```python
from pycm import *
import time
import math

class CVar:
	###################################
	IsTorqOn                                = False
	nScreenWidth                            = 0
	nScreenHeight                           = 0

	nTouch_XY_X0                            = 0
	nTouch_XY_Y0                            = 0
	nTouch_Pos0                             = 0
	nTouch_Pos_X0                           = 0
	nTouch_Pos_Y0                           = 0
	nTouch_XY_X1                            = 0
	nTouch_XY_Y1                            = 0
	nTouch_Pos1                             = 0
	nTouch_Pos_X1                           = 0
	nTouch_Pos_Y1                           = 0
	nButton_0                               = -1
	nButton_1                               = -1
	nBack_Background                        = 0

	btn0                                    = None
	btn1                                    = None

	nPage                                   = -1
	nPage_Prev                              = -1
	###################################
	#fYaw_first                               = 0.0
	#fYaw                                     = 0.0
	#fYaw_turn                                = 0.0
	###################################
	nPoint_L                                = 2
	IsWalking                               = False
	nDemoMode                               = 0 # 0 : None, 1 : Demo Motion
	nDemoIndex                              = 0
class CTimer:
	nTimer                                  = 0
	IsTimer                                 = False

	def __init__(self):
		self.nTimer = 0
		self.IsTimer = 0

	def Set(self):
		self.IsTimer = True
```

```
                self.nTimer = millis()

        def Get(self):
                if self.IsTimer :
                        return millis() - self.nTimer
                return 0

        def Destroy(self):
                self.IsTimer = False

_INIT_FX = 180#160
_INIT_FY = 0
```

Page 2

```

# The higher the number, the stronger the grip.
_LOAD = 50 # Grasp force in control mode
_LOAD_DEMO = 50 # Grab force in demo mode

#m_fX = 180.475251645602
m_fX = _INIT_FX#100.475251645602
m_fY = _INIT_FY#-66.544944861224

_IDS_ARM = [9, 10, 16]
#_IDS_HAND = [15, 13, 14]
_IDS_HAND = [15, 13, 14]

#                       Time, 9, 10, 16
m_afArm = [ 100, 0, 0, 0]
#                       Time, 10, 16, 15
m_afManipulator = [ 100, 0, 0, 0]
#                       Time, 15(Tilt), 13, 14
#m_afHand = [50,                        0, 0, 0 ]
m_afHand = [ 200 ,                     0, 0, 0 ]

#                           time, 1, ..                                                 .. 10, 13, 14, 15,
16
_MNT_READY = [ 1000, -40, -30, 40, -30, 30, -50, -30, -50, 0]#, -65, -20, 20, 75,
-90]

#                           time, 1, ..
#_MNT_DEMO_STEP_0 = [ 1000, -40, -10, 40, -50, 30, -70, -30, -30, 0]
#_MNT_DEMO_STEP_1 = [ 1000, -40, -50, 40, -10, 30, -30, -30, -70, 0]
#_MNT_DEMO_STEP_2 = [ 1000, -40, -30, 40, -30, 30, -50, -30, -50, 0]

nDemoV = 30
fDemoV_Per = 0.85
nDemoH = 30
nDemoH_Default = 30
nDemoStepTime = 300
_MNT_DEMO_STEP_Base = [ nDemoStepTime, -40                      , -nDemoH_Default, 40                      ,
-nDemoH_Default, 30                      , -nDemoH_Default-20, -30                      , -nDemoH_Default-20,
0                       ]
_MNT_DEMO_STEP_0 = [ nDemoStepTime, -40 + nDemoV, -nDemoH_Default, 40 + nDemoV,
-nDemoH_Default, 30 + nDemoV, -nDemoH_Default-20, -30 + nDemoV, -nDemoH_Default-20,
nDemoV * fDemoV_Per]
_MNT_DEMO_STEP_1 = [ nDemoStepTime, -40 - nDemoV, -nDemoH_Default, 40 - nDemoV,
-nDemoH_Default, 30 - nDemoV, -nDemoH_Default-20, -30 - nDemoV, -nDemoH_Default-20, -
```

```
nDemoV * fDemoV_Per]
 98 _MNT_DEMO = [
 99             _MNT_DEMO_STEP_0,
100             _MNT_DEMO_STEP_Base,
101             _MNT_DEMO_STEP_1,
102             _MNT_DEMO_STEP_Base
103 ]
104
105 _MNT_DEMO2_STEP_Base= [ nDemoStepTime, -40                                  , -nDemoH_Default+nDemoH, 40
, -nDemoH_Default+nDemoH, 30                                  , -nDemoH_Default-20+nDemoH, -30                                  , -
nDemoH_Default-20+nDemoH, 0                                  ]
106 _MNT_DEMO2_STEP_0 = [ nDemoStepTime, -40 + nDemoV, -nDemoH_Default+nDemoH, 40 +
nDemoV, -nDemoH_Default+nDemoH, 30 + nDemoV, -nDemoH_Default-20+nDemoH, -30 + nDemoV, -
nDemoH_Default-20+nDemoH, nDemoV * fDemoV_Per]
107 _MNT_DEMO2_STEP_1 = [ nDemoStepTime, -40 - nDemoV, -nDemoH_Default+nDemoH, 40 -
nDemoV, -nDemoH_Default+nDemoH, 30 - nDemoV, -nDemoH_Default-20+nDemoH, -30 - nDemoV, -
```

Page 3

```
nDemoH_Default-20+nDemoH, -nDemoV * fDemoV_Per]
108
109 _MNT_DEMO2 = [
110             _MNT_DEMO2_STEP_1,
111             _MNT_DEMO2_STEP_Base,
112             _MNT_DEMO2_STEP_0,
113             _MNT_DEMO2_STEP_Base
114 ]
115 lstDemo = None
116
117 _MNT_READY_ARM = [ 1000,                                  0, -65, -20, 20, 75,
-90]
118
119 #_MNT_F_STEP0 = [ 100, -40, -30, 50, -25, 10, -35, -30, -50, 0]
120 #_MNT_F_STEP1 = [ 100, -50, -30, 40, -30, 20, -35, -30, -50, 0]
121 #_MNT_F_STEP2 = [ 100, -60, -30, 30, -30, 30, -50, 0, -30, 0]
122 #_MNT_F_STEP3 = [ 100, -50, -25, 40, -30, 30, -50, -10, -35, 0]
123 #_MNT_F_STEP4 = [ 100, -40, -30, 50, -30, 30, -50, -20, -35, 0]
124 #_MNT_F_STEP5 = [ 100, -30, -30, 60, -30, 0, -30, -30, -30, 0]
125
126 _MOTION = []
127 def MotionSet(nDirection):
128             global _MOTION
129             #nTime_Strt = 250
130             #nTime_Scnd = 500
131             #nTime_Dely = 100
132             #nTime_Walk = 150
133             if (nDirection == 0): # F - Left
134                     nTime = 300 / (nSpeed + 1) # - - + - - - + - 0
135                     _MNT_F_STEP0 = [ nTime, -15, -50, 45, -50, 10, -20, -45, -30, 0 ]
136                     _MNT_F_STEP1 = [ nTime, -20, -30, 50, -50, 10, -10, -40, -50, 0 ]
137                     _MNT_F_STEP2 = [ nTime, -25, -30, 50, -50, 20, -10, -40, -40, 0 ]
138                     _MNT_F_STEP3 = [ nTime, -30, -30, 50, -50, 30, -10, -40, -50, 0 ]
139                     _MNT_F_STEP4 = [ nTime, -35, -30, 50, -50, 40, -10, -25, -40, 0 ]
140                     _MNT_F_STEP5 = [ nTime, -40, -30, 50, -50, 50, -10, -10, -30, 0 ]
141                     _MNT_F_STEP6 = [ nTime, -45, -50, 15, -50, 45, -30, -10, -20, 0 ]
142                     _MNT_F_STEP7 = [ nTime, -50, -50, 20, -30, 40, -50, -10, -10, 0 ]
143                     _MNT_F_STEP8 = [ nTime, -40, -50, 25, -30, 40, -40, -20, -10, 0 ]
144                     _MNT_F_STEP9 = [ nTime, -30, -50, 30, -30, 40, -50, -30, -10, 0 ]
```

```
145                 _MNT_F_STEP10 = [ nTime, -20, -50, 35, -30, 25, -40, -40, -10, 0 ]
146                 _MNT_F_STEP11 = [ nTime, -10, -50, 40, -30, 10, -30, -50, -10, 0 ]
147                 _MOTION = [
148                         _MNT_F_STEP0,
149                         _MNT_F_STEP1,
150                         _MNT_F_STEP2,
151                         _MNT_F_STEP3,
152                         _MNT_F_STEP4,
153                         _MNT_F_STEP5,
154                         _MNT_F_STEP6,
155                         _MNT_F_STEP7,
156                         _MNT_F_STEP8,
157                         _MNT_F_STEP9,
158                         _MNT_F_STEP10,
159                         _MNT_F_STEP11
160                 ]
161         elif (nDirection == 1):
162                 nTime = 300 / (nSpeed + 1) # - - + - - - + - 0
163                 _MNT_F_STEP0 = [ nTime, -15, -50, 45, -50, 10, -20, -45, -30, 0 ]
164                 _MNT_F_STEP1 = [ nTime, -20, -30, 50, -50, 10, -10, -40, -50, 0 ]
165                 _MNT_F_STEP2 = [ nTime, -25, -30, 40, -50, 20, -10, -40, -40, 0 ]
166                 _MNT_F_STEP3 = [ nTime, -30, -30, 30, -50, 30, -10, -40, -50, 0 ]
```



```
