industriales
etsii

Escuela Técnica
Superior
de Ingeniería
Industrial

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Escuela Técnica Superior de Ingeniería Industrial

Development and Automation of a Robotic Welding Cell Using Machine Vision in Halcon Programming Environment

APPENDIX

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Autor: **José Manuel Pastor Alcaraz**
Director: Juan Suardíaz Muro

Cartagena, Julio 2015

Universidad
Politécnica
de Cartagena

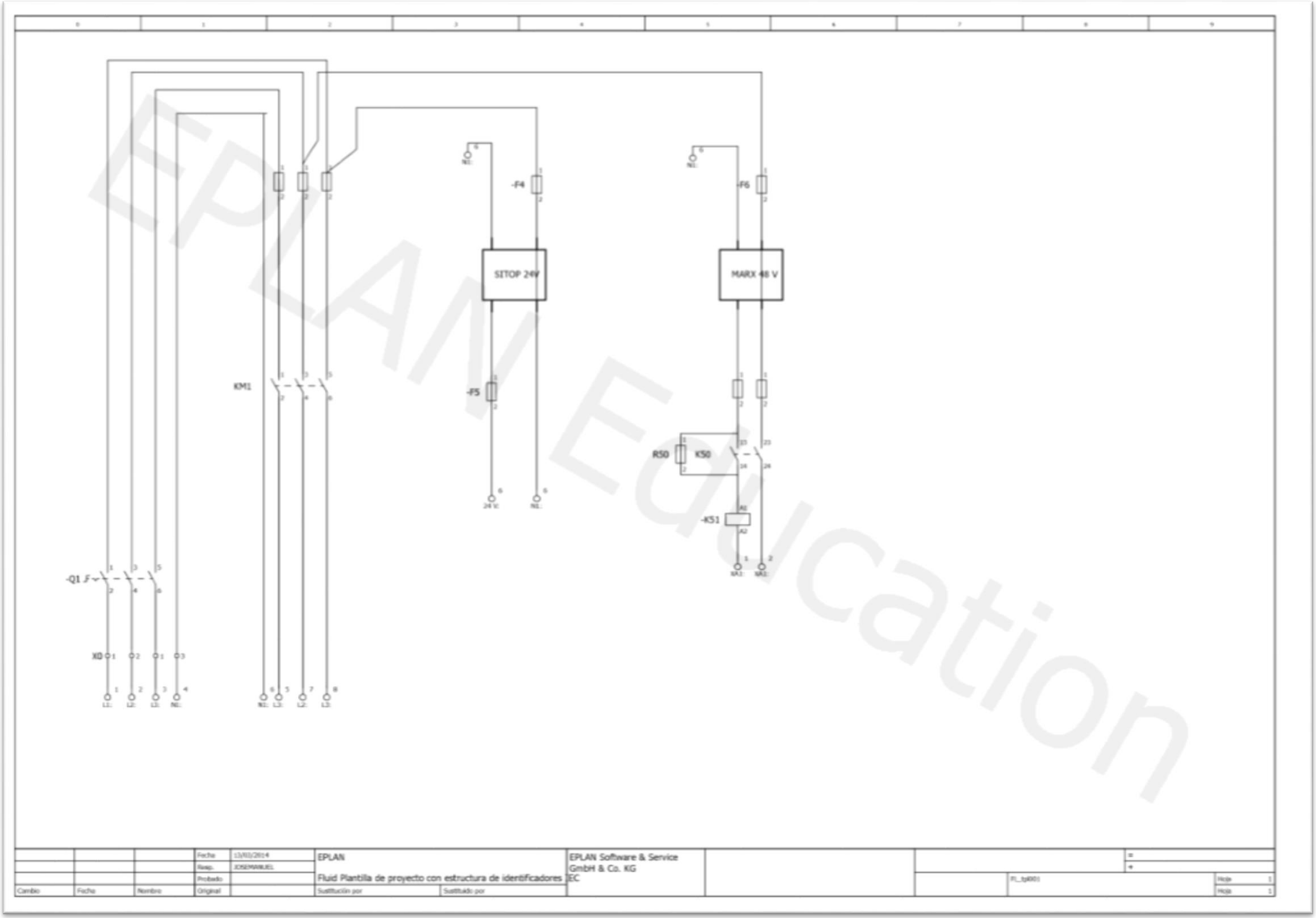
Appendix

Index

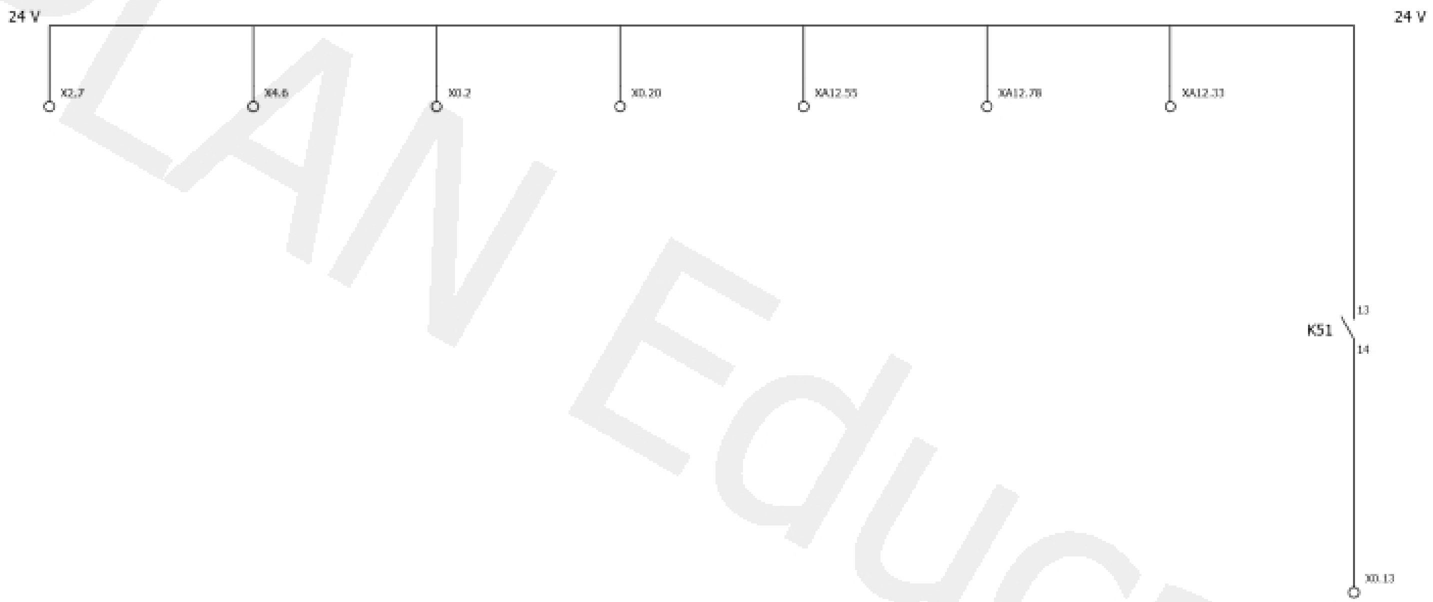
A1. Pre-Set and Installation

A1.1. Electical Shchematics of the Communication Cabinet

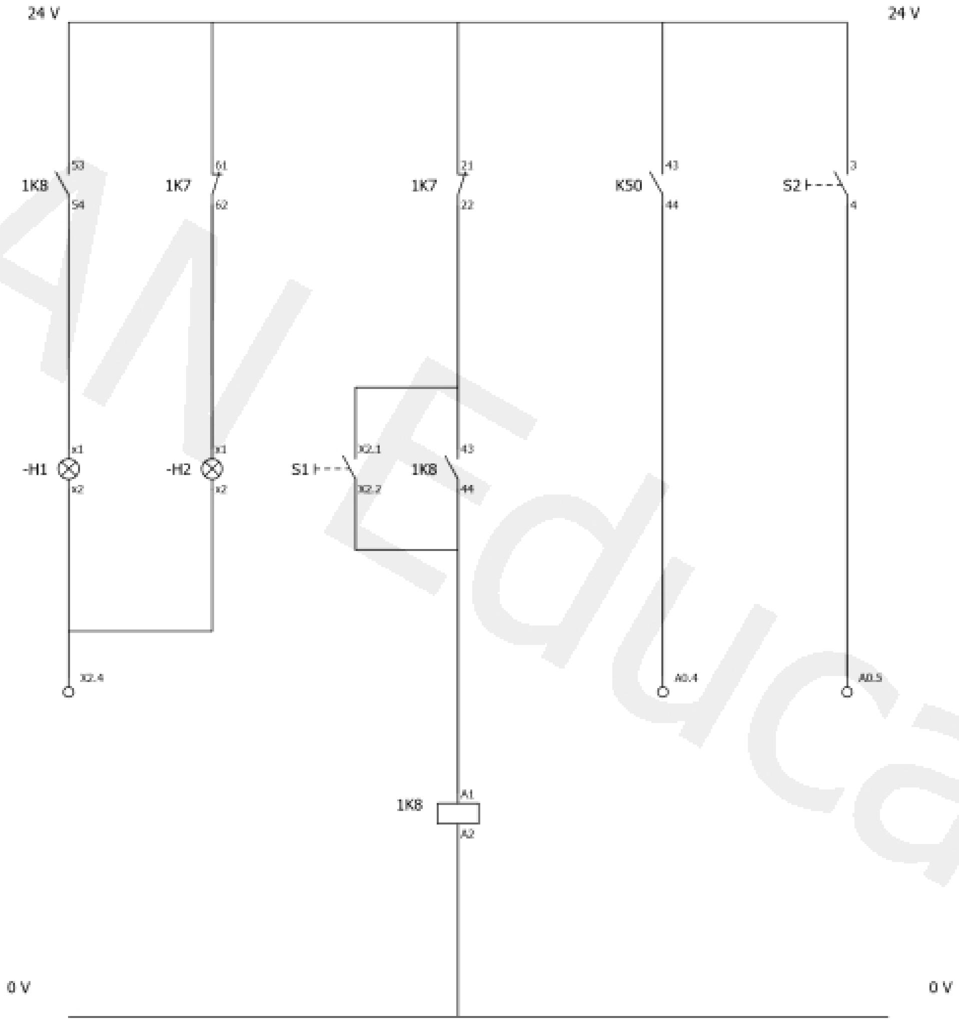
Cable Terminal	**Purpose**
X0	Cable from the robot controller
X2	Cable from the cleaner cell
XA1	Cable from welding power source
X4	DeviceNet/ CAN



			Fecha	13/03/2014	EPLAN	EPLAN Software & Service GmbH & Co. KG	=
			Resp.	JOSEMANUEL			+
			Probado		Fluid Plantilla de proyecto con estructura de identificadores IEC		FL_tpl001 Hoja 1
Cambio	Fecha	Nombre	Original		Sustitución por / Sustituido por		Hoja 1



			Fecha	14/03/2014	EPLAN		EPLAN Software & Service GmbH & Co. KG			=
			Resp.	JOSEMANUEL						+
			Probado		Fluid Plantilla de proyecto con estructura de identificadores IEC				FI_tpl001	Hoja 1
Cambio	Fecha	Nombre	Original		Sustitución por	Sustituido por				Hoja 1



1

			Fecha	14/03/2014
			Resp.	JOSEMANUEL
			Probado	
Cambio	Fecha	Nombre	Original	

EPLAN

Fluid Plantilla de proyecto con estructura de identificadores IEC

Sustitución por

Sustituido por

EPLAN Software & Service
GmbH & Co. KG

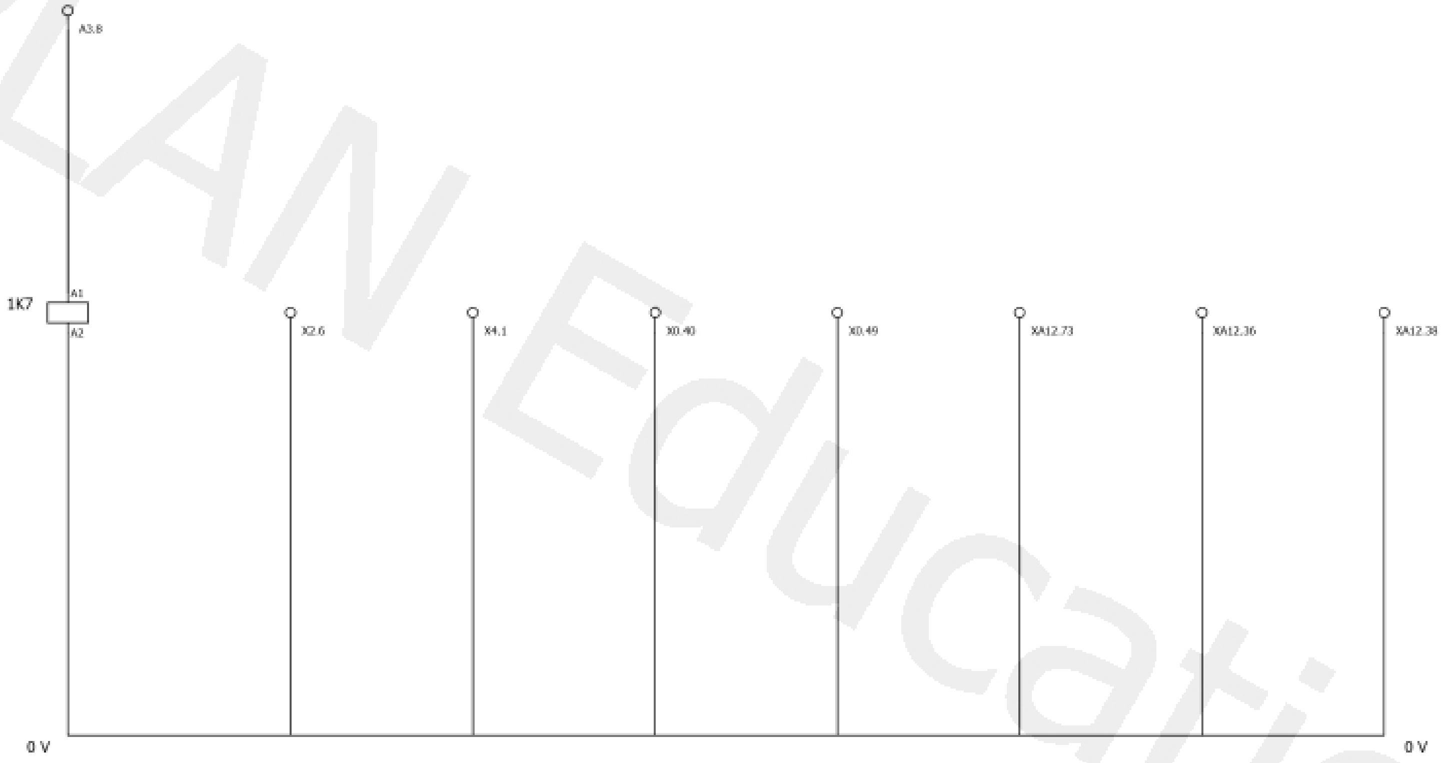
=

+

FL_tpl001

Hoja 2

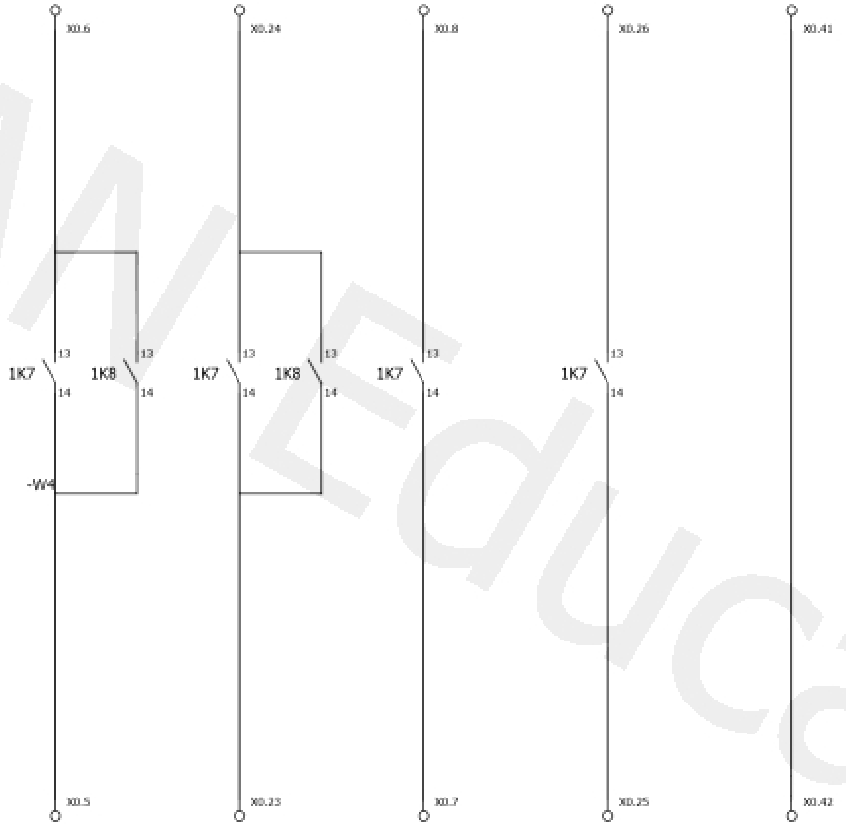
Hoja 2



1

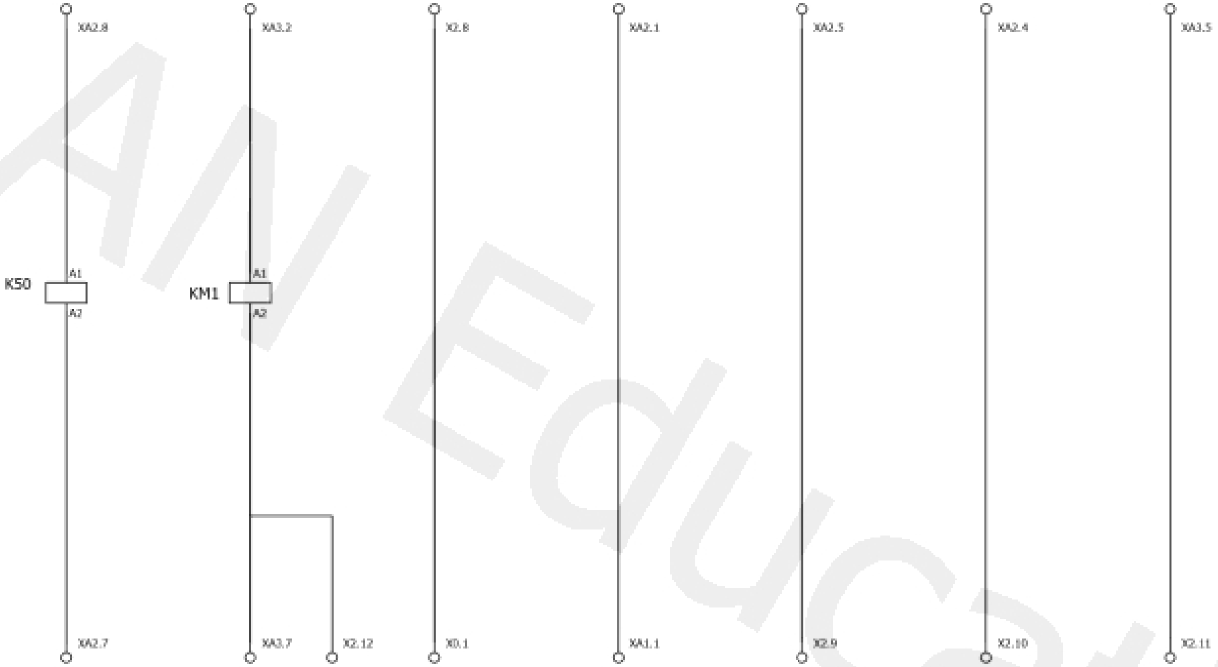
2

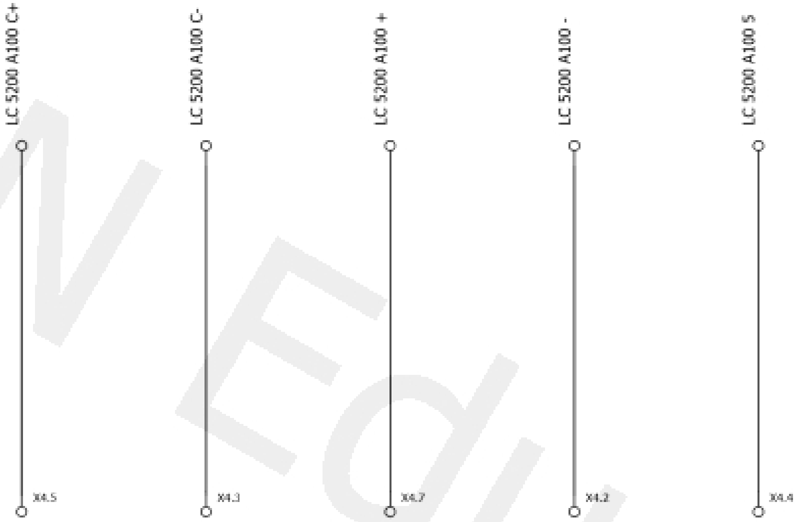
			Fecha	14/03/2014	EPLAN	EPLAN Software & Service GmbH & Co. KG			=
			Resp.	JOSEMANUEL					+
			Probado		Fluid Plantilla de proyecto con estructura de identificadores IEC			FL_tpl001	Hoja 1.a
Cambio	Fecha	Nombre	Original		Sustitución por / Sustituido por				Hoja 3



2

			Fecha	14/03/2014	EPLAN	EPLAN Software & Service GmbH & Co. KG		=
			Resp.	JOSEMANUEL				+
			Probado		Fluid Plantilla de proyecto con estructura de identificadores IEC		FL_tpl001	Hoja 3
Cambio	Fecha	Nombre	Original		Sustitución por / Sustituido por			Hoja 4


XA2.8
XA3.2
X2.8
XA2.1
XA2.5
XA2.4
XA3.5
K50
A1
A2
KM1
A1
A2
XA2.7
XA3.7
X2.12
X0.1
XA1.1
X2.9
X2.10
X2.11


LC 5200 A100 C+
LC 5200 A100 C-
LC 5200 A100 +
LC 5200 A100 -
LC 5200 A100 S
X4.5
X4.3
X4.7
X4.2
X4.4

A1.2. IOSYS.ini

```
;==========================================================
; IOSYS.INI - Configuration file for the IO-System
;==========================================================
; For configuration help go to the end of this file.
;----------------------------------------------------------
; ATTENTION !!!! Since V5.0 Build13 we have removed the DeviceNet
;                driver "dndrv.o". Now you have to use the driver
;                "dn2drv.o" and the appropriate syntax (form 2)

[CONFIG]
VERSION=2.00
[DRIVERS]
RSI=50,rsiLibInit,rsiLib.o
;CNKE2=21,cnke2CPInit,cnke2drv.o
;DNSC6=20,dnsc6Init,dnsc6drv.o
;DNSC5=19,dnsc5Init,dnsc5drv.o
;DNSC4=18,dnsc4Init,dnsc4drv.o
;DNSC3=17,dnsc3Init,dnsc3drv.o
;CNKE1=16,cnke1CPInit,cnke1drv.o
;INTERBUSPCI=15,ibsCPPciInit,ibpcidrv.o
;DSEIO=14,dseIoInit,dseiodrv.o
;DNSC2=13,dnsc2Init,dnsc2drv.o
;DNSC1=12,dnsc1Init,dnsc1drv.o
;PBMASL=11,pbmsInit,pfbmsdrv.o
DEVNET=2,dnInit,dn2drv.o
;INTERBUS=1,ibusInit,ibusdrv.o
;MFC=0,mfcEntry,mfcdrv.o

[RSI]
[MFC]
INW0=0        ;$IN[1-16]
OUTW0=0       ;$OUT[1-16]
OUTW2=2       ;$OUT[17-32]
```

```
[INTERBUS]
;------- Inputs ---------
;Slave Inputs
INW50=896    ;$IN[401-416]
INW52=898    ;$IN[417-432]
INW54=900    ;$IN[433-448]
INW56=902    ;$IN[449-464]
;------- Outputs --------
;Slave Outputs
OUTW50=896   ;$OUT[401-416]
OUTW52=898   ;$OUT[417-432]
OUTW54=900   ;$OUT[433-448]
OUTW56=902   ;$OUT[449-464]


[DEVNET]
;Devicenet MACID 1 (kast)
INB0=1,0,x1  ;$in[1-8]
OUTB0=1,0,x1 ;$out[1-8]

;Devicenet MACID 21 (lasbron FRONIUS) 12 bytes in en uitgangen
INB2=21,0,x1 ;$in[17-24]
INB3=21,1,x1 ;$in[25-32]
INB4=21,2,x1 ;$in[33-40]
INB5=21,3,x1 ;$in[41-48]
INB6=21,4,x1 ;$in[49-56]
INB7=21,5,x1 ;$in[57-64]
INB8=21,6,x1 ;$in[65-72]
INB9=21,7,x1 ;$in[73-80]
INB10=21,8,x1 ;$in[81-88]
INB11=21,9,x1 ;$in[89-96]
INB12=21,10,x1 ;$in[97-104]
INB13=21,11,x1 ;$in[105-112]

OUTB2=21,0,x1;$out[17-24]
OUTB3=21,1,x1;$out[25-32]
```

```
OUTB4=21,2,x1;$out[33-40]

OUTB5=21,3,x1;$out[41-48]

OUTB6=21,4,x1 ;$OUT[49-56]

OUTB7=21,5,x1 ;$OUT[57-64]

OUTB8=21,6,x1 ;$OUT[65-72]

OUTB9=21,7,x1 ;$OUT[73-80]

OUTB10=21,8,x1 ;$OUT[81-88]

OUTB11=21,9,x1 ;$OUT[89-96]

OUTB12=21,10,x1 ;$OUT[97-104]

OUTB13=21,11,x1 ;$OUT[105-112]


[PBMASL]

[DNSC1]

[DNSC2]

[DNSC3]

[DNSC4]

[DNSC5]

[DNSC6]


[DSEIO]

INDW0=0      ;$IN[1-32]

OUTDW0=0     ;$OUT[1-32]


[INTERBUSPCI]

;------- Inputs ---------

;Slave Inputs

INW50=896    ;$IN[401-416]

INW52=898    ;$IN[417-432]

INW54=900    ;$IN[433-448]

INW56=902    ;$IN[449-464]

;------- Outputs --------

;Slave Outputs

OUTW50=896   ;$OUT[401-416]

OUTW52=898   ;$OUT[417-432]

OUTW54=900   ;$OUT[433-448]

OUTW56=902   ;$OUT[449-464]
```

```
[CNKE1]
; =ConNo, additional offset,xSize

[CNKE2]
; =ConNo, additional offset,xSize

[VIO]
;INW0=0       ;$IN[1-16]
;INW8=2       ;$IN[65-80]
;OUTW0=0      ;$OUT[1-16]
;OUTW2=2      ;$OUT[17-32]

[O2I]
;INW4=0       ;$IN[33-48]
;INW6=2       ;$IN[49-64]
;OUTW4=0      ;$OUT[33-48]
;OUTW6=2      ;$OUT[49-64]

[IOLINKING]
[END SECTION]

;==========================================================
;Valid entries have the following formats.
;Arguments in squared brackets are optional.
;If nothing else is mentioned, arguments are decimal.

;Digital Inputs and Outputs:
;
; Form 1:
; {token}{offset}={byte}[,{multip}]
;
; {token}   INB (byte), INW (word), INDW (double word)
;           OUTB, OUTW, OUTDW
; {offset}  byte offset of robot IO System (0..m)
; {byte}    byte offset over all peripheral devices (0..m)
;           Offset starts with 0 at the first device and
;           ends with m at the end of the last device.
; {multip}  creats n dataobjects of {token}
; Example:
; OUTW4=2,x3
; Three words of the periphery, starting at byte 2,
; are mapped to the outputs 33-80.
;
; Form 2:
; {token}{offset}={address},{byte}[,{multip}]
;
; {token}   INB, INW, INDW, OUTB, OUTW, OUTDW
; {offset}  byte offset of robot IO System
; {address} address of a peripheral device (0..m)
;           driver specific information, see descr. below
; {byte}    byte offset at this peripheral device (0..m)
;           Offset starts with 0 at the every device
;           driver specific information, see descr. below
; {multip}  creats n dataobjects of {token}
; Example:
; INW4=10,0,x2
; Two words of the peripheral device with address 10 and
; up from byte 0 are mapped to the inputs 33-80.

;Analog Inputs and Outputs:
;
; Form 1:
; {token}{num}={byte},{res},{type}[,CAL{factor}]
;
; {token}   ANIN or ANOUT
; {num}     number of the analog channel (1..i)
; {byte}    byte offset over all peripheral devices (0..m)
;           Offset starts with 0 at the first device and
;           ends with m at the end of the last device.
; {res}     resolution of the analog value (number of bits)
; {type}    type of analog value
;           0 : right justified without sign
;           1 : right justified with    sign
;           2 : left  justified without sign
;           3 : left  justified with    sign
; {factor}  maximum analog value, decimal without prefix,
;           hexadec. with prefix 0x or octal with prefix 0
;           "CAL 0" or no entry sets factor to its maximum
; Example:
; ANIN1=10,12,3
; The analog input No.1 is used. The byte offset on
; peripheral side is 10, the resolution is 12 bit and the
; type of analog value is 3 (left justified with sign).
; The maximum binary analog value is 2047.
;
; Form 2:
; {token}{num}={address},{byte},{res},{type}[,CAL{factor}]
;
; {token}   ANIN or ANOUT
; {num}     number of the analog channel (1..i)
; {address} address of a peripheral device (0..m)
;           driver specific information, see descr. below
; {byte}    byte offset at this peripheral device (0..m)
;           Offset starts with 0 at the every device
;           driver specific information, see descr. below
; {res}     resolution of the analog value (number of bits)
; {type}    type of analog value
;           0 : right justified without sign
;           1 : right justified with    sign
;           2 : left  justified without sign
;           3 : left  justified with    sign
; {factor}  maximum analog value, decimal without prefix,
;           hexadec. with prefix 0x or octal with prefix 0
;           "CAL 0" or no entry sets factor to its maximum
; Example:
; ANIN3=30,0,16,2,CAL 0x6C00
; The analog input No.3 is used. The device address is 30,
; the byte offset at this device is 0, the resolution is
; 16 bit and the type of analog value is 2 (left justified
; without sign). The maximum binary analog value is 27648.
; The CAL-factor is especially required in case of using
; Profibus analog modules.

;particularities:
;[MFC] MFC-IO with KRC1 / CAN-IO-Modul with KRC2
;     Entries in form 1
;
;[INTERBUS/INTERBUSPCI] Interbus Phoenix Mast./Slave Cu/LWL
;     Entries in form 1
;     $IN/OUT[n_1]=(n+1)*8-7
;     $IN/OUT[n_8]=(n+1)*8

;[DEVNET] DeviceNet on the KUKA MFC
;     Entries in form 2 for driver dn2drv.o
;     {address}=DeviceNet MACID

;[DNSC1] DeviceNet LPDN scanner channel 1
;     Entries in form 2
;     {address} = DeviceNet slave MACID
;     {address} = MACID of CH1 ==> Slave part of LPDN CH1
;
;[DNSC2] DeviceNet LPDN scanner channel 2
;     Entries in form 2
;     {address} = DeviceNet slave MACID
;     {address} = MACID of CH2 ==> Slave part of LPDN CH2

;[DNSC3] DeviceNet LPDN scanner channel 1
;     Entries in form 2
;     {address} = DeviceNet slave MACID
;     {address} = MACID of CH1 ==> Slave part of LPDN CH1

;[DNSC4] DeviceNet LPDN scanner channel 2
;     Entries in form 2
;     {address} = DeviceNet slave MACID
;     {address} = MACID of CH2 ==> Slave part of LPDN CH2
;
;[DNSC5] DeviceNet LPDN scanner channel 1
;     Entries in form 2
;     {address} = DeviceNet slave MACID
;     {address} = MACID of CH1 ==> Slave part of LPDN CH1
;
;[DNSC6] DeviceNet LPDN scanner channel 2
;     Entries in form 2
;     {address} = DeviceNet slave MACID
;     {address} = MACID of CH2 ==> Slave part of LPDN CH2
;
;[PBMASL] ProfiBus Siemens Master/Slave CP5614
;     Entries in form 2
;     {address} = Slave DP-address
;     {address} = 127 ==> Slave part of CP5614
;
;[DSEIO] Digital inputs/outputs for KR C3A
;     Entries in form 1
;
;[CNKE1] ControlNet 1784PCIC LP-Elektronik
;     Entries in form 2
;     {address} = ConNo
;     {byte}    = additional offset
;
;[CNKE2] ControlNet 1784PCIC LP-Elektronik
;     Entries in form 2
;     {address} = ConNo
;     {byte}    = additional offset
;
;[VIO] inputs/outputs for Virtual IO driver over TCP/IP
;     Entries in form 1
;
;VIO=30,vioInit,vio_drv.o
;
;[O2I] inputs/outputs for 'output to input for software developers'
```

```
;     Entries in form 1
;
;O2I=31,o2iInit,o2i_drv.o
;
;[IOLINKING] Outputs follow inputs
;     Special form:
;     $OUT[{bitoffset}]=$IN[{bitoffset}]
;
;     {bitoffset}    Bit(!)offset in the robot I/O-System,
;                    starting with 1 (1..MAXIO)
;
;     Example: $OUT[512]=$IN[401]
;     In this case output nr. 512 (bit 8 of byte 63)
;     is linked to input  nr. 401 (bit 1 of byte 50)
;
;     Notes:
;     IOLINKING means outputs follow inputs in the robot
;     I/O-system (within ipo-cycle), regardless if they
;     are mapped to drivers.
;     Port ranges cannot be specified, each bit must be
;     linked by itself.
;     Only a maximum of MAX_IOLINKS can be configured (set
;     in progress.ini, if this value is increased, robot
;     functionality cannot be guaranteed!).
;----------------------------------------------------------
; 04/02/02  section [IOLINKING] added
```

