```
from pycm import *
#import gc
import time
import math
import urandom
class CVar:
	IsTorqOn			= False
	nScreenWidth		= 0
	nScreenHeight		= 0

	IsResetCommand		= False

	NOT					= -1
	nOffset				= 0

	nCamera_mode		= 0
	nCamera_mode_sub	= 0

	#IsSwitch			= False
	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

```

```
44          nDemoMode                   = 0
45
46          nSpeed                      = 0
47          nOrientation                = 2 # 1: vertical, 2: horizontal
48
49  class CTimer:
50          nTimer                      = 0
51          IsTimer                     = False
52
53          #def __init__(self):
54          # self.nTimer = 0
55          # self.IsTimer = 0
56
57          def Set(self):
58                  self.IsTimer = True
59                  self.nTimer = millis()
60
```

Page 2

```
61          def Get(self):
62                  if self.IsTimer :
63                          return millis() - self.nTimer
64                  return 0
65
66          def Destroy(self):
67                  self.IsTimer = False
68
69  _COLOR_NONE                 = 0
70  _COLOR_WHITE                = 1
71  _COLOR_BLACK                = 2
72  _COLOR_RED                  = 3
73  _COLOR_GREEN                = 4
74  _COLOR_BLUE                 = 5
75  _COLOR_YELLOW = 6
76  _COLOR_GRAY_LIGHT = 7
77  _COLOR_GRAY                 = 8
78  _COLOR_GRAY_DARK = 9
79
80  _SHOW_IMAGE                 = 0
81  _SHOW_TEXT                  = 1
82  _SHOW_SHAPE                 = 2
83  _SHOW_NUM                   = 3
84
85  _RATIO                      = 1000
86  _BTN_INDEX                  = 4
87  #1..5
88  #[left,top,right,bottom, 고유 번호(ButtonNumber)] : left, top < 0 then 1..5 position
89  # 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.]
90  _BTN_MENU                   = [20,40,160,150,1]
91  _BTN_MENU_V                 = [30,10,340,80,2]
92
93  #Page 0
```

```
#Menu-Control
_BTN_SUB_OUT                = [1,1,1000,700,60]
_BTN_SUB_X                  = [950,730,990,780,61]
_BTN_SUB_REMOTE = [100-50,850-100,100+50,850+100,62]
_BTN_SUB_GESTURE = [250-50,850-100,250+50,850+100,64]
_BTN_SUB_DEMO = [400-50,850-100,400+50,850+100,64]
_BTN_SUB_MOTOR = [700-50,850-100,700+50,850+100,65]
_BTN_SUB_OFFSET = [850-50,850-100,850+50,850+100,66]

#Page 1
#move button
_BTN_TURN_L                 = [30,320,100,440,6]
_BTN_TURN_R                 = [240,320,310,440,7]
_BTN_MOVE_UL                = [60,460,130,590,12]
_BTN_MOVE_U                 = [140,390,200,530,13]
_BTN_MOVE_UR                = [210,460,280,590,14]
_BTN_MOVE_DL                = [60,740,130,880,17]
_BTN_MOVE_D                 = [140,800,200,940,18]
_BTN_MOVE_DR                = [210,740,280,880,19]

#Torque ON/OFF button
_BTN_TORQ                   = [630,620,730,760,21]

#normal mode action button
_BTN_ACT_1                  = [770,460,860,580,30]
_BTN_ACT_2                  = [880,460,970,580,31]
```

Page 3

```
_BTN_ACT_3                  = [770,630,860,760,32]
_BTN_ACT_4                  = [880,630,970,760,33]
_BTN_ACT_5                  = [770,800,860,930,34]
_BTN_ACT_6                  = [880,800,970,930,35]

#Gesture button
#_BTN_GESTURE                 = [230,330,820,640,36]
#DEMO mode
_BTN_MODE_1                 = [650,30,790,150,40]
_BTN_MODE_2                 = [820,30,960,150,41]

#motion speed button
_BTN_SPD                    = [130,580,210,750,50]

# Motor ID Tag
_MOT_W = 80
_MOT_H = 50
_BTN_ID_1 = [280-_MOT_W, 840-_MOT_H, 280 + _MOT_W, 840 + _MOT_H, 101]
_BTN_ID_2 = [120-_MOT_W, 840-_MOT_H, 120 + _MOT_W, 840 + _MOT_H, 102]
_BTN_ID_3 = [720-_MOT_W, 840-_MOT_H, 720 + _MOT_W, 840 + _MOT_H, 103]
_BTN_ID_4 = [880-_MOT_W, 840-_MOT_H, 880 + _MOT_W, 840 + _MOT_H, 104]
_BTN_ID_5 = [280-_MOT_W, 730-_MOT_H, 280 + _MOT_W, 730 + _MOT_H, 105]
_BTN_ID_6 = [120-_MOT_W, 730-_MOT_H, 120 + _MOT_W, 730 + _MOT_H, 106]
_BTN_ID_7 = [720-_MOT_W, 730-_MOT_H, 720 + _MOT_W, 730 + _MOT_H, 107]
```

```
144 _BTN_ID_8 = [880-_MOT_W, 730-_MOT_H, 880 + _MOT_W, 730 + _MOT_H, 108]
145 _BTN_ID_9 = [370-_MOT_W, 620-_MOT_H, 370 + _MOT_W, 620 + _MOT_H, 109]
146 _BTN_ID_10 = [210-_MOT_W, 620-_MOT_H, 210 + _MOT_W, 620 + _MOT_H, 110]
147 _BTN_ID_11 = [630-_MOT_W, 620-_MOT_H, 630 + _MOT_W, 620 + _MOT_H, 111]
148 _BTN_ID_12 = [790-_MOT_W, 620-_MOT_H, 790 + _MOT_W, 620 + _MOT_H, 112]
149 _BTN_ID_13 = [370-_MOT_W, 950-_MOT_H, 370 + _MOT_W, 950 + _MOT_H, 113]
150 _BTN_ID_14 = [630-_MOT_W, 950-_MOT_H, 630 + _MOT_W, 950 + _MOT_H, 114]
151 _BTN_ID_15 = [500-_MOT_W, 450-_MOT_H, 500 + _MOT_W, 450 + _MOT_H, 115]
152 _BTN_ID_16 = [500-_MOT_W, 320-_MOT_H, 500 + _MOT_W, 320 + _MOT_H, 116]
153 _BTN_ID_17 = [500-_MOT_W, 190-_MOT_H, 500 + _MOT_W, 190 + _MOT_H, 117]
154
155
156 # Reset & Init ( Offset )
157 _BTN_RESET = [900-70,300-250,900+70,300,82]
158 _BTN_INIT = [900-70,300,900+70,300+250,83]
159
160 #Offset Setting Dialog
161 _BTN_DIALOG_OK = [600-100,800-80,600 + 100,800 + 80,84]
162 _BTN_DIALOG_CANCEL = [400-100,800-80,400+100,800+80,85]
163 _BTN_DIALOG_TORQ = [600-80,400-50,600+80,400+50,86]
164 _BTN_DIALOG_PLUS = [640-50,560-50,640 + 50,560 + 50,87]
165 _BTN_DIALOG_MINUS = [360-50,560-50,360 + 50,560 + 50,88]
166
167 _PAGE_MENU                                  = 0
168 _PAGE_MAIN                                  = 1
169 _PAGE_GESTURE                               = 2
170 _PAGE_DEMO                                  = 3
171 _PAGE_MOTOR_TEST                            = 5
172 _PAGE_OFFSET                                = 6
173 _PAGE_OFFSET_DIALOG1                        = 7
174 _PAGE_OFFSET_DIALOG2                        = 8
175 _PAGE_OFFSET_DIALOG_CLOSE = 9
176
177 #Offset applied value in normal motion, not motion
178 #                           0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
179 aOffset = [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
```

Page 4

```
180
181 btnList                                  = None
182 fAngle_Cam                               = 0.0
183 tmrTorqBtn                               = CTimer()
184
185 IsPhone                                  = False
186 Line_Back                                = 0
187 Motion_Prev                              = 0
188
189 move_Back = 0
190 move_Go = 0
191 move_Right = 0
192 move_Left = 0
193 Shaking = 0
```

```
194
195 def DelayForRun(nMilliSecond):
196         delay(nMilliSecond)
197 def ShowPage(nPage):
198         global markerList
199         global btnList
200
201         global move_Back
202         global move_Go
203         global move_Right
204         global move_Left
205         global Shaking
206
207         CVar.nPage_Prev = CVar.nPage
208         CVar.nPage = nPage
209
210         # Create a page here and assign the button to it. -step 2/2
211         if nPage == _PAGE_MENU:
212                 btnList = [
213                         _BTN_SUB_OUT,
214                         _BTN_SUB_X,
215                         _BTN_SUB_REMOTE,
216                         _BTN_SUB_GESTURE,
217                         _BTN_SUB_DEMO,
218                         _BTN_SUB_MOTOR,
219                         _BTN_SUB_OFFSET
220                         ]
221                 #Shows the menu bar
222                 Show_Image(500, 850, 1)
223
224                 #Smartphone landscape mode
225                 #SetSmartphoneOrientation()
226                 #rpi.mode(0)#0: Close, 1: Color, 2: Face Detection, 3: Streaming, 4: Marker, 5: Lane, 6: Emotion, 7: Hand
227                 #smart.write8(10700, 0) # smartphone streaming video screen 0: Close, 1: Display
