```
from pycm import *
import time
import math

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

        IsResetCommand                  = False

        IsModeChanging                  = False
        # When the marker is executed, if it is in Run state, not in debug mode, run without stopping.
        IsNonstop                       = 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
```

```
44              fYaw                            = 0.0
45              fYaw_turn                       = 0.0
46
47              nPoint_Joystick                 = 2
48              nPoint_Cam                      = 3
49
50  class CTimer:
51              nTimer                          = 0
52              IsTimer                         = False
53
54              #def __init__(self):
55              # self.nTimer = 0
56              # self.IsTimer = 0
57
58              def Set(self):
59                      self.IsTimer = True
60                      self.nTimer = millis()
```

Page 2

```
61
62              def Get(self):
63                      if self.IsTimer :
64                              return millis() - self.nTimer
65                      return 0
66
67              def Destroy(self):
68                      self.IsTimer = False
69
70  _COLOR_NONE                 = 0
71  _COLOR_WHITE                = 1
72  _COLOR_BLACK                = 2
73  _COLOR_RED                  = 3
74  _COLOR_GREEN                = 4
75  _COLOR_BLUE                 = 5
76  _COLOR_YELLOW = 6
77  _COLOR_GRAY_LIGHT = 7
78  _COLOR_GRAY                 = 8
79  _COLOR_GRAY_DARK = 9
80
81  _SHOW_IMAGE                 = 0
82  _SHOW_TEXT                  = 1
83  _SHOW_SHAPE                 = 2
84  _SHOW_NUM                   = 3
85
86  _RATIO                      = 1000
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
92  #Page 0
93  #Menu-Control
```

```
 94 _BTN_SUB_OUT                 = [1,1,1000,700,60]
 95 _BTN_SUB_X                   = [950,730,990,780,61]
 96 _BTN_SUB_REMOTE = [100-50,850-100,100+50,850+100,62]
 97 _BTN_SUB_STREAM = [250-50,850-100,250+50,850+100,63]
 98 _BTN_SUB_DEMO = [400-50,850-100,400+50,850+100,64]
 99 _BTN_SUB_MOTOR = [700-50,850-100,700+50,850+100,65]
100 _BTN_SUB_OFFSET = [850-50,850-100,850+50,850+100,66]
101
102 #Page 1
103 #move button
104 _BTN_TURN_L                 = [30,310,100,440,6]
105 _BTN_TURN_R                 = [240,310,310,440,7]
106 _BTN_MOVE_UL                = [40,490,110,630,12]
107 _BTN_MOVE_U                 = [140,390,200,530,13]
108 _BTN_MOVE_UR                = [224,490,298,630,14]
109 _BTN_MOVE_DL                = [40,700,110,830,17]
110 _BTN_MOVE_D                 = [140,800,200,930,18]
111 _BTN_MOVE_DR                = [224,700,298,830,19]
112
113 #Torque ON/OFF button
114 _BTN_TORQ                  = [828,40,960,150,21]
115 '''
116 #Normal mode action button
117 _BTN_ACT_1                  = [770,460,860,580,30]
118 _BTN_ACT_2                  = [880,460,970,580,31]
119 _BTN_ACT_3                  = [770,630,860,760,32]
```

Page 3

```
120 _BTN_ACT_4                  = [880,630,970,760,33]
121 _BTN_ACT_5                  = [770,800,860,930,34]
122 _BTN_ACT_6                  = [880,800,970,930,35]
123 '''
124 #motion speed button
125 _BTN_SPD_L1                 = [430,845,520,955,50]
126 _BTN_SPD_L2                 = [430,735,520,845,51]
127 _BTN_SPD_L3                 = [430,625,520,735,52]
128 _BTN_SPD_L4                 = [430,515,520,625,53]
129 _BTN_SPD_L5                 = [430,405,520,515,54]
130
131 _BTN_SPD_R1                 = [850,845,920,955,55]
132 _BTN_SPD_R2                 = [850,735,920,845,56]
133 _BTN_SPD_R3                 = [850,625,920,735,57]
134 _BTN_SPD_R4                 = [850,515,920,625,58]
135 _BTN_SPD_R5                 = [850,405,920,515,59]
136
137 #mode button
138 _BTN_MODE_LINE = [650,30,790,150,70]
139 _BTN_MODE_MARKER = [ 830 , 30 , 950 , 150 , 71 ]
140
141 #line detection (_BTN_MODE_LINE) color button
142 _BTN_COLOR_R                = [770,317,814,402,72]
143 _BTN_COLOR_G                = [830,317,870,402,73]
```

```
144 _BTN_COLOR_B                    = [885,317,930,402,74]
145
146 #Marker detection (_BTN_MODE_MARKER) Debug / Run button
147 _BTN_MODE_DEBUG = [85,300,236,360,75]
148 _BTN_MODE_RUN = [85,390,236,450,76]
149
150 # Motor ID Tag
151 _BTN_ID_1 = [300-80,500-50,300+80,500+50,77]
152 _BTN_ID_2 = [700-80,500-50,700+80,500+50,78]
153 _BTN_ID_3 = [300-80,200-50,300+80,200+50,79]
154 _BTN_ID_4 = [700-80,200-50,700+80,200+50,80]
155 _BTN_ID_5 = [500-80,600-50,500+80,600+50,81]
156
157 # Reset & Init ( Offset )
158 _BTN_RESET = [900-70,300-250,900+70,300,82]
159 _BTN_INIT = [900-70,300,900+70,300+250,83]
160
161 #Offset Setting Dialog
162 _BTN_DIALOG_OK = [ 600 - 100 , 800 - 80 , 600 + 100 , 800 + 80 , 84 ]
163 _BTN_DIALOG_CANCEL = [400-100,800-80,400+100,800+80,85]
164 _BTN_DIALOG_TORQ = [600-80,400-50,600+80,400+50,86]
165 _BTN_DIALOG_PLUS = [ 640 - 50 , 560 - 50 , 640 + 50 , 560 + 50 , 87 ]
166 _BTN_DIALOG_MINUS = [ 360 - 50 , 560 - 50 , 360 + 50 , 560 + 50 , 88 ]
167
168 _PAGE_MENU                          = 0
169 _PAGE_MAIN                          = 1
170 _PAGE_STREAM                        = 2
171 _PAGE_MARKER                        = 3
172 _PAGE_LINE                          = 4
173 _PAGE_MOTOR_TEST                    = 5
174 _PAGE_OFFSET                        = 6
175 _PAGE_OFFSET_DIALOG1                = 7
176 _PAGE_OFFSET_DIALOG2                = 8
177 _PAGE_OFFSET_DIALOG_CLOSE = 9
178
179 #Marker instruction array
```

Page 4

```
180 maxMarkerList                      = 100
181 markerList                         = [maxMarkerList]
182
183 # Offset applied value in normal motion, not motion
184 #                          0 1 2 3 4 5 6 7 8 9 10 11 12
185 aOffset = [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
186
187 btnList                    = None
188 fAngle_Cam                 = 0.0
189 tmrTorqBtn                 = CTimer()
190
191 IsPhone                    = False
192 Line_Back                  = 0
193 Line_Back2                   = 0
```

```
194
195
196 def DelayForRun(nMilliSecond):
197         delay(nMilliSecond)
198 def RunMarkerList():
199         global markerList
200         Clear_Text()
201         nSpd = 100
202         #nDelay = 500
203         #for item in markerList:
204         nLength = len (markerList)
205         fTurn = 0
206         for i in range(0, nLength):
207             currmarker = markerList [i]
208             next = 0
209             next_next = 0
210             if (i < nLength - 1):
211                 next = markerList[i+1]
212             if (i < nLength - 2):
213                 next_next = markerList[i+2]
214             Clear_Num()
215             Clear_Text()
216             nIndex2 = 0
217             #color (0: Unknown, 1: White, 2: Black, 3: Red, 4: Green, 5: Blue 6: Yellow, 7: Light Gray,
8: gray, 9: dark gray)
218             if currmarker > 0:
219                 Show_Num(400, (400 + nIndex2 * 100), i + 1, _COLOR_RED, 60)
220             Show_Text(500, (400 + nIndex2 * 100), currmarker, _COLOR_RED, 60)
221
222             nIndex2 = 1
223             if next > 0:
224                 Show_Num(400, (400 + nIndex2 * 100), i + 2, _COLOR_GREEN, 60)
225             Show_Text(500, (400 + nIndex2 * 100), next, _COLOR_GREEN, 60)
226             nIndex2 = 2
227             if next_next > 0:
228                 Show_Num(400, (400 + nIndex2 * 100), i + 3, _COLOR_BLUE, 60)
229             Show_Text(500, (400 + nIndex2 * 100), next_next, _COLOR_BLUE, 60)
230             if currmarker == 2: #Forward
231                 Move2(0, nSpd)
232                 DelayForRun(1000)
233                 if (CVar.IsNonstop == False):
234                     Move_Stop()
235                     DelayForRun(500)
236             elif currmarker == 3: #Backward
237                 Move2(180, nSpd)
238                 DelayForRun(1000)
```



```
239                 if (CVar.IsNonstop == False):
240                     Move_Stop()
241                     DelayForRun(500)
242             elif currmarker == 4: #Turn Left
```

```
                        if (CVar.IsNonstop == False):
                            Move_Turn(90, nSpd)
                        else:
                            if (currmarker == next):
                                fTurn = fTurn + 90
                            else:
                                fTurn = fTurn + 90
                                Move_Turn(fTurn, nSpd)
                                fTurn = 0
                    elif currmarker == 5: #Turn Right
                        if (CVar.IsNonstop == False):
                            Move_Turn(-90, nSpd)
                        else:
                            if (currmarker == next):
                                fTurn = fTurn - 90
                            else:
                                fTurn = fTurn - 90
                                Move_Turn(fTurn, nSpd)
                                fTurn = 0
                    elif currmarker == 6: #Delay 1s
                        DelayForRun(1000)
            Move_Stop()
            Clear_Num()
            Clear_Text()
def AddMarkerList(marker):
            global markerList
            nLength = len (markerList)
            if (nLength < maxMarkerList):
                Clear_Text()
                markerList.append(marker)
                Clear_Text()
                if len(markerList) > 0:
                    nIndex = 0
                    nIndex2 = 0
                    for val in markerList:
                        if nIndex >= len(markerList) - 3 :
                            if nIndex == len(markerList) - 1 :
                                Show_Num(400, (400 + nIndex2 * 100), nIndex + 1,
_COLOR_RED, 60)
                                Show_Text(500, (400 + nIndex2 * 100), val, _COLOR_RED, 60)
                            elif nIndex == len (markerList) - 2:
                                Show_Num(400, (400 + nIndex2 * 100), nIndex + 1,
_COLOR_RED, 60)
                                Show_Text(500, (400 + nIndex2 * 100), val, _COLOR_GREEN,
60)
                            else:
                                Show_Num(400, (400 + nIndex2 * 100), nIndex + 1,
_COLOR_RED, 60)
                                Show_Text(500, (400 + nIndex2 * 100), val, _COLOR_BLUE, 60)
                            nIndex2 = nIndex2 + 1
                        nIndex = nIndex + 1

                buzzer.melody(12)
                delay(1000)
            else:
                buzzer.melody(4)
```

```
295                 delay(2000)
```

