address of the host NIC as a string.

- o Support for M_GC_REMOTE_IP_ADDRESS: Returns the IP address of the camera as a MIL_UINT64.

- o Support for M_GC_ REMOTE _IP_ADDRESS_STRING: Returns the IP address of the camera as a string.

- o Support for M_GC_ REMOTE _MAC_ADDRESS: Returns the MAC address of the camera as a MIL_UINT64.

- o Support for M_GC_ REMOTE _MAC_ADDRESS_STRING: Returns the MAC address of the camera as a string.

- o

- Additions to MbufInquire():
  - o Support for M_PFNC_VALUE
    - Returns the buffer's format as defined in the GenICam Pixel Format Naming Convention (PFNC).

  - o Support for M_PFNC_SUPPORT
    - Returns M_YES if the PFNC format is a native MIL format.

    - Returns M_WITH_COMPENSATION if the PFNC format is not a native MIL format but can be converted to a MIL native format. A buffer of this type can only be used as a source buffer.

    - Returns M_NO if MIL cannot handle this type of format. In this case the user has to access the buffer's memory directly. MbufInquire() with M_HOST_ADDRESS can be used to get a pointer to the buffer's data.

1.10.4 Improvements

- M_DYNAMIC grab buffer support was added to the MIL GigE Vision driver. M_DYNAMIC buffers do not have a format defined at allocation time. Instead their rectangular size and format is defined after a grab has been performed in the M_DYNAMIC buffer. The buffer's rectangular size and format can be inquired after the grab using MbufInquire(). The next time the dynamic buffer is used for grabbing its rectangular size and format might change again if the format of the image sent by the camera changes. The Dynamic buffer's size in bytes must be large enough to accommodate the largest image that can be sent by the camera. See the MdigGrabDynamic board specific example, located in examples\board-specific\mdiggrabdynamic\c++, for details.

- The FeatureBrowser was redesigned to support multiple platforms.

1.10.5 Behavioral changes

- M_GRAB + M_PROC buffers now have a default pitch that corresponds to the host system's pitch. Grab buffers without M_PROC continue to have a pitch equal to the line size.

1.10.6 Bug fixes

- Fix rare exception in CaptureAssistant after the splash screen is displayed.

- Fixed heap corruption issue issue with camera returning invalid strings from its discovery acknowledge.

- Fix ACTION_CMD issue when sent by the host. The maximum trigger rate was limited because of a systematic timeout that occurred waiting for the reception of the ACTION_ACK.

- Fix crash caused by the cancellation of a digitizer command not resetting the digitizer's grab state.

- Fixed hang caused by endless pending acknowledge processing loop with some GigE Vision devices.

- Fixed string clipping issue when reading non cacheable features (such as DeviceTemperature) from a device. The string returned could be clipped in some situations.

- Fixed crash in 32 bits when logging an error during grab.

- Added clear of GenICam XML cache when loading of a GenICam XML fails.

- Fixed "Unsupported camera pixel format." error message that occurred under certain circumstances when a received stream packet entered the MIL GigE Vision driver's packet cache.

- Fixed packet size negotiation when NIC is not set to use jumbo frames (packets).

1.10.7 Known limitations and particularities

- Windows' automatic 8.3 file name creation needs to be enabled in order for the MIL installer to access the temp folder when the user name contains a space. This option allows Windows to create short file/folder name aliases for paths with long names for programs, such as the MIL installer, that don't support spaces in the file/folder names. Alternatively, the MIL installer needs to run from a user account that belongs to the administrators group, with no spaces in the name. Note that the same applies for uninstalling MIL.

- The required Visual C++ 2017 Redistributable needs the presence of KB2919442 and KB2919355. These will need to be obtained and applied before restarting this update.

- Building the MIL examples using Visual Studio 2015 or 2017 also requires the presence of Windows SDK version 8.1, which is installed from the Visual Studio setup.

- During driver installation under Windows 7, you might continue to be asked to trust Matrox software after selecting the option to "Always trust software from MATROX ELECTRONIC SYSTEMS, LTD". To avoid being asked mutliple times, refer to Microsoft KB2921916.

1.11 What's new in MIL 10 Update 54 for GigE Vision

Note that MIL 10 Update 54 (U54) is a cumulative update, including all content from MIL 10 Update 25 (U25).

For MIL 10 Update 54, information has either been integrated into the actual MILHelp, or is contained in this release note. MIL 10 Update 54 updates the official documentation in the MIL Reference for the dig and sys modules.

1.11.1 Standards compliance

- Support for GigE Vision 2.1.

- Support for GenICam 3.1.

- Support for Standard Feature Naming Convention (SFNC) version 2.3.

1.11.2 New features summary

- Support for GigE Vision 2.0. features:

  - JPEG payload types.

  - JPEG 2000 payload types.

  - Extended chunk mode payload types.

1.11.3 Additions to the command reference

- Additions to MdigInquireFeature():
  - Support for M_FEATURE_ENUM_ENTRY_TOOLTIP + N
    - MdigInquireFeature(). Returns the nth enum entry's tooltip string. The string size can be inquired by adding the M_STRING_SIZE combination constant.

  - Support for M_FEATURE_ENUM_ENTRY_DESCRIPTION + N
    - MdigInquireFeature(). Returns the nth enum entry's description string. The string size can be inquired by adding the M_STRING_SIZE combination constant.

  - Support for M_FEATURE_ENUM_ENTRY_ACCESS_MODE + N
    - MdigInquireFeature(). Returns the nth enum entry's access mode. See M_FEATURE_ACCESS_MODE in the MIL documentation for details.

  - Support for M_FEATURE_ENUM_ENTRY_VISIBILITY + N
    - MdigInquireFeature(). Returns the nth enum entry's visibility value. See M_FEATURE_VISIBILLITY in the MIL documentation for details.

  - Support for M_FEATURE_ENUM_ENTRY_CACHING_MODE + N
    - MdigInquireFeature(). Returns the nth enum entry's caching mode. See M_FEATURE_CACHING_MODE in the MIL documentation for details.

  - Support for M_FEATURE_ENUM_ENTRY_STREAMABLE + N
    - MdigInquireFeature(). Returns the nth enum entry's streamable value. See M_FEATURE_STREAMABLE in the MIL documentation for details.

  - Support for M_FEATURE_SELECTOR_COUNT
    - MdigInquireFeature(). Returns the number of features selected by the specified FeatureName parameter. Note that the FeatureName parameter must specify a selector type feature.

  - Support for M_FEATURE_SELECTOR_NAME + N
    - MdigInquireFeature(). Returns the nth feature name, as a string, selected by the specified feature. Note that the FeatureName parameter must specify a selector type feature.

  - Support for M_FEATURE_VALID_VALUE_COUNT
    - MdigInquireFeature(). Integer and floating point type features can support a fixed list of valid values instead of the traditional minimum, maximum and increment values. M_FEATURE_VALID_VALUE_COUNT returns the number of valid values supported. A value of 0 returned indicates the feature does not support a list of valid values; minimum, maximum and increment values must be used instead.

  - Support for M_FEATURE_VALID_VALUE + N
    - MdigInquireFeature(). Returns the nth valid value for an integer of floating point type feature.

1.11.4 Improvements

- Automatic conversion of 10, 12, 14 and 16 bit monochrome pixel formats to 8 bits when grab buffer is 8 bit per pixel.

- GigE Vision 1.x devices with a link speed between 1.0 and 10.0 Gbps were considered link aggregated. This caused NBASE-T devices (2.5 and 5.0 Gbps devices) to be considered link-aggregated as well, resulting in non optimal packet resend settings. NBASE-T devices now have packet resend settings correctly adjusted at allocation time.

- Test packets sent during packet size negotiation are now handled at a lower level in the networking stack, resulting in successful packet size negotiation when the Windows® firewall is enabled.

- Added support for M_PAGED grab buffers.

- Optimized stream packet reception in an effort to reduce CPU utilization when interfacing multiple devices.

- Improved performance of digitizer Bayer conversions with 16 bit Bayer data sources.

- Capture Assistant can now receive GigE Vision and USB3 Vision events. Event reception is reported through the corresponding feature browser features.

- Capture Assistant now remembers the user's view settings and restores them at start-up.

- Added support for YCbCr411_8 pixel format.

1.11.5 Behavioral changes

- M_GRAB buffers are now M_PAGED by default.

1.11.6 Bug fixes

- Fixed corrupt frame issue with Link-Aggregated devices.

- Devices with UTF-8 encoded strings are now correctly handled.

- Fixed MdigInquire(M_GC_PACKET_SIZE). The value returned was not always valid.

- Fixed issue with M_GRAB_END. Hook signaling would happen too soon when grabbing in internal buffers with Bayer pixel formats or monochrome packed (Mono10p or Mono12p) pixel formats.

- Fixed ACTION_CMD support. There were instances where the ACTIONCMD_ACK received was not properly handled from all cameras, leading to incorrect error messages about missing acks.

- Fixed crash caused by a race condition between MdigFree() and MdigInquire().

- Fixed issue with not properly reporting M_GRAB_SCALE when a camera was allocated with a decimation factor other than 1.

- Fixed crash at MsysAlloc() when 32 or more NICs are present in the system.

- Fixed MbufChildMove() issue where the subsequent grab in a moved child buffer was not properly updated.

- Fixed MdigGetHookInfo(M_IO_STATUS) issue where incorrect information was returned.

- Fixed MdigControl(M_IO_INTERRUPT_ACTIVATION) issue where IO interrupts were not generated on the correct edge when a change was made from rising edge to falling edge and vice versa.

- Fixed handle leak at MsysAlloc().

- Fixed long processing loop in the stream packet processing module caused by an improperly formed GigE Vision Stream packet. Prevented our heartbeat thread from performing its duties resulting in camera disconnection.

- Fixed memory leak in MdigProcess() where 16 bytes of memory were leaked per buffer passed to MdigProcess().

- Fixed issue related to grab scale not being properly applied during a continuous grab.

- Fixed issue with changing chunk data. There was a race condition where chunk data could change while the user was processing chunks from the MdigProcess call-back function.

- Fixed MdigInquire(M_GRAB_TRIGGER_STATE) on cameras that have custom TriggerSelector values.

- Fixed issue that appeared when the number of allocated digitizers was greater than 32.

- Fixed crash caused by a race condition where the same command was dispatched twice.

- Fixed Capture Assistant's display allocation; M_WINDOWED mode is now forced.

- Fixed a pitch issue with some devices where padding is applied in the x direction.

- Fixed Recording mode in Intellicam when all camera features are dumped to DCF. Some feature selectors were not properly handled.