228
229
230                 Other buttons except #_BTN_SUB_OUT have images.
231                 img = [2,3,4,5,7,8]
232                 #print (len (btnList))
233                 nIndex = 0
234                 for btn in btnList :
235                         if (btn != _BTN_SUB_OUT):
236                                 Show_Image(
237                                         round((btn[0] + btn[2]) / 2),
238                                         round((btn[1] + btn[3]) / 2),
239                                         img[nIndex]
240                                 )
241                                 nIndex = nIndex + 1
242         elif nPage == _PAGE_MAIN:
243                 Clear_All()
```

```
244
245                 if (CVar.nOrientation == 1): #portrait mode
246                         SetSmartphoneOrientation()
247
248                 btnList = [
249                         _BTN_MENU,
250                         _BTN_TURN_L,
251                         _BTN_TURN_R,
252                         _BTN_MOVE_UL,
253                         _BTN_MOVE_U,
254                         _BTN_MOVE_UR,
255                         _BTN_MOVE_DL,
256                         _BTN_MOVE_D,
257                         _BTN_MOVE_DR,
258
259                         #Torque ON/OFF button
260                         _BTN_TORQ,
261
262                         #Normal mode action button
263                         _BTN_ACT_1,
264                         _BTN_ACT_2,
265                         _BTN_ACT_3,
266                         _BTN_ACT_4,
267                         _BTN_ACT_5,
268                         _BTN_ACT_6,
269
270                         #Motion speed button
271                         _BTN_SPD
272                         ]
273                 #Show_Background(1)
274                 Show_Background(1 + CVar.nSpeed)
275
276         elif nPage == _PAGE_GESTURE:
277                 Clear_All()
278                 #Smartphone portrait mode
279                 #SetSmartphoneOrientation (False)
280                 btnList = [
281                         _BTN_MENU
282                         ]
283                 Show_Background(3)
284
285                 move_Back = 0
286                 move_Go = 0
287                 move_Right = 0
288                 move_Left = 0
289                 Shaking = 0
290
291
292         elif nPage == _PAGE_DEMO:
293                 Clear_All()
294
295                 if (CVar.nOrientation == 1): #portrait mode
296                         SetSmartphoneOrientation()
297
298                 btnList = [
299                         _BTN_MENU,
```

```
300                     _BTN_MODE_1,
301                     _BTN_MODE_2
302                     ]
303             Show_Background(5)
304             CVar.nDemoMode = 1
305             #Motion_Play(6)
306             Motion_Ready()
307             Motion_Wait()
308
309     elif nPage == _PAGE_MOTOR_TEST:
310             Clear_All()
311
312             if (CVar.nOrientation == 1): #portrait mode
313                     SetSmartphoneOrientation()
314
315             btnList = [
316                     _BTN_MENU,
317                     _BTN_ID_1,
318                     _BTN_ID_2,
319                     _BTN_ID_3,
320                     _BTN_ID_4,
321                     _BTN_ID_5,
322                     _BTN_ID_6,
323                     _BTN_ID_7,
324                     _BTN_ID_8,
325                     _BTN_ID_9,
326                     _BTN_ID_10,
327                     _BTN_ID_11,
328                     _BTN_ID_12,
329                     _BTN_ID_13,
330                     _BTN_ID_14,
331                     _BTN_ID_15,
332                     _BTN_ID_16,
333                     _BTN_ID_17
334                     ]
335             Show_Background(7)
336             Show_Motors(-1)
337             Move_Center(True)
338     elif nPage == _PAGE_OFFSET:
339             Clear_All()
340
341             if (CVar.nOrientation == 1): #portrait mode
342                     SetSmartphoneOrientation()
343
344             btnList = [
345                     _BTN_MENU,
346                     _BTN_ID_1,
347                     _BTN_ID_2,
348                     _BTN_ID_3,
349                     _BTN_ID_4,
```

```
                    _BTN_ID_5,
                    _BTN_ID_6,
                    _BTN_ID_7,
                    _BTN_ID_8,
                    _BTN_ID_9,
                    _BTN_ID_10,
                    _BTN_ID_11,
                    _BTN_ID_12,
                    _BTN_ID_13,
                    _BTN_ID_14,
                    _BTN_ID_15,
```



```
                    _BTN_ID_16,
                    _BTN_ID_17,
                    _BTN_RESET,
                    _BTN_INIT
                    ]
            Show_Background(7)
            Show_Motors_Offset(-1)
            # Reset & Init Pos
            Show_Image(900, 300, 9)

            Move_Center(True)
        elif nPage == _PAGE_OFFSET_DIALOG1:
            btnList = [
                    _BTN_DIALOG_OK,
                    _BTN_DIALOG_CANCEL,
                    _BTN_DIALOG_TORQ,
                    _BTN_DIALOG_PLUS,
                    _BTN_DIALOG_MINUS
                    ]
            Show_Dialog (1, CVar.nID)
        elif nPage == _PAGE_OFFSET_DIALOG2:
            btnList = [
                    _BTN_DIALOG_OK,
                    _BTN_DIALOG_CANCEL
                    ]
            Show_Dialog (2, CVar.nID)
        elif nPage == _PAGE_OFFSET_DIALOG_CLOSE:
            btnList = [
                    _BTN_MENU,
                    _BTN_ID_1,
                    _BTN_ID_2,
                    _BTN_ID_3,
                    _BTN_ID_4,
                    _BTN_ID_5,
                    _BTN_ID_6,
                    _BTN_ID_7,
                    _BTN_ID_8,
                    _BTN_ID_9,
                    _BTN_ID_10,
                    _BTN_ID_11,
```

```
401                             _BTN_ID_12,
402                             _BTN_ID_13,
403                             _BTN_ID_14,
404                             _BTN_ID_15,
405                             _BTN_ID_16,
406                             _BTN_ID_17,
407                             _BTN_RESET,
408                             _BTN_INIT
409                             ]
410                     # Clear-dialog background
411                     Show_Dialog(0)
412             else :
413                     btnList = [
414                             _BTN_MENU
415                             ]
416 def Show_Motors(nSelect):
417             #[off]11,12,13,... <-> [on]31,32,33,...
418             for i in range(0, 17):
419                     nAdd = 0
420                     #Selected motor displays [On]
421                     if nSelect == i + 1:
```

Page 8

```
422                             nAdd = nAdd + 20
423                     Show_Image(round((btnList[i+1][0] + btnList[i+1][2]) /
2),round((btnList[i+1][1] + btnList[i+1][3]) / 2), i + 11 + nAdd)
424
425 def Show_Motors_Offset(nSelect):
426             for i in range(0, 17):
427                     nAdd = 0
428                     #Selected motor displays [On]
429                     if nSelect == i + 1:
430                             nAdd = nAdd + 20
431                     Show_Image(round((btnList[i+1][0] + btnList[i+1][2]) /
2),round((btnList[i+1][1] + btnList[i+1][3]) / 2), i + 11 + nAdd)
432             '''
433             i = 4
434             nAdd = 0
435             if nSelect == i + 1:
436                     nAdd = nAdd + 20
437             Show_Image(round((btnList[i+1][0] + btnList[i+1][2]) / 2),round((btnList[i+1]
[1] + btnList[i+1][3]) / 2), i + 11 + nAdd)
438             '''
439 def Motion_Ready(nInit = 2):
440             # All motor torque On
441             TorqAll(True)
442             # Update the entire motor position
443             PositionUpdate()
444             if (nInit == 2) :
445                     Motion_Play(2)
446             else :
447                     Motion_Play(1)
448             Motion_Wait()
```

```
449 def Motion_Stop():
450         Motion_Play(-3)
451 def Motion_Play(nMotionIndex, nNextMotion = 0):
452         if (nMotionIndex <= 0):
453             # Stop operation (0)
454             #nMotionIndex => -3: End immediately, -2: End after executing step(Key-Frame), -1:page(motion
Unit) execution and exit, 0: page (motion unit) execution and exit after execution.