167                 _MNT_F_STEP4 = [ nTime, -35, -30, 20, -50, 40, -10, -25, -40, 0 ]
168                 _MNT_F_STEP5 = [ nTime, -40, -30, 10, -50, 50, -10, -10, -30, 0 ]
169                 _MNT_F_STEP6 = [ nTime, -45, -50, 15, -50, 45, -30, -10, -20, 0 ]
170                 _MNT_F_STEP7 = [ nTime, -50, -50, 20, -30, 40, -50, -10, -10, 0 ]
171                 _MNT_F_STEP8 = [ nTime, -40, -50, 25, -30, 40, -40, -20, -10, 0 ]
172                 _MNT_F_STEP9 = [ nTime, -30, -50, 30, -30, 40, -50, -30, -10, 0 ]
173                 _MNT_F_STEP10 = [ nTime, -20, -50, 35, -30, 25, -40, -40, -10, 0 ]
174                 _MNT_F_STEP11 = [ nTime, -10, -50, 40, -30, 10, -30, -50, -10, 0 ]
175                 _MOTION = [
176                         _MNT_F_STEP0,
177                         _MNT_F_STEP1,
178                         _MNT_F_STEP2,
179                         _MNT_F_STEP3,
180                         _MNT_F_STEP4,
181                         _MNT_F_STEP5,
182                         _MNT_F_STEP6,
183                         _MNT_F_STEP7,
184                         _MNT_F_STEP8,
185                         _MNT_F_STEP9,
186                         _MNT_F_STEP10,
187                         _MNT_F_STEP11
188                 ]
189                 '''
190                 _MNT_F_STEP0 = [ 300, -20, -30, 50, -50, 10, -10, -40, -50, 0 ]
191                 _MNT_F_STEP1 = [ 300, -40, -30, 10, -50, 50, -10, -40, -50, 0 ]
192                 _MNT_F_STEP2 = [ 300, -40, -30, 10, -50, 50, -10, -10, -30, 0 ]
193                 _MNT_F_STEP3 = [ 300, -50, -50, 20, -30, 40, -50, -10, -10, 0 ]
194                 _MNT_F_STEP4 = [ 300, -10, -50, 40, -30, 40, -50, -50, -10, 0 ]
195                 _MNT_F_STEP5 = [ 300, -10, -50, 40, -30, 10, -30, -50, -10, 0 ]
196                 _MOTION = [
197                         _MNT_F_STEP0,
198                         _MNT_F_STEP1,
199                         _MNT_F_STEP2,
200                         _MNT_F_STEP3,
```

```
                        _MNT_F_STEP4,
                        _MNT_F_STEP5
                ]
                '''
                '''
                _MNT_F_STEP0 = [ nTime_Strt, -30, -30, 30, -30, -30, -30, 30, -30, 0 ]
                _MNT_F_STEP1 = [ nTime_Walk, -30, -30, 45, -50, -30, -30, 30, -30, 0 ]
                _MNT_F_STEP2 = [ nTime_Walk, -30, -30, 60, -30, -30, -30, 30, -30, 0 ]
                _MNT_F_STEP3 = [ nTime_Walk, -45, -50, 60, -30, -30, -30, 30, -30, 0 ]
                _MNT_F_STEP4 = [ nTime_Walk, -60, -30, 60, -30, -30, -30, 30, -30, 0 ]
                _MNT_F_STEP5 = [ nTime_Scnd, -10, -30, 10, -30, 30, -50, -30, -50, 0 ]
                _MNT_F_STEP6 = [ nTime_Dely, -10, -30, 10, -30, 30, -50, -30, -50, 0 ]
                _MNT_F_STEP7 = [ nTime_Walk, -10, -30, 10, -30, 10, -70, -30, -50, 0 ]
                _MNT_F_STEP8 = [ nTime_Walk, -10, -30, 10, -30, -10, -50, -30, -50, 0 ]
                _MNT_F_STEP9 = [ nTime_Walk, -10, -30, 10, -30, -10, -50, -10, -70, 0 ]
                _MNT_F_STEP10= [ nTime_Walk, -10, -30, 10, -30, -10, -50, 10, -50, 0 ]
                _MOTION = [
                        _MNT_F_STEP0,
                        _MNT_F_STEP1,
                        _MNT_F_STEP2,
                        _MNT_F_STEP3,
                        _MNT_F_STEP4,
                        _MNT_F_STEP5,
                        _MNT_F_STEP6,
                        _MNT_F_STEP7,
                        _MNT_F_STEP8,
                        _MNT_F_STEP9,
```



```
                        _MNT_F_STEP10
                ]
                '''
        elif (nDirection == 2): # F - Right
                nTime = 300 / (nSpeed + 1) # - - + - - - + - 0
                _MNT_F_STEP0 = [ nTime, -15, -50, 45, -50, 10, -20, -45, -30, 0 ]
                _MNT_F_STEP1 = [ nTime, -20, -30, 50, -50, 10, -10, -40, -50, 0 ]
                _MNT_F_STEP2 = [ nTime, -25, -30, 40, -50, 20, -10, -40, -40, 0 ]
                _MNT_F_STEP3 = [ nTime, -30, -30, 30, -50, 30, -10, -40, -50, 0 ]
                _MNT_F_STEP4 = [ nTime, -35, -30, 20, -50, 40, -10, -25, -40, 0 ]
                _MNT_F_STEP5 = [ nTime, -40, -30, 10, -50, 50, -10, -10, -30, 0 ]

                _MNT_F_STEP6 = [ nTime, -45, -50, 15, -50, 45, -30, -10, -20, 0 ]
                _MNT_F_STEP7 = [ nTime, -50, -50, 20, -30, 40, -50, -10, -10, 0 ]
                _MNT_F_STEP8 = [ nTime, -50, -50, 25, -30, 40, -40, -20, -10, 0 ]
                _MNT_F_STEP9 = [ nTime, -50, -50, 30, -30, 40, -50, -30, -10, 0 ]
                _MNT_F_STEP10 = [ nTime, -50, -50, 35, -30, 25, -40, -40, -10, 0 ]
                _MNT_F_STEP11 = [ nTime, -50, -50, 40, -30, 10, -30, -50, -10, 0 ]
                _MOTION = [
                        _MNT_F_STEP0,
                        _MNT_F_STEP1,
                        _MNT_F_STEP2,
                        _MNT_F_STEP3,
                        _MNT_F_STEP4,
                        _MNT_F_STEP5,
                        _MNT_F_STEP6,
                        _MNT_F_STEP7,
                        _MNT_F_STEP8,
                        _MNT_F_STEP9,
```

```
                    _MNT_F_STEP10,
                    _MNT_F_STEP11
                ]
        elif (nDirection == 3):
                _MNT_TL_STEP0 = [100, -55, -40, 55, -30, 45, -50, -45, -60, 0]
                _MNT_TL_STEP1 = [ 100, -70, -30, 70, -30, 60, -50, -60, -50, 0]
                _MNT_TL_STEP2 = [ 100, -55, -30, 55, -40, 45, -60, -45, -50, 0]
                _MNT_TL_STEP3 = [ 100, -40, -30, 40, -30, 30, -50, -30, -50, 0]
                _MOTION = [
                    _MNT_TL_STEP0,
                    _MNT_TL_STEP1,
                    _MNT_TL_STEP2,
                    _MNT_TL_STEP3
                ]
        elif (nDirection == 5):
                _MNT_TR_STEP0 = [ 100, -55, -30, 55, -40, 45, -60, -45, -50, 0]
                _MNT_TR_STEP1 = [ 100, -70, -30, 70, -30, 60, -50, -60, -50, 0]
                _MNT_TR_STEP2 = [ 100, -55, -40, 55, -30, 45, -50, -45, -60, 0]
                _MNT_TR_STEP3 = [ 100, -40, -30, 40, -30, 30, -50, -30, -50, 0]
                _MOTION = [
                    _MNT_TR_STEP0,
                    _MNT_TR_STEP1,
                    _MNT_TR_STEP2,
                    _MNT_TR_STEP3
                ]
        elif (nDirection == 6): # B - Left
                nTime = 300 / (nSpeed + 1) # - - + - - - + - 0
                _MNT_F_STEP0 = [ nTime, -15, -50, 45, -50, 10, -20, -45, -30, 0 ]
                _MNT_F_STEP1 = [ nTime, -20, -30, 50, -50, 10, -10, -40, -50, 0 ]
                _MNT_F_STEP2 = [ nTime, -25, -30, 50, -50, 20, -10, -40, -40, 0 ]
                _MNT_F_STEP3 = [ nTime, -30, -30, 50, -50, 30, -10, -40, -50, 0 ]
                _MNT_F_STEP4 = [ nTime, -35, -30, 50, -50, 40, -10, -25, -40, 0 ]
```



```
                _MNT_F_STEP5 = [ nTime, -40, -30, 50, -50, 50, -10, -10, -30, 0 ]
                _MNT_F_STEP6 = [ nTime, -45, -50, 15, -50, 45, -30, -10, -20, 0 ]
                _MNT_F_STEP7 = [ nTime, -50, -50, 20, -30, 40, -50, -10, -10, 0 ]
                _MNT_F_STEP8 = [ nTime, -40, -50, 25, -30, 40, -40, -20, -10, 0 ]
                _MNT_F_STEP9 = [ nTime, -30, -50, 30, -30, 40, -50, -30, -10, 0 ]
                _MNT_F_STEP10 = [ nTime, -20, -50, 35, -30, 25, -40, -40, -10, 0 ]
                _MNT_F_STEP11 = [ nTime, -10, -50, 40, -30, 10, -30, -50, -10, 0 ]
                _MOTION = [
                    _MNT_F_STEP11,
                    _MNT_F_STEP10,
                    _MNT_F_STEP9,
                    _MNT_F_STEP8,
                    _MNT_F_STEP7,
                    _MNT_F_STEP6,
                    _MNT_F_STEP5,
                    _MNT_F_STEP4,
                    _MNT_F_STEP3,
                    _MNT_F_STEP2,
                    _MNT_F_STEP1,
                    _MNT_F_STEP0
                ]
        elif (nDirection == 7):