Page 6

```
296 def ShowPage(nPage):
297         global markerList
298         global btnList
299         CVar.nPage_Prev = CVar.nPage
300         CVar.nPage = nPage
301 
302         # Create a page here and assign the button to it. -step 2/2
303         if nPage == _PAGE_MENU:
304             btnList = [
305                 _BTN_SUB_OUT,
306                 _BTN_SUB_X,
307                 _BTN_SUB_REMOTE,
308                 _BTN_SUB_STREAM,
309                 _BTN_SUB_DEMO,
310                 _BTN_SUB_MOTOR,
311                 _BTN_SUB_OFFSET
312                 ]
313 
314             Move_Stop()
315             Move_Cam_Center()
316 
317             rpi.mode(0)#0: Close, 1: Color, 2: Face Detection, 3: Streaming, 4: Marker, 5: Lane, 6: Emotion, 7: Hand
318             smart.write8(10700, 0)# smartphone streaming video screen 0: close, 1: display
319 
320             #Shows the menu bar
321             Show_Image(500, 850, 1)
322 
323             Other buttons except #_BTN_SUB_OUT have images.
324             img = [2,3,4,5,7,8]
325             nIndex = 0
326             for btn in btnList :
327                 if (btn != _BTN_SUB_OUT):
328                     Show_Image(
329                         round((btn[0] + btn[2]) / 2),
330                         round((btn[1] + btn[3]) / 2),
331                         img[nIndex]
332                     )
333                     nIndex = nIndex + 1
334         elif nPage == _PAGE_MAIN:
335             Clear_All()
336 
337             btnList = [
338                 _BTN_MENU,
339                 _BTN_TURN_L,
340                 _BTN_TURN_R,
341                 _BTN_MOVE_UL,
342                 _BTN_MOVE_U,
343                 _BTN_MOVE_UR,
344                 _BTN_MOVE_DL,
345                 _BTN_MOVE_D,
```

```
                    _BTN_MOVE_DR,

                    #Torque ON/OFF button
                    _BTN_TORQ,

                    #Normal mode action button
                    #_BTN_ACT_1,
                    #_BTN_ACT_2,
                    #_BTN_ACT_3,
                    #_BTN_ACT_4,
```



```
                    #_BTN_ACT_5,
                    #_BTN_ACT_6,

                    #Motion speed button
                    _BTN_SPD_L1,
                    _BTN_SPD_L2,
                    _BTN_SPD_L3,
                    _BTN_SPD_L4,
                    _BTN_SPD_L5,

                    _BTN_SPD_R1,
                    _BTN_SPD_R2,
                    _BTN_SPD_R3,
                    _BTN_SPD_R4,
                    _BTN_SPD_R5
                    ]
            Show_Background(1)

            Show_Point(480, (458 + (110 * (5-CVar.nPoint_Joystick))), _COLOR_BLACK)
            Show_Point((885), (458 + (110 * (5-CVar.nPoint_Cam))), _COLOR_BLACK)

        elif nPage == _PAGE_STREAM:
            Clear_All()

            btnList = [
                    _BTN_MENU,
                    _BTN_TURN_L,
                    _BTN_TURN_R,
                    _BTN_MOVE_UL,
                    _BTN_MOVE_U,
                    _BTN_MOVE_UR,
                    _BTN_MOVE_DL,
                    _BTN_MOVE_D,
                    _BTN_MOVE_DR,

                    #Torque ON/OFF button
                    _BTN_TORQ,

                    #Normal mode action button
                    #_BTN_ACT_1,
```

```
396                     #_BTN_ACT_2,
397                     #_BTN_ACT_3,
398                     #_BTN_ACT_4,
399                     #_BTN_ACT_5,
400                     #_BTN_ACT_6,
401
402                     #Motion speed button
403                     _BTN_SPD_L1,
404                     _BTN_SPD_L2,
405                     _BTN_SPD_L3,
406                     _BTN_SPD_L4,
407                     _BTN_SPD_L5,
408
409                     _BTN_SPD_R1,
410                     _BTN_SPD_R2,
411                     _BTN_SPD_R3,
412                     _BTN_SPD_R4,
413                     _BTN_SPD_R5
414                     ]
415             Show_Background(3)
```

Page 8

```
416
417             rpi.mode(3)#0: Close, 1: Color, 2: Face Detection, 3: Streaming, 4: Marker, 5: Lane, 6: Emotion, 7: Hand
418             smart.write8(10700, 1)# smartphone streaming video screen 0: Close, 1: Display
419
420             Show_Point(480, (458 + (110 * (5-CVar.nPoint_Joystick))), _COLOR_BLACK)
421             Show_Point((885), (458 + (110 * (5-CVar.nPoint_Cam))), _COLOR_BLACK)
422         elif nPage == _PAGE_MARKER:
423             Clear_All()
424             btnList = [
425                     _BTN_MENU,
426                     _BTN_MODE_LINE,
427                     _BTN_MODE_MARKER,
428                     _BTN_MODE_DEBUG,
429                     _BTN_MODE_RUN
430                     ]
431             Show_Background(11)
432             Move_Stop()
433             Move_Cam_Center()
434             CVar.nCamera_mode = 4
435             rpi.mode(CVar.nCamera_mode) #0: Close, 1: Color, 2: Face Detection, 3: Streaming, 4: Marker, 5: Lane,
6: Emotion
436
437             markerList.clear()
438             Show_Text_Marker()
439
440         elif nPage == _PAGE_LINE:
441             Clear_All()
442
443             btnList = [
```

```
444                     _BTN_MENU,
445                     _BTN_MODE_LINE,
446                     _BTN_MODE_MARKER,
447                     _BTN_COLOR_R,
448                     _BTN_COLOR_G,
449                     _BTN_COLOR_B
450                     ]
451             Show_Background(5)
452             # Search angle
453             #Move_Cam(25, 30)
454             #Move_Cam(15, 30)
455
456             Move_Cam(45, 30)
457             #Move_Cam(30, 30)
458             CVar.nCamera_mode = 5
459             rpi.mode(CVar.nCamera_mode) #0: Close, 1: Color, 2: Face Detection, 3: Streaming, 4: Marker, 5: Lane,
6: Emotion
460             if CVar.nCamera_mode_sub == 0:
461                 CVar.nCamera_mode_sub = 1
462                 Line_Back = 0
463             rpi.sub_mode(CVar.nCamera_mode_sub) #1:red, 2:orange, 3:yellow, 4:green,
5:blue, 6:pupple, 7:white
464
465         elif nPage == _PAGE_MOTOR_TEST:
466             Clear_All()
467             btnList = [
468                 _BTN_MENU,
469                 _BTN_ID_1,
470                 _BTN_ID_2,
471                 _BTN_ID_3,
472                 _BTN_ID_4,
```