455             motion.play((int)(nMotionIndex))
456         else:
457             #print("Motion={0}, Next={1}".format(nMotionIndex, nNextMotion))
458             Setup_Speed(254, 0)
459             if (nNextMotion == 0) :
460                 motion.play((int)(nMotionIndex))
461             else :
462                 motion.play((int)(nMotionIndex), (int)(nNextMotion))
463 def Motion_Wait(bCheckButton = False, nCheckDmsSensorValue = 0) :
464         nRet = 0
465         while True:
466             if motion.status() == False:
467                 break
468             if bCheckButton:
469                 #Check the touch input of the smartphone
470                 GetTouch_Down()
471                 # Check button press
472                 nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
473                 if (nNum0 >= 0):
474                     nRet = 1
475                     break
476                 if (nCheckDmsSensorValue > 0):
```

Page 9

```
477                     valueDms = GetSensor_Dms()
478                     if valueDms >= nCheckDmsSensorValue:
479                         nRet = valueDms
480                         break
481         return nRet
482 def TorqAll(IsOn, IsSound = None):
483         TorqOnOff (-1, IsOn, IsSound)
484 def TorqOnOff(nNum, IsOn, IsSound = None):
485         if (IsOn == True) :
486             if IsSound == True :
487                 buzzer.melody(14)
488             if (nNum >= 0) :
489                 DXL(nNum).torque_on()
490             else :
491                 dxlbus.torque_on()
492                 CVar.IsTorqOn = True
493         else :
494             CVar.IsTorqOn = False
495             if IsSound == True :
496                 buzzer.melody(15)
```

```
497                     #Motion ends immediately
498                     #Motion_Play(-3)
499                     #Waiting for motion to end
500                     #waitMotionStop()
501                     if (nNum >= 0) :
502                             540 (nnum) .torque_off ()
503                     else :
504                             dxlbus.torque_off()
505
506 def GetResolution():
507             for i in range(0, 100):
508                     screen = smart.read32(10460)
509                     CVar.nScreenWidth = screen & 0x0000FFFF
510                     CVar.nScreenHeight = (screen & 0xFFFF0000) >> 16
511                     if CVar.nScreenWidth > 0 and CVar.nScreenWidth < 65535 and
CVar.nScreenHeight > 0 and CVar.nScreenHeight < 65535:
512                             break
513
514 def GetTouch_Down():
515             # 1 2 3 4 5
516             # 6 7 8 9 10
517             # 11 12 13 14 15
518             # 16 17 18 19 20
519             # 21 22 23 24 25
520             if (smart.is_connected() == True):
521                     XY_X0 = 0
522                     XY_Y0 = 0
523                     Pos_X0 = 0
524                     Pos_Y0 = 0
525                     Pos0 = 0
526                     XY_X1 = 0
527                     XY_Y1 = 0
528                     Pos_X1 = 0
529                     Pos_Y1 = 0
530                     Pos1 = 0
531
532                     # Touch - First
533                     Tmp = smart.read64(10470) # Touch input coordinate
534                     nTouch0 = Tmp[0] & 0xffffffff
535                     nTouch1 = Tmp[1] & 0xffffffff
```



```
536                     IsChanged = False
537                     if (nTouch0 > 0) :
538                             XY_X0 = nTouch0 & 0x0000FFFF
539                             XY_Y0 = (nTouch0 >> 16) & 0x0000FFFF
540                             Pos_X0 = (int)((XY_X0 / CVar.nScreenWidth) * 5 + 1)
541                             Pos_Y0 = (int)((XY_Y0 / CVar.nScreenHeight) * 5 + 1)
542                             Pos0 = Pos_X0 + (Pos_Y0 - 1) * 5
543                             XY_X0 = (int)(XY_X0 * _RATIO / CVar.nScreenWidth)
544                             XY_Y0 = (int)(XY_Y0 * _RATIO / CVar.nScreenHeight)
545                             if (nTouch1 > 0) :
```

```
546                                 XY_X1 = nTouch1 & 0x0000FFFF
547                                 XY_Y1 = (nTouch1 >> 16) & 0x0000FFFF
548                                 Pos_X1 = (int)((XY_X1 / CVar.nScreenWidth) * 5 + 1)
549                                 Pos_Y1 = (int)((XY_Y1 / CVar.nScreenHeight) * 5 + 1)
550                                 Pos1 = Pos_X1 + (Pos_Y1 - 1) * 5
551                                 XY_X1 = (int)(XY_X1 * _RATIO / CVar.nScreenWidth)
552                                 XY_Y1 = (int)(XY_Y1 * _RATIO / CVar.nScreenHeight)
553
554                         CVar.nTouch_Pos0 = Pos0
555                         CVar.nTouch_Pos_X0 = Pos_X0
556                         CVar.nTouch_Pos_Y0 = Pos_Y0
557                         CVar.nTouch_XY_X0 = XY_X0
558                         CVar.nTouch_XY_Y0 = XY_Y0
559
560                         CVar.nTouch_Pos1 = Pos1
561                         CVar.nTouch_Pos_X1 = Pos_X1
562                         CVar.nTouch_Pos_Y1 = Pos_Y1
563                         CVar.nTouch_XY_X1 = XY_X1
564                         CVar.nTouch_XY_Y1 = XY_Y1
565                 else :
566                         CVar.nTouch_Pos0 = 0
567                         CVar.nTouch_Pos_X0 = 0
568                         CVar.nTouch_Pos_Y0 = 0
569                         CVar.nTouch_XY_X0 = 0
570                         CVar.nTouch_XY_Y0 = 0
571
572                         CVar.nTouch_Pos1 = 0
573                         CVar.nTouch_Pos_X1 = 0
574                         CVar.nTouch_Pos_Y1 = 0
575                         CVar.nTouch_XY_X1 = 0
576                         CVar.nTouch_XY_Y1 = 0
577         else :
578                 IsPhone = False
579 # If you enter a coordinate value between 1 and 100, it is converted to a coordinate value suitable for a smartphone.