                nTime = 300 / (nSpeed + 1) # - - + - - - + - 0
                _MNT_F_STEP0 = [ nTime, -15, -50, 45, -50, 10, -20, -45, -30, 0 ]
```

```
313                 _MNT_F_STEP1 = [ nTime, -20, -30, 50, -50, 10, -10, -40, -50, 0 ]
314                 _MNT_F_STEP2 = [ nTime, -25, -30, 40, -50, 20, -10, -40, -40, 0 ]
315                 _MNT_F_STEP3 = [ nTime, -30, -30, 30, -50, 30, -10, -40, -50, 0 ]
316                 _MNT_F_STEP4 = [ nTime, -35, -30, 20, -50, 40, -10, -25, -40, 0 ]
317                 _MNT_F_STEP5 = [ nTime, -40, -30, 10, -50, 50, -10, -10, -30, 0 ]
318                 _MNT_F_STEP6 = [ nTime, -45, -50, 15, -50, 45, -30, -10, -20, 0 ]
319                 _MNT_F_STEP7 = [ nTime, -50, -50, 20, -30, 40, -50, -10, -10, 0 ]
320                 _MNT_F_STEP8 = [ nTime, -40, -50, 25, -30, 40, -40, -20, -10, 0 ]
321                 _MNT_F_STEP9 = [ nTime, -30, -50, 30, -30, 40, -50, -30, -10, 0 ]
322                 _MNT_F_STEP10 = [ nTime, -20, -50, 35, -30, 25, -40, -40, -10, 0 ]
323                 _MNT_F_STEP11 = [ nTime, -10, -50, 40, -30, 10, -30, -50, -10, 0 ]
324                 _MOTION = [
325                         _MNT_F_STEP11,
326                         _MNT_F_STEP10,
327                         _MNT_F_STEP9,
328                         _MNT_F_STEP8,
329                         _MNT_F_STEP7,
330                         _MNT_F_STEP6,
331                         _MNT_F_STEP5,
332                         _MNT_F_STEP4,
333                         _MNT_F_STEP3,
334                         _MNT_F_STEP2,
335                         _MNT_F_STEP1,
336                         _MNT_F_STEP0
337                 ]
338                 '''
339                 _MNT_B_STEP0 = [ nTime_Strt, -10, -30, 10, -30, -10, -50, 10, -50, 0 ]
340                 _MNT_B_STEP1 = [ nTime_Walk, -10, -30, 10, -30, -10, -50, -10, -70, 0 ]
341                 _MNT_B_STEP2 = [ nTime_Walk, -10, -30, 10, -30, -10, -50, -30, -50, 0 ]
342                 _MNT_B_STEP3 = [ nTime_Walk, -10, -30, 10, -30, 10, -70, -30, -50, 0 ]
343                 _MNT_B_STEP4 = [ nTime_Walk, -10, -30, 10, -30, 30, -50, -30, -50, 0 ]
344                 _MNT_B_STEP5 = [ nTime_Scnd, -60, -30, 60, -30, -30, -30, 30, -30, 0 ]
345                 _MNT_B_STEP6 = [ nTime_Dely, -60, -30, 60, -30, -30, -30, 30, -30, 0 ]
346                 _MNT_B_STEP7 = [ nTime_Walk, -45, -50, 60, -30, -30, -30, 30, -30, 0 ]
347                 _MNT_B_STEP8 = [ nTime_Walk, -30, -30, 60, -30, -30, -30, 30, -30, 0 ]
348                 _MNT_B_STEP9 = [ nTime_Walk, -30, -30, 45, -50, -30, -30, 30, -30, 0 ]
349                 _MNT_B_STEP10= [ nTime_Walk, -30, -30, 30, -30, -30, -30, 30, -30, 0 ]
```



```
350
351                 _MOTION = [
352                         _MNT_B_STEP0,
353                         _MNT_B_STEP1,
354                         _MNT_B_STEP2,
355                         _MNT_B_STEP3,
356                         _MNT_B_STEP4,
357                         _MNT_B_STEP5,
358                         _MNT_B_STEP6,
359                         _MNT_B_STEP7,
360                         _MNT_B_STEP8,
361                         _MNT_B_STEP9,
362                         _MNT_B_STEP10
363                 ]
364                 '''
365         elif (nDirection == 8): # B - Right
366                 nTime = 300 / (nSpeed + 1) # - - + - - - + - 0
367                 _MNT_F_STEP0 = [ nTime, -15, -50, 45, -50, 10, -20, -45, -30, 0 ]
368                 _MNT_F_STEP1 = [ nTime, -20, -30, 50, -50, 10, -10, -40, -50, 0 ]
```

```
                _MNT_F_STEP2 = [ nTime, -25, -30, 40, -50, 20, -10, -40, -40, 0 ]
                _MNT_F_STEP3 = [ nTime, -30, -30, 30, -50, 30, -10, -40, -50, 0 ]
                _MNT_F_STEP4 = [ nTime, -35, -30, 20, -50, 40, -10, -25, -40, 0 ]
                _MNT_F_STEP5 = [ nTime, -40, -30, 10, -50, 50, -10, -10, -30, 0 ]

                _MNT_F_STEP6 = [ nTime, -45, -50, 15, -50, 45, -30, -10, -20, 0 ]
                _MNT_F_STEP7 = [ nTime, -50, -50, 20, -30, 40, -50, -10, -10, 0 ]
                _MNT_F_STEP8 = [ nTime, -50, -50, 25, -30, 40, -40, -20, -10, 0 ]
                _MNT_F_STEP9 = [ nTime, -50, -50, 30, -30, 40, -50, -30, -10, 0 ]
                _MNT_F_STEP10 = [ nTime, -50, -50, 35, -30, 25, -40, -40, -10, 0 ]
                _MNT_F_STEP11 = [ nTime, -50, -50, 40, -30, 10, -30, -50, -10, 0 ]
                _MOTION = [
                        _MNT_F_STEP11,
                        _MNT_F_STEP10,
                        _MNT_F_STEP9,
                        _MNT_F_STEP8,
                        _MNT_F_STEP7,
                        _MNT_F_STEP6,
                        _MNT_F_STEP5,
                        _MNT_F_STEP4,
                        _MNT_F_STEP3,
                        _MNT_F_STEP2,
                        _MNT_F_STEP1,
                        _MNT_F_STEP0
                ]

_COLOR_NONE                     = 0
_COLOR_WHITE                    = 1
_COLOR_BLACK                    = 2
_COLOR_RED                      = 3
_COLOR_GREEN                    = 4
_COLOR_BLUE                     = 5
_COLOR_YELLOW = 6
_COLOR_GRAY_LIGHT = 7
_COLOR_GRAY                     = 8
_COLOR_GRAY_DARK = 9

_SHOW_IMAGE                     = 0
_SHOW_TEXT                      = 1
_SHOW_SHAPE                     = 2
_SHOW_NUM                       = 3
```

Page 8

```
_RATIO                          = 1000

_BTN_INDEX                      = 4

#1..5
#[left,top,right,bottom, 고유 번호(ButtonNumber)] : left, top < 0 then 1..5 position
# Put the coordinates of each button and the unique number you want to put here-step 1/2 [Note: ButtonNumber is different
Do not overlap with buttons.]

_BTN_ARM_F                      = [610,220,700,356,30]
_BTN_ARM_B                      = [610,400,700,530,31]
_BTN_ARM_ORG                    = [736,310,826,440,32]
_BTN_ARM_UP                     = [860,220,950,356,33]
_BTN_ARM_DN                     = [860,400,950,530,34]
```

```
424
425 _BTN_ARM_LEFT = [672,550,766,678,35]
426 _BTN_ARM_RIGHT = [800,550,892,678,36]
427
428 _BTN_HAND_IN                = [610,802,700,936,37]
429 _BTN_HAND_ORG = [736,802,826,936,38]
430 _BTN_HAND_OUT = [860,802,950,936,39]
431
432 #[Page 1]
433 #go button
434 _BTN_TURN_L                 = [30,320,100,440,6]
435 _BTN_TURN_R                 = [240,310,310,440,7]
436 _BTN_MOVE_UL                = [40,490,110,630,12]
437 _BTN_MOVE_U                 = [140,390,200,530,13]
438 _BTN_MOVE_UR                = [230,490,300,630,14]
439 _BTN_MOVE_DL                = [40,700,110,830,17]
440 _BTN_MOVE_D                 = [140,800,200,930,18]
441 _BTN_MOVE_DR                = [230,700,300,830,19]
442 _BTN_MOVE_X                 = [120,560,220,770,20]
443
444 #Torque ON/OFF button
445 _BTN_TORQ                   = [20,40,160,150,21]
446
447 #Demo Button
448 _BTN_DEMO                   = [830,40,970,150,22]
449
450 #Motion speed button (attached consecutive buttons placed from top to bottom)
451 _BTN_SPD_TOP                = 450 # TOP pressed Y coordinate of gauge
452 _BTN_SPD_BOTTM = 900 # BOTTOM pressed Y coordinate of gauge
453 _BTN_SPD_X                  = 458 # Center X coordinate of gauge
454 _BTN_SPD_W                  = 60 # left and right width of the button to be pressed
455 _BTN_COUNT                  = 5 # number of buttons
456 # Automatic calculation of button position from here
457 nSpdBtnW                    = int(_BTN_SPD_W / 2)
458 nSpdBtnH                    = int((_BTN_SPD_BOTTM - _BTN_SPD_TOP) / (_BTN_COUNT - 1) / 2)
459 nTmp = 0
460 _BTN_SPD_L5                 = [_BTN_SPD_X - nSpdBtnW, _BTN_SPD_TOP + (nSpdBtnH * nTmp)*2 -
nSpdBtnH, _BTN_SPD_X + nSpdBtnW, _BTN_SPD_TOP + (nSpdBtnH * (nTmp + 1))*2 + nSpdBtnH - 1,
1]
461 nTmp = 1
462 _BTN_SPD_L4                 = [_BTN_SPD_X - nSpdBtnW, _BTN_SPD_TOP + (nSpdBtnH * nTmp)*2 -