Page 9

```
473                     _BTN_ID_5
474                     ]
475             Show_Background(13)
476             Show_Motors(-1)
477             Move_Center(True)
478         elif nPage == _PAGE_OFFSET:
479             Clear_All()
480
481             btnList = [
482                 _BTN_MENU,
483                 _BTN_ID_1, # Not used (reserved)
484                 _BTN_ID_2, # not used (reserved)
485                 _BTN_ID_3, # Not used (reserved)
486                 _BTN_ID_4, # Not used (reserved)
487                 _BTN_ID_5,
488                 _BTN_RESET,
489                 _BTN_INIT
490                 ]
491             Show_Background(13)
```

```
                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, # Not used (reserved)
                        _BTN_ID_2, # not used (reserved)
                        _BTN_ID_3, # Not used (reserved)
                        _BTN_ID_4, # Not used (reserved)
                        _BTN_ID_5,
                        _BTN_RESET,
                        _BTN_INIT
                        ]
                # Clear-dialog background
                Show_Dialog(0)
        else :
                btnList = [
                        _BTN_MENU
                        ]
def Show_Motors(nSelect):
        #[off]11,12,13,... <-> [on]31,32,33,...
        for i in range(0, 5):
                nAdd = 0
```



```
                #Selected motor displays [On]
                if nSelect == i + 1:
                        nAdd = nAdd + 20
                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)

def Show_Motors_Offset(nSelect):
        i = 4
        nAdd = 0
        if nSelect == i + 1:
```

```
542                     nAdd = nAdd + 20
543             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)
544
545 def Motion_Ready(nInit = 2):
546             # All motor torque On
547             TorqAll(True)
548             # Update the entire motor position
549             PositionUpdate()
550             if (nInit == 2) :
551                     Motion_Play(2)
552             else :
553                     Motion_Play(1)
554             Motion_Wait()
555 def Motion_Play(nMotionIndex, nNextMotion = 0):
556             if (nNextMotion == 0) :
557                     motion.play((int)(nMotionIndex))
558             else :
559                     motion.play((int)(nMotionIndex), (int)(nNextMotion))
560 def Motion_Wait() :
561             while True:
562                     if motion.status() == False:
563                             break
564 def TorqAll(IsOn, IsSound = None):
565             TorqOnOff (-1, IsOn, IsSound)
566 def TorqOnOff(nNum, IsOn, IsSound = None):
567             if (IsOn == True) :
568                     if IsSound == True :
569                             buzzer.melody(14)
570                     if (nNum >= 0) :
571                             DXL(nNum).torque_on()
572                     else :
573                             dxlbus.torque_on()
574                             CVar.IsTorqOn = True
575             else :
576                     CVar.IsTorqOn = False
577                     if IsSound == True :
578                             buzzer.melody(15)
579                     #Motion ends immediately
580                     #Motion_Play(-3)
581                     #Waiting for motion to end
582                     #waitMotionStop()
583                     if (nNum >= 0) :
584                             540 (nnum) .torque_off ()
585                     else :
586                             dxlbus.torque_off()
587
588 def GetResolution():
589             for i in range(0, 100):
590                     screen = smart.read32(10460)
```

```
591                     CVar.nScreenWidth = screen & 0x0000FFFF
592                     CVar.nScreenHeight = (screen & 0xFFFF0000) >> 16
593                     if CVar.nScreenWidth > 0 and CVar.nScreenWidth < 65535 and
CVar.nScreenHeight > 0 and CVar.nScreenHeight < 65535:
594                             break
595
596 def GetTouch_Down():
597             # 1 2 3 4 5
598             # 6 7 8 9 10
599             # 11 12 13 14 15
600             # 16 17 18 19 20
601             # 21 22 23 24 25
602             if (smart.is_connected() == True):
603                     XY_X = 0
604                     XY_Y = 0
605                     Pos_X = 0
606                     Pos_Y = 0
607                     Pos = 0
608
609                     # Touch - First
610                     Tmp = smart.read64(10470) # Touch input coordinate
611                     nTouch0 = Tmp[0] & 0xffffffff
612                     nTouch1 = Tmp[1] & 0xffffffff
613                     IsChanged = False
614                     if (nTouch0 > 0) :
615                             XY_X = nTouch0 & 0x0000FFFF
616                             XY_Y = (nTouch0 >> 16) & 0x0000FFFF
617                             Pos_X = (int)((XY_X / CVar.nScreenWidth) * 5 + 1)
618                             Pos_Y = (int) ((XY_Y / CVar.nScreenHeight) * 5 + 1)
619                             Pos = Pos_X + (Pos_Y - 1) * 5
620                             XY_X = (int)(XY_X * _RATIO / CVar.nScreenWidth)
621                             XY_Y = (int)(XY_Y * _RATIO / CVar.nScreenHeight)
622
623                             CVar.nTouch_Pos0 = Pos
624                             CVar.nTouch_Pos_X0 = Pos_X
625                             CVar.nTouch_Pos_Y0 = Pos_Y
626                             CVar.nTouch_XY_X0 = XY_X
627                             CVar.nTouch_XY_Y0 = XY_Y
628
629                             if (nTouch1 > 0) :
630                                     XY_X = nTouch1 & 0x0000FFFF
631                                     XY_Y = (nTouch1 >> 16) & 0x0000FFFF
632                                     Pos_X = (int)((XY_X / CVar.nScreenWidth) * 5 + 1)
633                                     Pos_Y = (int) ((XY_Y / CVar.nScreenHeight) * 5 + 1)
634                                     Pos = Pos_X + (Pos_Y - 1) * 5
635                                     XY_X = (int)(XY_X * _RATIO / CVar.nScreenWidth)
636                                     XY_Y = (int)(XY_Y * _RATIO / CVar.nScreenHeight)
637
638                             CVar.nTouch_Pos1 = Pos
639                             CVar.nTouch_Pos_X1 = Pos_X
640                             CVar.nTouch_Pos_Y1 = Pos_Y
641                             CVar.nTouch_XY_X1 = XY_X
642                             CVar.nTouch_XY_Y1 = XY_Y
643                     else :
644                             CVar.nTouch_Pos0 = 0
```

```
645                         CVar.nTouch_Pos_X0 = 0
646                         CVar.nTouch_Pos_Y0 = 0
647                         CVar.nTouch_XY_X0 = 0
648                         CVar.nTouch_XY_Y0 = 0
649
650                         CVar.nTouch_Pos1 = 0
```



```
651                         CVar.nTouch_Pos_X1 = 0
652                         CVar.nTouch_Pos_Y1 = 0
653                         CVar.nTouch_XY_X1 = 0
654                         CVar.nTouch_XY_Y1 = 0
655             else :
656                     IsPhone = False
657 # If you enter a coordinate value between 1 and 100, it is converted into a coordinate value suitable for a smartphone
658 def Set(nX, nY):
659             nResX = nX * CVar.nScreenWidth / _RATIO
660             nResY = nY * CVar.nScreenHeight / _RATIO
661             return nResX, nResY
662
663 def IsButton(nX, nY, Btn):
664             if (Btn[0] < 0):
665                     right = (Btn[0] * -200)
666                     left = right - 200
667                     bottom = (Btn [1] * -200)
668                     top = bottom - 200
669                     if ((nY >= top) and (nY <= bottom)):
670                             if ((nX >= left) and (nX <= right)):
671                                     return True
672             else:
673                     if ((nY >= Btn[1]) and (nY <= Btn[3])):
674                             if ((nX >= Btn[0]) and (nX <= Btn[2])):
675                                     return True
676             return False
677
678 # Update all motor positions
679 def PositionUpdate():
680             etc.write8(65,3)
681             while(True) :
682                     if etc.read8(65) == 0 :
683                             break
684
685 #Print number simple test
686 def Show_Num(nX, nY, nValue, nColor = _COLOR_NONE, nSize = None):
687             if (nSize == 0):
688                     Clear_Num(nX, nY)
689             elif (nColor == _COLOR_NONE):
690                     Show(_SHOW_NUM, nX, nY, 60, _COLOR_RED, nValue)
691             elif (nSize == None):
692                     Show(_SHOW_NUM, nX, nY, 60, nColor, nValue)
693             else:
694                     Show(_SHOW_NUM, nX, nY, nSize, nColor, nValue)
695 def Show_Point(nX, nY, nColor, nSize = None):
```

```
696             if (nSize == None):
697                     Show(_SHOW_SHAPE, nX, nY, 20, nColor, 1)
698             else:
699                     Show(_SHOW_SHAPE, nX, nY, nSize, nColor, 1)
700 def Show_Text(nX, nY, nValue, nColor = _COLOR_NONE, nSize = None):
701             if (nColor == _COLOR_NONE):
702                     Show(_SHOW_TEXT, nX, nY, 60, _COLOR_RED, nValue)
703             elif (nSize == None):
704                     Show(_SHOW_TEXT, nX, nY, 60, nColor, nValue)
705             else:
706                     Show(_SHOW_TEXT, nX, nY, nSize, nColor, nValue)
707 def Show_Image(nX, nY, nValue, nSize = None):
708             if (nSize == None):
709                     Show(_SHOW_IMAGE, nX, nY, 1, 0, nValue)
710             else:
```



```
711                     Show(_SHOW_SHAPE, nX, nY, nSize, 0, nValue)
712 def Show_Background(nValue):
713             if (nValue != CVar.nBack_Background):
714                     smart.display.back_image(nValue)
715                     CVar.nBack_Background = nValue
716 #nShowType: 0-picture, 1-letter, 2-shape, 3-number
717 # nValue ([Image-Index], [Shape-1: Circle, 2: Square, 3: Triangle], [Text-Index], [Num-
Value])
718 # nColor (0: Unknown, 1: White, 2: Black, 3: Red, 4: Green, 5: Blue 6: Yellow, 7: Light Gray, 8: Gray,
9: dark gray)
719 def Show(nShowType, nX, nY, nSize, nColor, nValue):
720             if ((nShowType >= 0) and (nShowType < 4)):
721                     nX, nY = Set (nX, nY)
722                     smart.write32(10480, int(nX) | (int(nY) << 16))
723                     nTmp = nValue * 256 + nSize * 65536 + nColor * 16777216
724                     if (nShowType == _SHOW_IMAGE): # Do not use the color value of 0xff000000 digits
725                             smart.display.front_image(nTmp & 16777215) # & 0xffffff(=16777215)
726                     elif (nShowType == _SHOW_SHAPE) :
727                             smart.display.shape(nTmp)
728                     elif (nShowType == _SHOW_TEXT) :
729                             smart.display.text(nTmp)
730                     elif (nShowType == _SHOW_NUM) :
731                             smart.display.number(nTmp)
732 '''
733 def Clear(nShowType, nX, nY):
734             if ((nShowType >= 0) and (nShowType < 4)):
735                     nX, nY = Set (nX, nY)
736                     smart.write32(10480, int(nX) | (int(nY) << 16))
737                     if (nShowType == _SHOW_IMAGE) :
738                             smart.display.front_image(0)
739                     elif (nShowType == _SHOW_SHAPE) :
740                             smart.display.shape(0)
741                     elif (nShowType == _SHOW_TEXT) :
742                             smart.display.text(0)
743                     elif (nShowType == _SHOW_NUM) :
```

```
                smart.display.number(0)
'''
def Clear_All():
        smart.write32(10480, 0)
        smart.display.front_image(0)
        smart.display.shape(0)
        smart.display.text(0)
        smart.display.number(0)
#def Clear_Image():
# smart.write32(10480, 0)
# smart.display.front_image(0)
def Clear_Shape():
        smart.write32(10480, 0)
        smart.display.shape(0)
def Clear_Text():
        smart.write32(10480, 0)
        smart.display.text(0)
def Clear_Num(nX = None, nY = None):
        if (nX == None) or (nY == None) :
                smart.write32(10480, 0)
                smart.display.number(0)
        else :
                Show(_SHOW_NUM, nX, nY, 0, 0, 0)

# [nRed: 0~100], [nBlue: 0~100]
```

```
'''
def LedPwm(nRed, nBlue):
        #print("{0}/{1}".format(round(nRed), round(nBlue)))
        OLLO(2, const.OLLO_PWM_P).write(round(nRed)) # 0~100
        OLLO(2, const.OLLO_PWM_M).write(round(nBlue)) # 0~100

def Led(nValue = 0):
        red_led = OLLO(2, const.OLLO_PWM_P)
        blue_led = OLLO(2, const.OLLO_PWM_M)
        # Seq Cange
        if (nValue == 0) :
                if red_led.read() > 0 and blue_led.read() > 0:
                        red_led.write(0) # 0~100
                        blue_led.write(0) # 0~100
                        print("led pupple -> off")
                elif red_led.read() <= 0 and blue_led.read() <= 0:
                        red_led.write(100) # 0~100
                        blue_led.write(0) # 0~100
                        print("led off -> red")
                elif red_led.read() > 0 and blue_led.read() <= 0:
                        red_led.write(0) # 0~100
                        blue_led.write(100) # 0~100
                        print("led red -> blue")
                else:
                        red_led.write(100) # 0~100
```

```
794                         blue_led.write(100) # 0~100
795                         print("led blue -> puple")
796             # manual change
797             else :
798                     # Red
799                     if nValue == 1:
800                             red_led.write(100) # 0~100
801                             blue_led.write(0) # 0~100
802                             print("led off -> red")
803                     # Blue
804                     elif nValue == 2:
805                             red_led.write(0) # 0~100
806                             blue_led.write(100) # 0~100
807                             print("led red -> blue")
808                     # Purple
809                     elif nValue == 3:
810                             red_led.write(100) # 0~100
811                             blue_led.write(100) # 0~100
812                             print("led blue -> puple")
813                     # Off
814                     else:
815                             red_led.write(0) # 0~100
816                             blue_led.write(0) # 0~100
817                             print("led pupple -> off")
818 '''
819 # Function that converts angle to data used in motor (float -> int)
820 def CalcAngle2Raw (fAngle):
821             return (int) (round (fAngle * 4096.0 / 360.0 + 2048.0))
822 # Function that converts data used in motor into angle value (int -> float)
823 def Get_Position(nID):
824             return DXL(nID).present_position() - aOffset[nID]
825 def CalcRaw2Angle (nRaw):
826             return (float) (360.0 * ((nRaw - 2048.0) / 4096.0))
827 def Rad2Deg(rad):
828             return rad * 180.0 / math.pi
829 def Deg2Rad(Angle):
```



```
830             return Angle * math.pi / 180.0
831 '''
832 def Kine_Angle (nID):
833             # 5, 6, 8 opposite
834             #9, 10, 11 opposite
835             #16 opposite
836             fAngle = CalcRaw2Angle (Get_Position (nID))
837             if ((nID == 5) or (nID == 6) or ((nID> = 8) and (nID <= 11)) or (nID == 16)):
838                     fAngle = fAngle * -1
839             return fAngle
840             #return Deg2Rad (fAngle)
841 '''
842 def Sin (fAngle):
843             return math.sin (Deg2Rad (fAngle))
844 def Cos (fAngle):
```

```
        return math.cos (Deg2Rad (fAngle))
def Offset_GetData(nID):
        nValue = etc.read16 (200 + nID * 2)
        if (nValue > 32767) :
                nValue = (65536 - nValue) * -1
        return int(nValue)
def Offset_Read() :
        global aOffset
        etc.write8(199,1)
        delay(50)
        nMot_First = 5
        nMot_Cnt = 1
        print("Offset=")
        for i in range(nMot_First,nMot_First + nMot_Cnt):
                nID = i # i + 1
                aOffset [nID] = Offset_GetData (nID)
                print (nID, aOffset [nID])
def Offset_Write() :
        etc.write8(199,2)
def Offset_Clear() :
        global aOffset
        etc.write8(199,3)
        #aOffset.clear()
        nCnt = only (aOffset)
        aOffset = [0] * nCnt
        #aOffset = [0 for i in range(nCnt)]
def Test1():
        # 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(1)
        #Test Image : Show_Image(x, y, image index)

        if (CVar.nTouch_Pos0 > 0):
                Clear_Num()
                Clear_Shape()
                # X
                nPos = 50
                nGap = 20

                nX = CVar.nTouch_XY_X0
                nY = CVar.nTouch_XY_Y0
```



```
                if nX < 100 :
                        nX = 100
                elif nX > 900 :
                        nX = 900
                if nY < 100 :
```