580 def Set(nX, nY):
581         nResX = nX * CVar.nScreenWidth / _RATIO
582         nResY = nY * CVar.nScreenHeight / _RATIO
583         return nResX, nResY
584
585 def IsButton(nX, nY, Btn):
586         if (Btn[0] < 0):
587                 right = (Btn[0] * -200)
588                 left = right - 200
589                 bottom = (Btn [1] * -200)
590                 top = bottom - 200
591                 if ((nY >= top) and (nY <= bottom)):
592                         if ((nX >= left) and (nX <= right)):
593                                 return True
594         else:
595                 if ((nY >= Btn[1]) and (nY <= Btn[3])):
596                         if ((nX >= Btn[0]) and (nX <= Btn[2])):
```

```
597                                     return True
598             return False
599
600 # motor position update
601 def PositionUpdate():
602             etc.write8(65,3)
603             while(True) :
604                     if etc.read8(65) == 0 :
605                             break
606
607 #Print number simple test
608 def Show_Num(nX, nY, nValue, nColor = _COLOR_NONE, nSize = None):
609             if (nSize == 0):
610                     Clear_Num(nX, nY)
611             elif (nColor == _COLOR_NONE):
612                     Show(_SHOW_NUM, nX, nY, 60, _COLOR_RED, nValue)
613             elif (nSize == None):
614                     Show(_SHOW_NUM, nX, nY, 60, nColor, nValue)
615             else:
616                     Show(_SHOW_NUM, nX, nY, nSize, nColor, nValue)
617 def Show_Point(nX, nY, nColor, nSize = None):
618             if (nSize == None):
619                     Show(_SHOW_SHAPE, nX, nY, 20, nColor, 1)
620             else:
621                     Show(_SHOW_SHAPE, nX, nY, nSize, nColor, 1)
622 def Show_Text(nX, nY, nValue, nColor = _COLOR_NONE, nSize = None):
623             if (nColor == _COLOR_NONE):
624                     Show(_SHOW_TEXT, nX, nY, 60, _COLOR_RED, nValue)
625             elif (nSize == None):
626                     Show(_SHOW_TEXT, nX, nY, 60, nColor, nValue)
627             else:
628                     Show(_SHOW_TEXT, nX, nY, nSize, nColor, nValue)
629 def Show_Image(nX, nY, nValue, nSize = None):
630             if (nSize == None):
631                     Show(_SHOW_IMAGE, nX, nY, 1, 0, nValue)
632             else:
633                     Show(_SHOW_SHAPE, nX, nY, nSize, 0, nValue)
634 def Show_Background(nValue):
635             if (nValue != CVar.nBack_Background):
636                     smart.display.back_image(nValue)
637                     CVar.nBack_Background = nValue
638 #nShowType: 0-picture, 1-letter, 2-shape, 3-number
639 # nValue ([Image-Index], [Shape-1: Circle, 2: Square, 3: Triangle], [Text-Index], [Num-
Value])
640 # nColor (0: Unknown, 1: White, 2: Black, 3: Red, 4: Green, 5: Blue 6: Yellow, 7: Light Gray, 8: Gray,
9: dark gray)
641 def Show(nShowType, nX, nY, nSize, nColor, nValue):
642             if ((nShowType >= 0) and (nShowType < 4)):
643                     nX, nY = Set (nX, nY)
644                     smart.write32(10480, int(nX) | (int(nY) << 16))
645                     nTmp = nValue * 256 + nSize * 65536 + nColor * 16777216
646                     if (nShowType == _SHOW_IMAGE): # Do not use the color value of 0xff000000 digits
647                             smart.display.front_image(nTmp & 16777215) # & 0xffffff(=16777215)
648                     elif (nShowType == _SHOW_SHAPE) :
649                             smart.display.shape(nTmp)
```

```
650                     elif (nShowType == _SHOW_TEXT) :
651                             smart.display.text(nTmp)
652                     elif (nShowType == _SHOW_NUM) :
653                             smart.display.number(nTmp)
654 '''
```



```
655 def Clear(nShowType, nX, nY):
656             if ((nShowType >= 0) and (nShowType < 4)):
657                     nX, nY = Set (nX, nY)
658                     smart.write32(10480, int(nX) | (int(nY) << 16))
659                     if (nShowType == _SHOW_IMAGE) :
660                             smart.display.front_image(0)
661                     elif (nShowType == _SHOW_SHAPE) :
662                             smart.display.shape(0)
663                     elif (nShowType == _SHOW_TEXT) :
664                             smart.display.text(0)
665                     elif (nShowType == _SHOW_NUM) :
666                             smart.display.number(0)
667 '''
668 def Clear_All():
669             smart.write32(10480, 0)
670             smart.display.front_image(0)
671             smart.display.shape(0)
672             smart.display.text(0)
673             smart.display.number(0)
674 #def Clear_Image():
675 # smart.write32(10480, 0)
676 # smart.display.front_image(0)
677 def Clear_Shape():
678             smart.write32(10480, 0)
679             smart.display.shape(0)
680 def Clear_Text():
681             smart.write32(10480, 0)
682             smart.display.text(0)
683 def Clear_Num(nX = None, nY = None):
684             if (nX == None) or (nY == None) :
685                     smart.write32(10480, 0)
686                     smart.display.number(0)
687             else :
688                     Show(_SHOW_NUM, nX, nY, 0, 0, 0)
689 # Function that converts angle to data used in motor (float -> int)
690 def CalcAngle2Raw (fAngle):
691             if fAngle == None:
692                     fAngle = 0.0
693             return (int) (round (fAngle * 4096.0 / 360.0 + 2048.0))
694 # Function that converts data used in motor to angle value (int -> float)
695 def Get_Position(nID):
696             return DXL(nID).present_position() - aOffset[nID]
697 def CalcRaw2Angle (nRaw):
698             if nRaw == None :
699                     nRaw = 0
```

```
700             return (float) (360.0 * ((nRaw - 2048.0) / 4096.0))
701 def Rad2Deg(rad):
702             return rad * 180.0 / math.pi
703 def Deg2Rad(Angle):
704             return Angle * math.pi / 180.0
705 '''
706 def Kine_Angle (nID):
707             # 5, 6, 8 opposite
708             #9, 10, 11 opposite
709             #16 opposite
710             fAngle = CalcRaw2Angle (Get_Position (nID))
711             if ((nID == 5) or (nID == 6) or ((nID> = 8) and (nID <= 11)) or (nID == 16)):
712                     fAngle = fAngle * -1
713             return fAngle
714             #return Deg2Rad (fAngle)
```

Page 13

```
715 '''
716 def Sin (fAngle):
717             return math.sin (Deg2Rad (fAngle))
718 def Cos (fAngle):
719             return math.cos (Deg2Rad (fAngle))
720 def Offset_GetData(nID):
721             nValue = etc.read16 (200 + nID * 2)
722             if (nValue > 32767) :
723                     nValue = (65536 - nValue) * -1
724             return int(nValue)
725 def Offset_Read() :
726             global aOffset
727             etc.write8(199,1)
728             delay(50)
729             nMot_First = 17
730             nMot_Cnt = 1
731             #print("Offset=")
732             for i in range(nMot_First,nMot_First + nMot_Cnt):
733                     nID = i # i + 1
734                     aOffset [nID] = Offset_GetData (nID)
735                     #print (nID, aOffset [nID])
736 def Offset_Write() :
737             etc.write8(199,2)
738 def Offset_Clear() :
739             global aOffset
740             etc.write8(199,3)
741             #aOffset.clear()
742             nCnt = only (aOffset)
743             aOffset = [0] * nCnt
744             #aOffset = [0 for i in range(nCnt)]
745 def Test1():
746             # Print background image
747             if (CVar.nTouch_Pos0 > 0):
748                     # At the moment of touch, all basic shapes are erased.
749                     Clear_All()
```

```
750
751         # Basic figures and drawings to hear ...
752         Show_Background(1)
753         #Test Image : Show_Image(x, y, image index)
754
755         if (CVar.nTouch_Pos0 > 0):
756             Clear_Num()
757             Clear_Shape()
758             # X
759             nPos = 50
760             nGap = 20
761
762             nX = CVar.nTouch_XY_X0
763             nY = CVar.nTouch_XY_Y0
764             if nX < 100 :
765                 nX = 100
766             elif nX > 900 :
767                 nX = 900
768             if nY < 100 :
769                 nY = 100
770             elif nY > 900 :
771                 nY = 900
772
773             Show_Num (nX - nPos - nGap, nY-40, (int) (CVar.nTouch_XY_X0 / 100% 10))
774             Show_Num(nX - nPos,                                 nY-40, (int)(CVar.nTouch_XY_X0 / 10 % 10))
```



```
775             Show_Num (nX - nPos + nGap, nY-40, (int) (CVar.nTouch_XY_X0% 10))
776             # Y
777             nPos = 50
778             Show_Num (nX + nPos - nGap, nY-40, (int) (CVar.nTouch_XY_Y0 / 100% 10))
779             Show_Num(nX + nPos,                                 nY-40, (int)(CVar.nTouch_XY_Y0 / 10 % 10))
780             Show_Num (nX + nPos + nGap, nY-40, (int) (CVar.nTouch_XY_Y0% 10))
781
782             Show_Num(nX-30, nY+40, CVar.nTouch_Pos_X0, _COLOR_GREEN)
783             Show_Num(nX+30, nY+40, CVar.nTouch_Pos_Y0, _COLOR_GREEN)
784
785             Show_Point(CVar.nTouch_XY_X0, CVar.nTouch_XY_Y0, _COLOR_BLUE)
786
787         if (CVar.nTouch_Pos1 > 0):
788             # X
789             nPos = 50
790             nGap = 20
791
792             nX = CVar.nTouch_XY_X1
793             nY = CVar.nTouch_XY_Y1
794             if nX < 100 :
795                 nX = 100
796             elif nX > 900 :
797                 nX = 900
798             if nY < 100 :
799                 nY = 100
```

```
800                 elif nY > 900:
801                     nY = 900
802
803                 Show_Num (nX - nPos - nGap, nY-40, (int) (CVar.nTouch_XY_X1 / 100% 10))
804                 Show_Num(nX - nPos,                 nY-40, (int)(CVar.nTouch_XY_X1 / 10 % 10))
805                 Show_Num (nX - nPos + nGap, nY-40, (int) (CVar.nTouch_XY_X1% 10))
806                 # Y
807                 nPos = 50
808                 Show_Num (nX + nPos - nGap, nY-40, (int) (CVar.nTouch_XY_Y1 / 100% 10))
809                 Show_Num(nX + nPos,                 nY-40, (int)(CVar.nTouch_XY_Y1 / 10 % 10))
810                 Show_Num (nX + nPos + nGap, nY-40, (int) (CVar.nTouch_XY_Y1% 10))
811
812                 Show_Num(nX-30, nY+40, CVar.nTouch_Pos_X1, _COLOR_GREEN)
813                 Show_Num(nX+30, nY+40, CVar.nTouch_Pos_Y1, _COLOR_GREEN)
814
815                 Show_Point(CVar.nTouch_XY_X1, CVar.nTouch_XY_Y1, _COLOR_RED)
816
817 def Test2():
818         # Check button press
819         nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
820         if (Event_Dn0) :
821             print("Down0:{0}:{1}".format(nNum0, nNum1))
822         if (Event_Dn1) :
823             print("Down1:{0}:{1}".format(nNum0, nNum1))
824         if (Event_Up0) :
825             print("Up0:{0}:{1}".format(nNum0, nNum1))
826         if (Event_Up1) :
827             print("Up1:{0}:{1}".format(nNum0, nNum1))
828         if (CVar.nTouch_Pos0 > 0):
829             # Delete numbers.