nSpdBtnH, _BTN_SPD_X + nSpdBtnW, _BTN_SPD_TOP + (nSpdBtnH * (nTmp + 1))*2 + nSpdBtnH - 1,
1]
463 nTmp = 2
464 _BTN_SPD_L3                 = [_BTN_SPD_X - nSpdBtnW, _BTN_SPD_TOP + (nSpdBtnH * nTmp)*2 -
nSpdBtnH, _BTN_SPD_X + nSpdBtnW, _BTN_SPD_TOP + (nSpdBtnH * (nTmp + 1))*2 + nSpdBtnH - 1,
```



```
1]
465 nTmp = 3
466 _BTN_SPD_L2                 = [_BTN_SPD_X - nSpdBtnW, _BTN_SPD_TOP + (nSpdBtnH * nTmp)*2 -
nSpdBtnH, _BTN_SPD_X + nSpdBtnW, _BTN_SPD_TOP + (nSpdBtnH * (nTmp + 1))*2 + nSpdBtnH - 1,
1]
467 nTmp = 4
468 _BTN_SPD_L1                 = [_BTN_SPD_X - nSpdBtnW, _BTN_SPD_TOP + (nSpdBtnH * nTmp)*2 -
nSpdBtnH, _BTN_SPD_X + nSpdBtnW, _BTN_SPD_TOP + (nSpdBtnH * (nTmp + 1))*2 + nSpdBtnH - 1,
1]
```

```
# _BTN_SPD_L5                    = [410,450,500,540-1,1]
# _BTN_SPD_L4                    = [410,540,500,630-1,2]
# _BTN_SPD_L3                    = [410,630,500,720-1,3]
# _BTN_SPD_L2                    = [410,720,500,810-1,4]
# _BTN_SPD_L1                    = [410,810,500,900,5]

#_PAGE_MENU                              = 0
_PAGE_MAIN                              = 1
#_PAGE_MOTOR_TEST                       = 4
#_PAGE_OFFSET                           = 5
#_PAGE_OFFSET_DIALOG1                   = 6
#_PAGE_OFFSET_DIALOG2                   = 7
#_PAGE_OFFSET_DIALOG_CLOSE = 8

btnList                                 = None
tmrTorqBtn                              = CTimer()

IsPhone                                 = False
nSpeed                                  = 0
def DelayForRun(nMilliSecond):
        Wait(int(nMilliSecond / CVar.nPoint_L))

def ShowPage(nPage):
        global btnList
        global nSpeed
        CVar.nPage_Prev = CVar.nPage
        CVar.nPage = nPage

        # Create a page here and assign the button to it. -step 2/2
        if nPage == _PAGE_MAIN:
                Clear_All()

                btnList = [
                        _BTN_TORQ,
                        _BTN_DEMO,
                        #Arm action button
                        _BTN_ARM_F,
                        _BTN_ARM_B,
                        _BTN_ARM_ORG,
                        _BTN_ARM_UP,
                        _BTN_ARM_DN,
                        _BTN_ARM_LEFT,
                        _BTN_ARM_RIGHT,
                        _BTN_HAND_IN,
                        _BTN_HAND_ORG,
                        _BTN_HAND_OUT,
                        #Go button
                        _BTN_TURN_L,
                        _BTN_TURN_R,
                        _BTN_MOVE_UL,
                        _BTN_MOVE_U,
```



```
                        _BTN_MOVE_UR,
                        _BTN_MOVE_DL,
                        _BTN_MOVE_D,
                        _BTN_MOVE_DR,
```

```
                        _BTN_MOVE_X,
                        #Motion speed button
                        _BTN_SPD_L1,
                        _BTN_SPD_L2,
                        _BTN_SPD_L3,
                        _BTN_SPD_L4,
                        _BTN_SPD_L5
                        ]
                Show_Background(nSpeed + 1)
        else :
                btnList = [
                        _BTN_ROCK
                        ]
#nLeft_Prev = -1
#nRight_Prev = -1
def Clear_Point(nX, nY):
        Clear(_SHOW_SHAPE, nX, nY)
def ShowGage(nValue):
        if nValue != CVar.nPoint_L:
                if (CVar.nPoint_L > 0):
                        nGap = nSpdBtnH * 2 # (_ BTN_SPD_L1 [3] - _BTN_SPD_L5 [1]) / (5 - 1)
                        nX = _BTN_SPD_X#(_BTN_SPD_L5[0] + _BTN_SPD_L5[2]) / 2
                        #nY = (_BTN_SPD_L5 [1] + (nGap * (CVar.nPoint_L-1)))
                        nY = (_BTN_SPD_L1 [1] - (nGap * (CVar.nPoint_L-1))) + nSpdBtnH
                        Clear_Point(nX, nY)

                CVar.nPoint_L = nValue

                nGap = nSpdBtnH * 2 # (_ BTN_SPD_L1 [3] - _BTN_SPD_L5 [1]) / (5 - 1)
                nX = _BTN_SPD_X#(_BTN_SPD_L5[0] + _BTN_SPD_L5[2]) / 2
                #nY = (_BTN_SPD_L5 [1] + (nGap * (nValue-1)))
                nY = (_BTN_SPD_L1[1] - (nGap * (nValue-1))) + nSpdBtnH
                Show_Point(nX, nY, _COLOR_BLACK)

def TorqAll(IsOn, IsSound = None):
        TorqOnOff (-1, IsOn, IsSound)
def TorqOnOff(nNum, IsOn, IsSound = None):
        if (IsOn == True) :
                if IsSound == True :
                        buzzer.melody(14)
                if (nNum >= 0) :
                        DXL(nNum).torque_on()
                else :
                        dxlbus.torque_on()
                        CVar.IsTorqOn = True
        else :
                CVar.IsTorqOn = False
                if IsSound == True :
                        buzzer.melody(15)
                #Motion ends immediately
                #Motion_Play(-3)
                #Waiting for motion to end
                #waitMotionStop()
                if (nNum >= 0) :
                        540 (nnum) .torque_off ()
                else :
```

```
                dxlbus.torque_off()

def GetResolution():
    for i in range(0, 100):
        screen = smart.read32(10460)
        CVar.nScreenWidth = screen & 0x0000FFFF
        CVar.nScreenHeight = (screen & 0xFFFF0000) >> 16
        if CVar.nScreenWidth > 0 and CVar.nScreenWidth < 65535 and
CVar.nScreenHeight > 0 and CVar.nScreenHeight < 65535:
            break

def GetTouch_Down():
    # 1 2 3 4 5
    # 6 7 8 9 10
    # 11 12 13 14 15
    # 16 17 18 19 20
    # 21 22 23 24 25
    if (smart.is_connected() == True):
        XY_X0 = 0
        XY_Y0 = 0
        Pos_X0 = 0
        Pos_Y0 = 0
        Pos0 = 0
        XY_X1 = 0
        XY_Y1 = 0
        Pos_X1 = 0
        Pos_Y1 = 0
        Pos1 = 0

        # Touch - First
        Tmp = smart.read64(10470) # Touch input coordinate
        nTouch0 = Tmp[0] & 0xffffffff
        nTouch1 = Tmp[1] & 0xffffffff
        IsChanged = False
        if (nTouch0 > 0) :
            XY_X0 = nTouch0 & 0x0000FFFF
            XY_Y0 = (nTouch0 >> 16) & 0x0000FFFF
            Pos_X0 = (int)((XY_X0 / CVar.nScreenWidth) * 5 + 1)
            Pos_Y0 = (int)((XY_Y0 / CVar.nScreenHeight) * 5 + 1)
            Pos0 = Pos_X0 + (Pos_Y0 - 1) * 5
            XY_X0 = (int)(XY_X0 * _RATIO / CVar.nScreenWidth)
            XY_Y0 = (int)(XY_Y0 * _RATIO / CVar.nScreenHeight)
            if (nTouch1 > 0) :
                XY_X1 = nTouch1 & 0x0000FFFF
                XY_Y1 = (nTouch1 >> 16) & 0x0000FFFF
                Pos_X1 = (int)((XY_X1 / CVar.nScreenWidth) * 5 + 1)
                Pos_Y1 = (int)((XY_Y1 / CVar.nScreenHeight) * 5 + 1)
                Pos1 = Pos_X1 + (Pos_Y1 - 1) * 5
                XY_X1 = (int)(XY_X1 * _RATIO / CVar.nScreenWidth)
                XY_Y1 = (int)(XY_Y1 * _RATIO / CVar.nScreenHeight)

            CVar.nTouch_Pos0 = Pos0
            CVar.nTouch_Pos_X0 = Pos_X0
            CVar.nTouch_Pos_Y0 = Pos_Y0
            CVar.nTouch_XY_X0 = XY_X0
            CVar.nTouch_XY_Y0 = XY_Y0

            CVar.nTouch_Pos1 = Pos1
            CVar.nTouch_Pos_X1 = Pos_X1
            CVar.nTouch_Pos_Y1 = Pos_Y1
```

```
639                         CVar.nTouch_XY_X1 = XY_X1
```



```
640                         CVar.nTouch_XY_Y1 = XY_Y1
641                 else :
642                         CVar.nTouch_Pos0 = 0
643                         CVar.nTouch_Pos_X0 = 0
644                         CVar.nTouch_Pos_Y0 = 0
645                         CVar.nTouch_XY_X0 = 0
646                         CVar.nTouch_XY_Y0 = 0
647
648                         CVar.nTouch_Pos1 = 0
649                         CVar.nTouch_Pos_X1 = 0
650                         CVar.nTouch_Pos_Y1 = 0
651                         CVar.nTouch_XY_X1 = 0
652                         CVar.nTouch_XY_Y1 = 0
653         else :
654                 IsPhone = False
655
656 # If you enter a coordinate value between 1 and 1000, it is converted to a coordinate value suitable for a smartphone.