A1.3. CONFIG.dat

```
&REL 49
DEFDAT  $CONFIG
;FOLD BASISTECH GLOBALS
;=================================
; Default parameters for movement
; These values shouldn't be changed

;==================================

;----------------------------------

; general MOVEMENT - parameters:

;----------------------------------

INT DEF_OV_PRO=100

INT DEF_ADVANCE=3


; PTP - MOVEMENTS

;----------------------------------

INT DEF_VEL_PTP=100

INT DEF_ACC_PTP=50


; CP - MOVEMENTS

;----------------------------------

DECL CIRC_TYPE DEF_CIRC_TYP=#BASE

DECL  JERK_STRUC  DEF_JERK_STRUC={CP  500.0,ORI  50000.0,AX  {A1  1000.0,A2  1000.0,A3
1000.0,A4  1000.0,A5  1000.0,A6  1000.0,E1  1000.0,E2  1000.0,E3  1000.0,E4  1000.0,E5
1000.0,E6 1000.0}}

REAL DEF_VEL_CP=2.0

REAL DEF_VEL_ORI1=200.0

REAL DEF_VEL_ORI2=200.0

REAL DEF_VEL_ORIS=200.0

REAL DEF_ACC_CP=2.29999995

REAL DEF_ACC_ORI1=100.0

REAL DEF_ACC_ORI2=100.0

REAL DEF_ACC_ORIS=200.0

REAL DEF_VEL_FACT=1.0


; APO - parameters

;--------------------------------

INT DEF_APO_CPTP=50

INT DEF_APO_CVEL=100

REAL DEF_APO_CDIS=3.0

REAL DEF_APO_CORI=5.0

REAL DEF_APO_CORIS=80.0
```

```
;==================================

; Structures:

;==================================

ENUM BAS_COMMAND INITMOV,ACC_CP,ACC_PTP,VEL_CP,VEL_PTP,ACC_GLUE,TOOL,BASE,EX_BASE,PTP_DAT,CP_DAT,OUT_SYNC,OUT_ASYNC,GROUP,FRAMES,PTP_PARAMS,CP_PARAMS

ENUM OUT_MODETYPE TRIGGER_,CONT_,STOP_

ENUM IPO_M_T NONE,TCP,BASE

ENUM APO_MODE_T CPTP,CDIS

STRUC DIG_OUT_TYPE INT FIRST_BIT,LENGTH,PARITY,CODING

STRUC CTRL_IN_T INT IN_NR,CHAR NAME_NAT[20]

STRUC CTRL_OUT_T INT OUT_NR,BOOL INI,CHAR NAME_NAT[20]

STRUC FCT_OUT_T INT NO,REAL PULS_TIME,BOOL STATE

STRUC FCT_IN_T INT NO,BOOL STATE

STRUC PDAT REAL VEL,ACC,APO_DIST,APO_MODE_T APO_MODE

STRUC LDAT REAL VEL,ACC,APO_DIST,APO_FAC,ORI_TYPE ORI_TYP,CIRC_TYPE CIRC_TYP,REAL JERK_FAC,INT EXAX_IGN,CIRC_MODE CB

STRUC FDAT INT TOOL_NO,BASE_NO,IPO_MODE IPO_FRAME,CHAR POINT2[24],BOOL TQ_STATE

STRUC ODAT INT OUT_NO,BOOL STATE,REAL PULSE_TIME,OUT_MODETYPE OUT_MODE,REAL TIME_DELAY,OFFSET

STRUC BASIS_SUGG_T CHAR POINT1[24],POINT2[24],CP_PARAMS[24],PTP_PARAMS[24],CONT[24],CP_VEL[24],PTP_VEL[24],SYNC_PARAMS[24],SPL_NAME[24]


STRUC OUT_SUGG_T CHAR PARAMS[24]

STRUC MACHINE_DEF_T CHAR NAME[24],INT COOP_KRC_INDEX,CHAR PARENT[24],FRAME ROOT,ESYS MECH_TYPE,CHAR GEOMETRY[255]

STRUC MACHINE_TOOL_T INT MACH_DEF_INDEX,CHAR PARENT[24],CHAR GEOMETRY[255]

STRUC MACHINE_FRAME_T INT MACH_DEF_INDEX,CHAR PARENT[24],CHAR GEOMETRY[255]


;==================================

; TORQUE MONITORING

;==================================

SIGNAL sTQM_SPSACTIVE $IN[1026]

SIGNAL sTQM_SPSSTATUS $OUT[1024]


BOOL bTQM_ACTIVE=FALSE

BOOL bTQM_CYC=FALSE

BOOL bTQM_KCPSTATUS=FALSE
```

```
STRUC TM_SUGG_T CHAR TM_ID[24]

STRUC TQM_TQDAT_T INT
T11,T12,T13,T14,T15,T16,T21,T22,T23,T24,T25,T26,K1,K2,K3,K4,K5,K6,O1,O2,ID,OVM,REAL
TMF


DECL TQM_TQDAT_T TMDEFAULT={T11 200,T12 200,T13 200,T14 200,T15 200,T16 200,T21 1000,T22
1000,T23 1000,T24 1000,T25 1000,T26 1000,K1 280,K2 280,K3 280,K4 280,K5 280,K6 280,O1
20,O2 30,ID 0,OVM 0,TMF 1.0}

DECL TQM_TQDAT_T TM0={T11 200,T12 200,T13 200,T14 200,T15 200,T16 200,T21 1000,T22
1000,T23 1000,T24 1000,T25 1000,T26 1000,K1 280,K2 280,K3 280,K4 280,K5 280,K6 280,O1
20,O2 30,ID 0,OVM 0,TMF 1.0}


DECL TQM_TQDAT_T TQM_DATA_C[8]

DECL TQM_TQDAT_T TQM_TEMP


;==================================

; External declarations:

;==================================

EXT  BAS (BAS_COMMAND  :IN,REAL  :IN )

EXT  IR_STOPM ( )


;====================

; Signal declarations

; Do not change !!!!!

;====================

SIGNAL CHANNEL_1 $ANOUT[1]

SIGNAL CHANNEL_2 $ANOUT[2]

SIGNAL CHANNEL_3 $ANOUT[3]

SIGNAL CHANNEL_4 $ANOUT[4]

SIGNAL CHANNEL_5 $ANOUT[5]

SIGNAL CHANNEL_6 $ANOUT[6]

SIGNAL CHANNEL_7 $ANOUT[7]

SIGNAL CHANNEL_8 $ANOUT[8]

SIGNAL CHANNEL_9 $ANOUT[9]

SIGNAL CHANNEL_10 $ANOUT[10]

SIGNAL CHANNEL_11 $ANOUT[11]

SIGNAL CHANNEL_12 $ANOUT[12]
SIGNAL CHANNEL_13 $ANOUT[13]
SIGNAL CHANNEL_14 $ANOUT[14]
SIGNAL CHANNEL_15 $ANOUT[15]
SIGNAL CHANNEL_16 $ANOUT[16]
SIGNAL CHANNEL_17 $ANOUT[17]
SIGNAL CHANNEL_18 $ANOUT[18]
SIGNAL CHANNEL_19 $ANOUT[19]
SIGNAL CHANNEL_20 $ANOUT[20]
SIGNAL CHANNEL_21 $ANOUT[21]
SIGNAL CHANNEL_22 $ANOUT[22]
SIGNAL CHANNEL_23 $ANOUT[23]
SIGNAL CHANNEL_24 $ANOUT[24]
SIGNAL CHANNEL_25 $ANOUT[25]
SIGNAL CHANNEL_26 $ANOUT[26]
SIGNAL CHANNEL_27 $ANOUT[27]
SIGNAL CHANNEL_28 $ANOUT[28]
SIGNAL CHANNEL_29 $ANOUT[29]
SIGNAL CHANNEL_30 $ANOUT[30]
SIGNAL CHANNEL_31 $ANOUT[31]
SIGNAL CHANNEL_32 $ANOUT[32]

;==================================
; ID external user log-on:
;==================================
SIGNAL ExtUserID $IN[1026]  TO $IN[1026]
SIGNAL ExtUserIDWatchDog $IN[1026]


;==================================
; Variables:
;==================================
DECL PDAT PDEFAULT={VEL 100.0,ACC 100.0,APO_DIST 100.0}
DECL LDAT LDEFAULT={VEL 2.0,ACC 100.0,APO_DIST 100.0,APO_FAC 50.0,ORI_TYP #VAR,CIRC_TYP #BASE,JERK_FAC 50.0,EXAX_IGN 0}
DECL LDAT CDEFAULT={CB {AUX_PT {ORI #CONSIDER,E1 #CONSIDER,E2 #CONSIDER,E3 #CONSIDER,E4 #CONSIDER,E5 #CONSIDER,E6 #CONSIDER},TARGET_PT {ORI #INTERPOLATE,E1 #INTERPOLATE,E2 #INTERPOLATE,E3 #INTERPOLATE,E4 #INTERPOLATE,E5 #INTERPOLATE,E6 #INTERPOLATE}}}
DECL FDAT FDEFAULT={TOOL_NO 1,BASE_NO 0,IPO_FRAME #BASE,POINT2[] " ",TQ_STATE FALSE}
DECL FDAT FHOME={TOOL_NO 2,BASE_NO 0,IPO_FRAME #BASE,POINT2[] " ",TQ_STATE FALSE}
DECL ODAT ODEFAULT={OUT_NO 1,STATE TRUE,PULSE_TIME 0.0,OUT_MODE #STOP_,TIME_DELAY 0.0,OFFSET 0.0}

DECL PDAT PDAT_ACT
DECL LDAT LDAT_ACT
DECL FDAT FDAT_ACT
DECL ODAT ODAT_ACT
DECL INT ACT_FILTER
DECL INT ACT_DELAY
DECL INT ACT_DISTANCE

DECL CHAR SPL_NAME[24]

;GROUP-Definitions
;---------------------------------
INT COMPL_GROUP='B0001'

INT DEF_GROUP[10]
DEF_GROUP[1]='B1111' ; complete
DEF_GROUP[2]='B0001' ; robot only
DEF_GROUP[3]='B0011' ; robot and track
DEF_GROUP[4]='B0111' ; robot, track and ext. axis2
DEF_GROUP[5]='B1011' ; robot, track and ext. axis3
DEF_GROUP[6]='B1111' ; robot, track and ext. axis2+3
DEF_GROUP[7]='B1111' ; complete
DEF_GROUP[8]='B1111' ; complete
DEF_GROUP[9]='B1111' ; complete
DEF_GROUP[10]='B1111' ; complete

INT $ACT_GROUP='B0001'

; HOME POSITION
;---------------------------------
E6AXIS XHOME={A1 -1.17017603,A2 -100.564301,A3 141.246002,A4 11.1277399,A5 -8.59637642,A6 28.8773003,E1 0.0,E2 0.0,E3 0.0,E4 0.0,E5 0.0,E6 0.0}
```

```
E6AXIS XHOME1={A1 -18.8755703,A2 -85.7914429,A3 111.3461,A4 0.0842175931,A5 61.3649101,A6 -198.138504,E1 0.0,E2 0.0,E3 0.0,E4 0.0,E5 0.0,E6 0.0}

E6AXIS XHOME2={A1 0.0,A2 -90.0,A3 90.0,A4 0.0,A5 0.0,A6 0.0,E1 0.0,E2 0.0,E3 0.0,E4 0.0,E5 0.0,E6 0.0}

E6AXIS XHOME3={A1 0.0,A2 -90.0,A3 90.0,A4 0.0,A5 0.0,A6 0.0,E1 0.0,E2 0.0,E3 0.0,E4 0.0,E5 0.0,E6 0.0}

E6AXIS XHOME4={A1 0.0,A2 -90.0,A3 90.0,A4 0.0,A5 0.0,A6 0.0,E1 0.0,E2 0.0,E3 0.0,E4 0.0,E5 0.0,E6 0.0}

E6AXIS XHOME5={A1 0.0,A2 -90.0,A3 90.0,A4 0.0,A5 0.0,A6 0.0,E1 0.0,E2 0.0,E3 0.0,E4 0.0,E5 0.0,E6 0.0}


; REFERENCE POINTS

;-----------------------------------

DECL FRAME REF_PT[16]

REF_PT[1]={X 1078.5365,Y 489.396606,Z 466.818024,A -93.7216187,B -47.4161453,C 8.57072639}

REF_PT[2]={X 1154.42749,Y -179.490707,Z 393.444763,A -111.84304,B 53.7387619,C -23.9969902}

REF_PT[3]={X 838.096375,Y 26.5339928,Z 12.7995329,A -71.979332,B 21.460619,C -75.1847076}

REF_PT[4]={X 842.447205,Y 25.0927582,Z 12.0079546,A -113.674377,B 47.4745102,C -77.7672882}

REF_PT[5]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[6]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[7]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[8]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[9]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[10]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[11]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[12]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[13]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[14]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[15]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

REF_PT[16]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}


FRAME REF_TOOL={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}


; TOOL and BASE data

;-----------------------------------

BOOL AUTO_IPO_M=FALSE
BOOL STOPM_FLAG=FALSE
BOOL TOOL_CORR_ON=FALSE
BOOL TOOL_CORR_W_ON=FALSE
BOOL BASE_CORR_ON=FALSE
BOOL M_BAS_COR_ON=FALSE
FRAME TOOL_CORR={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
FRAME TOOL_CORR_W={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
FRAME BASE_CORR={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
FRAME M_BASE_CORR={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
INT COR_TOOL_NO=0

INT MAX_TOOL=16
DECL FRAME TOOL_DATA[16]
TOOL_DATA[1]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[2]={X -3.46700001,Y 1.99199998,Z 336.859009,A 1.00199997,B -37.9485016,C 101.487099}
TOOL_DATA[3]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[4]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[5]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[6]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[7]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[8]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[9]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[10]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[11]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[12]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[13]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[14]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[15]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
TOOL_DATA[16]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

DECL CHAR TOOL_NAME[16,24]
TOOL_NAME[1,]="Flens A6"
TOOL_NAME[2,]="welding_torch"
TOOL_NAME[3,]="       "
TOOL_NAME[4,]="        "
```

```
TOOL_NAME[5,]=" "
TOOL_NAME[6,]=" "
TOOL_NAME[7,]=" "
TOOL_NAME[8,]=" "
TOOL_NAME[9,]=" "
TOOL_NAME[10,]=" "
TOOL_NAME[11,]=" "
TOOL_NAME[12,]=" "
TOOL_NAME[13,]=" "
TOOL_NAME[14,]=" "
TOOL_NAME[15,]=" "
TOOL_NAME[16,]=" "

DECL LOAD LOAD_DATA[16]
LOAD_DATA[1]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[2]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[3]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[4]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[5]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[6]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[7]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[8]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[9]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[10]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[11]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[12]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[13]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[14]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[15]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}
LOAD_DATA[16]={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z 0.0}}

DECL IPO_M_T TOOL_TYPE[16]
TOOL_TYPE[1]=#BASE
TOOL_TYPE[2]=#BASE
TOOL_TYPE[3]=#BASE
TOOL_TYPE[4]=#BASE
```

```
TOOL_TYPE[5]=#NONE

TOOL_TYPE[6]=#NONE

TOOL_TYPE[7]=#NONE

TOOL_TYPE[8]=#NONE

TOOL_TYPE[9]=#NONE

TOOL_TYPE[10]=#NONE

TOOL_TYPE[11]=#NONE

TOOL_TYPE[12]=#NONE

TOOL_TYPE[13]=#NONE

TOOL_TYPE[14]=#NONE

TOOL_TYPE[15]=#NONE

TOOL_TYPE[16]=#NONE


DECL LOAD LOAD_A1_DATA={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z
0.0}}

DECL LOAD LOAD_A2_DATA={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z
0.0}}

DECL LOAD LOAD_A3_DATA={M -1.0,CM {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},J {X 0.0,Y 0.0,Z
0.0}}


INT MAX_BASE=32

DECL FRAME BASE_DATA[32]

BASE_DATA[1]={X  919.546997,Y  244.759003,Z  151.716003,A  1.11899996,B  -0.726999998,C
0.69599998}

BASE_DATA[2]={X 990.924194,Y -120.935097,Z 153.891602,A 91.0651016,B 9.99999975E-05,C
-179.990204}

BASE_DATA[3]={X 990.665405,Y -120.572899,Z 157.261703,A 0.983200014,B -0.00760000013,C
0.00319999992}

BASE_DATA[4]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

BASE_DATA[5]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

BASE_DATA[6]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

BASE_DATA[7]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

BASE_DATA[8]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

BASE_DATA[9]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

BASE_DATA[10]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

BASE_DATA[11]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

BASE_DATA[12]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

BASE_DATA[13]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
```

```
BASE_DATA[14]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[15]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[16]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[17]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[18]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[19]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[20]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[21]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[22]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[23]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[24]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[25]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[26]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[27]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[28]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[29]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[30]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[31]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}
BASE_DATA[32]={X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0}

DECL CHAR BASE_NAME[32,24]
BASE_NAME[1,]="pallet      "
BASE_NAME[2,]="bakjeGereedschappen"
BASE_NAME[3,]="bakjeGereedschappen2"
BASE_NAME[4,]=" "
BASE_NAME[5,]=" "
BASE_NAME[6,]=" "
BASE_NAME[7,]=" "
BASE_NAME[8,]=" "
BASE_NAME[9,]=" "
BASE_NAME[10,]=" "
BASE_NAME[11,]=" "
BASE_NAME[12,]=" "
BASE_NAME[13,]=" "
BASE_NAME[14,]=" "
BASE_NAME[15,]=" "
```

```
BASE_NAME[16,]=" "

BASE_NAME[17,]=" "

BASE_NAME[18,]=" "

BASE_NAME[19,]=" "

BASE_NAME[20,]=" "

BASE_NAME[21,]=" "

BASE_NAME[22,]=" "

BASE_NAME[23,]=" "

BASE_NAME[24,]=" "

BASE_NAME[25,]=" "

BASE_NAME[26,]=" "

BASE_NAME[27,]=" "

BASE_NAME[28,]=" "

BASE_NAME[29,]=" "

BASE_NAME[30,]=" "

BASE_NAME[31,]=" "

BASE_NAME[32,]=" "


DECL IPO_M_T BASE_TYPE[32]

BASE_TYPE[1]=#BASE

BASE_TYPE[2]=#BASE

BASE_TYPE[3]=#BASE

BASE_TYPE[4]=#NONE

BASE_TYPE[5]=#NONE

BASE_TYPE[6]=#NONE

BASE_TYPE[7]=#NONE

BASE_TYPE[8]=#NONE

BASE_TYPE[9]=#NONE

BASE_TYPE[10]=#NONE

BASE_TYPE[11]=#NONE

BASE_TYPE[12]=#NONE

BASE_TYPE[13]=#NONE

BASE_TYPE[14]=#NONE

BASE_TYPE[15]=#NONE

BASE_TYPE[16]=#NONE

BASE_TYPE[17]=#NONE
```

```
BASE_TYPE[18]=#NONE
BASE_TYPE[19]=#NONE
BASE_TYPE[20]=#NONE
BASE_TYPE[21]=#NONE
BASE_TYPE[22]=#NONE
BASE_TYPE[23]=#NONE
BASE_TYPE[24]=#NONE
BASE_TYPE[25]=#NONE
BASE_TYPE[26]=#NONE
BASE_TYPE[27]=#NONE
BASE_TYPE[28]=#NONE
BASE_TYPE[29]=#NONE
BASE_TYPE[30]=#NONE
BASE_TYPE[31]=#NONE
BASE_TYPE[32]=#NONE

;******************************************
; Variables for CELL DEFINITION
;******************************************

INT MAX_MACHINES=16
DECL MACHINE_DEF_T MACHINE_DEF[16]
MACHINE_DEF[1]={NAME[] "Robot",COOP_KRC_INDEX 1,PARENT[] "WORLD",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #ROBOT,GEOMETRY[] " "}
MACHINE_DEF[2]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[3]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[4]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[5]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[6]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[7]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[8]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[9]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
```

```
MACHINE_DEF[10]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[11]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[12]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[13]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[14]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[15]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}
MACHINE_DEF[16]={NAME[] " ",COOP_KRC_INDEX 0,PARENT[] " ",ROOT {X 0.0,Y 0.0,Z 0.0,A 0.0,B 0.0,C 0.0},MECH_TYPE #NONE,GEOMETRY[] " "}

DECL MACHINE_TOOL_T MACHINE_TOOL_DAT[16]
MACHINE_TOOL_DAT[1]={MACH_DEF_INDEX 1,PARENT[] "Robot",GEOMETRY[] " "}
MACHINE_TOOL_DAT[2]={MACH_DEF_INDEX 1,PARENT[] "Robot",GEOMETRY[] " "}
MACHINE_TOOL_DAT[3]={MACH_DEF_INDEX 1,PARENT[] "Robot",GEOMETRY[] " "}
MACHINE_TOOL_DAT[4]={MACH_DEF_INDEX 1,PARENT[] "Robot",GEOMETRY[] " "}
MACHINE_TOOL_DAT[5]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[6]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[7]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[8]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[9]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[10]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[11]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[12]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[13]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[14]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[15]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_TOOL_DAT[16]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

DECL MACHINE_FRAME_T MACHINE_FRAME_DAT[32]
MACHINE_FRAME_DAT[1]={MACH_DEF_INDEX 0,PARENT[] "WORLD",GEOMETRY[] " "}
MACHINE_FRAME_DAT[2]={MACH_DEF_INDEX 0,PARENT[] "WORLD",GEOMETRY[] " "}
MACHINE_FRAME_DAT[3]={MACH_DEF_INDEX 0,PARENT[] "WORLD",GEOMETRY[] " "}
MACHINE_FRAME_DAT[4]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
MACHINE_FRAME_DAT[5]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}
```

```
MACHINE_FRAME_DAT[6]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[7]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[8]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[9]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[10]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[11]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[12]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[13]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[14]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[15]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[16]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[17]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[18]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[19]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[20]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[21]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[22]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[23]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[24]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[25]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[26]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[27]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[28]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[29]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[30]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[31]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}

MACHINE_FRAME_DAT[32]={MACH_DEF_INDEX 0,PARENT[] " ",GEOMETRY[] " "}


;*****************************************

; Variables for axis CALIBRATING

;*****************************************

DECL INT CAL_AXIS

BOOL CONST_SPEED


;*****************************************

; Variables for added TechPackages
```

```
;******************************************

INT POWER

;******************************************
; Variables for TOUCHUP user limit
;******************************************
BOOL UM_TOUCHUP=FALSE
REAL CARTESIAN_TOL=5.0
REAL ROTATION_ANGLE=10.0
REAL EXTAX_TOL[6]
EXTAX_TOL[1]=10.0
EXTAX_TOL[2]=10.0
EXTAX_TOL[3]=10.0
EXTAX_TOL[4]=10.0
EXTAX_TOL[5]=10.0
EXTAX_TOL[6]=10.0

;******************************************
; Variables for Displaying and Konfiguring
;******************************************
DECL INT I[20]

I[1]=0
I[2]=0
I[3]=0
I[4]=0
I[5]=0
I[6]=0
I[7]=0
I[8]=0
I[9]=0
I[10]=0
I[11]=0
I[12]=0
I[13]=0
```

```
I[14]=0

I[15]=0

I[16]=0

I[17]=0

I[18]=0

I[19]=0

I[20]=0


;******************************************

; all for InterBus Mapping on optional segments

;******************************************


DECL INT IBUS_SEGMENT[16]

IBUS_SEGMENT[1]=0

IBUS_SEGMENT[2]=0

IBUS_SEGMENT[3]=0

IBUS_SEGMENT[4]=0

IBUS_SEGMENT[5]=0

IBUS_SEGMENT[6]=0

IBUS_SEGMENT[7]=0

IBUS_SEGMENT[8]=0

IBUS_SEGMENT[9]=0

IBUS_SEGMENT[10]=0

IBUS_SEGMENT[11]=0

IBUS_SEGMENT[12]=0

IBUS_SEGMENT[13]=0

IBUS_SEGMENT[14]=0

IBUS_SEGMENT[15]=0

IBUS_SEGMENT[16]=0

;ENDFOLD (BASISTECH GLOBALS)

;FOLD AUTOEXT GLOBALS

;==================================

; Structures:

;==================================

ENUM FUNCT_TYPE PGNO_GET,PGNO_ACKN,PGNO_FAULT

ENUM P00_COMMAND INIT_EXT,EXT_PGNO,CHK_HOME,EXT_ERR
```

```
STRUC SPS_PROG_TYPE INT PROG_NR,CHAR PROG_NAME[12]
;==================================
; External declarations:
;==================================
EXT  P00 (P00_COMMAND  :IN,FUNCT_TYPE  :IN,CHAR [],INT  :IN )
;External declaration  for  Submit controlled AutoExt
EXT  P00_SUBM (P00_COMMAND  :IN,FUNCT_TYPE  :IN )

;==================================
; Variables:
;==================================
; Variables for internal
; Communication:
;----------------------------------
BOOL ERROR_FLAG
BOOL CHECK_HOME=TRUE
BOOL PROG_CONTROL=FALSE
DECL CHAR PRO_NAME1_L[8]
PRO_NAME1_L[]="        "
DECL CHAR PRO_NAME1_A[8]
PRO_NAME1_A[]="        "

INT PGNO=0 ;copy of ext. pgno
INT PGNO_ERROR=0 ;transmission error
INT PGNO_TYPE=1 ;coding type of ext. pgno
INT REFLECT_PROG_NR=0 ; enable mirroring of program number inputs (1=enabled, 0=disabled)

; Variables for External
; Communication: I/O-Interface
;----------------------------------
INT PGNO_FBIT=33 ;first bit of ext. pgno input $IN[]
INT PGNO_FBIT_REFL=999 ;first bit of ext. pgno reflection output $OUT[]
INT PGNO_LENGTH=8 ;length of ext. pgno (max. 16)
INT PGNO_PARITY=41 ;parity bit of ext. pgno
INT PGNO_REQ=33 ;request ext. pgno input

INT PGNO_VALID=42 ;validate ext. pgno input

INT APPL_RUN=34 ;application program is running output

INT ERR_TO_PLC=35 ;error output to PLC

INT P01_STEP

INT CHK_STEP

INT PGNO_FLAG


; Table for assign SPS program number to program name

INT MAX_SPS_PROG=12

DECL SPS_PROG_TYPE SPS_PROG[12]

SPS_PROG[1]={PROG_NR 1,PROG_NAME[] "HP01()    "}

SPS_PROG[2]={PROG_NR 2,PROG_NAME[] "HP02()    "}

SPS_PROG[3]={PROG_NR 3,PROG_NAME[] "HP03()    "}

SPS_PROG[4]={PROG_NR 4,PROG_NAME[] "HP04()    "}

SPS_PROG[5]={PROG_NR 5,PROG_NAME[] "HP05()    "}

SPS_PROG[6]={PROG_NR 6,PROG_NAME[] "HP06()    "}

SPS_PROG[7]={PROG_NR 7,PROG_NAME[] "HP07()    "}

SPS_PROG[8]={PROG_NR 8,PROG_NAME[] "HP08()    "}

SPS_PROG[9]={PROG_NR 9,PROG_NAME[] "HP09()    "}

SPS_PROG[10]={PROG_NR 10,PROG_NAME[] "HP10()    "}

SPS_PROG[11]={PROG_NR 62,PROG_NAME[] "HP62()    "}

SPS_PROG[12]={PROG_NR 63,PROG_NAME[] "HP63()    "}

DECL CHAR TMPNAME[128]

TMPNAME[]="                                "

;ENDFOLD (AUTOEXT GLOBALS)

;FOLD Applications GLOBALS

;******************************************************

; Package :  Applications Version 1.0

; Function: Global definitions

; Date:   : 26/06/2008

; DEF Applications BLOCK

;******************************************************

;    Variable nomenclatur:

; APP_  = general Applications variable

; APP_TM_ = applications transfermanager specific variable
; APP_JLM_ = applications joblistmanager specific variable
; XX_b = bool
; XX_i = integer
; XX_e = enumeration
; XX_r = real
; XX_s = string
; XX_f = frame
; I_   = input
; O_   = output
; AO_  = analog output

;=====================================
; STRINGS
;=====================================
DECL CHAR APP_sDATA_PATH[200]
APP_sDATA_PATH[]="D:\ApplicationsData\"
DECL CHAR APP_sROB_PROG_PATH[200]
APP_sROB_PROG_PATH[]="C:\KRC\Roboter\KRC\R1\Program\"
DECL CHAR APP_sHTML_JOBINFO[200]
APP_sHTML_JOBINFO[]=" "

DECL CHAR APP_TM_sSOURCE_PATH[260]
APP_TM_sSOURCE_PATH[]="U:\KUKA_programma\08-22 met hoogte\"
DECL CHAR APP_TM_sPROJECTNAME[200]
APP_TM_sPROJECTNAME[]=" "

DECL CHAR APP_JLM_sLASTJOBLIST[200]
APP_JLM_sLASTJOBLIST[]="My_Joblist.jls  "

DECL CHAR APP_JLM_sTEMPLATE_PATH[260]
APP_JLM_sTEMPLATE_PATH[]="C:\KRC\TP\Applications\LIB\Templates\sw5_2\"

;=====================================

;=====================================
; INTEGERS
```

```
;=====================================
INT APP_iLAST_PSTATE1=1
;=====================================

;=====================================
; BOOLS
;=====================================
DECL BOOL APP_bMANUAL=TRUE ; KCP is in manual mode
DECL BOOL APP_bSAW=FALSE ; Line select is pressed
DECL BOOL APP_bCANCEL=FALSE ; Program Cancel is pressed
DECL BOOL APP_bRESET=FALSE ; Program Reset is pressed
DECL BOOL APP_bSTARTED=FALSE ; Program is started
DECL BOOL APP_bSTOPPED=FALSE ; Program is stoped
DECL BOOL APP_bFINISHED=FALSE ; Program is finished
DECL BOOL APP_bResult=TRUE
DECL BOOL APP_bJobListLoaded=FALSE ; Set to true on joblist loading, must be resetted by user KRL.
;=====================================

;=====================================
; Network socket
;=====================================
BOOL APP_bTCP_ENABLED=FALSE
BOOL APP_bTCP_LOG_ON=FALSE
BOOL APP_bUDP_ENABLED=TRUE
BOOL APP_bUDP_LOG_ON=FALSE
INT APP_iSOCKET_NIC_INDEX=1
INT APP_iTCP_PORT=9901
INT APP_iUDP_PORT=9902
INT APP_iTCP_STATE=0 ;0=Disabled 1=Listening 2=Connected 3=Error
DECL CHAR APP_sSOCKET_JOBLISTNAME[200]
APP_sSOCKET_JOBLISTNAME[]="SocketJobList"
;=====================================

;ENDFOLD (Applications GLOBALS)
;FOLD ARCTECHDIGITAL GLOBALS
```

```
;FOLD Structures and Definitions
ENUM A_BOOL_T_D ACTIVE,IDLE
ENUM A_FG_TYPE_D FG_MECHANIC,FG_THERMAL
STRUC FCT_IN_T_D INT NO,BOOL STATE
STRUC WELD_ST INT PRG_NO,REAL VELOCITY,START_TIME,INT WEAVFIG_MECH,REAL
WEAVLEN_MECH,WEAVAMP_MECH,WEAVANG_MECH,END_TIME
STRUC WELD_FI INT PRG_NO,REAL VELOCITY,INT WEAVFIG_MECH,REAL
WEAVLEN_MECH,WEAVAMP_MECH,WEAVANG_MECH,END_TIME
STRUC A20_SUGG_T CHAR WELD_MODE[24],CHAR WELD_SET[24],CHAR START_SET[24],CHAR
END_SET[24],CHAR COMMENT[24],CHAR PRG_NO[24],CHAR V_ARC[24],CHAR PGNO[24]
EXT  FLT_SERV_D (A_BOOL_T_D  :IN,A_BOOL_T_D  :IN ) ; A_WLD_ACTIVE,ARC_ON_ERROR
EXT  A20 (INT  :IN,WELD_ST  :IN,WELD_FI  :IN,INT  :IN )
EXT  WEAV_DEF_D (A_FG_TYPE_D  :IN,INT  :IN,INT  :IN,INT  :IN,INT  :IN )
;ENDFOLD (Structures and Definitions)
;FOLD Main Options
BOOL ARC20=TRUE ;Techpackage option
BOOL A_PROC_IN_T1=FALSE ;Active process in #T1 mode operation
BOOL A_PROC_ENABLE=TRUE ;Active process enable

INT A_CTRL_DELAY_D=1200 ;[ms] Delay to ignore critical ignition
INT A_FLT_SV_FCT_D=0 ;Number of user defined FLT_SERV_D-Subroutine

REAL A_AP_DIS_TCH_D=5.0 ;[mm] ARCTECH specified rough positioning parameter
REAL A_DRY_RN_VEL=0.150000006 ;Maximum speeed of DRY RUN  functionality
SIGNAL ARCD_PrgNr $OUT[33]  TO $OUT[36] ;{FIRST_BIT 40,LENGTH 4
;ENDFOLD (Main Options)
;FOLD ArcTech Outputs
DECL CTRL_OUT_T O_WELD_CTRL[15]
O_WELD_CTRL[1]={OUT_NR 17,INI FALSE,NAME_NAT[] "Start Welding       "}
O_WELD_CTRL[2]={OUT_NR 18,INI TRUE,NAME_NAT[] "Robot in Position   "}
O_WELD_CTRL[3]={OUT_NR 0,INI FALSE,NAME_NAT[] "Trigger Program Nr  "}
O_WELD_CTRL[4]={OUT_NR 28,INI FALSE,NAME_NAT[] "Acknowledge Errors  "}
O_WELD_CTRL[5]={OUT_NR 30,INI FALSE,NAME_NAT[] "Sneezer             "}
O_WELD_CTRL[6]={OUT_NR 2,INI FALSE,NAME_NAT[] "Cleaner Motor       "}
O_WELD_CTRL[7]={OUT_NR 0,INI FALSE,NAME_NAT[] "Sprayer             "}
O_WELD_CTRL[8]={OUT_NR 0,INI FALSE,NAME_NAT[] "                    "}
O_WELD_CTRL[9]={OUT_NR 25,INI FALSE,NAME_NAT[] "Gas purge           "}
```

```
O_WELD_CTRL[10]={OUT_NR 0,INI FALSE,NAME_NAT[] "Weld Error Signal   "}
O_WELD_CTRL[11]={OUT_NR 0,INI FALSE,NAME_NAT[] "                    "}
O_WELD_CTRL[12]={OUT_NR 0,INI FALSE,NAME_NAT[] "Target Point reached"}
O_WELD_CTRL[13]={OUT_NR 0,INI FALSE,NAME_NAT[] "Ignition ErrorSignal"}
O_WELD_CTRL[14]={OUT_NR 26,INI FALSE,NAME_NAT[] "WireFeed +          "}
O_WELD_CTRL[15]={OUT_NR 27,INI FALSE,NAME_NAT[] "WireFeed -          "}
;ENDFOLD (ArcTech Outputs)
;FOLD ArcTech Inputs
DECL CTRL_IN_T I_WELD_CTRL[15]
I_WELD_CTRL[1]={IN_NR 23,NAME_NAT[] "Power Supply Ready  "}
I_WELD_CTRL[2]={IN_NR 20,NAME_NAT[] "Current available   "}
I_WELD_CTRL[3]={IN_NR 48,NAME_NAT[] "Current over        "}
I_WELD_CTRL[4]={IN_NR 0,NAME_NAT[] "Wire available      "}
I_WELD_CTRL[5]={IN_NR 0,NAME_NAT[] "Water flow okay     "}
I_WELD_CTRL[6]={IN_NR 19,NAME_NAT[] "Key switch welding  "}
I_WELD_CTRL[7]={IN_NR 0,NAME_NAT[] "                    "}
I_WELD_CTRL[8]={IN_NR 0,NAME_NAT[] "                    "}
I_WELD_CTRL[9]={IN_NR 0,NAME_NAT[] "                    "}
I_WELD_CTRL[10]={IN_NR 0,NAME_NAT[] "                    "}
I_WELD_CTRL[11]={IN_NR 0,NAME_NAT[] "                    "}
I_WELD_CTRL[12]={IN_NR 0,NAME_NAT[] "                    "}
I_WELD_CTRL[13]={IN_NR 0,NAME_NAT[] "                    "}
I_WELD_CTRL[14]={IN_NR 0,NAME_NAT[] "                    "}
I_WELD_CTRL[15]={IN_NR 0,NAME_NAT[] "                    "}
;ENDFOLD (ArcTech Inputs)
;FOLD Default Datasets
DECL WELD_ST ADEFAULT={PRG_NO 1,VELOCITY 0.600000024,START_TIME
0.0120000001,WEAVFIG_MECH 0,WEAVLEN_MECH 4.0,WEAVAMP_MECH 1.5,WEAVANG_MECH
0.0,END_TIME 0.0}
DECL WELD_FI MDEFAULT={PRG_NO 1,VELOCITY 0.600000024,WEAVFIG_MECH 0,WEAVLEN_MECH
4.0,WEAVAMP_MECH 1.5,WEAVANG_MECH 0.0,END_TIME 0.0149999997}
;ENDFOLD (Default Datasets)
;FOLD Statuskey Variables
DECL A_BOOL_T_D A_HOT_WELD_D=#IDLE ; Statuskey Hot weld
DECL A_BOOL_T_D A_HPU_DRY_RN_D=#IDLE ; Statuskey Dry Run
INT A_HPU_WFD_D=0 ; Statuskey HPU wirefeed
;ENDFOLD (Statuskey Variables)
```

```
;FOLD Peripheral Output groups

; outputs weld start

DECL FCT_OUT_T O_WELD_START[3]

O_WELD_START[1]={NO 1,PULS_TIME 0.0,STATE TRUE} ;This pointer has to point to O_WELD_CTRL[1] ==>O_WELD_START[1]NO 1!

O_WELD_START[2]={NO 0,PULS_TIME 0.0,STATE TRUE}

O_WELD_START[3]={NO 0,PULS_TIME 0.0,STATE TRUE}


; outputs acknowledge start move

DECL FCT_OUT_T O_ACK_START[3]

O_ACK_START[1]={NO 2,PULS_TIME 0.300000012,STATE TRUE}

O_ACK_START[2]={NO 0,PULS_TIME 0.0,STATE TRUE}

O_ACK_START[3]={NO 0,PULS_TIME 0.0,STATE TRUE}


; strobe program number

DECL FCT_OUT_T O_STROB_PGNO[3]

O_STROB_PGNO[1]={NO 3,PULS_TIME 0.300000012,STATE TRUE}

O_STROB_PGNO[2]={NO 0,PULS_TIME 0.300000012,STATE TRUE}

O_STROB_PGNO[3]={NO 0,PULS_TIME 0.300000012,STATE TRUE}


; outputs weld end

DECL FCT_OUT_T O_SEAM_END[3]

O_SEAM_END[1]={NO 1,PULS_TIME 0.0,STATE FALSE}

O_SEAM_END[2]={NO 0,PULS_TIME 0.0,STATE FALSE}

O_SEAM_END[3]={NO 0,PULS_TIME 0.0,STATE FALSE}


; outputs acknowledge weld_end

DECL FCT_OUT_T O_ACK_WELD_E[3]

O_ACK_WELD_E[1]={NO 12,PULS_TIME 0.5,STATE FALSE}

O_ACK_WELD_E[2]={NO 0,PULS_TIME 0.5,STATE TRUE}

O_ACK_WELD_E[3]={NO 0,PULS_TIME 0.5,STATE TRUE}


; outputs fault while arc on

DECL FCT_OUT_T O_FLT_ARC_ON[3]

O_FLT_ARC_ON[1]={NO 13,PULS_TIME 0.0,STATE TRUE}

O_FLT_ARC_ON[2]={NO 1,PULS_TIME 0.0,STATE FALSE}
```

```
O_FLT_ARC_ON[3]={NO 10,PULS_TIME 0.0,STATE TRUE}


; outputs stop weld after interrupt on seam
DECL FCT_OUT_T O_FLT_WELD[3]
O_FLT_WELD[1]={NO 1,PULS_TIME 0.0,STATE FALSE}
O_FLT_WELD[2]={NO 10,PULS_TIME 0.0,STATE TRUE}
O_FLT_WELD[3]={NO 0,PULS_TIME 0.0,STATE FALSE}


; outputs stop cleaner after interrupt
DECL FCT_OUT_T O_FLT_CLEAN[3]
O_FLT_CLEAN[1]={NO 6,PULS_TIME 0.0,STATE FALSE}
O_FLT_CLEAN[2]={NO 0,PULS_TIME 0.0,STATE FALSE}
O_FLT_CLEAN[3]={NO 0,PULS_TIME 0.0,STATE FALSE}


; outputs acknowledge fault
DECL FCT_OUT_T O_ACK_FLT[3]
O_ACK_FLT[1]={NO 4,PULS_TIME 0.5,STATE TRUE}
O_ACK_FLT[2]={NO 13,PULS_TIME 0.0,STATE FALSE}
O_ACK_FLT[3]={NO 10,PULS_TIME 0.0,STATE FALSE}


; output for fault message
DECL FCT_OUT_T O_FLT_SIGNAL={NO 10,PULS_TIME 0.0,STATE TRUE}


;Wirefeed control
DECL FCT_OUT_T O_WRFEDP={NO 14,PULS_TIME 0.200000003,STATE TRUE}
DECL FCT_OUT_T O_WRFEDN={NO 15,PULS_TIME 0.200000003,STATE TRUE}
;ENDFOLD (Peripheral Output groups)
;FOLD Peripheral Input groups
; inputs as condition befor weld can start
DECL FCT_IN_T_D I_WELD_COND[3]
I_WELD_COND[1]={NO 1,STATE TRUE}
I_WELD_COND[2]={NO 0,STATE TRUE}
I_WELD_COND[3]={NO 0,STATE TRUE}


; inputs start moving
DECL FCT_IN_T_D I_START_MOVE[3]
```

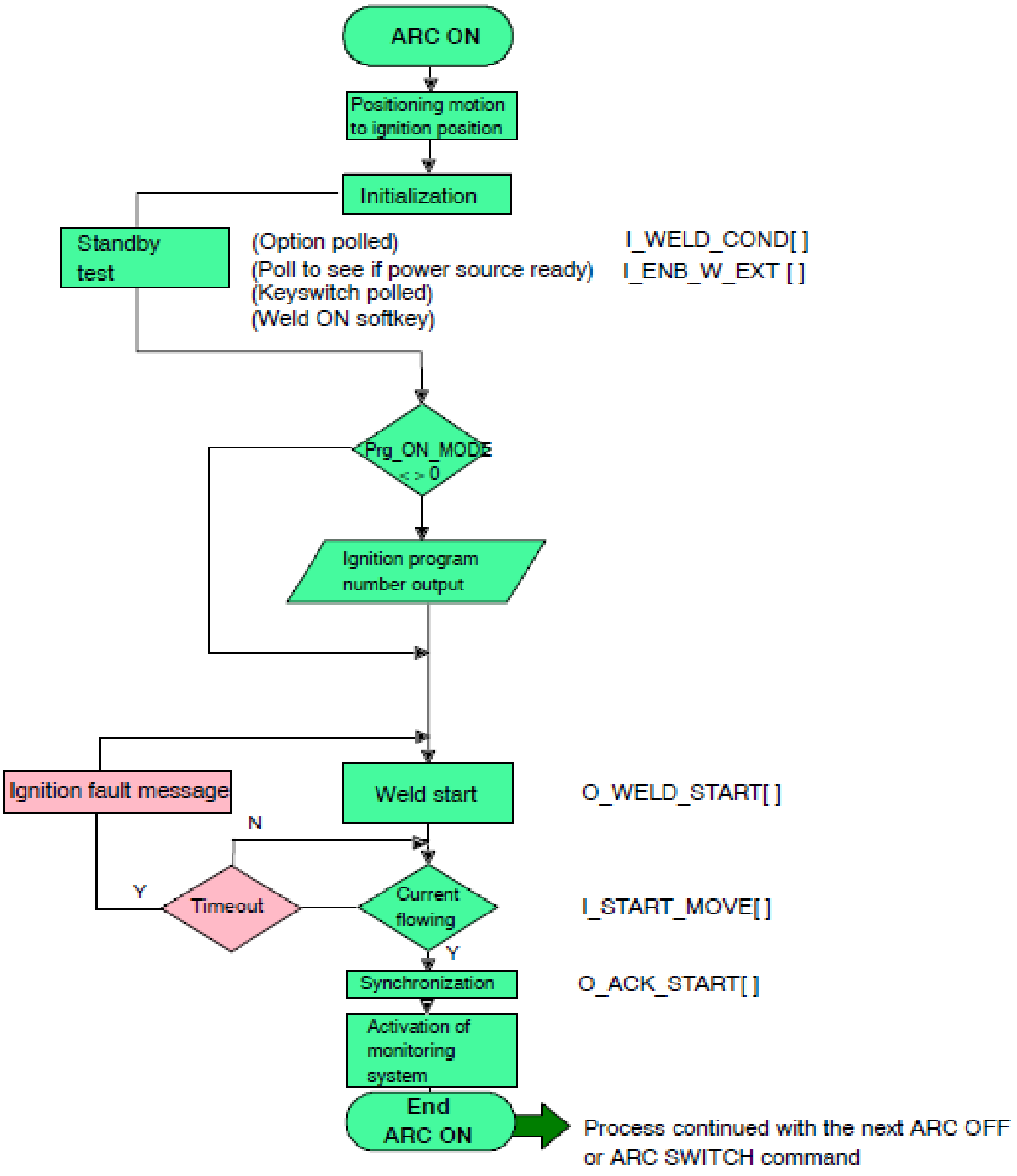
```
I_START_MOVE[1]={NO 2,STATE TRUE}
I_START_MOVE[2]={NO 0,STATE TRUE}
I_START_MOVE[3]={NO 0,STATE TRUE}

; inputs weld is ended
DECL FCT_IN_T_D I_WELD_END[3]
I_WELD_END[1]={NO 2,STATE FALSE}
I_WELD_END[2]={NO 0,STATE FALSE}
I_WELD_END[3]={NO 0,STATE FALSE}

;Counter for Interrupt definition
DECL INT FLT_NUM[3]

; inputs telling faults while welding
DECL FCT_IN_T_D I_WELD_FLT[3]
I_WELD_FLT[1]={NO 2,STATE TRUE}
I_WELD_FLT[2]={NO 4,STATE TRUE}
I_WELD_FLT[3]={NO 5,STATE TRUE}

; input for external enable weld
DECL FCT_IN_T_D I_ENB_W_EXT={NO 6,STATE FALSE}
;ENDFOLD (Peripheral Input groups)
;FOLD Internal System States
DECL WELD_FI FI_ACT={PRG_NO 4,VELOCITY 0.319999993,WEAVFIG_MECH 0,WEAVLEN_MECH 4.0,WEAVAMP_MECH 1.5,WEAVANG_MECH 0.0,END_TIME 0.0149999997}
DECL WELD_ST ST_ACT={PRG_NO 4,VELOCITY 0.319999993,START_TIME 0.0120000001,WEAVFIG_MECH 0,WEAVLEN_MECH 4.0,WEAVAMP_MECH 1.5,WEAVANG_MECH 0.0,END_TIME 0.0149999997}

BOOL B_RESTART=FALSE
BOOL FLY_WELD=FALSE
BOOL WELD_ACTIVE=FALSE
BOOL T1_L
BOOL T_STOP=FALSE ;flag for $CYCFLAG to avoid
BOOL TECH_MOTION_D=FALSE ; flag for technology motions

;current instruction
INT ARC_INSTR_D=0
```

```
; instruction code

INT ARC_INI=1

INT ARC_ON=2

INT ARC_OFF=3

INT ARC_SWI_F=4

INT ARC_SWIP=5

INT TECH_STOP1=6

INT TECH_STOP2=7

INT TECH_STOP=9

INT ARC_OFF_P=10

INT ARC_OFF_V=12

INT ARC_START=13

INT ARC_SWI_T=14

INT ARC_OFF_T=15

INT PRGNO=1

INT TMP_PRG=0


INT A_FG_MECH1_D=1 ; Mechanical function generator 1 called also in bas.src

INT A_FG_MECH2_D=2 ; Mechanical function generator 2 called also in bas.src


INT A_WEAV_GEN_D[6]

A_WEAV_GEN_D[1]=1 ; A_FG_MECH1

A_WEAV_GEN_D[2]=2 ; A_FG_MECH2

A_WEAV_GEN_D[3]=3 ; A_FG_THERM

A_WEAV_GEN_D[4]=4 ; A_FG_SENSOR


INT SAVE_OVR_D=0 ; prepared to online optimizing

INT A_LST_M1_FIG_D=0

INT A_LST_M2_FIG_D=0

INT ARCD_WeldStartIndex=17 ;Additional mechanism to avoid weld in T1


;ENDFOLD (Internal System States)

;ENDFOLD (ARCTECHDIGITAL GLOBALS)

;FOLD USER GLOBALS

;*******************************************

;Make your modifications -ONLY- here
```

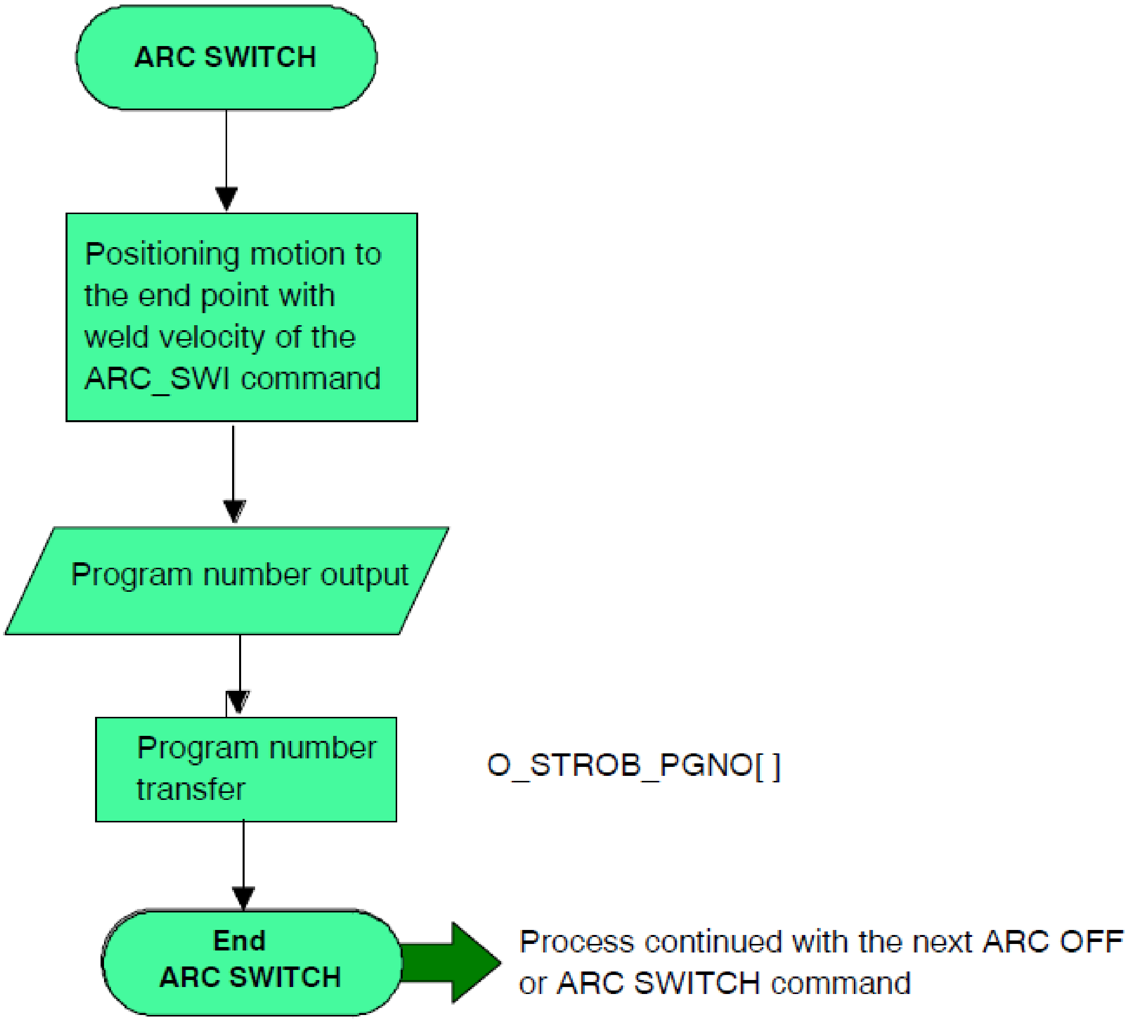
```
;*******************************************
;==================================
; Userdefined Types
;==================================

;==================================
; Userdefined Externals
;==================================

;==================================
; Userdefined Variables
;==================================
;serial port vars
INT HANDLE,OFFSET
REAL TIMEOUT
REAL R_VALUE
BOOL B_VALUE
DECL STATE_T SR_T,SW_T,SC_T
DECL MODUS_T MR_T,MW_T

;Mathys stapelpatroon vars
DECL INT POT_HOOGTE
DECL INT POT_DIAMETER
DECL INT AANTAL_LAGEN
DECL BOOL TUSSENVEL

DECL INT pallet_breedte,pallet_lengte,pallet_hoogte,pot_r
DECL INT aantal_breedte,aantal_lengte


;ENDFOLD (USER GLOBALS)
ENDDAT
```

A2. KUKA.ArcTech

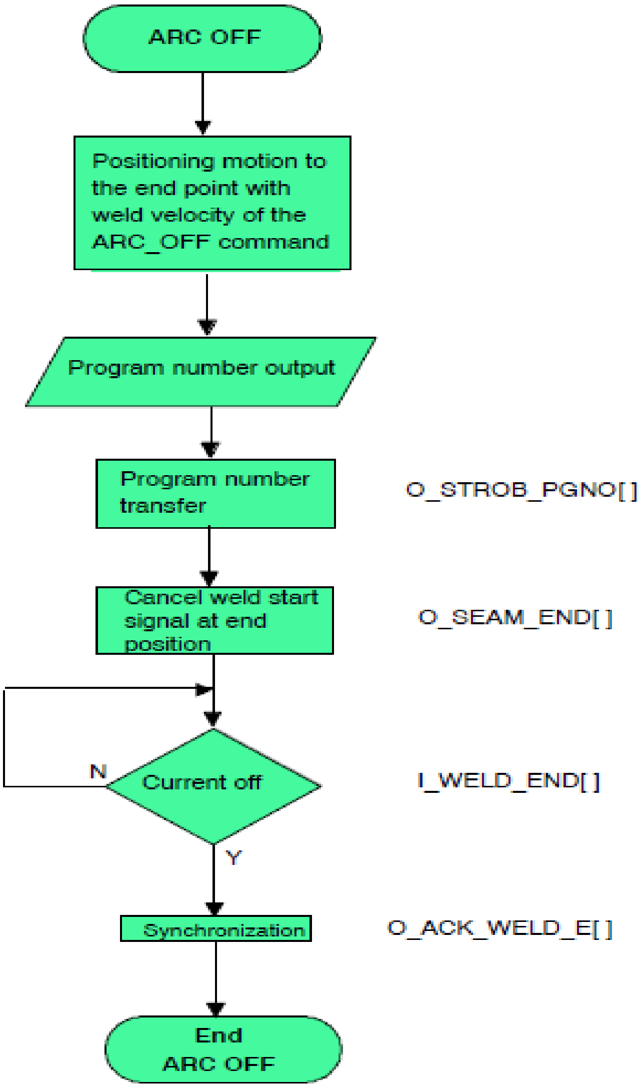
A2.1. ARC ON Flowchart.



A2.2. ARC SWITCH Flowchart


ARC SWITCH
Positioning motion to the end point with weld velocity of the ARC_SWI command
Program number output
Program number transfer
O_STROB_PGNO[ ]
End ARC SWITCH
Process continued with the next ARC OFF or ARC SWITCH command

A2.3. ARC OFF Flowchart



A3. KRL Codes

A3.1. Code of the welding experiment

```
&ACCESS RVP

&REL 267

&PARAM TEMPLATE = C:\KRC\Roboter\Template\vorgabe

&PARAM EDITMASK = *
```

```
DEF Jose( )


;FOLD INI

;FOLD OUT 28 '' State=TRUE CONT;%{PE}%R 5.6.13,%MKUKATPBASIS,%COUT,%VOUTX,%P 2:28, 3:,
5:TRUE, 6:CONTINUE


;MOTION COMMANDS

;INITIAL POSITION

;FOLD   PTP   RUST   CONT   Vel=100   %   PDAT13   Tool[3]:welding_torch2   Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:RUST, 3:C_PTP, 5:100, 7:PDAT13


;WELDING ACTION

;FOLD    PTP    P1    Vel=100    %    PDAT14    Tool[3]:welding_torch2    Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P1, 3:, 5:100, 7:PDAT14

;FOLD  LIN  P2  Vel=  2  m/s  CPDAT12  ARC_ON   Pgno=  4  S6  Seam6  Tool[3]:welding_torch2
Base[0];%{PE}%R  5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P  1:LIN,  2:P2,  3:,  5:2,  7:CPDAT12,
10:4, 11:S6, 12:Seam6

;FOLD LIN P3 CPDAT14 ARC_OFF  Pgno= 4 E3 Seam6 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.5.0,%MKUKATPA20,%CARC_OFF,%VLIN,%P  1:LIN,  2:P3,  3:,  5:2,  7:CPDAT14,  10:4,  11:E3,
12:Seam6

;FOLD    LIN    P4    Vel=2    m/s    CPDAT15    Tool[3]:welding_torch2    Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P4, 3:, 5:2, 7:CPDAT15


;END POSITION

;FOLD    PTP    RUST    Vel=100    %    PDAT19    Tool[3]:welding_torch2    Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:RUST, 3:, 5:100, 7:PDAT19


END
```

A3.2. Cylinder Welding Code

```
DEF Cylinder( )

FOLD INI


;Activation of output 28 to the welding power source (“No errors”)

FOLD OUT 28 '' State=TRUE CONT;%{PE}%R 5.6.13,%MKUKATPBASIS,%COUT,%VOUTX,%P 2:28, 3:,
5:TRUE, 6:CONTINUE


;Initial Position
```

```
FOLD PTP HOME Vel=100 % PDAT6;%{PE}%R 5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP,
2:HOME, 3:, 5:100, 7:PDAT6

;Motion Commands

;Constant point between welding sectors

FOLD PTP P1 Vel=100 % PDAT11 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P1, 3:, 5:100, 7:PDAT11

;First Sector

FOLD PTP P2 Vel=100 % PDAT13 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P2, 3:, 5:100, 7:PDAT13

FOLD PTP P3 Vel=100 % PDAT14 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P3, 3:, 5:100, 7:PDAT14

;Arc ignition

FOLD LIN P4 Vel= 2 m/s CPDAT14 ARC_ON  Pgno= 7 S2 Seam2 Tool[3]:Welding_torch2
Base[3]:Cylinder;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P4, 3:, 5:2,
7:CPDAT14, 10:1, 11:S2, 12:Seam2

;Perform the welding
FOLD CIRC P5 P6 CPDAT15 ARC_OFF  Pgno= 7 E2 Seam2 Tool[3]:Welding_torch2
Base[3]:Cylinder;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VCIRC,%P 1:CIRC, 2:P5, 3:P6, 4:,
6:2, 8:CPDAT15, 11:1, 12:E2, 13:Seam2

FOLD LIN P7 Vel=2 m/s CPDAT16 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P7, 3:, 5:2, 7:CPDAT16

;FOLD PTP P8 Vel=100 % PDAT15 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P8, 3:, 5:100, 7:PDAT15

;FOLD PTP P1 Vel=100 % PDAT16 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P1, 3:, 5:100, 7:PDAT16


;Second Sector

FOLD PTP P9 Vel=100 % PDAT17 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P9, 3:, 5:100, 7:PDAT17

FOLD PTP P10 Vel=100 % PDAT18 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P10, 3:, 5:100, 7:PDAT18

;Arc Ignition

FOLD LIN P11 Vel= 2 m/s CPDAT17 ARC_ON  Pgno= 7 S3 Seam3 Tool[3]:Welding_torch2
Base[3]:Cylinder;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P11, 3:, 5:2,
7:CPDAT17, 10:1, 11:S3, 12:Seam3

;Welding performance

FOLD CIRC P12 P13 CPDAT18 ARC_OFF  Pgno= 7 E3 Seam3 Tool[3]:Welding_torch2
Base[3]:Cylinder;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VCIRC,%P 1:CIRC, 2:P12, 3:P13,
4:, 6:2, 8:CPDAT18, 11:1, 12:E3, 13:Seam3

FOLD LIN P14 Vel=2 m/s CPDAT19 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P14, 3:, 5:2, 7:CPDAT19

;FOLD PTP P1 Vel=100 % PDAT19 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P1, 3:, 5:100, 7:PDAT19
```

```
;Third Sector

FOLD PTP P8 Vel=100 % PDAT20 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P8, 3:, 5:100, 7:PDAT20

FOLD PTP P7 Vel=100 % PDAT21 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P7, 3:, 5:100, 7:PDAT21

;Arc ignition

FOLD LIN P6 Vel= 2 m/s CPDAT20 ARC_ON  Pgno= 7 S4 Seam4 Tool[2]:welding_torch
Base[3]:Cylinder;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P6, 3:, 5:2,
7:CPDAT20, 10:1, 11:S4, 12:Seam4

;Welding performance

FOLD CIRC P15 P16 CPDAT21 ARC_OFF  Pgno= 7 E4 Seam4 Tool[3]:Welding_torch2
Base[3]:Cylinder;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VCIRC,%P 1:CIRC, 2:P15, 3:P16,
4:, 6:2, 8:CPDAT21, 11:1, 12:E4, 13:Seam4

FOLD LIN P17 Vel=2 m/s CPDAT22 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P17, 3:, 5:2, 7:CPDAT22

FOLD PTP P18 Vel=100 % PDAT22 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P18, 3:, 5:100, 7:PDAT22

FOLD PTP P1 Vel=100 % PDAT23 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P1, 3:, 5:100, 7:PDAT23


;Fourth Sector

;FOLD PTP P14 Vel=100 % PDAT24 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P14, 3:, 5:100, 7:PDAT24

;Arc Ignition

;FOLD LIN P13 Vel= 2 m/s CPDAT23 ARC_ON  Pgno= 7 S5 Seam5 Tool[2]:welding_torch
Base[3]:Cylinder;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P13, 3:, 5:2,
7:CPDAT23, 10:1, 11:S5, 12:Seam5

;Welding Performance

FOLD CIRC P19 P4 CPDAT24 ARC_OFF  Pgno= 7 E5 Seam5 Tool[3]:Welding_torch2
Base[3]:Cylinder;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VCIRC,%P 1:CIRC, 2:P19, 3:P4, 4:,
6:2, 8:CPDAT24, 11:1, 12:E5, 13:Seam5

;FOLD LIN P3 Vel=2 m/s CPDAT25 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P3, 3:, 5:2, 7:CPDAT25

;FOLD PTP P2 Vel=100 % PDAT25 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P2, 3:, 5:100, 7:PDAT25

;FOLD PTP P1 Vel=100 % PDAT26 Tool[3]:Welding_torch2 Base[3]:Cylinder;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P1, 3:, 5:100, 7:PDAT26


;Final Position

;FOLD PTP HOME Vel=100 % PDAT10;%{PE}%R 5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP,
2:HOME, 3:, 5:100, 7:PDAT10
```

```
END
```

