RobCom provider for Matrox-based vision systems

Version 1.0.1

User's guide

December 16, 2015

[Remarks]

[Revision history]

Version	Date	Content
1.0.0	2014-09-09	First edition.
1.0.1	2014-11-12	Add commands. SendCommand, ReceiveCommand, SendPositionRequest, WaitPositionReply
	2015-12-16	Add dll information.

Contents

1. Introduction

This is a users guide for RobCom provider that is one of the CAO provider. RobCom provider is designed to communicate with vision systems, including smart cameras, configured using the Matrox Design Assistant vision software.

RobCom provider sends/receives commands from/to a Matrox Design Assistant project using Communication: Robot steps (Figure 1-1) through EtherNet(TCP/IP) connection. Figure 1-2 shows the communication overview with a Matrox Design Assistant project (running for example on a Matrox Iris GT smart camera).

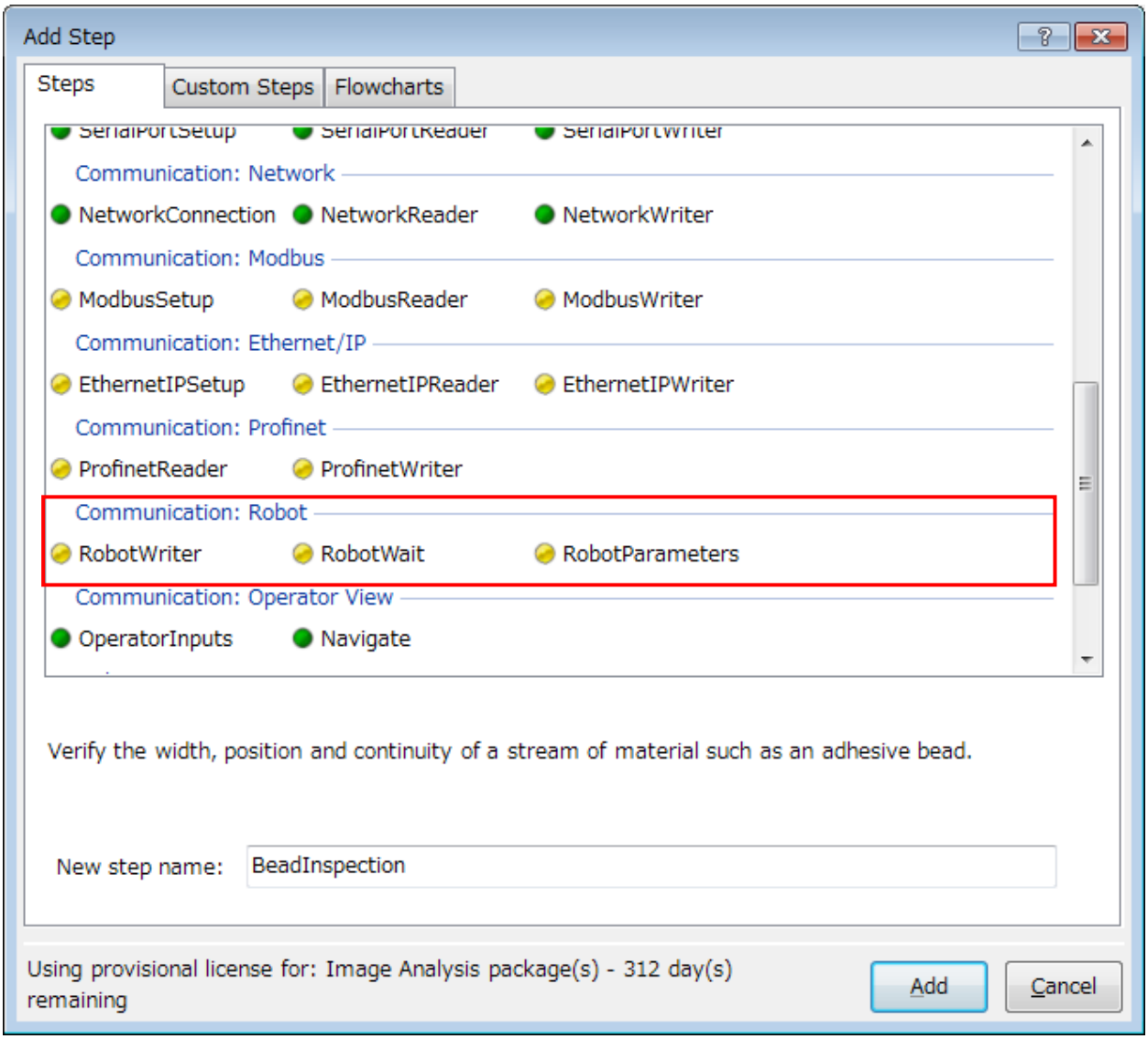

Figure 1-1Communication:Robot

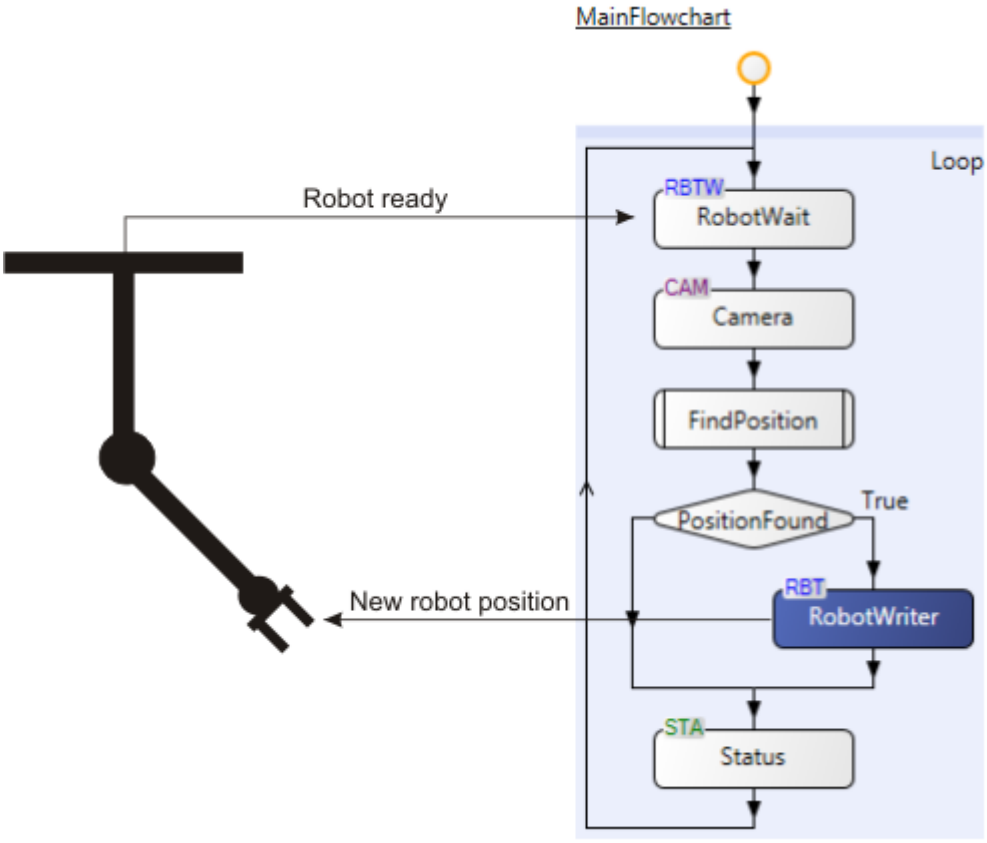

Figure 1-2 Communication overview

2. Outline of provider

2.1. Outline

RobCom provider activates its server at AddController execution. On the Matrox Design Assistant project side, you need to establish a connection with a port number that has been specified by AddController (see Figure 2-1). CaoController::Execute is provided as a way of command execution. For details about each command, refer to Section 2.5.

The file format of RobCom provider is DLL (Dynamic Link Library); that is automatically loaded from CAO engine when it is used. To use RobCom provider, install ORiN2SDK or manually register the registry in reference with the following table.