830             Clear_Num()
831             if (nNum0 >= 0) :
832                 Show_Num(450, 500, nNum0, _COLOR_RED)
833             if (nNum1 >= 0) :
834                 Show_Num(550, 500, nNum1, _COLOR_BLUE)
```



```
835
836 #Set motor operation speed (Based on Position): 0 initialization
837 def Setup_Speed(nID, nValue) :
838         DXL(nID).write32(112, nValue) # 112 : profile velocity
839 #Align all motors to the center (Motion_Offset)
840 def Move_Center(IsOffset = False) :
841         #profile speed adjustment
842         Setup_Speed(254, 20)
843         #syncwrite
844         etc.write8(1200,0)
845         etc.write16(1202,116)
846         etc.write8(1204,4)
847         nMot_First = 1
848         nMot_Cnt = 17
849         for i in range(nMot_First,nMot_First + nMot_Cnt):
```

```
                nID = i # i + 1
                nValue = 0
                if (IsOffset) :
                        nValue = Offset_GetData(nID)
                        #nValue = etc.read16 (200 + nID * 2)
                        #print(nValue)
                        #if (nValue > 32767) :
                        # nValue = (65536 - nValue) * -1
                etc.write8 (1205, nID)

                etc.write32(1206,2048 + nValue)
                etc.write8(1200,1)

        etc.write8(1200,2)
        #profile speed restore
        Setup_Speed(254, 0)
        # Align all motors to the center (Motion_Offset)

def Move (nID, fAngle, nSpeed = -1, IsOffset = False):
        if (nSpeed >= 0):
                Setup_Speed(nID, nSpeed)
        DXL (nID) .write32 (116, CalcAngle2Raw (fAngle) + aOffset [nID]) # 116: Goal position
def WaitButtonUp(nTouchMode = 0): # 0 : All, 1 : FirstTouch, 2 : SecondTouch
        while(True) :
                #Check the touch input of the smartphone
                GetTouch_Down()
                # Check button press
                nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 = GetButton(btnList)
                if (nTouchMode == 1):
                        if (nNum0 < 0):
                                break
                if (nTouchMode == 2):
                        if (nNum1 < 0):
                                break
                else:
                        if ((nNum0 < 0) and (nNum1 < 0)):
                                break

def GetButton(btns):
        nNum0 = -1
        nNum1 = -1
        nDown0 = 0
        nDown1 = 0
        nUp0 = 0
        nUp1 = 0
        Btn0 = None
        Btn1 = None
        nnum = 0
```

```
898             nCnt = 0
899             if (CVar.nTouch_Pos0 > 0) :
900                     nCnt = nCnt + 1
901             if (CVar.nTouch_Pos1 > 0) :
902                     nCnt = nCnt + 1
903
904             nPass = 0
905             for btn in btns:
906                     if (CVar.nTouch_Pos0 > 0):
907                             if (IsButton(CVar.nTouch_XY_X0, CVar.nTouch_XY_Y0, btn) == True) :
908                                     nNum0 = btn [_BTN_INDEX]
909                                     Btn0 = btn
910                                     nPass = nPass | 0x01
911                     if (CVar.nTouch_Pos1 > 0):
912                             if (IsButton(CVar.nTouch_XY_X1, CVar.nTouch_XY_Y1, btn) == True) :
913                                     nNum1 = btn [_BTN_INDEX]
914                                     Btn1 = btn
915                                     nPass = nPass | 0x10
916                     if (nCnt == 1) :
917                             if (nPass > 0):
918                                     break
919                     else:
920                             if (nPass == 0x11):
921                                     break
922                     nnum + 1 = nnum
923
924             if ((nNum1 == nNum0) and (nNum0 >= 0)):
925                     nNum1 = -1
926
927             # switching
928             #IsSwitching = False
929             if (((nNum0 >= 0) and (nNum0 == CVar.nButton_1)) or ((nNum1 >= 0) and (nNum1 ==
CVar.nButton_0))) :
930                     nNum2 = nNum1
931                     nNum1 = nNum0
932                     nNum0 = nNum2
933                     #IsSwitching = True
934             #Button Down Event
935             if (nNum0 >= 0):
936                     if (CVar.nButton_0 != nNum0):
937                             nDown0 = 1
938                     CVar.btn0 = Btn0
939             else :
940                     if (CVar.nButton_0 >= 0):
941                             nUp0 = 1
942             if (nNum1 >= 0):
943                     if (CVar.nButton_1 != nNum1):
944                             nDown1 = 1
945                     CVar.btn1 = Btn1
946             else :
947                     if (CVar.nButton_1 >= 0):
948                             nUp1 = 1
949
950             CVar.nButton_0 = nNum0
951             CVar.nButton_1 = nNum1
952             if nUp0 == 1:
```

```
953                     Btn0 = CVar.btn0
954             if nUp1 == 1:
955                     Btn1 = CVar.btn1
956
957             return nNum0, nNum1, nDown0, nDown1, nUp0, nUp1, Btn0, Btn1
958 def WaitMotor (nID):
959             tmr = CTimer ()
960             tmr.Set()
961             nPass = 0
962             while(True) :
963                     nMoving = DXL(nID).read8(122)
964                     if ((tmr.Get() >= 200) and (nPass == 0)) :
965                             break
966                     elif (nMoving! = 0):
967                             nPass = 1
968                     elif ((nMoving == 0) and (nPass == 1)) :
969                             nPass = 2
970                             break
971             tmr = None
972
973 def Page_MenuBar(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
974             if (Btn0 == _BTN_SUB_OUT):
975                     ShowPage(CVar.nPage_Prev)
976             #When pressing the outside part while the menu is displayed
977             elif (Btn0 == _BTN_SUB_REMOTE):
978                     ShowPage(_PAGE_MAIN)
979             elif (Btn0 == _BTN_SUB_GESTURE):
980                     ShowPage(_PAGE_GESTURE)
981             elif (Btn0 == _BTN_SUB_DEMO):
982                     ShowPage(_PAGE_DEMO)
983             elif (Btn0 == _BTN_SUB_MOTOR):
984                     ShowPage(_PAGE_MOTOR_TEST)
985             elif (Btn0 == _BTN_SUB_OFFSET):
986                     ShowPage(_PAGE_OFFSET)
987                     #print(aOffset)
988             elif (Btn0 == _BTN_SUB_X):
989                     ShowPage(CVar.nPage_Prev)
990
991             #Check the touch input of the smartphone
992             if ((nNum0 >= 0) or (nNum1 >= 0)):
993                     WaitButtonUp()
994 def Page_Main(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
995             global Motion_Prev
996             #Torque ON/OFF button-Since torque off is a function to pay attention to, it must be operated when only one touch comes in.