A3.3. Special Piece Code

```
DEF workpiece( )

FOLD INI

;Activation of output 28 to the welding power source (“No errors”)

FOUT 28 '' State=TRUE CONT;%{PE}%R 5.6.13,%MKUKATPBASIS,%COUT,%VOUTX,%P 2:28, 3:,
5:TRUE, 6:CONTINUE


;Initial position

FOLD PTP HOME  Vel= 100 % DEFAULT;%{PE}%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:HOME, 3:,
5:100, 7:DEFAULT


;Straight central joint

FOLD PTP P1 Vel=100 % PDAT1 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P1, 3:, 5:100, 7:PDAT1

FOLD PTP P2 Vel=100 % PDAT2 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P2, 3:, 5:100, 7:PDAT2

FOLD PTP P3 Vel=100 % PDAT4 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P3, 3:, 5:100, 7:PDAT4

;Arc Ignition

FOLD LIN P4 Vel= 2 m/s CPDAT1 ARC_ON  Pgno= 8 S2 Seam2 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P4, 3:, 5:2,
7:CPDAT1, 10:11, 11:S2, 12:Seam2

;Welding performance

FOLD LIN P5 CPDAT4 ARC_OFF  Pgno= 8 E2 Seam3 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.5.0,%MKUKATPA20,%CARC_OFF,%VLIN,%P 1:LIN, 2:P5, 3:, 5:2, 7:CPDAT4, 10:11, 11:E2,
12:Seam3

FOLD LIN P6 Vel=2 m/s CPDAT3 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P6, 3:, 5:2, 7:CPDAT3

FOLD PTP P7 Vel=100 % PDAT5 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P7, 3:, 5:100, 7:PDAT5

;Circular beam

;First Sector

FOLD PTP P8 Vel=100 % PDAT40 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P8, 3:, 5:100, 7:PDAT40

FOLD PTP P9 Vel=100 % PDAT41 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P9, 3:, 5:100, 7:PDAT41

;Arc Ignition
```

```
FOLD LIN P10 Vel= 2 m/s CPDAT40 ARC_ON Pgno= 7 S13 Seam6 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P10, 3:, 5:2,
7:CPDAT40, 10:4, 11:S13, 12:Seam6

;Welding performance

FOLD CIRC P15 P16 CPDAT41 ARC_OFF Pgno= 7 E12 Seam12 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VCIRC,%P 1:CIRC, 2:P15, 3:P16, 4:,
6:2, 8:CPDAT41, 11:4, 12:E12, 13:Seam12

FOLD LIN P60 Vel=2 m/s CPDAT38 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P60, 3:, 5:2, 7:CPDAT38

FOLD PTP P63 Vel=100 % PDAT38 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P63, 3:, 5:100, 7:PDAT38

;Second Sector

FOLD PTP P64 Vel=100 % PDAT42 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P64, 3:, 5:100, 7:PDAT42

;Arc Ignition

FOLD LIN P18 Vel= 2 m/s CPDAT42 ARC_ON Pgno= 7 S14 Seam13 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P18, 3:, 5:2,
7:CPDAT42, 10:4, 11:S14, 12:Seam13

;Welding performance

FOLD CIRC P19 P20 CPDAT43 ARC_OFF Pgno= 7 E13 Seam13 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VCIRC,%P 1:CIRC, 2:P19, 3:P20, 4:,
6:2, 8:CPDAT43, 11:4, 12:E13, 13:Seam13

FOLD LIN P65 Vel=2 m/s CPDAT44 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P65, 3:, 5:2, 7:CPDAT44

FOLD PTP P63 Vel=100 % PDAT43 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P63, 3:, 5:100, 7:PDAT43

;Third Sector

FOLD PTP P60 Vel=100 % PDAT44 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P60, 3:, 5:100, 7:PDAT44

;Arc Ignition

FOLD LIN P16 Vel= 2 m/s CPDAT45 ARC_ON Pgno= 7 S15 Seam14 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P16, 3:, 5:2,
7:CPDAT45, 10:4, 11:S15, 12:Seam14

;Welding performance

FOLD CIRC P17 P18 CPDAT46 ARC_OFF Pgno= 7 E14 Seam14 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VCIRC,%P 1:CIRC, 2:P17, 3:P18, 4:,
6:2, 8:CPDAT46, 11:4, 12:E14, 13:Seam14

FOLD LIN P64 Vel=2 m/s CPDAT47 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P64, 3:, 5:2, 7:CPDAT47

FOLD PTP P63 Vel=100 % PDAT45 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P63, 3:, 5:100, 7:PDAT45

;Fourth Sector

FOLD PTP P65 Vel=100 % PDAT46 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P65, 3:, 5:100, 7:PDAT46

;Arc Ignition
```

```
FOLD LIN P20 Vel= 2 m/s CPDAT48 ARC_ON  Pgno= 7 S16 Seam15 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P20, 3:, 5:2,
7:CPDAT48, 10:4, 11:S16, 12:Seam15

;Welding performance

FOLD CIRC P11 P12 CPDAT49 ARC_OFF  Pgno= 7 E15 Seam15 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VCIRC,%P 1:CIRC, 2:P11, 3:P12, 4:,
6:2, 8:CPDAT49, 11:4, 12:E15, 13:Seam15

FOLD LIN P13 Vel=2 m/s CPDAT50 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P13, 3:, 5:2, 7:CPDAT50

FOLD PTP P14 Vel=100 % PDAT47 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P14, 3:, 5:100, 7:PDAT47

;Square beam with 2 L perfils

;First Sector

FOLD PTP P27 Vel=100 % PDAT13 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P27, 3:, 5:100, 7:PDAT13

FOLD PTP P22 Vel=100 % PDAT11 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P22, 3:, 5:100, 7:PDAT11

;Arc Ignition

FOLD LIN P23 Vel= 2 m/s CPDAT16 ARC_ON  Pgno= 8 S7 Seam7 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P23, 3:, 5:2,
7:CPDAT16, 10:11, 11:S7, 12:Seam7

;Welding performance

FOLD LIN P24 CPDAT17 ARC_OFF  Pgno= 8 E5 Seam7 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VLIN,%P 1:LIN, 2:P24, 3:, 5:2,
7:CPDAT17, 10:11, 11:E5, 12:Seam7

FOLD LIN P25 Vel=2 m/s CPDAT18 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P25, 3:, 5:2, 7:CPDAT18

FOLD PTP P26 Vel=100 % PDAT12 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P26, 3:, 5:100, 7:PDAT12

FOLD PTP P27 Vel=100 % PDAT14 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P27, 3:, 5:100, 7:PDAT14


;Second Sector

FOLD PTP P28 Vel=100 % PDAT15 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P28, 3:, 5:100, 7:PDAT15

FOLD PTP P29 Vel=100 % PDAT16 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P29, 3:, 5:100, 7:PDAT16

;Arc Ignition

FOLD LIN P30 Vel= 2 m/s CPDAT19 ARC_ON  Pgno= 8 S8 Seam8 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P30, 3:, 5:2,
7:CPDAT19, 10:11, 11:S8, 12:Seam8

;Welding performance

FOLD LIN P31 CPDAT20 ARC_OFF  Pgno= 8 E6 Seam8 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VLIN,%P 1:LIN, 2:P31, 3:, 5:2,
7:CPDAT20, 10:11, 11:E6, 12:Seam8
```

```
FOLD LIN P32 Vel=2 m/s CPDAT21 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P32, 3:, 5:2, 7:CPDAT21

FOLD PTP P33 Vel=100 % PDAT17 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P33, 3:, 5:100, 7:PDAT17

FOLD PTP P27 Vel=100 % PDAT18 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P27, 3:, 5:100, 7:PDAT18

;Third Sector

FOLD PTP P34 Vel=100 % PDAT19 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P34, 3:, 5:100, 7:PDAT19

FOLD PTP P35 Vel=100 % PDAT20 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P35, 3:, 5:100, 7:PDAT20

;Arc Ignition

FOLD LIN P36 Vel= 2 m/s CPDAT23 ARC_ON  Pgno= 8 S9 Seam9 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P36, 3:, 5:2,
7:CPDAT23, 10:11, 11:S9, 12:Seam9

;Welding Performance

FOLD LIN P37 CPDAT24 ARC_OFF  Pgno= 8 E7 Seam9 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VLIN,%P 1:LIN, 2:P37, 3:, 5:2,
7:CPDAT24, 10:11, 11:E7, 12:Seam9

FOLD LIN P38 Vel=2 m/s CPDAT25 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P38, 3:, 5:2, 7:CPDAT25

FOLD PTP P39 Vel=100 % PDAT21 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P39, 3:, 5:100, 7:PDAT21

FOLD PTP P27 Vel=100 % PDAT22 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P27, 3:, 5:100, 7:PDAT22

;Fourth Sector

FOLD PTP P40 Vel=100 % PDAT23 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P40, 3:, 5:100, 7:PDAT23

FOLD PTP P41 Vel=100 % PDAT24 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P41, 3:, 5:100, 7:PDAT

;Arc Ignition

FOLD LIN P42 Vel= 2 m/s CPDAT26 ARC_ON  Pgno= 8 S10 Seam10 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P42, 3:, 5:2,
7:CPDAT26, 10:11, 11:S10, 12:Seam10

;Weldign performance

FOLD LIN P43 CPDAT27 ARC_OFF  Pgno= 8 E8 Seam10 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VLIN,%P 1:LIN, 2:P43, 3:, 5:2,
7:CPDAT27, 10:11, 11:E8, 12:Seam10

FOLD LIN P44 Vel=2 m/s CPDAT28 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P44, 3:, 5:2, 7:CPDAT28

FOLD PTP P45 Vel=100 % PDAT25 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P45, 3:, 5:100, 7:PDAT25


;Vertical beam
```

```
;First weld

FOLD PTP P46 Vel=100 % PDAT26 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P46, 3:, 5:100, 7:PDAT26

FOLD PTP P47 Vel=100 % PDAT27 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P47, 3:, 5:100, 7:PDAT27


;Arc Ignition

FOLD LIN P49 Vel= 2 m/s CPDAT29 ARC_ON Pgno= 6 S11 Seam11 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P49, 3:, 5:2,
7:CPDAT29, 10:4, 11:S11, 12:Seam11

;Welding performance

FOLD LIN P48 CPDAT30 ARC_OFF Pgno= 6 E9 Seam11 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VLIN,%P 1:LIN, 2:P48, 3:, 5:2,
7:CPDAT30, 10:4, 11:E9, 12:Seam11

FOLD LIN P50 Vel=2 m/s CPDAT31 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P50, 3:, 5:2, 7:CPDAT31

FOLD PTP P51 Vel=100 % PDAT28 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P51, 3:, 5:100, 7:PDAT28

FOLD PTP P27 Vel=100 % PDAT29 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P27, 3:, 5:100, 7:PDAT29

;Second weld

FOLD PTP P52 Vel=100 % PDAT30 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P52, 3:, 5:100, 7:PDAT30

FOLD PTP P53 Vel=100 % PDAT31 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P53, 3:, 5:100, 7:PDAT31

;Arc Ignition

FOLD LIN P55 Vel= 2 m/s CPDAT32 ARC_ON Pgno= 6 S12 Seam12 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_ON,%VLIN,%P 1:LIN, 2:P55, 3:, 5:2,
7:CPDAT32, 10:4, 11:S12, 12:Seam12;

Welding performance

FOLD LIN P54 CPDAT33 ARC_OFF Pgno= 6 E10 Seam12 Tool[2]:welding_torch
Base[2]:Base1;%{PE}%R 5.5.0,%MKUKATPA20,%CARC_OFF,%VLIN,%P 1:LIN, 2:P54, 3:, 5:2,
7:CPDAT33, 10:4, 11:E10, 12:Seam12

FOLD LIN P56 Vel=2 m/s CPDAT34 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P56, 3:, 5:2, 7:CPDAT34

FOLD PTP P57 Vel=100 % PDAT32 Tool[2]:welding_torch Base[2]:Base1;%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P57, 3:, 5:100, 7:PDAT32


;Final Position

FOLD PTP HOME Vel=100 % PDAT36;%{PE}%R 5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP,
2:HOME, 3:, 5:100, 7:PDAT36


END
```

A3.4. Cleaner Code

```
DEF Cleaner( )

;FOLD INI

;MOVEMENT FROM WELDING ZONE TO CLEANER ZONE

;FOLD PTP HOME Vel=100 % DEFAULT;%{PE}%R 5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP,
2:HOME, 3:, 5:100, 7:DEFAULT

;FOLD PTP P1 Vel=100 % PDAT1 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P1, 3:, 5:100, 7:PDAT1

;FOLD PTP P2 Vel=100 % PDAT2 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P2, 3:, 5:100, 7:PDAT2

;FOLD PTP P11 Vel=100 % PDAT11 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P11, 3:, 5:100, 7:PDAT11

;FOLD PTP P12 Vel=100 % PDAT12 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P12, 3:, 5:100, 7:PDAT12


;WIRECUTTING

;FOLD LIN P14 Vel=0.5 m/s CPDAT17 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P14, 3:, 5:0.5, 7:CPDAT17

;FOLD LIN P13 Vel=0.5 m/s CPDAT15 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P13, 3:, 5:0.5, 7:CPDAT15

;FOLD PULSE 26 '' State=TRUE CONT Time=0.1 sec;%{PE}%R
5.6.13,%MKUKATPBASIS,%COUT,%VPULSE,%P 2:26, 3:, 5:TRUE, 6:CONTINUE, 8:0.1

;FOLD PULSE 3 '' State=TRUE CONT Time=1 sec;%{PE}%R
5.6.13,%MKUKATPBASIS,%COUT,%VPULSE,%P 2:3, 3:, 5:TRUE, 6:CONTINUE, 8:1

;FOLD WAIT Time=2 sec;%{PE}%R 5.6.13,%MKUKATPBASIS,%CWAIT,%VWAIT,%P 2:2

;FOLD LIN P12 Vel=0.5 m/s CPDAT16 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P12, 3:, 5:0.5, 7:CPDAT16


;WELD TORCH CLEANING

;FOLD PTP P11 Vel=100 % PDAT14 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P11, 3:, 5:100, 7:PDAT14

;FOLD PTP P3 Vel=100 % PDAT3 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P3, 3:, 5:100, 7:PDAT3

;FOLD LIN P4 Vel=0.5 m/s CPDAT6 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P4, 3:, 5:0.5, 7:CPDAT6

;FOLD LIN P5 Vel=0.5 m/s CPDAT7 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P5, 3:, 5:0.5, 7:CPDAT7

;FOLD LIN P6 Vel=0.5 m/s CPDAT8 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P6, 3:, 5:0.5, 7:CPDAT8

;FOLD LIN P7 Vel=0.5 m/s CPDAT9 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P7, 3:, 5:0.5, 7:CPDAT9
```

```
;FOLD LIN P8 Vel=0.5 m/s CPDAT10 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P8, 3:, 5:0.5, 7:CPDAT10

;FOLD OUT 2 '' State=TRUE CONT;%{PE}%R 5.6.13,%MKUKATPBASIS,%COUT,%VOUTX,%P 2:2, 3:,
5:TRUE, 6:CONTINUE

;FOLD OUT 3 '' State=TRUE CONT;%{PE}%R 5.6.13,%MKUKATPBASIS,%COUT,%VOUTX,%P 2:3, 3:,
5:TRUE, 6:CONTINUE

;FOLD WAIT Time=3 sec;%{PE}%R 5.6.13,%MKUKATPBASIS,%CWAIT,%VWAIT,%P 2:3

;FOLD OUT 3 '' State=FALSE CONT;%{PE}%R 5.6.13,%MKUKATPBASIS,%COUT,%VOUTX,%P 2:3, 3:,
5:FALSE, 6:CONTINUE

;FOLD WAIT Time=0.7 sec;%{PE}%R 5.6.13,%MKUKATPBASIS,%CWAIT,%VWAIT,%P 2:0.7

;FOLD OUT 2 '' State=FALSE CONT;%{PE}%R 5.6.13,%MKUKATPBASIS,%COUT,%VOUTX,%P 2:2, 3:,
5:FALSE, 6:CONTINUE


;LUBRICATION

;FOLD LIN P9 Vel=0.5 m/s CPDAT12 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P9, 3:, 5:0.5, 7:CPDAT12

;FOLD LIN P10 Vel=0.5 m/s CPDAT11 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P10, 3:, 5:0.5, 7:CPDAT11

;FOLD WAIT Time=2 sec;%{PE}%R 5.6.13,%MKUKATPBASIS,%CWAIT,%VWAIT,%P 2:3

;FOLD LIN P9 Vel=0.5 m/s CPDAT13 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P9, 3:, 5:0.5, 7:CPDAT13


;COME BACK TO WELDING ZONE

;FOLD LIN P4 Vel=0.5 m/s CPDAT14 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VLIN,%P 1:LIN, 2:P4, 3:, 5:0.5, 7:CPDAT14

;FOLD PTP P3 Vel=100 % PDAT8 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P3, 3:, 5:100, 7:PDAT8

;FOLD PTP P2 Vel=100 % PDAT9 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P2, 3:, 5:100, 7:PDAT9

;FOLD PTP P1 Vel=100 % PDAT10 Tool[3]:welding_torch2 Base[0];%{PE}%R
5.6.13,%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:P1, 3:, 5:100, 7:PDAT10

;FOLD PTP HOME  Vel= 100 % DEFAULT;%{PE}%MKUKATPBASIS,%CMOVE,%VPTP,%P 1:PTP, 2:HOME,
3:, 5:100, 7:DEFAULT


END
```

A4. HDevelop Codes

A4.1. Camera Calibration Example

```
*****************************
***UEYE CAMERA CALIBRATION***
*****************************
*** Close any possible acquirement device that could remain open***
close_all_framegrabbers()


***Camera Calibration Initialization***
CalTabDescrFile := 'caltab_70mm.descr'
StartCamPar            :=            [0.0416569701915549,0,1.09711074971488e-005,1.1e-005,1155.47676730987,782.699003739001,2048,1536]
create_calib_data ('calibration_object', 1, 1, CalibDataID)
set_calib_data_cam_param (CalibDataID, 0, 'area_scan_division', StartCamPar)
set_calib_data_calib_object (CalibDataID, 0, CalTabDescrFile)


***Acquirement Initialization***
open_framegrabber ('uEye', 1, 1, 0, 0, 0, 0, 'default', -1, 'default', -1, 'default', 'default', 'default', -1, -1, AcqHandle)
get_framegrabber_param (AcqHandle, 'camera_info', CameraInfo)
get_framegrabber_param (AcqHandle, 'sensor_type', SensorType)
get_framegrabber_param (AcqHandle, 'image_width', ImageWidth)
get_framegrabber_param (AcqHandle, 'image_height', ImageHeight)


*** Result Window creation***
dev_close_window ()
dev_open_window (0, 0, ImageWidth / 2, ImageHeight / 2, 'black', WindowHandle)
dev_update_off ()
dev_set_draw ('margin')
dev_set_line_width (3)
set_display_font (WindowHandle, 14, 'mono', 'true', 'false')


***Ten image where the calibration plate has to be detected***
NumImages := 10
```

```
for I := 1 to NumImages by 1

       grab_image(Image,AcqHandle)

       dev_display (Image)

       Message := 'Find calibration plate in\nall calibration images (' + I + '/' +
       NumImages + ')'

       disp_message (WindowHandle, Message, 'window', 12, 12, 'black', 'true')

       *** Find the calibration plate***

       find_calib_object (Image, CalibDataID, 0, 0, I - 1, [], [])

       *** Superimpose the obtained information in the image***

       get_calib_data (CalibDataID, 'camera', 0, 'init_params', StartCamPar)

       get_calib_data_observ_points (CalibDataID, 0, 0, I - 1, Row, Column, Index,
       Pose)

       get_calib_data_observ_contours (Contours, CalibDataID, 'caltab', 0, 0, I - 1)

       gen_cross_contour_xld (Cross, Row, Column, 6, 0.785398)

       dev_set_color ('green')

       dev_display (Contours)

       dev_set_color ('red')

       dev_display (Cross)

       disp_continue_message (WindowHandle, 'black', 'true')

       stop ()

endfor


***CAMERA CALIBRATION***

calibrate_cameras (CalibDataID, Error)

get_calib_data (CalibDataID, 'camera', 0, 'params', CamParam)


*** Save Camera Parameters into a File***

write_cam_par(CamParam,'UeyeParam.dat')


***Close Camera Acquisition***

close_framegrabber(AcqHandle)
```

A4.2. 3D Shape-Based Example

```
******************************************

*** SHAPE BASED MATCHING WITH STL MODEL***

******************************************
```

```
dev_update_off ()

dev_get_preferences ('suppress_handled_exceptions_dlg', SuppressHandledExceptionsDlg)

dev_set_preferences ('suppress_handled_exceptions_dlg', 'true')


*** Close any framegrabber that could remain open***

close_all_framegrabbers()


*** Read internal camera parameters from file***

read_cam_par('UeyeParam.dat', CamParam)


*** Creation of Result Window***

Width := CamParam[6]/4

Height := CamParam[7]/4

dev_close_window ()

dev_open_window (0, 0, Width, Height, 'white', WindowHandle)

set_display_font (WindowHandle, 14, 'mono', 'true', 'false')

dev_update_off ()


***Acquirement Initialization***

open_framegrabber ('uEye', 1, 1, 0, 0, 0, 0, 'default', -1, 'default', -1, 'default',
'default', 'default', -1, -1, AcqHandle)


*** Read the object model from a STL file***

read_object_model_3d ('Test1.stl','mm', ['file_type','invert_normals'],
['stl','true'], ObjectModel3DID, STLStatus)


*** Prepare the object model for the use within the 3d shape-based matching***

prepare_object_model_3d (ObjectModel3DID, 'shape_based_matching_3d', 'true', [], []


*** Shape-Based 3D Model Creation***

disp_message (WindowHandle,'Reading the 3D shape model file from disk ...' , 'window',
12, 12, 'black', 'true')

try

       read_shape_model_3d('Test1_Shape_Model.sm3',ShapeModel3DID)

       disp_message(WindowHandle, 'Reading Succesful...',
       'window',42,12,'black','true')

catch(Exception)
```

```
       disp_message (WindowHandle, 'Reading the 3D shape model file from disk ...File
       not found', 'window', 12, 12, 'black', 'true')

       Message := 'Creating the 3D shape model (may take several minutes) ...'

       disp_message (WindowHandle, Message, 'window', 62, 12, 'black', 'true')

       count_seconds (S1)

       create_shape_model_3d  (ObjectModel3DID,  CamParam,  0,  0,  0,  'gba',  -
       rad(100),rad(100),           -rad(60), rad(60), 0, rad(360), 0.49, 0.52, 30,
       ['lowest_model_level'], [3], ShapeModel3DID)

       write_shape_model_3d(ShapeModel3DID, 'Test1_Shape_Model.sm3')

       count_seconds (S2)

       T := S2 - S1

       disp_message (WindowHandle, 'Creation time: ' + T$'.3' + ' s', 'window', 92, 12,
       'black', 'false')

endtry

disp_continue_message(WindowHandle, 'black','true')

stop()


***Clear 3D Object Model***

clear_object_model_3d (ObjectModel3DID)


***Find instances of a 3D shape model in some run-time images***

Times := []

for ImageNo := 1 to 4 by 1

       grab_image(Image, AcqHandle)

       dev_display (Image)

       count_seconds (Seconds1)

       find_shape_model_3d (Image, ShapeModel3DID, 0.0, 0.0, 0,

       ['num_matches','max_overlap'],  [1,0], Pose, CovPose, Score)

       count_seconds (Seconds2)

       Time := Seconds2 - Seconds1

       Times := [Times,Time]

       * Visualize the found matches in the image by projecting

       * the 3D shape model into it using the pose of the match

       for I := 0 to |Score| - 1 by 1

              *Save information of matched Pose

              PoseTmp := Pose[I * 7:I * 7 + 6]

              write_pose(PoseTmp,'Test1_Pose.dat')

              *Display contours of the match
              project_shape_model_3d   (ModelContours,   ShapeModel3DID,   CamParam,
              PoseTmp, 'true', 0)

              dev_set_color ('green')

              dev_display (ModelContours)

              * Display the coordinate system of the 3D object model

              dev_set_color ('red')

              disp_3d_coord_system (WindowHandle, CamParam, PoseTmp, 0.015)

       endfor

       disp_message (WindowHandle, |Score| + ' Match(es) found in ' + Time$'4.2f' + '
       s', 'window', 12, 12, 'black', 'true')

       if (ImageNo<4)

              disp_continue_message (WindowHandle, 'black', 'true')

              stop ()

              dev_clear_window()

       endif

endfor


***Close the Acquirement connection***

close_all_framegrabbers()


Message := 'Program terminated. Press \'Run\' to clear the 3D shape model.'

disp_message (WindowHandle, Message, 'window', Height - 30, 12, 'black', 'true')

stop ()


*** Free the 3D shape model ***

clear_shape_model_3d (ShapeModel3DID)

dev_get_preferences ('suppress_handled_exceptions_dlg', SuppressHandledExceptionsDlg)
```

A4.3. Cylinder 3D Shape-Based Matching

```
******************************************

*** SHAPE BASED MATCHING WITH STL MODEL***

******************************************

dev_update_off ()

dev_get_preferences ('suppress_handled_exceptions_dlg', SuppressHandledExceptionsDlg)
```

```
dev_set_preferences ('suppress_handled_exceptions_dlg', 'true')


*** Close Any framegrabber that could remain open***

close_all_framegrabbers()


***Create Camera connection***

open_framegrabber ('uEye', 1, 1, 0, 0, 0, 0, 'default', -1, 'default', -1, 'default',
'default', 'default', -1, -1, AcqHandle)


*** Read internal camera parameters from file***

read_cam_par('C:/Documents and Settings/acro/Desktop/SmartFactory/Calibration/Camera
Calibration/cam_file_pose/cam_param.cal', CamParam)


*** Creation of Result Window***

Width := CamParam[6]/4

Height := CamParam[7]/4

dev_close_window ()

dev_open_window (0, 0, Width, Height, 'white', WindowHandle)

set_display_font (WindowHandle, 14, 'mono', 'true', 'false')

dev_update_off ()


*** Read the object model from a STL file***

read_object_model_3d ('Piece1.stl','mm', ['file_type','invert_normals'],
['stl','true'], ObjectModel3DID, STLStatus)

disp_object_model_3d(WindowHandle, ObjectModel3DID,CamParam,[],[],[])

disp_continue_message (WindowHandle, 'black', 'true')

stop()


*** Prepare the object model for the use within the 3d shape-based matching***

prepare_object_model_3d (ObjectModel3DID, 'shape_based_matching_3d', 'true', [], [])

***Clear Result Window***

dev_clear_window ()

*** Sape-Based 3D Model Procedure***

disp_message (WindowHandle,'Reading the 3D shape model file from disk ...' ,
'window', 12, 12, 'black', 'true')

 try
```

```
        read_shape_model_3d('C:/Documents and
        Settings/acro/Desktop/SmartFactory/matching/Cylinder/Piece1_Shape_Model_new_ca
        libration.sm3',ShapeModel3DID)

        disp_message(WindowHandle, 'Reading Succesful...',
        'window',62,12,'black','true')

 catch(Exception)

        disp_message(WindowHandle, 'Reading failed, the procedure to shape model
        creation is starting', 'window',72,12,'black','true')

        disp_continue_message(WindowHandle, 'black','true')

        stop()


        *** Grab one image for the inspect process***

        grab_image(Image, AcqHandle)

        dev_display(Image)

        disp_continue_message (WindowHandle, 'black', 'true')

        stop()


        *** Inspect the 3D object model and specify the desired pose range

        *** in which the 3D shape model is to be created***

        inspect_object_model_3d (Image, ObjectModel3DID, CamParam, RefRotX, RefRotY,
        RefRotZ, LongitudeMin, LongitudeMax, LatitudeMin, LatitudeMax, CamRollMin,
        CamRollMax, DistMin, DistMax, MinFaceAngle)

        dev_clear_window ()

        Message := 'Creating the 3D shape model (may take several minutes) ...'

        disp_message (WindowHandle, Message, 'window', 82, 12, 'black', 'true')

        count_seconds (S1)

        *** Create shape model process***

        create_shape_model_3d (ObjectModel3DID, CamParam, RefRotX, RefRotY, RefRotZ,
        'gba', LongitudeMin, LongitudeMax, LatitudeMin, LatitudeMax, CamRollMin,
        CamRollMax, 0.95,1.10, 10, ['lowest_model_level'], [3], ShapeModel3DID)

        count_seconds (S2)

        write_shape_model_3d(ShapeModel3DID, 'Piece1_Shape_Model_new_calibration.sm3')

        T := S2 - S1

        disp_message (WindowHandle, 'Creation time: ' + T$'.3' + ' s', 'window', 102,
        12, 'black', 'false')

endtry


disp_continue_message(WindowHandle, 'black','true')

stop()
```

```
***Clear 3D Object Model***
* After creating the 3D shape model we do not longer need
* the 3D object model***
clear_object_model_3d (ObjectModel3DID)

***Find instances of a 3D shape model in some run-time images***
Times := []
*** A total of 50 matching can be performed in a row***
for ImageNo := 1 to 50 by 1
    grab_image(Image, AcqHandle)
    dev_display(Image)
    count_seconds (Seconds1)
    *** Find up to two instances of the 3D shape model***
    find_shape_model_3d (Image, ShapeModel3DID, 0.4, 0, 10,
    ['num_matches','border_model'], [1,'true'], Pose, CovPose, Score)
    count_seconds (Seconds2)
    Time:=Seconds2-Seconds1
    Times := [Times,Time]
    *** Visualize result matches in the image by projecting***
    *** the 3D shape model into it using the pose of the match***

    for I := 0 to |Score| - 1 by 1
        ***Display Contours***
        PoseTmp := Pose[I * 7:I * 7 + 6]
        ***Perform the homogeneous transformation to situate the coordinate
        ***system correctly***
        pose_to_hom_mat3d(PoseTmp,obj_H_cam)
        hom_mat3d_identity(HomMat3DIdentity)
        hom_mat3d_translate_local(HomMat3DIdentity,0.024,0.048,0.024,HomMat3D_R
        )
        hom_mat3d_rotate_local(HomMat3D_T,1.570796327,'x',HomMat3D_T_R)
        hom_mat3d_compose(obj_H_cam, HomMat3D_T_R, obj_H_cam)
        hom_mat3d_to_pose(obj_H_cam,PoseTmp1)
        write_pose(PoseTmp1,'CylinderinCam_Pose.dat')
        project_shape_model_3d (ModelContours, ShapeModel3DID, CamParam,
        PoseTmp, 'true', 0)
        dev_set_color ('green')
        dev_display (ModelContours)
```

```
              *** Display the coordinate system of the 3D object model***

              dev_set_color ('red')

              disp_3d_coord_system (WindowHandle, CamParam, PoseTmp1, 0.015)

       endfor

       disp_message (WindowHandle, |Score| + ' Match(es) found in ' + Time$'4.2f' +
       's', 'window', 12, 12, 'black', 'true')

       if (ImageNo<50)

              disp_continue_message (WindowHandle, 'black', 'true')

              stop ()

              dev_clear_window()

       endif

endfor


Message := 'Program finished. Press \'Run\' to clear the 3D shape model.'

disp_message (WindowHandle, Message, 'window', Height - 30, 12, 'black', 'true')

stop ()

*** Free the memory space of the 3D shape model***

clear_shape_model_3d (ShapeModel3DID)

dev_get_preferences ('suppress_handled_exceptions_dlg', SuppressHandledExceptionsDlg)
```

A4.4. Grabbing calibration images and robot info.hdev

```
*TCP/IP Connection information

local_IP_adres:='10.10.107.20'

remote_IP_adres:='10.10.107.25'

Local_port_number:=9901

Remote_port_number:=9901

TCP_or_UDP:='TCP'

Timeout=200


*Close any fralmegrabber that could remain open

 close_all_framegrabbers()


* open uEye camera driver

open_framegrabber ('uEye', 1, 1, 0, 0, 0, 0, 'default', -1, 'default', -1, 'default',
'default', 'default', -1, -1, AcqHandle)
```

```
* get some camera information parameters

get_framegrabber_param (AcqHandle, 'image_width', ImageWidth)

get_framegrabber_param (AcqHandle, 'image_height', ImageHeight)


*opening the socket

open_socket_connect (remote_IP_adres, 9901, ['protocol','timeout'], ['TCP',timeout],
Socket)

*preparing xml file to send

data_send_read:='<?xml version="1.0" encoding="UTF-8"?><ShowVar VarName="$POS_ACT"/>'


*Results window creation

dev_close_window ()

dev_open_window (0, 0, ImageWidth/3 , ImageHeight/3 , 'black', WindowHandle)


*Perform the information grabbing

for i := 1 to 100 by 1

       grab_image(Image,AcqHandle)

       write_image(Image, 'png', 0, 'Hand_EyeCal' + i$'002')

       *Sending data request to the robot

       send_data (Socket, 'z', data_send_read, [])

       *Recieve data

       receive_data (Socket, 'z', DataReceived, From)

       *Extract the robot position information

       tuple_strstr (DataReceived, 'X', Position)

       tuple_substr (DataReceived, Position+2, Position+8, Rob_X)

       tuple_number(Rob_X,Rob_X)

       tuple_strstr (DataReceived, 'Y', Position)

       tuple_substr (DataReceived, Position+2, Position+7, Rob_Y)

       tuple_number(Rob_Y,Rob_Y)

       tuple_strstr (DataReceived, 'Z', Position)

       tuple_substr (DataReceived, Position+2, Position+8, Rob_Z)

       tuple_number(Rob_Z,Rob_Z)

       tuple_strstr (DataReceived, ',A ', Position)

       tuple_substr (DataReceived, Position+2, Position+8, Rob_A)

       tuple_number(Rob_A,Rob_A)

       tuple_strstr (DataReceived, ',B ', Position)
```

```
        tuple_substr (DataReceived, Position+2, Position+8, Rob_B)

        tuple_number(Rob_B,Rob_B)

        tuple_strstr (DataReceived, ',C ', Position)

        tuple_substr (DataReceived, Position+2, Position+8, Rob_C)

        tuple_number(Rob_C,Rob_C)

        * Save the information in a pose

        create_pose(Rob_X,Rob_Y,Rob_Z,Rob_C,Rob_B,Rob_A,'Rp+T', 'abg', 'point'
        ,Robot_Pose)

        write_pose(Robot_Pose,'Robot_Pose' + i$'002'+'.dat')

        disp_continue_message(WindowHandle,'black','true')

        dev_display(Image)

        stop()

        dev_clear_window()

        endfor


*Close framegrabber

close_framegrabber (AcqHandle)

*Close Socket

close_socket(Socket)
```

A4.5. Hand-Eye Calibration

```
***********************************************************

************STATIONARY HAND-EYE CALIBRATION****************

***********************************************************

dev_update_off ()

* Directories with calibration images and data files

ImageNameStart := 'C:/Documents and
Settings/acro/Desktop/SmartFactory/Calibration/Hand-eye
calibration/images/Hand_EyeCal'

DataNameStart := 'C:/Documents and
Settings/acro/Desktop/SmartFactory/Calibration/Hand-eye
calibration/Robot_poses/Robot_Pose'

NumImages :=12

* Read image

read_image (Image, ImageNameStart + '01')

get_image_size (Image, Width, Height)
```

```
* Create result window

dev_close_window ()

dev_open_window (0, 0, Width/3, Height/3, 'black', WindowHandle)

dev_set_line_width (2)

dev_set_draw ('margin')


* Load the calibration plate description file.

* Make sure that the file is in the current directory,

* the HALCONROOT/calib directory, or use an absolut path

CalTabFile := 'caltab_70mm.descr'


* Read the values of the internal camera parameters

read_cam_par ('C:/Documents and Settings/acro/Desktop/SmartFactory/Calibration/Camera
Calibration/cam_file_pose/cam_param.cal', StartCamParam)


* Create the calibration model for the hand eye calibration

create_calib_data ('hand_eye_stationary_cam', 1, 1, CalibDataID)

set_calib_data_cam_param (CalibDataID, 0, 'area_scan_division', StartCamParam)

set_calib_data_calib_object (CalibDataID, 0, CalTabFile)

disp_continue_message (WindowHandle, 'black', 'true')

stop ()


* Start the loop over the calibration images

for I := 0 to NumImages-1 by 1

       read_image (Image, ImageNameStart + (I+1)$'002')

       dev_display (Image)

       * Search for the calibration plate, extract the marks and the

       * pose of it, and store the results in the calibration data model of the

       * hand-eye calibration

       find_calib_object (Image, CalibDataID, 0, 0, I, ['alpha'], [0.26])

       get_calib_data_observ_contours (Caltab, CalibDataID, 'caltab', 0, 0, I)

       get_calib_data_observ_points (CalibDataID, 0, 0, I, RCoord, CCoord, Index,
       CalObjInCamPose)

       * Visualize the extracted calibration marks and the estimated pose (coordinate
       system)

       dev_set_color ('green')

       dev_display (Image)
       dev_display (Caltab)
       dev_set_color ('yellow')
       disp_cross (WindowHandle, RCoord, CCoord, 20, 0)
       dev_set_colored (3)
       disp_3d_coord_system (WindowHandle, StartCamParam, CalObjInCamPose, 0.05)
       * Read pose of tool in robot base coordinates (ToolInBasePose)
       read_pose (DataNameStart + (I+1)$'002d' + '.dat', ToolInBasePose)
       *Set the pose in m, the robot gives the information in mm
       Rob_X:=ToolInBasePose[0]/1000
       Rob_Y:=ToolInBasePose[1]/1000
       Rob_Z:=ToolInBasePose[2]/1000
       create_pose(Rob_X,Rob_Y,Rob_Z,Rob_C,Rob_B,Rob_A,'Rp+T', 'abg', 'point'
       ,ToolInBasePose)
       * Set the pose tool in robot base coordinates in the calibration data model
       set_calib_data (CalibDataID, 'tool', I, 'tool_in_base_pose', ToolInBasePose)


       disp_continue_message (WindowHandle, 'black', 'true')
       stop ()
endfor
disp_message (WindowHandle, 'All relevant data has been set in the calibration data
model', 'window', 12, 12, 'black', 'true')
disp_continue_message (WindowHandle, 'black', 'true')
stop ()


* Perform hand-eye calibration
dev_display (Image)
disp_message (WindowHandle, 'Performing the hand-eye calibration', 'window', 12, 12,
'black', 'true')
calibrate_hand_eye (CalibDataID, PoseErrors)


* Query the Robot Base in Camera Pose and saved it
get_calib_data (CalibDataID, 'camera', 0, 'base_in_cam_pose', BaseInCamPose)
write_pose (BaseInCamPose, DataNameStart + 'final_pose_cam_base.dat')


* Display calibration errors of the hand-eye calibration
Message := 'Quality of the results:      root mean square    maximum'
Message[1] := 'Translation part in m:          ' + PoseErrors[0]$'6.4f' + '
' + PoseErrors[2]$'6.4f'
```

```
Message[2] := 'Rotation part in degree:        ' + PoseErrors[1]$'6.4f' + '
' + PoseErrors[3]$'6.4f'
disp_message (WindowHandle, Message, 'window', 12, 12, 'black', 'true')
disp_continue_message (WindowHandle, 'black', 'true')
stop ()

* Clear the data model
clear_calib_data (CalibDataID)
```

A4.6. Send Pose to Robot System

```
remote_IP_address:='10.10.107.25'
port:=9901
timeout:=200

*Read Poses from the calibration and the matching
read_pose('C:/Documents and Settings/acro/Desktop/SmartFactory/Calibration/Hand-eye
calibration/Robot_poses/Robot_Posefinal_pose_cam_base.dat', PoseBaseinCam)
pose_to_hom_mat3d(PoseBaseinCam,Cam_H_RobBase)
read_pose('C:/Documents and
Settings/acro/Desktop/SmartFactory/matching/Cylinder/Cylinder47mminCam_Pose.dat',
PoseCylinderinCam)
pose_to_hom_mat3d(PoseCylinderinCam,Cam_H_Cylinder)

*Do transformation to obtain coordinate of the cylinder in robot base coordinate system
* RobBase_H_Cylinder= RobBase_H_Cam * Cam_H_Cylinder
hom_mat3d_invert(Cam_H_RobBase,RobBase_H_Cam)
hom_mat3d_compose(RobBase_H_Cam,Cam_H_Cylinder,RobBase_H_Cylinder)
hom_mat3d_to_pose(RobBase_H_Cylinder,Pose_Cylinder_in_RobBase)

* Set Offsets
Pose_Cylinder_in_RobBase[0]:=(Pose_Cylinder_in_RobBase[0]-0.0045)*1000
Pose_Cylinder_in_RobBase[1]:=(Pose_Cylinder_in_RobBase[1]-0.0025)*1000
Pose_Cylinder_in_RobBase[2]:=(Pose_Cylinder_in_RobBase[2]+0.0041)*1000
Change:=Pose_Cylinder_in_RobBase[5]
Pose_Cylinder_in_RobBase[5]:=Pose_Cylinder_in_RobBase[3]
```

```
Pose_Cylinder_in_RobBase[3]:= Change

convert_pose_type(Pose_Cylinder_in_RobBase,'Rp+T','abg','point',Pose_Cylinder_in_RobB
ase)

write_pose(Pose_Cylinder_in_RobBase,'CylinderinrobBase_Pose.dat')


*Convert pose to send into a string

tuple_string(Pose_Cylinder_in_RobBase,'-',Cylinder_RobBase_String)


*opening the socket

open_socket_connect (remote_IP_adres, 9901, ['protocol','timeout'], ['TCP',timeout],
Socket)


*preparing data to send

data_send_write:= '<?xml version="1.0" encoding="UTF-8"?><SetVar VarValue="{FRAME: X
'+Cylinder_RobBase_String[0]+',Y '+Cylinder_RobBase_String[1]+',Z
'+Cylinder_RobBase_String[2]+',A 0,B 0,C 0}" VarName="BASE_DATA[3]"/>'


*sending data

send_data (Socket, 'z', data_send_write, [])


*closing socket

close_socket(Socket)
```

A5. Halcon Files

A5.1. CamParam.calc

```
#

# INTERNAL CAMERA PARAMETERS

#

# ===== generic parameter syntax ==========================================

#

# Syntax:      ParGroup: <pargroupname>;

#                        <pargroup_description>;

#
```

```
#               <name> : <shortname> : <default_value>;
#                        <type> : <lower_bound> : <upper_bound>;
#                        <description>;
#
#                [ <type> ::= BOOL|XBOOL|INT|FLOAT|DOUBLE|STRING ]
#
# =========================================================================
##############################################################################
#
#  Camera : Parameter
#    >  Focus
#    >  Kappa
#    >  Sx
#    >  Sy
#    >  Cx
#    >  Cy
#    >  ImageWidth
#    >  ImageHeight
#
##############################################################################
ParGroup: Camera: Parameter;
       "Internal camera parameters";
Focus:foc:    0.0267014393180477;
       DOUBLE:0.0:;
       "Focal length of the lens";
Kappa:kappa:  -273.173782923715;
       DOUBLE::;
       "Radial distortion coefficient";
Sx:sx: 3.19209913486419e-006;
       DOUBLE:0.0:;
       "Width of a cell on the sensor";
Sy:sy: 3.2e-006;
       DOUBLE:0.0:;
       "Height of a cell on the sensor";
Cx:cx: 1174.32362533896;
       DOUBLE::;
       "X-coordinate of the image center";
Cy:cy: 595.948638426092;
       DOUBLE::;
       "Y-coordinate of the image center";
ImageWidth:imgw:     2048;
       INT:1:32768;
       "Width of the images";
ImageHeight:imgh:    1536;
       INT:1:32768;
       "Height of the images";
```

A5.2. Cam_Pose.dat

```
#
# 3D POSE PARAMETERS: rotation and translation
#
# Used representation type:
f 0
# Rotation angles [deg] or Rodriguez vector:
r 342.224549017376 1.85621062204843 88.6465031518263
# Translation vector (x y z [m]):
t -0.10898647773032 0.0811096600584384 1.02764481550875
```

A6. Visual Basic Codes

A6.1. Acquirement.vb

```
Option Strict Off
Option Explicit
Imports HalconDotNet
Module HDevelopExportAcquirement
Public  hv_ExpDefaultWinHandle As HTuple

    Public Sub HDevelopStopAcq()
        Call MessageBox.Show("Image Acquired Succesfully")
    End Sub

' Main procedure
    Private Sub actionAcq()

        ' Local control variables
        Dim hv_AcqHandle As HTuple = New HTuple
        Dim ho_Image As HObject = Nothing

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_Image)
        HOperatorSet.CloseAllFramegrabbers()


        HOperatorSet.OpenFramegrabber(New HTuple("uEye"), New HTuple(1), New
HTuple(1), _
            New HTuple(0), New HTuple(0), New HTuple(0), New HTuple(0), New
HTuple("default"), _
            New HTuple(8), New HTuple("default"), New HTuple(-1), New
HTuple("false"), _
            New HTuple("default"), New HTuple("0"), New HTuple(0), New HTuple(-1),
hv_AcqHandle)
        HOperatorSet.GrabImageStart(hv_AcqHandle, New HTuple(-1))

        ho_Image.Dispose()
        HOperatorSet.GrabImage(ho_Image, hv_AcqHandle)
        HOperatorSet.DispObj(ho_Image, hv_ExpDefaultWinHandle)
        HDevelopStopAcq()

#If USE_DO_EVENTS Then
    ' Please note: The call of DoEvents() is only a hack to
    ' enable VB to react on events. Please change the code
    ' so that it can handle events in a standard way.
    System.Windows.Forms.Application.DoEvents()
#End If

        HOperatorSet.CloseFramegrabber(hv_AcqHandle)
        ho_Image.Dispose()

    End Sub



    Sub InitHalcon()
        ' Default settings used in HDevelop
        Call HOperatorSet.SetSystem(New HTuple("width"), New HTuple(512))

        Call HOperatorSet.SetSystem(New HTuple("height"), New HTuple(512))

    End Sub

    Sub RunHalconAcq(ByRef Window As HTuple)
        hv_ExpDefaultWinHandle = Window
        Call actionAcq()
    End Sub

End Module
```