657 def Set(nX, nY):
658         nResX = nX * CVar.nScreenWidth / _RATIO
659         nResY = nY * CVar.nScreenHeight / _RATIO
660         return nResX, nResY
661
662 def IsButton(nX, nY, Btn):
663         if (Btn[0] < 0):
664                 right = (Btn[0] * -200)
665                 left = right - 200
666                 bottom = (Btn [1] * -200)
667                 top = bottom - 200
668                 if ((nY >= top) and (nY <= bottom)):
669                         if ((nX >= left) and (nX <= right)):
670                                 return True
671         else:
672                 if ((nY >= Btn[1]) and (nY <= Btn[3])):
673                         if ((nX >= Btn[0]) and (nX <= Btn[2])):
674                                 return True
675         return False
676
677 # Update all motor positions
678 def PositionUpdate():
679         etc.write8(65,3)
680         while(True) :
681                 if etc.read8(65) == 0 :
682                         break
683
684 #Print number simple test
685 def Show_Num(nX, nY, nValue, nColor = _COLOR_NONE, nSize = None):
686         if (nSize == 0):
687                 Clear_Num(nX, nY)
688         elif (nColor == _COLOR_NONE):
689                 Show(_SHOW_NUM, nX, nY, 60, _COLOR_RED, nValue)
690         elif (nSize == None):
691                 Show(_SHOW_NUM, nX, nY, 60, nColor, nValue)
692         else:
693                 Show(_SHOW_NUM, nX, nY, nSize, nColor, nValue)
```

```
694 def Show_Point(nX, nY, nColor, nSize = None):
695         if (nSize == None):
696             Show(_SHOW_SHAPE, nX, nY, 20, nColor, 1)
697         else:
698             Show(_SHOW_SHAPE, nX, nY, nSize, nColor, 1)
699 def Show_Text(nX, nY, nValue, nColor = _COLOR_NONE, nSize = None):
```



```
700         if (nColor == _COLOR_NONE):
701             Show(_SHOW_TEXT, nX, nY, 60, _COLOR_RED, nValue)
702         elif (nSize == None):
703             Show(_SHOW_TEXT, nX, nY, 60, nColor, nValue)
704         else:
705             Show(_SHOW_TEXT, nX, nY, nSize, nColor, nValue)
706 def Show_Image(nX, nY, nValue, nSize = None):
707         if (nSize == None):
708             Show(_SHOW_IMAGE, nX, nY, 1, 0, nValue)
709         else:
710             Show(_SHOW_SHAPE, nX, nY, nSize, 0, nValue)
711 def Show_Background(nValue):
712         if (nValue != CVar.nBack_Background):
713             smart.display.back_image(nValue)
714             CVar.nBack_Background = nValue
715 #nShowType: 0-picture, 1-letter, 2-shape, 3-number
716 # nValue ([Image-Index], [Shape-1: Circle, 2: Square, 3: Triangle], [Text-Index], [Num-
Value])
717 # nColor (0: Unknown, 1: White, 2: Black, 3: Red, 4: Green, 5: Blue 6: Yellow, 7: Light Gray, 8: Gray,
9: dark gray)
718 def Show(nShowType, nX, nY, nSize, nColor, nValue):
719         if ((nShowType >= 0) and (nShowType < 4)):
720             nX, nY = Set (nX, nY)
721             smart.write32(10480, int(nX) | (int(nY) << 16))
722             nTmp = nValue * 256 + nSize * 65536 + nColor * 16777216
723             if (nShowType == _SHOW_IMAGE): # Do not use the color value of 0xff000000 digits
724                 smart.display.front_image(nTmp & 16777215) # & 0xffffff(=16777215)
725             elif (nShowType == _SHOW_SHAPE) :
726                 smart.display.shape(nTmp)
727             elif (nShowType == _SHOW_TEXT) :
728                 smart.display.text(nTmp)
729             elif (nShowType == _SHOW_NUM) :
730                 smart.display.number(nTmp)
731
732 def Clear(nShowType, nX, nY):
733         if ((nShowType >= 0) and (nShowType < 4)):
734             nX, nY = Set (nX, nY)
735             smart.write32(10480, int(nX) | (int(nY) << 16))
736             if (nShowType == _SHOW_IMAGE) :
737                 smart.display.front_image(0)
738             elif (nShowType == _SHOW_SHAPE) :
739                 smart.display.shape(0)
740             elif (nShowType == _SHOW_TEXT) :
741                 smart.display.text(0)
742             elif (nShowType == _SHOW_NUM) :
743                 smart.display.number(0)
744
745 def Clear_All():
746         smart.write32(10480, 0)
```

```
747             smart.display.front_image(0)
748             smart.display.shape(0)
749             smart.display.text(0)
750             smart.display.number(0)
751 def Clear_Image():
752             smart.write32(10480, 0)
753             smart.display.front_image(0)
754 def Clear_Shape():
755             smart.write32(10480, 0)
756             smart.display.shape(0)
757 def Clear_Text():
```



```
758             smart.write32(10480, 0)
759             smart.display.text(0)
760 def Clear_Num(nX = None, nY = None):
761             if (nX == None) or (nY == None) :
762                     smart.write32(10480, 0)
763                     smart.display.number(0)
764             else :
765                     Show(_SHOW_NUM, nX, nY, 0, 0, 0)
766
767 # Function that converts angle to data used in motor (float -> int)
768 def CalcAngle2Raw (fAngle):
769             if fAngle == None:
770                     fAngle = 0.0
771             return (int) (round (fAngle * 4096.0 / 360.0 + 2048.0))
772 # Function that converts data used in motor into angle value (int -> float)
773 def Get_Load(nID):
774             return DXL(nID).read16(126)
775             #return DXL(nID).present_current()
776 def Get_Position(nID):
777             return DXL(nID).present_position() - aOffset[nID]
778 def CalcRaw2Angle (nRaw):
779             if nRaw == None :
780                     nRaw = 0
781             return (float) (360.0 * ((nRaw - 2048.0) / 4096.0))
782 def Rad2Deg(rad):
783             return rad * 180.0 / math.pi
784 def Deg2Rad(Angle):
785             return Angle * math.pi / 180.0
786
787 def Atan (x, y):
788             #print(math.atan(5/3), math.atan2(5))
789             return Rad2Deg(math.atan2(y, x))
790 def Acos(val):
791             tmp=math.acos(val)
792             return Rad2Deg(tmp)
793             #return Rad2Deg(math.acos(val))
794 def Sin (fAngle):
795             return math.sin (Deg2Rad (fAngle))
796 def Cos (fAngle):
797             return math.cos (Deg2Rad (fAngle))
798 def Get_Angle(nID):
799             return CalcRaw2Angle(DXL(nID).present_position())
800 def CalcCosAngle(a, b, another):
801             return Acos((a*a + b*b - another*another) / (2 * a * b))
```

```
802 def GetXY():
803
804             16: station
805             15: station
806             14: station
807             13: tablet
808             '''
809             MotBase = Get_Angle(10)
810             MotElbow = Get_Angle(16)
811             return CalcXY(MotBase, MotElbow)
812 def CalcXY(MotBase, MotElbow):
813             t10 = MotBase#Get_Angle(10)
814             t16 = MotElbow#-Get_Angle(16)
815
816             P = Cos(45)
817             #P = Sin(45)
```



```
818             C10 = Cos (t10)
819             S10 = Sin (t10)
820             C14 = Cos (t16)
821             S14 = Sin (t16)
822             r0 = C10*P+S10*P
823             r1 = C10*P-S10*P
824             r2 = r0*S14 - r1*C14
825             r3 = r1*S14 + r0*C14
826             x =-r2 * 55 + r3 * 40 + r0*45 + C10*55
827             y = r3 * 55 + r2 * 40 - r1 * 45 + S10 * 55
828             return x,-y
829
830 def SetXY(x, y, z, nTime=100, IsDelay=True):
831             try:
832                     # Move the reference coordinate
833                     X_Trans = 0
834                     Y_Trans = 0
835                     Z_Trans = 75
836                     # Reference angle rotation
837                     X_Rot = 90
838                     Y_Rot = 0
839                     Z_Rot = 0
840
841                     Angle_Pan = 0
842
843                     bar_Wing = 0#70
844
845                     # Tightening
846                     bar_Link0_h = 45*Cos(45)
847                     # Short axis
848                     bar_Link0_d = 55+45*Cos(45)
849
850                     # Tightening
851                     bar_Link1_h = 55
852                     # Short axis
853                     bar_Link1_d = 40
854
855                     # Rot X
856                     fX0 = x-X_Trans
```

```
fY0 = y-Y_Trans
fZ0 = z-Z_Trans
fAngle = X_Rot
fX1 = fX0
fY1 = fY0 * Cos (fAngle) - fZ0 * Sin (fAngle)
fZ1 = fY0 * Sin (fAngle) + fZ0 * Cos (fAngle)

# Rot Z
fX0 = fX1
fY0 = fY1
fZ0 = fZ1
fAngle = Z_Rot
fX1 = fX0 * Cos (fAngle) - fY0 * Sin (fAngle)
fZ1 = fX0 * Sin (fAngle) + fY0 * Cos (fAngle)
fZ1 = fZ0

# Rot Y
fX0 = fX1
fY0 = fY1
fZ0 = fZ1
fAngle = Y_Rot + Angle_Pan
```



```
fX = fX0 * Cos (fAngle) + fZ0 * Sin (fAngle)
fY = fY0
fZ = -fX0 * Sin (fAngle) + fZ0 * Cos (fAngle)

# Finding the angle of the wing motor
val = -90-Atan (fX, fY)
a=-180
clip0 = 0
if (val < a):
        clip0 = 1

tW=-(clip0*360+val)
#//Calculate the length of the bar reduced by rotation
len_w = bar_Wing * Sin (tW)
len_h = bar_Wing * Cos(tW)

#// Make the coordinate system 0 point around the hip joint.
fX0 = fX+len_w
fY0 = fY + len_h

hh0=bar_Link0_h*bar_Link0_h + bar_Link0_d*bar_Link0_d
h0=math.sqrt(hh0)
angle_0_tmp=Atan(bar_Link0_h,bar_Link0_d)
val = bar_Link0_d
a=0
clip0 = 0
if (val < a):
        clip0 = 1
angle_0=angle_0_tmp-clip0*360
hh1=bar_Link1_h*bar_Link1_h + bar_Link1_d*bar_Link1_d
h1=math.sqrt(hh1)
angle_1_tmp=Atan(bar_Link1_h,bar_Link1_d)

val=bar_Link1_d
```

```
                a=0
                clip1 = 0
                if (val < a):
                        clip1 = 1
                angle_1=angle_1_tmp-clip1*360
                #//// Knee
                dd=fX0*fX0 + fY0*fY0 + fZ*fZ
                d=math.sqrt(dd)

                #// ankle
                bb = hh0
                b = h0
                cc=hh1
                c = h1
                aa=dd
                a=d
                value=(bb+cc-aa) / (2*b*c)
                angle_knee=Acos(value)
                val=(180-angle_knee)+angle_0+angle_1
                tKnee = val + 360
                angle_up=Acos((aa+bb-cc) / (2*a*b))
                angle_up1=Atan(-fY0, fZ)

                tUp=angle_up+angle_up1 + angle_0
                #// If tUp exceeds 180 degrees, subtract it...