1.11.7 Known limitations and particularities

- Windows' automatic 8.3 file name creation needs to be enabled in order for the MIL installer to access the temp folder when the user name contains a space. This option allows Windows to create short file/folder name aliases for paths with long names for programs, such as the MIL installer, that don't support spaces in the file/folder names. Alternatively, the MIL installer needs to run from a user account that belongs to the administrators group, with no spaces in the name. Note that the same applies for uninstalling MIL.

- The MdigAlloc() M_GC_DEVICE_USER_NAME InitFlag was renamed to M_GC_USER_NAME. M_GC_DEVICE_USER_NAME will be deprecated in a future major release of MIL.

- The required Visual C++ 2017 Redistributable needs the presence of KB2919442 and KB2919355. These will need to be obtained and applied before restarting this update.

- Building the MIL examples using Visual Studio 2015 or 2017 also requires the presence of Windows SDK version 8.1, which is installed from the Visual Studio setup.

- The MIL update requires a Windows installation that supports device drivers with SHA-2 digital certificates. Consequently, some Windows 7 installations will require that a Windows Monthly Rollup be applied before the MIL update can be installed.

- During driver installation under Windows 7, you might continue to be asked to trust Matrox software after selecting the option to "Always trust software from MATROX ELECTRONIC SYSTEMS, LTD". To avoid being asked mutliple times, refer to Microsoft KB2921916.

1.12 What's new in MIL 10 Update 25

1.12.1 Standards compliance

- Support for GigE Vision 2.0.

- Support for GenICam 3.0.

- Support for Standard Feature Naming Convention (SFNC) version 2.0.

- Support for Microsoft® Network Driver Interface Specification 6.20.

1.12.2 New features summary

- Matrox GigE Vision driver:
  - Support for GigE Vision 2.0 features:
    - Image data payload type.
    - Image data + chunk mode payload type.
    - IEEE 1588 PTP (Precision Time Protocol).

- Scheduled action command.

- Single link configuration.

- Static link aggregation group configuration.

- Dynamic link aggregation group configuration.

o Support for GigE Vision 1.2 features:

- Action command.

o Support for GigE Vision 1.1 features:

- PENDING_ACK message.

o Support for GenICam feature change notification.

o Support for GenICam command type features in DCFs.

o Support for new PixelFormats.

- Board-specific examples
  o Modified the milgige GigE Vision specific example, located in examples\board-specific\milgige\c++. The example now prints the camera's various capabilities.

  o Modified the chunkmode GenICam specific example, located in examples\board-specific\chunkmode\c++. The example no longer hard-codes the chunks used. Instead the example enumerates the available chunks and lets the user select the chunks to use.

  o Modified the enumfeatures GenICam specific example, located in examples\board-specific\enumfeatures\c++. The example now uses MdigHookFunction (M_GC_FEATURE_CHANGE) and MdigGetHookInfo(M_GC_FEATURE_CHANGE_NAME) to receive notifications of GenICam feature changes.

  o New actiontrigger.cpp example, located in examples\board-specific\actiontrigger\c++. This example uses the GigE Vision Action command to trigger multiple devices using a single broadcast action packet. The example then continues and uses the GigE Vision Scheduled Action command to trigger multiple devices using a single broadcast scheduled action packet. The Scheduled Action command relies on IEEE-1588 PTP (Precision Time Protocol) to allow for finer precision of trigger events sent to multiple devices.

- CaptureAssistant
  o Redesigned to support USB3 Vision devices.

  o Added support of GigE Vision 2.0 devices.

  o Improved support for GigE Vision multicast modes.

  o Capture Assistant no longer requires administrative privileges at start-up.

  o Capture Assistant is now DPI aware. This allows for sharper user interface rendering in large fonts.

  o Can now select DCF file to use for device allocation.

1.12.3 Additions to the command reference

- Additions to MsysControl()/MsysInquire():
  - Action and Scheduled Action command (see actiontrigger.cpp board-specific example):

    - Support for M_GC_ACTION_DEVICE_KEY+N

      - MsysControl()/MsysInquire(). Adds an action device key to action context N. N can range from M_GC_ACTION0 to M_GC_ACTION31. Inquire returns a MIL_INT.

    - Support for M_GC_ACTION_GROUP_KEY+N

      - MsysControl()/MsysInquire(). Adds an action group key to action context N. N can range from M_GC_ACTION0 to M_GC_ACTION31. Inquire returns a MIL_INT.

    - Support for M_GC_ACTION_GROUP_MASK+N

      - MsysControl()/MsysInquire(). Adds an action group mask to action context N. N can range from M_GC_ACTION0 to M_GC_ACTION31. Inquire returns a MIL_INT.

    - Support for M_GC_ACTION_ACKNOWLEDGE_NUMBER+N

      - MsysControl()/MsysInquire(). Lets the GigE Vision driver know how many action command acknowledge packets will be received for action context N. If not specified, the number of calls to M_GC_ACTION_ADD_DEVICE for this action context will be used as the number of acknowledge packets expected. N can range from M_GC_ACTION0 to M_GC_ACTION31. Inquire returns a MIL_INT.

    - Support for M_GC_ACTION_EXECUTE+N

      - MsysControl(). Broadcast a GigE Vision Action command packet or scheduled action command packet containing the device key, group key, and group mask associated to action context N. N can range from M_GC_ACTION0 to M_GC_ACTION31. Control value must be M_DEFAULT. Note that the device key, group key, and group mask must have previously been programmed in each GigE Vision device associated to this action context using MdigControlFeature with "ActionDeviceKey", "ActionSelector", "ActionGroupKey", and "ActionGroupMask". See SFNC (Standard Feature Naming Convention) for details.

    - Support for M_GC_ACTION_ADD_DEVICE+N

      - MsysControl()/MsysInquire(). Associates a GigE Vision device with action context N. N can range from M_GC_ACTION0 to M_GC_ACTION31. The control value must be a valid MIL Digitizer ID. Inquire returns a MIL_ID.

    - Support for M_GC_ACTION_REMOVE_DEVICE+N

      - MsysControl()/MsysInquire(). Removes a GigE Vision device from action context N. N can range from M_GC_ACTION0 to M_GC_ACTION31. The control value must be a valid MIL Digitizer ID. Inquire returns a MIL_ID.

    - Support for M_GC_ACTION_CLEAR_DEVICES+N

      - MsysControl(). Removes all GigE Vision devices associated to action context N. N can range from M_GC_ACTION0 to M_GC_ACTION31. The control value must M_DEFAULT.

    - Support for M_GC_ACTION_TIME+N

      - MsysControl()/MsysInquire(). Associates an action time to action context N. N can range from M_GC_ACTION0 to M_GC_ACTION31. This control must be

used only if it is intended to send a scheduled action command. The time specified must be a time in the future. Inquire returns a MIL_DOUBLE. The GigE Vision devices associated to this action context must have IEEE 1588 PTP enabled using MdigControlFeature() with "GevIEEE1588", "GevIEEE1588ClockAccuracy", and "GevIEEE1588Status". See SFNC (Standard Feature Naming Convention) for details. The current time can be inquired with MdigInquire(M_GC_CAMERA_TIME_STAMP) or MdigGetHookInfo (M_GC_CAMERA_TIME_STAMP). The current time is valid only if IEEE 1588 PTP is enabled on all devices when the inquire is made.

- Additions to MdigControl()/MdigInquire()
  - Inquiring GigE Vision camera capabilities (see actiontrigger.cpp board-specific example):

    - Support for M_GC_CONTROL_PROTOCOL_CAPABILITY

      - MdigInquire(). Returns a MIL_INT that represents the control channel capabilities of the GigE Vision device. Capability bits returned can be any combination of the following:

        - M_GC_USER_DEFINED_NAME_SUPPORT

        - M_GC_SERIAL_NUMBER_SUPPORT

        - M_GC_HEARTBEAT_DISABLE_SUPPORT

        - M_GC_LINK_SPEED_REGISTER_SUPPORT

        - M_GC_PORT_AND_IP_REGISTER_SUPPORT

        - M_GC_MANIFEST_TABLE_SUPPORT

        - M_GC_TEST_DATA_SUPPORT

        - M_GC_DISCOVERY_ACK_DELAY_SUPPORT

        - M_GC_WRITABLE_DISCOVERY_ACK_DELAY_SUPPORT

        - M_GC_EXTENDED_STATUS_CODES_1_SUPPORT

        - M_GC_PRIMARY_APP_SWITCHOVER_SUPPORT

        - M_GC_UNCONDITIONAL_ACTION_SUPPORT

        - M_GC_IEEE_1588_SUPPORT

        - M_GC_EXTENDED_STATUS_CODES_2_SUPPORT

        - M_GC_SCHEDULED_ACTION_SUPPORT

        - M_GC_ACTION_SUPPORT

        - M_GC_PENDING_ACK_SUPPORT

        - M_GC_EVENT_DATA_SUPPORT

        - M_GC_EVENT_SUPPORT

        - M_GC_PACKET_RESEND_SUPPORT

        - M_GC_WRITE_MEM_SUPPORT

        - M_GC_CONCATENATION_SUPPORT

    - Support for M_GC_STREAM_PROTOCOL_CAPABILITY

- MdigInquire(). Returns a MIL_INT that represents the stream protocol capabilities of the GigE Vision device. Capability bits returned can be any combination of the following:

    o M_GC_FIREWALL_TRAVERSAL_SUPPORT

    o M_GC_LEGACY_16BIT_BLOCK_SUPPORT

- Support for M_GC_MESSAGE_PROTOCOL_CAPABILITY

    - MdigInquire(). Returns a MIL_INT that represents the message protocol capabilities of the GigE Vision device. Capability bits returned can be any combination of the following:

        o M_GC_FIREWALL_TRAVERSAL_SUPPORT

- Support for M_GC_NETWORK_INTERFACE_CAPABILITY

    - MdigInquire(). Returns a MIL_INT that represents the GigE Vision device networking interface capabilities. Capability bits returned are can be any combination of the following:

        o M_GC_PAUSE_RECEPTION_SUPPORT

        o M_GC_PAUSE_GENERATION_SUPPORT

        o M_GC_LINK_LOCAL_ADDRESS_SUPPORT

        o M_GC_DHCP_SUPPORT

        o M_GC_PERSISTENT_IP_SUPPORT

- Support for M_GC_NETWORK_INTERFACE_CONFIGURATION

    - MdigControl()/MdigInquire(). Controls the GigE Vision device's persistent network interface configuration. The Inquire returns a MIL_INT which represents the currently enabled configuration. The configuration bits that can be used can be any of the following:

        o M_GC_PAUSE_RECEPTION_SUPPORT

        o M_GC_PAUSE_GENERATION_SUPPORT

        o M_GC_LINK_LOCAL_ADDRESS_SUPPORT

        o M_GC_DHCP_SUPPORT

        o M_GC_PERSISTENT_IP_SUPPORT

- Support for M_GC_PHYSICAL_LINK_CONFIGURATION_CAPABILITY

    - MdigInquire(). Returns a MIL_INT that represents the GigE Vision device physical link configuration capabilities. Capability bits returned can be any combination of the following:

        o M_GC_SINGLE_LINK_SUPPORT

        o M_GC_MULTIPLE_LINK_SUPPORT

        o M_GC_STATIC_LINK_AGGREGATION_SUPPORT

        o M_GC_DYNAMIC_LINK_AGGREGATION_SUPPORT

- Support for M_GC_STREAM_CHANNEL_CAPABILITY

- MdigInquire(). Returns a MIL_INT that represents the stream channel capabilities of the GigE Vision device. Capability bits returned can be any combination of the following:

    o M_GC_BIG_AND_LITTLE_ENDIAN_SUPPORT

    o M_GC_IP_REASSEMBLY_SUPPORT

    o M_GC_MULTI_ZONE_SUPPORT

    o M_GC_PACKET_RESEND_OPTION_SUPPORT

    o M_GC_ALL_IN_SUPPORT

    o M_GC_UNCONDITIONAL_STREAMING_SUPPORT

    o M_GC_EXTENDED_CHUNK_DATA_SUPPORT

- Additions to MdigInquireFeature()/MdigControlFeature()
    o The FeatureType parameter has been changed to UserVarType. This was done to simplify writing code with MdigControl/InquireFeature(). UserVarType must always reflect the type of the pointer passed to the UserVarPtr parameter. Legacy code is transparently supported, but we recommend you update your code. Note that M_TYPE_REGISTER now becomes M_TYPE_UINT8, M_TYPE_ENUMERATION now becomes M_TYPE_INT64 or M_TYPE_STRING, and M_TYPE_COMMAND now becomes M_DEFAULT. Data type conversions are made, whenever possible, in cases where the feature's "native" data type is different than the UserVarType supplied. Regardless of a feature's "native" data type it can always be read as a string. See Board-specific examples for details.

      The following is a list of example calls using the new UserVatType:

      - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Width"),
        M_TYPE_INT64, &Int64Var)

      - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Gain"),
        M_TYPE_DOUBLE, &DoubleVar)

      - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("ReverseX"),
        M_TYPE_BOOLEAN, &BoolVar)

      - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("PixelFormat"),
        M_TYPE_STRING, MIL_TEXT("Mono8"))

      - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("LUTValueAll"),
        M_TYPE_UINT8, Uint8Array)

      - MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("AcquisitionStart"),
        M_DEFAULT, M_NULL)

      - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Width"),
        M_TYPE_INT64, &Int64Var)

      - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Gain"),
        M_TYPE_DOUBLE, &DoubleVar)

      - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("ReverseX"),
        M_TYPE_BOOLEAN, &BoolVar)

      - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE + M_STRING_SIZE, MIL_TEXT
        ("PixelFormat"), M_TYPE_MIL_INT, &MilIntVar)

      - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("PixelFormat"),
        M_TYPE_STRING, MilTextCharArray)

- MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("LUTValueAll"), M_TYPE_UINT8, Uint8Array)

  o M_FEATURE_USER_ARRAY_SIZE() can now be used with MdigInquireFeature when the data type returned is a string or an array of bytes (register). The M_FEATURE_USER_ARRAY_SIZE() macro is used to pass the size of the user-allocated buffer passed to MdigInquireFeature's UserVarPtr parameter. M_FEATURE_USER_ARRAY_SIZE() is passed using the UserVarType parameter. See MilGige board specific example for sample usage.

    The following is a list of example calls using M_FEATURE_USER_ARRAY_SIZE():

    - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("PixelFormat"), M_TYPE_STRING + M_FEATURE_USER_ARRAY_SIZE(N), MilTextCharArray); N being equal to the number of MIL_TEXT_CHAR in the MilTextCharArray.

    - MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("LUTValueAll"), M_TYPE_UINT8 + M_FEATURE_USER_ARRAY_SIZE(N), Uint8Array); N being equal to the number of Uint8 in the Uint8Array.

  o M_FEATURE_ENUM_ENTRY_DISPLAY_NAME can now be used to inquire possible enumeration string entry to use for display purposes. See M_FEATURE_ENUM_ENTRY_NAME in the MIL documentation.

  o M_FEATURE_VALUE_AS_STRING is now deprecated.

    - To read a feature's value as a string and get the required string length use:

      • MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE + M_STRING_SIZE, MIL_TEXT("Width"), M_TYPE_MIL_INT, &MilIntVar);

    - To read a feature's value as a string use:

      • MdigInquireFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Width"), M_TYPE_STRING+M_FEATURE_USER_ARRAY_SIZE(ArraySize), MilTextCharArray);

    - To write a feature's value from a string use:

      • MdigControlFeature(MilDigitizer, M_FEATURE_VALUE, MIL_TEXT("Width"), M_TYPE_STRING, MIL_TEXT("1024"));

  o M_FEATURE_CHANGE_HOOK. Identifies the specified FeatureName to trigger the M_FEATURE_CHANGE hook callback. You must be hooked to the M_FEATURE_CHANGE hook type using MdigHookFunction().

- Additions to MdigHookFunction()
  o Hooking to a GenICam feature change event (see enumfeatures.cpp board-specific example):

    - M_GC_FEATURE_CHANGE can be used as a hook type. The hook is triggered when a GenICam feature gets invalidated. This usually occurs when a feature or a dependent feature is written.

- Additions to MdigGetHookInfo()
  o Hooking to a GenICam feature change event (see enumfeatures.cpp board-specific example):

    - M_GC_FEATURE_CHANGE_NAME can be used from a M_GC_FEATURE_CHANGE hook function. The function returns the name of the GenICam feature that triggered the hook. UserVarPtr must point to a user allocated array of type MIL_TEXT_CHAR.

    - M_GC_FEATURE_CHANGE_NAME_SIZE can be used from a M_GC_FEATURE_CHANGE hook function. The function returns the number of characters in the string returned by M_GC_FEATURE_CHANGE_NAME. UserVarPtr must point to a MIL_INT.

1.12.4 Improvements

- Fixed M_GC_MANIFEST_ENTRY() macro; it could not receive a variable as a parameter. Only integer literals were working.

- Intellicam now accepts adding GenICam commands to DCF in recording mode.

1.12.5 Behavioral changes

- A loop of MdigGrab() called in M_SYNCHRONOUS mode resulted in unnecessary packet resend requests. This has been fixed in MIL 10 Update 25.

- The MtxGigeService.exe is now set to automatically restart after a crash.

- CaptureAssistant no longer requires administrative privileges at start-up.

- CaptureAssistant is now aware of trigger mode changes done through the Feature Browser.

- Camera features are no longer written when calling MdigAlloc() with an empty DCF.

- Calling MdigGetHookInfo(M_GC_FRAME_BLOCK_ID) with a GigE Vision 2.0 will return the camera's 64-bit frame id counter. GigE Vision 1.x devices still return the 16-bit frame id counter.

1.12.6 Bug fixes

- Fixed issue in Intellicam where some DCF files were not saved to disk when the file did not end with expected characters.

- Boolean features were not correctly saved with DCF set in recording mode.

- Fixed blue-screen when low memory resource conditions occurred.

- Fixed deadlock that occurred when MdigProcess(M_STOP) was called from the MdigProcess callback function.

- Fixed blue-screen that occurred when changing Ethernet controller parameters while grabbing (under Window 7 and Windows 8).

- Fixed issue in Intellicam when dumping all camera features to DCF (32-bit version only). The resulting DCF was not valid.

- Fixed crash that occurred with MdigControlFeature and MdigInquireFeature with NULL feature names.

- Fixed issue with MdigControl(M_GC_PACKET_SIZE); the internal GenICam feature associated was not updated, resulting in an invalid value returned by MdigInquireFeature with the "GevSCPSPacketSize" feature.

- Fixed handle leak when MsysHookFunction(M_CAMERA_PRESENT) was not unhooked.

- Fixed MsysGetHookInfo that returns strings (MIL_TEXT_PTR) when used with DMIL.

- Fixed packet size negotiation when MdigAlloc() was called with M_GC_MULTICAST_MASTER.

1.13 What's new in MIL 10

1.13.1 Standards compliance

- Support for GenICam 2.3.1.

- Support for Standard Feature Naming Convention (SFNC) version 1.3.

● Support for Microsoft® Network Driver Interface Specification 6.20.

1.13.2 New features summary

● Matrox GigE Vision driver
  ○ Support for GigE Vision multicast mode in MIL.

  ○ Support for GigE Vision multicast mode in the Matrox Capture Assistant.

  ○ Accelerated MdigAlloc() by avoiding the XML description file download, when possible.

  ○ Overhaul of the MdigControlFeature() and MdigInquireFeature(). See the MIL Online help for details.

  ○ The Matrox GigE Vision filter driver now complies with the Microsoft® NDIS 6.20 specification when installed on Windows 7 and Windows 8.

  ○ Implemented MIL's new I/O specification.

● New gigevision_multicast_monitor.dcf
  ○ The gigevision_multicast_monitor.dcf can be used to allocate a digitizer in M_GC_MULTICAST_MONITOR mode. This DCF includes stream and message channel parameters that must be set to the multicast IP address and UDP port number for each of the stream and (optionally) the message channel. It also includes parameters for specifying the expected image Width, and image Height, as well as the expected PixelFormat that will be transmitted by the GigE Vision device operating in Multicast mode.

● Board-specific examples
  ○ New multicastmaster.cpp example, located in examples\board-specific\gigevision\C++\multicastmaster. This can be used to allocate a GigE Vision device in multicast mode with control access.

  ○ New multicastslave.cpp example, located in examples\board-specific\gigevision\C++\multicastslave. This can be used to allocate a GigE Vision device in multicast mode without control (read only access mode). The example also adapts to changes made by the master process.

  ○ New multicastmonitor.cpp example, located in examples\board-specific\gigevision\C++\multicastmonitor. The multicast monitor mode can be used in cases where the multicast master allocates the GigE Vision device in exclusive mode. When a GigE Vision device is allocated in exclusive mode, secondary applications will no longer have read access to the device, making the multicastslave example unusable. In this case, the multicastmonitor example can be used. Manual configuration is required for the multicast IP address and UDP port used.

  ○ New enumfeatures.cpp example, located in examples\board-specific\gigevision\C++\enumfeatures. This can be used to enumerate all features present in your GenICam compliant camera. It can also be used to query various feature properties.

  ○ New chunkmode.cpp example, located in examples\board-specific\gigevision\C++\chunkmode. This can be used to enable chunk mode. With chunk mode enabled, additionnal data will get streamed along with the image to the host. MIL will automatically extract the chunk information and update the corresponding chunk features and extract the image data.