A6.2. ShapeModelCreation.vb

```
Option Strict Off
Option Explicit On

Imports HalconDotNet

Module HDevelopExportShapeModelCreation

    Public hv_ExpDefaultWinHandle As HTuple
    Public hv_ExpDefaultWinMenuHandle As HTuple
    Public hv_ExpDefaultWinViewHandle As HTuple
    Public hv_ExpDefaultWinBufferHandle As HTuple

    Public Sub HDevelopStop()
        Call MessageBox.Show("Press button to continue", "Program stop")
    End Sub

    ' Procedures
    ' External procedures
    ' Chapter: Develop
    ' Short Description: Switch dev_update_pc, dev_update_var and dev_update_window
to 'off'.
    Public Sub dev_update_off()
        ' Initialize local and output iconic variables

        'This procedure sets different update settings to 'off'.
        'This is useful to get the best performance and reduce overhead.
        '
        ' dev_update_pc(...); only in hdevelop
        ' dev_update_var(...); only in hdevelop
        ' dev_update_window(...); only in hdevelop

        Exit Sub
    End Sub


    ' Chapter: Graphics / Text
    ' Short Description: This procedure writes a text message.
    Public Sub disp_message(ByVal hv_WindowHandle As HTuple, ByVal hv_String As
HTuple, _
        ByVal hv_CoordSystem As HTuple, ByVal hv_Row As HTuple, ByVal hv_Column As
HTuple, _
        ByVal hv_Color As HTuple, ByVal hv_Box As HTuple)


        ' Local control variables
        Dim hv_Red As HTuple = New HTuple, hv_Green As HTuple = New HTuple
        Dim hv_Blue As HTuple = New HTuple, hv_Row1Part As HTuple = New HTuple
        Dim hv_Column1Part As HTuple = New HTuple, hv_Row2Part As HTuple = New HTuple
        Dim hv_Column2Part As HTuple = New HTuple, hv_RowWin As HTuple = New HTuple
        Dim hv_ColumnWin As HTuple = New HTuple, hv_WidthWin As HTuple = New HTuple
        Dim hv_HeightWin As HTuple = New HTuple, hv_MaxAscent As HTuple = New HTuple
        Dim hv_MaxDescent As HTuple = New HTuple, hv_MaxWidth As HTuple = New HTuple
        Dim hv_MaxHeight As HTuple = New HTuple, hv_R1 As HTuple = New HTuple
        Dim hv_C1 As HTuple = New HTuple, hv_FactorRow As HTuple = New HTuple
        Dim hv_FactorColumn As HTuple = New HTuple, hv_Width As HTuple = New HTuple
        Dim hv_Index As HTuple = New HTuple, hv_Ascent As HTuple = New HTuple
        Dim hv_Descent As HTuple = New HTuple, hv_W As HTuple = New HTuple
        Dim hv_H As HTuple = New HTuple, hv_FrameHeight As HTuple = New HTuple
        Dim hv_FrameWidth As HTuple = New HTuple, hv_R2 As HTuple = New HTuple
        Dim hv_C2 As HTuple = New HTuple, hv_DrawMode As HTuple = New HTuple
        Dim hv_Exception As HTuple = New HTuple, hv_CurrentColor As HTuple = New
HTuple

        Dim hv_Color_COPY_INP_TMP As HTuple
        hv_Color_COPY_INP_TMP = hv_Color.Clone()
        Dim hv_Column_COPY_INP_TMP As HTuple
        hv_Column_COPY_INP_TMP = hv_Column.Clone()
        Dim hv_Row_COPY_INP_TMP As HTuple
        hv_Row_COPY_INP_TMP = hv_Row.Clone()
        Dim hv_String_COPY_INP_TMP As HTuple
        hv_String_COPY_INP_TMP = hv_String.Clone()

        ' Initialize local and output iconic variables

        'This procedure displays text in a graphics window.
        '
        'Input parameters:
        'WindowHandle: The WindowHandle of the graphics window, where
        '   the message should be displayed
        'String: A tuple of strings containing the text message to be displayed
        'CoordSystem: If set to 'window', the text position is given
        '   with respect to the window coordinate system.
        '   If set to 'image', image coordinates are used.
        '   (This may be useful in zoomed images.)
        'Row: The row coordinate of the desired text position
        '   If set to -1, a default value of 12 is used.
        'Column: The column coordinate of the desired text position
        '   If set to -1, a default value of 12 is used.
        'Color: defines the color of the text as string.
        '   If set to [], '' or 'auto' the currently set color is used.
        '   If a tuple of strings is passed, the colors are used cyclically
        '   for each new textline.
        'Box: If set to 'true', the text is written within a white box.
        '
        'prepare window
        HOperatorSet.GetRgb(hv_WindowHandle, hv_Red, hv_Green, hv_Blue)
        HOperatorSet.GetPart(hv_WindowHandle, hv_Row1Part, hv_Column1Part,
hv_Row2Part, _
            hv_Column2Part)
        HOperatorSet.GetWindowExtents(hv_WindowHandle, hv_RowWin, hv_ColumnWin, _
            hv_WidthWin, hv_HeightWin)
        HOperatorSet.SetPart(hv_WindowHandle, New HTuple(0), New HTuple(0),
hv_HeightWin.TupleSub( _
            New HTuple(1)), hv_WidthWin.TupleSub(New HTuple(1)))
        '
        'default settings
        If New HTuple(hv_Row_COPY_INP_TMP.TupleEqual(New HTuple(-1))).I() Then
            hv_Row_COPY_INP_TMP = New HTuple(12)
        End If
        If New HTuple(hv_Column_COPY_INP_TMP.TupleEqual(New HTuple(-1))).I() Then
            hv_Column_COPY_INP_TMP = New HTuple(12)
        End If
        If New HTuple(hv_Color_COPY_INP_TMP.TupleEqual(New HTuple())).I() Then
            hv_Color_COPY_INP_TMP = New HTuple("")
        End If
        '
        hv_String_COPY_INP_TMP = (((((New
HTuple("")).TupleAdd(hv_String_COPY_INP_TMP))).TupleAdd( _
            New HTuple("")))).TupleSplit(New HTuple("" + Chr(10)))
        '
        'Estimate extentions of text depending on font size.
        HOperatorSet.GetFontExtents(hv_WindowHandle, hv_MaxAscent, hv_MaxDescent, _
            hv_MaxWidth, hv_MaxHeight)
        If New HTuple(hv_CoordSystem.TupleEqual(New HTuple("window"))).I() Then
            hv_R1 = hv_Row_COPY_INP_TMP.Clone()
            hv_C1 = hv_Column_COPY_INP_TMP.Clone()
        Else
            'transform image to window coordinates
            hv_FactorRow = (((New
HTuple(1.0)).TupleMult(hv_HeightWin))).TupleDiv(((hv_Row2Part.TupleSub( _
                hv_Row1Part))).TupleAdd(New HTuple(1)))
            hv_FactorColumn = (((New
HTuple(1.0)).TupleMult(hv_WidthWin))).TupleDiv(((hv_Column2Part.TupleSub( _
                hv_Column1Part))).TupleAdd(New HTuple(1)))
            hv_R1 = ((((hv_Row_COPY_INP_TMP.TupleSub(hv_Row1Part))).TupleAdd(New
HTuple(0.5)))).TupleMult( _
                hv_FactorRow)
            hv_C1 =
((((hv_Column_COPY_INP_TMP.TupleSub(hv_Column1Part))).TupleAdd(New
HTuple(0.5)))).TupleMult( _
                hv_FactorColumn)
        End If
        '
        'display text box depending on text size
        If New HTuple(hv_Box.TupleEqual(New HTuple("true"))).I() Then
            'calculate box extents
            hv_String_COPY_INP_TMP = (((New HTuple("
")).TupleAdd(hv_String_COPY_INP_TMP))).TupleAdd( _
                New HTuple(" "))
            hv_Width = New HTuple()
            For hv_Index = (New HTuple(0)) To ((New
HTuple(hv_String_COPY_INP_TMP.TupleLength( _
                )))).TupleSub(New HTuple(1)) Step (New HTuple(1))
                HOperatorSet.GetStringExtents(hv_WindowHandle,
hv_String_COPY_INP_TMP.TupleSelect( _
                    hv_Index), hv_Ascent, hv_Descent, hv_W, hv_H)
                hv_Width = hv_Width.TupleConcat(hv_W)
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            hv_FrameHeight = hv_MaxHeight.TupleMult(New
HTuple(hv_String_COPY_INP_TMP.TupleLength( _
                )))
            hv_FrameWidth = (((New HTuple(0)).TupleConcat(hv_Width))).TupleMax()
            hv_R2 = hv_R1.TupleAdd(hv_FrameHeight)
            hv_C2 = hv_C1.TupleAdd(hv_FrameWidth)
            'display rectangles
            HOperatorSet.GetDraw(hv_ExpDefaultWinHandle, hv_DrawMode)
            HOperatorSet.SetDraw(hv_ExpDefaultWinHandle, New HTuple("fill"))
            HOperatorSet.SetColor(hv_ExpDefaultWinHandle, New HTuple("light gray"))
            HOperatorSet.DispRectangle1(hv_ExpDefaultWinHandle, hv_R1.TupleAdd(New
HTuple(3)), _
                hv_C1.TupleAdd(New HTuple(3)), hv_R2.TupleAdd(New HTuple(3)),
hv_C2.TupleAdd( _
                New HTuple(3)))
            HOperatorSet.SetColor(hv_ExpDefaultWinHandle, New HTuple("white"))
            HOperatorSet.DispRectangle1(hv_ExpDefaultWinHandle, hv_R1, hv_C1, hv_R2,
hv_C2)
            HOperatorSet.SetDraw(hv_ExpDefaultWinHandle, hv_DrawMode)
        ElseIf New HTuple(hv_Box.TupleNotEqual(New HTuple("false"))).I() Then
            hv_Exception = New HTuple("Wrong value of control parameter Box")
            Throw New HalconException(hv_Exception)
        End If
        'Write text.
        For hv_Index = (New HTuple(0)) To ( _
            (New HTuple(hv_String_COPY_INP_TMP.TupleLength()))).TupleSub(New
HTuple(1)) Step (New HTuple(1))
            hv_CurrentColor = hv_Color_COPY_INP_TMP.TupleSelect(hv_Index.TupleMod(New
HTuple( _
                hv_Color_COPY_INP_TMP.TupleLength())))
            If ((New HTuple(hv_CurrentColor.TupleNotEqual(New
HTuple(""))))).TupleAnd(New HTuple( _
```

```
                hv_CurrentColor.TupleNotEqual(New HTuple("auto")))).I() Then
                HOperatorSet.SetColor(hv_ExpDefaultWinHandle, hv_CurrentColor)
            Else
                HOperatorSet.SetRgb(hv_ExpDefaultWinHandle, hv_Red, hv_Green,
hv_Blue)
            End If
            hv_Row_COPY_INP_TMP = hv_R1.TupleAdd(hv_MaxHeight.TupleMult(hv_Index))
            HOperatorSet.SetTposition(hv_ExpDefaultWinHandle, hv_Row_COPY_INP_TMP,
hv_C1)
            HOperatorSet.WriteString(hv_ExpDefaultWinHandle,
hv_String_COPY_INP_TMP.TupleSelect( _
                hv_Index))
#If USE_DO_EVENTS Then
    ' Please note: The call of DoEvents() is only a hack to
    ' enable VB to react on events. Please change the code
    ' so that it can handle events in a standard way.
    System.Windows.Forms.Application.DoEvents()
#End If
        Next
        'reset changed window settings
        HOperatorSet.SetRgb(hv_ExpDefaultWinHandle, hv_Red, hv_Green, hv_Blue)
        HOperatorSet.SetPart(hv_ExpDefaultWinHandle, hv_Row1Part, hv_Column1Part,
hv_Row2Part, _
            hv_Column2Part)

        Exit Sub
    End Sub

    ' Chapter: Graphics / Text
    ' Short Description: Set font independent of OS
    Public Sub set_display_font(ByVal hv_WindowHandle As HTuple, ByVal hv_Size As
HTuple, _
        ByVal hv_Font As HTuple, ByVal hv_Bold As HTuple, ByVal hv_Slant As HTuple)


        ' Local control variables
        Dim hv_OS As HTuple = New HTuple, hv_Exception As HTuple = New HTuple
        Dim hv_BoldString As HTuple = New HTuple, hv_SlantString As HTuple = New
HTuple
        Dim hv_AllowedFontSizes As HTuple = New HTuple, hv_Distances As HTuple = New
HTuple
        Dim hv_Indices As HTuple = New HTuple, hv_Fonts As HTuple = New HTuple
        Dim hv_FontSelRegexp As HTuple = New HTuple, hv_FontsCourier As HTuple = New
HTuple

        Dim hv_Bold_COPY_INP_TMP As HTuple
        hv_Bold_COPY_INP_TMP = hv_Bold.Clone()
        Dim hv_Font_COPY_INP_TMP As HTuple
        hv_Font_COPY_INP_TMP = hv_Font.Clone()
        Dim hv_Size_COPY_INP_TMP As HTuple
        hv_Size_COPY_INP_TMP = hv_Size.Clone()
        Dim hv_Slant_COPY_INP_TMP As HTuple
        hv_Slant_COPY_INP_TMP = hv_Slant.Clone()

        ' Initialize local and output iconic variables

        'This procedure sets the text font of the current window with
        'the specified attributes.
        'It is assumed that following fonts are installed on the system:
        'Windows: Courier New, Arial Times New Roman
        'Mac OS X: CourierNewPS, Arial, TimesNewRomanPS
        'Linux: courier, helvetica, times
        'Because fonts are displayed smaller on Linux than on Windows,
        'a scaling factor of 1.25 is used the get comparable results.
        'For Linux, only a limited number of font sizes is supported,
        'to get comparable results, it is recommended to use one of the
        'following sizes: 9, 11, 14, 16, 20, 27
        '(which will be mapped internally on Linux systems to 11, 14, 17, 20, 25, 34)
        '
        'Input parameters:
        'WindowHandle: The graphics window for which the font will be set
        'Size: The font size. If Size=-1, the default of 16 is used.
        'Bold: If set to 'true', a bold font is used
        'Slant: If set to 'true', a slanted font is used
        '
        HOperatorSet.GetSystem(New HTuple("operating_system"), hv_OS)
        ' dev_get_preferences(...); only in hdevelop
        ' dev_set_preferences(...); only in hdevelop
        If ((New HTuple(hv_Size_COPY_INP_TMP.TupleEqual(New HTuple()))).TupleOr(New
HTuple( _
            hv_Size_COPY_INP_TMP.TupleEqual(New HTuple(-1))))).I() Then
            hv_Size_COPY_INP_TMP = New HTuple(16)
        End If
        If New HTuple(((hv_OS.TupleSubstr(New HTuple(0), New
HTuple(2)))).TupleEqual(New HTuple("Win"))).I() Then
            'Set font on Windows systems
            If ((((New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("mono"))))).TupleOr( _
                New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("Courier"))))))).TupleOr( _
                New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("courier")))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("Courier New")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("sans"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("Arial")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("serif"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("Times New Roman")
            End If
            If New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                hv_Bold_COPY_INP_TMP = New HTuple(1)
            ElseIf New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_Bold_COPY_INP_TMP = New HTuple(0)
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Bold")
                Throw New HalconException(hv_Exception)
            End If
            If New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                hv_Slant_COPY_INP_TMP = New HTuple(1)
            ElseIf New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_Slant_COPY_INP_TMP = New HTuple(0)
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Slant")
                Throw New HalconException(hv_Exception)
            End If
            Try
                HOperatorSet.SetFont(hv_WindowHandle, (((((((((((((New HTuple("-
")).TupleAdd( _
                    hv_Font_COPY_INP_TMP))).TupleAdd(New HTuple("-
")))).TupleAdd(hv_Size_COPY_INP_TMP))).TupleAdd( _
                    New HTuple("-*-
")))).TupleAdd(hv_Slant_COPY_INP_TMP))).TupleAdd(New HTuple("-*-*-")))).TupleAdd( _
                    hv_Bold_COPY_INP_TMP))).TupleAdd(New HTuple("-")))
                ' catch (Exception)
            Catch HDevExpDefaultException1 As HalconException
```

```
                HDevExpDefaultException1.ToHTuple(hv_Exception)
                'throw (Exception)
            End Try
        ElseIf New HTuple(((hv_OS.TupleSubstr(New HTuple(0), New
HTuple(2)))).TupleEqual( _
            New HTuple("Dar"))).I() Then
            'Set font on Mac OS X systems
            If New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                hv_BoldString = New HTuple("Bold")
            ElseIf New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_BoldString = New HTuple("")
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Bold")
                Throw New HalconException(hv_Exception)
            End If
            If New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                hv_SlantString = New HTuple("Italic")
            ElseIf New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_SlantString = New HTuple("")
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Slant")
                Throw New HalconException(hv_Exception)
            End If
            If ((((New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("mono"))))).TupleOr( _
                New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("Courier")))))).TupleOr( _
                New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("courier")))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("CourierNewPS")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("sans"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("Arial")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("serif"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("TimesNewRomanPS")
            End If
            If ((New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New
HTuple("true"))))).TupleOr( _
                New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New HTuple("true")))).I()
Then
                hv_Font_COPY_INP_TMP = ((((hv_Font_COPY_INP_TMP.TupleAdd(New
HTuple("-")))).TupleAdd( _
                    hv_BoldString))).TupleAdd(hv_SlantString)
            End If
            hv_Font_COPY_INP_TMP = hv_Font_COPY_INP_TMP.TupleAdd(New HTuple("MT"))
            Try
                HOperatorSet.SetFont(hv_WindowHandle,
((hv_Font_COPY_INP_TMP.TupleAdd( _
                    New HTuple("-")))).TupleAdd(hv_Size_COPY_INP_TMP))
                ' catch (Exception)
            Catch HDevExpDefaultException1 As HalconException
                HDevExpDefaultException1.ToHTuple(hv_Exception)
                'throw (Exception)
            End Try
        Else
            'Set font for UNIX systems
            hv_Size_COPY_INP_TMP = hv_Size_COPY_INP_TMP.TupleMult(New HTuple(1.25))
            hv_AllowedFontSizes = (((((New
HTuple(11)).TupleConcat(14)).TupleConcat(17)).TupleConcat( _
                20)).TupleConcat(25)).TupleConcat(34)
            If New
HTuple(((hv_AllowedFontSizes.TupleFind(hv_Size_COPY_INP_TMP))).TupleEqual( _
                New HTuple(-1))).I() Then
                hv_Distances =
((hv_AllowedFontSizes.TupleSub(hv_Size_COPY_INP_TMP))).TupleAbs( _
                    )
                HOperatorSet.TupleSortIndex(hv_Distances, hv_Indices)
                hv_Size_COPY_INP_TMP =
hv_AllowedFontSizes.TupleSelect(hv_Indices.TupleSelect( _
                    New HTuple(0)))
            End If
            If ((New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("mono"))))).TupleOr( _
                New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("Courier")))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("courier")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("sans"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("helvetica")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("serif"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("times")
            End If
            If New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                hv_Bold_COPY_INP_TMP = New HTuple("bold")
            ElseIf New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_Bold_COPY_INP_TMP = New HTuple("medium")
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Bold")
                Throw New HalconException(hv_Exception)
            End If
            If New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                If New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("times"))).I() Then
                    hv_Slant_COPY_INP_TMP = New HTuple("i")
                Else
                    hv_Slant_COPY_INP_TMP = New HTuple("o")
                End If
            ElseIf New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_Slant_COPY_INP_TMP = New HTuple("r")
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Slant")
                Throw New HalconException(hv_Exception)
            End If
            Try
                HOperatorSet.SetFont(hv_WindowHandle, (((((((((((((New HTuple("-
adobe-")).TupleAdd( _
                    hv_Font_COPY_INP_TMP))).TupleAdd(New HTuple("-
")))).TupleAdd(hv_Bold_COPY_INP_TMP))).TupleAdd( _
                    New HTuple("-")))).TupleAdd(hv_Slant_COPY_INP_TMP))).TupleAdd(New
HTuple("-normal-*-")))).TupleAdd( _
                    hv_Size_COPY_INP_TMP))).TupleAdd(New HTuple("-*-*-*-*-*-*-*")))
                ' catch (Exception)
            Catch HDevExpDefaultException1 As HalconException
                HDevExpDefaultException1.ToHTuple(hv_Exception)
                If ((New HTuple(((hv_OS.TupleSubstr(New HTuple(0), New
HTuple(4)))).TupleEqual( _
                    New HTuple("Linux"))))).TupleAnd(New
HTuple(hv_Font_COPY_INP_TMP.TupleEqual( _
                    New HTuple("courier")))).I() Then
                    HOperatorSet.QueryFont(hv_ExpDefaultWinHandle, hv_Fonts)
                    hv_FontSelRegexp = (((((New HTuple("^-[^-]*-[^-]*[Cc]ourier[^-]*-
")).TupleAdd( _
                        hv_Bold_COPY_INP_TMP))).TupleAdd(New HTuple("-
")))).TupleAdd(hv_Slant_COPY_INP_TMP)
                    hv_FontsCourier =
((hv_Fonts.TupleRegexpSelect(hv_FontSelRegexp))).TupleRegexpMatch( _
                        hv_FontSelRegexp)
                    If New HTuple(((New
HTuple(hv_FontsCourier.TupleLength()))).TupleEqual(New HTuple(0))).I() Then
                        hv_Exception = New HTuple("Wrong font name")
                        'throw (Exception)
                    Else
                        Try
                            HOperatorSet.SetFont(hv_WindowHandle,
((((((hv_FontsCourier.TupleSelect( _
                                New HTuple(0)))).TupleAdd(New HTuple("-normal-*-
")))).TupleAdd(hv_Size_COPY_INP_TMP))).TupleAdd( _
                                New HTuple("-*-*-*-*-*-*-*")))
                            ' catch (Exception)
                        Catch HDevExpDefaultException2 As HalconException
                            HDevExpDefaultException2.ToHTuple(hv_Exception)
                            'throw (Exception)
                        End Try
                    End If
                End If
                'throw (Exception)
            End Try
        End If
        ' dev_set_preferences(...); only in hdevelop

        Exit Sub
    End Sub

    ' Local procedures
    ' Chapter: Matching-3D
    ' Short Description: Creates rectangular regions at the top or bottom of the
current window
    Public Sub gen_menu_regions(ByRef ho_MenuRegions As HObject, ByVal
hv_WindowHandle As HTuple, _
        ByVal hv_NumCols As HTuple, ByVal hv_NumRows As HTuple, ByVal
hv_PercentageHeight As HTuple, _
        ByVal hv_TopBottom As HTuple)


        ' Stack for temporary objects
        Dim OTemp(10) As HObject

        ' Local iconic variables
        Dim ho_Rectangle As HObject = Nothing


        ' Local control variables
        Dim hv_PartRow1 As HTuple = New HTuple, hv_PartCol1 As HTuple = New HTuple
        Dim hv_PartRow2 As HTuple = New HTuple, hv_PartCol2 As HTuple = New HTuple
        Dim hv_Height As HTuple = New HTuple, hv_Width As HTuple = New HTuple
        Dim hv_MenuHeight As HTuple = New HTuple, hv_MenuWidth As HTuple = New HTuple
        Dim hv_ButtonHeight As HTuple = New HTuple, hv_ButtonWidth As HTuple = New
HTuple
        Dim hv_Row0 As HTuple = New HTuple, hv_Col0 As HTuple = New HTuple
        Dim hv_c As HTuple = New HTuple, hv_r As HTuple = New HTuple

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_MenuRegions)
        HOperatorSet.GenEmptyObj(ho_Rectangle)
        Try
            'Generate the menu buttons
            If ((New HTuple(hv_TopBottom.TupleNotEqual(New
HTuple("top"))))).TupleAnd(New HTuple( _
                hv_TopBottom.TupleNotEqual(New HTuple("bottom")))).I() Then
                HDevelopStop()
                'Wrong parameter: TopBottom must be 'top' or 'bottom'
            End If
            HOperatorSet.GetPart(hv_WindowHandle, hv_PartRow1, hv_PartCol1,
hv_PartRow2, _
                hv_PartCol2)
            hv_Height = ((hv_PartRow2.TupleSub(hv_PartRow1))).TupleAdd(New HTuple(1))
            hv_Width = ((hv_PartCol2.TupleSub(hv_PartCol1))).TupleAdd(New HTuple(1))
            hv_MenuHeight = ((hv_Height.TupleMult(hv_PercentageHeight))).TupleDiv(New
HTuple(100.0))
            hv_MenuWidth = hv_Width.Clone()
            hv_ButtonHeight = hv_MenuHeight.TupleDiv(hv_NumRows)
            hv_ButtonWidth = hv_MenuWidth.TupleDiv(hv_NumCols)
            If New HTuple(hv_TopBottom.TupleEqual(New HTuple("top"))).I() Then
                hv_Row0 = hv_PartRow1.Clone()
            End If
            If New HTuple(hv_TopBottom.TupleEqual(New HTuple("bottom"))).I() Then
                hv_Row0 = hv_PartRow2.TupleSub(hv_MenuHeight)
            End If
            hv_Col0 = hv_PartCol1.Clone()
            ho_MenuRegions.Dispose()
            HOperatorSet.GenEmptyObj(ho_MenuRegions)
            For hv_c = (New HTuple(0)) To hv_NumCols.TupleSub( _
                New HTuple(1)) Step (New HTuple(1))
                For hv_r = (New HTuple(0)) To hv_NumRows.TupleSub( _
                    New HTuple(1)) Step (New HTuple(1))
                    ho_Rectangle.Dispose()
                    HOperatorSet.GenRectangle1(ho_Rectangle,
hv_Row0.TupleAdd(hv_r.TupleMult( _
                        hv_ButtonHeight)),
hv_Col0.TupleAdd(hv_c.TupleMult(hv_ButtonWidth)), _

((((hv_Row0.TupleAdd(hv_r.TupleMult(hv_ButtonHeight)))).TupleAdd(hv_ButtonHeight))).T
upleSub( _
                        New HTuple(1)),
((((hv_Col0.TupleAdd(hv_c.TupleMult(hv_ButtonWidth)))).TupleAdd( _
                        hv_ButtonWidth))).TupleSub(New HTuple(1)))
                    HOperatorSet.ConcatObj(ho_MenuRegions, ho_Rectangle, OTemp(0))
                    ho_MenuRegions.Dispose()
                    ho_MenuRegions = OTemp(0)
#If USE_DO_EVENTS Then
        ' Please note: The call of DoEvents() is only a hack to
        ' enable VB to react on events. Please change the code
        ' so that it can handle events in a standard way.
        System.Windows.Forms.Application.DoEvents()
#End If
                Next
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            ho_Rectangle.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_Rectangle.Dispose()
            Throw HDevExpDefaultException
        End Try
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Determine the text of the selected button
    Public Sub get_selected_button(ByVal ho_MenuRegions As HObject, ByVal hv_Texts As
HTuple, _
        ByVal hv_WindowHandle As HTuple, ByVal hv_Row As HTuple, ByVal hv_Col As
HTuple, _
        ByRef hv_SelectedButton As HTuple)


        Dim sys As New HSystem()

        ' Local iconic variables
        Dim ho_MenuRegion As HObject = Nothing


        ' Local control variables
        Dim hv_NumberRegions As HTuple = New HTuple
        Dim hv_NumberTexts As HTuple = New HTuple, hv_i As HTuple = New HTuple
        Dim hv_IsInside As HTuple = New HTuple

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_MenuRegion)

        Try
            HOperatorSet.CountObj(ho_MenuRegions, hv_NumberRegions)
            hv_NumberTexts = New HTuple(hv_Texts.TupleLength())
            If New HTuple(hv_NumberRegions.TupleNotEqual(hv_NumberTexts)).I() Then
                HDevelopStop()
                'Wrong number of regions or texts
            End If
            hv_SelectedButton = New HTuple("")
            ' (dev_)set_check ("~give_error")
            For hv_i = (New HTuple(1)) To hv_NumberRegions Step (New HTuple(1))
                Try
                    ho_MenuRegion.Dispose()
                    HOperatorSet.SelectObj(ho_MenuRegions, ho_MenuRegion, hv_i)
                Catch e As HalconException
                    Dim err As Integer
                    err = e.GetErrorNumber()
                    If err < 0 Then
                        Throw e
                    End If
                End Try
                Try
                    HOperatorSet.TestRegionPoint(ho_MenuRegion, hv_Row, hv_Col,
hv_IsInside)
                Catch e As HalconException
                    Dim err As Integer
                    err = e.GetErrorNumber()
                    If err < 0 Then
                        Throw e
                    End If
                End Try
                If hv_IsInside.I() Then
                    hv_SelectedButton = hv_Texts.TupleSelect(hv_i.TupleSub(New
HTuple(1)))
                End If
#If USE_DO_EVENTS Then
        ' Please note: The call of DoEvents() is only a hack to
        ' enable VB to react on events. Please change the code
        ' so that it can handle events in a standard way.
```

```
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            ' (dev_)set_check ("give_error")
            ho_MenuRegion.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_MenuRegion.Dispose()

            Throw HDevExpDefaultException
        End Try
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Displays the menu buttons
    Public Sub disp_menu(ByVal ho_MenuRegions As HObject, ByVal hv_WindowHandle As
HTuple, _
        ByVal hv_Texts As HTuple)

        ' Local iconic variables
        Dim ho_MenuRegion As HObject = Nothing


        ' Local control variables
        Dim hv_NumberRegions As HTuple = New HTuple
        Dim hv_NumberTexts As HTuple = New HTuple, hv_i As HTuple = New HTuple

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_MenuRegion)

        Try
            'Display the previously created menu buttons
            HOperatorSet.CountObj(ho_MenuRegions, hv_NumberRegions)
            hv_NumberTexts = New HTuple(hv_Texts.TupleLength())
            If New HTuple(hv_NumberRegions.TupleNotEqual(hv_NumberTexts)).I() Then
                HDevelopStop()
                'Wrong number of regions or texts
            End If
            For hv_i = (New HTuple(1)) To hv_NumberRegions Step (New HTuple(1))
                ho_MenuRegion.Dispose()
                HOperatorSet.SelectObj(ho_MenuRegions, ho_MenuRegion, hv_i)
                disp_menu_button(ho_MenuRegion, hv_WindowHandle,
hv_Texts.TupleSelect(hv_i.TupleSub( _
                    New HTuple(1))))
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            ho_MenuRegion.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_MenuRegion.Dispose()

            Throw HDevExpDefaultException
        End Try
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Generate a contour in form of an arrow
    Public Sub gen_arrow_cont(ByRef ho_Arrow As HObject, ByVal hv_Row1 As HTuple,
ByVal hv_Column1 As HTuple, _
        ByVal hv_Row2 As HTuple, ByVal hv_Column2 As HTuple)


        ' Local iconic variables
        Dim ho_Cross1 As HObject = Nothing, ho_Cross2 As HObject = Nothing
        Dim ho_CrossP1 As HObject = Nothing, ho_CrossP2 As HObject = Nothing


        ' Local control variables
        Dim hv_Length As HTuple = New HTuple, hv_Angle As HTuple = New HTuple
        Dim hv_MinArrowLength As HTuple = New HTuple, hv_DRow As HTuple = New HTuple
        Dim hv_DCol As HTuple = New HTuple, hv_ArrowLength As HTuple = New HTuple
        Dim hv_Phi As HTuple = New HTuple, hv_P1R As HTuple = New HTuple
        Dim hv_P1C As HTuple = New HTuple, hv_P2R As HTuple = New HTuple
        Dim hv_P2C As HTuple = New HTuple

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_Arrow)
        HOperatorSet.GenEmptyObj(ho_Cross1)
        HOperatorSet.GenEmptyObj(ho_Cross2)
        HOperatorSet.GenEmptyObj(ho_CrossP1)
        HOperatorSet.GenEmptyObj(ho_CrossP2)

        Try
            'Generate a contour in form of a arrow
            hv_Length = New HTuple(7)
            hv_Angle = New HTuple(40)
            hv_MinArrowLength = New HTuple(2)
            hv_DRow = hv_Row2.TupleSub(hv_Row1)
            hv_DCol = hv_Column2.TupleSub(hv_Column1)
            hv_ArrowLength =
((((hv_DRow.TupleMult(hv_DRow))).TupleAdd(hv_DCol.TupleMult( _
                hv_DCol)))).TupleSqrt()
            If New HTuple(hv_ArrowLength.TupleLess(hv_MinArrowLength)).I() Then
                hv_Length = New HTuple(0)
            End If
            HOperatorSet.TupleAtan2(hv_DRow, hv_DCol.TupleNeg(), hv_Phi)
            hv_P1R =
hv_Row2.TupleSub(hv_Length.TupleMult(((hv_Phi.TupleSub(hv_Angle.TupleRad( _
                )))).TupleSin()))
            hv_P1C =
hv_Column2.TupleAdd(hv_Length.TupleMult(((hv_Phi.TupleSub(hv_Angle.TupleRad( _
                )))).TupleCos()))
            hv_P2R =
hv_Row2.TupleSub(hv_Length.TupleMult(((hv_Phi.TupleAdd(hv_Angle.TupleRad( _
                )))).TupleSin()))
            hv_P2C =
hv_Column2.TupleAdd(hv_Length.TupleMult(((hv_Phi.TupleAdd(hv_Angle.TupleRad( _
                )))).TupleCos()))
            ho_Cross1.Dispose()
            HOperatorSet.GenCrossContourXld(ho_Cross1, hv_Row1, hv_Column1, New
HTuple(6), _
                New HTuple(0.785398))
            ho_Cross2.Dispose()
            HOperatorSet.GenCrossContourXld(ho_Cross2, hv_Row2, hv_Column2, New
HTuple(6), _
                New HTuple(0.785398))
            ho_CrossP1.Dispose()
            HOperatorSet.GenCrossContourXld(ho_CrossP1, hv_P1R, hv_P1C, New
HTuple(6), New HTuple(0.785398))
            ho_CrossP2.Dispose()
            HOperatorSet.GenCrossContourXld(ho_CrossP2, hv_P2R, hv_P2C, New
HTuple(6), New HTuple(0.785398))
```

```
            ho_Arrow.Dispose()
            HOperatorSet.GenContourPolygonXld(ho_Arrow,
((((((hv_Row1.TupleConcat(hv_Row2))).TupleConcat( _
                hv_P1R))).TupleConcat(hv_Row2))).TupleConcat(hv_P2R),
((((((hv_Column1.TupleConcat( _

hv_Column2))).TupleConcat(hv_P1C))).TupleConcat(hv_Column2))).TupleConcat( _
                hv_P2C))
            ho_Cross1.Dispose()
            ho_Cross2.Dispose()
            ho_CrossP1.Dispose()
            ho_CrossP2.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_Cross1.Dispose()
            ho_Cross2.Dispose()
            ho_CrossP1.Dispose()
            ho_CrossP2.Dispose()

            Throw HDevExpDefaultException
        End Try
    End Sub
    ' Chapter: Matching-3D
    ' Short Description: Compute the vector cross product
    Public Sub tuple_vector_cross_product(ByVal hv_V1 As HTuple, ByVal hv_V2 As
HTuple, _
        ByRef hv_VC As HTuple)

        ' Initialize local and output iconic variables

        'The caller must ensure that the length of both input vectors is 3
        hv_VC = ((((hv_V1.TupleSelect(New
HTuple(1)))).TupleMult(hv_V2.TupleSelect(New HTuple(2))))).TupleSub( _
            ((hv_V1.TupleSelect(New HTuple(2)))).TupleMult(hv_V2.TupleSelect(New
HTuple(1))))
        hv_VC = hv_VC.TupleConcat(((((hv_V1.TupleSelect(New
HTuple(2)))).TupleMult(hv_V2.TupleSelect( _
            New HTuple(0))))).TupleSub(((hv_V1.TupleSelect(New
HTuple(0)))).TupleMult(hv_V2.TupleSelect( _
            New HTuple(2)))))
        hv_VC = hv_VC.TupleConcat(((((hv_V1.TupleSelect(New
HTuple(0)))).TupleMult(hv_V2.TupleSelect( _
            New HTuple(1))))).TupleSub(((hv_V1.TupleSelect(New
HTuple(1)))).TupleMult(hv_V2.TupleSelect( _
            New HTuple(0)))))

        Exit Sub
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Determine the optimum viewing distance for a 3d model
    Public Sub determine_optimum_viewing_distance(ByVal hv_ObjectModel3DID As HTuple,
_
        ByVal hv_CamParam As HTuple, ByVal hv_MinImageSize As HTuple, ByVal
hv_ImageCoverage As HTuple, _
        ByRef hv_Dist As HTuple)


        ' Local control variables
        Dim hv_BoundingBox As HTuple = New HTuple, hv_ReferencePoint As HTuple = New
HTuple
        Dim hv_P1 As HTuple = New HTuple, hv_P2 As HTuple = New HTuple
        Dim hv_P3 As HTuple = New HTuple, hv_P4 As HTuple = New HTuple
        Dim hv_P5 As HTuple = New HTuple, hv_P6 As HTuple = New HTuple
        Dim hv_P7 As HTuple = New HTuple, hv_P8 As HTuple = New HTuple
        Dim hv_D As HTuple = New HTuple, hv_MaxProjectionRadius As HTuple = New
HTuple
        Dim hv_Aperture As HTuple = New HTuple

        ' Initialize local and output iconic variables

        'Get the 3d bounding box and the center of the 3d model
        HOperatorSet.GetObjectModel3dParams(hv_ObjectModel3DID, New
HTuple("bounding_box1"), _
            hv_BoundingBox)
        HOperatorSet.GetObjectModel3dParams(hv_ObjectModel3DID, New
HTuple("reference_point"), _
            hv_ReferencePoint)
        '
        'Compute the corner points of the bounding box
        hv_P1 = ((((hv_BoundingBox.TupleSelect(New
HTuple(0)))).TupleConcat(hv_BoundingBox.TupleSelect( _
            New HTuple(1))))).TupleConcat(hv_BoundingBox.TupleSelect(New HTuple(2)))
        hv_P2 = ((((hv_BoundingBox.TupleSelect(New
HTuple(3)))).TupleConcat(hv_BoundingBox.TupleSelect( _
            New HTuple(1))))).TupleConcat(hv_BoundingBox.TupleSelect(New HTuple(2)))
        hv_P3 = ((((hv_BoundingBox.TupleSelect(New
HTuple(0)))).TupleConcat(hv_BoundingBox.TupleSelect( _
            New HTuple(4))))).TupleConcat(hv_BoundingBox.TupleSelect(New HTuple(2)))
        hv_P4 = ((((hv_BoundingBox.TupleSelect(New
HTuple(0)))).TupleConcat(hv_BoundingBox.TupleSelect( _
            New HTuple(1))))).TupleConcat(hv_BoundingBox.TupleSelect(New HTuple(5)))
        hv_P5 = ((((hv_BoundingBox.TupleSelect(New
HTuple(3)))).TupleConcat(hv_BoundingBox.TupleSelect( _
            New HTuple(4))))).TupleConcat(hv_BoundingBox.TupleSelect(New HTuple(2)))
        hv_P6 = ((((hv_BoundingBox.TupleSelect(New
HTuple(3)))).TupleConcat(hv_BoundingBox.TupleSelect( _
            New HTuple(1))))).TupleConcat(hv_BoundingBox.TupleSelect(New HTuple(5)))
        hv_P7 = ((((hv_BoundingBox.TupleSelect(New
HTuple(0)))).TupleConcat(hv_BoundingBox.TupleSelect( _
            New HTuple(4))))).TupleConcat(hv_BoundingBox.TupleSelect(New HTuple(5)))
        hv_P8 = ((((hv_BoundingBox.TupleSelect(New
HTuple(3)))).TupleConcat(hv_BoundingBox.TupleSelect( _
            New HTuple(4))))).TupleConcat(hv_BoundingBox.TupleSelect(New HTuple(5)))
        '
        'Compute the maximum distance from the model center to the corners of
        'the bounding box
        hv_D =
((((hv_P1.TupleSub(hv_ReferencePoint))).TupleMult(hv_P1.TupleSub(hv_ReferencePoint)))
).TupleSum( _
            )
        hv_D =
((hv_D.TupleConcat(((((hv_P2.TupleSub(hv_ReferencePoint))).TupleMult(hv_P2.TupleSub(
_

            hv_ReferencePoint)))).TupleSum()))).TupleMax()
        hv_D =
((hv_D.TupleConcat(((((hv_P3.TupleSub(hv_ReferencePoint))).TupleMult(hv_P3.TupleSub(
_

            hv_ReferencePoint)))).TupleSum()))).TupleMax()
        hv_D =
((hv_D.TupleConcat(((((hv_P4.TupleSub(hv_ReferencePoint))).TupleMult(hv_P4.TupleSub(
_

            hv_ReferencePoint)))).TupleSum()))).TupleMax()
        hv_D =
((hv_D.TupleConcat(((((hv_P5.TupleSub(hv_ReferencePoint))).TupleMult(hv_P5.TupleSub(
_

            hv_ReferencePoint)))).TupleSum()))).TupleMax()
        hv_D =
((hv_D.TupleConcat((((hv_P6.TupleSub(hv_ReferencePoint))).TupleMult(hv_P6.TupleSub(
_
            hv_ReferencePoint)))).TupleSum()))).TupleMax()
        hv_D =
((hv_D.TupleConcat((((hv_P7.TupleSub(hv_ReferencePoint))).TupleMult(hv_P7.TupleSub(
_
            hv_ReferencePoint)))).TupleSum()))).TupleMax()
        hv_D =
((hv_D.TupleConcat((((hv_P8.TupleSub(hv_ReferencePoint))).TupleMult(hv_P8.TupleSub(
_
            hv_ReferencePoint)))).TupleSum()))).TupleMax()
        '
        'Compute the optimum distance to make the projection cover the desired
        'image part.
        hv_MaxProjectionRadius = (((((New
HTuple(0.5)).TupleMult(hv_ImageCoverage))).TupleMult( _
            hv_MinImageSize))).TupleMult((((hv_CamParam.TupleSelect(New
HTuple(2)))).TupleConcat( _
            hv_CamParam.TupleSelect(New HTuple(3)))))).TupleMin())
        hv_Aperture = hv_MaxProjectionRadius.TupleAtan2(hv_CamParam.TupleSelect(New
HTuple(0)))
        hv_Dist = ((hv_D.TupleSqrt())).TupleDiv(hv_Aperture.TupleSin())

        Exit Sub
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Reflect the movement of the background image
    Public Sub analyze_graph_event_image(ByVal ho_BackgroundImage As HObject, ByVal
ho_ModelContours As HObject, _
        ByVal ho_TrackballContour As HObject, ByVal hv_Button As HTuple, ByVal hv_Row
As HTuple, _
        ByVal hv_Column As HTuple, ByVal hv_WindowHandle As HTuple, ByVal
hv_WindowHandleBuffer As HTuple, _
        ByVal hv_OffsetRowIn As HTuple, ByVal hv_OffsetColIn As HTuple, ByVal
hv_ImageNo As HTuple, _
        ByVal hv_VisualizeTrackball As HTuple, ByVal hv_ModelColor As HTuple, ByRef
hv_OffsetRowOut As HTuple, _
        ByRef hv_OffsetColOut As HTuple)


        Dim sys As New HSystem()

        ' Local iconic variables
        Dim ho_BackgroundImageTrans As HObject = Nothing


        ' Local control variables
        Dim hv_ErrorVar As HTuple
        Dim hv_Row1 As HTuple = New HTuple, hv_Column1 As HTuple = New HTuple
        Dim hv_Width As HTuple = New HTuple, hv_Height As HTuple = New HTuple
        Dim hv_HomMat2DIdentity As HTuple = New HTuple, hv_MRow1 As HTuple = New
HTuple
        Dim hv_MCol1 As HTuple = New HTuple, hv_Error As HTuple = New HTuple
        Dim hv_MRow2 As HTuple = New HTuple, hv_MCol2 As HTuple = New HTuple
        Dim hv_HomMat2DTranslate As HTuple = New HTuple

        Dim hv_Button_COPY_INP_TMP As HTuple
        hv_Button_COPY_INP_TMP = hv_Button.Clone()
        Dim hv_Column_COPY_INP_TMP As HTuple
        hv_Column_COPY_INP_TMP = hv_Column.Clone()
        Dim hv_Row_COPY_INP_TMP As HTuple
        hv_Row_COPY_INP_TMP = hv_Row.Clone()
        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_BackgroundImageTrans)

        Try
            hv_OffsetRowOut = hv_OffsetRowIn.Clone()
            hv_OffsetColOut = hv_OffsetColIn.Clone()
            If New HTuple(hv_ImageNo.TupleLessEqual(New HTuple(0))).I() Then
                ho_BackgroundImageTrans.Dispose()

                Exit Sub
            End If
            HOperatorSet.GetWindowExtents(hv_WindowHandle, hv_Row1, hv_Column1,
hv_Width, _
                hv_Height)
            ' Error variable 'hv_ErrorVar' activated
            hv_ErrorVar = New HTuple(2)
            HOperatorSet.HomMat2dIdentity(hv_HomMat2DIdentity)
            If New HTuple(hv_Button_COPY_INP_TMP.TupleEqual(New HTuple(1))).I() Then
                hv_MRow1 = hv_Row_COPY_INP_TMP.Clone()
                hv_MCol1 = hv_Column_COPY_INP_TMP.Clone()
                Do While New HTuple(hv_Button_COPY_INP_TMP.TupleEqual(New
HTuple(1))).I()
                    ' (dev_)set_check ("~give_error")
                    Try
                        hv_ErrorVar = 2
                        HOperatorSet.GetMposition(hv_WindowHandle,
hv_Row_COPY_INP_TMP, _
                            hv_Column_COPY_INP_TMP, hv_Button_COPY_INP_TMP)
                     Catch e As HalconException
                        hv_ErrorVar = New HTuple(e.GetErrorNumber())
                        If hv_ErrorVar.I() < 0 Then
                            Throw e
                        End If
                    End Try
                    hv_Error = hv_ErrorVar.Clone()
                    ' (dev_)set_check ("give_error")
                    If New HTuple(hv_Error.TupleEqual(New HTuple(2))).I() Then
                        hv_MRow2 = hv_Row_COPY_INP_TMP.Clone()
                        hv_MCol2 = hv_Column_COPY_INP_TMP.Clone()
                        If IsNothing(hv_OffsetRowOut) Then
                            hv_OffsetRowOut = New HTuple
                        End If
                        hv_OffsetRowOut(hv_ImageNo.TupleSub(New HTuple(1))) =
((((hv_OffsetRowOut.TupleSelect( _
                            hv_ImageNo.TupleSub(New
HTuple(1))))).TupleAdd(hv_MRow2))).TupleSub( _
                            hv_MRow1)
                        If IsNothing(hv_OffsetColOut) Then
                            hv_OffsetColOut = New HTuple
                        End If
                        hv_OffsetColOut(hv_ImageNo.TupleSub(New HTuple(1))) =
((((hv_OffsetColOut.TupleSelect( _
                            hv_ImageNo.TupleSub(New
HTuple(1))))).TupleAdd(hv_MCol2))).TupleSub( _
                            hv_MCol1)
                        HOperatorSet.SetColor(hv_WindowHandleBuffer, hv_ModelColor)
                        If New HTuple(hv_ImageNo.TupleGreater(New HTuple(0))).I()
Then
                            HOperatorSet.ClearWindow(hv_WindowHandleBuffer)
                            HOperatorSet.HomMat2dTranslate(hv_HomMat2DIdentity,
hv_OffsetRowOut.TupleSelect( _
                                hv_ImageNo.TupleSub(New HTuple(1))),
hv_OffsetColOut.TupleSelect( _
                                hv_ImageNo.TupleSub(New HTuple(1))),
hv_HomMat2DTranslate)
```

```
                            ho_BackgroundImageTrans.Dispose()
                            HOperatorSet.AffineTransImage(ho_BackgroundImage, 
ho_BackgroundImageTrans, _
                                hv_HomMat2DTranslate, New HTuple("constant"), New 
HTuple("false"))
                            HOperatorSet.DispImage(ho_BackgroundImageTrans, 
hv_WindowHandleBuffer)
                            HOperatorSet.SetTposition(hv_WindowHandleBuffer, New 
HTuple(10), New HTuple(10))
                            HOperatorSet.WriteString(hv_WindowHandleBuffer, (New 
HTuple("Image No.: ")).TupleAdd( _
                                hv_ImageNo.TupleString((New HTuple("2d")).S())))
                        Else
                            HOperatorSet.SetTposition(hv_WindowHandleBuffer, New 
HTuple(10), New HTuple(10))
                            HOperatorSet.WriteString(hv_WindowHandleBuffer, New 
HTuple("(No Background Image)"))
                        End If
                        HOperatorSet.SetTposition(hv_WindowHandleBuffer, New 
HTuple(10), hv_Width.TupleSub( _
                            New HTuple(200)))
                        HOperatorSet.WriteString(hv_WindowHandleBuffer, New 
HTuple("Mouse Mode: Move Image"))
                        HOperatorSet.DispXld(ho_ModelContours, hv_WindowHandleBuffer)
                        'Visualize the trackball if desired
                        If New HTuple(hv_VisualizeTrackball.TupleEqual(New 
HTuple("always"))).I() Then
                            HOperatorSet.SetColor(hv_WindowHandleBuffer, New 
HTuple("dim gray"))
                            HOperatorSet.DispXld(ho_TrackballContour, 
hv_WindowHandleBuffer)
                        End If
                        HOperatorSet.CopyRectangle(hv_WindowHandleBuffer, 
hv_WindowHandle, New HTuple(0), _
                            New HTuple(0), hv_Height.TupleSub(New HTuple(1)), 
hv_Width.TupleSub( _
                            New HTuple(1)), New HTuple(0), New HTuple(0))
                        '
                        hv_MRow1 = hv_MRow2.Clone()
                        hv_MCol1 = hv_MCol2.Clone()
                    End If
#If USE_DO_EVENTS Then
        ' Please note: The call of DoEvents() is only a hack to
        ' enable VB to react on events. Please change the code
        ' so that it can handle events in a standard way.
        System.Windows.Forms.Application.DoEvents()
#End If
                Loop
            End If
            ho_BackgroundImageTrans.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_BackgroundImageTrans.Dispose()

            Throw HDevExpDefaultException
        End Try
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Inspect a 3d object model and/or determine the 3d shape 
model parameters
    Public Sub inspect_object_model_3d(ByVal ho_BackgroundImages As HObject, ByVal 
hv_ObjectModel3DID As HTuple, _
```

```
        ByVal hv_CamParam As HTuple, ByRef hv_RefRotX As HTuple, ByRef hv_RefRotY As
HTuple, _
        ByRef hv_RefRotZ As HTuple, ByRef hv_LongitudeMin As HTuple, ByRef
hv_LongitudeMax As HTuple, _
        ByRef hv_LatitudeMin As HTuple, ByRef hv_LatitudeMax As HTuple, ByRef
hv_CamRollMin As HTuple, _
        ByRef hv_CamRollMax As HTuple, ByRef hv_DistMin As HTuple, ByRef hv_DistMax
As HTuple, _
        ByRef hv_MinFaceAngle As HTuple)


        Dim sys As New HSystem()

        ' Stack for temporary objects
        Dim OTemp(10) As HObject

        ' Local iconic variables
        Dim ho_TrackballContour As HObject = Nothing
        Dim ho_ModelContours As HObject = Nothing, ho_MenuRegions As HObject =
Nothing
        Dim ho_BackgroundImage As HObject = Nothing, ho_BackgroundImageTrans As
HObject = Nothing

        Dim ho_BackgroundImages_COPY_INP_TMP As HObject
        ho_BackgroundImages_COPY_INP_TMP = ho_BackgroundImages.CopyObj(1, -1)


        ' Local control variables
        Dim hv_ErrorVar As HTuple
        Dim hv_TrackballSize As HTuple = New HTuple
        Dim hv_VisualizeTrackball As HTuple = New HTuple, hv_VirtualTrackball As
HTuple = New HTuple
        Dim hv_ModelColor As HTuple = New HTuple, hv_ImageScale As HTuple = New
HTuple
        Dim hv_ClipRegion As HTuple = New HTuple, hv_Deg2Rad As HTuple = New HTuple
        Dim hv_Width As HTuple = New HTuple, hv_Height As HTuple = New HTuple
        Dim hv_Cx As HTuple = New HTuple, hv_Cy As HTuple = New HTuple
        Dim hv_WidthPose As HTuple = New HTuple, hv_HeightPose As HTuple = New HTuple
        Dim hv_WidthMenu As HTuple = New HTuple, hv_HeightMenu As HTuple = New HTuple
        Dim hv_WindowHandleMenu As HTuple = New HTuple, hv_WindowHandlePose As HTuple
= New HTuple
        Dim hv_WindowHandle As HTuple = New HTuple, hv_WindowHandleBuffer As HTuple =
New HTuple
        Dim hv_MinImageSize As HTuple = New HTuple, hv_TrackballRadiusPixel As HTuple
= New HTuple
        Dim hv_ReferencePoint As HTuple = New HTuple, hv_RefPose As HTuple = New
HTuple
        Dim hv_RefPoseReset As HTuple = New HTuple, hv_Dist As HTuple = New HTuple
        Dim hv_CamPose As HTuple = New HTuple, hv_FontSize As HTuple = New HTuple
        Dim hv_MenuText As HTuple = New HTuple, hv_RelQuaternion As HTuple = New
HTuple
        Dim hv_NumAdd As HTuple = New HTuple, hv_NumImage As HTuple = New HTuple
        Dim hv_ImageNo As HTuple = New HTuple, hv_HiddenSurfaceRemoval As HTuple =
New HTuple
        Dim hv_MouseMode As HTuple = New HTuple, hv_OffsetRow As HTuple = New HTuple
        Dim hv_OffsetCol As HTuple = New HTuple, hv_HomMat2DIdentity As HTuple = New
HTuple
        Dim hv_RefHomMat As HTuple = New HTuple, hv_RelHomMat As HTuple = New HTuple
        Dim hv_AbsHomMat As HTuple = New HTuple, hv_AbsPose As HTuple = New HTuple
        Dim hv_HomMat2DTranslate As HTuple = New HTuple, hv_MenuEvent As HTuple = New
HTuple
        Dim hv_GraphEvent As HTuple = New HTuple, hv_GraphButtonRow As HTuple = New
HTuple
        Dim hv_GraphButtonColumn As HTuple = New HTuple, hv_GraphButton As HTuple =
New HTuple
```

```
        Dim hv_ErrorGraph As HTuple = New HTuple, hv_MenuButtonRow As HTuple = New
HTuple
        Dim hv_MenuButtonColumn As HTuple = New HTuple, hv_MenuButton As HTuple = New
HTuple
        Dim hv_ErrorMenu As HTuple = New HTuple, hv_SelectedButton As HTuple = New
HTuple
        Dim hv_Exit As HTuple = New HTuple

        Dim hv_CamParam_COPY_INP_TMP As HTuple
        hv_CamParam_COPY_INP_TMP = hv_CamParam.Clone()
        hv_WindowHandle = hv_ExpDefaultWinHandle
        hv_WindowHandleMenu = hv_ExpDefaultWinMenuHandle
        hv_WindowHandlePose = hv_ExpDefaultWinViewHandle
        hv_WindowHandleBuffer = hv_ExpDefaultWinBufferHandle

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_TrackballContour)
        HOperatorSet.GenEmptyObj(ho_ModelContours)
        HOperatorSet.GenEmptyObj(ho_MenuRegions)
        HOperatorSet.GenEmptyObj(ho_BackgroundImage)
        HOperatorSet.GenEmptyObj(ho_BackgroundImageTrans)

        Try
            'The procedure inspect_object_mode_3d can be used to visualize a 3d
object
            'model. If the 3d object model is used to create a 3d shape mode,
            'inspect_object_model_3d can also be used to find the right values for
the
            'input control parameters of create_shape_model_3d. The usage is briefly
            'described in the following:
            '
            'Three windows are opened to interact with the user. The upper left
window
            'shows the current camera view of the 3d object model. If desired, one of
            'several background images can be additionally visualized. In this case,
the
            'background images must be passed in the input object parameter
            'BackgroundImages as a tuple. If no background images are available an
empty
            'tuple must be passed instead. The number of the currently displayed
            'background image is displayed in the upper left image corner.
            'If the current mouse mode, which is displayed in the upper right image
corner, is
            'set to 'Move Camera' the mouse can be used to change the current camera
            'view. The position of the camera on a virtual sphere around the 3d
object model
            'can be changed by moving the mouse while holding the left mouse
            'pressed. For this, a virtual trackball (displayed as a gray circle)
indicates around
            'which axis the camera should be rotated. If the mouse is inside the
circle the
            ' camera is rotated around the x and y axis, while otherwise the rotation
is
            'performed around the z axis. The distance between the 3d object model
and the
            'camera can be changed by moving the mouse in vertical direction while
holding
            'the right  button pressed.
            'If the current mouse mode is set to 'Move Image', the background image
can be
            'moved by using the left mouse button. This is useful, for example, to
make an
            'image of the 3d object model coincide with the current camera view. Note
that
```

```
            'by moving the image an artificial perspective is created that cannot
occur in a
            'real image.
            '
            'The upper right window visualizes the position of the camera (yellow
coordinate
            'system) on the virtual (blue) sphere around the 3d object model.
Additionally,
            'the 3d object model is shown viewed from the current camera reference
            'The camera reference pose in general determines the mean pose under
which
            'the 3d object model is seen in the images, in which the 3d object model
should
            'be found. The range of poses under which the 3d object model should be
found
            'is specified by using spherical coordinates longitude and latitude. If
latitude and
            'longitude both are 0, the current viewing direction coincides with the
viewing
            'direction of the reference pose.
            'In order to improve the interpretation of the spherical coordinates,
additionally
            'the meridians and the circles of latitude (both with a step width of
30°) are
            'visualized together with the sphere.
            'After moving the camera on the sphere (as described above) the yellow
camera
            'coordinate system is updated accordingly. Additionally, the reference
pose
            'angles and the spherical coordinates of the current view are displayed
in
            'numbers. Note that the order of the reference pose angles is 'gba'.
Furthermore,
            ' the current camera roll angle (rotation angle around the z axis of the
camera
            'with respect to the reference pose), the distance between the center of
the
            '3d object model and the camera, as well as the minimum face angle is
displayed.
            'In addition to the values of the current camera pose (longitude,
latitude, camera
            'roll angle, and distance) also the respective intervals of the selected
pose range
            'are displayed. The longitude and latitude range is also displayed in
form of a
            'magenta region on the blue sphere.
            '
            'The lower window provides several menu buttons.
            '
            '- 'Set Reference Pose' sets the reference pose to the current pose
            '- 'Add to Pose Range' expands the pose range to include the current pose
            '- 'Next Image' displays the next background image in the tuple
            '- 'Previous Image' displays the previous background image in the tuple
            '- 'Increase MinFaceAngle' increases the minimum face angle by 1 degree
            '- 'Decrease MinFaceAngle' decreases the minimum face angle by 1 degree
            '- 'Mouse Mode: Move Camera' switch to the mode which allows to move the
            '  camera
            '- 'Mouse Mode: Move Image' switch to the mode which allows to move the
            '   background image
            '- 'Reset Reference Pose' sets the reference pose angles to 0 and resets
the
            '  pose range
            '- 'Reset Pose Range' deflates the pose range to the reference pose
            '- 'Hidden Line Removal' switches the hidden line removal on or off
            '- 'Exit' exits the inspect_3d_model procedure and returns the current
values
            '   for the reference pose angles, the pose range, and the minimum face
angle
            '   These values can directly be passed to create_shape_model_3d. Note
that
            '   for most applications it is useful to manually extend the interval of
the
            '   camera roll angle to the full circle [0,360°].
            '
            '
            'User Defines
            '
            'TrackballSize defines the diameter of the trackball in the image with
respect
            'to the smaller image dimension.
            hv_TrackballSize = New HTuple(0.8)
            '
            'VisualizeTrackball determines whether the Trackball shall be displayed
in
            'displayed in the image. Possible values are:
            '  - 'always': always displays the trackball
            '  - 'never': never displays the trackball
            '  - 'on_button': displays the trackball while the mouse button is
pressed
            hv_VisualizeTrackball = New HTuple("always")
            'VisualizeTrackball := 'never'
            'VisualizeTrackball := 'on_button'
            '
            'VirtualTrackball defines the type of virtual trackball that shall
            'shall be used ('shoemake' or 'bell').
            hv_VirtualTrackball = New HTuple("shoemake")
            'VirtualTrackball := 'bell'
            '
            'ModelColor defines the color of the visualized model
            hv_ModelColor = New HTuple("green")
            '
            'ImageScale defines the scaling of the displayed image. This is
especially useful
            'if the original camera image is too large to fit onto the screen. If
ImageScale is
            'set to 1, the image is displayed using its original size. If ImageScale
is set to
            'a value smaller than 1, the displayed image will become smaller by this
factor.
            hv_ImageScale = New HTuple(0.25)
            '
            '
            '
            ' dev_set_preferences(...); only in hdevelop
            HOperatorSet.GetSystem(New HTuple("clip_region"), hv_ClipRegion)
            HOperatorSet.SetSystem(New HTuple("clip_region"), New HTuple("false"))
            ' dev_update_pc(...); only in hdevelop
            ' dev_update_window(...); only in hdevelop
            ' dev_update_var(...); only in hdevelop
            ' dev_update_time(...); only in hdevelop
            hv_Deg2Rad = New HTuple(0.017453292)
            '
            'Apply the user defined image scaling by adapting the camera parameters
            If IsNothing(hv_CamParam_COPY_INP_TMP) Then
                hv_CamParam_COPY_INP_TMP = New HTuple
            End If
            hv_CamParam_COPY_INP_TMP(New HTuple(2)) =
((hv_CamParam_COPY_INP_TMP.TupleSelect( _
                New HTuple(2)))).TupleDiv(hv_ImageScale)
            If IsNothing(hv_CamParam_COPY_INP_TMP) Then
                hv_CamParam_COPY_INP_TMP = New HTuple
            End If
            hv_CamParam_COPY_INP_TMP(New HTuple(3)) =
((hv_CamParam_COPY_INP_TMP.TupleSelect( _
                New HTuple(3)))).TupleDiv(hv_ImageScale)
            If IsNothing(hv_CamParam_COPY_INP_TMP) Then
                hv_CamParam_COPY_INP_TMP = New HTuple
            End If
            hv_CamParam_COPY_INP_TMP(New HTuple(4)) =
((hv_CamParam_COPY_INP_TMP.TupleSelect( _
                New HTuple(4)))).TupleMult(hv_ImageScale)
            If IsNothing(hv_CamParam_COPY_INP_TMP) Then
                hv_CamParam_COPY_INP_TMP = New HTuple
            End If
            hv_CamParam_COPY_INP_TMP(New HTuple(5)) =
((hv_CamParam_COPY_INP_TMP.TupleSelect( _
                New HTuple(5)))).TupleMult(hv_ImageScale)
            If IsNothing(hv_CamParam_COPY_INP_TMP) Then
                hv_CamParam_COPY_INP_TMP = New HTuple
            End If
            hv_CamParam_COPY_INP_TMP(New HTuple(6)) =
((((hv_CamParam_COPY_INP_TMP.TupleSelect( _
                New HTuple(6)))).TupleMult(hv_ImageScale))).TupleInt()
            If IsNothing(hv_CamParam_COPY_INP_TMP) Then
                hv_CamParam_COPY_INP_TMP = New HTuple
            End If
            hv_CamParam_COPY_INP_TMP(New HTuple(7)) =
((((hv_CamParam_COPY_INP_TMP.TupleSelect( _
                New HTuple(7)))).TupleMult(hv_ImageScale))).TupleInt()
            '
            'Scale the background images accordingly
            HOperatorSet.ZoomImageFactor(ho_BackgroundImages_COPY_INP_TMP, OTemp(0),
hv_ImageScale, _
                hv_ImageScale, New HTuple("weighted"))
            ho_BackgroundImages_COPY_INP_TMP.Dispose()
            ho_BackgroundImages_COPY_INP_TMP = OTemp(0)
            '
            hv_Width = hv_CamParam_COPY_INP_TMP.TupleSelect(New HTuple(6))
            hv_Height = hv_CamParam_COPY_INP_TMP.TupleSelect(New HTuple(7))
            hv_Cx = hv_CamParam_COPY_INP_TMP.TupleSelect(New HTuple(4))
            hv_Cy = hv_CamParam_COPY_INP_TMP.TupleSelect(New HTuple(5))
            '
            hv_WidthPose = New HTuple(300)
            hv_HeightPose = ((hv_Height.TupleConcat(New HTuple(480)))).TupleMax()
            '
            'Open a window containing the menu structure
            hv_WidthMenu = ((((((hv_Width.TupleAdd(New
HTuple(7)))).TupleAdd(hv_WidthPose))).TupleConcat( _
                New HTuple(840)))).TupleMax()
            hv_HeightMenu = New HTuple(100)
            'dev_open_window(...);
            HOperatorSet.SetPart(hv_WindowHandleMenu, New HTuple(0), New HTuple(0),
hv_HeightMenu.TupleSub( _
                New HTuple(1)), hv_WidthMenu.TupleSub(New HTuple(1)))
            '
            'Open a window containing the pose information
            'dev_open_window(...);
            HOperatorSet.SetPart(hv_WindowHandlePose, New HTuple(0), New HTuple(0),
hv_HeightPose.TupleSub( _
                New HTuple(1)), hv_WidthPose.TupleSub(New HTuple(1)))
            HOperatorSet.SetLineWidth(hv_WindowHandlePose, New HTuple(1))
            '
            'Open one buffer window and one visible window to avoid flickering
            'dev_open_window(...);
            HOperatorSet.SetPart(hv_WindowHandle, New HTuple(0), New HTuple(0),
hv_Height.TupleSub( _
                New HTuple(1)), hv_Width.TupleSub(New HTuple(1)))
            HOperatorSet.SetLineWidth(hv_WindowHandle, New HTuple(1))
            'open_window(...);
            HOperatorSet.SetPart(hv_WindowHandleBuffer, New HTuple(0), New HTuple(0),
hv_Height.TupleSub( _
                New HTuple(1)), hv_Width.TupleSub(New HTuple(1)))
            HOperatorSet.SetLineWidth(hv_WindowHandleBuffer, New HTuple(1))
            '
            'Compute the trackball
            hv_MinImageSize = ((hv_Width.TupleConcat(hv_Height))).TupleMin()
            hv_TrackballRadiusPixel =
((hv_TrackballSize.TupleMult(hv_MinImageSize))).TupleDiv( _
                New HTuple(2.0))
            ho_TrackballContour.Dispose()
            HOperatorSet.GenEllipseContourXld(ho_TrackballContour, hv_Cy, hv_Cx, New
HTuple(0), _
                hv_TrackballRadiusPixel, hv_TrackballRadiusPixel, New HTuple(0), New
HTuple(6.28318), _
                New HTuple("positive"), New HTuple(1.5))
            '
            'Set the initial model reference pose. The orientation is parallel to the
model
            'coordinate system, the position is at the center of gravity of the
model.
            HOperatorSet.GetObjectModel3dParams(hv_ObjectModel3DID, New
HTuple("reference_point"), _
                hv_ReferencePoint)
            HOperatorSet.CreatePose(((hv_ReferencePoint.TupleSelect(New
HTuple(0)))).TupleNeg( _
                ), ((hv_ReferencePoint.TupleSelect(New HTuple(1)))).TupleNeg(),
((hv_ReferencePoint.TupleSelect( _
                New HTuple(2)))).TupleNeg(), New HTuple(0), New HTuple(0), New
HTuple(0), _
                New HTuple("Rp+T"), New HTuple("gba"), New HTuple("point"),
hv_RefPose)
            '
            'Remember the original reference pose in order to be able to perform a
reset
            hv_RefPoseReset = hv_RefPose.Clone()
            '
            'Compute the optimum distance of the camera for visualization
            determine_optimum_viewing_distance(hv_ObjectModel3DID,
hv_CamParam_COPY_INP_TMP, _
                hv_MinImageSize, New HTuple(0.8), hv_Dist)
            hv_CamPose = hv_RefPose
            If IsNothing(hv_CamPose) Then
                hv_CamPose = New HTuple
            End If
            hv_CamPose(New HTuple(2)) = ((hv_RefPose.TupleSelect(New
HTuple(2)))).TupleAdd( _
                hv_Dist)
            ho_ModelContours.Dispose()
            HOperatorSet.ProjectObjectModel3d(ho_ModelContours, hv_ObjectModel3DID,
hv_CamParam_COPY_INP_TMP, _
                hv_CamPose, New HTuple("true"), (New HTuple(10)).TupleRad())
            '
            hv_FontSize = New HTuple(12)
            set_display_font(hv_WindowHandle, hv_FontSize, New HTuple("mono"), New
HTuple("true"), _
                New HTuple("false"))
            set_display_font(hv_WindowHandleBuffer, hv_FontSize, New HTuple("mono"),
New HTuple("true"), _
                New HTuple("false"))
            set_display_font(hv_WindowHandlePose, hv_FontSize, New HTuple("mono"),
New HTuple("true"), _
                New HTuple("false"))
            set_display_font(hv_WindowHandleMenu, hv_FontSize, New HTuple("mono"),
New HTuple("true"), _
                New HTuple("false"))
            '
            hv_MenuText = ((((((((((New HTuple(New HTuple("Set Reference
Pose"))).TupleConcat( _
                New HTuple("Add to Pose Range"))).TupleConcat(New HTuple("Next
Image"))).TupleConcat( _
                New HTuple("Previous Image"))).TupleConcat(New HTuple("Increase
MinFaceAngle"))).TupleConcat( _
                New HTuple("Decrease MinFaceAngle"))).TupleConcat(New HTuple("Mouse
Mode: Move Camera"))).TupleConcat( _
                New HTuple("Mouse Mode: Move Image"))).TupleConcat(New HTuple("Reset
Reference Pose"))).TupleConcat( _
                New HTuple("Reset Pose Range"))).TupleConcat(New HTuple("Hidden Line
Removal"))).TupleConcat( _
                New HTuple("Exit"))
            ho_MenuRegions.Dispose()
            gen_menu_regions(ho_MenuRegions, hv_WindowHandleMenu, New HTuple(3), New
HTuple(4), _
                New HTuple(100), New HTuple("top"))
            disp_menu(ho_MenuRegions, hv_WindowHandleMenu, hv_MenuText)
            '
            'Initialize some values
            ' Error variable 'hv_ErrorVar' activated
            hv_ErrorVar = New HTuple(2)
            hv_RelQuaternion = (((New
HTuple(1)).TupleConcat(0)).TupleConcat(0)).TupleConcat( _
                0)
            hv_NumAdd = New HTuple(0)
            HOperatorSet.CountObj(ho_BackgroundImages_COPY_INP_TMP, hv_NumImage)
            hv_ImageNo = New HTuple(0)
            hv_HiddenSurfaceRemoval = New HTuple("true")
            hv_MouseMode = New HTuple("move_camera")
            hv_OffsetRow = HTuple.TupleGenConst(hv_NumImage, New HTuple(0))
            hv_OffsetCol = HTuple.TupleGenConst(hv_NumImage, New HTuple(0))
            '
            'Initialize the return values
            hv_LongitudeMin = New HTuple(0)
            hv_LongitudeMax = New HTuple(0)
            hv_LatitudeMin = New HTuple(0)
            hv_LatitudeMax = New HTuple(0)
            hv_CamRollMin = New HTuple(0)
            hv_CamRollMax = New HTuple(0)
            hv_DistMin = hv_Dist.Clone()
            hv_DistMax = hv_Dist.Clone()
            hv_RefRotX = New HTuple(0)
            hv_RefRotY = New HTuple(0)
            hv_RefRotZ = New HTuple(0)
            hv_MinFaceAngle = (New HTuple(30)).TupleRad()
            '
            'Start the visualization loop
            HOperatorSet.HomMat2dIdentity(hv_HomMat2DIdentity)
            Do While (New HTuple(1)).I()
                HOperatorSet.PoseToHomMat3d(hv_RefPose, hv_RefHomMat)
                HOperatorSet.QuatToHomMat3d(hv_RelQuaternion, hv_RelHomMat)
                HOperatorSet.HomMat3dCompose(hv_RelHomMat, hv_RefHomMat,
hv_AbsHomMat)
                HOperatorSet.HomMat3dToPose(hv_AbsHomMat, hv_AbsPose)
                If IsNothing(hv_AbsPose) Then
                    hv_AbsPose = New HTuple
                End If
                hv_AbsPose(New HTuple(2)) = ((hv_AbsPose.TupleSelect(New
HTuple(2)))).TupleAdd( _
                    hv_Dist)
                ho_ModelContours.Dispose()
                HOperatorSet.ProjectObjectModel3d(ho_ModelContours,
hv_ObjectModel3DID, hv_CamParam_COPY_INP_TMP, _
                    hv_AbsPose, hv_HiddenSurfaceRemoval, hv_MinFaceAngle)
                '
                'Clear the window and display the background image if desired
                If New HTuple(hv_ImageNo.TupleGreater(hv_NumImage)).I() Then
                    hv_ImageNo = New HTuple(0)
                End If
                If New HTuple(hv_ImageNo.TupleLess(New HTuple(0))).I() Then
                    hv_ImageNo = hv_NumImage.Clone()
                End If
                HOperatorSet.ClearWindow(hv_WindowHandleBuffer)
                HOperatorSet.SetColor(hv_WindowHandleBuffer, hv_ModelColor)
                If New HTuple(hv_ImageNo.TupleGreater(New HTuple(0))).I() Then
                    ho_BackgroundImage.Dispose()
                    HOperatorSet.SelectObj(ho_BackgroundImages_COPY_INP_TMP,
ho_BackgroundImage, _
                        hv_ImageNo)
                    HOperatorSet.HomMat2dTranslate(hv_HomMat2DIdentity,
hv_OffsetRow.TupleSelect( _
                        hv_ImageNo.TupleSub(New HTuple(1))),
hv_OffsetCol.TupleSelect(hv_ImageNo.TupleSub( _
                        New HTuple(1))), hv_HomMat2DTranslate)
                    ho_BackgroundImageTrans.Dispose()
                    HOperatorSet.AffineTransImage(ho_BackgroundImage,
ho_BackgroundImageTrans, _
                        hv_HomMat2DTranslate, New HTuple("constant"), New
HTuple("false"))
                    HOperatorSet.DispImage(ho_BackgroundImageTrans,
hv_ExpDefaultWinHandle)
                    HOperatorSet.SetTposition(hv_WindowHandleBuffer, New HTuple(10),
New HTuple(10))
                    HOperatorSet.WriteString(hv_WindowHandleBuffer, (New
HTuple("Image No.: ")).TupleAdd( _
                        hv_ImageNo.TupleString((New HTuple("2d")).S())))
                Else
                    ho_BackgroundImage.Dispose()
                    HOperatorSet.GenEmptyObj(ho_BackgroundImage)
                    HOperatorSet.SetTposition(hv_WindowHandleBuffer, New HTuple(10),
New HTuple(10))
                    HOperatorSet.WriteString(hv_WindowHandleBuffer, New HTuple("(No
Background Image)"))
                End If
                HOperatorSet.SetTposition(hv_WindowHandleBuffer, New HTuple(10),
hv_Width.TupleSub( _
                    New HTuple(180)))
                If New HTuple(hv_MouseMode.TupleEqual(New HTuple("move_camera"))).I()
Then
                    HOperatorSet.WriteString(hv_WindowHandleBuffer, New HTuple("Mouse
Mode: Move Camera"))
                Else
                    HOperatorSet.WriteString(hv_WindowHandleBuffer, New HTuple("Mouse
Mode: Move Image"))
                End If
                HOperatorSet.DispXld(ho_ModelContours, hv_WindowHandleBuffer)
                '
                'Visualize the trackball if desired
                If New HTuple(hv_VisualizeTrackball.TupleEqual(New
HTuple("always"))).I() Then
                    HOperatorSet.SetColor(hv_WindowHandleBuffer, New HTuple("dim
gray"))
```

```
                    HOperatorSet.DispXld(ho_TrackballContour, hv_WindowHandleBuffer)
                End If
                HOperatorSet.CopyRectangle(hv_WindowHandleBuffer, hv_WindowHandle, 
New HTuple(0), _
                    New HTuple(0), hv_Height.TupleSub(New HTuple(1)), 
hv_Width.TupleSub(New HTuple(1)), _
                    New HTuple(0), New HTuple(0))
                '
                'Update the parameter ranges
                update_pose_information(hv_WindowHandlePose, hv_ObjectModel3DID, 
hv_CamParam_COPY_INP_TMP, _
                    hv_RefPose, hv_RelQuaternion, hv_Dist, hv_LatitudeMin, 
hv_LatitudeMax, _
                    hv_LongitudeMin, hv_LongitudeMax, hv_CamRollMin, hv_CamRollMax, 
hv_DistMin, _
                    hv_DistMax, hv_MinFaceAngle, hv_HiddenSurfaceRemoval, 
hv_ModelColor)
                '
                'Avoid multiple responses for a single mouse click
                HOperatorSet.WaitSeconds(New HTuple(0.1))
                hv_MenuEvent = New HTuple(0)
                hv_GraphEvent = New HTuple(0)
                Do While (New HTuple(1)).I()
                    ' (dev_)set_check ("~give_error")
                    Try
                        hv_ErrorVar = 2
                        HOperatorSet.GetMposition(hv_WindowHandle, hv_GraphButtonRow, 
hv_GraphButtonColumn, _
                            hv_GraphButton)
                    Catch e As HalconException
                        hv_ErrorVar = New HTuple(e.GetErrorNumber())
                        If hv_ErrorVar.I() < 0 Then
                            Throw e
                        End If
                    End Try
                    hv_ErrorGraph = hv_ErrorVar.Clone()
                    Try
                        hv_ErrorVar = 2
                        HOperatorSet.GetMposition(hv_WindowHandleMenu, 
hv_MenuButtonRow, hv_MenuButtonColumn, _
                            hv_MenuButton)
                    Catch e As HalconException
                        hv_ErrorVar = New HTuple(e.GetErrorNumber())
                        If hv_ErrorVar.I() < 0 Then
                            Throw e
                        End If
                    End Try
                    hv_ErrorMenu = hv_ErrorVar.Clone()
                    ' dev_update_pc(...); only in hdevelop
                    ' (dev_)set_check ("~give_error")
                    If ((New HTuple(hv_ErrorGraph.TupleEqual(New 
HTuple(2))))).TupleAnd(New HTuple( _
                        hv_GraphButton.TupleNotEqual(New HTuple(0)))).I() Then
                        hv_GraphEvent = New HTuple(1)
                        Exit Do
                    End If
                    If ((New HTuple(hv_ErrorMenu.TupleEqual(New 
HTuple(2))))).TupleAnd(New HTuple( _
                        hv_MenuButton.TupleNotEqual(New HTuple(0)))).I() Then
                        hv_MenuEvent = New HTuple(1)
                        Exit Do
                    End If
#If USE_DO_EVENTS Then
        ' Please note: The call of DoEvents() is only a hack to
        ' enable VB to react on events. Please change the code
        ' so that it can handle events in a standard way.
        System.Windows.Forms.Application.DoEvents()
#End If
                Loop
                If hv_GraphEvent.I() Then
                    If New HTuple(hv_MouseMode.TupleEqual(New
HTuple("move_camera"))).I() Then
                        'Reflect the movement of the camera
                        analyze_graph_event_camera(ho_BackgroundImage,
ho_TrackballContour, hv_GraphButton, _
                            hv_GraphButtonRow, hv_GraphButtonColumn, hv_WindowHandle,
hv_WindowHandleBuffer, _
                            hv_VirtualTrackball, hv_TrackballSize,
hv_VisualizeTrackball, hv_ObjectModel3DID, _
                            hv_CamParam_COPY_INP_TMP, hv_HiddenSurfaceRemoval,
hv_ImageNo, hv_ModelColor, _
                            hv_RefPose, hv_MinFaceAngle, hv_OffsetRow, hv_OffsetCol,
hv_RelQuaternion, _
                            hv_NumAdd, hv_Dist, hv_RelQuaternion, hv_NumAdd, hv_Dist)
                    Else
                        'Reflect the movement of the background image
                        analyze_graph_event_image(ho_BackgroundImage,
ho_ModelContours, ho_TrackballContour, _
                            hv_GraphButton, hv_GraphButtonRow, hv_GraphButtonColumn,
hv_WindowHandle, _
                            hv_WindowHandleBuffer, hv_OffsetRow, hv_OffsetCol,
hv_ImageNo, hv_VisualizeTrackball, _
                            hv_ModelColor, hv_OffsetRow, hv_OffsetCol)
                    End If
                End If
                If hv_MenuEvent.I() Then
                    'Reflect the selection of the menu button
                    analyze_menu_event(ho_MenuRegions, hv_MenuText,
hv_WindowHandleMenu, hv_MenuButtonRow, _
                        hv_MenuButtonColumn, hv_LongitudeMin, hv_LongitudeMax,
hv_LatitudeMin, _
                        hv_LatitudeMax, hv_CamRollMin, hv_CamRollMax, hv_DistMin,
hv_DistMax, _
                        hv_MinFaceAngle, hv_RefPose, hv_RelQuaternion, hv_ImageNo,
hv_HiddenSurfaceRemoval, _
                        hv_MouseMode, hv_Dist, hv_ObjectModel3DID, hv_RefPoseReset,
hv_SelectedButton, _
                        hv_Exit, hv_LongitudeMin, hv_LongitudeMax, hv_LatitudeMin,
hv_LatitudeMax, _
                        hv_CamRollMin, hv_CamRollMax, hv_DistMin, hv_DistMax,
hv_MinFaceAngle, _
                        hv_RefPose, hv_RelQuaternion, hv_ImageNo,
hv_HiddenSurfaceRemoval, hv_MouseMode)
                    If hv_Exit.I() Then
                        Exit Do
                    End If
                End If
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Loop
            '
            'Extract the reference pose angles from the reference pose
            hv_RefRotX = ((hv_RefPose.TupleSelect(New
HTuple(3)))).TupleMult(hv_Deg2Rad)
            hv_RefRotY = ((hv_RefPose.TupleSelect(New
HTuple(4)))).TupleMult(hv_Deg2Rad)
```

```
            hv_RefRotZ = ((hv_RefPose.TupleSelect(New
HTuple(5)))).TupleMult(hv_Deg2Rad)
            '
            'Clean up
            Try
                hv_ErrorVar = 2
                HOperatorSet.SetSystem(New HTuple("clip_region"), hv_ClipRegion)
            Catch e As HalconException
                hv_ErrorVar = New HTuple(e.GetErrorNumber())
                If hv_ErrorVar.I() < 0 Then
                    Throw e
                End If
            End Try
            Try
                hv_ErrorVar = 2
                'close_window(...);
            Catch e As HalconException
                hv_ErrorVar = New HTuple(e.GetErrorNumber())
                If hv_ErrorVar.I() < 0 Then
                    Throw e
                End If
            End Try
            ' dev_set_preferences(...); only in hdevelop
            'dev_set_window(...);
            'dev_close_window(...);
            'dev_set_window(...);
            'dev_close_window(...);
            'dev_set_window(...);
            'dev_close_window(...);
            ho_BackgroundImages_COPY_INP_TMP.Dispose()
            ho_TrackballContour.Dispose()
            ho_ModelContours.Dispose()
            ho_MenuRegions.Dispose()
            ho_BackgroundImage.Dispose()
            ho_BackgroundImageTrans.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_BackgroundImages_COPY_INP_TMP.Dispose()
            ho_TrackballContour.Dispose()
            ho_ModelContours.Dispose()
            ho_MenuRegions.Dispose()
            ho_BackgroundImage.Dispose()
            ho_BackgroundImageTrans.Dispose()

            Throw HDevExpDefaultException
        End Try
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Project an image point onto the trackball
    Public Sub project_point_on_trackball(ByVal hv_X As HTuple, ByVal hv_Y As HTuple,
_
        ByVal hv_VirtualTrackball As HTuple, ByVal hv_TrackballSize As HTuple, ByRef
hv_V As HTuple)


        ' Local control variables
        Dim hv_R As HTuple = New HTuple, hv_XP As HTuple = New HTuple
        Dim hv_YP As HTuple = New HTuple, hv_ZP As HTuple = New HTuple

        ' Initialize local and output iconic variables

        If New HTuple(hv_VirtualTrackball.TupleEqual(New HTuple("shoemake"))).I()
Then
```

```
            'Virtual Trackball according to Shoemake
            hv_R = 
((((hv_X.TupleMult(hv_X))).TupleAdd(hv_Y.TupleMult(hv_Y)))).TupleSqrt( _
                )
            If New HTuple(hv_R.TupleLessEqual(hv_TrackballSize)).I() Then
                hv_XP = hv_X.Clone()
                hv_YP = hv_Y.Clone()
                hv_ZP = 
((((hv_TrackballSize.TupleMult(hv_TrackballSize))).TupleSub(hv_R.TupleMult( _
                    hv_R)))).TupleSqrt()
            Else
                hv_XP = ((hv_X.TupleMult(hv_TrackballSize))).TupleDiv(hv_R)
                hv_YP = ((hv_Y.TupleMult(hv_TrackballSize))).TupleDiv(hv_R)
                hv_ZP = New HTuple(0)
            End If
        Else
            'Virtual Trackball according to Bell
            hv_R = 
((((hv_X.TupleMult(hv_X))).TupleAdd(hv_Y.TupleMult(hv_Y)))).TupleSqrt( _
                )
            If New HTuple(hv_R.TupleLessEqual(hv_TrackballSize.TupleMult(New 
HTuple(0.70710678)))).I() Then
                hv_XP = hv_X.Clone()
                hv_YP = hv_Y.Clone()
                hv_ZP = 
((((hv_TrackballSize.TupleMult(hv_TrackballSize))).TupleSub(hv_R.TupleMult( _
                    hv_R)))).TupleSqrt()
            Else
                hv_XP = hv_X.Clone()
                hv_YP = hv_Y.Clone()
                hv_ZP = (((((New 
HTuple(0.5)).TupleMult(hv_TrackballSize))).TupleMult(hv_TrackballSize))).TupleDiv( _
                    hv_R)
            End If
        End If
        hv_V = ((hv_XP.TupleConcat(hv_YP))).TupleConcat(hv_ZP)

        Exit Sub
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Reflect the movement of the camera
    Public Sub analyze_graph_event_camera(ByVal ho_BackgroundImage As HObject, ByVal 
ho_TrackballContour As HObject, _
        ByVal hv_Button As HTuple, ByVal hv_Row As HTuple, ByVal hv_Column As HTuple, 
_
        ByVal hv_WindowHandle As HTuple, ByVal hv_WindowHandleBuffer As HTuple, ByVal 
hv_VirtualTrackball As HTuple, _
        ByVal hv_TrackballSize As HTuple, ByVal hv_VisualizeTrackball As HTuple, 
ByVal hv_ObjectModel3DID As HTuple, _
        ByVal hv_CamParam As HTuple, ByVal hv_HiddenSurfaceRemoval As HTuple, ByVal 
hv_ImageNo As HTuple, _
        ByVal hv_ModelColor As HTuple, ByVal hv_RefPose As HTuple, ByVal 
hv_MinFaceAngle As HTuple, _
        ByVal hv_OffsetRow As HTuple, ByVal hv_OffsetCol As HTuple, ByVal 
hv_RelQuaternionIn As HTuple, _
        ByVal hv_NumAddIn As HTuple, ByVal hv_DistIn As HTuple, ByRef 
hv_RelQuaternionOut As HTuple, _
        ByRef hv_NumAddOut As HTuple, ByRef hv_DistOut As HTuple)


        Dim sys As New HSystem()

        ' Local iconic variables
        Dim ho_ModelContours As HObject = Nothing
        Dim ho_BackgroundImageTrans As HObject = Nothing


        ' Local control variables
        Dim hv_ErrorVar As HTuple
        Dim hv_Cx As HTuple = New HTuple, hv_Cy As HTuple = New HTuple
        Dim hv_Width As HTuple = New HTuple, hv_Height As HTuple = New HTuple
        Dim hv_MinImageSize As HTuple = New HTuple, hv_HomMat2DIdentity As HTuple =
New HTuple
        Dim hv_MRow1 As HTuple = New HTuple, hv_MCol1 As HTuple = New HTuple
        Dim hv_Error As HTuple = New HTuple, hv_MRow2 As HTuple = New HTuple
        Dim hv_MCol2 As HTuple = New HTuple, hv_MX1 As HTuple = New HTuple
        Dim hv_MY1 As HTuple = New HTuple, hv_MX2 As HTuple = New HTuple
        Dim hv_MY2 As HTuple = New HTuple, hv_DeltaQuatRotation As HTuple = New
HTuple
        Dim hv_RelHomMat As HTuple = New HTuple, hv_RefHomMat As HTuple = New HTuple
        Dim hv_AbsHomMat As HTuple = New HTuple, hv_AbsPose As HTuple = New HTuple
        Dim hv_HomMat2DTranslate As HTuple = New HTuple, hv_DRow As HTuple = New
HTuple

        Dim hv_Button_COPY_INP_TMP As HTuple
        hv_Button_COPY_INP_TMP = hv_Button.Clone()
        Dim hv_Column_COPY_INP_TMP As HTuple
        hv_Column_COPY_INP_TMP = hv_Column.Clone()
        Dim hv_DistIn_COPY_INP_TMP As HTuple
        hv_DistIn_COPY_INP_TMP = hv_DistIn.Clone()
        Dim hv_RelQuaternionIn_COPY_INP_TMP As HTuple
        hv_RelQuaternionIn_COPY_INP_TMP = hv_RelQuaternionIn.Clone()
        Dim hv_Row_COPY_INP_TMP As HTuple
        hv_Row_COPY_INP_TMP = hv_Row.Clone()

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_ModelContours)
        HOperatorSet.GenEmptyObj(ho_BackgroundImageTrans)

        Try
            hv_Cx = hv_CamParam.TupleSelect(New HTuple(4))
            hv_Cy = hv_CamParam.TupleSelect(New HTuple(5))
            hv_Width = hv_CamParam.TupleSelect(New HTuple(6))
            hv_Height = hv_CamParam.TupleSelect(New HTuple(7))
            hv_MinImageSize = ((hv_Width.TupleConcat(hv_Height))).TupleMin()
            ' Error variable 'hv_ErrorVar' activated
            hv_ErrorVar = New HTuple(2)
            HOperatorSet.HomMat2dIdentity(hv_HomMat2DIdentity)
            If New HTuple(hv_Button_COPY_INP_TMP.TupleEqual(New HTuple(1))).I() Then
                hv_MRow1 = hv_Row_COPY_INP_TMP.Clone()
                hv_MCol1 = hv_Column_COPY_INP_TMP.Clone()
                Do While New HTuple(hv_Button_COPY_INP_TMP.TupleEqual(New
HTuple(1))).I()
                    ' (dev_)set_check ("~give_error")
                    Try
                        hv_ErrorVar = 2
                        HOperatorSet.GetMposition(hv_WindowHandle,
hv_Row_COPY_INP_TMP, _
                            hv_Column_COPY_INP_TMP, hv_Button_COPY_INP_TMP)
                    Catch e As HalconException
                        hv_ErrorVar = New HTuple(e.GetErrorNumber())
                        If hv_ErrorVar.I() < 0 Then
                            Throw e
                        End If
                    End Try
                    hv_Error = hv_ErrorVar.Clone()
                    ' (dev_)set_check ("give_error")
                    If New HTuple(hv_Error.TupleEqual(New HTuple(2))).I() Then
                        hv_MRow2 = hv_Row_COPY_INP_TMP.Clone()
                        hv_MCol2 = hv_Column_COPY_INP_TMP.Clone()
                        'Transform the pixel coordinates to relative image
coordinates
                        hv_MX1 = ((hv_Cx.TupleSub(hv_MCol1))).TupleDiv((New
HTuple(0.5)).TupleMult( _
                            hv_MinImageSize))
                        hv_MY1 = ((hv_Cy.TupleSub(hv_MRow1))).TupleDiv((New
HTuple(0.5)).TupleMult( _
                            hv_MinImageSize))
                        hv_MX2 = ((hv_Cx.TupleSub(hv_MCol2))).TupleDiv((New
HTuple(0.5)).TupleMult( _
                            hv_MinImageSize))
                        hv_MY2 = ((hv_Cy.TupleSub(hv_MRow2))).TupleDiv((New
HTuple(0.5)).TupleMult( _
                            hv_MinImageSize))
                        'Compute the quaternion rotation that corresponds to the
mouse
                        'movement
                        trackball(hv_MX1, hv_MY1, hv_MX2, hv_MY2,
hv_VirtualTrackball, hv_TrackballSize, _
                            hv_DeltaQuatRotation)
                        'Add the incremental quaternion
                        HOperatorSet.QuatCompose(hv_DeltaQuatRotation,
hv_RelQuaternionIn_COPY_INP_TMP, _
                            hv_RelQuaternionIn_COPY_INP_TMP)
                        hv_NumAddOut = hv_NumAddIn.TupleAdd(New HTuple(1))
                        'Normalize the quaternion every 1000 additions
                        If New HTuple(hv_NumAddOut.TupleGreater(New
HTuple(1000))).I() Then

HOperatorSet.QuatNormalize(hv_RelQuaternionIn_COPY_INP_TMP,
hv_RelQuaternionIn_COPY_INP_TMP)
                            hv_NumAddOut = New HTuple(0)
                        End If
                        'Transform the quaternion to a rotation matrix
                        HOperatorSet.QuatToHomMat3d(hv_RelQuaternionIn_COPY_INP_TMP,
hv_RelHomMat)
                        HOperatorSet.PoseToHomMat3d(hv_RefPose, hv_RefHomMat)
                        HOperatorSet.HomMat3dCompose(hv_RelHomMat, hv_RefHomMat,
hv_AbsHomMat)
                        HOperatorSet.HomMat3dToPose(hv_AbsHomMat, hv_AbsPose)
                        If IsNothing(hv_AbsPose) Then
                            hv_AbsPose = New HTuple
                        End If
                        hv_AbsPose(New HTuple(2)) = ((hv_AbsPose.TupleSelect(New
HTuple(2)))).TupleAdd( _
                            hv_DistIn_COPY_INP_TMP)
                        ho_ModelContours.Dispose()
                        HOperatorSet.ProjectObjectModel3d(ho_ModelContours,
hv_ObjectModel3DID, _
                            hv_CamParam, hv_AbsPose, hv_HiddenSurfaceRemoval,
hv_MinFaceAngle)
                        '
                        HOperatorSet.ClearWindow(hv_WindowHandleBuffer)
                        HOperatorSet.SetColor(hv_WindowHandleBuffer, hv_ModelColor)
                        If New HTuple(hv_ImageNo.TupleGreater(New HTuple(0))).I()
Then
                            HOperatorSet.HomMat2dTranslate(hv_HomMat2DIdentity,
hv_OffsetRow.TupleSelect( _
                                hv_ImageNo.TupleSub(New HTuple(1))),
hv_OffsetCol.TupleSelect(hv_ImageNo.TupleSub( _
                                New HTuple(1))), hv_HomMat2DTranslate)
                            ho_BackgroundImageTrans.Dispose()
                            HOperatorSet.AffineTransImage(ho_BackgroundImage,
ho_BackgroundImageTrans, _
```

```
                                hv_HomMat2DTranslate, New HTuple("constant"), New
HTuple("false"))
                            HOperatorSet.DispImage(ho_BackgroundImageTrans,
hv_WindowHandleBuffer)
                            HOperatorSet.SetTposition(hv_WindowHandleBuffer, New
HTuple(10), New HTuple(10))
                            HOperatorSet.WriteString(hv_WindowHandleBuffer, (New
HTuple("Image No.: ")).TupleAdd( _
                                hv_ImageNo.TupleString((New HTuple("2d")).S())))
                        Else
                            HOperatorSet.SetTposition(hv_WindowHandleBuffer, New
HTuple(10), New HTuple(10))
                            HOperatorSet.WriteString(hv_WindowHandleBuffer, New
HTuple("(No Background Image)"))
                        End If
                        HOperatorSet.SetTposition(hv_WindowHandleBuffer, New
HTuple(10), hv_Width.TupleSub( _
                            New HTuple(180)))
                        HOperatorSet.WriteString(hv_WindowHandleBuffer, New
HTuple("Mouse Mode: Move Camera"))
                        HOperatorSet.DispXld(ho_ModelContours, hv_WindowHandleBuffer)
                        If ((New HTuple(hv_VisualizeTrackball.TupleEqual(New
HTuple("always"))))).TupleOr( _
                            New HTuple(hv_VisualizeTrackball.TupleEqual(New
HTuple("on_button")))).I() Then
                            HOperatorSet.SetColor(hv_WindowHandleBuffer, New
HTuple("dim gray"))
                            HOperatorSet.DispXld(ho_TrackballContour,
hv_WindowHandleBuffer)
                        End If
                        HOperatorSet.CopyRectangle(hv_WindowHandleBuffer,
hv_WindowHandle, New HTuple(0), _
                            New HTuple(0), hv_Height.TupleSub(New HTuple(1)),
hv_Width.TupleSub( _
                            New HTuple(1)), New HTuple(0), New HTuple(0))
                        '
                        hv_MRow1 = hv_MRow2.Clone()
                        hv_MCol1 = hv_MCol2.Clone()
                    End If
#If USE_DO_EVENTS Then
        ' Please note: The call of DoEvents() is only a hack to
        ' enable VB to react on events. Please change the code
        ' so that it can handle events in a standard way.
        System.Windows.Forms.Application.DoEvents()
#End If
                Loop
            End If
            If New HTuple(hv_Button_COPY_INP_TMP.TupleEqual(New HTuple(4))).I() Then
                hv_MRow1 = hv_Row_COPY_INP_TMP.Clone()
                Do While New HTuple(hv_Button_COPY_INP_TMP.TupleEqual(New
HTuple(4))).I()
                    ' (dev_)set_check ("~give_error")
                    Try
                        hv_ErrorVar = 2
                        HOperatorSet.GetMposition(hv_WindowHandle,
hv_Row_COPY_INP_TMP, _
                            hv_Column_COPY_INP_TMP, hv_Button_COPY_INP_TMP)
                    Catch e As HalconException
                        hv_ErrorVar = New HTuple(e.GetErrorNumber())
                        If hv_ErrorVar.I() < 0 Then
                            Throw e
                        End If
                    End Try
                    hv_Error = hv_ErrorVar.Clone()
                    ' (dev_)set_check ("give_error")
                    If New HTuple(hv_Error.TupleEqual(New HTuple(2))).I() Then
                        hv_MRow2 = hv_Row_COPY_INP_TMP.Clone()
                        hv_DRow = hv_MRow2.TupleSub(hv_MRow1)
                        hv_DistIn_COPY_INP_TMP =
hv_DistIn_COPY_INP_TMP.TupleSub(((hv_DistIn_COPY_INP_TMP.TupleMult( _
                            hv_DRow))).TupleMult(New HTuple(0.003)))
                        HOperatorSet.QuatToHomMat3d(hv_RelQuaternionIn_COPY_INP_TMP,
hv_RelHomMat)
                        HOperatorSet.PoseToHomMat3d(hv_RefPose, hv_RefHomMat)
                        HOperatorSet.HomMat3dCompose(hv_RelHomMat, hv_RefHomMat,
hv_AbsHomMat)
                        HOperatorSet.HomMat3dToPose(hv_AbsHomMat, hv_AbsPose)
                        If IsNothing(hv_AbsPose) Then
                            hv_AbsPose = New HTuple
                        End If
                        hv_AbsPose(New HTuple(2)) = ((hv_AbsPose.TupleSelect(New
HTuple(2)))).TupleAdd( _
                            hv_DistIn_COPY_INP_TMP)
                        ho_ModelContours.Dispose()
                        HOperatorSet.ProjectObjectModel3d(ho_ModelContours,
hv_ObjectModel3DID, _
                            hv_CamParam, hv_AbsPose, hv_HiddenSurfaceRemoval,
hv_MinFaceAngle)
                        '
                        HOperatorSet.ClearWindow(hv_WindowHandleBuffer)
                        HOperatorSet.SetColor(hv_WindowHandleBuffer, hv_ModelColor)
                        If New HTuple(hv_ImageNo.TupleGreater(New HTuple(0))).I()
Then
                            HOperatorSet.HomMat2dTranslate(hv_HomMat2DIdentity,
hv_OffsetRow.TupleSelect( _
                                hv_ImageNo.TupleSub(New HTuple(1))),
hv_OffsetCol.TupleSelect(hv_ImageNo.TupleSub( _
                                New HTuple(1))), hv_HomMat2DTranslate)
                            ho_BackgroundImageTrans.Dispose()
                            HOperatorSet.AffineTransImage(ho_BackgroundImage,
ho_BackgroundImageTrans, _
                                hv_HomMat2DTranslate, New HTuple("constant"), New
HTuple("false"))
                            HOperatorSet.DispImage(ho_BackgroundImageTrans,
hv_WindowHandleBuffer)
                            HOperatorSet.SetTposition(hv_WindowHandleBuffer, New
HTuple(10), New HTuple(10))
                            HOperatorSet.WriteString(hv_WindowHandleBuffer, (New
HTuple("Image No.: ")).TupleAdd( _
                                hv_ImageNo.TupleString((New HTuple("2d")).S())))
                        Else
                            HOperatorSet.SetTposition(hv_WindowHandleBuffer, New
HTuple(10), New HTuple(10))
                            HOperatorSet.WriteString(hv_WindowHandleBuffer, New
HTuple("(No Background Image)"))
                        End If
                        HOperatorSet.SetTposition(hv_WindowHandleBuffer, New
HTuple(10), hv_Width.TupleSub( _
                            New HTuple(180)))
                        HOperatorSet.WriteString(hv_WindowHandleBuffer, New
HTuple("Mouse Mode: Move Camera"))
                        HOperatorSet.DispXld(ho_ModelContours, hv_WindowHandleBuffer)
                        If ((New HTuple(hv_VisualizeTrackball.TupleEqual(New
HTuple("always"))))).TupleOr( _
                            New HTuple(hv_VisualizeTrackball.TupleEqual(New
HTuple("on_button")))).I() Then
                            HOperatorSet.SetColor(hv_WindowHandleBuffer, New
HTuple("dim gray"))
                            HOperatorSet.DispXld(ho_TrackballContour,
hv_WindowHandleBuffer)
```

```
                        End If
                        HOperatorSet.CopyRectangle(hv_WindowHandleBuffer,
hv_WindowHandle, New HTuple(0), _
                            New HTuple(0), hv_Height.TupleSub(New HTuple(1)),
hv_Width.TupleSub( _
                            New HTuple(1)), New HTuple(0), New HTuple(0))
                        '
                        hv_MRow1 = hv_MRow2.Clone()
                    End If
#If USE_DO_EVENTS Then
        ' Please note: The call of DoEvents() is only a hack to
        ' enable VB to react on events. Please change the code
        ' so that it can handle events in a standard way.
        System.Windows.Forms.Application.DoEvents()
#End If
                Loop
            End If
            hv_DistOut = hv_DistIn_COPY_INP_TMP.Clone()
            hv_RelQuaternionOut = hv_RelQuaternionIn_COPY_INP_TMP.Clone()
            ho_ModelContours.Dispose()
            ho_BackgroundImageTrans.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_ModelContours.Dispose()
            ho_BackgroundImageTrans.Dispose()

            Throw HDevExpDefaultException
        End Try
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Creates a menu button
    Public Sub disp_menu_button(ByVal ho_MenuRegion As HObject, ByVal hv_WindowHandle
As HTuple, _
        ByVal hv_Text As HTuple)



        ' Local iconic variables
        Dim ho_RegionBorder As HObject = Nothing


        ' Local control variables
        Dim hv_ActiveWindowHandle As HTuple = New HTuple
        Dim hv_DrawMode As HTuple = New HTuple, hv_Rows As HTuple = New HTuple
        Dim hv_Cols As HTuple = New HTuple, hv_Ascent As HTuple = New HTuple
        Dim hv_Descent As HTuple = New HTuple, hv_Width As HTuple = New HTuple
        Dim hv_Height As HTuple = New HTuple

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_RegionBorder)

        Try
            'dev_get_window(...);
            HOperatorSet.GetDraw(hv_WindowHandle, hv_DrawMode)
            HOperatorSet.SetColor(hv_WindowHandle, New HTuple("light gray"))
            'dev_set_window(...);
            HOperatorSet.SetDraw(hv_WindowHandle, New HTuple("fill"))
            HOperatorSet.SetColor(hv_WindowHandle, New HTuple("light gray"))
            HOperatorSet.DispRegion(ho_MenuRegion, hv_WindowHandle)
            'dev_display (MenuRegion)
            HOperatorSet.GetRegionContour(ho_MenuRegion, hv_Rows, hv_Cols)
            HOperatorSet.SetColor(hv_WindowHandle, New HTuple("dim gray"))
            'dev_set_color ('dim gray')
            ho_RegionBorder.Dispose()
            HOperatorSet.GenRegionPoints(ho_RegionBorder, hv_Rows, hv_Cols)
            HOperatorSet.DispRegion(ho_RegionBorder, hv_WindowHandle)
            'dev_display (RegionBorder)
            'dev_set_color ('black')
            HOperatorSet.SetColor(hv_WindowHandle, New HTuple("black"))
            HOperatorSet.GetStringExtents(hv_WindowHandle, hv_Text, hv_Ascent,
hv_Descent, _
                hv_Width, hv_Height)
            HOperatorSet.SetTposition(hv_WindowHandle,
((hv_Rows.TupleMin())).TupleAdd( _
                New HTuple(1)),
(((((hv_Cols.TupleMax())).TupleAdd(hv_Cols.TupleMin()))).TupleDiv( _
                New HTuple(2)))).TupleSub(hv_Width.TupleDiv(New HTuple(2))))
            'set_tposition (WindowHandle, min(Rows) + 1, min(Cols) + 1)
            HOperatorSet.WriteString(hv_WindowHandle, hv_Text)
            HOperatorSet.SetDraw(hv_WindowHandle, hv_DrawMode)
            'dev_set_window(...);
            ho_RegionBorder.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_RegionBorder.Dispose()

            Throw HDevExpDefaultException
        End Try
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Compute the 3d rotation from the mose movement
    Public Sub trackball(ByVal hv_MX1 As HTuple, ByVal hv_MY1 As HTuple, ByVal hv_MX2
As HTuple, _
        ByVal hv_MY2 As HTuple, ByVal hv_VirtualTrackball As HTuple, ByVal
hv_TrackballSize As HTuple, _
        ByRef hv_QuatRotation As HTuple)


        ' Local control variables
        Dim hv_P1 As HTuple = New HTuple, hv_P2 As HTuple = New HTuple
        Dim hv_RotAxis As HTuple = New HTuple, hv_D As HTuple = New HTuple
        Dim hv_T As HTuple = New HTuple, hv_RotAngle As HTuple = New HTuple
        Dim hv_Len As HTuple = New HTuple, hv_LenInv As HTuple = New HTuple

        ' Initialize local and output iconic variables

        If ((New HTuple(hv_MX1.TupleEqual(hv_MX2)))).TupleAnd(New
HTuple(hv_MY1.TupleEqual( _
            hv_MY2))).I() Then
            hv_QuatRotation = (((New
HTuple(1)).TupleConcat(0)).TupleConcat(0)).TupleConcat( _
                0)

            Exit Sub
        End If
        'Project the image point onto the track ball
        project_point_on_trackball(hv_MX1, hv_MY1, hv_VirtualTrackball,
hv_TrackballSize, _
            hv_P1)
        project_point_on_trackball(hv_MX2, hv_MY2, hv_VirtualTrackball,
hv_TrackballSize, _
            hv_P2)
        'The cross product of the projected points defines the rotation axis
        tuple_vector_cross_product(hv_P1, hv_P2, hv_RotAxis)
        'Compute the rotation angle
        hv_D = hv_P2.TupleSub(hv_P1)
        hv_T = ((((((hv_D.TupleMult(hv_D))).TupleSum())).TupleSqrt())).TupleDiv((New
HTuple(2.0)).TupleMult( _
            hv_TrackballSize))
        If New HTuple(hv_T.TupleGreater(New HTuple(1.0))).I() Then
            hv_T = New HTuple(1.0)
        End If
        If New HTuple(hv_T.TupleLess(New HTuple(-1.0))).I() Then
            hv_T = New HTuple(-1.0)
        End If
        hv_RotAngle = (New HTuple(2.0)).TupleMult(hv_T.TupleAsin())
        hv_Len = ((((hv_RotAxis.TupleMult(hv_RotAxis))).TupleSum())).TupleSqrt()
        If New HTuple(hv_Len.TupleGreater(New HTuple(0.0))).I() Then
            hv_LenInv = (New HTuple(1.0)).TupleDiv(hv_Len)
            hv_RotAxis = hv_RotAxis.TupleMult(hv_LenInv)
        End If
        HOperatorSet.AxisAngleToQuat(hv_RotAxis.TupleSelect(New HTuple(0)),
hv_RotAxis.TupleSelect( _
            New HTuple(1)), hv_RotAxis.TupleSelect(New HTuple(2)), hv_RotAngle,
hv_QuatRotation)

        Exit Sub
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Visualize the pose information of a 3d model
    Public Sub update_pose_information(ByVal hv_WindowHandle As HTuple, ByVal
hv_ObjectModel3DID As HTuple, _
        ByVal hv_CamParam As HTuple, ByVal hv_RefPose As HTuple, ByVal
hv_RelQuaternion As HTuple, _
        ByVal hv_Dist As HTuple, ByVal hv_LatitudeMin As HTuple, ByVal hv_LatitudeMax
As HTuple, _
        ByVal hv_LongitudeMin As HTuple, ByVal hv_LongitudeMax As HTuple, ByVal
hv_CamRollMin As HTuple, _
        ByVal hv_CamRollMax As HTuple, ByVal hv_DistMin As HTuple, ByVal hv_DistMax
As HTuple, _
        ByVal hv_MinFaceAngle As HTuple, ByVal hv_HiddenSurfaceRemoval As HTuple,
ByVal hv_ModelColor As HTuple)


        ' Stack for temporary objects
        Dim OTemp(10) As HObject

        ' Local iconic variables
        Dim ho_Image1 As HObject = Nothing, ho_Image2 As HObject = Nothing
        Dim ho_Image3 As HObject = Nothing, ho_CirclesLon As HObject = Nothing
        Dim ho_CircleLon As HObject = Nothing, ho_CirclesLat As HObject = Nothing
        Dim ho_CircleLat As HObject = Nothing, ho_XAxis As HObject = Nothing
        Dim ho_YAxis As HObject = Nothing, ho_ZAxis As HObject = Nothing
        Dim ho_LineLong As HObject = Nothing, ho_LineLat As HObject = Nothing
        Dim ho_ModelContours As HObject = Nothing, ho_ReferencePointCross As HObject
= Nothing
        Dim ho_ContPoseRange As HObject = Nothing, ho_Image As HObject = Nothing


        ' Local control variables
        Dim hv_CoordAxesLength As HTuple = New HTuple
        Dim hv_Row As HTuple = New HTuple, hv_Column As HTuple = New HTuple
        Dim hv_Width As HTuple = New HTuple, hv_Height As HTuple = New HTuple
        Dim hv_MinImageSize As HTuple = New HTuple, hv_Cx As HTuple = New HTuple
        Dim hv_Cy As HTuple = New HTuple, hv_CamParamSmall As HTuple = New HTuple
        Dim hv_SphereRadius As HTuple = New HTuple, hv_SpherePos As HTuple = New
HTuple
        Dim hv_Lon As HTuple = New HTuple, hv_Lat As HTuple = New HTuple
        Dim hv_RelHomMat As HTuple = New HTuple, hv_RelPose As HTuple = New HTuple
        Dim hv_RelHomMatInv As HTuple = New HTuple, hv_X As HTuple = New HTuple
        Dim hv_Y As HTuple = New HTuple, hv_Z As HTuple = New HTuple
        Dim hv_CamPos As HTuple = New HTuple, hv_CamXAx As HTuple = New HTuple
        Dim hv_CamYAx As HTuple = New HTuple, hv_CamZAx As HTuple = New HTuple
        Dim hv_ScaleFactor As HTuple = New HTuple, hv_CamPosProj As HTuple = New
HTuple
        Dim hv_CamXAxProj As HTuple = New HTuple, hv_CamYAxProj As HTuple = New
HTuple
        Dim hv_CamZAxProj As HTuple = New HTuple, hv_Vec As HTuple = New HTuple
        Dim hv_CamRow As HTuple = New HTuple, hv_CamCol As HTuple = New HTuple
        Dim hv_CamXAxRow As HTuple = New HTuple, hv_CamXAxCol As HTuple = New HTuple
        Dim hv_CamYAxRow As HTuple = New HTuple, hv_CamYAxCol As HTuple = New HTuple
        Dim hv_CamZAxRow As HTuple = New HTuple, hv_CamZAxCol As HTuple = New HTuple
        Dim hv_CamLon As HTuple = New HTuple, hv_CamLat As HTuple = New HTuple
        Dim hv_Radius As HTuple = New HTuple, hv_CamRoll As HTuple = New HTuple
        Dim hv_XRangeCamPos As HTuple = New HTuple, hv_YRangeCamPos As HTuple = New
HTuple
        Dim hv_StepLat As HTuple = New HTuple, hv_ViewingDist As HTuple = New HTuple
        Dim hv_LongitudeMinDisp As HTuple = New HTuple, hv_LongitudeMaxDisp As HTuple
= New HTuple
        Dim hv_LatitudeMinDisp As HTuple = New HTuple, hv_LatitudeMaxDisp As HTuple =
New HTuple
        Dim hv_XRangeLatMin As HTuple = New HTuple, hv_YRangeLatMin As HTuple = New
HTuple
        Dim hv_XRangeLatMax As HTuple = New HTuple, hv_YRangeLatMax As HTuple = New
HTuple
        Dim hv_Long As HTuple = New HTuple, hv_XRangeLongMin As HTuple = New HTuple
        Dim hv_YRangeLongMin As HTuple = New HTuple, hv_XRangeLongMax As HTuple = New
HTuple
        Dim hv_YRangeLongMax As HTuple = New HTuple, hv_LineStyle As HTuple = New
HTuple
        Dim hv_MaxDist As HTuple = New HTuple, hv_Precision As HTuple = New HTuple
        Dim hv_DistFormat As HTuple = New HTuple, hv_DegreeSign As HTuple = New
HTuple

        Dim hv_RefPose_COPY_INP_TMP As HTuple
        hv_RefPose_COPY_INP_TMP = hv_RefPose.Clone()

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_Image1)
        HOperatorSet.GenEmptyObj(ho_Image2)
        HOperatorSet.GenEmptyObj(ho_Image3)
        HOperatorSet.GenEmptyObj(ho_CirclesLon)
        HOperatorSet.GenEmptyObj(ho_CircleLon)
        HOperatorSet.GenEmptyObj(ho_CirclesLat)
        HOperatorSet.GenEmptyObj(ho_CircleLat)
        HOperatorSet.GenEmptyObj(ho_XAxis)
        HOperatorSet.GenEmptyObj(ho_YAxis)
        HOperatorSet.GenEmptyObj(ho_ZAxis)
        HOperatorSet.GenEmptyObj(ho_LineLong)
        HOperatorSet.GenEmptyObj(ho_LineLat)
        HOperatorSet.GenEmptyObj(ho_ModelContours)
        HOperatorSet.GenEmptyObj(ho_ReferencePointCross)
        HOperatorSet.GenEmptyObj(ho_ContPoseRange)
        HOperatorSet.GenEmptyObj(ho_Image)

        Try
            '
            'Initialize some values
            hv_CoordAxesLength = New HTuple(50)
            HOperatorSet.GetWindowExtents(hv_WindowHandle, hv_Row, hv_Column,
hv_Width, _
                hv_Height)
            ho_Image1.Dispose()
            HOperatorSet.GenImageConst(ho_Image1, New HTuple("byte"), hv_Width,
hv_Height)
```

```
            ho_Image2.Dispose()
            HOperatorSet.GenImageConst(ho_Image2, New HTuple("byte"), hv_Width,
hv_Height)
            ho_Image3.Dispose()
            HOperatorSet.GenImageConst(ho_Image3, New HTuple("byte"), hv_Width,
hv_Height)
            hv_MinImageSize = ((hv_Width.TupleConcat(hv_Height))).TupleMin()
            hv_Cx = hv_MinImageSize.TupleDiv(New HTuple(2))
            hv_Cy = hv_MinImageSize.TupleDiv(New HTuple(2))
            hv_CamParamSmall = hv_CamParam.Clone()
            If IsNothing(hv_CamParamSmall) Then
                hv_CamParamSmall = New HTuple
            End If
            hv_CamParamSmall(New HTuple(4)) = hv_Cx
            If IsNothing(hv_CamParamSmall) Then
                hv_CamParamSmall = New HTuple
            End If
            hv_CamParamSmall(New HTuple(5)) = hv_Cy
            If IsNothing(hv_CamParamSmall) Then
                hv_CamParamSmall = New HTuple
            End If
            hv_CamParamSmall(New HTuple(6)) = hv_MinImageSize
            If IsNothing(hv_CamParamSmall) Then
                hv_CamParamSmall = New HTuple
            End If
            hv_CamParamSmall(New HTuple(7)) = hv_MinImageSize
            hv_SphereRadius = ((hv_MinImageSize.TupleMult(New
HTuple(0.8)))).TupleDiv(New HTuple(2))
            hv_SpherePos = hv_MinImageSize.TupleDiv(New HTuple(2))
            '
            'Project the viewing sphere
            ho_CirclesLon.Dispose()
            HOperatorSet.GenContourPolygonXld(ho_CirclesLon,
((hv_SpherePos.TupleSub(hv_SphereRadius))).TupleConcat( _
                hv_SpherePos.TupleAdd(hv_SphereRadius)),
hv_SpherePos.TupleConcat(hv_SpherePos))
            For hv_Lon = (New HTuple(30)) To (New HTuple(90)) Step (New HTuple(30))
                ho_CircleLon.Dispose()
                HOperatorSet.GenEllipseContourXld(ho_CircleLon, hv_SpherePos,
hv_SpherePos, _
                    New HTuple(0),
hv_SphereRadius.TupleMult(((hv_Lon.TupleRad())).TupleSin( _
                    )), hv_SphereRadius, New HTuple(0), New HTuple(6.28318), New
HTuple("positive"), _
                    New HTuple(1.5))
                HOperatorSet.ConcatObj(ho_CirclesLon, ho_CircleLon, OTemp(0))
                ho_CirclesLon.Dispose()
                ho_CirclesLon = OTemp(0)
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            ho_CirclesLat.Dispose()
            HOperatorSet.GenEmptyObj(ho_CirclesLat)
            For hv_Lat = (New HTuple(-60)) To (New HTuple(60)) Step (New HTuple(30))
                ho_CircleLat.Dispose()
                HOperatorSet.GenContourPolygonXld(ho_CircleLat,
((hv_SpherePos.TupleSub(hv_SphereRadius.TupleMult( _

((hv_Lat.TupleRad())).TupleSin())))).TupleConcat(hv_SpherePos.TupleSub( _
                    hv_SphereRadius.TupleMult(((hv_Lat.TupleRad())).TupleSin()))),
((hv_SpherePos.TupleSub( _
                    hv_SphereRadius.TupleMult(((((New
HTuple(90.0)).TupleSub(hv_Lat))).TupleRad( _

))).TupleSin()))))).TupleConcat(hv_SpherePos.TupleAdd(hv_SphereRadius.TupleMult( _
                    (((((New
HTuple(90.0)).TupleSub(hv_Lat))).TupleRad())).TupleSin())))))
                HOperatorSet.ConcatObj(ho_CirclesLat, ho_CircleLat, OTemp(0))
                ho_CirclesLat.Dispose()
                ho_CirclesLat = OTemp(0)
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            '
            'Project the camera position
            HOperatorSet.QuatToHomMat3d(hv_RelQuaternion, hv_RelHomMat)
            HOperatorSet.HomMat3dToPose(hv_RelHomMat, hv_RelPose)
            If IsNothing(hv_RelPose) Then
                hv_RelPose = New HTuple
            End If
            hv_RelPose(New HTuple(2)) = hv_Dist
            HOperatorSet.PoseToHomMat3d(hv_RelPose, hv_RelHomMat)
            HOperatorSet.HomMat3dInvert(hv_RelHomMat, hv_RelHomMatInv)
            HOperatorSet.AffineTransPoint3d(hv_RelHomMatInv, New HTuple(0), New
HTuple(0), _
                New HTuple(0), hv_X, hv_Y, hv_Z)
            hv_CamPos = ((hv_X.TupleConcat(hv_Y))).TupleConcat(hv_Z)
            HOperatorSet.AffineTransPoint3d(hv_RelHomMatInv, New HTuple(1.0), New
HTuple(0), _
                New HTuple(0), hv_X, hv_Y, hv_Z)
            hv_CamXAx = ((hv_X.TupleConcat(hv_Y))).TupleConcat(hv_Z)
            HOperatorSet.AffineTransPoint3d(hv_RelHomMatInv, New HTuple(0), New
HTuple(1.0), _
                New HTuple(0), hv_X, hv_Y, hv_Z)
            hv_CamYAx = ((hv_X.TupleConcat(hv_Y))).TupleConcat(hv_Z)
            HOperatorSet.AffineTransPoint3d(hv_RelHomMatInv, New HTuple(0), New
HTuple(0), _
                New HTuple(1.0), hv_X, hv_Y, hv_Z)
            hv_CamZAx = ((hv_X.TupleConcat(hv_Y))).TupleConcat(hv_Z)
            'Scale the position to the viewing sphere
            hv_ScaleFactor =
hv_SphereRadius.TupleDiv(((((hv_CamPos.TupleMult(hv_CamPos))).TupleSum( _
                ))).TupleSqrt())
            hv_CamPosProj = hv_ScaleFactor.TupleMult(hv_CamPos)
            hv_CamXAxProj = hv_ScaleFactor.TupleMult(hv_CamXAx)
            hv_CamYAxProj = hv_ScaleFactor.TupleMult(hv_CamYAx)
            hv_CamZAxProj = hv_ScaleFactor.TupleMult(hv_CamZAx)
            'Scale the coordinate axis to the desired length
            hv_Vec = hv_CamXAxProj.TupleSub(hv_CamPosProj)
            hv_ScaleFactor =
hv_CoordAxesLength.TupleDiv(((((hv_Vec.TupleMult(hv_Vec))).TupleSum( _
                ))).TupleSqrt())
            hv_CamXAxProj =
hv_CamPosProj.TupleAdd(hv_ScaleFactor.TupleMult(hv_CamXAxProj.TupleSub( _
                hv_CamPosProj)))
            hv_CamYAxProj =
hv_CamPosProj.TupleAdd(hv_ScaleFactor.TupleMult(hv_CamYAxProj.TupleSub( _
                hv_CamPosProj)))
            hv_CamZAxProj =
hv_CamPosProj.TupleAdd(hv_ScaleFactor.TupleMult(hv_CamZAxProj.TupleSub( _
                hv_CamPosProj)))
            hv_CamRow = hv_SpherePos.TupleAdd(hv_CamPosProj.TupleSelect(New
HTuple(1)))
            hv_CamCol = hv_SpherePos.TupleAdd(hv_CamPosProj.TupleSelect(New
HTuple(0)))
            hv_CamXAxRow = hv_SpherePos.TupleAdd(hv_CamXAxProj.TupleSelect(New
HTuple(1)))
            hv_CamXAxCol = hv_SpherePos.TupleAdd(hv_CamXAxProj.TupleSelect(New
HTuple(0)))
            hv_CamYAxRow = hv_SpherePos.TupleAdd(hv_CamYAxProj.TupleSelect(New
HTuple(1)))
            hv_CamYAxCol = hv_SpherePos.TupleAdd(hv_CamYAxProj.TupleSelect(New
HTuple(0)))
            hv_CamZAxRow = hv_SpherePos.TupleAdd(hv_CamZAxProj.TupleSelect(New
HTuple(1)))
            hv_CamZAxCol = hv_SpherePos.TupleAdd(hv_CamZAxProj.TupleSelect(New
HTuple(0)))
            ho_XAxis.Dispose()
            gen_arrow_cont(ho_XAxis, hv_CamRow, hv_CamCol, hv_CamXAxRow,
hv_CamXAxCol)
            ho_YAxis.Dispose()
            gen_arrow_cont(ho_YAxis, hv_CamRow, hv_CamCol, hv_CamYAxRow,
hv_CamYAxCol)
            ho_ZAxis.Dispose()
            gen_arrow_cont(ho_ZAxis, hv_CamRow, hv_CamCol, hv_CamZAxRow,
hv_CamZAxCol)
            '
            'Compute the current longitude, latitude, and camera roll angle
            HOperatorSet.ConvertPoint3dCartToSpher(hv_CamPos.TupleSelect(New
HTuple(0)), _
                hv_CamPos.TupleSelect(New HTuple(1)), hv_CamPos.TupleSelect(New
HTuple(2)), _
                New HTuple("-y"), New HTuple("-z"), hv_CamLon, hv_CamLat, hv_Radius)
            hv_CamRoll = ((hv_RelPose.TupleSelect(New HTuple(5)))).TupleNeg()
            Do While New HTuple(hv_CamRoll.TupleGreater(New HTuple(180.0))).I()
                hv_CamRoll = hv_CamRoll.TupleSub(New HTuple(360.0))
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Loop
            Do While New HTuple(hv_CamRoll.TupleLess(New HTuple(-180.0))).I()
                hv_CamRoll = hv_CamRoll.TupleAdd(New HTuple(360.0))
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Loop
            hv_CamRoll = ((hv_CamRoll.TupleMult(New HTuple(3.1416)))).TupleDiv(New
HTuple(180.0))
            '
            'Indicate longitude and latitude by lines
            ho_LineLong.Dispose()
            HOperatorSet.GenContourPolygonXld(ho_LineLong,
hv_SpherePos.TupleConcat(hv_SpherePos), _

hv_SpherePos.TupleConcat(hv_SpherePos.TupleAdd(hv_SphereRadius.TupleMult( _
                hv_CamLon.TupleSin()))))
            hv_XRangeCamPos = New HTuple()
            hv_YRangeCamPos = New HTuple()
            hv_StepLat = (New HTuple(1)).TupleRad()
            If New HTuple(hv_CamLat.TupleGreater(New HTuple(0))).I() Then
```

```
                hv_StepLat = (New HTuple(1)).TupleRad()
            Else
                hv_StepLat = (((New HTuple(1)).TupleRad())).TupleNeg()
            End If
            For hv_Lat = (New HTuple(0)) To hv_CamLat Step hv_StepLat
                HOperatorSet.ConvertPoint3dSpherToCart(hv_CamLon, hv_Lat,
hv_SphereRadius, _
                    New HTuple("-y"), New HTuple("-z"), hv_X, hv_Y, hv_Z)
                hv_XRangeCamPos = hv_XRangeCamPos.TupleConcat(hv_X)
                hv_YRangeCamPos = hv_YRangeCamPos.TupleConcat(hv_Y)
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            ho_LineLat.Dispose()
            HOperatorSet.GenContourPolygonXld(ho_LineLat,
hv_SpherePos.TupleAdd(hv_YRangeCamPos), _
                hv_SpherePos.TupleAdd(hv_XRangeCamPos))
            '
            'Project the model using the reference pose
            determine_optimum_viewing_distance(hv_ObjectModel3DID, hv_CamParamSmall,
hv_MinImageSize, _
                New HTuple(0.7), hv_ViewingDist)
            If IsNothing(hv_RefPose_COPY_INP_TMP) Then
                hv_RefPose_COPY_INP_TMP = New HTuple
            End If
            hv_RefPose_COPY_INP_TMP(New HTuple(2)) =
((hv_RefPose_COPY_INP_TMP.TupleSelect( _
                New HTuple(2)))).TupleAdd(hv_ViewingDist)
            ho_ModelContours.Dispose()
            HOperatorSet.ProjectObjectModel3d(ho_ModelContours, hv_ObjectModel3DID,
hv_CamParamSmall, _
                hv_RefPose_COPY_INP_TMP, hv_HiddenSurfaceRemoval, hv_MinFaceAngle)
            ho_ReferencePointCross.Dispose()
            HOperatorSet.GenCrossContourXld(ho_ReferencePointCross, hv_Cy, hv_Cx, New
HTuple(6), _
                New HTuple(0))
            '
            'The pose range can only be displayed from -90° to +90°
            hv_LongitudeMinDisp = ((hv_LongitudeMin.TupleConcat((((New
HTuple(90)).TupleRad( _
                ))).TupleNeg()))).TupleMax()
            hv_LongitudeMaxDisp = ((hv_LongitudeMax.TupleConcat((New
HTuple(90)).TupleRad( _
                )))).TupleMin()
            hv_LatitudeMinDisp = ((hv_LatitudeMin.TupleConcat((((New
HTuple(90)).TupleRad( _
                ))).TupleNeg()))).TupleMax()
            hv_LatitudeMaxDisp = ((hv_LatitudeMax.TupleConcat((New
HTuple(90)).TupleRad()))).TupleMin( _
                )
            '
            'Project the border of the pose range at LatituteMin and LatitudeMax
            hv_XRangeLatMin = New HTuple()
            hv_YRangeLatMin = New HTuple()
            hv_XRangeLatMax = New HTuple()
            hv_YRangeLatMax = New HTuple()
            For hv_Long = hv_LongitudeMinDisp To hv_LongitudeMaxDisp Step ( _
                New HTuple(1)).TupleRad()
                HOperatorSet.ConvertPoint3dSpherToCart(hv_Long, hv_LatitudeMinDisp,
hv_SphereRadius, _
                    New HTuple("-y"), New HTuple("-z"), hv_X, hv_Y, hv_Z)
                hv_XRangeLatMin = hv_XRangeLatMin.TupleConcat(hv_X)
                hv_YRangeLatMin = hv_YRangeLatMin.TupleConcat(hv_Y)
                HOperatorSet.ConvertPoint3dSpherToCart(hv_Long, hv_LatitudeMaxDisp, 
hv_SphereRadius, _
                    New HTuple("-y"), New HTuple("-z"), hv_X, hv_Y, hv_Z)
                hv_XRangeLatMax = hv_XRangeLatMax.TupleConcat(hv_X)
                hv_YRangeLatMax = hv_YRangeLatMax.TupleConcat(hv_Y)
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            '
            'Project the border of the pose range at LongitudeMin and LongitudeMax
            hv_XRangeLongMin = New HTuple()
            hv_YRangeLongMin = New HTuple()
            hv_XRangeLongMax = New HTuple()
            hv_YRangeLongMax = New HTuple()
            For hv_Lat = hv_LatitudeMinDisp To hv_LatitudeMaxDisp Step ( _
                New HTuple(1)).TupleRad()
                HOperatorSet.ConvertPoint3dSpherToCart(hv_LongitudeMinDisp, hv_Lat, 
hv_SphereRadius, _
                    New HTuple("-y"), New HTuple("-z"), hv_X, hv_Y, hv_Z)
                hv_XRangeLongMin = hv_XRangeLongMin.TupleConcat(hv_X)
                hv_YRangeLongMin = hv_YRangeLongMin.TupleConcat(hv_Y)
                HOperatorSet.ConvertPoint3dSpherToCart(hv_LongitudeMaxDisp, hv_Lat, 
hv_SphereRadius, _
                    New HTuple("-y"), New HTuple("-z"), hv_X, hv_Y, hv_Z)
                hv_XRangeLongMax = hv_XRangeLongMax.TupleConcat(hv_X)
                hv_YRangeLongMax = hv_YRangeLongMax.TupleConcat(hv_Y)
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            ho_ContPoseRange.Dispose()
            HOperatorSet.GenContourPolygonXld(ho_ContPoseRange, 
hv_SpherePos.TupleAdd(((( _

(hv_YRangeLatMin.TupleConcat(hv_YRangeLongMax))).TupleConcat(hv_YRangeLatMax.TupleInv
erse( _
                )))).TupleConcat(hv_YRangeLongMin.TupleInverse())), 
hv_SpherePos.TupleAdd( _

((((hv_XRangeLatMin.TupleConcat(hv_XRangeLongMax))).TupleConcat(hv_XRangeLatMax.Tuple
Inverse( _
                )))).TupleConcat(hv_XRangeLongMin.TupleInverse())))
            ho_Image.Dispose()
            HOperatorSet.Compose3(ho_Image1, ho_Image2, ho_Image3, ho_Image)
            HOperatorSet.PaintXld(ho_ContPoseRange, ho_Image, OTemp(0), ((New 
HTuple(80)).TupleConcat( _
                0)).TupleConcat(80))
            ho_Image.Dispose()
            ho_Image = OTemp(0)
            '
            'Display the graphics
            'dev_set_window(...);
            HOperatorSet.DispObj(ho_Image, hv_WindowHandle)
            HOperatorSet.SetColor(hv_WindowHandle, New HTuple("blue"))
            HOperatorSet.DispObj(ho_CirclesLon, hv_WindowHandle)
            HOperatorSet.DispObj(ho_CirclesLat, hv_WindowHandle)
            HOperatorSet.SetColor(hv_WindowHandle, hv_ModelColor)
            HOperatorSet.DispObj(ho_ModelContours, hv_WindowHandle)
            HOperatorSet.SetColor(hv_WindowHandle, New HTuple("light gray"))
            HOperatorSet.DispObj(ho_ReferencePointCross, hv_WindowHandle)
            HOperatorSet.SetColor(hv_WindowHandle, New HTuple("magenta"))
            HOperatorSet.DispObj(ho_ContPoseRange, hv_WindowHandle)
            If ((New HTuple(hv_CamLon.TupleGreater((((New
HTuple(90)).TupleRad())).TupleNeg( _
                ))))).TupleAnd(New HTuple(hv_CamLon.TupleLess((New
HTuple(90)).TupleRad())))).I() Then
                HOperatorSet.SetColor(hv_WindowHandle, New HTuple("yellow"))
            Else
                HOperatorSet.SetColor(hv_WindowHandle, New HTuple("tan"))
            End If
            HOperatorSet.DispObj(ho_XAxis, hv_WindowHandle)
            HOperatorSet.DispObj(ho_YAxis, hv_WindowHandle)
            HOperatorSet.DispObj(ho_ZAxis, hv_WindowHandle)
            HOperatorSet.SetTposition(hv_WindowHandle, hv_CamXAxRow, hv_CamXAxCol)
            HOperatorSet.WriteString(hv_WindowHandle, New HTuple("x"))
            HOperatorSet.SetTposition(hv_WindowHandle, hv_CamYAxRow, hv_CamYAxCol)
            HOperatorSet.WriteString(hv_WindowHandle, New HTuple("y"))
            HOperatorSet.SetTposition(hv_WindowHandle, hv_CamZAxRow, hv_CamZAxCol)
            HOperatorSet.WriteString(hv_WindowHandle, New HTuple("z"))
            HOperatorSet.GetLineStyle(hv_WindowHandle, hv_LineStyle)
            HOperatorSet.SetLineStyle(hv_WindowHandle, (New HTuple(4)).TupleConcat( _
                4))
            HOperatorSet.DispObj(ho_LineLong, hv_WindowHandle)
            HOperatorSet.DispObj(ho_LineLat, hv_WindowHandle)
            HOperatorSet.SetLineStyle(hv_WindowHandle, hv_LineStyle)
            '
            'Display the text
            'Compute the format string in dependence of the distance magnitude
            hv_MaxDist =
((((hv_Dist.TupleConcat(hv_DistMin))).TupleConcat(hv_DistMax))).TupleMax( _
                )
            If New HTuple(hv_MaxDist.TupleGreater(New HTuple(0))).I() Then
                hv_Precision = (((New HTuple(0)).TupleConcat((New
HTuple(4)).TupleSub(((hv_MaxDist.TupleLog10( _
                    ))).TupleInt())))).TupleMax()
                hv_Precision = ((hv_Precision.TupleConcat(New HTuple(4)))).TupleMin()
            Else
                hv_Precision = New HTuple(4)
            End If
            hv_DistFormat = (((New
HTuple("6.")).TupleAdd(hv_Precision))).TupleAdd(New HTuple("f"))
            HOperatorSet.SetColor(hv_ExpDefaultWinHandle, New HTuple("white"))
            HOperatorSet.TupleChr(New HTuple(&HB0), hv_DegreeSign)
            HOperatorSet.SetTposition(hv_WindowHandle, hv_MinImageSize, New
HTuple(30))
            HOperatorSet.WriteString(hv_WindowHandle, New HTuple("Current Reference
Pose Angles:"))
            HOperatorSet.SetTposition(hv_WindowHandle, hv_MinImageSize.TupleAdd(New
HTuple(20)), _
                New HTuple(30))
            HOperatorSet.WriteString(hv_WindowHandle, (((((((((((((((((New HTuple("
a =")).TupleAdd( _
                ((hv_RefPose_COPY_INP_TMP.TupleSelect(New
HTuple(3)))).TupleString((New HTuple("4.0f")).S()))))).TupleAdd( _
                hv_DegreeSign))).TupleAdd(New HTuple(";")))).TupleAdd(New HTuple(" b
=")))).TupleAdd( _
                ((hv_RefPose_COPY_INP_TMP.TupleSelect(New
HTuple(4)))).TupleString((New HTuple("4.0f")).S()))))).TupleAdd( _
                hv_DegreeSign))).TupleAdd(New HTuple(";")))).TupleAdd(New HTuple(" g
=")))).TupleAdd( _
```

```
                ((hv_RefPose_COPY_INP_TMP.TupleSelect(New
HTuple(5)))).TupleString((New HTuple("4.0f")).S())))).TupleAdd( _
                hv_DegreeSign))
            HOperatorSet.SetTposition(hv_WindowHandle, hv_MinImageSize.TupleAdd(New
HTuple(50)), _
                New HTuple(30))
            HOperatorSet.WriteString(hv_WindowHandle, (((((((((((((((New
HTuple("Longitude: ")).TupleAdd( _
                ((((hv_CamLon.TupleMult(New HTuple(180)))).TupleDiv(New
HTuple(3.1416)))).TupleString(( _
                New HTuple("+4.0f")).S()))))).TupleAdd(hv_DegreeSign))).TupleAdd(New
HTuple("; [")))).TupleAdd( _
                ((((hv_LongitudeMin.TupleMult(New HTuple(180)))).TupleDiv(New
HTuple(3.1416)))).TupleString(( _
                New HTuple("+4.0f")).S()))))).TupleAdd(hv_DegreeSign))).TupleAdd(New
HTuple("; ")))).TupleAdd( _
                ((((hv_LongitudeMax.TupleMult(New HTuple(180)))).TupleDiv(New
HTuple(3.1416)))).TupleString(( _
                New HTuple("+4.0f")).S()))))).TupleAdd(hv_DegreeSign))).TupleAdd(New
HTuple("]")))
            HOperatorSet.SetTposition(hv_WindowHandle, hv_MinImageSize.TupleAdd(New
HTuple(70)), _
                New HTuple(30))
            HOperatorSet.WriteString(hv_WindowHandle, (((((((((((((((New
HTuple("Latitude:  ")).TupleAdd( _
                ((((hv_CamLat.TupleMult(New HTuple(180)))).TupleDiv(New
HTuple(3.1416)))).TupleString(( _
                New HTuple("+4.0f")).S()))))).TupleAdd(hv_DegreeSign))).TupleAdd(New
HTuple("; [")))).TupleAdd( _
                ((((hv_LatitudeMin.TupleMult(New HTuple(180)))).TupleDiv(New
HTuple(3.1416)))).TupleString(( _
                New HTuple("+4.0f")).S()))))).TupleAdd(hv_DegreeSign))).TupleAdd(New
HTuple("; ")))).TupleAdd( _
                ((((hv_LatitudeMax.TupleMult(New HTuple(180)))).TupleDiv(New
HTuple(3.1416)))).TupleString(( _
                New HTuple("+4.0f")).S()))))).TupleAdd(hv_DegreeSign))).TupleAdd(New
HTuple("]")))
            HOperatorSet.SetTposition(hv_WindowHandle, hv_MinImageSize.TupleAdd(New
HTuple(90)), _
                New HTuple(30))
            HOperatorSet.WriteString(hv_WindowHandle, (((((((((((((((New
HTuple("CamRoll:   ")).TupleAdd( _
                ((((hv_CamRoll.TupleMult(New HTuple(180)))).TupleDiv(New
HTuple(3.1416)))).TupleString(( _
                New HTuple("+4.0f")).S()))))).TupleAdd(hv_DegreeSign))).TupleAdd(New
HTuple("; [")))).TupleAdd( _
                ((((hv_CamRollMin.TupleMult(New HTuple(180)))).TupleDiv(New
HTuple(3.1416)))).TupleString(( _
                New HTuple("+4.0f")).S()))))).TupleAdd(hv_DegreeSign))).TupleAdd(New
HTuple("; ")))).TupleAdd( _
                ((((hv_CamRollMax.TupleMult(New HTuple(180)))).TupleDiv(New
HTuple(3.1416)))).TupleString(( _
                New HTuple("+4.0f")).S()))))).TupleAdd(hv_DegreeSign))).TupleAdd(New
HTuple("]")))
            HOperatorSet.SetTposition(hv_WindowHandle, hv_MinImageSize.TupleAdd(New
HTuple(110)), _
                New HTuple(30))
            HOperatorSet.WriteString(hv_WindowHandle, (((((((((New
HTuple("Distance: ")).TupleAdd( _
                hv_Dist.TupleString(hv_DistFormat.S()))))).TupleAdd(New HTuple(";
[")))).TupleAdd( _
                hv_DistMin.TupleString(hv_DistFormat.S()))))).TupleAdd(New HTuple(";
")))).TupleAdd( _
                hv_DistMax.TupleString(hv_DistFormat.S()))))).TupleAdd(New
HTuple("]")))
```

```
            HOperatorSet.SetTposition(hv_WindowHandle, hv_MinImageSize.TupleAdd(New
HTuple(140)), _
                New HTuple(30))
            HOperatorSet.WriteString(hv_WindowHandle, (((New HTuple("MinFaceAngle:
")).TupleAdd( _
                ((((hv_MinFaceAngle.TupleMult(New HTuple(180)))).TupleDiv(New
HTuple(3.1416)))).TupleString(( _
                New HTuple("4.0f")).S()))))).TupleAdd(hv_DegreeSign))
            ho_Image1.Dispose()
            ho_Image2.Dispose()
            ho_Image3.Dispose()
            ho_CirclesLon.Dispose()
            ho_CircleLon.Dispose()
            ho_CirclesLat.Dispose()
            ho_CircleLat.Dispose()
            ho_XAxis.Dispose()
            ho_YAxis.Dispose()
            ho_ZAxis.Dispose()
            ho_LineLong.Dispose()
            ho_LineLat.Dispose()
            ho_ModelContours.Dispose()
            ho_ReferencePointCross.Dispose()
            ho_ContPoseRange.Dispose()
            ho_Image.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_Image1.Dispose()
            ho_Image2.Dispose()
            ho_Image3.Dispose()
            ho_CirclesLon.Dispose()
            ho_CircleLon.Dispose()
            ho_CirclesLat.Dispose()
            ho_CircleLat.Dispose()
            ho_XAxis.Dispose()
            ho_YAxis.Dispose()
            ho_ZAxis.Dispose()
            ho_LineLong.Dispose()
            ho_LineLat.Dispose()
            ho_ModelContours.Dispose()
            ho_ReferencePointCross.Dispose()
            ho_ContPoseRange.Dispose()
            ho_Image.Dispose()

            Throw HDevExpDefaultException
        End Try
    End Sub

    ' Chapter: Matching-3D
    ' Short Description: Reflect the selection of a menu button
    Public Sub analyze_menu_event(ByVal ho_MenuRegions As HObject, ByVal hv_MenuText
As HTuple, _
        ByVal hv_WindowHandleMenu As HTuple, ByVal hv_Row As HTuple, ByVal hv_Col As
HTuple, _
        ByVal hv_LongitudeMinIn As HTuple, ByVal hv_LongitudeMaxIn As HTuple, ByVal
hv_LatitudeMinIn As HTuple, _
        ByVal hv_LatitudeMaxIn As HTuple, ByVal hv_CamRollMinIn As HTuple, ByVal
hv_CamRollMaxIn As HTuple, _
        ByVal hv_DistMinIn As HTuple, ByVal hv_DistMaxIn As HTuple, ByVal
hv_MinFaceAngleIn As HTuple, _
        ByVal hv_RefPoseIn As HTuple, ByVal hv_RelQuaternionIn As HTuple, ByVal
hv_ImageNoIn As HTuple, _
        ByVal hv_HiddenSurfaceRemovalIn As HTuple, ByVal hv_MouseModeIn As HTuple,
ByVal hv_Dist As HTuple, _
```

```
        ByVal hv_ObjectModel3DID As HTuple, ByVal hv_RefPoseReset As HTuple, ByRef
hv_SelectedButton As HTuple, _
        ByRef hv_Exit As HTuple, ByRef hv_LongitudeMinOut As HTuple, ByRef
hv_LongitudeMaxOut As HTuple, _
        ByRef hv_LatitudeMinOut As HTuple, ByRef hv_LatitudeMaxOut As HTuple, ByRef
hv_CamRollMinOut As HTuple, _
        ByRef hv_CamRollMaxOut As HTuple, ByRef hv_DistMinOut As HTuple, ByRef
hv_DistMaxOut As HTuple, _
        ByRef hv_MinFaceAngleOut As HTuple, ByRef hv_RefPoseOut As HTuple, ByRef
hv_RelQuaternionOut As HTuple, _
        ByRef hv_ImageNoOut As HTuple, ByRef hv_HiddenSurfaceRemovalOut As HTuple,
ByRef hv_MouseModeOut As HTuple)



        ' Local control variables
        Dim hv_RefHomMat As HTuple = New HTuple, hv_RelHomMat As HTuple = New HTuple
        Dim hv_AbsHomMat As HTuple = New HTuple, hv_RelPose As HTuple = New HTuple
        Dim hv_RelPoseInv As HTuple = New HTuple, hv_CameraPos As HTuple = New HTuple
        Dim hv_CurrentLongitude As HTuple = New HTuple, hv_CurrentLatitude As HTuple
= New HTuple
        Dim hv_Radius As HTuple = New HTuple, hv_CurrentCamRoll As HTuple = New
HTuple

        ' Initialize local and output iconic variables

        hv_Exit = New HTuple(0)
        get_selected_button(ho_MenuRegions, hv_MenuText, hv_WindowHandleMenu, hv_Row,
hv_Col, _
            hv_SelectedButton)
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Exit"))).I() Then
            hv_Exit = New HTuple(1)
        End If
        '
        'Copy the input parameters to the output parameters
        hv_LongitudeMinOut = hv_LongitudeMinIn.Clone()
        hv_LongitudeMaxOut = hv_LongitudeMaxIn.Clone()
        hv_LatitudeMinOut = hv_LatitudeMinIn.Clone()
        hv_LatitudeMaxOut = hv_LatitudeMaxIn.Clone()
        hv_CamRollMinOut = hv_CamRollMinIn.Clone()
        hv_CamRollMaxOut = hv_CamRollMaxIn.Clone()
        hv_DistMinOut = hv_DistMinIn.Clone()
        hv_DistMaxOut = hv_DistMaxIn.Clone()
        hv_MinFaceAngleOut = hv_MinFaceAngleIn.Clone()
        hv_RefPoseOut = hv_RefPoseIn.Clone()
        hv_RelQuaternionOut = hv_RelQuaternionIn.Clone()
        hv_ImageNoOut = hv_ImageNoIn.Clone()
        hv_HiddenSurfaceRemovalOut = hv_HiddenSurfaceRemovalIn.Clone()
        hv_MouseModeOut = hv_MouseModeIn.Clone()
        '
        'Check which menu button was pressed.
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Set Reference
Pose"))).I() Then
            HOperatorSet.PoseToHomMat3d(hv_RefPoseOut, hv_RefHomMat)
            HOperatorSet.QuatToHomMat3d(hv_RelQuaternionOut, hv_RelHomMat)
            HOperatorSet.HomMat3dCompose(hv_RelHomMat, hv_RefHomMat, hv_AbsHomMat)
            HOperatorSet.HomMat3dToPose(hv_AbsHomMat, hv_RefPoseOut)
            hv_RelQuaternionOut = (((New
HTuple(1)).TupleConcat(0)).TupleConcat(0)).TupleConcat( _
                0)
        End If
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Add to Pose
Range"))).I() Then
            HOperatorSet.QuatToHomMat3d(hv_RelQuaternionIn, hv_RelHomMat)
            HOperatorSet.HomMat3dToPose(hv_RelHomMat, hv_RelPose)
            If IsNothing(hv_RelPose) Then
                hv_RelPose = New HTuple
            End If
            hv_RelPose(New HTuple(2)) = hv_Dist
            HOperatorSet.PoseInvert(hv_RelPose, hv_RelPoseInv)
            hv_CameraPos = ((((hv_RelPoseInv.TupleSelect(New
HTuple(0)))).TupleConcat(hv_RelPoseInv.TupleSelect( _
                New HTuple(1))))).TupleConcat(hv_RelPoseInv.TupleSelect(New
HTuple(2)))
            'Compute the current longitude, latitude, and camera roll angle
            HOperatorSet.ConvertPoint3dCartToSpher(hv_CameraPos.TupleSelect(New
HTuple(0)), _
                hv_CameraPos.TupleSelect(New HTuple(1)), hv_CameraPos.TupleSelect(New
HTuple(2)), _
                New HTuple("-y"), New HTuple("-z"), hv_CurrentLongitude,
hv_CurrentLatitude, _
                hv_Radius)
            hv_CurrentCamRoll = hv_RelPose.TupleSelect(New HTuple(5))
            Do While New HTuple(hv_CurrentCamRoll.TupleGreater(New
HTuple(180.0))).I()
                hv_CurrentCamRoll = hv_CurrentCamRoll.TupleSub(New HTuple(360.0))
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Loop
            Do While New HTuple(hv_CurrentCamRoll.TupleLess(New HTuple(-180.0))).I()
                hv_CurrentCamRoll = hv_CurrentCamRoll.TupleAdd(New HTuple(360.0))
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Loop
            hv_CurrentCamRoll = ((hv_CurrentCamRoll.TupleMult(New
HTuple(3.1416)))).TupleDiv( _
                New HTuple(180.0))
            hv_LongitudeMinOut =
((hv_LongitudeMinOut.TupleConcat(hv_CurrentLongitude))).TupleMin( _
                )
            hv_LongitudeMaxOut =
((hv_LongitudeMaxOut.TupleConcat(hv_CurrentLongitude))).TupleMax( _
                )
            hv_LatitudeMinOut =
((hv_LatitudeMinOut.TupleConcat(hv_CurrentLatitude))).TupleMin( _
                )
            hv_LatitudeMaxOut =
((hv_LatitudeMaxOut.TupleConcat(hv_CurrentLatitude))).TupleMax( _
                )
            hv_CamRollMinOut =
((hv_CamRollMinOut.TupleConcat(hv_CurrentCamRoll))).TupleMin( _
                )
            hv_CamRollMaxOut =
((hv_CamRollMaxOut.TupleConcat(hv_CurrentCamRoll))).TupleMax( _
                )
            hv_DistMinOut = ((hv_DistMinOut.TupleConcat(hv_Dist))).TupleMin()
            hv_DistMaxOut = ((hv_DistMaxOut.TupleConcat(hv_Dist))).TupleMax()
        End If
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Increase
MinFaceAngle"))).I() Then
            hv_MinFaceAngleOut = hv_MinFaceAngleOut.TupleAdd((New
HTuple(1)).TupleRad())
```

```
            If New HTuple(hv_MinFaceAngleOut.TupleGreater((New
HTuple(180)).TupleRad())).I() Then
                hv_MinFaceAngleOut = (New HTuple(180)).TupleRad()
            End If
        End If
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Decrease
MinFaceAngle"))).I() Then
            hv_MinFaceAngleOut = hv_MinFaceAngleOut.TupleSub((New
HTuple(1)).TupleRad())
            If New HTuple(hv_MinFaceAngleOut.TupleLess(New HTuple(0))).I() Then
                hv_MinFaceAngleOut = New HTuple(0)
            End If
        End If
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Reset Reference
Pose"))).I() Then
            'Reset the reference pose
            hv_RefPoseOut = hv_RefPoseReset.Clone()
            'Reset the pose range
            hv_LongitudeMinOut = New HTuple(0)
            hv_LongitudeMaxOut = New HTuple(0)
            hv_LatitudeMinOut = New HTuple(0)
            hv_LatitudeMaxOut = New HTuple(0)
            hv_CamRollMinOut = New HTuple(0)
            hv_CamRollMaxOut = New HTuple(0)
            hv_DistMinOut = hv_Dist.Clone()
            hv_DistMaxOut = hv_Dist.Clone()
            'Reset the current relative pose
            hv_RelQuaternionOut = (((New
HTuple(1)).TupleConcat(0)).TupleConcat(0)).TupleConcat( _
                0)
        End If
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Reset Pose
Range"))).I() Then
            hv_LongitudeMinOut = New HTuple(0)
            hv_LongitudeMaxOut = New HTuple(0)
            hv_LatitudeMinOut = New HTuple(0)
            hv_LatitudeMaxOut = New HTuple(0)
            hv_CamRollMinOut = New HTuple(0)
            hv_CamRollMaxOut = New HTuple(0)
            hv_DistMinOut = hv_Dist.Clone()
            hv_DistMaxOut = hv_Dist.Clone()
        End If
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Next Image"))).I()
Then
            hv_ImageNoOut = hv_ImageNoIn.TupleAdd(New HTuple(1))
        End If
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Previous Image"))).I()
Then
            hv_ImageNoOut = hv_ImageNoIn.TupleSub(New HTuple(1))
        End If
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Hidden Line
Removal"))).I() Then
            If New HTuple(hv_HiddenSurfaceRemovalIn.TupleEqual(New
HTuple("true"))).I() Then
                hv_HiddenSurfaceRemovalOut = New HTuple("false")
            Else
                hv_HiddenSurfaceRemovalOut = New HTuple("true")
            End If
        End If
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Mouse Mode: Move
Camera"))).I() Then
            hv_MouseModeOut = New HTuple("move_camera")
        End If
        If New HTuple(hv_SelectedButton.TupleEqual(New HTuple("Mouse Mode: Move
Image"))).I() Then
```

```
            hv_MouseModeOut = New HTuple("move_image")
        End If

        Exit Sub
    End Sub

    ' Main procedure
    Private Sub actionCreateShapeModel()

        ' Local iconic variables
        Dim ho_Image As HObject = Nothing


        ' Local control variables
        Dim hv_CamParam As HTuple = New HTuple, hv_Width As HTuple = New HTuple
        Dim hv_Height As HTuple = New HTuple, hv_WindowHandle As HTuple = New HTuple
        Dim hv_ObjectModel3DID As HTuple = New HTuple, hv_STLStatus As HTuple = New
HTuple
        Dim hv_RefRotX As HTuple = New HTuple, hv_RefRotY As HTuple = New HTuple
        Dim hv_RefRotZ As HTuple = New HTuple, hv_LongitudeMin As HTuple = New HTuple
        Dim hv_LongitudeMax As HTuple = New HTuple, hv_LatitudeMin As HTuple = New
HTuple
        Dim hv_LatitudeMax As HTuple = New HTuple, hv_CamRollMin As HTuple = New
HTuple
        Dim hv_CamRollMax As HTuple = New HTuple, hv_DistMin As HTuple = New HTuple
        Dim hv_DistMax As HTuple = New HTuple, hv_MinFaceAngle As HTuple = New HTuple
        Dim hv_Message As HTuple = New HTuple, hv_S1 As HTuple = New HTuple
        Dim hv_S2 As HTuple = New HTuple
        Dim hv_T As HTuple = New HTuple

        ' Local control variables
        Dim hv_AcqHandle As HTuple = New HTuple

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_Image)



        HOperatorSet.CloseAllFramegrabbers()

        HOperatorSet.OpenFramegrabber(New HTuple("uEye"), New HTuple(1), New
HTuple(1), _
            New HTuple(0), New HTuple(0), New HTuple(0), New HTuple(0), New
HTuple("default"), _
            New HTuple(8), New HTuple("default"), New HTuple(-1), New
HTuple("false"), _
            New HTuple("default"), New HTuple("0"), New HTuple(0), New HTuple(-1),
hv_AcqHandle)
        HOperatorSet.GrabImageStart(hv_AcqHandle, New HTuple(-1))

        Try
            'Read internal camera parameters from file***
            HOperatorSet.ReadCamPar(CamParamDirectory, hv_CamParam)

            '** Creation of Result Window***
            hv_Width = ((hv_CamParam.TupleSelect(New HTuple(6)))).TupleDiv(New
HTuple(4))
            hv_Height = ((hv_CamParam.TupleSelect(New HTuple(7)))).TupleDiv(New
HTuple(4))
            'dev_close_window(...);
            'dev_open_window(...);
            set_display_font(hv_ExpDefaultWinHandle, New HTuple(14), New
HTuple("mono"), _
                New HTuple("true"), New HTuple("false"))
            dev_update_off()
            '** Read the object model from a STL file***
            HOperatorSet.ReadObjectModel3d(ModelDirectory, New HTuple("mm"), _
                (New HTuple(New HTuple("file_type"))).TupleConcat(New
HTuple("invert_normals")), _
                (New HTuple(New HTuple("stl"))).TupleConcat(New HTuple("true")),
hv_ObjectModel3DID, _
                hv_STLStatus)

            '** Prepare the object model for the use within the 3d shape-based
matching***
            HOperatorSet.PrepareObjectModel3d(hv_ObjectModel3DID, New
HTuple("shape_based_matching_3d"), _
                New HTuple("true"), New HTuple(), New HTuple())

            '**Clear Result Window***
            HOperatorSet.ClearWindow(hv_ExpDefaultWinHandle)

            '**Creation Shape Model Procedure***
            '***Initial image is necesarry***
            ho_Image.Dispose()
            HOperatorSet.GrabImage(ho_Image, hv_AcqHandle)


            inspect_object_model_3d(ho_Image, hv_ObjectModel3DID, hv_CamParam,
hv_RefRotX, _
                hv_RefRotY, hv_RefRotZ, hv_LongitudeMin, hv_LongitudeMax,
hv_LatitudeMin, _
                hv_LatitudeMax, hv_CamRollMin, hv_CamRollMax, hv_DistMin, hv_DistMax,
hv_MinFaceAngle)
            HOperatorSet.ClearWindow(hv_ExpDefaultWinHandle)
            HOperatorSet.ClearWindow(hv_ExpDefaultWinMenuHandle)
            HOperatorSet.ClearWindow(hv_ExpDefaultWinBufferHandle)
            HOperatorSet.ClearWindow(hv_ExpDefaultWinViewHandle)

            hv_Message = New HTuple("Creating the 3D shape model (may take several
minutes) ...")
            disp_message(hv_ExpDefaultWinHandle, hv_Message, New HTuple("window"),
New HTuple(12), _
                New HTuple(12), New HTuple("black"), New HTuple("true"))

            HOperatorSet.CreateShapeModel3d(hv_ObjectModel3DID, hv_CamParam,
hv_RefRotX, _
                hv_RefRotY, hv_RefRotZ, New HTuple("gba"), hv_LongitudeMin,
hv_LongitudeMax, _
                hv_LatitudeMin, hv_LatitudeMax, hv_CamRollMin, hv_CamRollMax, New
HTuple(0.95), _
                New HTuple(1.1), New HTuple(10), New HTuple("lowest_model_level"),
New HTuple(3), _
                hv_ShapeModel3DID)
            hv_Message = New HTuple("Creation Completed...")
            disp_message(hv_ExpDefaultWinHandle, hv_Message, New HTuple("window"),
New HTuple(82), _
                New HTuple(12), New HTuple("black"), New HTuple("true"))





            '**Clear 3D Object Model***
            'After creating the 3D shape model we do not longer need
            'the 3D object model.
            HOperatorSet.ClearObjectModel3d(hv_ObjectModel3DID)
            HOperatorSet.CloseFramegrabber(hv_AcqHandle)
            ho_Image.Dispose()

        Catch HDevExpDefaultException As HalconException
            ho_Image.Dispose()
            Throw HDevExpDefaultException
        End Try
        ho_Image.Dispose()

    End Sub

    Private Sub actionSaveShapeModel()
        '**Save 3D Shape Model***
        HOperatorSet.WriteShapeModel3d(hv_ShapeModel3DID, ShapeModelDirectory)
    End Sub



    Sub InitHalcon()
        ' Default settings used in HDevelop
        Call HOperatorSet.SetSystem(New HTuple("width"), New HTuple(512))

        Call HOperatorSet.SetSystem(New HTuple("height"), New HTuple(512))

    End Sub

    Sub RunHalconCreateShapeModel(ByRef WindowMenu As HTuple, ByRef WindowSelect As
HTuple, ByRef WindowView As HTuple, ByRef WindowBuffer As HTuple)
        hv_ExpDefaultWinHandle = WindowSelect
        hv_ExpDefaultWinMenuHandle = WindowMenu
        hv_ExpDefaultWinViewHandle = WindowView
        hv_ExpDefaultWinBufferHandle = WindowBuffer
        Call actionCreateShapeModel()
    End Sub

    Sub RunHalconSaveShapeModel()
        Call actionSaveShapeModel()
    End Sub

End Module
```