All.
997             if (Btn0 == _BTN_TORQ):# or (Btn1 == _BTN_TORQ):
998                     if Event_Dn0 == 1:
999                             tmrTorqBtn.Set()
100
```

```
                    if (CVar.IsTorqOn == True):
                        TorqAll(False, True)
                    else :
                        TorqAll(True, True)
                        Motion_Ready()
```



```
                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()

        if (Btn0 == _BTN_SPD) or (Btn1 == _BTN_SPD):
            CVar.nSpeed = (CVar.nSpeed + 1) % 2
            Show_Background(1 + CVar.nSpeed)
            if (Btn0 == _BTN_SPD):
                WaitButtonUp(1)
```

```
                else :
                        WaitButtonUp(2)
        nMotion = 0
        if (Btn0 == _BTN_TURN_L) or (Btn1 == _BTN_TURN_L):
                nMotion = 14 + CVar.nSpeed * 2 # 14, 16
        if (Btn0 == _BTN_TURN_R) or (Btn1 == _BTN_TURN_R):
                nMotion = 15 + CVar.nSpeed * 2 # 15, 17
        if (Btn0 == _BTN_MOVE_UL) or (Btn1 == _BTN_MOVE_UL):
                nMotion = 18 + CVar.nSpeed * 2 # 18, 20
        if (Btn0 == _BTN_MOVE_U) or (Btn1 == _BTN_MOVE_U):
```



```
                nMotion = 10 + CVar.nSpeed                                      # 10, 11
        if (Btn0 == _BTN_MOVE_UR) or (Btn1 == _BTN_MOVE_UR):
                nMotion = 19 + CVar.nSpeed * 2 # 19, 21
        if (Btn0 == _BTN_MOVE_DL) or (Btn1 == _BTN_MOVE_DL):
                nMotion = 22 + CVar.nSpeed * 2 # 22, 24
        if (Btn0 == _BTN_MOVE_D) or (Btn1 == _BTN_MOVE_D):
                nMotion = 12 + CVar.nSpeed                                      # 12, 13
        if (Btn0 == _BTN_MOVE_DR) or (Btn1 == _BTN_MOVE_DR):
                nMotion = 23 + CVar.nSpeed * 2 # 23, 25


        if (nMotion > 0):
                Motion_Play(nMotion, nMotion)
                Motion_Wait()
        #If there are no more commands after the walking motion is over, return to the default posture
        if (Motion_Prev != nMotion):
```

```
105                 if (nMotion == 0):
105                     Motion_Play(2)
105                     Motion_Wait()
105
105
105                 Motion_Prev = nMotion
105
105             nAction = 0
105             if Event_Dn0 == 1:
105                 if (Btn0 == _BTN_ACT_1):
105                     nAction = 3
106                 if (Btn0 == _BTN_ACT_2):
106                     nAction = 4
106                 if (Btn0 == _BTN_ACT_3):
106                     nAction = 5
106                 if (Btn0 == _BTN_ACT_4):
```



```
106                     nAction = 6
106                 if (Btn0 == _BTN_ACT_5):
106                     nAction = 7
106                 if (Btn0 == _BTN_ACT_6):
106                     nAction = 8
107             if Event_Dn1 == 1:
107                 if (Btn1 == _BTN_ACT_1):
107                     nAction = 3
107                 if (Btn1 == _BTN_ACT_2):
107                     nAction = 4
107
```

```
        if (Btn1 == _BTN_ACT_3):
            nAction = 5
        if (Btn1 == _BTN_ACT_4):
            nAction = 6
        if (Btn1 == _BTN_ACT_5):
            nAction = 7
        if (Btn1 == _BTN_ACT_6):
            nAction = 8
    if (nAction > 0):
        Motion_Play(nAction)
        Motion_Wait()


def Page_Gesture(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
    global move_Back
    global move_Go
    global move_Right
    global move_Left
    global Shaking

    nMotion = 0
```



```
    nMotion_Next = 0
    if Shaking > 50 :
        Motion_Play(5,3) # Stinger
        Motion_Wait()
        Motion_Play(3) # Stinger
```

```
                Motion_Wait()
    else :
                if (move_Go >= 30):
                        if (move_Right >= 30):
                                nMotion = 19 # Right while moving forward
                        elif (move_Left >= 30):
                                nMotion = 18 #Left during forward
                        else:
                                nMotion = 10 #forward
                elif (move_Back >= 30):
                        if (move_Right >= 30):
                                nMotion = 23 #Right in reverse
                        elif (move_Left >= 30):
                                nMotion = 22 #Left during reverse
                        else:
                                nMotion = 12 #reverse
                else:
                        if (move_Right >= 30):
                                nMotion = 15 # turn right
                        elif (move_Left >= 30):
                                nMotion = 14 # turn left
                nMotion_next = nMotion
    if (nMotion > 0):
                Motion_Play(nMotion, nMotion_next)
```

```
105         move_Back = smart.sensor.tilt_up() # Back
106         move_Go = smart.sensor.tilt_down() # Go
107         move_Right = smart.sensor.tilt_left() # move right
108         move_Left = smart.sensor.tilt_right()# move left
109         Shaking = smart.sensor.shaking()
110
111         if (nMotion > 0):
112             Motion_Wait()
113
114         '''
115         if valueUp >= 70 and valueUp <= 90:
116             #Motion_Stop()#stop
117             pass
118         elif valueUp <= 50:
119             if valueRight == 90 or valueLeft == 90:
120                 nMotion = 14 # turn left
121             elif valueRight < 20 and valueLeft < 20:
122                 nMotion = 15 # turn right
123             elif valueRight >= 50:
124                 nMotion = 10 #forward
125             elif valueLeft >= 50:
126                 nMotion = 12 #reverse
127         if (nMotion > 0):
128             Motion_Play(nMotion)
129             Motion_Wait()
130         '''
131
132 def GetSensor_Dms(nChannel = 1):
133         nDms = OLLO(nChannel, const.OLLO_DMS).read()
```

```
115 3             return nDms
115 4
```

Page 23

```
115 5 def Page_Demo(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
115 6             if (Btn0 == _BTN_MODE_1):
115 7                 Show_Background(5)
115 8                 CVar.nDemoMode = 1
115 9
116 0             if (Btn0 == _BTN_MODE_2):
116 1                 Show_Background(6)
116 2                 CVar.nDemoMode = 2
116 3             ########################################################
116 4             if (CVar.nDemoMode == 1):
116 5                 tmr = CTimer ()
116 6                 tmr.Set()
116 7                 nAttack = 0
116 8                 nTurn = 0
116 9                 nDmsLimit = 250
117 0                 IsAttack = False
117 1                 lstRandom = [6,8,11,2,4,10, 11, 11, 10, 10] # Dreamer, Shake, Crawlx2,
ready, guard, drawl
117 2                 while True:
117 3                     #Check the touch input of the smartphone
117 4                     GetTouch_Down()
117 5                     # Check button press
117 6                     nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
```

```
                if (nNum0 >= 0):
                    break


                nDms = GetSensor_Dms()
                if nDms < nDmsLimit:
                    if (tmr.Get() > 100):
                        nAttack = 0
```

Page 24

```
                        nTurn = 0
                        nMotionNum = urandom.choice(lstRandom)#(1,len(lstRandom) + 1)
                        Motion_Play(nMotionNum)
                        nRet = Motion_Wait(True, nDmsLimit) # Button input, set DMS value or more
Stop motion when entering
                        # nRet => 0: Normal end, 1: button, 2~: Dms Value
                        if nRet != 0:
                            Motion_Stop()
                            Motion_Wait()
                            if (nRet == 1): # Button
                                Motion_Ready()
                                Motion_Wait()
                                break
                            else: # Dms Sensor
                                IsAttack = True
                        tmr.Set()
                else :
                    IsAttack = True
```

```
                    if (IsAttack == True):
                        IsAttack = False
                        nAttack = nAttack + 1


                        if (nAttack > 3):
                            if (nTurn < 3):
                                nTurn = nTurn + 1
                                Motion_Play(14) # Turn Left
                            else:
                                Motion_Play(23) # crawlRightBack
                        else:
```



```
                            Motion_Play(9) # Stinger : 5, Stinger2 : 9
                        nRet = Motion_Wait(True)
                        if (nRet == 1): # Button
                            Motion_Stop()
                            Motion_Wait()
                            Motion_Ready()
                            Motion_Wait()
                            break
                        tmr.Set()
    elif (CVar.nDemoMode == 2):
        tmr = CTimer ()
        tmr.Set()
        IsAttack = False
```

```
    nDmsLimit = 250
    while True:
        #Check the touch input of the smartphone
        GetTouch_Down()
        # Check button press
        nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
        if (nNum0 >= 0):
            break

        nDms = GetSensor_Dms()
        if nDms < nDmsLimit:
            if (tmr.Get() > 2000):
                Motion_Play(7)
                nRet = Motion_Wait(True, nDmsLimit) # Button input, set DMS value or more
Stop motion when entering

                # nRet => 0: Normal end, 1: button, 2~: Dms Value
                if nRet != 0:
```



```
                    Motion_Stop()
                    Motion_Wait()
                    if (nRet == 1): # Button
                        Motion_Ready()
                        Motion_Wait()
                        break
                    else: # Dms Sensor
                        IsAttack = True
```

```
                                        tmr.Set()
                                else :
                                        IsAttack = True


                                if (IsAttack == True):
                                        IsAttack = False
                                        Motion_Play(9) # Stinger : 5, Stinger2 : 9
                                        nRet = Motion_Wait(True)
                                        if (nRet == 1): # Button
                                                Motion_Stop()
                                                Motion_Wait()
                                                Motion_Ready()
                                                Motion_Wait()
                                                break
                                        tmr.Set()


def Page_MotorTest(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
        if (Event_Dn0 == 1):
                nID = -1
                #Get the index of the first actuator.