1.13.3 Additions to the command reference

● Additions to MdigAlloc()
  ○ The InitFlag parameter now supports the following settings:

    - Support for M_GC_XML_FORCE_DOWNLOAD.

      Forces the MIL driver to download the GigE Vision device's XML description file at MdigAlloc(). If this setting is omitted, the cached version present in the MIL XML repository is used instead.

    - Support for M_GC_MULTICAST_MASTER

Specifies that the allocated device will be the master digitizer in a multicasting master-slave relationship. When this setting is used, the GigE Vision device's stream and message channel (if present) will be programmed to use a destination IPv4 address in the multicast address range (from 224.0.0.0 to 239.255.255.255). By default, MIL chooses an administratively scoped IP multicast address in the range 239.0.0.0 to 239.255.255.255. IP multicast also requires a network designed to deliver a multicast service. See the IP multicast section of the Matrox Capture Assistant help for more details.

- Support for M_GC_MULTICAST_SLAVE

    Specifies that the allocated device will be a slave digitizer in a multicasting master-slave relationship. See the IP multicast section of the Matrox Capture Assistant help for more details.

- Support for M_GC_MULTICAST_MONITOR

    Specifies that the allocated device will be a monitor digitizer in a multicasting master-slave relationship.

- Additions to MdigControl()/MdigInquire()
  - Support for M_GC_STREAM_CHANNEL_MULTICAST_ADDRESS_STRING

    - MdigControl()/MdigInquire(). Controls the multicast address used by the device to stream image data. The device must have been allocated with the M_GC_MULTICAST_MASTER InitFlag. Supported control values are any valid IPv4 multicast address in the range 224.0.0.0 to 239.255.255.255. The inquire returns a MIL_TEXT_CHAR string.

  - Support for M_GC_MESSAGE_CHANNEL_MULTICAST_ADDRESS_STRING

    - MdigControl()/MdigInquire(). Controls the multicast address used by the device to send message channel event data. The device must have been allocated with th M_GC_MULTICAST_MASTER InitFlag. Supported control values are any valid IPv4 multicast address in the range 224.0.0.0 to 239.255.255.255. The inquire returns a MIL_TEXT_CHAR string.

  - Support for M_GC_MULTICAST_MASTER_CONNECTED

    - MdigInquire(). Inquire returns a MIL_INT. Returns M_TRUE if a master digitizer is connected to the device.

  - Support for Support for M_GC_STREAM_PORT.

    - MdigControl()/MdigInquire(). Controls the multicast UDP port number used for the stream channel. The device must have been allocated with the M_GC_MULTICAST_MASTER InitFlag. The inquire returns a MIL_INT.

  - Support for M_GC_MESSAGE_PORT.

    - MdigControl()/MdigInquire(). Controls the multicast UDP port number used for the message channel. The device must have been allocated with the M_GC_MULTICAST_MASTER InitFlag. The inquire returns a MIL_INT.

  - Support for M_GC_NUMBER_OF_STREAM_CHANNELS.

    - MdigControl()/MdigInquire(). Specifies the number of stream channels on the GigE Vision device. This control can only be used when the digitizer is allocated with the M_GC_MULTICAST_MONITOR InitFlag. The inquire can be used to query the number of stream channels. The inquire can always be used. The inquire returns a MIL_INT.

  - Support for M_GC_PIXEL_FORMAT.

- MdigControl()/MdigInquire(). Controls the camera's pixel format. The pixel format value must be a valid GigE Vision pixel format. See GVSP_Pixel_Formats.h for a list of supported values. The inquire returns a MIL_INT.

1.13.4 Improvements

- Resolved issue where missing the data leader packet lead to a missed frame.

1.13.5 Behavioral changes

- Calling MdigAlloc() on a device that supports the GigE Vision manifest table no longer downloads the device's XML description file every time. Instead the cached version, if present in the MIL XML repository, is used. If the device does not support the GigE Vision manifest table, then the XML description file is downloaded every time. To force the download, use M_GC_XML_FORCE_DOWNLOAD with MdigAlloc().

- Missing the first packet of a frame caused a missed frame to be reported even if all other packets of the frame were grabbed. This has been fixed by changing the default behavior of the driver's packet resend engine. The driver is now allowed, by default, to issue packet resend requests for the first 5 packets of a frame (if missed).

1.13.6 Bug fixes

- Fixed a blue-screen that occurred when MdigAlloc() was called with M_GC_MULTICAST_SLAVE and the master digitizer had not yet been allocated (reproducible only with DriverVerifier enabled on the mtxgige.sys driver).

- Fixed crash that occurred with non standard devices that had UserDefinedNames greater than 16 characters.

- Fixed a blue-screen that occurred on an Intel 10 Gbps NIC.

- Fixed bad image reconstruction when the Microsoft NDIS_PACKET was split into multiple MDLs. Occurred on 10 Gbps NICs.

- An internal GenICam error has been fixed when using the GigE Vision message channel.

- A crash has been fixed at MdigFree() with some specific cameras.

- Fixed issue when using MdigControl/Inquire() with M_GC_CAMERA_TIME_STAMP.

- Fixed issue when using MdigGetHookInfo() with M_MODIFIED_BUFFER+M_BUFFER_ID this always returned 0 when called from M_GRAB_XXX hooks. The MdigProcess() hook callback was not affected by this issue.

- Fixed issue with MtxGigeService on systems with more than 16 NICs installed.

1.13.7 Define name changes

- M_GC_SLAVE is now M_GC_MULTICAST_SLAVE.

- M_GC_PIXELFORMAT_SWITCHING is now M_GC_PIXEL_FORMAT_SWITCHING.

- M_SOURCE_BUFFER_FORMAT is now M_SOURCE_DATA_FORMAT.

- M_GC_IP_ADDRESS is now M_GC_IP_ADDRESS_STRING

- M_GC_NIC_IP_ADDRESS is now M_GC_NIC_IP_ADDRESS_STRING

- M_GC_MAC_ADDRESS is now M_GC_MAC_ADDRESS_STRING

- M_GC_NIC_MAC_ADDRESS is now M_GC_NIC_MAC_ADDRESS_STRING

1.14 What's new in MIL 9 Update 52

1.14.1 Standards compliance

- Support for GenICam 2.3.1.

- Support for Standard Feature Naming Convention (SFNC) version 1.3.

1.14.2 New features summary

- Matrox Capture Assistant (for GigE Vision devices)
    - Added acquisition section.

    - Added a control section.

    - Added new views.

    - Added statistics for NICs.

    - Added new set of DCFs.

- Matrox Intellicam
    - The device feature browser is now in its own window.

    - Can now mark features in a DCF as optional. Will not generate an error at MdigAlloc() if feature does not exist in the target GigE Vision device.

    - The Direct Feature Control tab now includes several drop-down list boxes to more robustly specify the value and data type of the feature entry selected.

- Matrox device feature browser
    - Can view real-time chunk data.

    - Added polling support for GenICam features that support polling.

    - New combo box allows filtering of features according to user level (beginner, expert, guru).

    - Added feature search menu.

- Matrox GigE Vision driver
    - Support for GigE Vision Chunk mode.

    - Improved acquisition robustness through the addition of a packet cache. System conditions could cause missed frames when packet resend requests were not received in time. This caused one or more subsequent frames to be missed. These frames now get stored temporarily in a packet cache before being delivered to the user's grab buffer.

    - Tweaked use case where packet resend requests transmitted by the driver were delayed.

    - Added new pixel format support:

        - YUV411Packed

        - YUV444Packed

        - RGB10Packed

        - RGB12Packed

    - Packet resend parameters are now automatically adjusted for link aggregated devices. When the detected link speed is greater than 1 Gbps, the MIL driver assumes the attached device is link aggregated and adjusts packet resend parameters automatically at MdigAlloc().

- The default maximum negotiated packet size is now 9014 bytes; this has been increased from 6144 bytes.

- The automatic packet size negotiation process now takes into account GenICam's minimum, maximum and increment values for the packet size register (GevSCPSPacketSize feature). This allows for successful packet size negotiation on devices which restrict accepted packet size values.

- The M_SYSTEM_GIGE_VISION system type can now use the SoliosGigE NICs by default. This was blocked in previous releases.

- New InitFlags supported for MdigAlloc().

- New modes for MdigControlFeature/MdigInquireFeature().

- Added gigevision_max_sensor_size.dcf. This DCF allows a digitizer to be allocated using the device's largest possible width and height values, while keeping the ROI values to a minimum.

- Added new set of DCFs. DCFs that have the _continuous suffix in the file name will force the GigE Vision device in continuous mode, disabling triggers if any.

- GigE Vision-specific examples
  - mtxgige.cpp now reads and prints LUT information if present in a GigE Vision device.

1.14.3 Additions to command reference

- Additions to MdigAlloc
  - The InitFlag parameter can now have the following flags set:

    - Support for M_GC_XML_DOWNLOAD_SKIP. Prevents the MIL driver from downloading the GigE Vision device's XML description file at MdigAlloc(). The cached version present in the MIL XML repository is used instead. The M_GC_XML_DOWNLOAD_SKIP flag is ignored if:

      - The XML description file is not present in the MIL XML repository.

      - The XML description file name and/or file size, as reported by the GigE Vision device, has changed.

  - Support for M_GC_PACKET_SIZE_NEGOTIATION_SKIP.

    Prevents the MIL driver from negotiating a packet size with the GigE Vision device at MdigAlloc(). Instead, the GigE Vision device's packet size register is read and this value is used by the MIL driver. If the read packet size is too small, acquisition reliability will suffer. Using this scheme, the packet size can be forced by storing it in the DCF or by using MdigControl with M_GC_PACKET_SIZE. In either case, acquisition reliability will be affected if the forced packet size is too small or if the packet size is not supported by the network infrastructure.

- Additions to MdigControl/MdigInquire
  - Support for M_GC_FEATURE_POLLING.

    - MdigControl/Inquire. Inquire returns a MIL_INT. Controls the feature browser's polling thread. Your GigE Vision device can define polling values for features that can change without the user's input. An example of this is the "DeviceTemperature" feature. The polling thread can be used when the device's feature browser is displayed. Its role is to poll features that specify a polling period. Supported control values are:

      - M_DEFAULT: Same as M_ENABLE.

      - M_ENABLE: Enables the polling thread. Camera features that report polling support will be read periodically.

      - M_DISABLE: Disables the polling thread.

- o Support for M_DIGITIZER_INTERNAL_BUFFERS_NUM.

  - MdigControl/Inquire. Inquire returns a MIL_INT. Controls the number of internal grab buffers allocated and used when the MIL driver cannot directly grab into the user's buffer. Internal grab buffers are needed when the GenICam chunk mode is active. They are also needed when the grab buffer's format is not compatible with the camera's current pixel format. Supported control values are:

    - M_DEFAULT: Five internal grab buffers will be allocated.