Table 2-1 RobCom provider

File name	CaoProvMatroxRobCom.dll
ProgID	CaoProv.Matrox.Robcom
Registry registration[1]	Regsvr32 CaoProvMatroxRobCom.dll
Registry unregistration	Regsvr32 /u CaoProvMatroxRobCom.dll

2.2. Method and property

2.2.1. CaoWorkspace::AddController method

RobCom provider establishes communication connection with communication connection parameters at AddController execution. Specify the timeout period and listening port in option character string. On the Matrox Design Assistant project side, specify a controller port number that is specified by “ConfigureRobots”(see Figure 2-1).

Syntax AddController(<bstrCtrlName:VT_BSTR>,<bstrProvName:VT_BSTR>,
<bstrPcName:VT_BSTR > [,<bstrOption:VT_BSTR>])

bstrCtrlName : [in] Controller name (given name)
bstrProvName : [in] Provider name. Fixed value =”CaoProv.Matrox.Robcom”
bstrPcName : [in] Machine name where provider operates
bstrOption : [in] Option character string

Table 2-2 shows items specified in option character string

Table 2-2 Option character string of CaoWorkspace::AddController

[1] Manual registration/unregistration is not necessary if the provider is installed by ORiN SDK.

Option	Description
Timeout[=<Timeout period>]	Specify the timeout period by milliseconds at sending and receiving. (Default: 500)
Port[=<Port number>]	Open a TCP port specified for listening

Example

```
Dim caoCtrl as Object
caoCtrl = cao.AddController("RobCom", "caoProv.Matrox.RobCom", "", "timeout=1000, port=2001")
```