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

        Show_Num (nX - nPos - nGap, nY-40, (int) (CVar.nTouch_XY_X0 / 100% 10))
        Show_Num(nX - nPos,                                     nY-40, (int)(CVar.nTouch_XY_X0 / 10 % 10))
        Show_Num (nX - nPos + nGap, nY-40, (int) (CVar.nTouch_XY_X0% 10))
        # Y
        nPos = 50
        Show_Num (nX + nPos - nGap, nY-40, (int) (CVar.nTouch_XY_Y0 / 100% 10))
        Show_Num(nX + nPos,                                     nY-40, (int)(CVar.nTouch_XY_Y0 / 10 % 10))
        Show_Num (nX + nPos + nGap, nY-40, (int) (CVar.nTouch_XY_Y0% 10))

        Show_Num(nX-30, nY+40, CVar.nTouch_Pos_X0, _COLOR_GREEN)
        Show_Num(nX+30, nY+40, CVar.nTouch_Pos_Y0, _COLOR_GREEN)

        Show_Point(CVar.nTouch_XY_X0, CVar.nTouch_XY_Y0, _COLOR_BLUE)
    '''
    if (CVar.nTouch_Pos1 > 0):
        # X
        nPos = 50
        nGap = 20

        nX = CVar.nTouch_XY_X1
        nY = CVar.nTouch_XY_Y1
        if nX < 100 :
                nX = 100
        elif nX > 900 :
                nX = 900
        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)
    '''
#Set the motor's operating speed (Based on Position): 0 initialization
def Setup_Speed(nID, nValue) :
    DXL(nID).write32(112, nValue) # 112 : profile velocity
# Center all motors (Motion_Offset)
def Move_Center(IsOffset = False) :
    #profile speed adjustment
    Setup_Speed(254, 20)
```

```
950             #syncwrite
951             etc.write8(1200,0)
952             etc.write16(1202,116)
953             etc.write8(1204,4)
954             nMot_First = 5
955             nMot_Cnt = 1
956             for i in range(nMot_First,nMot_First + nMot_Cnt):
957                     nID = i # i + 1
958                     nValue = 0
959                     if (IsOffset) :
960                             nValue = Offset_GetData(nID)
961                             #nValue = etc.read16 (200 + nID * 2)
962                             print(nValue)
963                             #if (nValue > 32767) :
964                             # nValue = (65536 - nValue) * -1
965                     etc.write8 (1205, nID)
966
967                     etc.write32(1206,2048 + nValue)
968                     etc.write8(1200,1)
969
970             etc.write8(1200,2)
971             #profile speed restore
972             Setup_Speed(254, 0)
973
974 def WaitButtonUp():
975             while(True) :
976                     #Check the touch input of the smartphone
977                     GetTouch_Down()
978                     # Check button press
979                     nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
980                     if ((nNum0 < 0) and (nNum1 < 0)):
981                             break
982
983 def GetButton(btns):
984             nNum0 = -1
985             nNum1 = -1
986             nDown0 = 0
987             nDown1 = 0
988             nUp0 = 0
989             nUp1 = 0
990             Btn0 = None
991             Btn1 = None
992             nnum = 0
993             nCnt = 0
994             if (CVar.nTouch_Pos0 > 0) :
995                     nCnt = nCnt + 1
996             if (CVar.nTouch_Pos1 > 0) :
997                     nCnt = nCnt + 1
998
999             nPass = 0
```

```
    for btn in btns:
        if (CVar.nTouch_Pos0 > 0):
            if (IsButton(CVar.nTouch_XY_X0, CVar.nTouch_XY_Y0, btn) == True) :
                nNum0 = btn [4]
```



```
                Btn0 = btn
                nPass = nPass | 0x01
        if (CVar.nTouch_Pos1 > 0):
            if (IsButton(CVar.nTouch_XY_X1, CVar.nTouch_XY_Y1, btn) == True) :
                nNum1 = btn [4]
                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
def WaitMotor (nID):
    tmr = CTimer ()
    tmr.Set()
    nPass = 0
    while(True) :
        nMoving = DXL(nID).read8(122)
        if ((tmr.Get() >= 200) and (nPass == 0)) :
            break
        elif (nMoving! = 0):
            nPass = 1
        elif ((nMoving == 0) and (nPass == 1)) :
```



```
            nPass = 2
            break
    tmr = None


def Move_Cam (fAngle, nSpd):
    if fAngle> = 134:
        fAngle = 134
    elif fAngle <= -45:
        fAngle = -45
    Value = (int) (CalcAngle2Raw (fAngle) + aOffset [5])
```

```
    # 448
    DXL(5).write32(112, nSpd)
    DXL(5).write32(116, Value)
    return fAngle


def Set_Rot():
    CVar.fYaw_first = imu.yaw()/100.0
    CVar.fYaw = CVar.fYaw_first
    CVar.fYaw_turn = 0.0


def Get_Rot():
    yaw = imu.yaw () / 100.0
    #CCW - ex) yaw = 170(바뀜), CVar.fYaw = -170 => 340
    if (yaw-CVar.fYaw)> 200: # 180 is fine, but with 200
        CVar.fYaw_turn = CVar.fYaw_turn - 1.0
    #CW - ex) yaw = -170 (바뀜), CVar.fYaw = 170 => -340
    elif (yaw-CVar.fYaw) <-200: # 180 is fine, but with a margin of 200
        CVar.fYaw_turn = CVar.fYaw_turn + 1.0
    CVar.fYaw = yaw
    return yaw + 360 * CVar.fYaw_turn - CVar.fYaw_first
```



```


def Move_Turn(fAngle, nSpd, IsWait = True):
    Set_Rot()
    if fAngle <0:
        nSpd = -nSpd
```

```
    nSpeed = nSpd
    nSlow = nSpd
    if (abs(nSpd) > 25):
        if (fAngle <0):
            nSlow = -25
        else:
            nSlow = 25
    #Set the Inposition value by speed difference.
    fInposition = 1
    fSlowPosition = 10
    if (IsWait):
        while(True):
            Wheel(-nSpeed, nSpeed)
            diff = Get_Rot()
            if (abs(diff) >= abs(fAngle) - fInposition):
                break
            # From this section, the speed decreases.
            if (abs(diff) >= abs(fAngle) - fSlowPosition):
                nSpeed = nSlow
            else:
                nSpeed = nSpd
        Move_Stop()
    else:
        Wheel(-nSpeed, nSpeed)
def Move2 (fAngle = 0.0, spd = 100, dist = 500):
```

```
    Move (Cos (fAngle), Sin (fAngle), spd, dist)
def Move_Stop():
    DXL(1).write32(104, 0)
    DXL(2).write32(104, 0)
def Move(f = 1.0, w = 0.0, spd = 100, dist = 500):
    # Back and forth
    fX=round(f,3)
    # Right and left
    fY=round(w,3)
    # Height (let's just use speed here)
    fZ=spd


    IsForward = True
    if (f < 0):
        IsForward = False
        fX = -fX
        fY = -fY


    dir = 0
    if (fY < 0) :
        dir = -1
    elif (fY> 0):
        dir = 1
    V=fX*fZ
    # Circle diameter
    width=140
    # Distance from outer wheel to rotation point
    R=dist*abs(fY) + width/2
```