    - A positive integer between 2 and 1024 representing the number of internal grab buffers to use. An error will get generated if the amount of non paged memory available is insufficient to allocate the buffers.

- o Support for M_SOURCE_BUFFER_FORMAT.

  - MdigInquire. Inquire returns a MIL_INT64. Returns a value representing the MIL buffer format that is compatible with the camera's current PixelFormat. Useful for allocating grab buffers and avoiding costly color space conversions, Ex:

    - Mono8 PixelFormat will return M_MONO8+M_PLANAR

    - Mono10 PixelFormat will return M_MONO16+M_PLANAR

    - BayerBG8 PixelFormat will return M_YUV16_YUYV+M_PACKED

    - BayerBG12Packed PixelFormat will return M_RGB48+M_PACKED

    - YUYVPacked PixelFormat will return M_YUV16_YUYV+M_PACKED...

- o Support for M_GC_TOTAL_PACKET_CACHE_HITS.

  - MdigInquire. Inquire returns a MIL_INT. Represents the total number of packets that passed through the packet cache.

- o Support for M_GC_TOTAL_FRAME_CACHE_HITS.

  - MdigInquire. Inquire returns a MIL_INT. Represents the total number of partial or complete frames that passed through the packet cache.

- o Support for M_GC_NIC_FRIENDLY_NAME.

  - MdigInquire. Inquire returns a MIL_TEXT_CHAR string. Returns the network adapter's friendly name connected to your GigE Vision device.

- o Support for M_GC_FRAME_STATUS_CODE.

  - MdigInquire. Inquire returns a MIL_INT. Returns the status code from the last image acquired. The status code is extracted from the GigE Vision device's stream packets.

- Additions to MdigControlFeature/MdigInquireFeature
  - o Support for M_TYPE_REGISTER_FROM_STRING.

    - MdigControlFeature/MdigInquireFeature. Supports reading from and writing to IRegister feature types. FeatureValuePtr must be a MIL_TEXT_PTR.

    - MdigInquireFeature supports the M_LENGTH inquire flag. When used, it returns a MIL_INT representing the string length required to read the IRegister value.

  - o Support for M_TYPE_VALUE_FROM_STRING.

    - MdigControlFeature/MdigInquireFeature. Supports reading from and writing to any feature type using strings. FeatureValuePtr must be a MIL_TEXT_PTR.

- MdigInquireFeature supports the M_LENGTH inquire flag. When used, it returns a MIL_INT representing the sting length required to read the feature value as a string.

- Additions to MdigGetHookInfo
  - Support for M_GC_FRAME_STATUS_CODE.

    - Returns a MIL_INT. Returns the status code from the last image acquired. The status code is extracted from the GigE Vision device's stream packets.

1.14.4 GigE Vision utilities

- Matrox Capture Assistant
  - Added acquisition section.

    - Monoshot acquisition.

    - Continuous acquisition.

    - Buffering control (specify among 5, 10, 20, 50 or 100 buffers to reduce missed frames).

    - Inter-packet delay calculator.

    - Save image to file.

    - MIL display controls.

    - Support of live plug/unplug of GigE Vision devices.

  - Added a control section.

    - Can display the device's feature browser.

    - Can control MIL digitizer properties.

    - Bayer conversion.

    - White balance.

    - Triggers.

    - Packet-resend parameters.

  - Added new views.

    - A details view.

    - A tree view that shows the relationship between a Gige Vision device and its parent NIC.

  - Added statistics for NICs.

  - Icon overlays graphically indicate device status information.

  - Owner process of device is displayed as a tooltip.

- Matrox device feature browser
  - Made performance optimizations.

  - Can view real-time chunk data.

  - Added polling support for GenICam features that support polling.

  - Added combo box that allows filtering of features according to user level.

  - Added check box for feature polling support.

- o Added tooltips on all nodes.

- o Fixed tooltips when used in Intellicam.

- o Added feature search menu.

1.14.5 Improvements

- A packet cache is now maintained by the MIL driver to alleviate missed frame issues.

- The packet resend engine has been optimized in use cases where packet resend requests were delayed.

- The packet resend engine's parameters are now automatically adjusted for link aggregated GigE Vision devices. Adjustment occurs when the detected Ethernet link speed is greater than 1 Gbps. Ethernet link speed detection under Windows XP requires the MtxGigeService to be running.

- Automatic packet size negotiation now takes into account the GigE Vision device's limitations regarding Minimum, Maximum and Increment values.

- Optimized the device feature browser.

- Added new pixel format support.

- GenICam's chunk mode is now supported.

- The Matrox Capture Assistant now has an acquisition section.

1.14.6 Behavioral changes

- The trigger mode of GigE Vision devices are no longer reset at MdigAlloc(). If you wish to force your GigE Vision device to continuous mode at MdigAlloc() use the DCFs with the _continuous suffix.

- Five internal buffers are now allocated per digitizer up from three to resolve missed frame issues. These are used when the MIL driver cannot grab directly in the user's buffers.

- When internal buffers run out and the MIL driver cannot grab directly in the user's buffers because of format conversion issues (e.g. chunk mode is used or the camera's PixelFormat is not compatible with the buffer's format), then, if conditions permit, the buffers passed to MdigProcess can be used by the MIL driver as a temporary storage location for raw image data. This raw data will then be converted to the proper format in a subsequent operation. It is guaranteed that the data in the buffer signaled as grabbed in the MdigProcess callback has been correctly formatted and is available for use by the application.

1.14.7 Bug fixes

- Fixed handle leak related to the MdigProcess grab monitor when called in a loop with high frame rate cameras.

- Fixed UI bugs in feature browser.

- Fixed memory and handle leaks in feature browser.

- Fixed handle leak that occurred during MdigProcess M_START or M_SEQUENCE / M_STOP cycles.

- Fixed crash in Matrox Capture Assistant that occurs right after logon on Windows XP Embedded systems.

- Fixed issue with MtxGigeService that occurs during uninstallation/installation cycles. Sometimes the uninstallation does not complete normally (requires a reboot) and prevents the new installation from completing properly.

- Fixed unreported errors at MdigAlloc() when loading from a DCF containing invalid features.

- Fixed issue with MdigGetHookInfo called from an M_GRAB_END hook.

- Missing installation of XMLLite is now reported through Matrox Capture Assistant or through a MIL error at MsysAlloc().

- Fixed crash that occurred when GenICam was not properly installed.

- Fixed bug with MdigInquire(M_GC_NIC_MAC_ADDRESS). It did not report the correct address if the GigE Vision device is reconnected to another host NIC port.

- Fixed issue at MdigAlloc() where the wrong error message is reported. Occurs when a GigE Vision device is unplugged and MdigAlloc() is called before the mtxgigeservice has time to update its cache. The error reported was "The camera has denied the application access to its configuration registers"; it should have been "GigE Vision device not found".

- Fixed issue at MdigAlloc() on systems with less than 512 MB of RAM and no page file (virtual memory).

- Fixed crash that occurs during an internal GenICam exception that is mishandled.

- Fixed issue with MdigInquire(M_TYPE). Inquire returned 16 for Mono10 and Mono12 pixel formats. It now returns 10 and 12 respectively. MdigInquire(M_SIZE_BIT) will return 16 for these pixel types.

- Fixed issue in mtxgigefilter.sys. GigE Vision stream packets were not filtered out when no grabs were pending and the camera was streaming images. This resulted in CPU spikes.

- Fixed issue in Matrox Capture Assistant. ToolStrip buttons did not respond to mouse clicks if the Matrox Capture Assistant did not have input focus.

- Fixed exception in mtxgigeservice that occurs at boot up right after the login on Windows XP Embedded systems.

- Fixed handle leak seen during MdigAlloc()/MdigFree() cycles.

1.15 What's new in MIL 9 Update 34

1.15.1 Standards compliance

- Support for GenICam 2.1.1.

- Support for Standard Feature Naming Convention (SFNC) version 1.3.

1.15.2 New features summary

- Matrox GigE Vision and Configuration Assistant:
    - Added option in System Configuration page to prevent the driver from automatically resolving IP configurations conflicts.

    - Added option in System Configuration page to change the maximum number of devices that can be used, default has been increased from 16 to 32.

    - Added a reset checkbox to Persistent IP Configuration section. Forces device to reboot and use the newly written IP settings.

    - Added button to spawn device feature browser.

    - Added option to allow the M_SYSTEM_GIGE_VISION system to acquire from cameras connected to a SoliosGigE.

- Matrox Intellicam
    - Simplified DCF dialog.

    - Overview page now lists all cameras with more descriptive information.

    - Direct Feature Control now guarantees that all selectors are written to the DCF when a selectee feature is written through the feature browser.

- o New option that allows writing all streamable features to the DCF with a single button click. The camera needs to support streamable features for this option to work.

- o Added check-box in the System Preferences dialog (Options -> Preferences menu) to disable the digitizer controller dialog. Avoids bug in Toshiba TELI GiantDragon family of cameras.

- Matrox Device Feature Browser
  - o Value column now prints the "Display Name" of a feature value.

- GigE Vision-specific examples
  - o New cameraevents GigE Vision-specific example. Demonstrates the use of GigE Vision message channel events with MIL.

  - o Updated milgige GigE Vision-specific example. Now prints important camera features; support for the different triggering schemes supported by the Standard Features Naming Convention.

  - o Updated packetdelay GigE Vision-specific example. Resolved issue about algorithm not converging. Algorithm now runs faster and has the option to compute inter-packet delay for all PixelFormats supported by the camera.

  - o Updated multicamera GigE Vision-specific example. Added per NIC statistics. Fixed race condition between initial digitizer allocation and camera present hook.

- Support of more that 16 GigE Vision devices. The total number of devices that can be used can now be controlled through GigE Assistant's "System Configuration" page.

- Implemented new mechanism in mtxgigefilter to solve missing packets in SingleFrame or MultiFrame AcquisitionMode when packets are missing on the last frame of the stream.

- Multiple versions of GenICam can now be supported simultaneously.

- MdigHalt now automatically issues an M_GC_STREAMING_STOP (AcquisitionStop) command to the camera.

- Enhanced error reporting when MdigControlFeature is called with the wrong FeatureName and FeatureDataType combination.

1.15.3 Additions to command reference

- Additions to MdigControl/MdigInquire
  - o Support for M_GC_STREAMING_MODE.

    - MdigControl/Inquire. Inquire returns a MIL_INT. Controls the camera's image stream activation mechanism. Supported control values are:

      - M_DEFAULT: Lets the MIL driver be in control when the camera's image stream is started/stopped.

      - M_AUTOMATIC: Same as M_DEFAULT.

      - M_MANUAL: Lets the user be in control when the camera's image stream is started/stopped.