                #// 1 for less than, 0 for equal or greater
```



```
                val=tUp
                a=180
                clip0 = 0
                if (val < a):
                        clip0 = 1
                tUp=val+(clip1-1)*360

                v0=tW
                v1 = - (tUp + 270) # - (tUp + 270) # 90-tUp
                v2=-(tKnee-360-180+45)#-360-180+45
                #print("xyz={0},{1},{2} : M0,M1,M2={3},{4},{5}".format(x,y,z,v0, v1, v2))
                #return Move_Manipulator(v1, v2, nTime, -20)
                Move_Manipulator(v1, v2, nTime, -nTime*2/3, IsDelay)
                return True
        except ValueError:
                #print("[Warnning]SetXY - {0}".format(ValueError))
                return False
def Test1(nBackgroundImage = 1, nImage = 0, x=500,y=500):
        # Print background image
        if (CVar.nTouch_Pos0 > 0):
                # At the moment of touch, all basic shapes are erased.
                Clear_All()

        # Basic figures and drawings to hear ...
        Show_Background(nBackgroundImage)
        #Test Image : Show_Image(x, y, image index)
        if (x>0) and (y>0) and (nImage>0):
                Show_Image (x, y, nImage)

```

```
967         if (CVar.nTouch_Pos0 > 0):
968             Clear_Num()
969             Clear_Shape()
970             # X
971             nPos = 50
972             nGap = 20
973 
974             nX = CVar.nTouch_XY_X0
975             nY = CVar.nTouch_XY_Y0
976             if nX < 100 :
977                 nX = 100
978             elif nX > 900 :
979                 nX = 900
980             if nY < 100 :
981                 nY = 100
982             elif nY > 900 :
983                 nY = 900
984 
985             Show_Num (nX - nPos - nGap, nY-40, (int) (CVar.nTouch_XY_X0 / 100% 10))
986             Show_Num(nX - nPos,                 nY-40, (int)(CVar.nTouch_XY_X0 / 10 % 10))
987             Show_Num (nX - nPos + nGap, nY-40, (int) (CVar.nTouch_XY_X0% 10))
988             # Y
989             nPos = 50
990             Show_Num (nX + nPos - nGap, nY-40, (int) (CVar.nTouch_XY_Y0 / 100% 10))
991             Show_Num(nX + nPos,                 nY-40, (int)(CVar.nTouch_XY_Y0 / 10 % 10))
992             Show_Num (nX + nPos + nGap, nY-40, (int) (CVar.nTouch_XY_Y0% 10))
993 
994             Show_Num(nX-30, nY+40, CVar.nTouch_Pos_X0, _COLOR_GREEN)
995             Show_Num(nX+30, nY+40, CVar.nTouch_Pos_Y0, _COLOR_GREEN)
996 
997             Show_Point(CVar.nTouch_XY_X0, CVar.nTouch_XY_Y0, _COLOR_BLUE)
```



```
998 
999         if (CVar.nTouch_Pos1 > 0):
100 
    0             # X
100 
    1             nPos = 50
100 
    2             nGap = 20
100 
    3 
100 
    4             nX = CVar.nTouch_XY_X1
100 
    5             nY = CVar.nTouch_XY_Y1
100 
    6             if nX < 100 :
100 
    7                 nX = 100
100 
    8             elif nX > 900 :
100 
    9                 nX = 900
101 
    0             if nY < 100 :
```

```
                nY = 100
        elif nY > 900 :
                nY = 900

        Show_Num (nX - nPos - nGap, nY-40, (int) (CVar.nTouch_XY_X1 / 100% 10))
        Show_Num(nX - nPos,                     nY-40, (int)(CVar.nTouch_XY_X1 / 10 % 10))
        Show_Num (nX - nPos + nGap, nY-40, (int) (CVar.nTouch_XY_X1% 10))
        # Y
        nPos = 50
        Show_Num (nX + nPos - nGap, nY-40, (int) (CVar.nTouch_XY_Y1 / 100% 10))
        Show_Num(nX + nPos,                     nY-40, (int)(CVar.nTouch_XY_Y1 / 10 % 10))
        Show_Num (nX + nPos + nGap, nY-40, (int) (CVar.nTouch_XY_Y1% 10))

        Show_Num(nX-30, nY+40, CVar.nTouch_Pos_X1, _COLOR_GREEN)
        Show_Num(nX+30, nY+40, CVar.nTouch_Pos_Y1, _COLOR_GREEN)

        Show_Point(CVar.nTouch_XY_X1, CVar.nTouch_XY_Y1, _COLOR_RED)
'''
```



```
def Test2():
        # Check button press
        nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
        if (Event_Dn0) :
                print("Down0:{0}:{1}".format(nNum0, nNum1))
        if (Event_Dn1) :
                print("Down1:{0}:{1}".format(nNum0, nNum1))
        if (Event_Up0) :
                print("Up0:{0}:{1}".format(nNum0, nNum1))
```

```
103
    8             if (Event_Up1) :
103
    9                 print("Up1:{0}:{1}".format(nNum0, nNum1))
104
    0             if (CVar.nTouch_Pos0 > 0):
104
    1                 # Delete numbers.
104
    2                 Clear_Num()
104
    3                 if (nNum0 >= 0) :
104
    4                     Show_Num(450, 500, nNum0, _COLOR_RED)
104
    5                 if (nNum1 >= 0) :
104
    6                     Show_Num(550, 500, nNum1, _COLOR_BLUE)
104
    7 '''
104
    8 #Set motor operation speed (Based on Position): 0 initialization
104
    9 def Setup_Speed(nID, nValue) :
105
    0             DXL(nID).write32(112, nValue) # 112 : profile velocity
105
    1 def Move_Ready(nWith_Arm = 1):
105
    2             global m_afArm
105
    3             global m_afManipulator
105
    4
105
    5             IDs = [1,2,3,4,5,6,7,8,9]#,10, 13,14,15,16]
105
    6             Move(IDs, _MNT_READY)
105
    7             Move_Yaw(0, 1.0, False)
```



```
105
    8             if nWith_Arm == 1:
105
    9                 Move_Ready_Arm()
106
    0
106
    1                 Move_Gripper(-50, 10)
106
    2
106
    3 def Move_Ready_Arm():
106
    4             global m_fX
```

```
1065        global m_fY
1066        m_fX = _INIT_FX
1067        m_fY = _INIT_FY
1068        #Move_Gripper(0)
1069        SetXY(0, m_fY, m_fX, 1000)
1070
1071 def Delay_Body_Step(motions, nStepIndex, fPercent = 1.0):
1072        motion_step = motions[nStepIndex]
1073        tmr = CTimer ()
1074        #DXL(9).write32(112, 40)
1075        tmr.Set()
1076        while(tmr.Get() < int(motion_step[0] * fPercent)):
1077            if btnList != None:
1078                # Check button press-consumption of touch
1079                nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
1080            #pass
1081 def Move_Body_Step(motions, nStepIndex, IsDelay = True, fPercent = 1.0):
1082        #if (nSpeed == 0):
1083        # = 1.0 fPercent
1084        #else :
1085        # = 0.8 fPercent
1086        motion_step = motions[nStepIndex]
1087
```



```
1088        IDs = [1,2,3,4,5,6,7,8]
1089        if (len(motion_step) == 9 + 1):
1090            IDs = [1,2,3,4,5,6,7,8,9]
1091        if (nStepIndex >= 0) and (nStepIndex < len(motions)):
1092
```

```
                Writes(IDs, 116, motion_step, 4, fPercent)
109
    #Move(IDs, motion_step, fPercent)
109
    if (IsDelay):
109
        Wait(int(motion_step[0] * fPercent))
109
def Moves_Body(motions, fPercent = 1.0, fPercent_Delay = 0.0):
109
    #if (nSpeed == 0):
109
    # = 1.0 fPercent
109
    #else :
110
    # = 0.8 fPercent
110
    IDs = [1,2,3,4,5,6,7,8,9]#,10]
110
    for values in motions:
110
        Move(IDs, values, fPercent, fPercent_Delay)
110
#fElbow(-),
110
def Move_Manipulator(fBase, fElbow, fTime, fDelay, IsDelay = True):
110
    bRet = True
110
    # data clipping ...
110
    if (fElbow < -90):