A6.3. CylinderMatching.vb

```
Option Strict Off

Option Explicit On

Imports HalconDotNet

Module HDevelopExportCylinderMatching

    Public hv_ExpDefaultWinHandle As HTuple


    Public Sub HDevelopStopMatch()
        Call MessageBox.Show("Piece Matched Succesfully")
    End Sub

    ' Procedures
    ' External procedures
    ' Chapter: Graphics / Text
    ' Short Description: Set font independent of OS
    Public Sub set_display_font(ByVal hv_WindowHandle As HTuple, ByVal hv_Size As
HTuple, _
        ByVal hv_Font As HTuple, ByVal hv_Bold As HTuple, ByVal hv_Slant As HTuple)


        ' Local control variables
        Dim hv_OS As HTuple = New HTuple, hv_Exception As HTuple = New HTuple
        Dim hv_BoldString As HTuple = New HTuple, hv_SlantString As HTuple = New
HTuple
        Dim hv_AllowedFontSizes As HTuple = New HTuple, hv_Distances As HTuple = New
HTuple
        Dim hv_Indices As HTuple = New HTuple, hv_Fonts As HTuple = New HTuple
        Dim hv_FontSelRegexp As HTuple = New HTuple, hv_FontsCourier As HTuple = New
HTuple

        Dim hv_Bold_COPY_INP_TMP As HTuple
        hv_Bold_COPY_INP_TMP = hv_Bold.Clone()
        Dim hv_Font_COPY_INP_TMP As HTuple
        hv_Font_COPY_INP_TMP = hv_Font.Clone()
        Dim hv_Size_COPY_INP_TMP As HTuple
        hv_Size_COPY_INP_TMP = hv_Size.Clone()
        Dim hv_Slant_COPY_INP_TMP As HTuple
        hv_Slant_COPY_INP_TMP = hv_Slant.Clone()

        ' Initialize local and output iconic variables

        'This procedure sets the text font of the current window with
        'the specified attributes.
        'It is assumed that following fonts are installed on the system:
        'Windows: Courier New, Arial Times New Roman
        'Mac OS X: CourierNewPS, Arial, TimesNewRomanPS
        'Linux: courier, helvetica, times
        'Because fonts are displayed smaller on Linux than on Windows,
        'a scaling factor of 1.25 is used the get comparable results.
        'For Linux, only a limited number of font sizes is supported,
        'to get comparable results, it is recommended to use one of the
        'following sizes: 9, 11, 14, 16, 20, 27
        '(which will be mapped internally on Linux systems to 11, 14, 17, 20, 25, 34)
        '
        'Input parameters:
        'WindowHandle: The graphics window for which the font will be set
        'Size: The font size. If Size=-1, the default of 16 is used.
        'Bold: If set to 'true', a bold font is used
        'Slant: If set to 'true', a slanted font is used
        '
        HOperatorSet.GetSystem(New HTuple("operating_system"), hv_OS)
        ' dev_get_preferences(...); only in hdevelop
        ' dev_set_preferences(...); only in hdevelop
        If ((New HTuple(hv_Size_COPY_INP_TMP.TupleEqual(New HTuple())))).TupleOr(New
HTuple( _
            hv_Size_COPY_INP_TMP.TupleEqual(New HTuple(-1)))).I() Then
            hv_Size_COPY_INP_TMP = New HTuple(16)
        End If
        If New HTuple(((hv_OS.TupleSubstr(New HTuple(0), New
HTuple(2)))).TupleEqual(New HTuple("Win"))).I() Then
            'Set font on Windows systems
            If ((((New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("mono"))))).TupleOr( _
                New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("Courier")))))).TupleOr( _
                New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("courier")))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("Courier New")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("sans"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("Arial")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("serif"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("Times New Roman")
            End If
            If New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                hv_Bold_COPY_INP_TMP = New HTuple(1)
            ElseIf New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_Bold_COPY_INP_TMP = New HTuple(0)
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Bold")
                Throw New HalconException(hv_Exception)
            End If
            If New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                hv_Slant_COPY_INP_TMP = New HTuple(1)
            ElseIf New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_Slant_COPY_INP_TMP = New HTuple(0)
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Slant")
                Throw New HalconException(hv_Exception)
            End If
            Try
                HOperatorSet.SetFont(hv_ExpDefaultWinHandle, ((((((((((((((New
HTuple("-")).TupleAdd( _
                    hv_Font_COPY_INP_TMP))).TupleAdd(New HTuple("-
")))).TupleAdd(hv_Size_COPY_INP_TMP))).TupleAdd( _
                    New HTuple("-*-
")))).TupleAdd(hv_Slant_COPY_INP_TMP))).TupleAdd(New HTuple("-*-*-")))).TupleAdd( _
                    hv_Bold_COPY_INP_TMP))).TupleAdd(New HTuple("-")))
                ' catch (Exception)
            Catch HDevExpDefaultException1 As HalconException
                HDevExpDefaultException1.ToHTuple(hv_Exception)
                'throw (Exception)
            End Try
        ElseIf New HTuple(((hv_OS.TupleSubstr(New HTuple(0), New
HTuple(2)))).TupleEqual( _
            New HTuple("Dar"))).I() Then
            'Set font on Mac OS X systems
            If New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                hv_BoldString = New HTuple("Bold")
            ElseIf New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_BoldString = New HTuple("")
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Bold")
                Throw New HalconException(hv_Exception)
            End If
            If New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                hv_SlantString = New HTuple("Italic")
            ElseIf New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_SlantString = New HTuple("")
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Slant")
                Throw New HalconException(hv_Exception)
            End If
            If ((((New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("mono"))))).TupleOr( _
                New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("Courier")))))).TupleOr( _
                New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("courier")))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("CourierNewPS")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("sans"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("Arial")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("serif"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("TimesNewRomanPS")
            End If
            If ((New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New
HTuple("true"))))).TupleOr( _
                New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New HTuple("true")))).I()
Then
                hv_Font_COPY_INP_TMP = ((((hv_Font_COPY_INP_TMP.TupleAdd(New
HTuple("-")))).TupleAdd( _
                    hv_BoldString))).TupleAdd(hv_SlantString)
            End If
            hv_Font_COPY_INP_TMP = hv_Font_COPY_INP_TMP.TupleAdd(New HTuple("MT"))
            Try
                HOperatorSet.SetFont(hv_ExpDefaultWinHandle,
((hv_Font_COPY_INP_TMP.TupleAdd( _
                    New HTuple("-")))).TupleAdd(hv_Size_COPY_INP_TMP))
                ' catch (Exception)
            Catch HDevExpDefaultException1 As HalconException
                HDevExpDefaultException1.ToHTuple(hv_Exception)
                'throw (Exception)
            End Try
        Else
            'Set font for UNIX systems
            hv_Size_COPY_INP_TMP = hv_Size_COPY_INP_TMP.TupleMult(New HTuple(1.25))
            hv_AllowedFontSizes = (((((New
HTuple(11)).TupleConcat(14)).TupleConcat(17)).TupleConcat( _
                20)).TupleConcat(25)).TupleConcat(34)
            If New
HTuple(((hv_AllowedFontSizes.TupleFind(hv_Size_COPY_INP_TMP))).TupleEqual( _
                New HTuple(-1))).I() Then
                hv_Distances =
((hv_AllowedFontSizes.TupleSub(hv_Size_COPY_INP_TMP))).TupleAbs( _
                    )
                HOperatorSet.TupleSortIndex(hv_Distances, hv_Indices)
                hv_Size_COPY_INP_TMP =
hv_AllowedFontSizes.TupleSelect(hv_Indices.TupleSelect( _
                    New HTuple(0)))
            End If
            If ((New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("mono"))))).TupleOr( _
                New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("Courier")))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("courier")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("sans"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("helvetica")
            ElseIf New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("serif"))).I() Then
                hv_Font_COPY_INP_TMP = New HTuple("times")
            End If
            If New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                hv_Bold_COPY_INP_TMP = New HTuple("bold")
            ElseIf New HTuple(hv_Bold_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_Bold_COPY_INP_TMP = New HTuple("medium")
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Bold")
                Throw New HalconException(hv_Exception)
            End If
            If New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New HTuple("true"))).I()
Then
                If New HTuple(hv_Font_COPY_INP_TMP.TupleEqual(New
HTuple("times"))).I() Then
                    hv_Slant_COPY_INP_TMP = New HTuple("i")
                Else
                    hv_Slant_COPY_INP_TMP = New HTuple("o")
                End If
            ElseIf New HTuple(hv_Slant_COPY_INP_TMP.TupleEqual(New
HTuple("false"))).I() Then
                hv_Slant_COPY_INP_TMP = New HTuple("r")
            Else
                hv_Exception = New HTuple("Wrong value of control parameter Slant")
                Throw New HalconException(hv_Exception)
            End If
            Try
                HOperatorSet.SetFont(hv_ExpDefaultWinHandle, (((((((((((((New
HTuple("-adobe-")).TupleAdd( _
                    hv_Font_COPY_INP_TMP))).TupleAdd(New HTuple("-
")))).TupleAdd(hv_Bold_COPY_INP_TMP))).TupleAdd( _
                    New HTuple("-")))).TupleAdd(hv_Slant_COPY_INP_TMP))).TupleAdd(New
HTuple("-normal-*-")))).TupleAdd( _
                    hv_Size_COPY_INP_TMP))).TupleAdd(New HTuple("-*-*-*-*-*-*-*")))
                ' catch (Exception)
            Catch HDevExpDefaultException1 As HalconException
                HDevExpDefaultException1.ToHTuple(hv_Exception)
                If ((New HTuple(((hv_OS.TupleSubstr(New HTuple(0), New
HTuple(4)))).TupleEqual( _
                    New HTuple("Linux"))))).TupleAnd(New
HTuple(hv_Font_COPY_INP_TMP.TupleEqual( _
                    New HTuple("courier")))).I() Then
                    HOperatorSet.QueryFont(hv_ExpDefaultWinHandle, hv_Fonts)
                    hv_FontSelRegexp = ((((New HTuple("^-[^-]*-[^-]*[Cc]ourier[^-]*-
")).TupleAdd( _
                        hv_Bold_COPY_INP_TMP))).TupleAdd(New HTuple("-
")))).TupleAdd(hv_Slant_COPY_INP_TMP)
                    hv_FontsCourier =
((hv_Fonts.TupleRegexpSelect(hv_FontSelRegexp))).TupleRegexpMatch( _
                        hv_FontSelRegexp)
                    If New HTuple(((New
HTuple(hv_FontsCourier.TupleLength()))).TupleEqual(New HTuple(0))).I() Then
                        hv_Exception = New HTuple("Wrong font name")
                        'throw (Exception)
                    Else
                        Try
                            HOperatorSet.SetFont(hv_ExpDefaultWinHandle,
(((((hv_FontsCourier.TupleSelect( _
                                New HTuple(0)))).TupleAdd(New HTuple("-normal-*-
")))).TupleAdd(hv_Size_COPY_INP_TMP))).TupleAdd( _
                                New HTuple("-*-*-*-*-*-*-*")))
                            ' catch (Exception)
                        Catch HDevExpDefaultException2 As HalconException
                            HDevExpDefaultException2.ToHTuple(hv_Exception)
                            'throw (Exception)
                        End Try
                    End If
                End If
                'throw (Exception)
            End Try
        End If
        ' dev_set_preferences(...); only in hdevelop

        Exit Sub
    End Sub

    ' Chapter: Graphics / Output
    ' Short Description: Display the axes of a 3d coordinate system
    Public Sub disp_3d_coord_system(ByVal hv_WindowHandle As HTuple, ByVal 
hv_CamParam As HTuple, _
        ByVal hv_Pose As HTuple, ByVal hv_CoordAxesLength As HTuple)


        ' Local iconic variables
        Dim ho_Arrows As HObject = Nothing


        ' Local control variables
        Dim hv_TransWorld2Cam As HTuple = New HTuple
        Dim hv_OrigCamX As HTuple = New HTuple, hv_OrigCamY As HTuple = New HTuple
        Dim hv_OrigCamZ As HTuple = New HTuple, hv_Row0 As HTuple = New HTuple
        Dim hv_Column0 As HTuple = New HTuple, hv_X As HTuple = New HTuple
        Dim hv_Y As HTuple = New HTuple, hv_Z As HTuple = New HTuple
        Dim hv_RowAxX As HTuple = New HTuple, hv_ColumnAxX As HTuple = New HTuple
        Dim hv_RowAxY As HTuple = New HTuple, hv_ColumnAxY As HTuple = New HTuple
        Dim hv_RowAxZ As HTuple = New HTuple, hv_ColumnAxZ As HTuple = New HTuple
        Dim hv_Distance As HTuple = New HTuple, hv_HeadLength As HTuple = New HTuple
        Dim hv_Red As HTuple = New HTuple, hv_Green As HTuple = New HTuple
        Dim hv_Blue As HTuple = New HTuple

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_Arrows)

        Try
            'This procedure displays a 3D coordinate system.
            'It needs the procedure gen_arrow_contour_xld.
            '
            'Input parameters:
            'WindowHandle: The window where the coordinate system shall be displayed
            'CamParam: The camera paramters
            'Pose: The pose to be displayed
            'CoordAxesLength: The length of the coordinate axes in world coordinates
            '
            'Check, if Pose is a correct pose tuple.
            If New HTuple(((New HTuple(hv_Pose.TupleLength()))).TupleNotEqual(New 
HTuple(7))).I() Then
                ho_Arrows.Dispose()

                Exit Sub
            End If
            '
            'Poses with Z position zero cannot be projected
            '(that would lead to a division by zero error).
            If New HTuple(((hv_Pose.TupleSelect(New HTuple(2)))).TupleEqual(New 
HTuple(0.0))).I() Then
                ho_Arrows.Dispose()

                Exit Sub
            End If
            'Convert to pose to a transformation matrix
            HOperatorSet.PoseToHomMat3d(hv_Pose, hv_TransWorld2Cam)
            'Project the world origin into the image
            HOperatorSet.AffineTransPoint3d(hv_TransWorld2Cam, New HTuple(0), New 
HTuple(0), _
```

```
                New HTuple(0), hv_OrigCamX, hv_OrigCamY, hv_OrigCamZ)
            HOperatorSet.Project3dPoint(hv_OrigCamX, hv_OrigCamY, hv_OrigCamZ,
hv_CamParam, _
                hv_Row0, hv_Column0)
            'Project the coordinate axes into the image
            HOperatorSet.AffineTransPoint3d(hv_TransWorld2Cam, hv_CoordAxesLength,
New HTuple(0), _
                New HTuple(0), hv_X, hv_Y, hv_Z)
            HOperatorSet.Project3dPoint(hv_X, hv_Y, hv_Z, hv_CamParam, hv_RowAxX,
hv_ColumnAxX)
            HOperatorSet.AffineTransPoint3d(hv_TransWorld2Cam, New HTuple(0),
hv_CoordAxesLength, _
                New HTuple(0), hv_X, hv_Y, hv_Z)
            HOperatorSet.Project3dPoint(hv_X, hv_Y, hv_Z, hv_CamParam, hv_RowAxY,
hv_ColumnAxY)
            HOperatorSet.AffineTransPoint3d(hv_TransWorld2Cam, New HTuple(0), New
HTuple(0), _
                hv_CoordAxesLength, hv_X, hv_Y, hv_Z)
            HOperatorSet.Project3dPoint(hv_X, hv_Y, hv_Z, hv_CamParam, hv_RowAxZ,
hv_ColumnAxZ)
            '
            'Generate an XLD contour for each axis

HOperatorSet.DistancePp(((hv_Row0.TupleConcat(hv_Row0))).TupleConcat(hv_Row0), _
                ((hv_Column0.TupleConcat(hv_Column0))).TupleConcat(hv_Column0),
((hv_RowAxX.TupleConcat( _
                hv_RowAxY))).TupleConcat(hv_RowAxZ),
((hv_ColumnAxX.TupleConcat(hv_ColumnAxY))).TupleConcat( _
                hv_ColumnAxZ), hv_Distance)
            hv_HeadLength = (((((((hv_Distance.TupleMax())).TupleDiv(New
HTuple(12.0)))).TupleConcat( _
                New HTuple(5.0)))).TupleMax())).TupleInt()
            ho_Arrows.Dispose()
            gen_arrow_contour_xld(ho_Arrows,
((hv_Row0.TupleConcat(hv_Row0))).TupleConcat( _
                hv_Row0),
((hv_Column0.TupleConcat(hv_Column0))).TupleConcat(hv_Column0), _
                ((hv_RowAxX.TupleConcat(hv_RowAxY))).TupleConcat(hv_RowAxZ),
((hv_ColumnAxX.TupleConcat( _
                hv_ColumnAxY))).TupleConcat(hv_ColumnAxZ), hv_HeadLength,
hv_HeadLength)
            '
            'Display coordinate system
            HOperatorSet.DispXld(ho_Arrows, hv_ExpDefaultWinHandle)
            '
            HOperatorSet.GetRgb(hv_ExpDefaultWinHandle, hv_Red, hv_Green, hv_Blue)
            HOperatorSet.SetRgb(hv_ExpDefaultWinHandle, hv_Red.TupleSelect(New
HTuple(0)), _
                hv_Green.TupleSelect(New HTuple(0)), hv_Blue.TupleSelect(New
HTuple(0)))
            HOperatorSet.SetTposition(hv_ExpDefaultWinHandle, hv_RowAxX.TupleAdd(New
HTuple(3)), _
                hv_ColumnAxX.TupleAdd(New HTuple(3)))
            HOperatorSet.WriteString(hv_ExpDefaultWinHandle, New HTuple("X"))
            HOperatorSet.SetRgb(hv_ExpDefaultWinHandle, hv_Red.TupleSelect((New
HTuple(1)).TupleMod( _
                New HTuple(hv_Red.TupleLength()))), hv_Green.TupleSelect((New
HTuple(1)).TupleMod( _
                New HTuple(hv_Green.TupleLength()))), hv_Blue.TupleSelect((New
HTuple(1)).TupleMod( _
                New HTuple(hv_Blue.TupleLength()))))
            HOperatorSet.SetTposition(hv_ExpDefaultWinHandle, hv_RowAxY.TupleAdd(New
HTuple(3)), _
                hv_ColumnAxY.TupleAdd(New HTuple(3)))
            HOperatorSet.WriteString(hv_ExpDefaultWinHandle, New HTuple("Y"))
            HOperatorSet.SetRgb(hv_ExpDefaultWinHandle, hv_Red.TupleSelect((New
HTuple(2)).TupleMod( _
                New HTuple(hv_Red.TupleLength()))), hv_Green.TupleSelect((New
HTuple(2)).TupleMod( _
                New HTuple(hv_Green.TupleLength()))), hv_Blue.TupleSelect((New
HTuple(2)).TupleMod( _
                New HTuple(hv_Blue.TupleLength()))))
            HOperatorSet.SetTposition(hv_ExpDefaultWinHandle, hv_RowAxZ.TupleAdd(New
HTuple(3)), _
                hv_ColumnAxZ.TupleAdd(New HTuple(3)))
            HOperatorSet.WriteString(hv_ExpDefaultWinHandle, New HTuple("Z"))
            HOperatorSet.SetRgb(hv_ExpDefaultWinHandle, hv_Red, hv_Green, hv_Blue)
            ho_Arrows.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_Arrows.Dispose()

            Throw HDevExpDefaultException
        End Try
    End Sub

    ' Chapter: Develop
    ' Short Description: Switch dev_update_pc, dev_update_var and dev_update_window
to 'off'.
    Public Sub dev_update_off()
        ' Initialize local and output iconic variables

        'This procedure sets different update settings to 'off'.
        'This is useful to get the best performance and reduce overhead.
        '
        ' dev_update_pc(...); only in hdevelop
        ' dev_update_var(...); only in hdevelop
        ' dev_update_window(...); only in hdevelop

        Exit Sub
    End Sub

    ' Chapter: Graphics / Text
    ' Short Description: This procedure displays 'Click 'Run' to continue' in the
lower right corner of the screen.
    Public Sub disp_continue_message(ByVal hv_WindowHandle As HTuple, ByVal hv_Color
As HTuple, _
        ByVal hv_Box As HTuple)


        ' Local control variables
        Dim hv_ContinueMessage As HTuple = New HTuple
        Dim hv_Row As HTuple = New HTuple, hv_Column As HTuple = New HTuple
        Dim hv_Width As HTuple = New HTuple, hv_Height As HTuple = New HTuple
        Dim hv_Ascent As HTuple = New HTuple, hv_Descent As HTuple = New HTuple
        Dim hv_TextWidth As HTuple = New HTuple, hv_TextHeight As HTuple = New HTuple

        ' Initialize local and output iconic variables

        'This procedure displays 'Press Run (F5) to continue' in the
        'lower right corner of the screen.
        'It uses the procedure disp_message.
        '
        'Input parameters:
        'WindowHandle: The window, where the text shall be displayed
        'Color: defines the text color.
        '   If set to '' or 'auto', the currently set color is used.
        'Box: If set to 'true', the text is displayed in a box.
        '
        hv_ContinueMessage = New HTuple("Press Run (F5) to continue")
        HOperatorSet.GetWindowExtents(hv_ExpDefaultWinHandle, hv_Row, hv_Column, 
hv_Width, _
            hv_Height)
        HOperatorSet.GetStringExtents(hv_ExpDefaultWinHandle, (((New HTuple(" 
")).TupleAdd( _
            hv_ContinueMessage))).TupleAdd(New HTuple(" ")), hv_Ascent, hv_Descent, 
hv_TextWidth, _
            hv_TextHeight)
        disp_message(hv_ExpDefaultWinHandle, hv_ContinueMessage, New 
HTuple("window"), _
            ((hv_Height.TupleSub(hv_TextHeight))).TupleSub(New HTuple(12)), 
((hv_Width.TupleSub( _
            hv_TextWidth))).TupleSub(New HTuple(12)), hv_Color, hv_Box)

        Exit Sub
    End Sub

    ' Chapter: Graphics / Text
    ' Short Description: This procedure writes a text message.
    Public Sub disp_message(ByVal hv_WindowHandle As HTuple, ByVal hv_String As 
HTuple, _
        ByVal hv_CoordSystem As HTuple, ByVal hv_Row As HTuple, ByVal hv_Column As 
HTuple, _
        ByVal hv_Color As HTuple, ByVal hv_Box As HTuple)


        ' Local control variables
        Dim hv_Red As HTuple = New HTuple, hv_Green As HTuple = New HTuple
        Dim hv_Blue As HTuple = New HTuple, hv_Row1Part As HTuple = New HTuple
        Dim hv_Column1Part As HTuple = New HTuple, hv_Row2Part As HTuple = New HTuple
        Dim hv_Column2Part As HTuple = New HTuple, hv_RowWin As HTuple = New HTuple
        Dim hv_ColumnWin As HTuple = New HTuple, hv_WidthWin As HTuple = New HTuple
        Dim hv_HeightWin As HTuple = New HTuple, hv_MaxAscent As HTuple = New HTuple
        Dim hv_MaxDescent As HTuple = New HTuple, hv_MaxWidth As HTuple = New HTuple
        Dim hv_MaxHeight As HTuple = New HTuple, hv_R1 As HTuple = New HTuple
        Dim hv_C1 As HTuple = New HTuple, hv_FactorRow As HTuple = New HTuple
        Dim hv_FactorColumn As HTuple = New HTuple, hv_Width As HTuple = New HTuple
        Dim hv_Index As HTuple = New HTuple, hv_Ascent As HTuple = New HTuple
        Dim hv_Descent As HTuple = New HTuple, hv_W As HTuple = New HTuple
        Dim hv_H As HTuple = New HTuple, hv_FrameHeight As HTuple = New HTuple
        Dim hv_FrameWidth As HTuple = New HTuple, hv_R2 As HTuple = New HTuple
        Dim hv_C2 As HTuple = New HTuple, hv_DrawMode As HTuple = New HTuple
        Dim hv_Exception As HTuple = New HTuple, hv_CurrentColor As HTuple = New 
HTuple

        Dim hv_Color_COPY_INP_TMP As HTuple
        hv_Color_COPY_INP_TMP = hv_Color.Clone()
        Dim hv_Column_COPY_INP_TMP As HTuple
        hv_Column_COPY_INP_TMP = hv_Column.Clone()
        Dim hv_Row_COPY_INP_TMP As HTuple
        hv_Row_COPY_INP_TMP = hv_Row.Clone()
        Dim hv_String_COPY_INP_TMP As HTuple
        hv_String_COPY_INP_TMP = hv_String.Clone()

        ' Initialize local and output iconic variables

        'This procedure displays text in a graphics window.
        '
        'Input parameters:
        'WindowHandle: The WindowHandle of the graphics window, where
        '   the message should be displayed
        'String: A tuple of strings containing the text message to be displayed
        'CoordSystem: If set to 'window', the text position is given
        '   with respect to the window coordinate system.
        '   If set to 'image', image coordinates are used.
        '   (This may be useful in zoomed images.)
        'Row: The row coordinate of the desired text position
        '   If set to -1, a default value of 12 is used.
        'Column: The column coordinate of the desired text position
        '   If set to -1, a default value of 12 is used.
        'Color: defines the color of the text as string.
        '   If set to [], '' or 'auto' the currently set color is used.
        '   If a tuple of strings is passed, the colors are used cyclically
        '   for each new textline.
        'Box: If set to 'true', the text is written within a white box.
        '
        'prepare window
        HOperatorSet.GetRgb(hv_ExpDefaultWinHandle, hv_Red, hv_Green, hv_Blue)
        HOperatorSet.GetPart(hv_ExpDefaultWinHandle, hv_Row1Part, hv_Column1Part,
hv_Row2Part, _
            hv_Column2Part)
        HOperatorSet.GetWindowExtents(hv_ExpDefaultWinHandle, hv_RowWin,
hv_ColumnWin, _
            hv_WidthWin, hv_HeightWin)
        HOperatorSet.SetPart(hv_ExpDefaultWinHandle, New HTuple(0), New HTuple(0),
hv_HeightWin.TupleSub( _
            New HTuple(1)), hv_WidthWin.TupleSub(New HTuple(1)))
        '
        'default settings
        If New HTuple(hv_Row_COPY_INP_TMP.TupleEqual(New HTuple(-1))).I() Then
            hv_Row_COPY_INP_TMP = New HTuple(12)
        End If
        If New HTuple(hv_Column_COPY_INP_TMP.TupleEqual(New HTuple(-1))).I() Then
            hv_Column_COPY_INP_TMP = New HTuple(12)
        End If
        If New HTuple(hv_Color_COPY_INP_TMP.TupleEqual(New HTuple())).I() Then
            hv_Color_COPY_INP_TMP = New HTuple("")
        End If
        '
        hv_String_COPY_INP_TMP = (((((New
HTuple("")).TupleAdd(hv_String_COPY_INP_TMP))).TupleAdd( _
            New HTuple("")))).TupleSplit(New HTuple("" + Chr(10)))
        '
        'Estimate extentions of text depending on font size.
        HOperatorSet.GetFontExtents(hv_ExpDefaultWinHandle, hv_MaxAscent,
hv_MaxDescent, _
            hv_MaxWidth, hv_MaxHeight)
        If New HTuple(hv_CoordSystem.TupleEqual(New HTuple("window"))).I() Then
            hv_R1 = hv_Row_COPY_INP_TMP.Clone()
            hv_C1 = hv_Column_COPY_INP_TMP.Clone()
        Else
            'transform image to window coordinates
            hv_FactorRow = (((New
HTuple(1.0)).TupleMult(hv_HeightWin))).TupleDiv(((hv_Row2Part.TupleSub( _
                hv_Row1Part))).TupleAdd(New HTuple(1)))
            hv_FactorColumn = (((New
HTuple(1.0)).TupleMult(hv_WidthWin))).TupleDiv(((hv_Column2Part.TupleSub( _
                hv_Column1Part))).TupleAdd(New HTuple(1)))
            hv_R1 = ((((hv_Row_COPY_INP_TMP.TupleSub(hv_Row1Part))).TupleAdd(New
HTuple(0.5)))).TupleMult( _
                hv_FactorRow)
            hv_C1 =
((((hv_Column_COPY_INP_TMP.TupleSub(hv_Column1Part))).TupleAdd(New
HTuple(0.5)))).TupleMult( _
                hv_FactorColumn)
        End If
        '
        'display text box depending on text size
        If New HTuple(hv_Box.TupleEqual(New HTuple("true"))).I() Then
```

```
            'calculate box extents
            hv_String_COPY_INP_TMP = (((New HTuple("
")).TupleAdd(hv_String_COPY_INP_TMP))).TupleAdd( _
                New HTuple(" "))
            hv_Width = New HTuple()
            For hv_Index = (New HTuple(0)) To ((New
HTuple(hv_String_COPY_INP_TMP.TupleLength( _
                )))).TupleSub(New HTuple(1)) Step (New HTuple(1))
                HOperatorSet.GetStringExtents(hv_ExpDefaultWinHandle,
hv_String_COPY_INP_TMP.TupleSelect( _
                    hv_Index), hv_Ascent, hv_Descent, hv_W, hv_H)
                hv_Width = hv_Width.TupleConcat(hv_W)
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            hv_FrameHeight = hv_MaxHeight.TupleMult(New
HTuple(hv_String_COPY_INP_TMP.TupleLength( _
                )))
            hv_FrameWidth = (((New HTuple(0)).TupleConcat(hv_Width))).TupleMax()
            hv_R2 = hv_R1.TupleAdd(hv_FrameHeight)
            hv_C2 = hv_C1.TupleAdd(hv_FrameWidth)
            'display rectangles
            HOperatorSet.GetDraw(hv_ExpDefaultWinHandle, hv_DrawMode)
            HOperatorSet.SetDraw(hv_ExpDefaultWinHandle, New HTuple("fill"))
            HOperatorSet.SetColor(hv_ExpDefaultWinHandle, New HTuple("light gray"))
            HOperatorSet.DispRectangle1(hv_ExpDefaultWinHandle, hv_R1.TupleAdd(New
HTuple(3)), _
                hv_C1.TupleAdd(New HTuple(3)), hv_R2.TupleAdd(New HTuple(3)),
hv_C2.TupleAdd( _
                New HTuple(3)))
            HOperatorSet.SetColor(hv_ExpDefaultWinHandle, New HTuple("white"))
            HOperatorSet.DispRectangle1(hv_ExpDefaultWinHandle, hv_R1, hv_C1, hv_R2,
hv_C2)
            HOperatorSet.SetDraw(hv_ExpDefaultWinHandle, hv_DrawMode)
        ElseIf New HTuple(hv_Box.TupleNotEqual(New HTuple("false"))).I() Then
            hv_Exception = New HTuple("Wrong value of control parameter Box")
            Throw New HalconException(hv_Exception)
        End If
        'Write text.
        For hv_Index = (New HTuple(0)) To ( _
            (New HTuple(hv_String_COPY_INP_TMP.TupleLength()))).TupleSub(New
HTuple(1)) Step (New HTuple(1))
            hv_CurrentColor = hv_Color_COPY_INP_TMP.TupleSelect(hv_Index.TupleMod(New
HTuple( _
                hv_Color_COPY_INP_TMP.TupleLength())))
            If ((New HTuple(hv_CurrentColor.TupleNotEqual(New
HTuple(""))))).TupleAnd(New HTuple( _
                hv_CurrentColor.TupleNotEqual(New HTuple("auto")))).I() Then
                HOperatorSet.SetColor(hv_ExpDefaultWinHandle, hv_CurrentColor)
            Else
                HOperatorSet.SetRgb(hv_ExpDefaultWinHandle, hv_Red, hv_Green,
hv_Blue)
            End If
            hv_Row_COPY_INP_TMP = hv_R1.TupleAdd(hv_MaxHeight.TupleMult(hv_Index))
            HOperatorSet.SetTposition(hv_ExpDefaultWinHandle, hv_Row_COPY_INP_TMP,
hv_C1)
            HOperatorSet.WriteString(hv_ExpDefaultWinHandle,
hv_String_COPY_INP_TMP.TupleSelect( _
                hv_Index))
#If USE_DO_EVENTS Then
    ' Please note: The call of DoEvents() is only a hack to
    ' enable VB to react on events. Please change the code
    ' so that it can handle events in a standard way.
    System.Windows.Forms.Application.DoEvents()
#End If
        Next
        'reset changed window settings
        HOperatorSet.SetRgb(hv_ExpDefaultWinHandle, hv_Red, hv_Green, hv_Blue)
        HOperatorSet.SetPart(hv_ExpDefaultWinHandle, hv_Row1Part, hv_Column1Part,
hv_Row2Part, _
            hv_Column2Part)

        Exit Sub
    End Sub

    ' Chapter: XLD / Creation
    ' Short Description: Creates an arrow shaped XLD contour.
    Public Sub gen_arrow_contour_xld(ByRef ho_Arrow As HObject, ByVal hv_Row1 As
HTuple, _
        ByVal hv_Column1 As HTuple, ByVal hv_Row2 As HTuple, ByVal hv_Column2 As
HTuple, _
        ByVal hv_HeadLength As HTuple, ByVal hv_HeadWidth As HTuple)


        ' Stack for temporary objects
        Dim OTemp(10) As HObject

        ' Local iconic variables
        Dim ho_TempArrow As HObject = Nothing


        ' Local control variables
        Dim hv_Length As HTuple = New HTuple, hv_ZeroLengthIndices As HTuple = New
HTuple
        Dim hv_DR As HTuple = New HTuple, hv_DC As HTuple = New HTuple
        Dim hv_HalfHeadWidth As HTuple = New HTuple, hv_RowP1 As HTuple = New HTuple
        Dim hv_ColP1 As HTuple = New HTuple, hv_RowP2 As HTuple = New HTuple
        Dim hv_ColP2 As HTuple = New HTuple, hv_Index As HTuple = New HTuple

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_Arrow)
        HOperatorSet.GenEmptyObj(ho_TempArrow)

        Try
            'This procedure generates arrow shaped XLD contours,
            'pointing from (Row1, Column1) to (Row2, Column2).
            'If starting and end point are identical, a contour consisting
            'of a single point is returned.
            '
            'input parameteres:
            'Row1, Column1: Coordinates of the arrows' starting points
            'Row2, Column2: Coordinates of the arrows' end points
            'HeadLength, HeadWidth: Size of the arrow heads in pixels
            '
            'output parameter:
            'Arrow: The resulting XLD contour
            '
            'The input tuples Row1, Column1, Row2, and Column2 have to be of
            'the same length.
            'HeadLength and HeadWidth either have to be of the same length as
            'Row1, Column1, Row2, and Column2 or have to be a single element.
            'If one of the above restrictions is violated, an error will occur.
            '
            '
            'Init
            ho_Arrow.Dispose()
            HOperatorSet.GenEmptyObj(ho_Arrow)
            '
            'Calculate the arrow length
            HOperatorSet.DistancePp(hv_Row1, hv_Column1, hv_Row2, hv_Column2,
hv_Length)
            '
            'Mark arrows with identical start and end point
            '(set Length to -1 to avoid division-by-zero exception)
            hv_ZeroLengthIndices = hv_Length.TupleFind(New HTuple(0))
            If New HTuple(hv_ZeroLengthIndices.TupleNotEqual(New HTuple(-1))).I()
Then
                If IsNothing(hv_Length) Then
                    hv_Length = New HTuple
                End If
                hv_Length(hv_ZeroLengthIndices) = New HTuple(-1)
            End If
            '
            'Calculate auxiliary variables.
            hv_DR = (((New
HTuple(1.0)).TupleMult(hv_Row2.TupleSub(hv_Row1)))).TupleDiv(hv_Length)
            hv_DC = (((New
HTuple(1.0)).TupleMult(hv_Column2.TupleSub(hv_Column1)))).TupleDiv( _
                hv_Length)
            hv_HalfHeadWidth = hv_HeadWidth.TupleDiv(New HTuple(2.0))
            '
            'Calculate end points of the arrow head.
            hv_RowP1 =
((hv_Row1.TupleAdd(((hv_Length.TupleSub(hv_HeadLength))).TupleMult( _
                hv_DR)))).TupleAdd(hv_HalfHeadWidth.TupleMult(hv_DC))
            hv_ColP1 =
((hv_Column1.TupleAdd(((hv_Length.TupleSub(hv_HeadLength))).TupleMult( _
                hv_DC)))).TupleSub(hv_HalfHeadWidth.TupleMult(hv_DR))
            hv_RowP2 =
((hv_Row1.TupleAdd(((hv_Length.TupleSub(hv_HeadLength))).TupleMult( _
                hv_DR)))).TupleSub(hv_HalfHeadWidth.TupleMult(hv_DC))
            hv_ColP2 =
((hv_Column1.TupleAdd(((hv_Length.TupleSub(hv_HeadLength))).TupleMult( _
                hv_DC)))).TupleAdd(hv_HalfHeadWidth.TupleMult(hv_DR))
            '
            'Finally create output XLD contour for each input point pair
            For hv_Index = (New HTuple(0)) To ( _
                (New HTuple(hv_Length.TupleLength()))).TupleSub(New HTuple(1)) Step
(New HTuple(1))
                If New HTuple(((hv_Length.TupleSelect(hv_Index))).TupleEqual(New
HTuple(-1))).I() Then
                    'Create_ single points for arrows with identical start and end
point
                    ho_TempArrow.Dispose()
                    HOperatorSet.GenContourPolygonXld(ho_TempArrow,
hv_Row1.TupleSelect(hv_Index), _
                        hv_Column1.TupleSelect(hv_Index))
                Else
                    'Create arrow contour
                    ho_TempArrow.Dispose()
                    HOperatorSet.GenContourPolygonXld(ho_TempArrow,
((((((((((hv_Row1.TupleSelect( _

hv_Index))).TupleConcat(hv_Row2.TupleSelect(hv_Index)))).TupleConcat( _

hv_RowP1.TupleSelect(hv_Index)))).TupleConcat(hv_Row2.TupleSelect(hv_Index)))).TupleC
oncat( _

hv_RowP2.TupleSelect(hv_Index)))).TupleConcat(hv_Row2.TupleSelect(hv_Index)), _

((((((((((hv_Column1.TupleSelect(hv_Index))).TupleConcat(hv_Column2.TupleSelect( _
hv_Index)))).TupleConcat(hv_ColP1.TupleSelect(hv_Index)))).TupleConcat( _

hv_Column2.TupleSelect(hv_Index)))).TupleConcat(hv_ColP2.TupleSelect( _
                        hv_Index)))).TupleConcat(hv_Column2.TupleSelect(hv_Index)))
                End If
                HOperatorSet.ConcatObj(ho_Arrow, ho_TempArrow, OTemp(0))
                ho_Arrow.Dispose()
                ho_Arrow = OTemp(0)
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            ho_TempArrow.Dispose()

            Exit Sub
        Catch HDevExpDefaultException As HalconException
            ho_TempArrow.Dispose()

            Throw HDevExpDefaultException
        End Try
    End Sub

    ' Main procedure
    Private Sub actionMatch()

        ' Local iconic variables
        Dim ho_ImageMatching As HObject = Nothing, ho_ModelContours As HObject =
Nothing


        ' Local control variables
        Dim hv_AcqHandle As HTuple = New HTuple, hv_CamParam As HTuple = New HTuple
        Dim hv_Width As HTuple = New HTuple, hv_Height As HTuple = New HTuple
        Dim hv_WindowHandle As HTuple = New HTuple
        Dim hv_Times As HTuple = New HTuple, hv_Seconds1 As HTuple = New HTuple
        Dim hv_Pose As HTuple = New HTuple, hv_CovPose As HTuple = New HTuple
        Dim hv_Score As HTuple = New HTuple, hv_Seconds2 As HTuple = New HTuple
        Dim hv_Time As HTuple = New HTuple, hv_I As HTuple = New HTuple
        Dim hv_PoseTmp As HTuple = New HTuple, hv_obj_H_cam As HTuple = New HTuple
        Dim hv_HomMat3DIdentity As HTuple = New HTuple, hv_HomMat3D_T As HTuple = New
HTuple
        Dim hv_HomMat3D_T_R As HTuple = New HTuple

        ' Initialize local and output iconic variables
        HOperatorSet.GenEmptyObj(ho_ImageMatching)
        HOperatorSet.GenEmptyObj(ho_ModelContours)

        Try
            '*****************************************
            '** SHAPE BASED MATCHING WITH STL MODEL***
            '*****************************************

            dev_update_off()
            ' dev_get_preferences(...); only in hdevelop
            ' dev_set_preferences(...); only in hdevelop
            '

            '** Close Any framegrabber that could remain open***
            HOperatorSet.CloseAllFramegrabbers()
            HOperatorSet.OpenFramegrabber(New HTuple("uEye"), New HTuple(1), New
HTuple(1), _
                New HTuple(0), New HTuple(0), New HTuple(0), New HTuple(0), New
HTuple("default"), _
                New HTuple(-1), New HTuple("default"), New HTuple(-1), New
HTuple("default"), _
                New HTuple("default"), New HTuple("default"), New HTuple(-1), New
HTuple(-1), _
                hv_AcqHandle)


            'Read internal camera parameters from file***
            HOperatorSet.ReadCamPar(CamParamDirectory, hv_CamParam)
            'Clear Window'
            HOperatorSet.ClearWindow(hv_ExpDefaultWinHandle)

            '** Creation of Result Window***
            hv_Width = ((hv_CamParam.TupleSelect(New HTuple(6)))).TupleDiv(New
HTuple(4))
            hv_Height = ((hv_CamParam.TupleSelect(New HTuple(7)))).TupleDiv(New
HTuple(4))
            'dev_open_window(...);
            set_display_font(hv_ExpDefaultWinHandle, New HTuple(14), New
HTuple("mono"), _
                New HTuple("true"), New HTuple("false"))
            dev_update_off()

            If ShapeRead = False Then
                '** Read 3D Shape Model***
                disp_message(hv_ExpDefaultWinHandle, New HTuple("Reading the 3D shape
model file from disk ..."), _
                    New HTuple("window"), New HTuple(12), New HTuple(12), New
HTuple("black"), _
                    New HTuple("true"))
                HOperatorSet.ReadShapeModel3d(ShapeModelDirectory, hv_ShapeModel3DID)
                disp_message(hv_ExpDefaultWinHandle, New HTuple("Reading
Succesful..."), New HTuple("window"), _
                    New HTuple(62), New HTuple(12), New HTuple("black"), New
HTuple("true"))
                ShapeRead = True
            End If

            '**Find instances of a 3D shape model in some run-time images***
            hv_Times = New HTuple()
            ho_ImageMatching.Dispose()
            HOperatorSet.GrabImage(ho_ImageMatching, hv_AcqHandle)
            HOperatorSet.DispObj(ho_ImageMatching, hv_ExpDefaultWinHandle)
            HOperatorSet.CountSeconds(hv_Seconds1)
            'Find  instances of the 3D shape model
            HOperatorSet.FindShapeModel3d(ho_ImageMatching, hv_ShapeModel3DID, New
HTuple(0.4), New HTuple(0), _
                New HTuple(10), (New HTuple(New
HTuple("num_matches"))).TupleConcat(New HTuple("border_model")), _
                (New HTuple(1)).TupleConcat(New HTuple("true")), hv_Pose, hv_CovPose,
hv_Score)
            HOperatorSet.CountSeconds(hv_Seconds2)
            hv_Time = hv_Seconds2.TupleSub(hv_Seconds1)
            hv_Times = hv_Times.TupleConcat(hv_Time)
            'Visualize the found matches in the image by projecting
            'the 3D shape model into it using the pose of the match

            For hv_I = (New HTuple(0)) To ( _
                (New HTuple(hv_Score.TupleLength()))).TupleSub(New HTuple(1)) Step
(New HTuple(1))
                'Display Contours
                hv_PoseTmp = hv_Pose.TupleSelectRange(hv_I.TupleMult(New HTuple(7)),
((hv_I.TupleMult( _
                    New HTuple(7)))).TupleAdd(New HTuple(6)))
                'homogeneus transformation correct alignment of the pose
                HOperatorSet.PoseToHomMat3d(hv_PoseTmp, hv_obj_H_cam)
                HOperatorSet.HomMat3dIdentity(hv_HomMat3DIdentity)
                HOperatorSet.HomMat3dTranslateLocal(hv_HomMat3DIdentity, New
HTuple(0.024), _
                    New HTuple(0.048), New HTuple(0.024), hv_HomMat3D_T)
                HOperatorSet.HomMat3dRotateLocal(hv_HomMat3D_T, New
HTuple(1.570796327), New HTuple("x"), _
                    hv_HomMat3D_T_R)
                HOperatorSet.HomMat3dCompose(hv_obj_H_cam, hv_HomMat3D_T_R,
hv_obj_H_cam)

                HOperatorSet.HomMat3dToPose(hv_obj_H_cam, hv_PoseTmp1)

                'Num of pieces detected
                hv_NumPieces = hv_Score.Length
                'Show Pose information in window
                hv_PosX = hv_PoseTmp1.TupleSelect(New HTuple(0)) * 1000
                hv_PosY = hv_PoseTmp1.TupleSelect(New HTuple(1)) * 1000
                hv_PosZ = hv_PoseTmp1.TupleSelect(New HTuple(2)) * 1000
                hv_RotX = hv_PoseTmp1.TupleSelect(New HTuple(3))
                hv_RotY = hv_PoseTmp1.TupleSelect(New HTuple(4))
                hv_RotZ = hv_PoseTmp1.TupleSelect(New HTuple(5))
                HOperatorSet.ProjectShapeModel3d(ho_ModelContours, hv_ShapeModel3DID,
hv_CamParam, _
                    hv_PoseTmp, New HTuple("true"), New HTuple(0))
                HOperatorSet.SetColor(hv_ExpDefaultWinHandle, New HTuple("green"))
                HOperatorSet.DispObj(ho_ModelContours, hv_ExpDefaultWinHandle)
                'Display the coordinate system of the 3D object model
                HOperatorSet.SetColor(hv_ExpDefaultWinHandle, New HTuple("red"))
                disp_3d_coord_system(hv_ExpDefaultWinHandle, hv_CamParam,
hv_PoseTmp1, New HTuple(0.015))

                'Display the parameters of the found pose
#If USE_DO_EVENTS Then
      ' Please note: The call of DoEvents() is only a hack to
      ' enable VB to react on events. Please change the code
      ' so that it can handle events in a standard way.
      System.Windows.Forms.Application.DoEvents()
#End If
            Next
            disp_message(hv_ExpDefaultWinHandle, ((((((New
HTuple(hv_Score.TupleLength())))).TupleAdd( _
                New HTuple(" Match(es) found in
")))).TupleAdd(hv_Time.TupleString((New HTuple("4.2f")).S())))).TupleAdd( _
                New HTuple(" s")), New HTuple("window"), New HTuple(12), New
HTuple(12), _
                New HTuple("black"), New HTuple("true"))

            HDevelopStopMatch()

        Catch HDevExpDefaultException As HalconException
            ho_ImageMatching.Dispose()
            ho_ModelContours.Dispose()

            Throw HDevExpDefaultException
        End Try
        ho_ImageMatching.Dispose()
        ho_ModelContours.Dispose()

        HOperatorSet.CloseFramegrabber(hv_AcqHandle)
```

```
    End Sub



    Sub InitHalcon()
        ' Default settings used in HDevelop
        Call HOperatorSet.SetSystem(New HTuple("width"), New HTuple(512))

        Call HOperatorSet.SetSystem(New HTuple("height"), New HTuple(512))
    End Sub

    Sub FinishHalcon()
        If ShapeRead = True Then
            HOperatorSet.ClearShapeModel3d(hv_ShapeModel3DID)
        End If

        End
    End Sub

    Sub RunHalconMatch(ByRef Window As HTuple)
        hv_ExpDefaultWinHandle = Window
        Call actionMatch()
    End Sub

End Module
```