  - o Support for M_GC_STREAM_START

    - MdigControl. Starts the camera's image stream. M_GC_STREAMING_MODE must be set to M_MANUAL. Supported control values are:

      - M_DEFAULT.

  - o Support for M_GC_STREAM_STOP

    - MdigControl. Stops the camera's image stream. Supported control values are:

- M_DEFAULT.

    o Support for M_GC_NUMBER_OF_STREAM_CHANNELS

       - MdigInquire. Inquire returns a MIL_INT. Returns the number of stream channels supported by the GigE Vision device.

- Additions to MdigControlFeature/MdigInquireFeature
  - MdigControlFeature now accepts a new ControlFlag parameter. Supported ControlFlags are:

    - M_DEFAULT: When a feature is written the driver will block the call until the camera acknowledges the operation as complete.

    - M_SYNCHRONOUS: Same as M_DEFAULT.

    - M_ASYNCHRONOUS: When a feature is written, the driver will not request an acknowledgment from the camera and will return control to the user immediately. Can be used to speed-up programming of the device. This option effectively disables the flow control between the camera and the driver. Care must be taken to not "overflow" the device with data.

  - MdigInquireFeature now accepts new ControlFlag parameters. Supported control flags are:

    - M_ELEMENT_COUNT: Returns a MIL_INT. When used with M_TYPE_STRING_ENUMERATION or M_TYPE_INTEGER_ENUMERATION FeatureDataType, returns the number of enumeration entries in the enumeration.

    - M_ELEMENT+M_LENGTH+N: Returns a MIL_INT. When used with M_TYPE_STRING_ENUMERATION FeatureDataType, returns the number of characters of the Nth enumeration entry.

    - M_ELEMENT+N: Returns a MIL_TEXT_CHAR string. When used with M_TYPE_STRING_ENUMERATION FeatureDataType, returns the name of the Nth enumeration entry.

    - M_ELEMENT+N: Returns a MIL_INT64. When used with M_TYPE_INTEGER_ENUMERATION FeatureDataType, returns the numerical value of the Nth element.

1.15.4 GigE Vision utilities

1.15.5 Improvements

- Implemented a timer object in the packet resend mechanism. Useful when the camera's AcquisitionMode is not Continuous.

- Fixed performance issue in packet resend mechanism when the last packets of a frame were missed.

- Performance optimizations done to the GigE Vision stream filter engine (mtxgigefilter).

- Clarified error messages when MdigControlFeature/MdigInquireFeature is used with the wrong FeatureDataType parameter.

- Support for non-streaming GigE Vision devices.

- A discovery is now performed at MsysAlloc when the MtxGigeService device cache is empty.

- MdigInquire(M_PROCESS_FRAME_RATE) is now much more precise.

1.15.6 Behavioral change

- The MtxGigeService no longer waits 120 seconds before performing the first discovery after the OS boot.

- It is now possible to grab with M_SYSTEM_GIGE_VISION through the SoliosGigE to support packet resend.

- The MtxGigeService no longer performs MdigAlloc calls upon initial device discovery. Avoids side effect when multiple MtxGigeServices are running on the same sub-network.

- A discovery command is sent at MsysAlloc when the MtxGigeService's camera cache is empty.

1.15.7 Bug fixes

- Fixed issues with MdigControlFeature/MdigInquireFeature that control string based features when used with distributed MIL.

- Fixed issue with MdigProcess which would not stop under certain conditions.

- Fixed issue where an internal error was not reported to the user.

- Fixed issue with MdigProcess when camera's AcquisitionMode is set to MultiFrame. The camera's stream was started too soon resulting in the possibility of missing the first frame of the sequence.

- Fixed issue with MtxGigeService where it prevented the OS from performing a shutdown operation.

- Fixed issue with MtxGigeService where an extra discovery command was sent.

- Fixed issue with MtxGigeService that would fail to start promptly at OS boot-up.

- Fixed memory leak in MtxGigeService.

- Fixed issue with Matrox GigE Vision and Configuration Assistant where persistent data was written to the wrong device.

- Fixed bug related to MdigControl(M_GRAB_EXPOSURE_...) controls.

- Fixed blue-screen that occurred in the kernel socket library. Occurred on a failed socket allocation.

- Fixed MdigProcess frame counter issue upon camera disconnection.

- Fixed issue in feature browser where some features were not being displayed. Occurred if feature was marked as not implemented during initial node tree walkthrough.

- Fixed blue-screen that occurred when calling MdigGrabContinuous with a buffer not selected to the display. Blue-screen occurred when the camera was disconnected while grabbing.

- Fixed camera bandwidth statistics issue in multicamera GigE Vision specific example.

- Fixed issue with Basler Ace in YUV422_YUYV_Packed pixel format.

- Fixed heap corruption issue in MdigInquireFeature when FeatureDataType was set to M_TYPE_STRING or M_TYPE_STRING_ENUMERATION.

- Fixed issue in Matrox GigE Vision and Configuration Assistant where camera list is sometimes not displayed at application startup (need to click on the Perform Discovery button).

- Fixed issue with SVS-VISTEK Link Aggregated cameras. Camera streams its image top down on one interface and bottom up on the other; we were only capturing half of the image.

- Fixed issue in feature browser where some features were not being displayed when they are initially tagged as "Not Implemented".

1.16 What's new in MIL 9 Update 9

1.16.1 Standards compliance

- Support for new GigE Vision 1.1 features.
  - Firewall traversal (if supported by camera).

  - New pixel formats: Mono14, Bayerxx10Packed, Bayerxx12Packed, Bayerxx16 and YUYVPacked.

  - Manifest table.

- Support for GenICam 2.0.1.

- Support for Standard Feature Naming Convention (SFNC) version 1.3.

1.16.2 New features summary

- Matrox GigE Vision and Configuration Assistant:
  - Added GigE Vision device IP configuration tool. Allows to control Persistent IP and DHCP settings in the camera.

  - Added detection of firewall presence. Note that firewalls can prevent device discovery as well as prevent host-camera packet size negotiation from taking place.

  - Added ability to change camera PacketSize.

  - Added ability to change inter-packet delay.

  - Added GigE Vision capabilities window. This lists the selected camera's supported GigE Vision features.

  - Added GigE Vision statistics window. This lists various statistics (e.g. bandwidth, missed packets, recovered packets, missed frames, corrupted frames...) for a given GigE Vision camera.

  - Added System Configuration tab. This controls various GigE Vision driver parameters.

  - Added Troubleshooting tab. This allows for the recovery of inaccessible GigE Vision cameras.

- Re-engineered the packet re-transmission algorithm in the driver. This removes limitations in the previous algorithm.

- Support of camera connect / disconnect events (see multicamera.cpp example for more details).
  - Newly attached cameras can now be allocated post MsysAlloc.

  - Allocated digitizers are seamlessly re-initialized when an allocated camera is disconnected / re-connected.

  - New multicamera.cpp example, located in examples\board-specific\gigevision\multicamera\. This can be used to detect and handle camera connection / disconnection events.

  - Updated example since last release of this update to reflect changes in mtxgigeservice device persistence modifications (see the mtxgigeservice section in this document).

- Support of GigE Vision message channel through MdigHookFunction with M_GC_EVENT hook types.

- Support of MIL User-bits on GigE Vision devices through SFNC compliant feature names.

- Support of link aggregated cameras. See M_GC_TOTAL_PACKETS_RECEIVED_OUT_OF_ORDER and M_GC_MAX_NBR_PACKETS_OUT_OF_ORDER in this document.

- Added licensing support of Matrox GatorEye GigE Vision Cameras. Important note: In order to use the Matrox GatorEye camera with no other Matrox hardware, the Camera Discovery Service must be running otherwise a licensing error will be reported. The Service can be activated using the GigE page in MilConfig.

- New packetdelay.cpp example, located in: examples\board-specific\gigevision\packetdelay\. This can be used to determine inter-packet delay values to be programmed in the camera, which is required for its reliable operation.

1.16.3 Additions to command reference

- Additions to MsysHookFunction:
  - Can now hook a callback to M_CAMERA_PRESENT. Note that M_CAMERA_PRESENT hook support requires the use of the mtxgigeservice process. If the service is not started, a MIL error will be generated when calling MsysHookFunction with M_CAMERA_PRESENT.

- Additions to MsysGetHookInfo:
  - Support for M_CAMERA_PRESENT.

    - Returns a MIL_INT. Returns M_TRUE if a camera has been added or re-connected to the system. Returns M_FALSE if a camera has been removed from the system.

  - Support for M_NUMBER.

    - Returns a MIL_INT. Returns the system assigned digitizer device number associated with the camera that generated an M_CAMERA_PRESENT event.

  - Support for M_TIME_STAMP.

    - Returns a MIL_DOUBLE. Returns the operating system's time stamp of the event that generates a service request, in sec. The time stamp is generated by the operating system's performance counter.

  - Support for M_GC_USER_NAME_LENGTH.

    - Returns a MIL_INT. Returns the maximum number of characters in the camera's user-defined name string.

  - Support for M_GC_USER_NAME.

    - Returns a MIL_INT. Returns the user-defined name string for the camera.

  - Support for M_GC_IP_ADDRESS.

    - Returns a MIL_INT64. Returns the 32-bit internet protocol (IPv4) address of the GigE Vision device. Note that these addresses are returned in a 64-bit variable.

  - Support for M_GC_MAC_ADDRESS.

    - Returns a MIL_INT. Returns the system assigned digitizer device number associated with the camera that generated an M_CAMERA_PRESENT event.

- Additions to MdigAlloc:
  - M_GC_CAMERA_ID(MT("")) can now be used with the DigNum parameter (MdigAlloc), in conjunction with the following InitFlag parameter settings: M_GC_DEVICE_IP_ADDRESS or M_GC_DEVICE_USER_NAME. For example, the following 2 calls to MdigAlloc will allocate a camera whose IP address is 169.254.23.237 and whose user-defined name is MyCameraName, respectively.

    - MdigAlloc(..., M_GC_CAMERA_ID(MT("169.254.23.237")), ..., M_GC_DEVICE_IP_ADDRESS, ...);

    - MdigAlloc(..., M_GC_CAMERA_ID(MT("MyCameraName")), ..., M_GC_DEVICE_USER_NAME, ...);

  - Created M_DEV_NUMBER InitFlag to properly support customary allocation using device numbers.

- Additions to MdigControl/MdigInquire():
  - Support for M_GC_STREAMING_STOP.

    - MdigControl. Control value: M_DEFAULT. Stops the camera's image stream. Note that

MIL will automatically stop the camera's image stream if no grab commands are queued. However for tighter control of the stream this control can be used.

○ Support for M_GC_STREAMING_STOP_CHECK_PERIOD.

- MdigControl. Control value: MIL_INT. Represents the time period in milliseconds (ms), or M_DEFAULT. Default is 1000 ms. This is the period at which the driver checks if the camera's image stream should be stopped.