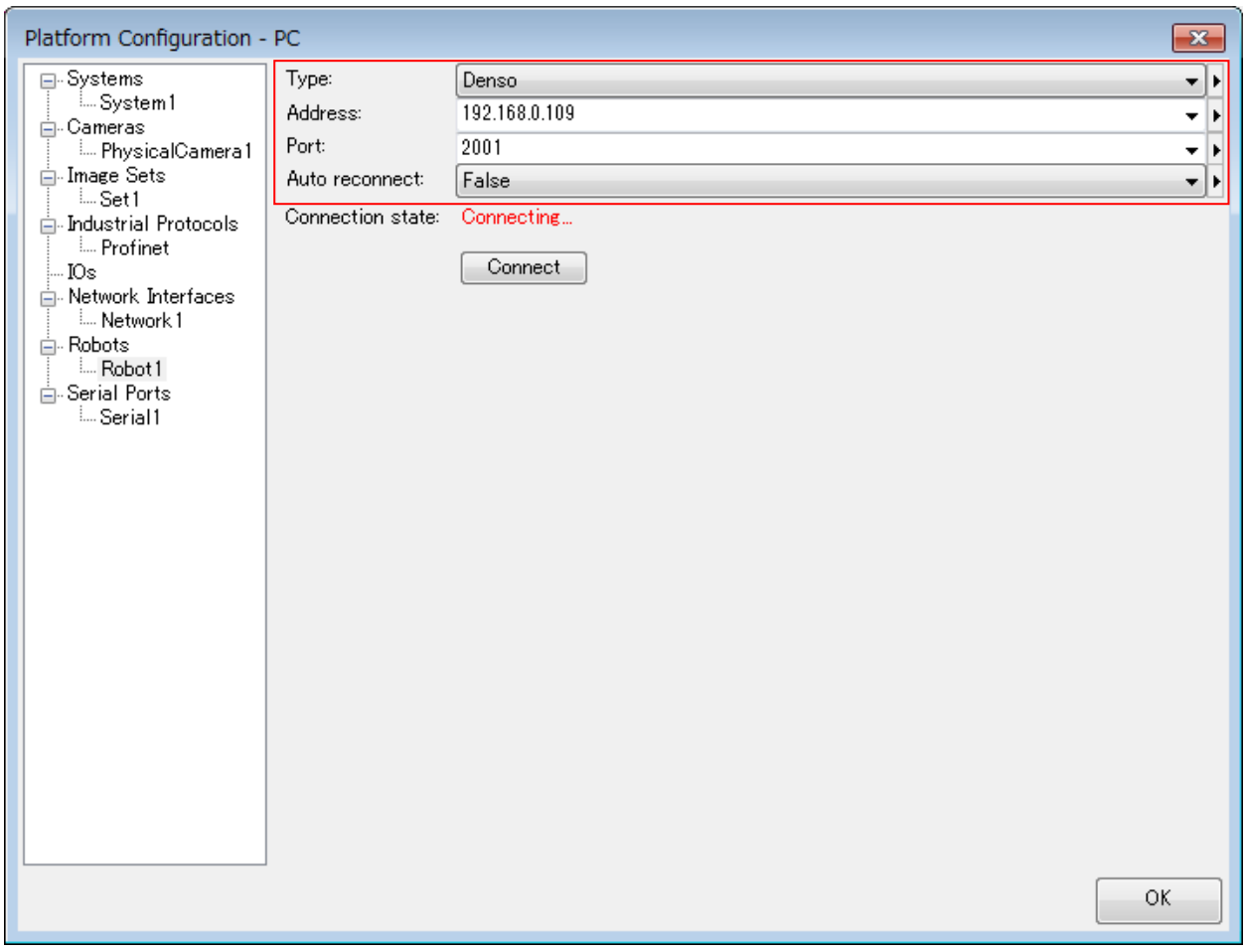

Figure 2-1 Setting window in Matrox Design Assistant

2.2.2. CaoController::Execute method

Send and receive commands from vision system. Specify a command name and command parameter for the first argument and the second argument, respectively. For detail about each command, refer to capture 2.5

Syntax Execute (<bstrCommandName:VT_BSTR>,[<vntParam : VT_VARIANT>])

bstrCommandName: [in] Command name

vntParam : [in] Parameter

2.2.3. CaoController::AddVariable method

Create variables to obtain system information. For about available variables, refer to Table 2-3.

Syntax AddVariable(<bstrVariableName:VT_BSTR>,[< bstrOption: VT_BSTR >])

bstrVariableName : [in] Variable name

bstrOption : [in] Option character string (unused)

Example

```
Dim bstrVal as String
bstrVal = caoCtrl.AddVariable("@VERSION", "")

bstrVal : "1.0.1"
```

2.2.4. CaoVariable::put_value property

At present, variable class does not support put_value property.

2.2.5. CaoVariable::get_value property

You can obtain properties with the format of Table 2-3 Controller class System variable list.

2.3. Variable list

2.3.1. Controller class

Table 2-3 Controller class System variable list

Variable name	Data type	Description	Attribute	
			get	put
@MAKER_NAME	VT_BSTR	Obtain a manufacturer name that have created provider	○	-
@VERSION	VT_BSTR	Obtain version information of provider	○	-
@STATUS	VT_BOOL	Obtain the connection status to vision system.	○	-

		True : Connection is confirmed False: Connection is not confirmed		

2.4. Error Code

In RobCom provider, specific error codes written in Table 2-4 are defined. For about ORiN2 common errors, refer to the error code section on "ORiN2 Programming Guide."

Table 2-4Specific error code list

Error name	Error number	Description
E_NOCLIENT	0x80100001	Not connected to vision system.
E_CONECTED	0x80100002	Connected to vision system.
E_ROBCOMSTATUS	0x80100003	Vision system returns an error.
E_WINDOWS_MASK	0x8091xxxx	Store an Winsock error code (16bits) in the XXXX part (lower two bites). Example） 10053 (WSAECONNABORTED) Software caused connection abort → 0x80912745

2.5. Command Reference

This section describes each command of CaoController::Execute method.

Table 2-5 CaoController::Execute Command list

RobCom command	Command	Description	
Communication command			
—	SendPos	Send ID and position data to vision system and request it to proceed.	P10
	ReceivePos	Receive position data from vision system.	P10
	SendPositionRequest	Send status, ID and position data to vision system and request it to proceed.	P11
	WaitPositionReply	Receive status, ID and position data from vision system.	P12
	SendCommand	Send communication command data to vision system.	P12

	ReceiveCommand	Receive communication command data from vision system.	P13
Supplemental command			
	DisconnectClient	Disconnect from currently connected vision system.	P14
	IsConnected	Check the connection status to vision system.	P14
	Clear	Clear data that have been received	P15
	SetTimeout	Set timeout period at data reception	P15

2.5.1. Communication command

2.5.1.1. CaoController::Execute(“SendPos”) command

Send position data to vision system and request it to proceed. To receive data from the vision system, use a ReceivePos command.