A6.4. ConvertPose.vb

```
Option Strict Off

Option Explicit On

Imports HalconDotNet

Module HDevelopExportConvertPose

    Public hv_ExpDefaultWinHandle As HTuple
    ' Main procedure
    Private Sub actionConvert()

        ' Local control variables
        Dim hv_PoseBaseinCam As HTuple = New HTuple
        Dim hv_Cam_H_RobBase As HTuple = New HTuple, hv_RobBase_H_Cam As HTuple = New
HTuple
        Dim hv_Cam_H_Cylinder As HTuple = New HTuple, hv_RobBase_H_Cylinder As HTuple
= New HTuple


        ' Initialize local and output iconic variables

        'Read Poses from the calibration and the matching
        HOperatorSet.ReadPose(CamRobDirectory, hv_PoseBaseinCam)
        HOperatorSet.PoseToHomMat3d(hv_PoseBaseinCam, hv_Cam_H_RobBase)

        HOperatorSet.PoseToHomMat3d(hv_PoseTmp1, hv_Cam_H_Cylinder)
        'Do transformation to obtain coordinate of the cylinder in robot base
coordinate system
        'RobBase_H_Cylinder= RobBase_H_Cam * Cam_H_Cylinder
        HOperatorSet.HomMat3dInvert(hv_Cam_H_RobBase, hv_RobBase_H_Cam)
        HOperatorSet.HomMat3dCompose(hv_RobBase_H_Cam, hv_Cam_H_Cylinder,
hv_RobBase_H_Cylinder)
```

```
        HOperatorSet.HomMat3dToPose(hv_RobBase_H_Cylinder,
hv_Pose_Cylinder_in_RobBase)
        If IsNothing(hv_Pose_Cylinder_in_RobBase) Then
            hv_Pose_Cylinder_in_RobBase = New HTuple
        End If
        hv_Pose_Cylinder_in_RobBase(New HTuple(0)) =
((hv_Pose_Cylinder_in_RobBase.TupleSelect( _
            New HTuple(0)))).TupleSub(New HTuple(0.0045))
        If IsNothing(hv_Pose_Cylinder_in_RobBase) Then
            hv_Pose_Cylinder_in_RobBase = New HTuple
        End If
        hv_Pose_Cylinder_in_RobBase(New HTuple(1)) =
((hv_Pose_Cylinder_in_RobBase.TupleSelect( _
            New HTuple(1)))).TupleAdd(New HTuple(0.025))
        If IsNothing(hv_Pose_Cylinder_in_RobBase) Then
            hv_Pose_Cylinder_in_RobBase = New HTuple
        End If
        hv_Pose_Cylinder_in_RobBase(New HTuple(2)) =
((hv_Pose_Cylinder_in_RobBase.TupleSelect( _
            New HTuple(2)))).TupleAdd(New HTuple(0.0041))
        If IsNothing(hv_Pose_Cylinder_in_RobBase) Then
            hv_Pose_Cylinder_in_RobBase = New HTuple
        End If


        hv_PosX = hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(0)) * 1000
        hv_PosY = hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(1)) * 1000
        hv_PosZ = hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(2)) * 1000
        hv_RotX = hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(3))
        hv_RotY = hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(4))
        hv_RotZ = hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(5))

    End Sub

    Private Sub actionSavePose()
        '** Save Pose***
        HOperatorSet.WritePose(hv_Pose_Cylinder_in_RobBase, FinalPoseDirectory)
    End Sub



    Sub InitHalcon()
        ' Default settings used in HDevelop
        Call HOperatorSet.SetSystem(New HTuple("width"), New HTuple(512))

        Call HOperatorSet.SetSystem(New HTuple("height"), New HTuple(512))

    End Sub

    Sub RunHalconConvert()

        Call actionConvert()
    End Sub

    Sub RunHalconSavePose()
        Call actionSavePose()
    End Sub

End Module
```