110
        bRet = False
111
        fElbow = -90
111
    fTilt = fBase-fElbow+45 #Tilt
111
    # data clipping ...
111
    if (fTilt > 105):
111
        bRet = False
111
        fTilt = 105
111
    IDs = [10, 16, 15]
111
    Writes(IDs, 116, [fTime, fBase, fElbow, fTilt], 4, 1.0)
```



```
111
8
111
    m_afArm[2] = fBase #Base
```

```
112
    m_afArm[3] = fElbow #Elbow
112
    #                       15(Tilt), 13 & 14
112
    m_afHand[1] = fTilt #Tilt
112
    m_afManipulator[1] = fBase #Base
112
    m_afManipulator[2] = fElbow #Elbow
112
    m_afManipulator[3] = fTilt #Tilt
112
112
    if (IsDelay):
112
        DelayForRun(int(fTime + fDelay))
112
    #print("Manipulator:{0},{1},{2}".format(fBase, fElbow, fTilt))
113
    return bRet
113
def Move_Yaw(fYaw, fPercent = 1.0, IsDelay = True):
113
    global m_afArm
113
    if (fYaw >= 30):
113
        fYaw = 30
113
    if (fYaw <= -30):
113
        fYaw = -30
113
    m_afArm[1] = fYaw #fYaw
113
    fSpeed = 100
113
    Writes([9], 116, [fSpeed, fYaw], 4, fPercent)
114
    if (IsDelay):
114
        DelayForRun(int(m_afArm[0] * fPercent * 0.5))
114
    #Move([9], [fSpeed, fYaw], fPercent)
114
def IMove_Yaw(fYaw, fPercent = 1.0, IsDelay = True):
114
    Move_Yaw(m_afArm[1] + fYaw, fPercent, IsDelay)
114
def Move_Tilt_Gripper(fTilt, nTime = 0):
114
    global m_afHand
114
    m_afHand[1] = fTilt #Tilt
```

```
114
    8           Writes([15], 116, [nTime, fTilt], 4, 1.0)
114
    9           if nTime > 0:
115
    0               DelayForRun(int(nTime))
115
    1 def Move_Gripper(fGrip, fPercent = 1.0, IsDelay = True):
115
    2           global m_afHand
115
    3           if (fGrip >= 25): # + grab direction
115
    4               fGrip = 25
115
    5           if (fGrip <= -30): #-opening direction
115
    6               fGrip = -30
115
    7           #m_afHand[1] = fTilt #Tilt
115
    8           m_afHand[2] = fGrip
115
    9           m_afHand[3] = -fGrip
116
    0           #print(fGrip)
116
    1           Writes(_IDS_HAND, 116, m_afHand, 4, fPercent)
116
    2           if (IsDelay):
116
    3               DelayForRun(int(m_afHand[0] * fPercent * 0.5))
116
    4           #Move(_IDS_HAND, m_afHand, fPercent)
116
    5 def IMove_Gripper(fGrip, fPercent = 1.0, IsDelay = True):
116
    6           Move_Gripper(m_afHand[2] + fGrip, fPercent, IsDelay)
116
    7 def IMove_Tilt_Gripper(fTilt, nTime = 0):
116
    8           Move_Tilt_Gripper(m_afHand[1] + fTilt, nTime)
116
    9 def Move(IDs, values, fPercent = 1.0, fPercent_Delay = 0.0) :
117
    0           if (fPercent_Delay == 0):
117
    1               fPercent_Delay = fPercent
117
    2           Writes(IDs, 116, values, 4, fPercent)
117
    3           Wait(int(values[0] * fPercent_Delay))
117
    4 def Wait(nMilliseconds) :
117
    5           tmr = CTimer ()
117
    6           tmr.Set()
117
    7           while(tmr.Get() < nMilliseconds) :
```

```
117
    8                   if btnList != None:
117
    9                       #Check the touch input of the smartphone
118
    0                       GetTouch_Down()
118
    1                       # Check button press-consumption of touch
118
    2                       nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
118
    3                   delay(10)
118
    4           tmr = None
118
    5
118
    6 def Writes(IDs, nAddress, values, size = 4, fPercent = 1.0):
118
    7           #profile speed adjustment
118
    8           Setup_Speed(254, int(values[0] * fPercent))
118
    9           #syncwrite
119
    0           etc.write8(1200,0)
119
    1           etc.write16(1202,nAddress)
119
    2           etc.write8(1204,size)
119
    3           nMot_First = 5
119
    4           nMot_Cnt = 1
119
    5           nPos = 1
119
    6           for i in IDs:
119
    7               nID = i
119
    8               etc.write8 (1205, nID)
119
    9               etc.write32(1206,CalcAngle2Raw(values[nPos]))
120
    0               etc.write8(1200,1)
120
    1               nPos = nPos + 1
120
    2           etc.write8(1200,2)
120
    3 '''
120
    4 def Writes(IDs, nAddress, values, size = 4, fPercent = 1.0):
```

```
1205            #profile speed adjustment
1206            #if CVar.IsWalking:
```



```
1207            # Setup_Speed(254, int(values[0] * fPercent))
1208            #else :
1209            # Setup_Speed(254, int(values[0] / CVar.nPoint_L * fPercent))
1210            Setup_Speed(254, int(values[0] / CVar.nPoint_L * fPercent))
1211            #syncwrite
1212            etc.write8(1200,0)
1213            etc.write16(1202,nAddress)
1214            etc.write8(1204,size)
1215            nMot_First = 5
1216            nMot_Cnt = 1
1217            nPos = 1
1218            for i in IDs:
1219                    nID = i
1220                    etc.write8 (1205, nID)
1221                    etc.write32(1206,CalcAngle2Raw(values[nPos]))
1222                    etc.write8(1200,1)
1223                    nPos = nPos + 1
1224            etc.write8(1200,2)
1225            #profile speed restore
1226            #Setup_Speed(254, 0)
1227    '''
1228    def WaitButtonUp():
1229            while(True) :
1230                    #Check the touch input of the smartphone
1231                    GetTouch_Down()
```

```
123
    2                   # Check button press
123
    3                   nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
123
    4                   if ((nNum0 < 0) and (nNum1 < 0)):
123
    5                           break
```

```
123
    6
123
    7 def GetButton(btns):
123
    8           nNum0 = -1
123
    9           nNum1 = -1
124
    0           nDown0 = 0
124
    1           nDown1 = 0
124
    2           nUp0 = 0
124
    3           nUp1 = 0
124
    4           Btn0 = None
124
    5           Btn1 = None
124
    6           nnum = 0
124
    7           nCnt = 0
124
    8           if (CVar.nTouch_Pos0 > 0) :
124
    9                   nCnt = nCnt + 1
125
    0           if (CVar.nTouch_Pos1 > 0) :
125
    1                   nCnt = nCnt + 1
125
    2
125
    3           nPass = 0
125
    4           for btn in btns:
125
    5                   if (CVar.nTouch_Pos0 > 0):
125
    6                           if (IsButton(CVar.nTouch_XY_X0, CVar.nTouch_XY_Y0, btn) == True) :
125
    7                                   nNum0 = btn [_BTN_INDEX]
125
```

```
                    Btn0 = btn
                    nPass = nPass | 0x01
            if (CVar.nTouch_Pos1 > 0):
                if (IsButton(CVar.nTouch_XY_X1, CVar.nTouch_XY_Y1, btn) == True) :
                    nNum1 = btn [_BTN_INDEX]
                    Btn1 = btn
                    nPass = nPass | 0x10
            if (nCnt == 1) :
```



```
                if (nPass > 0):
                    break
            else:
                if (nPass == 0x11):
                    break
            nnum + 1 = nnum

        if ((nNum1 == nNum0) and (nNum0 >= 0)):
            nNum1 = -1

        # switching
        #IsSwitching = False
        if (((nNum0 >= 0) and (nNum0 == CVar.nButton_1)) or ((nNum1 >= 0) and (nNum1 ==
CVar.nButton_0))) :
            nNum2 = nNum1
            nNum1 = nNum0
            nNum0 = nNum2
            #IsSwitching = True
        #Button Down Event
        if (nNum0 >= 0):
```

```
                if (CVar.nButton_0 != nNum0):
                        nDown0 = 1
                CVar.btn0 = Btn0
        else :
                if (CVar.nButton_0 >= 0):
                        nUp0 = 1
        if (nNum1 >= 0):
                if (CVar.nButton_1 != nNum1):
                        nDown1 = 1
                CVar.btn1 = Btn1
        else :
```



```
                if (CVar.nButton_1 >= 0):
                        nUp1 = 1

        CVar.nButton_0 = nNum0
        CVar.nButton_1 = nNum1
        if nUp0 == 1:
                Btn0 = CVar.btn0
        if nUp1 == 1:
                Btn1 = CVar.btn1

        return nNum0, nNum1, nDown0, nDown1, nUp0, nUp1, Btn0, Btn1
################################################################################################
################
#console(USB)
#console (BLE)
console(UART)
```

```
# controller direction : 0-vertical(Humanoid), 1-Horizontal
eeprom.imu_type(1)
# Turn off the torque and modify the actuator settings and controller settings according to the robot.
TorqAll (False)
# profile -> velocity-based
DXL(254).write8(10, 4) # 0 -> velocity-based profile, 4 -> time-based profile
# Secondary ID(255: No Use, 0 ~ 252: ID)
DXL(254).write8(12, 255) # 255 -> No Use(Clear)
# Operation Mode(1:velocity[wheel], 3:position)
DXL(254).mode(3) # position
TorqAll(True)

# Turn off torque and change all actuator settings to Time-based profile.
```



```
TorqAll (False)
DXL(254).write8(10, 4) # 0 -> velocity-based profile, 4 -> time-based profile

TorqAll(True)
# In actual use, nTest = 0 should be used.
nTest = 0 # 0-Normal, 1-Coordinate output, 2-Click test (Click the designated button position (nTest_X, nTest_Y)
Test)
nTest_BackgroundImage = 1

Move_Ready()
nDemoPos_X_0 = 55
nDemoPos_X_1 = 55 # 25
nDemoPos_X = nDemoPos_X_0
IsGrapped = False
while(True) :
```

```
1339      if (IsPhone == False) :
1340
1341          if (smart.is_connected() == True):
1342
1343              #Check if it is connected to the smartphone.
1344
1345              smart.wait_connected()
1346
1347              #Set the smartphone screen horizontally. (0: Auto, 1: Vertical, 2: Horizontal)
1348
1349              smart.display.screen_orientation(2)
1350
1351              # Waiting for the screen to switch before getting the screen's width and height.
1352
1353              delay(500)
1354
1355
1356
1357              # Get the resolution of the screen and put it in CVar.nScreenWidth, CVar.nScreenHeight.