```
115
2
115
3           VL=V*(1-dir*width/(2*R))
```



```
115
4           VR=V*(1+dir*width/(2*R))
115
5           V=fZ
115
6           VL2=(fX-fY)*V
115
7           VR2=(fX+fY)*V
115
8
115
9           t1 = VL + VL2
116
0           t2=VR+VR2
116
1           if (IsForward == False):
116
2                   t1 = -t1
116
3                   t2 = -t2
116
4           #print(t1, t2)
116
5           Wheel(round(t1), round(t2))
116
6 def Wheel(nSpd_L, nSpd_R):
116
7           nGap_L = 0 # 10
116
8           nGap_R = 0 # 10
116
9           if (nSpd_L < 0):
117
0                   nGap_L = -10
117
1           if (nSpd_R < 0):
117
2                   nGap_R = -10
117
3           etc.write8(1200,0)
117
4           etc.write16(1202,108)
117
5           etc.write8(1204,4)
117
6           #
117
```

```
117        nID = 1
117
118        etc.write8 (1205, nID)
117
118        etc.write32(1206,0)
118
118        etc.write8(1200,1)
118
118        #
118
118        nID = 2
118
118        etc.write8 (1205, nID)
```



```
118
118        etc.write32(1206,0)
118
118        etc.write8(1200,1)
118
118        #
118
118        etc.write8(1200,2)
118
118        ####################
118
119        etc.write8(1200,0)
119
119        etc.write16(1202,104)
119
119        etc.write8(1204,4)
119
119        #
119
119        nID = 1
119
119        etc.write8 (1205, nID)
119
119        etc.write32(1206,-nSpd_R + nGap_R)
119
119        etc.write8(1200,1)
119
119        #
119
119        nID = 2
119
119        etc.write8 (1205, nID)
120
120        etc.write32 (1206, nSpd_L + nGap_L)
120
120        etc.write8(1200,1)
120
```

```
    #
120
    etc.write8(1200,2)
120
def Page_MenuBar(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
120
    if (Btn0 == _BTN_SUB_OUT):
120
        ShowPage(CVar.nPage_Prev)
120
    #When pressing the outside part while the menu is displayed
120
    elif (Btn0 == _BTN_SUB_OUT):
120
        ShowPage(CVar.nPage_Prev)
121
    elif (Btn0 == _BTN_SUB_REMOTE):
121
        ShowPage(_PAGE_MAIN)
121
    elif (Btn0 == _BTN_SUB_DEMO):
121
        ShowPage(_PAGE_MARKER)
```



```
121
    elif (Btn0 == _BTN_SUB_STREAM):
121
        ShowPage(_PAGE_STREAM)
121
    elif (Btn0 == _BTN_SUB_MOTOR):
121
        ShowPage(_PAGE_MOTOR_TEST)
121
    elif (Btn0 == _BTN_SUB_OFFSET):
121
        ShowPage(_PAGE_OFFSET)
122
        #print(aOffset)
122
    elif (Btn0 == _BTN_SUB_X):
122
        ShowPage(CVar.nPage_Prev)
122
122
    #Check the touch input of the smartphone
122
    if ((nNum0 >= 0) or (nNum1 >= 0)):
122
        WaitButtonUp()
122
```

```
def Page_Main(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1):
        nSpeed = CVar.nPoint_Joystick
        #
        nSpd = round(250 * nSpeed / 5 / 2)
        nSpd_Limit = round(250 * 3 / 5 / 2)
        IsMoving = False
        nX = 1
        nY = 1
        if (Btn0 == _BTN_TURN_L) or (Btn1 == _BTN_TURN_L):
                Wheel(-nSpd, nSpd)
                IsMoving = True
        if (Btn0 == _BTN_TURN_R) or (Btn1 == _BTN_TURN_R):
                Wheel(nSpd, -nSpd)
                IsMoving = True
        if (Btn0 == _BTN_MOVE_UL) or (Btn1 == _BTN_MOVE_UL):
                if (nSpeed > 3) :
                        #nX = 0.5
```



```
                        nSpd = nSpd_Limit
                Move(nX,nY,nSpd, 300)
                #Move2(45, 100, 300)
                IsMoving = True
        if (Btn0 == _BTN_MOVE_U) or (Btn1 == _BTN_MOVE_U):
                Move(nX,0,nSpd, 0)
                IsMoving = True
        if (Btn0 == _BTN_MOVE_UR) or (Btn1 == _BTN_MOVE_UR):
```

```
                if (nSpeed > 3) :
                    #nX = 0.5
                    nSpd = nSpd_Limit
                Move(nX,-nY,nSpd, 300)
                #Move2(-45, 100, 300)
                IsMoving = True
        if (Btn0 == _BTN_MOVE_DL) or (Btn1 == _BTN_MOVE_DL):
                if (nSpeed > 3) :
                    nSpd = nSpd_Limit
                Move(-nX,-nY,nSpd, 300)
                IsMoving = True
        if (Btn0 == _BTN_MOVE_D) or (Btn1 == _BTN_MOVE_D):
                Move(-nX,0,nSpd, 300)
                IsMoving = True
        if (Btn0 == _BTN_MOVE_DR) or (Btn1 == _BTN_MOVE_DR):
                if (nSpeed > 3) :
                    nSpd = nSpd_Limit
                Move(-nX,nY,nSpd, 300)
                IsMoving = True
        if (IsMoving == False):
                Move_Stop()
        #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.
```



```
        if (Btn0 == _BTN_TORQ):# or (Btn1 == _BTN_TORQ):
                if Event_Dn0 == 1:
                    tmrTorqBtn.Set()
```

```
127                     if (CVar.IsTorqOn == True):
127
128                         TorqAll(False, True)
128
129                     else :
129
130                         TorqAll(True, True)
130
131             elif Event_Up0 == 1:
131
132                     tmrTorqBtn.Destroy()
132
133             ##If you keep holding down
133
134             else:
134
135                     #Reboot All
135
136                     if (tmrTorqBtn.Get() >= 3000):
136
137                         #Command - Reboot
137
138                         dxlbus.reboot()
138
139                         #Reboot - Beep
139
140                         buzzer.melody(1)
140
141                         #TorqOff variable
141
142                         CVar.IsTorqOn = False
142
143                         # No more timer detection.
143
144                         tmrTorqBtn.Destroy()
144
145     #Normal mode action button
145
146     '''
146
147     if (Btn0 == _BTN_ACT_1):
147
148             if (Event_Dn0 == 1):
148
149                     Move_Turn(90, nSpd)
149
150     elif (Btn1 == _BTN_ACT_1):
150
151             if (Event_Dn1 == 1):
151
152                     Move_Turn(90, nSpd)
152
153     if (Btn0 == _BTN_ACT_2):
```

```
130
    4                 if (Event_Dn0 == 1):
130
    5                         Move_Turn(-90, nSpd)
130
    6         elif (Btn1 == _BTN_ACT_2):
130
    7                 if (Event_Dn1 == 1):
130
    8                         Move_Turn(-90, nSpd)
130
    9         #
131
    0         if (Btn0 == _BTN_ACT_3):
131
    1                 if (Event_Dn0 == 1):
131
    2                         Move_Turn(180, nSpd)
131
    3         elif (Btn1 == _BTN_ACT_3):
131
    4                 if (Event_Dn1 == 1):
131
    5                         Move_Turn(180, nSpd)
131
    6         if (Btn0 == _BTN_ACT_4):
131
    7                 if (Event_Dn0 == 1):
131
    8                         Move_Turn(-180, nSpd)
131
    9         elif (Btn1 == _BTN_ACT_4):
132
    0                 if (Event_Dn1 == 1):
132
    1                         Move_Turn(-180, nSpd)
132
    2         #
132
    3         if (Btn0 == _BTN_ACT_5):
132
    4                 if (Event_Dn0 == 1):
132
    5                         Move_Turn(360, nSpd)
132
    6         elif (Btn1 == _BTN_ACT_5):
132
    7                 if (Event_Dn1 == 1):
132
    8                         Move_Turn(360, nSpd)
132
    9         if (Btn0 == _BTN_ACT_6):
```

```
                if (Event_Dn0 == 1):
                        Move_Turn(-360, nSpd)
        elif (Btn1 == _BTN_ACT_6):
                if (Event_Dn1 == 1):
```



```
                        Move_Turn(-360, nSpd)
        '''
        #Motion speed button
        if (Event_Dn0 == 1):
                if (Btn0 == _BTN_SPD_L1):
                        nSpeed = 1
                elif (Btn0 == _BTN_SPD_L2):
                        nSpeed = 2
                elif (Btn0 == _BTN_SPD_L3):
                        nSpeed = 3
                elif (Btn0 == _BTN_SPD_L4):
                        nSpeed = 4
                elif (Btn0 == _BTN_SPD_L5):
                        nSpeed = 5
        elif (Event_Dn1 == 1):
                if (Btn1 == _BTN_SPD_L1):
                        nSpeed = 1
                elif (Btn1 == _BTN_SPD_L2):
                        nSpeed = 2
                elif (Btn1 == _BTN_SPD_L3):
                        nSpeed = 3
```

```
135                 elif (Btn1 == _BTN_SPD_L4):
135
6                       nSpeed = 4
135
7                   elif (Btn1 == _BTN_SPD_L5):
135
8                       nSpeed = 5
135
9
136
0           nSpeed2 = 3#CVar.nPoint_Cam
136
1
136
2           # Camera control
136
3           fInposition = 0.2
```



```
136
4           if (Btn0 == _BTN_SPD_R1) or (Btn1 == _BTN_SPD_R1):
136
5                   Move_Cam(CalcRaw2Angle(Get_Position(5)) - 2, 0)
136
6                   nSpeed2 = 1
136
7           elif (Btn0 == _BTN_SPD_R2) or (Btn1 == _BTN_SPD_R2):
136
8                   Move_Cam(CalcRaw2Angle(Get_Position(5)) - 1, 10)
136
9                   nSpeed2 = 2
137
0           elif (Btn0 == _BTN_SPD_R3):
137
1                   nSpeed2 = 3
137
2           elif (Btn0 == _BTN_SPD_R4) or (Btn1 == _BTN_SPD_R4):
137
3                   Move_Cam(CalcRaw2Angle(Get_Position(5)) + 1, 10)
137
4                   nSpeed2 = 4
137
5           elif (Btn0 == _BTN_SPD_R5) or (Btn1 == _BTN_SPD_R5):
137
6                   Move_Cam(CalcRaw2Angle(Get_Position(5)) + 2, 0)
137
7                   nSpeed2 = 5
137
8           if (((Event_Up0 == 1) and (Btn0 == _BTN_SPD_R3)) or ((Event_Up1 == 1) and (Btn1
== _BTN_SPD_R3))):
137
```