                nFirstIndex = _BTN_ID_1[4]
                for i in range(0,18):
```



```
                        nButtonIndex = i + nFirstIndex
                        if (nButtonIndex == nNum0) or (nButtonIndex == nNum1):
                                nID = i + 1
```

```
    5
127
    6                   if nID >= 0:
127
    7                           Show_Motors(nID)
127
    8                           #All LEDs Off
127
    9                           DXL(254).write8(65, 0)
128
    0                           #LED on selected motor
128
    1                           DXL(nID).write8(65, 1)
128
    2                           fRange = 10
128
    3                           # test moving
128
    4                           Move (nID, 0, 30)
128
    5                           WaitMotor (nID)
128
    6
128
    7                           Move(nID, fRange, 30)
128
    8                           #DXL(nID).write32(116, 2098)
128
    9                           WaitMotor (nID)
129
    0
129
    1                           Move(nID, -fRange, 30)
129
    2                           #DXL(nID).write32(116, 1998)
129
    3                           WaitMotor (nID)
129
    4
129
    5                           Move (nID, 0, 30)
129
    6                           #DXL(nID).write32(116, 2048)
129
    7                           WaitMotor (nID)
129
    8
129
    9                           Setup_Speed(nID, 0)
130
    0 def Page_Offset(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
130
    1               if (Event_Dn0 == 1):
```

```
130
    2                   nID = -1
130
    3                   #Get the index of the first actuator.
130
    4                   nFirstIndex = _BTN_ID_1[4]
130
    5                   for i in range(0,18):
130
    6                           nButtonIndex = i + nFirstIndex
130
    7                           if (nButtonIndex == nNum0) or (nButtonIndex == nNum1):
130
    8                                   nID = i + 1
130
    9
131
    0                   if (nID >= 0):
131
    1                           Show_Motors_Offset(nID)
131
    2                           #All LEDs Off
131
    3                           DXL(254).write8(65, 0)
131
    4                           #LED on selected motor
131
    5                           DXL(nID).write8(65, 1)
131
    6
131
    7                           CVar.nID = nID
131
    8                           CVar.nOffset = Offset_GetData(nID)
131
    9                           ShowPage(_PAGE_OFFSET_DIALOG1)
132
    0                           WaitButtonUp()
132
    1                   else:
132
    2                           CVar.nID = -1
132
    3                           if (Btn0 == _BTN_RESET):
132
    4                                   ShowPage(_PAGE_OFFSET_DIALOG2)
132
    5                                   WaitButtonUp()
132
    6                                   #CVar.nOffset = 0
132
    7                                   CVar.IsResetCommand = True
```

```
132 8                       elif (Btn0 == _BTN_INIT):
132 9                           TorqAll(True,False)
133 0                           Move_Center(True)
133 1 def Page_Offset_Dialog_1(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1,
```

Page 29

```
Btn0, Btn1):
133 2       #+
133 3       if (Btn0 == _BTN_DIALOG_PLUS):
133 4           if (DXL(CVar.nID).read8(64) == 0):
133 5               #Torq Off -> On
133 6               TorqOnOff(CVar.nID, True, False)
133 7               Show_Image(600, 400, 53)
133 8           CVar.nOffset = CVar.nOffset + 1
133 9           etc.write16(200 + CVar.nID * 2, (int)(CVar.nOffset))
134 0           #Setup_Speed(CVar.nID, 0)
134 1           DXL(CVar.nID).goal_position(2048 + (int)(CVar.nOffset))
134 2           #print(CVar.nOffset)
134 3       #-
134 4       if (Btn0 == _BTN_DIALOG_MINUS):
134 5           if (DXL(CVar.nID).read8(64) == 0):
134 6               #Torq Off -> On
134 7               TorqOnOff(CVar.nID, True, False)
134 8               Show_Image(600, 400, 53)
134 9           CVar.nOffset = CVar.nOffset - 1
135 0           etc.write16(200 + CVar.nID * 2, (int)(CVar.nOffset))
135 1           DXL(CVar.nID).goal_position(2048 + (int)(CVar.nOffset))
135
```

```
1352            #print(CVar.nOffset)
1353
1354        if (CVar.nID > 0):
1355            Show_Num_Offset (CVar.nID)
1356
1357        if (Event_Dn0 == 1):
1358            if (Btn0 == _BTN_DIALOG_OK):
1359
1360                if (DXL(CVar.nID).read8(64) == 0):
1361                    CVar.nOffset = DXL(CVar.nID).goal_position() - 2048
```



```
1362
1363                # Put Offset data
1364                etc.write16(200 + CVar.nID * 2, (int)(CVar.nOffset))
1365
1366                # Save
1367                ShowPage(_PAGE_OFFSET_DIALOG2)
1368                TorqAll(True, False)
1369
1370            if (Btn0 == _BTN_DIALOG_CANCEL):
1371                ShowPage(_PAGE_OFFSET)
1372                #ShowPage(_PAGE_OFFSET_DIALOG_CLOSE)
1373            if (Btn0 == _BTN_DIALOG_TORQ):
1374                if (CVar.nID > 0) :
1375                    if (DXL(CVar.nID).read8(64) == 0):
1376                        #Torq Off -> On
1377                        TorqOnOff (CVar.nID, True)
```

```
137                                     Show_Image(600, 400, 53)
137
 9
138
 0                                      CVar.nOffset = DXL(CVar.nID).goal_position() - 2048
138
 1                                      etc.write16(200 + CVar.nID * 2, (int)(CVar.nOffset))
138
 2                              else :
138
 3                                      #Torq On -> Off
138
 4                                      TorqOnOff (CVar.nID, False)
138
 5                                      Show_Image(600, 400, 54)
138
 6
138
 7                                      etc.write16(200 + CVar.nID * 2, (int)(CVar.nOffset))
138
 8 def Page_Offset_Dialog_2(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1,
Btn0, Btn1):
138
 9          global aOffset
139
 0          if (Event_Dn0 == 1):
139
 1                  if (Btn0 == _BTN_DIALOG_OK):
```



```
139
 2                          if (CVar.IsResetCommand == True):
139
 3                                  Offset_Clear()
139
 4                          else :
139
 5                                  Offset_Write()
139
 6                                  aOffset[CVar.nID] = CVar.nOffset
139
 7                          # Done - Close the dialog
139
 8                          ShowPage(_PAGE_OFFSET)
139
 9                          Move_Center(True)
140
 0                  if (Btn0 == _BTN_DIALOG_CANCEL):
140
 1                          ShowPage(_PAGE_OFFSET)
140
 2                          #ShowPage(_PAGE_OFFSET_DIALOG_CLOSE)
```

```
140 3
140
4 def Show_Dialog(nStep, nID=5) :
140
5       # Show Offset Dialog
140
6       if (nStep == 1) :
140
7           CVar.IsResetCommand = False
140
8           # Show dialog background
140
9           Show_Image(500, 500, 51)
141
0           #Torq On/Off Image
141
1           Show_Image(600, 400, 53)
141
2           #DXL ID display
141
3           Show_Num(475, 210, nID, _COLOR_WHITE, 80)
141
4       #Show Save Dialog
141
5       elif (nStep == 2):
141
6           #Clear all numbers
141
7           Clear_Num()
141
8           # Show dialog background
141
9           Show_Image(500, 500, 52)
142
0           #DXL ID display
142
1           Show_Num(475, 210, nID, _COLOR_WHITE, 80)
```



```
142
2           #SAVE text output
142
3           Show_Text(500,500,11,_COLOR_BLACK,100)
142
4       else :
142
5           CVar.IsResetCommand = False
142
6           #Clear-dialog background
142
7           Show_Image(500, 500, 0)
```

```
142
    8                   #Clear - Torq On/Off Image
142
    9                   Show_Image(600, 400, 0)
143
    0                   #Clear - Show Text "Save"
143
    1                   Show_Text(500,500,0)
143
    2                   #Clear all numbers
143
    3                   Clear_Num()
143
    4                   # If you close the window, do Torq On -> Move Center.