○ Support for M_GC_STREAMING_STOP_DELAY.

- MdigControl. Control value: MIL_INT. Represents the delay, in ms, before stopping the camera's image stream. Default is 0 ms. When the M_GC_STREAMING_STOP_CHECK_PERIOD time elapses, this delay is applied before a check is made to stop the stream.

○ Support for M_GC_ACQUISITION_MODE.

- MdigControl/Inquire. Inquire returns a MIL_INT. Controls the camera's AcquisitionMode feature. Can be set to M_CONTINUOUS/M_DEFAULT or M_SINGLE_FRAME.

○ Support for M_GC_FRAME_TIMESTAMP.

- MdigInquire. Inquire returns a MIL_DOUBLE. Inquires the timestamp, in seconds, of the last frame grabbed.

○ Support for M_GC_COUNTER_TICK_FREQUENCY.

- MdigInquire. Inquire returns a MIL_INT64. Inquires the camera's tick frequency register. A value of 0 Hz indicates the camera does not support time stamps.

○ Support for M_GC_HEARTBEAT_STATE (if supported by the camera).

- MdigControl/Inquire. Inquire returns a MIL_INT. Supported values: M_ENABLE, M_DISABLE, M_DEFAULT (same as M_ENABLE). Controls or inquires if the heartbeat mechanism is used to keep the GigE Vision Control channel open. Can only be used with GigE Vision cameras that support the heartbeat disable feature.

○ Support for M_GC_NIC_IP_ADDRESS.

- MdigInquire. Inquire returns a MIL_INT64. Returns the host NIC's IP address used with the camera.

○ Support for M_GC_NIC_MAC_ADDRESS.

- MdigInquire. Inquire returns a MIL_INT64. Returns the host NIC's MAC address used with the camera.

○ Support for M_GC_FIREWALL_PRESENT.

- MdigInquire. Inquire returns a MIL_INT set to M_TRUE or M_FALSE, depending on whether a firewall was detected. Note that if the camera associated to the digitizer supports firewall traversal, this inquire will always return M_FALSE.

○ Support for M_GC_PACKET_TIMEOUT.

- MdigControl/Inquire. Inquire returns a MIL_INT. Value is in ms. Default is 10 ms. This is the maximum amount of time to wait before flagging a packet as dropped.

○ Support for M_GC_FRAME_TIMEOUT.

- MdigControl/Inquire. Inquire returns a MIL_INT. Value is in ms. Default is 100 ms. This is the maximum amount of time to wait for the remaining packets of a frame after the trailer packet is received. If packets are missing, the frame is flagged as corrupted.

o Support for M_GC_PACKET_MAX_RETRIES.

- MdigControl/Inquire. Inquire returns a MIL_INT. Default is 3. This is the maximum number of retries for a packet. If reached, the frame is corrupted.

o Support for M_GC_FRAME_MAX_RETRIES.

- MdigControl/Inquire. Inquire returns a MIL_INT. Default is 30. This is the maximum number of packet retries per frame. If reached, the frame is corrupted.

o Support for M_GC_TOTAL_PACKETS_MISSED.

- MdigInquire. Inquire returns a MIL_INT. This is the total number of packets that are missing. Does not include recovered packets.

o Support for M_GC_TOTAL_PACKETS_RECEIVED.

- MdigInquire. Inquire returns a MIL_INT. This is the total number of packets received.

o Support for M_GC_TOTAL_PACKETS_RESENDS_NUM.

- MdigInquire. Inquire returns a MIL_INT. This is the total number of emitted PACKETRESEND commands. One resend command can contain multiple consecutive packets.

o Support for M_GC_TOTAL_PACKETS_RECOVERED.

- MdigInquire. Inquire returns a MIL_INT. This is the total number of packets recovered.

o Support for M_GC_TOTAL_FRAMES_GRABBED.

- MdigInquire. Inquire returns a MIL_INT. This is the total number of frames grabbed.

o Support for M_GC_TOTAL_FRAMES_CORRUPTED.

- MdigInquire. Inquire returns a MIL_INT. Returns the number of corrupted frames.

o Support for M_GC_TOTAL_PACKETS_RECEIVED_OUT_OF_ORDER.

- MdigInquire. Inquire returns a MIL_INT. This is the total number of packets that were received out of order, which usually happens with link aggregated cameras. If this number is high, increase the value of M_GC_MAX_NBR_PACKETS_OUT_OF_ORDER.

o Support for M_GC_STATISTICS_RESET.

- MdigControl. Control value must be M_DEFAULT. This resets the statistics.

o Support for M_GC_MAX_NBR_PACKETS_OUT_OF_ORDER.

- MdigControl/Inquire. Inquire returns a MIL_INT. Default value is 0. Controls the number of packets to wait before asking for a retransmission. This is useful when using link aggregated cameras where packets are received out of order.

o Support for M_PROCESS_PENDING_GRAB_NUM.

- MdigInquire. Inquire returns a MIL_INT. Returns the number of buffers remaining in the list of buffers used by MdigProcess(). Note that this number includes the buffer used by the grab in progress. When dealing with round-robin grabbing, this number will typically be relatively close to the total number of allocated grab buffers.

o Support for M_USER_BIT_FORMAT+Number

- MdigControl/Inquire. Inquire returns a MIL_INT. Controls the current electrical format of the specified physical input or output Line. Number can range from 0 or 1 up to the

maximum number of Lines in the camera. Supported ControlValues are:

- M_NO_CONNECT: The line is not connected.

- M_TRI_STATE: The Line is currently in Tri-State mode (Not driven).

- M_TTL: The Line is currently accepting or sending TTL level signals.

- M_RS422: The Line is currently accepting or sending RS422 level signals.

- M_LVDS: The Line is currently accepting or sending LVDS level signals.

- M_OPTO: The Line is Opto-Coupled.

- A MIL error will be generated for this control type if the specified Line number does not exist in the camera or if the Line format cannot be changed by the camera.

- A MIL error will be generated for any of the previous ControlValues if the specified format does not exist for the specified Line number.

o Support for M_USER_BIT_MODE+Number

- MdigControl/Inquire. Inquire returns a MIL_INT. Controls if the physical Line is used to input or output a signal. Number can range from 0 or 1 up to the maximum number of Lines in the camera. Supported ControlValues are:

- M_INPUT: The selected physical line is used to input an electrical signal.

- M_OUTPUT: The selected physical line is used to output an electrical signal.

- A MIL error will be generated for this control type if the specified Line number does not exist in the camera or if the Line mode cannot be changed by the camera.

- A MIL error will be generated for any of the previous ControlValues if the specified mode does not exist for the specified Line number.

o Support for M_AUX_SIGNAL_SOURCE+Number

- MdigControl/Inquire. Inquire returns a MIL_INT. Selects which internal signal to output on the selected Line. The corresponding M_USER_BIT_MODE must be M_OUTPUT. Number can range from 0 or 1 up to the maximum number of Lines in the camera. Supported ControlValues are:

- M_USER_BIT: The line will output a user-bit signal.

- M_TIMER1: The line will output if the chosen timer is in an active state.

- M_TIMER2: The line will output if the chosen timer is in an active state.

- A MIL error will be generated for this control type if the specified Line number does not exist in the camera or if the Line mode is set to M_INPUT.

- A MIL error will be generated for any of the previous ControlValues if the specified signal source does not exist for the specified Line number.

o Support for M_USER_BIT_VALUE+Number

- MdigControl/Inquire. Inquire returns a MIL_INT. For a control, this sets the value of the specified user bit. For an inquire, this returns the current status of the selected input or output Line. Number can range from 0 or 1 up to the maximum number of user-defined signals in the camera. Supported ControlValues are:

- M_ON: The selected physical line is used to input an electrical signal.

- M_OFF: The selected physical line is used to output an electrical signal.

- A MIL error will be generated for this control type if the specified Line number does not exist in the camera.

o Support for M_GC_FEATURE_BROWSER

- MdigControl. Creates the device feature browser dialog from a MIL console application. Supported control values are:

- M_DEFAULT: Opens the feature browser dialog. The calling thread is blocked.

- M_OPEN+M_SYNCHRONOUS: same as M_DEFAULT.

- M_OPEN+M_ASYNCHRONOUS: Opens the feature browser dialog. The calling thread returns as soon as the feature browser is created.

- M_CLOSE: Closes the feature browser dialog.

- Additions to MdigHookFunction
  o Support for M_GC_EVENT.

- Hooks a MIL function callback to a generic GigE Vision camera event. Note that the underlying event must be enabled in the camera with MdigControlFeature.

o Support for M_GC_EVENT+M_ACQUISITION_TRIGGER.

- Hooks a MIL function callback to the GenICam "AcquisitionTrigger" camera event.

o Support for M_GC_EVENT+M_ACQUISITION_START.

- Hooks a MIL function callback to the GenICam "AcquisitionStart" camera event.

o Support for M_GC_EVENT+M_ACQUISITION_END.

- Hooks a MIL function callback to the GenICam "AcquisitionEnd" camera event.

o Support for M_GC_EVENT+M_ACQUISITION_TRANSFER_START.

- Hooks a MIL function callback to the GenICam "AcquisitionTransferStart" camera event.

o Support for M_GC_EVENT+M_ACQUISITION_TRANSFER_END.

- Hooks a MIL function callback to the GenICam "AcquisitionTransferEnd" camera event.

o Support for M_GC_EVENT+M_ACQUISITION_ERROR.

- Hooks a MIL function callback to the GenICam "AcquisitionError" camera event.

o Support for M_GC_EVENT+M_FRAME_TRIGGER.

- Hooks a MIL function callback to the GenICam "FrameTrigger" camera event.

o Support for M_GC_EVENT+M_FRAME_START.

- Hooks a MIL function callback to the GenICam "FrameStart" camera event.

o Support for M_GC_EVENT+M_FRAME_END.

- Hooks a MIL function callback to the GenICam "FrameEnd" camera event.

o Support for M_GC_EVENT+M_FRAME_TRANSFER_START.

- Hooks a MIL function callback to the GenICam "FrameTransferStart" camera event.

- Support for M_GC_EVENT+M_FRAME_TRANSFER_END.

    - Hooks a MIL function callback to the GenICam "FrameTransferEnd" camera event.

- Support for M_GC_EVENT+M_EXPOSURE_START.

    - Hooks a MIL function callback to the GenICam "ExposureStart" camera event.

- Support for M_GC_EVENT+M_EXPOSURE_END.

    - Hooks a MIL function callback to the GenICam "ExposureEnd" camera event.

- Support for M_GC_EVENT+M_COUNTER_START+Number.

    - Hooks a MIL function callback to the GenICam "CounterXStart" camera event. Number determines the index of the Counter used to generate the event. Number can range from 0 to 8.