```
1389                    Move_Cam_Center()
1390                    nSpeed2 = 3
1391
1392
1393
1394            if (nSpeed != CVar.nPoint_Joystick) or (nSpeed2 != CVar.nPoint_Cam):
1395                    Clear_Shape()
1396                    CVar.nPoint_Joystick = nSpeed
1397                    CVar.nPoint_Cam = nSpeed2
1398                    Show_Point(480, (458 + (110 * (5-CVar.nPoint_Joystick))), _COLOR_BLACK)
1399                    Show_Point((885), (458 + (110 * (5-CVar.nPoint_Cam))), _COLOR_BLACK)
1400
1401 def Page_Stream(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
1402            nSpeed = CVar.nPoint_Joystick
1403            #
```

Page 31

```
1404            nSpd = round(250 * nSpeed / 5 / 2)
1405            nSpd_Limit = round(250 * 3 / 5 / 2)
1406            IsMoving = False
1407            nX = 1
1408            nY = 1
1409            if (Btn0 == _BTN_TURN_L) or (Btn1 == _BTN_TURN_L):
1410                    Wheel(-nSpd, nSpd)
1411                    IsMoving = True
1412            if (Btn0 == _BTN_TURN_R) or (Btn1 == _BTN_TURN_R):
1413                    Wheel(nSpd, -nSpd)
1414                    IsMoving = True
```

```
1404        if (Btn0 == _BTN_MOVE_UL) or (Btn1 == _BTN_MOVE_UL):
1405            if (nSpeed > 3) :
1406                #nX = 0.5
1407                nSpd = nSpd_Limit
1408            Move(nX,nY,nSpd, 300)
1409            #Move2(45, 100, 300)
1410            IsMoving = True
1411        if (Btn0 == _BTN_MOVE_U) or (Btn1 == _BTN_MOVE_U):
1412            Move(nX,0,nSpd, 300)
1413            IsMoving = True
1414        if (Btn0 == _BTN_MOVE_UR) or (Btn1 == _BTN_MOVE_UR):
1415            if (nSpeed > 3) :
1416                #nX = 0.5
1417                nSpd = nSpd_Limit
1418            Move(nX,-nY,nSpd, 300)
1419            #Move2(-45, 100, 300)
1420            IsMoving = True
1421        if (Btn0 == _BTN_MOVE_DL) or (Btn1 == _BTN_MOVE_DL):
1422            if (nSpeed > 3) :
```



```
1423                nSpd = nSpd_Limit
1424            Move(-nX,-nY,nSpd, 300)
1425            IsMoving = True
1426        if (Btn0 == _BTN_MOVE_D) or (Btn1 == _BTN_MOVE_D):
1427            Move(-nX,0,nSpd, 300)
1428            IsMoving = True
```

```
        if (Btn0 == _BTN_MOVE_DR) or (Btn1 == _BTN_MOVE_DR):
            if (nSpeed > 3) :
                nSpd = nSpd_Limit
            Move(-nX,nY,nSpd, 300)
            IsMoving = True
        if (IsMoving == False):
            Move_Stop()
        #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.
        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)
            elif Event_Up0 == 1:
                tmrTorqBtn.Destroy()
            ##If you keep holding down
            else:
                #Reboot All
                if (tmrTorqBtn.Get() >= 3000):
                    #Command - Reboot
                    dxlbus.reboot()
```



```
                    #Reboot - Beep
```

```
145                                 buzzer.melody(1)
145
4                                   #TorqOff variable
145
5                                   CVar.IsTorqOn = False
145
6                                   # No more timer detection.
145
7                                   tmrTorqBtn.Destroy()
145
8           #Normal mode action button
145
9           '''
146
0           if (Btn0 == _BTN_ACT_1):
146
1               if (Event_Dn0 == 1):
146
2                   Move_Turn(90, nSpd)
146
3           elif (Btn1 == _BTN_ACT_1):
146
4               if (Event_Dn1 == 1):
146
5                   Move_Turn(90, nSpd)
146
6           if (Btn0 == _BTN_ACT_2):
146
7               if (Event_Dn0 == 1):
146
8                   Move_Turn(-90, nSpd)
146
9           elif (Btn1 == _BTN_ACT_2):
147
0               if (Event_Dn1 == 1):
147
1                   Move_Turn(-90, nSpd)
147
2           #
147
3           if (Btn0 == _BTN_ACT_3):
147
4               if (Event_Dn0 == 1):
147
5                   Move_Turn(180, nSpd)
147
6           elif (Btn1 == _BTN_ACT_3):
147
7               if (Event_Dn1 == 1):
147
8                   Move_Turn(180, nSpd)
147
9           if (Btn0 == _BTN_ACT_4):
148
0               if (Event_Dn0 == 1):
```

```
148
 1                      Move_Turn(-180, nSpd)
```



```
148
 2          elif (Btn1 == _BTN_ACT_4):
148
 3                  if (Event_Dn1 == 1):
148
 4                      Move_Turn(-180, nSpd)
148
 5          #
148
 6          if (Btn0 == _BTN_ACT_5):
148
 7                  if (Event_Dn0 == 1):
148
 8                      Move_Turn(360, nSpd)
148
 9          elif (Btn1 == _BTN_ACT_5):
149
 0                  if (Event_Dn1 == 1):
149
 1                      Move_Turn(360, nSpd)
149
 2          if (Btn0 == _BTN_ACT_6):
149
 3                  if (Event_Dn0 == 1):
149
 4                      Move_Turn(-360, nSpd)
149
 5          elif (Btn1 == _BTN_ACT_6):
149
 6                  if (Event_Dn1 == 1):
149
 7                      Move_Turn(-360, nSpd)
149
 8          '''
149
 9          #Motion speed button
150
 0          if (Event_Dn0 == 1):
150
 1                  if (Btn0 == _BTN_SPD_L1):
150
 2                      nSpeed = 1
150
 3                  elif (Btn0 == _BTN_SPD_L2):
150
 4                      nSpeed = 2
150
 5                  elif (Btn0 == _BTN_SPD_L3):
```

```
150
    6                       nSpeed = 3
150
    7               elif (Btn0 == _BTN_SPD_L4):
150
    8                       nSpeed = 4
150
    9               elif (Btn0 == _BTN_SPD_L5):
151
    0                       nSpeed = 5
151
    1       elif (Event_Dn1 == 1):
```



```
151
    2               if (Btn1 == _BTN_SPD_L1):
151
    3                       nSpeed = 1
151
    4               elif (Btn1 == _BTN_SPD_L2):
151
    5                       nSpeed = 2
151
    6               elif (Btn1 == _BTN_SPD_L3):
151
    7                       nSpeed = 3
151
    8               elif (Btn1 == _BTN_SPD_L4):
151
    9                       nSpeed = 4
152
    0               elif (Btn1 == _BTN_SPD_L5):
152
    1                       nSpeed = 5
152
    2
152
    3       nSpeed2 = 3
152
    4
152
    5       # Camera control
152
    6       fInposition = 0.2
152
    7       if (Btn0 == _BTN_SPD_R1) or (Btn1 == _BTN_SPD_R1):
152
    8               Move_Cam(CalcRaw2Angle(Get_Position(5)) - 2, 0)
152
    9               nSpeed2 = 1
153
    0       elif (Btn0 == _BTN_SPD_R2) or (Btn1 == _BTN_SPD_R2):
```

```
153
    1                   Move_Cam(CalcRaw2Angle(Get_Position(5)) - 1, 10)
153
    2                   nSpeed2 = 2
153
    3           elif (Btn0 == _BTN_SPD_R3):
153
    4                   nSpeed2 = 3
153
    5           elif (Btn0 == _BTN_SPD_R4) or (Btn1 == _BTN_SPD_R4):
153
    6                   Move_Cam(CalcRaw2Angle(Get_Position(5)) + 1, 10)
153
    7                   nSpeed2 = 4
153
    8           elif (Btn0 == _BTN_SPD_R5) or (Btn1 == _BTN_SPD_R5):
153
    9                   Move_Cam(CalcRaw2Angle(Get_Position(5)) + 2, 0)
154
    0                   nSpeed2 = 5
154
    1           if (((Event_Up0 == 1) and (Btn0 == _BTN_SPD_R3)) or ((Event_Up1 == 1) and (Btn1
== _BTN_SPD_R3))):
```

```
154
    2                   Move_Cam_Center()
154
    3                   nSpeed2 = 3
154
    4
154
    5
154
    6           if (nSpeed != CVar.nPoint_Joystick) or (nSpeed2 != CVar.nPoint_Cam):
154
    7                   Clear_Shape()
154
    8                   CVar.nPoint_Joystick = nSpeed
154
    9                   CVar.nPoint_Cam = nSpeed2
155
    0                   Show_Point(480, (458 + (110 * (5-CVar.nPoint_Joystick))), _COLOR_BLACK)
155
    1                   Show_Point((885), (458 + (110 * (5-CVar.nPoint_Cam))), _COLOR_BLACK)
155
    2 def Show_Text_Marker():
155
    3           Show_Text(500, 450, 32, _COLOR_GREEN, 60)
155
    4           Show_Text(500, 550, 33, _COLOR_GREEN, 60)
155
    5 def Page_Marker(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
```

```
Btn1):
155
    nMode = rpi.mode()
155
    if nMode != 4 :#and CVar.IsModeChanging != False:
155
        Show_Text(500,500,31,_COLOR_WHITE,100)
155
        CVar.IsModeChanging = True
156
    else: #if nMode == 4:
156
        if CVar.IsModeChanging == True:
156
            CVar.IsModeChanging = False
156
            Clear_Text()
156
            Show_Text_Marker()
156
156
        area = rpi.area()
156
        positionX = rpi.position_x()
156
        positionY = rpi.position_y()
156
        marker = rpi.marker()
157
157
        # Command card & wheel movement
```



```
157
        if marker == 1:#Run
157
            buzzer.melody(2)
157
            delay(4000)
157
            RunMarkerList ()
157
            Show_Text_Marker()
157
        elif marker >= 2 and marker <= 6 : #Forward(2), Backward(3), Turn Left(4),
Turn Right(5), Delay 1s(6)
157
            AddMarkerList (marker)
157
158
```

```
                if (Event_Dn0 == 1):
158
                    if (Btn0 == _BTN_MODE_LINE):
158
                        ShowPage(_PAGE_LINE)
158
                    elif (Btn0 == _BTN_MODE_MARKER):
158
                        ShowPage(_PAGE_MARKER)
158
                    elif (Btn0 == _BTN_MODE_DEBUG):
158
                        Show_Background(11)
158
                        CVar.IsNonstop = False
158
                    elif (Btn0 == _BTN_MODE_RUN):
158
                        Show_Background(12)
159
                        CVar.IsNonstop = True
159
def Filter_LowPass(fWeight, fVal_Curr, fVal_Prev) :
159
    return int(round(float(fVal_Curr) * fWeight + (1.0 - fWeight) *
float(fVal_Prev), 0))
159
3
159
line_prev = 0
159
line2_prev = 0
159
#line3_prev = 0
159
7
159
def Page_Line(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
159
    global Line_Back
160
0
    global line_prev
```