Syntax SendPos(<ID >, <Position>)

<ID > : [in] ID (VT_I4)

<Position> : [in] Position data (VT_R4 | VT_ARRAY)
X, Y, Z, Rx, Ry, Rz

Return value : [out] None

If the connection is not established, E_NOCLIENT will be returned.

Example

```
Dim lID as Long
Dim vntPos as Variant

lID = 1
vntPos = Array(100, 100, 200, 10, 10, 10)
caoCtrl.Execute "SendPos", Array(lID, vntPos)
```

2.5.1.2. CaoController::Execute ("ReceivePos”) command

Receive Position data from vision system.

Syntax ReceivePos ()

Argument : None

Return value : [out] Position data (VT_R4 | VT_ARRAY)
X, Y, Z, Rx, Ry, Rz

When the elapsed time passes the value specified by the timeout period, 0x80000900 will be returned. When

the Status of RobotWriter is Error, E_ROBCOMSTATUS will be returned.

Example

```
Dim vntPos as Variant

vntPos = caoCtrl.Execute("ReceivePos")
```

2.5.1.3. CaoController::Execute(“SendPositionRequest”) command

This command sends ID, Position data, and Status to the vision system to go on the next step. To receive data from the vision system, use a WaitPositionReply command.

Syntax SendPositionRequest (<Status>, <ID >, <Position>)

<Status>	:	[in] Status code (VT_I4)
		0 : M_COM_SUCCESS
		1 : M_COM_ERROR
<ID >	:	[in] Object ID (VT_I4)
<Position>	:	[in] Position data (VT_R4 | VT_ARRAY)
		X, Y, Z, Rx, Ry, Rz
Return value	:	[out] None

If the connection is not established, E_NOCLIENT will be returned.

Example

```
Dim lStatus as Long
Dim lID as Long
Dim vntPos as Variant

lStatus = 0
lID = 1
vntPos = Array(100, 100, 200, 10, 10, 10)
caoCtrl.Execute "SendPositionRequest", Array(lStatus , lID, vntPos)
```

2.5.1.4. CaoController::Execute ("WaitPositionReply") command

Receive data sent from the vision system

Syntax WaitPositionReply ()

Argument	:	None
Return value	:	[out] <Status>, <ID>, <Position> (VT_ARRAY | VT_VARIANT)
		<Status> : Status (VT_I4)
		<ID> : Object ID (VT_I4)
		<Position> : Position data (VT_R4 | VT_ARRAY)
		X, Y, Z, Rx, Ry, Rz

When the elapsed time passes the value specified by the timeout period, 0x80000900 will be returned.

Example

```
Dim vntRet as Variant
Dim lStatus as Long
Dim lID as Long
Dim vntPos as Variant

vntRet = caoCtrl.Execute("WaitPositionReply")
lStatus = vntRet(0)
lID = vntRet(1)
vntPos = vntRet(2)
```

2.5.1.5. CaoController::Execute("SendCommand") command

Send a command to the vision system.

Syntax SendCommand (< OperaCode>, <Status>, <ID>, <Position>)

< OperaCode>	:	[in] Operation code(VT_I4)
		1 : M_COM_ROBOT_FIND_POSITION
< Status>	:	[in] Status(VT_I4)
		0 : M_COM_SUCCESS
		1 : M_COM_ERROR
<ID>	:	[in] ID (VT_I4)
<Position>	:	[in] Position data (VT_R4 | VT_ARRAY)
		X, Y, Z, Rx, Ry, Rz
Return value	:	[out] none

Create a command data and then send it to the vision system. To receive the response, use a

ReceiveCommand command.

If the connection is not established, E_NOCLIENT will be returned.