A6.5. Communication.vb

```
Option Strict Off

Option Explicit On

Imports HalconDotNet

Module HDevelopExportCommunication

    ' Main procedures
    Private Sub actionDisconnect()

        'closing socket
        HOperatorSet.CloseSocket(Socket)
        Status = False
    End Sub
    Private Sub actionConnect()

        ' Local control variables

        Dim hv_timeout As HTuple = New HTuple


        ' Initialize local and output iconic variables

        hv_timeout = New HTuple(200)

        'opening the socket
        HOperatorSet.OpenSocketConnect(ip, New HTuple(Port), (New HTuple(New
HTuple("protocol"))).TupleConcat(New HTuple("timeout")), (New
HTuple("TCP")).TupleConcat(hv_timeout), Socket)
        Status = True

    End Sub



    Private Sub actionSend()
        Dim hv_Cylinder_RobBase_String As HTuple = New HTuple
        Dim hv_data_send_read As HTuple = New HTuple, hv_data_send_write As HTuple =
New HTuple
        Dim hv_Change As HTuple = New HTuple

        'Convert pose to let the robot control understands it
        hv_Change = hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(3))
        hv_Pose_Cylinder_in_RobBase(New HTuple(3)) =
hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(5))
        hv_Pose_Cylinder_in_RobBase(New HTuple(5)) = hv_Change

        hv_Pose_Cylinder_in_RobBase(New HTuple(0)) =
hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(0)) * 1000
        hv_Pose_Cylinder_in_RobBase(New HTuple(1)) =
hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(1)) * 1000
        hv_Pose_Cylinder_in_RobBase(New HTuple(2)) =
hv_Pose_Cylinder_in_RobBase.TupleSelect(New HTuple(2)) * 1000

        If IsNothing(hv_Pose_Cylinder_in_RobBase) Then

            hv_Pose_Cylinder_in_RobBase = New HTuple
        End If
        hv_Pose_Cylinder_in_RobBase(New HTuple(5)) = hv_Change
        HOperatorSet.ConvertPoseType(hv_Pose_Cylinder_in_RobBase, New HTuple("Rp+T"),
New HTuple("abg"), _
```

```
            New HTuple("point"), hv_Pose_Cylinder_in_RobBase)



        'Convert pose to send into a string
        HOperatorSet.TupleString(hv_Pose_Cylinder_in_RobBase, New HTuple("-"),
hv_Cylinder_RobBase_String)

        'preparing data to send
        hv_data_send_write = ((((((((((New HTuple("<?xml version=""1.0""
encoding=""UTF-8""?><SetVar VarValue=""{FRAME: X ")).TupleAdd( _
            hv_Cylinder_RobBase_String.TupleSelect(New HTuple(0))))).TupleAdd(New
HTuple(",Y ")))).TupleAdd( _
            hv_Cylinder_RobBase_String.TupleSelect(New HTuple(1))))).TupleAdd(New
HTuple(",Z ")))).TupleAdd( _
            hv_Cylinder_RobBase_String.TupleSelect(New HTuple(2))))).TupleAdd(New
HTuple(",A 0,B 0,C 0}"" VarName=""BASE_DATA[3]""/>"))
        hv_data_send_read = New HTuple("<?xml version=""1.0"" encoding=""UTF-
8""?><ShowVar VarName=""BASE_DATA[3]""/>")

        'sending data
        HOperatorSet.SendData(Socket, New HTuple("z"), hv_data_send_write, New
HTuple())

    End Sub



    Sub InitHalcon()
        ' Default settings used in HDevelop
        Call HOperatorSet.SetSystem(New HTuple("width"), New HTuple(512))

        Call HOperatorSet.SetSystem(New HTuple("height"), New HTuple(512))

    End Sub

    Sub RunHalconConnect()
        Call actionConnect()
    End Sub

    Sub RunHalconDisconnect()
        Call actionDisconnect()
    End Sub

    Sub RunHalconSend()
        Call actionSend()
    End Sub

End Module
```