```
160
1
    global line2_prev
160
2
    #global line3_prev
160
3
    nMode = rpi.mode()
160
4
    if nMode != 5 : #and CVar.IsModeChanging != False:
```

```
160
    5                   Show_Text(500,500,31,_COLOR_WHITE,100)
160
    6                   CVar.IsModeChanging = True
160
    7           else : #if nMode == 5:
160
    8                   if CVar.IsModeChanging == True:
160
    9                           CVar.IsModeChanging = False
161
    0                           Clear_Text()
161
    1
161
    2                   IsShow_Text = False
161
    3                   #Commando search speed
161
    4                   Speed = 100#30 #40 #20
161
    5                   line = DXL(201).read8(68)
161
    6                   line2 = DXL(201).read8(69)
161
    7                   If it exceeds the value of #127, it is received as a-value, so it is substituted with a positive number.
161
    8                   if (line < 0):
161
    9                           line = 256 + line
162
    0                   if (line2 < 0):
162
    1                           line2 = 256 + line2
162
    2
162
    3                   #if line == 0 :
162
    4
162
    5                   '''
162
    6                   line = 0#DXL(201).read8(68)
162
    7                   line2 = 0#DXL(201).read8(69)
162
    8                   #line3 = DXL(201).read8(70)
162
    9                   nCnt_Line = 0
163
    0                   nRange1 = 78 # 72 # 72
```

```
163
    1                   nRange2 = 78 # 75
163
    2                   for i in range(68,nRange1):
163
    3                           tmp = DXL(201).read8(i)
163
    4                           #if (i != 68):
163
    5                           # tmp = Filter_LowPass(0.5, tmp, tmp2)
163
    6                           If it exceeds the value of #127, it is received as a-value, so it is substituted with a positive number.
163
    7                           if (tmp < 0):
163
    8                                   tmp = 256 + tmp
163
    9                           line += tmp
164
    0                           if tmp != 0:
164
    1                                   nCnt_Line += 1
164
    2                           #tmp2 = tmp
164
    3                   if nCnt_Line > 0:
164
    4                           line /= nCnt_Line
164
    5                           line -= 90
164
    6                           # Delete if schedule search has not been done
164
    7                           #if (nCnt_Line < ((nRange1 - 64) / 2)):
164
    8                           # line = 0
164
    9                   '''
165
    0                   '''
165
    1                   nCnt_Line = 0
165
    2                   for i in range(72,nRange2):
165
    3                           tmp = DXL(201).read8(i)
165
    4                           #if (i != 75):
165
    5                           # tmp = Filter_LowPass(0.3, tmp, tmp2)
165
    6                           If it exceeds the value of #127, it is received as a-value, so it is substituted with a positive number.
165
```

```
                if (tmp < 0):
                    tmp = 256 + tmp

                line2 += tmp

                if tmp != 0:
```



```
                    nCnt_Line += 1

                #tmp2 = tmp
            if nCnt_Line > 0:
                line2 /= nCnt_Line

                line2 -= 90
            '''
            #The closer to 0 degrees, the right, 90 degrees forward, and 0 is not found.
            nFind = 0
            if line != 0 :
                nFind = 3
                #0 degree changed to forward
                line -= 90
                if (line2 != 0):
                    line2 = (line + (line2-90))/2
                line2 -= 90
            elif line2! = 0:
                #nFind = 3
                line2 -= 90
                #line = line2
            if IsShow_Text:
                Clear_Num()
```

```
1682                    Clear_Shape()
1683
1684                    if (line >= 0):
1685                            Show_Num(500, 800, (int)(line), _COLOR_WHITE)
1686                    else:
1687                            Show_Num(500, 800, (int)(-line), _COLOR_RED)
1688                    if (line2 >= 0):
1689                            Show_Num(500, 850, (int)(line2), _COLOR_YELLOW)
1690                    else:
```



```
1691                            Show_Num(500, 850, (int)(-line2), _COLOR_RED)
1692                    Show_Point(500, 900, _COLOR_BLUE)
1693
1694            if (nFind > 0):
1695                    if (Line_Back != 0):
1696                            line = Filter_LowPass(0.5, line, Line_Back)
1697                    tmp = line
1698
1699
1700                    if IsShow_Text:
1701                            Color = _COLOR_RED
1702                            if line == 0 :
1703                                    #tmp = line2
1704                                    Color = _COLOR_YELLOW
1705                            nX = 400 * Cos(tmp+90)
1706                            nY = 400 * Sin(tmp+90)
1707                            Show_Point(500 + nX, 900 - nY, Color)
```

```
1708
1709                    #if tmp != 0:
1710
1711                    # Move2(line/4, Speed, 600)
1712
1713                    #if (line2_prev != 0):
1714
1715                    # line = (line2_prev - line) / 4 + line - 90
1716
1717                    #else :
1718
1719                    # line = line-90
1720
1721
1722                    '''
1723                    if tmp != 0:
1724                            val = abs(tmp)
1725                            if val < 5 :
1726                                    Move2(0, Speed, 1500)
```



```
1727                            elif val <30:
1728                                    Move2(line/6, Speed, 1500)
1729                            elif val <50:
1730                                    Move2(line/3, Speed, 1500)
1731                            else :
1732                                    Move2(line, Speed, 1500)
1733                    '''
1734
1735                    if tmp != 0:
1736                            val = abs(tmp)
1737                            if val < 5 :
1738                                    Move2(0, Speed, 1500)
```

```
                else:
                    Move2(line, Speed, 1500)


            #Move2(line/2, Speed, 1500)
            Line_Back = line
        else:

            # Turn in the direction you were turning.
            if Line_Back >= 0:
                Move_Turn(50, Speed, False)
            else :
                Move_Turn(-50, Speed, False)

        if line_prev != 0 and line != 0:
            line_prev = line
        if line2_prev != 0 and line2 != 0 :
            line2_prev = line2


        tmp = 0
        if CVar.nCamera_mode_sub == 1: # Red
            #Color = 1
            tmp = 5
        elif CVar.nCamera_mode_sub == 4: # Green
```

```
1757                    #Color = 4
1758                    tmp = 6
1759
1760            elif CVar.nCamera_mode_sub == 5: # Blue
1761
1762                    #Color = 5
1763
1764                    tmp = 7
1765
1766            if tmp != 0:
1767
1768                    Show_Background(tmp + nFind)
1769
1770
1771
1772            if (Event_Dn0 == 1):
1773
1774                    #IsColorChange = False
1775
1776                    if (Btn0 == _BTN_MODE_LINE):
1777
1778                            ShowPage(_PAGE_LINE)
1779
1780                    elif (Btn0 == _BTN_MODE_MARKER):
1781
1782                            ShowPage(_PAGE_MARKER)
1783
1784                    else:
1785
1786                            Color = 0
1787
1788                            if Btn0 == _BTN_COLOR_R:
1789
1790                                    Color = 1
1791
1792                            elif Btn0 == _BTN_COLOR_G:
1793
1794                                    Color = 4
1795
1796                            elif Btn0 == _BTN_COLOR_B:
1797
1798                                    Color = 5
1799
1800                            #print(Color)
1801
1802                            if (Color):
```



```
1803                                    if CVar.nCamera_mode_sub != Color:
1804
```

```
                                        CVar.nCamera_mode_sub = Color
                                        rpi.sub_mode(CVar.nCamera_mode_sub)

                                        Show_Background(tmp+Btn0[4]-_BTN_COLOR_R[4]) # 5,6,7

                                        Line_Back = 0

                        #elif (Btn0 == _BTN_COLOR_R):

                        # if CVar.nCamera_mode_sub != 1:

                        #               CVar.nCamera_mode_sub = 1

                        #               IsColorChange = True

                        #elif (Btn0 == _BTN_COLOR_G):

                        # if CVar.nCamera_mode_sub != 4:

                        #               CVar.nCamera_mode_sub = 4

                        #               IsColorChange = True

                        #elif (Btn0 == _BTN_COLOR_B):

                        # if CVar.nCamera_mode_sub != 5:

                        #               CVar.nCamera_mode_sub = 5

                        #               IsColorChange = True

                        #if (IsColorChange):

                        # rpi.sub_mode(CVar.nCamera_mode_sub)

                        # Show_Background(5+Btn0[4]-_BTN_COLOR_R[4]) # 5,6,7

                        # Line_Back = 0

def Page_MotorTest(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
        if (Event_Dn0 == 1):
                nID = -1
                if (Btn0 == _BTN_ID_1):
                        nID = 1
                elif (Btn0 == _BTN_ID_2):
                        nID = 2
                elif (Btn0 == _BTN_ID_3):
```

```
1810                    nID = 3
```



```
181
1               elif (Btn0 == _BTN_ID_4):
181
2                   nID = 4
181
3               elif (Btn0 == _BTN_ID_5):
181
4                   nID = 5
181
5               if nID >= 0:
181
6                   Show_Motors(nID)
181
7                   #All LEDs Off
181
8                   DXL(254).write8(65, 0)
181
9                   #LED on selected motor
182
0                   DXL(nID).write8(65, 1)
182
1                   # Turn off the LEDs of other actuators that are synced.
182
2                   if (nID == 1):
182
3                       DXL(3).write8(65, 0)
182
4                   # Turn off the LEDs of other actuators that are synced.
182
5                   if (nID == 2):
182
6                       DXL(4).write8(65, 0)
182
7
182
8                   if nID > 4:
182
9                       #DXL(nID).write32(112, 30)
183
0                       #DXL(nID).write32(116, 2048)
183
1                       Move_Cam(45, 30)
183
2                       WaitMotor (nID)
183
3
183
4                       Move_Cam(55, 30)
183
```

```
                #DXL(nID).write32(116, 2098)
                WaitMotor (nID)

                Move_Cam(35, 30)
                #DXL(nID).write32(116, 1998)
                WaitMotor (nID)
```



```

                Move_Cam(45, 30)
                #DXL(nID).write32(116, 2048)
                WaitMotor (nID)


def Page_Offset(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0,
Btn1):
        if (Event_Dn0 == 1):
                nID = -1
                if (Btn0 == _BTN_ID_5):
                        nID = 5


                if (nID >= 0):
                        CVar.nID = nID
                        CVar.nOffset = Offset_GetData(nID)
                        ShowPage(_PAGE_OFFSET_DIALOG1)
                else:
                        CVar.nID = -1
                        if (Btn0 == _BTN_RESET):
                                ShowPage(_PAGE_OFFSET_DIALOG2)
```