143
    5                   TorqAll(True, False)
143
    6                   if (CVar.nPage == _PAGE_OFFSET):
143
    7                           Move_Center(True)
143
    8                   else :
143
    9                           Move_Center()
144
    0
144
    1 def Show_Num_Offset(nID) :
144
    2           nOffset = DXL(nID).goal_position() - 2048
144
    3           nColor = 5
144
    4           nGap = 30
144
    5           if (nOffset < 0) :
144
    6                   nOffset = -nOffset
144
    7                   nColor = 3
144
    8           nPos_X = 420
144
    9           nPos_Y = 560
145
    0           ### Clear
145
    1           nSize = 0
```



```
145
    2           for j in range(0, 4) :
```

```
145
3                       if (nOffset < pow(10, (j + 1))) :
145
4                               for i in range(0, 4 - j) :
145
5                                       Clear_Num(nPos_X + nGap * i, nPos_Y) # Set Show_Num Size to 0
```
It can be erased, but for readability, a Clear_Num() function is created and used.
```
145
6                               break
145
7               ### Display
145
8               nSize = 120
145
9               for i in range(0, 5) :
146
0                       j = (int)(pow(10, (4 - i)))
146
1                       if (nOffset >= j) or (i == 4): The digit of # 1 (i == 4) must always be marked, so it is placed in the if statement.
```
It is tied with the or condition so that it does not occur.
```
146
2                               nValue = (int)(nOffset / j) % 10
146
3                               Show_Num(nPos_X + nGap * i, nPos_Y, nValue, nColor, nSize)
146
4 #IsHorizontal => True: Landscape mode, False: Portrait mode
146
5 def SetSmartphoneOrientation(IsHorizontal = True):
146
6               #Set the smartphone screen horizontally. (0: Auto, 1: Vertical, 2: Horizontal)
146
7               if (IsHorizontal == True):
146
8                       smart.display.screen_orientation(2)
146
9                       CVar.nOrientation = 2
147
0               else:
147
1                       smart.display.screen_orientation(1)
147
2                       CVar.nOrientation = 1
147
3               # Waiting for the screen to switch before getting the screen's width and height.
147
4               delay(500)
147
5
147
6               # Get the resolution of the screen and put it in CVar.nScreenWidth, CVar.nScreenHeight.
147
7               GetResolution()
147
8
147
9
```

```
148
    0 console(USB)
148
    1 #console (BLE)
148
    2 #console(UART)
148
    3
148
    4 # controller direction : 0-vertical(Humanoid), 1-Horizontal
148
    5 eeprom.imu_type(1)
148
    6 # Turn off the torque and modify the actuator and controller settings to suit the robot.
148
    7 TorqAll (False)
148
    8 # profile -> velocity-based
148
    9 DXL(254).write8(10, 0) # 0 -> velocity-based profile, 4 -> time-based profile
149
    0 # Secondary ID(255: No Use, 0 ~ 252: ID)
149
    1 DXL(254).write8(12, 255) # 255 -> No Use(Clear)
149
    2 # Operation Mode(1:velocity[wheel], 3:position)
149
    3 DXL(254).mode(3) # position
149
    4 TorqAll(True)
149
    5
149
    6 # In actual use, nTest = 0 should be used.
149
    7 nTest = 0 # 0-Normal, 1-Coordinate output, 2-Click test (Click the designated button position (nTest_X, nTest_Y)
Test)
149
    8
149
    9 # The following variables nTest_X, nTest_Y are used only when nTest == 2
150
    0 nTest_X = 18
150
    1 nTest_Y = 47
150
    2
150
```

```
# Return all motor speeds to initial state
Setup_Speed(254, 0)

# In the test mode, do not perform the initial position
if nTest == 0 :
        Motion_Ready()
```



```

#Read All Offset
Offset_Read()

while(True) :
        if (IsPhone == False) :
                if (smart.is_connected() == True):
                        #Check if it is connected to the smartphone.
                        smart.wait_connected()
                        SetSmartphoneOrientation()

                        # Print background image
                        ShowPage(_PAGE_MAIN)

                        # Record the smartphone connection in a variable
                        IsPhone = True
        else :
                #Check the touch input of the smartphone
                GetTouch_Down()
```

```
152
    # Test 1 => Coordinate output
152
    if (nTest == 1) :
153
        Test1()
153
    elif (nTest == 2): # Red dot when clicking with touch 0, blue dot when clicking with touch 1 (second touch)
153
        Test2 ()
153
    else: #Run
153
        # Check button press
153
        nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
153
        if (Event_Dn1 == 1):
153
            smart.etc.vibrate(10)
```

```
153
        elif (Event_Dn0 == 1):
153
            smart.etc.vibrate(10)
154
0
154
        #delay(1000)
154
        #The menu bar is floating
154
        if (CVar.nPage == _PAGE_MENU):
154
            # All motor torque On
154
            TorqAll(True)
154
            #All LEDs Off
154
            DXL(254).write8(65, 0)
154
            #Open the menu bar.
154
            Page_MenuBar(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)
155
0
155
```

```
                else: #If the menu bar is not floating
155
                    if (Btn0 == _BTN_MENU) or (Btn0 == _BTN_MENU_V): # Press the menu button
155
                        #if (CVar.nPage == _PAGE_GESTURE):
155
                        # #Smartphone landscape mode
155
                        # SetSmartphoneOrientation()
155
6
155
                        ShowPage(_PAGE_MENU) # Bring up the menu bar.
155
                        WaitButtonUp()                  # Wait for the button to be released
155
                    else:
156
                        if (CVar.nPage == _PAGE_MAIN):
156
                            Page_Main(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)
156
                        elif (CVar.nPage == _PAGE_GESTURE):
156
                            Page_Gesture(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)
156
                        elif (CVar.nPage == _PAGE_DEMO):
156
                            Page_Demo(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)
```



```
156
                        elif (CVar.nPage == _PAGE_MOTOR_TEST):
156
                            Page_MotorTest(nNum0, nNum1, Event_Dn0, Event_Dn1,
Event_Up0, Event_Up1, Btn0, Btn1)
156
                        elif (CVar.nPage == _PAGE_OFFSET):
156
                            Page_Offset(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)
157
                        elif (CVar.nPage == _PAGE_OFFSET_DIALOG1):
157
                            Page_Offset_Dialog_1(nNum0, nNum1, Event_Dn0, Event_Dn1,
Event_Up0, Event_Up1, Btn0, Btn1)
157
                        elif (CVar.nPage == _PAGE_OFFSET_DIALOG2):
157
```

3 Page_Offset_Dialog_2(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1)
157
4