1358
1359              GetResolution()
1360
1361
1362              #Print background image
1363
1364              ShowPage(_PAGE_MAIN)
1365
1366
```



```
1367
1368              #Show speed gauge
1369
1370              ShowGage(2)
1371
1372
1373              # Writes the smartphone connection to the variable
1374
1375              IsPhone = True
1376      else :
1377          #Check the touch input of the smartphone
1378
1379          GetTouch_Down()
1380
1381          # Test 1 => Coordinate output
1382
1383          if (nTest == 1) :
1384
1385              Test1(nTest_BackgroundImage)
1386
1387          #elif (nTest == 2): # Red dot when clicking with touch 0, blue dot when clicking with touch 1 (second touch)
```

```
136
    6                   # Test2 ()
136
    7                   else: #Run
136
    8                           # Check button press
136
    9                           nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
137
    0
137
    1                           if (Btn0 == _BTN_DEMO) and (Event_Dn0):
137
    2                                   if CVar.nDemoMode != 0:
137
    3                                           CVar.nDemoMode = 0
137
    4                                           Move_Ready()
137
    5                                   else:
137
    6                                           Move_Yaw(0, 10.0, True)
137
    7
137
    8                                           CVar.nDemoMode = 1
137
    9                                           CVar.nDemoIndex = 0
138
    0                                           IsGrapped = False
138
    1                                           lstDemo = _MNT_DEMO
138
    2                                           nStartPos_X = _INIT_FX + 20 #_INIT_FX
```



```
138
    3                                           nStartPos_Y = _INIT_FY -20
138
    4                                           nLength = 50 #70
138
    5                                           nSpeed_Demo = 50
138
    6                                           SetXY(0, _INIT_FY, nStartPos_X, 1000, False)
138
    7                                           Wait(1000)
138
    8                                           SetXY(0, nStartPos_Y, nStartPos_X, 1000, False)
138
    9                                           Wait(1000)
139
    0                                           for i in range(0, nLength, 2): # increase from 0 to nLength by 2
139
    1                                                   SetXY(0, nStartPos_Y, nStartPos_X + i, nSpeed_Demo, False)
139
```

```
                Wait(int(nSpeed_Demo * 0.8))
        #print(CVar.nDemoMode)
        if CVar.nDemoMode != 0:
            if (CVar.nDemoIndex == 0):
                delay(100)
                if (lstDemo == _MNT_DEMO):
                    lstDemo = _MNT_DEMO2
                elif lstDemo == _MNT_DEMO2:
                    lstDemo = _MNT_DEMO
                    if (nDemoPos_X == nDemoPos_X_0):
                        nDemoPos_X = nDemoPos_X_1
                    else:
                        nDemoPos_X = nDemoPos_X_0
                else:
                    lstDemo = _MNT_DEMO
                    if (nDemoPos_X == nDemoPos_X_0):
                        nDemoPos_X = nDemoPos_X_1
                    else:
                        nDemoPos_X = nDemoPos_X_0

            if (IsGrapped):
```



```
                Move_Body_Step(lstDemo, CVar.nDemoIndex, False)

                nGap_H = 0
                nGap_F = 0
                if (lstDemo[CVar.nDemoIndex][2] == -nDemoH_Default+nDemoH):
                    nGap_H = -5
                    nGap_F = 10
```

```
                                        else:
                                                nGap_H = +40
                                                nGap_F = 0

                                        SetXY (0, _INIT_FY + nGap_H, _INIT_FX + nDemoPos_X + nGap_F,
lstDemo[CVar.nDemoIndex][0], False)

                                        Delay_Body_Step(lstDemo, CVar.nDemoIndex, 0.8)

                                        CVar.nDemoIndex = (CVar.nDemoIndex + 1) % len(lstDemo)
                                        #Moves_Body(_MNT_DEMO)
                                        #continue
                                else:
                                        #Gripper
                                        nLoad1 = abs(Get_Load(13))
                                        nLoad2 = abs(Get_Load(14))
                                        if (nLoad1 < _LOAD_DEMO) and (nLoad2 < _LOAD_DEMO) :
                                                IMove_Gripper(5, 1.0, False)
                                                if (m_afHand[2] >= 25):
                                                        CVar.nDemoMode = 0 # Exit Demo mode
                                                        Move_Ready()
                                        else :
                                                IsGrapped = True
```



```
                        if (Event_Dn1 == 1):
                                smart.etc.vibrate(10)
                        elif (Event_Dn0 == 1):
                                smart.etc.vibrate(10)
```

```
1447
1448                    nBtn_Left = 0
1449
1450                    nClick = 0
1451
1452                    if (Btn0 == _BTN_SPD_L1) or (Btn1 == _BTN_SPD_L1):
1453
1454                            nBtn_Left = 1
1455
1456                    elif (Btn0 == _BTN_SPD_L2) or (Btn1 == _BTN_SPD_L2):
1457
1458                            nBtn_Left = 2
1459
1460                    elif (Btn0 == _BTN_SPD_L3) or (Btn1 == _BTN_SPD_L3):
1461
1462                            nBtn_Left = 3
1463
1464                    elif (Btn0 == _BTN_SPD_L4) or (Btn1 == _BTN_SPD_L4):
1465
1466                            nBtn_Left = 4
1467
1468                    elif (Btn0 == _BTN_SPD_L5) or (Btn1 == _BTN_SPD_L5):
1469
1470                            nBtn_Left = 5
1471
1472
1473                    if (nBtn_Left > 0):
1474
1475                            ShowGage(nBtn_Left)
1476
1477
1478                    IsWalking = False
1479
1480                    nMove = -1
1481
1482                    if (Btn0 == _BTN_TURN_L) or (Btn1 == _BTN_TURN_L):
1483
1484                            IsWalking = True
1485
1486                            nMove = 3
1487
1488                    if (Btn0 == _BTN_TURN_R) or (Btn1 == _BTN_TURN_R):
1489
1490                            IsWalking = True
1491
1492                            nMove = 5
1493
1494                    if (Btn0 == _BTN_MOVE_UL) or (Btn1 == _BTN_MOVE_UL):
```



```
1495
1496                            IsWalking = True
1497
1498                            nMove = 0
```

```
147 5                   if (Btn0 == _BTN_MOVE_U) or (Btn1 == _BTN_MOVE_U):
147 6                       IsWalking = True
147 7                       nMove = 1
147 8                   if (Btn0 == _BTN_MOVE_UR) or (Btn1 == _BTN_MOVE_UR):
147 9                       IsWalking = True
148 0                       nMove = 2
148 1                   if (Btn0 == _BTN_MOVE_DL) or (Btn1 == _BTN_MOVE_DL):
148 2                       IsWalking = True
148 3                       nMove = 6
148 4                   if (Btn0 == _BTN_MOVE_D) or (Btn1 == _BTN_MOVE_D):
148 5                       IsWalking = True
148 6                       nMove = 7
148 7                   if (Btn0 == _BTN_MOVE_DR) or (Btn1 == _BTN_MOVE_DR):
148 8                       IsWalking = True
148 9                       nMove = 8
149 0
149 1                   if (nMove >= 0):
149 2                       if (CVar.IsWalking == False):
149 3                           CVar.IsWalking = True
149 4                           IDs = [9,10,15,16]
149 5                           Move(IDs, [1000, 0, -85, -65, 30])
149 6
149 7                       MotionSet(nMove)
149 8                       Moves_Body(_MOTION, 1.0, 0.6)
149 9                   if CVar.IsWalking == True and IsWalking == False:
150 0                       CVar.IsWalking = False
150 1                       Move_Ready(0)
150 2                       Move_Ready_Arm()
```

```
150
    3
150
    4                   IsChangeSpeed = False
150
    5                   if (Btn0 == _BTN_MOVE_X) or (Btn1 == _BTN_MOVE_X):
150
    6                           if (Btn0 == _BTN_MOVE_X):
150
    7                                   if (Event_Dn0):
150
    8                                           IsChangeSpeed = True
150
    9                           else:
151
    0                                   if (Event_Dn1):
151
    1                                           IsChangeSpeed = True
151
    2                           if (IsChangeSpeed):
151
    3                                   nSpeed = (nSpeed + 1) % 2
151
    4                                   Show_Background(nSpeed + 1)
151
    5                   #if (fLength < 4):
151
    6                   # fLength = 4
151
    7                   #nTime = 1500 / (CVar.nPoint_L)
151
    8                   #nTime = int(400 / CVar.nPoint_L)) # 5 : 100, 1 : 500
151
    9                   #nTime = int(100 + CVar.nPoint_L * 400 / 5) # 5 : 100, 1 : 400
152
    0                   #fLength = (CVar.nPoint_L - 1) * 1.5 + 4#5-(CVar.nPoint_L-1)#3#3
152
    1                   fLength = (CVar.nPoint_L - 1) + 3#5-(CVar.nPoint_L-1)#3#3
152
    2                   nTime = int(50 + CVar.nPoint_L * 400 / 5) # 5 : 100, 1 : 400
152
    3                   if (Btn0 == _BTN_ARM_F) or (Btn1 == _BTN_ARM_F):
152
    4                           fX = m_fX + fLength
152
    5                           if (fX <= 235):
152
    6                                   if SetXY(0, m_fY, fX, nTime) == True:
152
    7                                           m_fX = m_fX + fLength
152
    8                   if (Btn0 == _BTN_ARM_B) or (Btn1 == _BTN_ARM_B):
152
    9                           fX = m_fX - fLength
153
    0                           if (fX >= _INIT_FX):#120):
153
    1                                   if SetXY(0, m_fY, fX, nTime) == True:
153
    2                                           m_fX = m_fX - fLength
```

```
153
    if (Btn0 == _BTN_ARM_ORG) or (Btn1 == _BTN_ARM_ORG):
153
        Move_Ready_Arm()
153
    if (Btn0 == _BTN_ARM_UP) or (Btn1 == _BTN_ARM_UP):
153
        fY = m_fY + fLength
153
        if SetXY(0, fY, m_fX, nTime) == True:
153
            m_fY = m_fY + fLength
153
    if (Btn0 == _BTN_ARM_DN) or (Btn1 == _BTN_ARM_DN):
154
0       fY = m_fY - fLength
154
1       if (fY >= -40):
154
2           if SetXY(0, fY, m_fX, nTime) == True:
154
3               m_fY = m_fY - fLength
154
4   nMulti = 1
154
5   if (Btn0 == _BTN_ARM_LEFT) or (Btn1 == _BTN_ARM_LEFT):
154
6       IMove_Yaw(CVar.nPoint_L * -nMulti)
154
7   if (Btn0 == _BTN_ARM_RIGHT) or (Btn1 == _BTN_ARM_RIGHT):
154
8       IMove_Yaw(CVar.nPoint_L * nMulti)
154
9   if (Btn0 == _BTN_HAND_IN) or (Btn1 == _BTN_HAND_IN):
155
0       nLoad1 = abs(Get_Load(13))
155
1       nLoad2 = abs(Get_Load(14))
155
2       if (nLoad1 < _LOAD) and (nLoad2 < _LOAD) :
155
3           IMove_Gripper(CVar.nPoint_L * nMulti)
155
4       #else:
155
5       # print("I got you")
155
6   if (Btn0 == _BTN_HAND_ORG) or (Btn1 == _BTN_HAND_ORG):
155
7       Move_Gripper(-20, 10)
155
8
155
9   if (Btn0 == _BTN_HAND_OUT) or (Btn1 == _BTN_HAND_OUT):
156
```

```
                IMove_Gripper(CVar.nPoint_L * -nMulti)
            #Torque ON/OFF button-Since torque off is a function to be aware of, it must be activated when only one touch comes in. Make it small.
            if (Btn0 == _BTN_TORQ):# or (Btn1 == _BTN_TORQ):
```



```
                if Event_Dn0 == 1:
                    tmrTorqBtn.Set()
                    if (CVar.IsTorqOn == True):
                        TorqAll(False, True)
                    else :
                        TorqAll(True, True)
                        Move_Ready(0)
                elif Event_Up0 == 1:
                    tmrTorqBtn.Destroy()
                ##If you keep holding down
                else:
                    #Reboot All
                    if (tmrTorqBtn.Get() >= 3000):
                        #Command - Reboot
                        dxlbus.reboot()
                        #Reboot - Beep
                        buzzer.melody(1)
                        #TorqOff variable
                        CVar.IsTorqOn = False
                        # No more timer detection.
                        tmrTorqBtn.Destroy()
```