- Support for M_GC_EVENT+M_COUNTER_END+Number.

    - Hooks a MIL function callback to the GenICam "CounterXEnd" camera event. Number determines the index of the Counter used to generate the event. Number can range from 0 to 8.

- Support for M_GC_EVENT+M_TIMER_START+Number.

    - Hooks a MIL function callback to the GenICam "TimerXStart" camera event. Number determines the index of the Timer used to generate the event. Number can range from 0 to 8.

- Support for M_GC_EVENT+M_TIMER_END+Number.

    - Hooks a MIL function callback to the GenICam "TimerXEnd" camera event. Number determines the index of the Timer used to generate the event. Number can range from 0 to 8.

- Support for M_GC_EVENT+M_LINE_RISING_EDGE+Number.

    - Hooks a MIL function callback to the GenICam "LineXRisingEdge" camera event. Number determines the index of the Line used to generate the event. Number can range from 0 to 32.

- Support for M_GC_EVENT+M_LINE_FALLING_EDGE+Number.

    - Hooks a MIL function callback to the GenICam "LineXFallingEdge" camera event. Number determines the index of the Line used to generate the event. Number can range from 0 to 32.

- Support for M_GC_EVENT+M_LINE_ANY_EDGE+Number.

    - Hooks a MIL function callback to the GenICam "LineXRisingEdge" and "LineXFallingEdge" camera events. Number determines the index of the Line used to generate the event. Number can range from 0 to 32.

- Notes:

    - M_UNHOOK can be used with all of the above M_GC_EVENT events to disable the underlying event in the camera.

    - The above message channel events must be implemented in the GigE Vision camera, otherwise calling MdigHookFunction with M_GC_EVENT+... will generate a MIL error message.

- Additions to MdigGetHookInfo

- o Support for M_GC_PACKETS_MISSED.

  - Returns a MIL_INT. Returns the number of missing packets in the corrupted frame. If 0, the frame is not corrupted.

- o Support for M_GC_PACKETS_RECEIVED.

  - Returns a MIL_INT. This is the number of packets in the frame.

- o Support for M_GC_PACKETS_RESENDS_NUM.

  - Returns a MIL_INT. This is the number of resend commands sent to the camera for a frame. One resend command can contain multiple consecutive packets.

- o Support for M_GC_PACKETS_RECOVERED.

  - Returns a MIL_INT. This is the number of packets recovered in the frame.

- o Support for M_GC_FRAME_ERROR_CODE.

  - Returns a MIL_INT. This is the error code returned by the camera for a frame.

- o Support for M_GC_FRAME_LINE_COUNT.

  - Returns a MIL_INT. This is the number of lines in the frame.

- o Support for M_GC_FRAME_BLOCK_ID.

  - Returns a MIL_INT. This is the frame BlockID value from the camera. The BlockID in the camera is implemented as a 16-bit running counter.

- o Support for M_GC_FRAME_TIMESTAMP.

  - Returns a MIL_DOUBLE in seconds. This is the frame timestamp from the camera.

- o Support for M_COUNTER_INDEX.

  - Returns a MIL_INT. This is the index of the counter that generated the event.

- o Support for M_TIMER_INDEX.

  - Returns a MIL_INT. This is the index of the timer that generated the event.

- o Support for M_LINE_INDEX.

  - Returns a MIL_INT. This is the index of the line that generated the event.

- o Support for M_GC_EVENT_TYPE.

  - Returns a MIL_INT. This is the raw GigE Vision event sent by the camera.

- Additions to MdigInquireFeature():
  - o Support for M_INCREMENT (InquireFlag) when using M_TYPE_MIL_INT64 (FeatureDataType). Returns a MIL_INT64, representing the feature's increment value.

  - o Support for M_INCREMENT (InquireFlag) when using M_TYPE_MIL_INT32 (FeatureDataType). Returns a MIL_INT32, representing the feature's increment value.

  - o Support for M_INCREMENT (InquireFlag) when using M_TYPE_DOUBLE (FeatureDataType). Returns a MIL_DOUBLE, representing the feature's increment value.

1.16.4 GigE Vision utilities

- Matrox GigE Vision Feature Browser.

- Features below a selector feature are now grouped as such in the tree control.

- Features that are read-only or become read-only at run-time are now painted in a lighter color.

- Feature changes that affect other features now trigger a refresh of the dependent features.

- Fixed MFC control paint issues.

- Standalone feature browser can now be re-sized.

- Fixed integer value representation issues in UI.

- Support for the slider control on Float features.

- Camera model and name are now clearly identified.

- Fixed issues when large fonts are used.

- Matrox GigE Vision and Configuration Assistant.
  - Supports discovery of GigE Vision devices.

  - Supports camera packet size change.

  - Supports camera inter-packet delay change.

  - Supports camera user-defined name change.

  - Supports camera IP configuration change.

  - Detection of firewalls.

  - Camera GigE Vision Capability window.

    - Lists GigE Vision features supported by the camera.

  - Camera Statistics window. This lists:

    - Total Frames grabbed.

    - Total Data received.

    - Camera Ethernet bandwidth.

    - Camera frame rate.

    - Total packets received.

    - Total frames missed or corrupted.

    - Total packets recovered.

    - Total packets resent.

    - Total packets missed.

    - Total packets that timed out.

  - New System Configuration tab to control various GigE Vision driver parameters:

    - Camera discovery timeout.

    - Camera IP config timeout.

    - Discovery mode.

- Packed size search mode.

- Packet size search timeout.

- Maximum Packet size.

  - Limits the maximum packet size assigned by the driver to the camera. Useful in multi camera to single NIC scenarios as it reduces packet collisions.

- Camera heartbeat timeout.

- Camera acknowledge timeout.

- Camera stream stop delay.

- Camera manifest table use.

- Control of automatic Bayer conversion.

- Control of format compensation.

- Discovery service settings.

  - Discovery period.

  - Action on Ethernet link status change event.

  - First discovery delay after OS boot.

  - Controllable GigE Vision device cache.

- Matrox GigE Vision service (mtxgigeservice.exe)
  - Accelerated camera discovery and configuration.

  - Mtxgigeservice is now in automatic start mode.

  - Removed device persistence from service. Cameras that are removed from the system will get their device numbers recycled upon the next discovery cycle.

1.16.5 Improvements

- GenICam exceptions now wrapped through MIL errors.

- Now validating packet size passed to MdigControl(..., M_GC_PACKET_SIZE, ...) through GenICam if the camera supports the SFNC's standard name for PacketSize (GevSCPSPacketSize). This avoids programming an invalid packet size in the camera.

- Improved camera discovery performance.

- Support for RGB48 packed and planar buffer formats for grab (for use with 16 bit Bayer pixel formats).

- Maximum packet size negotiated by the driver is now 6KB by default. This improves acquisition reliability when multiple cameras are connected to the same NIC. Smaller packet sizes reduce the occurrence of packet collisions that occur in thernet switches thus reducing the frequency of corrupted frames. This parameter can be changed from the GigE Assistant's System configuration tab. See theMatrox GigE Vision and Configuration Assistant section of this document.

1.16.6 Behavioral change

- MIL User bits and hardware ports now follow the camera's numbering scheme. If the camera implements "Line" numbers in a zero based scheme, then M_USET_BIT+0 and M_HARDWARE_PORT_0 can be used to address the first Line of the camera (Line0). If the camera implements "Line" numbers in a one based scheme, then M_USET_BIT+0 and M_HARDWARE_PORT_0 cannot be used: M_USET_BIT+1 and

M_HARDWARE_PORT_1 must be used instead to address the first Line of the camera (Line1).

1.16.7 Bug fixes

- Fixed issue with mtxgigeservice where the service did not stop properly.

- Fixed issue with mtxgigeservice where it caused latencies in MIL hooks.

- Fixed digitizer device ordering issues when mtxgigeservice was used. Some cameras could become unusable.

- Fixed issue with User-defined Name in Matrox GigE Vision Assistant. User-defined names could not be written on some camera models.

- Fixed exception in Matrox GigE Vision Assistant when certain types of network devices are present on the system.

- Fixed issue when starting Matrox GigE Vision Assistant and mtxgigeservice was not started.

- Fixed memory and handle leak in mtxgige driver under special conditions.

- Fixed Windows stop error when the grab buffer's physical address is located above 2GB on 32 bit systems.

- Fixed issue with certain camera models where the first frame output by the device after the stream was started was being lost.

- Fixed issue with automatic Bayer conversion that was not working in the mtxgige driver.

- Fixed Windows stop error that occurred on fast systems (Core i7) grabbing simultaneously from multiple high frame rate cameras.

- Fixed issue with MdigGrab in M_ASYNCHRONOUS mode when digitizer format compensation was used. The wrong image could be transferred to the user's buffer.

- Fixed issue related to host UDP port numbers used during device discovery and digitizer allocation. There were instances where, on some systems, the port numbers selected were already allocated by third party applications. This resulted in failed camera discovery and/or failed digitizer allocations.

- Fixed issue related to driver issuing a GigE Vision WRITEMEM_CMD to the camera. The length parameter was sometimes not properly, set resulting in the command being ignored by the device.

- Fixed issue related to Mono10Packed and Mono12Packed pixel formats. There were instances where the image was not completely unpacked.

- Fixed issue related to triggered grabs with cameras that do not support the FrameStart TriggerSelector.

- Fixed Windows stop error that occurred at MsysAlloc when certain third party drivers were installed on the system.

- Fixed crash in milgigevision.dll that occurred with certain xml files.

- Fixed issue related to the IP address generated for a camera. Under certain circumstances, the address generated was invalid.

1.17 Limitations and particularities

- The following GigE Vision 2.0 features are not supported:
  - All-in transmission mode.

  - Raw data payload type.

  - File payload type.

- JPEG payload type.

- JPEG 2000 payload type.

- H.264 payload type.

- Multi-zone image payload type.

- Device specific payload type.

- Zeroconf device discovery.

- Multiple links configuration.

- The following GigE Vision 1.2 features are not supported:
  - Pending ack.

  - Primary application switchover.

- Note that, some GigE Vision-compatible cameras include libraries that, when installed, override your network interface card's native driver. While these third-party libraries have their own benefits, we cannot guarantee their compatibility with MIL. For best results, the computer being used to communicate with your GigE Vision-compatible camera should always use your network interface card's native driver (for example, an Intel driver for an Intel network interface card).

2. Supported operating systems

This section lists all the supported operating systems.

- 32-bit Windows® 7

- 64-bit Windows® 7

- 32-bit Windows® 8

- 64-bit Windows® 8

- 32-bit Windows® 10

- 64-bit Windows® 10

3. Location of examples (in the help file)

- In the help file, the location information written at the top of examples might not be up-to-date. Use MIL Example Launcher to find an example on disk.