A6.7. Vision Tool Form

```
Option Strict Off
Option Explicit On

Imports HalconDotNet
Imports System.IO


Public Class VisionTool
```

```
    Inherits System.Windows.Forms.Form

    'Variables Declaration
    Dim ShapeModelName(2) As String
    Dim CamParamName(2) As String




    Friend WithEvents txbShapeModel As System.Windows.Forms.TextBox
    Friend WithEvents txbRotZ As System.Windows.Forms.TextBox
    Friend WithEvents txbRotY As System.Windows.Forms.TextBox
    Friend WithEvents txbRotX As System.Windows.Forms.TextBox
    Friend WithEvents txbPosZ As System.Windows.Forms.TextBox
    Friend WithEvents txbPosY As System.Windows.Forms.TextBox
    Friend WithEvents txbPosX As System.Windows.Forms.TextBox
    Friend WithEvents txbNumPieces As System.Windows.Forms.TextBox
    Friend WithEvents ofdShapeModel As System.Windows.Forms.OpenFileDialog
    Friend WithEvents mnsVisionForm As System.Windows.Forms.MenuStrip
    Friend WithEvents FilesToolStripMenuItem As
System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents mnsOpenCamParam As System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents mnsOpenShapeModel As System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents ActionsToolStripMenuItem As
System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents mnsAcquire As System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents mnsCreateShapeModel As System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents mnsMatchPiece As System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents mnsClose As System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents HelpToolStripMenuItem As System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents ForFurtherInformationToolStripMenuItem As
System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents WwwacrobeToolStripMenuItem As
System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents WwwhalconcomToolStripMenuItem As
System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents DesignInformationToolStripMenuItem As
System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents DevelopedByJoseManuelPastorAlcarazForACROKHLimToolStripMenuItem
As System.Windows.Forms.ToolStripMenuItem
    Friend WithEvents txbCamParam As System.Windows.Forms.TextBox


#Region " Windows Form Designer generated code "

    Public Sub New()
        MyBase.New()

        'This call is required by the Windows Form Designer.
        InitializeComponent()

        'Add any initialization after the InitializeComponent() call

    End Sub

    'Form overrides dispose to clean up the component list.
    Protected Overloads Overrides Sub Dispose(ByVal disposing As Boolean)
        If disposing Then
            If Not (components Is Nothing) Then
                components.Dispose()
            End If
        End If
        MyBase.Dispose(disposing)
    End Sub
    'Required by the Windows Form Designer
    Private components As System.ComponentModel.IContainer

    'NOTE: The following procedure is required by the Windows Form Designer
    'It can be modified using the Windows Form Designer.
    'Do not modify it using the code editor.
    Private WithEvents HWinCtrl As HalconDotNet.HWindowControl
    Private WithEvents Status As System.Windows.Forms.Label
    Friend WithEvents grbModel As System.Windows.Forms.GroupBox
    Friend WithEvents btnOpenModel As System.Windows.Forms.Button
    Friend WithEvents lbl1 As System.Windows.Forms.Label
    Friend WithEvents btnClose As System.Windows.Forms.Button
    Friend WithEvents btnMatching As System.Windows.Forms.Button
    Friend WithEvents PictureBox1 As System.Windows.Forms.PictureBox
    Friend WithEvents grbResults As System.Windows.Forms.GroupBox
    Friend WithEvents btnSend As System.Windows.Forms.Button
    Friend WithEvents lbl9 As System.Windows.Forms.Label
    Friend WithEvents lbl8 As System.Windows.Forms.Label
    Friend WithEvents lbl7 As System.Windows.Forms.Label
    Friend WithEvents lbl6 As System.Windows.Forms.Label
    Friend WithEvents lbl5 As System.Windows.Forms.Label
    Friend WithEvents lbl4 As System.Windows.Forms.Label
    Friend WithEvents lbl3 As System.Windows.Forms.Label
    Friend WithEvents lbl2 As System.Windows.Forms.Label
    Friend WithEvents btnCreateShapeModel As System.Windows.Forms.Button
    Friend WithEvents GroupBox1 As System.Windows.Forms.GroupBox
    Friend WithEvents lbl10 As System.Windows.Forms.Label
    Friend WithEvents btnCameraParameters As System.Windows.Forms.Button
    Friend WithEvents ofdCamera As System.Windows.Forms.OpenFileDialog
    Private WithEvents btnAcquirement As System.Windows.Forms.Button
    <System.Diagnostics.DebuggerStepThrough()> Private Sub InitializeComponent()
        Dim resources As System.ComponentModel.ComponentResourceManager = New
System.ComponentModel.ComponentResourceManager(GetType(VisionTool))
        Me.HWinCtrl = New HalconDotNet.HWindowControl()
        Me.btnAcquirement = New System.Windows.Forms.Button()
        Me.Status = New System.Windows.Forms.Label()
        Me.grbModel = New System.Windows.Forms.GroupBox()
        Me.txbShapeModel = New System.Windows.Forms.TextBox()
        Me.btnOpenModel = New System.Windows.Forms.Button()
        Me.lbl1 = New System.Windows.Forms.Label()
        Me.btnClose = New System.Windows.Forms.Button()
        Me.btnMatching = New System.Windows.Forms.Button()
        Me.PictureBox1 = New System.Windows.Forms.PictureBox()
        Me.grbResults = New System.Windows.Forms.GroupBox()
        Me.txbRotZ = New System.Windows.Forms.TextBox()
        Me.txbRotY = New System.Windows.Forms.TextBox()
        Me.txbRotX = New System.Windows.Forms.TextBox()
        Me.txbPosZ = New System.Windows.Forms.TextBox()
        Me.txbPosY = New System.Windows.Forms.TextBox()
        Me.txbPosX = New System.Windows.Forms.TextBox()
        Me.txbNumPieces = New System.Windows.Forms.TextBox()
        Me.btnSend = New System.Windows.Forms.Button()
        Me.lbl9 = New System.Windows.Forms.Label()
        Me.lbl8 = New System.Windows.Forms.Label()
        Me.lbl7 = New System.Windows.Forms.Label()
        Me.lbl6 = New System.Windows.Forms.Label()
        Me.lbl5 = New System.Windows.Forms.Label()
        Me.lbl4 = New System.Windows.Forms.Label()
        Me.lbl3 = New System.Windows.Forms.Label()
        Me.lbl2 = New System.Windows.Forms.Label()
        Me.btnCreateShapeModel = New System.Windows.Forms.Button()
        Me.GroupBox1 = New System.Windows.Forms.GroupBox()
        Me.txbCamParam = New System.Windows.Forms.TextBox()
        Me.lbl10 = New System.Windows.Forms.Label()
        Me.btnCameraParameters = New System.Windows.Forms.Button()
        Me.ofdCamera = New System.Windows.Forms.OpenFileDialog()
        Me.ofdShapeModel = New System.Windows.Forms.OpenFileDialog()
        Me.mnsVisionForm = New System.Windows.Forms.MenuStrip()
        Me.FilesToolStripMenuItem = New System.Windows.Forms.ToolStripMenuItem()
        Me.mnsOpenCamParam = New System.Windows.Forms.ToolStripMenuItem()
        Me.mnsOpenShapeModel = New System.Windows.Forms.ToolStripMenuItem()
        Me.ActionsToolStripMenuItem = New System.Windows.Forms.ToolStripMenuItem()
        Me.mnsAcquire = New System.Windows.Forms.ToolStripMenuItem()
        Me.mnsCreateShapeModel = New System.Windows.Forms.ToolStripMenuItem()
        Me.mnsMatchPiece = New System.Windows.Forms.ToolStripMenuItem()
        Me.mnsClose = New System.Windows.Forms.ToolStripMenuItem()
        Me.HelpToolStripMenuItem = New System.Windows.Forms.ToolStripMenuItem()
        Me.ForFurtherInformationToolStripMenuItem = New
System.Windows.Forms.ToolStripMenuItem()
        Me.WwwacrobeToolStripMenuItem = New System.Windows.Forms.ToolStripMenuItem()
        Me.WwwhalconcomToolStripMenuItem = New
System.Windows.Forms.ToolStripMenuItem()
        Me.DesignInformationToolStripMenuItem = New
System.Windows.Forms.ToolStripMenuItem()
        Me.DevelopedByJoseManuelPastorAlcarazForACROKHLimToolStripMenuItem = New
System.Windows.Forms.ToolStripMenuItem()
        Me.grbModel.SuspendLayout()
        CType(Me.PictureBox1, System.ComponentModel.ISupportInitialize).BeginInit()
        Me.grbResults.SuspendLayout()
        Me.GroupBox1.SuspendLayout()
        Me.mnsVisionForm.SuspendLayout()
        Me.SuspendLayout()
        '
        'HWinCtrl
        '
        Me.HWinCtrl.BackColor = System.Drawing.SystemColors.MenuText
        Me.HWinCtrl.BorderColor = System.Drawing.SystemColors.MenuText
        Me.HWinCtrl.ImagePart = New System.Drawing.Rectangle(0, 0, 2024, 1536)
        Me.HWinCtrl.Location = New System.Drawing.Point(12, 27)
        Me.HWinCtrl.Name = "HWinCtrl"
        Me.HWinCtrl.Size = New System.Drawing.Size(569, 385)
        Me.HWinCtrl.TabIndex = 0
        Me.HWinCtrl.WindowSize = New System.Drawing.Size(569, 385)
        '
        'btnAcquirement
        '
        Me.btnAcquirement.Cursor = System.Windows.Forms.Cursors.Default
        Me.btnAcquirement.Location = New System.Drawing.Point(545, 546)
        Me.btnAcquirement.Name = "btnAcquirement"
        Me.btnAcquirement.Size = New System.Drawing.Size(122, 31)
        Me.btnAcquirement.TabIndex = 1
        Me.btnAcquirement.Text = "Acquire New Image"
        '
        'Status
        '
        Me.Status.Location = New System.Drawing.Point(0, 528)
        Me.Status.Name = "Status"
        Me.Status.Size = New System.Drawing.Size(608, 16)
        Me.Status.TabIndex = 2
        Me.Status.UseWaitCursor = True
        '
        'grbModel
        '
        Me.grbModel.BackColor = System.Drawing.SystemColors.MenuBar
        Me.grbModel.Controls.Add(Me.txbShapeModel)
        Me.grbModel.Controls.Add(Me.btnOpenModel)
        Me.grbModel.Controls.Add(Me.lbl1)
        Me.grbModel.Cursor = System.Windows.Forms.Cursors.Default
        Me.grbModel.FlatStyle = System.Windows.Forms.FlatStyle.Flat
        Me.grbModel.Location = New System.Drawing.Point(284, 450)
        Me.grbModel.Name = "grbModel"
        Me.grbModel.Size = New System.Drawing.Size(231, 138)
        Me.grbModel.TabIndex = 3
        Me.grbModel.TabStop = False
        Me.grbModel.Text = " Shape Model "
        '
        'txbShapeModel
        '
        Me.txbShapeModel.BackColor = System.Drawing.SystemColors.ControlLightLight
        Me.txbShapeModel.Enabled = False
        Me.txbShapeModel.Location = New System.Drawing.Point(9, 55)
        Me.txbShapeModel.Name = "txbShapeModel"
        Me.txbShapeModel.Size = New System.Drawing.Size(216, 20)
        Me.txbShapeModel.TabIndex = 5
        Me.txbShapeModel.TextAlign = System.Windows.Forms.HorizontalAlignment.Right
        '
        'btnOpenModel
        '
        Me.btnOpenModel.BackColor = System.Drawing.SystemColors.Menu
        Me.btnOpenModel.Cursor = System.Windows.Forms.Cursors.Default
        Me.btnOpenModel.Location = New System.Drawing.Point(88, 96)
        Me.btnOpenModel.Name = "btnOpenModel"
        Me.btnOpenModel.Size = New System.Drawing.Size(139, 29)
        Me.btnOpenModel.TabIndex = 4
        Me.btnOpenModel.Text = "Open Shape Model"
        Me.btnOpenModel.UseVisualStyleBackColor = False
        '
        'lbl1
        '
        Me.lbl1.AutoSize = True
        Me.lbl1.Location = New System.Drawing.Point(6, 39)
        Me.lbl1.Name = "lbl1"
        Me.lbl1.Size = New System.Drawing.Size(29, 13)
        Me.lbl1.TabIndex = 2
        Me.lbl1.Text = "File: "
        '
        'btnClose
        '
        Me.btnClose.Cursor = System.Windows.Forms.Cursors.Default
        Me.btnClose.Location = New System.Drawing.Point(929, 547)
        Me.btnClose.Name = "btnClose"
        Me.btnClose.Size = New System.Drawing.Size(122, 30)
        Me.btnClose.TabIndex = 4
        Me.btnClose.Text = "Close"
        Me.btnClose.UseVisualStyleBackColor = True
        '
        'btnMatching
        '
        Me.btnMatching.Cursor = System.Windows.Forms.Cursors.Default
        Me.btnMatching.Location = New System.Drawing.Point(801, 546)
        Me.btnMatching.Name = "btnMatching"
        Me.btnMatching.Size = New System.Drawing.Size(122, 31)
        Me.btnMatching.TabIndex = 5
        Me.btnMatching.Text = "Match Piece"
        Me.btnMatching.UseVisualStyleBackColor = True
        '
        'PictureBox1
        '
        Me.PictureBox1.Image = CType(resources.GetObject("PictureBox1.Image"),
System.Drawing.Image)
        Me.PictureBox1.Location = New System.Drawing.Point(641, 27)
        Me.PictureBox1.Name = "PictureBox1"
        Me.PictureBox1.Size = New System.Drawing.Size(401, 122)
        Me.PictureBox1.TabIndex = 6
        Me.PictureBox1.TabStop = False
```

```
'
'grbResults
'
Me.grbResults.BackColor = System.Drawing.SystemColors.MenuBar
Me.grbResults.Controls.Add(Me.txbRotZ)
Me.grbResults.Controls.Add(Me.txbRotY)
Me.grbResults.Controls.Add(Me.txbRotX)
Me.grbResults.Controls.Add(Me.txbPosZ)
Me.grbResults.Controls.Add(Me.txbPosY)
Me.grbResults.Controls.Add(Me.txbPosX)
Me.grbResults.Controls.Add(Me.txbNumPieces)
Me.grbResults.Controls.Add(Me.btnSend)
Me.grbResults.Controls.Add(Me.lbl9)
Me.grbResults.Controls.Add(Me.lbl8)
Me.grbResults.Controls.Add(Me.lbl7)
Me.grbResults.Controls.Add(Me.lbl6)
Me.grbResults.Controls.Add(Me.lbl5)
Me.grbResults.Controls.Add(Me.lbl4)
Me.grbResults.Controls.Add(Me.lbl3)
Me.grbResults.Controls.Add(Me.lbl2)
Me.grbResults.Cursor = System.Windows.Forms.Cursors.Default
Me.grbResults.Location = New System.Drawing.Point(641, 166)
Me.grbResults.Name = "grbResults"
Me.grbResults.Size = New System.Drawing.Size(401, 287)
Me.grbResults.TabIndex = 7
Me.grbResults.TabStop = False
Me.grbResults.Text = "Results of the Matching "
'
'txbRotZ
'
Me.txbRotZ.BackColor = System.Drawing.SystemColors.ControlLightLight
Me.txbRotZ.Enabled = False
Me.txbRotZ.Location = New System.Drawing.Point(106, 211)
Me.txbRotZ.Name = "txbRotZ"
Me.txbRotZ.Size = New System.Drawing.Size(272, 20)
Me.txbRotZ.TabIndex = 22
Me.txbRotZ.TextAlign = System.Windows.Forms.HorizontalAlignment.Right
'
'txbRotY
'
Me.txbRotY.BackColor = System.Drawing.SystemColors.ControlLightLight
Me.txbRotY.Enabled = False
Me.txbRotY.Location = New System.Drawing.Point(106, 180)
Me.txbRotY.Name = "txbRotY"
Me.txbRotY.Size = New System.Drawing.Size(272, 20)
Me.txbRotY.TabIndex = 21
Me.txbRotY.TextAlign = System.Windows.Forms.HorizontalAlignment.Right
'
'txbRotX
'
Me.txbRotX.BackColor = System.Drawing.SystemColors.ControlLightLight
Me.txbRotX.Enabled = False
Me.txbRotX.Location = New System.Drawing.Point(106, 153)
Me.txbRotX.Name = "txbRotX"
Me.txbRotX.Size = New System.Drawing.Size(272, 20)
Me.txbRotX.TabIndex = 20
Me.txbRotX.TextAlign = System.Windows.Forms.HorizontalAlignment.Right
'
'txbPosZ
'
Me.txbPosZ.BackColor = System.Drawing.SystemColors.ControlLightLight
Me.txbPosZ.Enabled = False
Me.txbPosZ.Location = New System.Drawing.Point(106, 126)
Me.txbPosZ.Name = "txbPosZ"
Me.txbPosZ.Size = New System.Drawing.Size(272, 20)
Me.txbPosZ.TabIndex = 19
Me.txbPosZ.TextAlign = System.Windows.Forms.HorizontalAlignment.Right
'
'txbPosY
'
Me.txbPosY.BackColor = System.Drawing.SystemColors.ControlLightLight
Me.txbPosY.Enabled = False
Me.txbPosY.Location = New System.Drawing.Point(106, 100)
Me.txbPosY.Name = "txbPosY"
Me.txbPosY.Size = New System.Drawing.Size(272, 20)
Me.txbPosY.TabIndex = 18
Me.txbPosY.TextAlign = System.Windows.Forms.HorizontalAlignment.Right
'
'txbPosX
'
Me.txbPosX.BackColor = System.Drawing.SystemColors.ControlLightLight
Me.txbPosX.Enabled = False
Me.txbPosX.Location = New System.Drawing.Point(106, 75)
Me.txbPosX.Name = "txbPosX"
Me.txbPosX.Size = New System.Drawing.Size(272, 20)
Me.txbPosX.TabIndex = 17
Me.txbPosX.TextAlign = System.Windows.Forms.HorizontalAlignment.Right
'
'txbNumPieces
'
Me.txbNumPieces.BackColor = System.Drawing.SystemColors.ControlLightLight
Me.txbNumPieces.Enabled = False
Me.txbNumPieces.Location = New System.Drawing.Point(117, 23)
Me.txbNumPieces.Name = "txbNumPieces"
Me.txbNumPieces.Size = New System.Drawing.Size(47, 20)
Me.txbNumPieces.TabIndex = 16
Me.txbNumPieces.TextAlign = System.Windows.Forms.HorizontalAlignment.Right
'
'btnSend
'
Me.btnSend.BackColor = System.Drawing.SystemColors.Menu
Me.btnSend.Cursor = System.Windows.Forms.Cursors.Default
Me.btnSend.Location = New System.Drawing.Point(240, 253)
Me.btnSend.Name = "btnSend"
Me.btnSend.Size = New System.Drawing.Size(138, 28)
Me.btnSend.TabIndex = 15
Me.btnSend.Text = "Send Pose to Robot"
Me.btnSend.UseVisualStyleBackColor = False
'
'lbl9
'
Me.lbl9.AutoSize = True
Me.lbl9.Location = New System.Drawing.Point(53, 214)
Me.lbl9.Name = "lbl9"
Me.lbl9.Size = New System.Drawing.Size(43, 13)
Me.lbl9.TabIndex = 8
Me.lbl9.Text = "Rot(Z): "
'
'lbl8
'
Me.lbl8.AutoSize = True
Me.lbl8.Location = New System.Drawing.Point(53, 183)
Me.lbl8.Name = "lbl8"
Me.lbl8.Size = New System.Drawing.Size(43, 13)
Me.lbl8.TabIndex = 7
Me.lbl8.Text = "Rot(Y): "
'
'lbl7
'
Me.lbl7.AutoSize = True
Me.lbl7.Location = New System.Drawing.Point(53, 156)
Me.lbl7.Name = "lbl7"
Me.lbl7.Size = New System.Drawing.Size(43, 13)
Me.lbl7.TabIndex = 6
Me.lbl7.Text = "Rot(X): "
'
'lbl6
'
Me.lbl6.AutoSize = True
Me.lbl6.Location = New System.Drawing.Point(53, 129)
Me.lbl6.Name = "lbl6"
Me.lbl6.Size = New System.Drawing.Size(44, 13)
Me.lbl6.TabIndex = 5
Me.lbl6.Text = "Pos(Z): "
'
'lbl5
'
Me.lbl5.AutoSize = True
Me.lbl5.Location = New System.Drawing.Point(53, 103)
Me.lbl5.Name = "lbl5"
Me.lbl5.Size = New System.Drawing.Size(44, 13)
Me.lbl5.TabIndex = 4
Me.lbl5.Text = "Pos(Y): "
'
'lbl4
'
Me.lbl4.AutoSize = True
Me.lbl4.Location = New System.Drawing.Point(53, 78)
Me.lbl4.Name = "lbl4"
Me.lbl4.Size = New System.Drawing.Size(44, 13)
Me.lbl4.TabIndex = 3
Me.lbl4.Text = "Pos(X): "
'
'lbl3
'
Me.lbl3.AutoSize = True
Me.lbl3.Location = New System.Drawing.Point(6, 54)
Me.lbl3.Name = "lbl3"
Me.lbl3.Size = New System.Drawing.Size(233, 13)
Me.lbl3.TabIndex = 2
Me.lbl3.Text = "Pose Information in Camera Coordinate System: "
'
'lbl2
'
Me.lbl2.AutoSize = True
Me.lbl2.Location = New System.Drawing.Point(6, 26)
Me.lbl2.Name = "lbl2"
Me.lbl2.Size = New System.Drawing.Size(105, 13)
Me.lbl2.TabIndex = 0
Me.lbl2.Text = "Nº Pieces detected: "
'
'btnCreateShapeModel
'
Me.btnCreateShapeModel.Cursor = System.Windows.Forms.Cursors.Default
Me.btnCreateShapeModel.Location = New System.Drawing.Point(673, 546)
Me.btnCreateShapeModel.Name = "btnCreateShapeModel"
Me.btnCreateShapeModel.Size = New System.Drawing.Size(122, 31)
Me.btnCreateShapeModel.TabIndex = 8
Me.btnCreateShapeModel.Text = "Create Shape Model "
Me.btnCreateShapeModel.UseVisualStyleBackColor = True
'
'GroupBox1
'
Me.GroupBox1.BackColor = System.Drawing.SystemColors.MenuBar
Me.GroupBox1.Controls.Add(Me.txbCamParam)
```

```
        Me.GroupBox1.Controls.Add(Me.lbl10)
        Me.GroupBox1.Controls.Add(Me.btnCameraParameters)
        Me.GroupBox1.Location = New System.Drawing.Point(23, 450)
        Me.GroupBox1.Name = "GroupBox1"
        Me.GroupBox1.Size = New System.Drawing.Size(244, 138)
        Me.GroupBox1.TabIndex = 9
        Me.GroupBox1.TabStop = False
        Me.GroupBox1.Text = "Camera Parameters"
        '
        'txbCamParam
        '
        Me.txbCamParam.BackColor = System.Drawing.SystemColors.ControlLightLight
        Me.txbCamParam.Enabled = False
        Me.txbCamParam.Location = New System.Drawing.Point(9, 55)
        Me.txbCamParam.Name = "txbCamParam"
        Me.txbCamParam.Size = New System.Drawing.Size(229, 20)
        Me.txbCamParam.TabIndex = 6
        Me.txbCamParam.TextAlign = System.Windows.Forms.HorizontalAlignment.Right
        '
        'lbl10
        '
        Me.lbl10.AutoSize = True
        Me.lbl10.Location = New System.Drawing.Point(6, 39)
        Me.lbl10.Name = "lbl10"
        Me.lbl10.Size = New System.Drawing.Size(26, 13)
        Me.lbl10.TabIndex = 5
        Me.lbl10.Text = "File:"
        '
        'btnCameraParameters
        '
        Me.btnCameraParameters.BackColor = System.Drawing.SystemColors.Menu
        Me.btnCameraParameters.Cursor = System.Windows.Forms.Cursors.Default
        Me.btnCameraParameters.Location = New System.Drawing.Point(80, 94)
        Me.btnCameraParameters.Name = "btnCameraParameters"
        Me.btnCameraParameters.Size = New System.Drawing.Size(158, 31)
        Me.btnCameraParameters.TabIndex = 5
        Me.btnCameraParameters.Text = "Open Camera Parameters"
        Me.btnCameraParameters.UseVisualStyleBackColor = False
        '
        'mnsVisionForm
        '
        Me.mnsVisionForm.BackColor = System.Drawing.SystemColors.MenuBar
        Me.mnsVisionForm.Items.AddRange(New System.Windows.Forms.ToolStripItem()
{Me.FilesToolStripMenuItem, Me.ActionsToolStripMenuItem, Me.HelpToolStripMenuItem})
        Me.mnsVisionForm.Location = New System.Drawing.Point(0, 0)
        Me.mnsVisionForm.Name = "mnsVisionForm"
        Me.mnsVisionForm.Size = New System.Drawing.Size(1059, 24)
        Me.mnsVisionForm.TabIndex = 10
        '
        'FilesToolStripMenuItem
        '
        Me.FilesToolStripMenuItem.DropDownItems.AddRange(New
System.Windows.Forms.ToolStripItem() {Me.mnsOpenCamParam, Me.mnsOpenShapeModel})
        Me.FilesToolStripMenuItem.Name = "FilesToolStripMenuItem"
        Me.FilesToolStripMenuItem.Size = New System.Drawing.Size(40, 20)
        Me.FilesToolStripMenuItem.Text = "Files"
        '
        'mnsOpenCamParam
        '
        Me.mnsOpenCamParam.Name = "mnsOpenCamParam"
        Me.mnsOpenCamParam.Size = New System.Drawing.Size(209, 22)
        Me.mnsOpenCamParam.Text = "Open Camera Parameters"
        '
        'mnsOpenShapeModel
        '
        Me.mnsOpenShapeModel.Name = "mnsOpenShapeModel"
        Me.mnsOpenShapeModel.Size = New System.Drawing.Size(209, 22)
        Me.mnsOpenShapeModel.Text = "Open 3D Shape Model "
        '
        'ActionsToolStripMenuItem
        '
        Me.ActionsToolStripMenuItem.DropDownItems.AddRange(New
System.Windows.Forms.ToolStripItem() {Me.mnsAcquire, Me.mnsCreateShapeModel,
Me.mnsMatchPiece, Me.mnsClose})
        Me.ActionsToolStripMenuItem.Name = "ActionsToolStripMenuItem"
        Me.ActionsToolStripMenuItem.Size = New System.Drawing.Size(57, 20)
        Me.ActionsToolStripMenuItem.Text = "Actions "
        '
        'mnsAcquire
        '
        Me.mnsAcquire.Name = "mnsAcquire"
        Me.mnsAcquire.Size = New System.Drawing.Size(187, 22)
        Me.mnsAcquire.Text = "Acquire a New Image"
        '
        'mnsCreateShapeModel
        '
        Me.mnsCreateShapeModel.Name = "mnsCreateShapeModel"
        Me.mnsCreateShapeModel.Size = New System.Drawing.Size(187, 22)
        Me.mnsCreateShapeModel.Text = "Create Shape Model "
        '
        'mnsMatchPiece
        '
        Me.mnsMatchPiece.Name = "mnsMatchPiece"
        Me.mnsMatchPiece.Size = New System.Drawing.Size(187, 22)
        Me.mnsMatchPiece.Text = "Match Piece"
        '
        'mnsClose
        '
        Me.mnsClose.Name = "mnsClose"
        Me.mnsClose.Size = New System.Drawing.Size(187, 22)
        Me.mnsClose.Text = "Close Application"
        '
        'HelpToolStripMenuItem
        '
        Me.HelpToolStripMenuItem.DropDownItems.AddRange(New
System.Windows.Forms.ToolStripItem() {Me.ForFurtherInformationToolStripMenuItem,
Me.DesignInformationToolStripMenuItem})
        Me.HelpToolStripMenuItem.Name = "HelpToolStripMenuItem"
        Me.HelpToolStripMenuItem.Size = New System.Drawing.Size(40, 20)
        Me.HelpToolStripMenuItem.Text = "Help"
        '
        'ForFurtherInformationToolStripMenuItem
        '
        Me.ForFurtherInformationToolStripMenuItem.DropDownItems.AddRange(New
System.Windows.Forms.ToolStripItem() {Me.WwwacrobeToolStripMenuItem,
Me.WwwhalconcomToolStripMenuItem})
        Me.ForFurtherInformationToolStripMenuItem.Name =
"ForFurtherInformationToolStripMenuItem"
        Me.ForFurtherInformationToolStripMenuItem.Size = New System.Drawing.Size(197,
22)
        Me.ForFurtherInformationToolStripMenuItem.Text = "For Further information"
        '
        'WwwacrobeToolStripMenuItem
        '
        Me.WwwacrobeToolStripMenuItem.Name = "WwwacrobeToolStripMenuItem"
        Me.WwwacrobeToolStripMenuItem.Size = New System.Drawing.Size(167, 22)
        Me.WwwacrobeToolStripMenuItem.Text = "www.acro.be"
        '
        'WwwhalconcomToolStripMenuItem
        '
        Me.WwwhalconcomToolStripMenuItem.Name = "WwwhalconcomToolStripMenuItem"
        Me.WwwhalconcomToolStripMenuItem.Size = New System.Drawing.Size(167, 22)
        Me.WwwhalconcomToolStripMenuItem.Text = "www.halcon.com"
        '
        'DesignInformationToolStripMenuItem
        '
        Me.DesignInformationToolStripMenuItem.DropDownItems.AddRange(New
System.Windows.Forms.ToolStripItem()
{Me.DevelopedByJoseManuelPastorAlcarazForACROKHLimToolStripMenuItem})
        Me.DesignInformationToolStripMenuItem.Name =
"DesignInformationToolStripMenuItem"
        Me.DesignInformationToolStripMenuItem.Size = New System.Drawing.Size(197, 22)
        Me.DesignInformationToolStripMenuItem.Text = "Design Information"
        '
        'DevelopedByJoseManuelPastorAlcarazForACROKHLimToolStripMenuItem
        '
        Me.DevelopedByJoseManuelPastorAlcarazForACROKHLimToolStripMenuItem.Name =
"DevelopedByJoseManuelPastorAlcarazForACROKHLimToolStripMenuItem"
        Me.DevelopedByJoseManuelPastorAlcarazForACROKHLimToolStripMenuItem.Size = New
System.Drawing.Size(373, 22)
        Me.DevelopedByJoseManuelPastorAlcarazForACROKHLimToolStripMenuItem.Text =
"Developed by Jose Manuel Pastor Alcaraz for ACRO (KHLim)"
        '
        'VisionTool
        '
        Me.AutoScaleDimensions = New System.Drawing.SizeF(6.0!, 13.0!)
        Me.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font
        Me.ClientSize = New System.Drawing.Size(1059, 600)
        Me.Controls.Add(Me.GroupBox1)
        Me.Controls.Add(Me.btnCreateShapeModel)
        Me.Controls.Add(Me.grbResults)
        Me.Controls.Add(Me.PictureBox1)
        Me.Controls.Add(Me.btnMatching)
        Me.Controls.Add(Me.btnClose)
        Me.Controls.Add(Me.grbModel)
        Me.Controls.Add(Me.Status)
        Me.Controls.Add(Me.btnAcquirement)
        Me.Controls.Add(Me.HWinCtrl)
        Me.Controls.Add(Me.mnsVisionForm)
        Me.Cursor = System.Windows.Forms.Cursors.Default
        Me.HelpButton = True
        Me.Icon = CType(resources.GetObject("$this.Icon"), System.Drawing.Icon)
        Me.MainMenuStrip = Me.mnsVisionForm
        Me.Name = "VisionTool"
        Me.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen
        Me.Text = "3D Shape-Based Matching "
        Me.grbModel.ResumeLayout(False)
        Me.grbModel.PerformLayout()
        CType(Me.PictureBox1, System.ComponentModel.ISupportInitialize).EndInit()
        Me.grbResults.ResumeLayout(False)
        Me.grbResults.PerformLayout()
        Me.GroupBox1.ResumeLayout(False)
        Me.GroupBox1.PerformLayout()
        Me.mnsVisionForm.ResumeLayout(False)
        Me.mnsVisionForm.PerformLayout()
        Me.ResumeLayout(False)
        Me.PerformLayout()

    End Sub

#End Region

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
```

```
    End Sub

    Private Sub btnAcquirement_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAcquirement.Click
        Dim WindowID As HTuple = New HTuple(HWinCtrl.HalconID())
        Status.Refresh()
        Call RunHalconAcq(WindowID)
    End Sub

    Private Sub btnOpenModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnOpenModel.Click
        ofdShapeModel.InitialDirectory = "C:\Documents and
Settings\acro\Desktop\SmartFactory\matching\Cylinder"
        ofdShapeModel.Filter = "stl files (*.sm3)|*.sm3|All files (*.*)|*.*"
        ofdShapeModel.FilterIndex = 1
        ofdShapeModel.RestoreDirectory = True
        ofdShapeModel.ShowDialog()

        If Len(ofdShapeModel.FileName) = 0 Then
            'Do nothing
        Else
            ShapeModelName = Split(ofdShapeModel.SafeFileName, ".")

            If ShapeModelName(ShapeModelName.Length - 1) = "sm3" Then
                txbShapeModel.Text = ofdShapeModel.SafeFileName()
                ShapeModelDirectory = (ofdShapeModel.FileName())
                ShapeRead = False
            Else
                txbShapeModel.Text = "File extension invalid, charge again"
            End If
        End If


    End Sub



    Private Sub btnClose_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnClose.Click

        Call FinishHalcon()
        End
    End Sub


    Private Sub btnCreateShapeModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCreateShapeModel.Click
        If Len(ofdCamera.FileName) = 0 Or CamParamName(CamParamName.Length - 1) <>
"cal" Then
            Call MessageBox.Show("Camera Parameters has to be properly selected",
Me.Text, MessageBoxButtons.OK)
        Else
            ShapeCreationForm.Show()
        End If

    End Sub

    Private Sub btnCameraParameters_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCameraParameters.Click
        ofdCamera.InitialDirectory = "C:\Documents and
Settings\acro\Desktop\SmartFactory\Calibration\Camera Calibration\cam_file_pose"
        ofdCamera.Filter = "cal files (*.cal)|*.cal|All files (*.*)|*.*"
        ofdCamera.FilterIndex = 1
        ofdCamera.RestoreDirectory = True
        ofdCamera.ShowDialog()
        If Len(ofdCamera.FileName) = 0 Then
            'Do nothing
        Else
            CamParamName = Split(ofdCamera.SafeFileName, ".")
            If CamParamName(CamParamName.Length - 1) = "cal" Then
                txbCamParam.Text = ofdCamera.SafeFileName()
                CamParamDirectory = (ofdCamera.FileName())

            Else
                txbCamParam.Text = "File extension invalid, charge again"
            End If
        End If


    End Sub

    Private Sub btnMatching_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnMatching.Click

        If Len(ofdCamera.FileName) = 0 Or Len(ofdShapeModel.FileName) = 0 Or
CamParamName(CamParamName.Length - 1) <> "cal" Or
ShapeModelName(ShapeModelName.Length - 1) <> "sm3" Then

            Call MessageBox.Show("Select correct Camera Parameters and 3D Shape
Model", Me.Text, MessageBoxButtons.OK)
        Else

            Dim WindowID As HTuple = New HTuple(HWinCtrl.HalconID())
            Status.Refresh()
            Call RunHalconMatch(WindowID)
            CylinPoseMatch = True


            'Show the number of pieces detected
            txbNumPieces.Text = hv_NumPieces.ToString
            'Show Pose results in window
            txbPosX.Text = hv_PosX.ToString & " mm"
            txbPosY.Text = hv_PosY.ToString & " mm"
            txbPosZ.Text = hv_PosZ.ToString & " mm"
            txbRotX.Text = hv_RotX.ToString & "°"
            txbRotY.Text = hv_RotY.ToString & "°"
            txbRotZ.Text = hv_RotZ.ToString & "°"
        End If
    End Sub

    Private Sub btnSend_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSend.Click

        If CylinPoseMatch = True Then
            CommunicationForm.Show()
        Else
            Call MessageBox.Show("There is no pose to send", Me.Text,
MessageBoxButtons.OK)
        End If


    End Sub


    'Menu Strip Declaration

    Private Sub mnsOpenCamParam_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsOpenCamParam.Click
        Call btnCameraParameters_Click(sender, e)
    End Sub

    Private Sub mnsOpenShapeModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsOpenShapeModel.Click
        Call btnOpenModel_Click(sender, e)
    End Sub

    Private Sub mnsAcquire_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsAcquire.Click
        Call btnAcquirement_Click(sender, e)
    End Sub

    Private Sub mnsCreateShapeModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsCreateShapeModel.Click
        Call btnCreateShapeModel_Click(sender, e)
    End Sub

    Private Sub mnsMatchPiece_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsMatchPiece.Click
        Call btnMatching_Click(sender, e)
    End Sub

    Private Sub mnsClose_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsClose.Click
        Call btnClose_Click(sender, e)
    End Sub
End Class
```

A6.8. Shape Creation Form

```
Option Strict Off
Option Explicit On

Imports HalconDotNet
Imports System.IO



Public Class ShapeCreationForm
    'Variables Declaration
    Dim StlModelName(2) As String

    Private Sub ShapeCreationForm_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    End Sub

    Private Sub btnOpenModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnOpenModel.Click
        ofdSTLModel.InitialDirectory = "C:\Documents and
Settings\acro\Desktop\SmartFactory\matching\Cylinder"
        ofdSTLModel.Filter = "STL files (*.stl)|*.stl|All files (*.*)|*.*"
        ofdSTLModel.FilterIndex = 1
        ofdSTLModel.RestoreDirectory = True
        ofdSTLModel.ShowDialog()


        If Len(ofdSTLModel.FileName) = 0 Then
            'Do nothing
        Else
```

```
            StlModelName = Split(ofdSTLModel.SafeFileName, ".")
            If StlModelName(StlModelName.Length - 1) = "STL" Then
                ModelDirectory = (ofdSTLModel.FileName())
            Else
                Call MessageBox.Show("File Extension invalid", Me.Text,
MessageBoxButtons.OK)
            End If
        End If

    End Sub

    Private Sub btnClose_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnClose.Click



        ShapeCreated = False
        Me.Close()
    End Sub

    Private Sub btnSaveShapeModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSaveShapeModel.Click


        If ShapeCreated = True Then
            sfdShapeModel.Filter = "sm3 Files (*.sm3*)|*.sm3"
            sfdShapeModel.InitialDirectory = "C:\Documents and
Settings\acro\Desktop\SmartFactory\matching\Cylinder"
            sfdShapeModel.ShowDialog()
            If sfdShapeModel.ShowDialog = Windows.Forms.DialogResult.OK Then
                If sfdShapeModel.FileName <> "" Then
                    ShapeModelDirectory = (sfdShapeModel.FileName())
                    Call RunHalconSaveShapeModel()
                    Call MessageBox.Show("3D Shape Model saved succesfully", Me.Text,
MessageBoxButtons.OK)
                End If
            End If
        Else
            Call MessageBox.Show("3D Shape Model has to be created first", Me.Text,
MessageBoxButtons.OK)
        End If

    End Sub

    Private Sub btnCreateShapeModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCreateShapeModel.Click
        Dim WindowIDMenu As HTuple = New HTuple(HWindowMenu.HalconID())
        Dim WindowIDSelect As HTuple = New HTuple(HWindowSelect.HalconID())
        Dim WindowIDView As HTuple = New HTuple(HWindowView.HalconID())
        Dim WindowIDBuffer As HTuple = New HTuple(HWindowBuffer.HalconID())

        ShapeCreated = False

        If Len(ofdSTLModel.FileName) = 0 Or StlModelName(StlModelName.Length - 1) <>
"STL" Then
            Call MessageBox.Show("A 3D Model has to be selected", Me.Text,
MessageBoxButtons.OK)
        Else

            Call RunHalconCreateShapeModel(WindowIDMenu, WindowIDSelect,
WindowIDView, WindowIDBuffer)
            ShapeCreated = True
            ShapeRead = False
            Call MessageBox.Show("3D Shape Model created succesfully", Me.Text,
MessageBoxButtons.OK)
```

```
        End If

    End Sub


    'Menu Strip Declaration
    Private Sub mnsOpenSTLModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsOpenSTLModel.Click
        Call btnOpenModel_Click(sender, e)
    End Sub

    Private Sub mnsCreateShapeModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsCreateShapeModel.Click
        Call btnCreateShapeModel_Click(sender, e)
    End Sub

    Private Sub mnsSaveShapeModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsSaveShapeModel.Click
        Call btnSaveShapeModel_Click(sender, e)
    End Sub

    Private Sub mnsExit_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsExit.Click
        Call btnClose_Click(sender, e)
    End Sub
End Class
```

A6.9. Communication Form

```
Option Strict Off
Option Explicit On

Imports HalconDotNet
Imports System.IO


Public Class CommunicationForm

    'Variables Declaration
    Dim CamRobName(2) As String
    Dim Poseconverted As Boolean = False


    Private Sub CommunicationForm_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

        txbIP.Text = IpMemo
        txbPort.Text = PortMemo
        txbIP.Enabled = True
        txbPort.Enabled = True
        btnDisconnect.Enabled = False
        btnConnect.Enabled = True
        btnSend.Enabled = False
        txbStatus.Text = "Disconnected"




    End Sub
    Private Sub btnClose_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnClose.Click
        If Status = True Then
            Call RunHalconDisconnect()
        End If
        Poseconverted = False
        Me.Close()
    End Sub

    Private Sub btnConvert_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnConvert.Click
        If Len(ofdCamRob.FileName) = 0 Or CamRobName(CamRobName.Length - 1) <> "dat"
Then

            Call MessageBox.Show("Select a correct Camera to Robot Pose", Me.Text,
MessageBoxButtons.OK)
        Else


            Call RunHalconConvert()
            Poseconverted = True

            'Show Pose results in window
            txbPosXCon.Text = hv_PosX.ToString & " mm"
            txbPosYCon.Text = hv_PosY.ToString & " mm"
            txbPosZCon.Text = hv_PosZ.ToString & " mm"
            txbRotXCon.Text = hv_RotX.ToString & "°"
            txbRotYCon.Text = hv_RotY.ToString & "°"
            txbRotZCon.Text = hv_RotZ.ToString & "°"

        End If

    End Sub

    Private Sub btnFileCamPose_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnFileCamPose.Click
        ofdCamRob.InitialDirectory = "C:\Documents and
Settings\acro\Desktop\SmartFactory\Calibration\Hand-eye calibration\Backup Results"
        ofdCamRob.Filter = "dat files (*.dat)|*.dat|All files (*.*)|*.*"
        ofdCamRob.FilterIndex = 1
        ofdCamRob.RestoreDirectory = True
        ofdCamRob.ShowDialog()

        If Len(ofdCamRob.FileName) = 0 Then
            'Do nothing
        Else
            CamRobName = Split(ofdCamRob.SafeFileName, ".")

            If CamRobName(CamRobName.Length - 1) = "dat" Then
                txbCamRobPos.Text = ofdCamRob.SafeFileName()
                CamRobDirectory = (ofdCamRob.FileName())

            Else
                txbCamRobPos.Text = "File extension invalid, charge again"
            End If
        End If
    End Sub

    Private Sub btnSend_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSend.Click
        If Poseconverted = True Then
            Call RunHalconSend()
            Call MessageBox.Show("Pose sent succesfully", Me.Text,
MessageBoxButtons.OK)
```

```
        Else
            Call MessageBox.Show("Pose has to be converted", Me.Text,
MessageBoxButtons.OK)
        End If

    End Sub



    Private Sub btnDisconnect_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnDisconnect.Click
        Call RunHalconDisconnect()
        txbIP.Enabled = True
        txbPort.Enabled = True
        btnDisconnect.Enabled = False
        btnConnect.Enabled = True
        btnSend.Enabled = False
        txbStatus.Text = "Disconnected"


    End Sub

    Private Sub btnConnect_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnConnect.Click

        If Len(txbPort.Text) <> 0 And Len(txbIP.Text) <> 0 Then
            Dim IPstring As String
            IPstring = txbIP.Text
            ip = txbIP.Text
            Port = Convert.ToInt32(txbPort.Text)
            Dim IPParts() As String

            IPParts = IPstring.Split(".")
            If IPParts.Length = 4 Then
                If IsNumeric(CInt(IPParts(0)) > 0 And CInt(IPParts(0)) < 255) Then
                    If IsNumeric(CInt(IPParts(1)) > 0 And CInt(IPParts(1)) < 255)
Then
                        If IsNumeric(CInt(IPParts(2)) > 0 And CInt(IPParts(2)) < 255)
Then
                            If IsNumeric(CInt(IPParts(3)) > 0 And CInt(IPParts(3)) <
255) Then
                                'Run the procedure of connection
                                Call RunHalconConnect()
                                txbIP.Enabled = False
                                txbPort.Enabled = False
                                btnDisconnect.Enabled = True
                                btnConnect.Enabled = False
                                btnSend.Enabled = True
                                txbStatus.Text = "Connected"
                                IpMemo = txbIP.Text
                                PortMemo = txbPort.Text


                            Else
                                Call MessageBox.Show("IP is not correct", Me.Text,
MessageBoxButtons.OK)
                            End If
                        Else
                            Call MessageBox.Show("IP is not correct", Me.Text,
MessageBoxButtons.OK)
                        End If
                    Else
                        Call MessageBox.Show("IP is not correct", Me.Text,
MessageBoxButtons.OK)
                    End If
                Else
                    Call MessageBox.Show("IP is not correct", Me.Text,
MessageBoxButtons.OK)
                End If
            Else
                Call MessageBox.Show("IP is not correct", Me.Text,
MessageBoxButtons.OK)
            End If
        Else
            Call MessageBox.Show("Introduce correct IP and Port", Me.Text,
MessageBoxButtons.OK)
        End If


    End Sub

    Private Sub btnSavePose_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSavePose.Click
        If Poseconverted = True Then
            sfdSavePose.Filter = "DAT Files (*.dat*)|*.dat"
            sfdSavePose.InitialDirectory = "C:\Documents and
Settings\acro\Desktop\SmartFactory\matching\Cylinder\Result Poses"
            sfdSavePose.ShowDialog()
            If sfdSavePose.FileName <> "" Then
                If sfdSavePose.ShowDialog = Windows.Forms.DialogResult.OK Then
                    FinalPoseDirectory = (sfdSavePose.FileName())
                    RunHalconSavePose()
                    Call MessageBox.Show("Pose saved succesfully", Me.Text,
MessageBoxButtons.OK)
                End If
            End If
        Else
            Call MessageBox.Show("Pose has to be converted", Me.Text,
MessageBoxButtons.OK)
        End If
    End Sub


    'Menu Strip Declaration
    Private Sub mnsOpendSTLModel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsOpenCamRobPose.Click
        Call btnFileCamPose_Click(sender, e)
    End Sub

    Private Sub mnsConnect_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsConnect.Click
        Call btnConnect_Click(sender, e)
    End Sub

    Private Sub mnsDisconnect_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsDisconnect.Click
        Call btnDisconnect_Click(sender, e)
    End Sub

    Private Sub mnsConvert_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsConvert.Click
        Call btnConvert_Click(sender, e)
    End Sub

    Private Sub mnsSave_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsSave.Click
        Call btnSavePose_Click(sender, e)
    End Sub
    Private Sub mnsSend_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsSend.Click
        Call btnSend_Click(sender, e)
    End Sub

    Private Sub mnsExit_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mnsExit.Click
        Call btnClose_Click(sender, e)
    End Sub
End Class
```

A6.10. Module1.vb

```
Imports HalconDotNet
Imports System.IO


Module Module1

    'Global Variables Declaration

    'Open Files from folders
    Public ShapeModelDirectory As HTuple = New HTuple
    Public CamParamDirectory As HTuple = New HTuple
    Public CamRobDirectory As HTuple = New HTuple
    Public ModelDirectory As HTuple = New HTuple
    Public FinalPoseDirectory As HTuple = New HTuple

    'Control the creation of 3D Shape Model
    Public ShapeRead As Boolean = False
    Public hv_ShapeModel3DID As HTuple = New HTuple
    Public ShapeCreated As Boolean = False
    'Control the send of matching poses to robot
    Public CylinPoseMatch As Boolean = False

    'Matching Results Control
    Public hv_PoseTmp1 As HTuple = New HTuple
    Public hv_NumPieces As HTuple = New HTuple

    'Show Matching Results in window
    Public hv_Pose_Cylinder_in_RobBase As HTuple = New HTuple
    Public hv_PosX As HTuple = New HTuple, hv_RotZ As HTuple = New HTuple
    Public hv_PosY As HTuple = New HTuple, hv_PosZ As HTuple = New HTuple
    Public hv_RotX As HTuple = New HTuple, hv_RotY As HTuple = New HTuple

    'Communication
    Public ip As HTuple = New HTuple, Port As Integer
    Public Status As Boolean = False
    Public Socket As HTuple = New HTuple
    Public IpMemo As String = "10.10.107.25"
    Public PortMemo As String = "9901"

End Module
```