Example

```
Dim lOpCode as Long
Dim lStatus as Long
Dim lID as Long
Dim vntPos as Variant

lOpCode = 1
lStatus = 0
lID = 1
vntPos = Array(100, 100, 200, 10, 10, 10)
caoCtrl.Execute "SendCommand", Array(lOpCode, lStatus, lID, vntPos)
```

2.5.1.6. CaoController::Execute ("ReceiveCommand") command

Receive a command response from the vision system.

Syntax ReceiveCommand ()

Argument	:	None
Return value	:	[out] < OperaCode>, <Status>, <ID>, <Position>
		(VT_ARRAY | VT_VARIANT)
		< OperaCode> : Operation code (VT_I4)
		2 : M_COM_ROBOT_FIND_POSITION_RESULT
		<Status> : Status (VT_I4)
		0 : M_COM_SUCCESS
		1 : M_COM_ERROR
		<ID> : Object ID (VT_I4)
		<Position> : Position data (VT_R4 | VT_ARRAY)
		X, Y, Z, Rx, Ry, Rz

When the elapsed time passes the value specified by the timeout period, 0x80000900 will be returned.

Example

```
Dim vntRet as Variant
```

```
Dim lOperaCode as Long
Dim lStatus as Long
Dim lID as Long
Dim vntPos as Variant

vntRet = caoCtrl.Execute("ReceiveCommand")
lOperaCode = vntRet(0)
lStatus = vntRet(1)
lID = vntRet(2)
vntPos = vntRet(3)
```

2.5.2. Supplemental command

2.5.2.1. CaoController::Execute ("DisconnectClient") command

Disconnect from currently connected vision system.

Syntax DisconnectClient()

Argument	:	none
Return value	:	none

Example

```
caoCtrl.Execute "DisconnectClient"
```

2.5.2.2. CaoController::Execute ("IsConnected") command

Check the connection status to vision system.

Syntax IsConnected ()

Argument	:	none
Return value	:	[out] Connection status (VT_BOOL) TRUE : Connection is confirmed FALSE : Connection is not confirmed

Example

```
Dim lVal as Long
lVal = caoCtrl.Execute("IsConnected")
```

2.5.2.3. CaoController::Execute ("Clear") command

Clear data that have been received

Syntax Clear

Argument : none
Return value : none

Example

```
Call caoCtrl.Execute("Clear")
```

2.5.2.4. CaoController::Execute ("SetTimeout") command

Change the timeout period

Syntax SetTimeout (<Timeout>)

Timeout : [in] Timeout period msec (VT_I4)
Return value : none

Example

```
Call caoCtrl.Execute("SetTimeout", 1000)
```

3. Sample program

The following shows a sample program written with RC8 Pacscript..

```
'!TITLE "MatroxRobComSample"

#include <Variant.h>

Sub Main
     Dim caoCtrl as Object

     ' socket bind (Port=PortNumber)
     caoCtrl = cao.AddController("RobCom", "CaoProv.Matrox.RobCom", "", "Port=2001,
timeout=6000")

     Do
          ' Wait for Connection
          if caoCtrl.IsConnected Then
               Exit Do
          End if
          delay 100
     Loop

     ' Send Position and Request
     caoCtrl.SendPos 1, VarChangeType(CurPos, VT_R8 + VT_ARRAY)

     ' Get Position
     P11 = caoCtrl.ReceivePos

     ' Disconnect Client
     caoCtrl.DisconnectClient

     ' Close
     cao.Controllers.Remove caoCtrl.Index
     caoCtrl = Nothing
End Sub
```

4. Tips

4.1. About Communication Command

RobCom provider equips three types of communication commands designed to communicate with vision system's project created by DA.

4.1.1. SendPos and ReceivePos

This program will obtain a workpiece's position data that has been found by the vision system. If you set the Status of the RobotWriter becoming "Error" when a workpiece is not found, the ReceivePos will return "E_ROBCOMSTATUS". In that case, do the error processing with an OnError statement or others.

```
'!TITLE "MatroxRobComSample"

#include <Variant.h>

Sub Main
	On Error Goto ErrorProc
	Dim caoCtrl as Object

	' socket bind (Port=PortNumber)
	caoCtrl = cao.AddController("RobCom", "CaoProv.Matrox.RobCom", "", "Port=2001,
timeout=6000")

Connect:
	Do
		' Wait for Connection
		if caoCtrl.IsConnected Then
			Exit Do
		End if
		delay 100
	Loop

SendAndRecieve:
	Do
		' Send Position and Request
		caoCtrl.SendPos 1, VarChangeType(CurPos, VT_R8 + VT_ARRAY)
Recieve:
		' Get Position
		P11 = caoCtrl.ReceivePos
	Loop
	Exit Sub

ErrorProc:
	On Error Goto ErrorProc
	PrintDbg "Error:" & HEX(Err.OriginalNumber)

	' Work not found
	If (Err.OriginalNumber = &H80100003) Then
		Goto SendAndRecieve
	' Timeout
	ElseIf (Err.OriginalNumber = &H80000900) Then
		Goto Recieve
	' Error
	Else
		caoCtrl.DisconnectClient
		Goto Connect
	End If
```

```
End Sub
```

4.1.2. SendPositionRequest and WaitPositionReply

This program will obtain not only the position data but also ID and Status of a workpiece that have been found by the vision system at the same time. Status provided by DA's RobotWriter will be stored in the return values of WaitPositionReply. Therefore, E_ROBCOMSTATUS will not be returned at an error occurrence.

```
'!TITLE "MatroxRobComSample"

#Include < Variant.h >

Sub Main
    On Error GoTo ErrorProc
    Dim caoCtrl As Object
    Dim vntRet as Variant

    ' socket bind (Port=PortNumber)
    caoCtrl = Cao.AddController( "RobCom", "CaoProv.Matrox.RobCom", "", "Port=2001,
timeout=6000" )

Connect:
    Do
        ' Wait for Connection
        If caoCtrl.IsConnected Then
            Exit Do
        End If
        Delay 100
    Loop

SendAndRecieve:
    Do
        ' Send Position and Request
        caoCtrl.SendPositionRequest 0, 1, VarChangeType( CurPos, VT_R8 + VT_ARRAY )
Recieve:
        ' Get Position
        vntRet = caoCtrl.WaitPositionReply
        I11 = vntRet(0)    ' Status
        I12 = vntRet(1)    ' ID
        P11 = vntRet(2)    ' Position
    Loop
    Exit Sub

ErrorProc:
    On Error GoTo ErrorProc
    PrintDbg "Error:" & Hex( Err.OriginalNumber )

    ' Work not found
    If ( Err.OriginalNumber = &H80000900 ) Then
        GoTo Recieve
    ' Error
    Else
        caoCtrl.DisconnectClient
        GoTo Connect
    End If

End Sub
```

4.1.3. SendCommand and ReceiveCommand

This program enables to send a given command to the vision system. Use this program when a new command is specified in DA.

```
'!TITLE "MatroxRobComSample"

#Include < Variant.h >

Sub Main
     On Error GoTo ErrorProc
     Dim caoCtrl As Object
     Dim vntRet as Variant

     ' socket bind (Port=PortNumber)
     caoCtrl = Cao.AddController( "RobCom", "CaoProv.Matrox.RobCom", "", "Port=2001,
timeout=6000" )

Connect:
     Do
          ' Wait for Connection
          If caoCtrl.IsConnected Then
               Exit Do
          End If
          Delay 100
     Loop

SendAndRecieve:
     Do
          ' Send Command
          caoCtrl.SendCommand 1, 0, 1, VarChangeType( CurPos, VT_R8 + VT_ARRAY )
Recieve:
          ' Get Command reply
          vntRet = caoCtrl.ReceiveCommand
          I10 = vntRet(0)    ' OperaCode
          I11 = vntRet(1)    ' Status
          I12 = vntRet(2)    ' ID
          P11 = vntRet(3)    ' Position
     Loop
     Exit Sub

ErrorProc:
     On Error GoTo ErrorProc
     PrintDbg "Error:" & Hex( Err.OriginalNumber )

     ' Work not found
     If ( Err.OriginalNumber = &H80000900 ) Then
          GoTo Recieve
     ' Error
     Else
          caoCtrl.DisconnectClient
          GoTo Connect
     End If

End Sub
```