```
1860                                    #CVar.nOffset = 0
1861                                    CVar.IsResetCommand = True
1862                            elif (Btn0 == _BTN_INIT):
1863                                    TorqAll(True,False)
1864                                    Move_Center(True)
1865 def Page_Offset_Dialog_1(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1,
Btn0, Btn1):
1866
1867             #+
1868             if (Btn0 == _BTN_DIALOG_PLUS):
1869                     if (DXL(CVar.nID).read8(64) == 0):
```



```
1870                             #Torq Off -> On
1871                             TorqOnOff(CVar.nID, True, False)
1872                             Show_Image(600, 400, 53)
1873                     CVar.nOffset = CVar.nOffset + 1
1874                     etc.write16(200 + CVar.nID * 2, (int)(CVar.nOffset))
1875                     #Setup_Speed(CVar.nID, 0)
1876                     DXL(CVar.nID).goal_position(2048 + (int)(CVar.nOffset))
1877                     #print(CVar.nOffset)
1878             #-
1879             if (Btn0 == _BTN_DIALOG_MINUS):
1880                     if (DXL(CVar.nID).read8(64) == 0):
1881                             #Torq Off -> On
1882                             TorqOnOff(CVar.nID, True, False)
1883                             Show_Image(600, 400, 53)
1884
```

```
                CVar.nOffset = CVar.nOffset - 1
                etc.write16(200 + CVar.nID * 2, (int)(CVar.nOffset))
                DXL(CVar.nID).goal_position(2048 + (int)(CVar.nOffset))
                #print(CVar.nOffset)


        if (CVar.nID > 0):
                Show_Num_Offset (CVar.nID)


        if (Event_Dn0 == 1):
                if (Btn0 == _BTN_DIALOG_OK):


                        if (DXL(CVar.nID).read8(64) == 0):
                                CVar.nOffset = DXL(CVar.nID).goal_position() - 2048


                        # Put Offset data
                        etc.write16(200 + CVar.nID * 2, (int)(CVar.nOffset))
```



```


                        # Save
                        ShowPage(_PAGE_OFFSET_DIALOG2)
                        TorqAll(True, False)


                if (Btn0 == _BTN_DIALOG_CANCEL):
                        ShowPage(_PAGE_OFFSET)
                        #ShowPage(_PAGE_OFFSET_DIALOG_CLOSE)
                if (Btn0 == _BTN_DIALOG_TORQ):
```

```
                    if (CVar.nID > 0) :
                        if (DXL(CVar.nID).read8(64) == 0):
                            #Torq Off -> On
                            TorqOnOff (CVar.nID, True)
                            Show_Image(600, 400, 53)

                            CVar.nOffset = DXL(CVar.nID).goal_position() - 2048
                            etc.write16(200 + CVar.nID * 2, (int)(CVar.nOffset))
                        else :
                            #Torq On -> Off
                            TorqOnOff (CVar.nID, False)
                            Show_Image(600, 400, 54)

                            etc.write16(200 + CVar.nID * 2, (int)(CVar.nOffset))
def Page_Offset_Dialog_2(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1,
Btn0, Btn1):
    global aOffset
    if (Event_Dn0 == 1):
        if (Btn0 == _BTN_DIALOG_OK):
            if (CVar.IsResetCommand == True):
                Offset_Clear()
            else :
```



```
                Offset_Write()
                aOffset[CVar.nID] = CVar.nOffset
            # Done - Close the dialog
            ShowPage(_PAGE_OFFSET)
```

```
193 4                       Move_Center(True)
193 5               if (Btn0 == _BTN_DIALOG_CANCEL):
193 6                       ShowPage(_PAGE_OFFSET)
193 7                       #ShowPage(_PAGE_OFFSET_DIALOG_CLOSE)
193 8
193 9 def Show_Dialog(nStep, nID=5) :
194 0       # Show Offset Dialog
194 1       if (nStep == 1) :
194 2               CVar.IsResetCommand = False
194 3               # Show dialog background
194 4               Show_Image(500, 500, 51)
194 5               #Torq On/Off Image
194 6               Show_Image(600, 400, 53)
194 7               #DXL ID display
194 8               Show_Num(475, 210, nID, _COLOR_WHITE, 80)
194 9       #Show Save Dialog
195 0       elif (nStep == 2):
195 1               #Clear all numbers
195 2               Clear_Num()
195 3               # Show dialog background
195 4               Show_Image(500, 500, 52)
195 5               #DXL ID display
195 6               Show_Num(475, 210, nID, _COLOR_WHITE, 80)
195 7               #SAVE text output
195 8               Show_Text(500,500,11,_COLOR_BLACK,100)
195 9       else :
```

```
196
    0                   CVar.IsResetCommand = False
196
    1                   #Clear-dialog background
196
    2                   Show_Image(500, 500, 0)
196
    3                   #Clear - Torq On/Off Image
196
    4                   Show_Image(600, 400, 0)
196
    5                   #Clear - Show Text "Save"
196
    6                   Show_Text(500,500,0)
196
    7                   #Clear all numbers
196
    8                   Clear_Num()
196
    9                   # If you close the window, do Torq On -> Move Center.
197
    0                   TorqAll(True, False)
197
    1                   if (CVar.nPage == _PAGE_OFFSET):
197
    2                           Move_Center(True)
197
    3                   else :
197
    4                           Move_Center()
197
    5 def Move_Cam_Center():
197
    6           fAngle_Cam = 45.0
197
    7           Move_Cam(fAngle_Cam, 30)
197
    8 def Show_Num_Offset(nID) :
197
    9           nOffset = DXL(nID).goal_position() - 2048
198
    0           nColor = 5
198
    1           nGap = 30
198
    2           if (nOffset < 0) :
198
    3                   nOffset = -nOffset
198
    4                   nColor = 3
198
    5           nPos_X = 420
198
    6           nPos_Y = 560
198
```

```
    ### Clear
    nSize = 0
    for j in range(0, 4) :
```



```
        if (nOffset < pow(10, (j + 1))) :
            for i in range(0, 4 - j) :
                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.

```
                break
    ### Display
    nSize = 120
    for i in range(0, 5) :
        j = (int)(pow(10, (4 - i)))
        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.

```
            nValue = (int)(nOffset / j) % 10
            Show_Num(nPos_X + nGap * i, nPos_Y, nValue, nColor, nSize)
def CameraRun(nMode):
    rpi.mode(nMode)#0: Close, 1: Color, 2: Face Detection, 3: Streaming, 4: Marker, 5: Lane, 6: Emotion, 7: Hand
    if (nMode == 3):
        smart.write8(10700, 1)# smartphone streaming video screen 0: Close, 1: Display
    elif (CVar.nCameraRun == 3):
        smart.write8(10700, 0)
    CVar.nCameraRun = nMode

################################################################################################
#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 and controller settings to suit the robot.
TorqAll (False)
# profile -> velocity-based
```



```
DXL(254).write8(10, 0) # 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)
DXL(3).write8(12, 1) # sync with ID 1
DXL(4).write8(12, 2) # sync with ID 2
# Operation Mode(1:velocity[wheel], 3:position)
DXL(1).mode(const.WHEEL) # wheel
DXL(2).mode(const.WHEEL) # wheel
DXL(3).mode(const.WHEEL) # wheel
DXL(4).mode(const.WHEEL) # wheel
DXL(5).mode(const.JOINT) # position
TorqAll(True)

# In actual use, nTest = 0 should be used.
nTest = 0 # 0-Normal, 1-Coordinate output

# Return all motor speeds to initial state
```

```
Setup_Speed(254, 0)

#Read All Offset
Offset_Read()
Move_Cam_Center()

DXL(1).write32(108, 0)
DXL(2).write32(108, 0)

DXL(1).write32(104, 0)
DXL(2).write32(104, 0)

DXL(3).write32(112, 20)
```

Page 53

```
DXL(3).write32(116, 2048)

rpi.mode(0)#0: Close, 1: Color, 2: Face detection, 3: Streaming, 4: Marker, 5: Lane, 6: Emotion, 7: Hand
smart.write8(10700, 0)# smartphone streaming video screen 0: Close, 1: Display
while(True) :
        if (IsPhone == False) :
                if (smart.is_connected() == True):
                        #Check if it is connected to the smartphone.
                        smart.wait_connected()

                        #Set the smartphone screen horizontally. (0: Auto, 1: Vertical, 2: Horizontal)
                        smart.display.screen_orientation(2)
```

```
                # Waiting for the screen to switch before getting the screen's width and height.
                delay(500)

                # If there is a camera in action, close it.
                #0: Close, 1: Color, 2: Face Detection, 3: Streaming, 4: Marker, 5: Lane, 6: Emotion, 7: Hand
                rpi.mode(0)
                smart.write8(10700, 0)# smartphone streaming video screen 0: close, 1: display

                # Get the resolution of the screen and put it in CVar.nScreenWidth, CVar.nScreenHeight.
                GetResolution()

                # 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()
                # Test 1 => Coordinate output
                if (nTest == 1) :
                        Test1()
                #elif (nTest == 2): # Red dot when clicking with touch 0, blue dot when clicking with touch 1 (second touch)
                # Test2 ()
                else: #Run
```

```
        # Check button press
        nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0, Event_Up1, Btn0, Btn1 =
GetButton(btnList)
        if (Event_Dn1 == 1):
            smart.etc.vibrate(10)
        elif (Event_Dn0 == 1):
            smart.etc.vibrate(10)


        Get_Rot()
        #delay(1000)
        #The menu bar is floating
        if (CVar.nPage == _PAGE_MENU):
            # All motor torque On
            TorqAll(True)
            #All LEDs Off
            DXL(254).write8(65, 0)
            #Position in front of the camera
            Move_Cam_Center()
            #Open the menu bar.
            Page_MenuBar(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)


        else: #If the menu bar is not floating
            if (Btn0 == _BTN_MENU): # Press the menu button
```



```
                ShowPage(0)                 # Open the menu bar.
                WaitButtonUp() # Wait for the button to be released
```

```
        else:
            if (CVar.nPage == _PAGE_MAIN):
                Page_Main(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)
            elif (CVar.nPage == _PAGE_STREAM):
                Page_Stream(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)
            elif (CVar.nPage == _PAGE_MARKER):
                Page_Marker(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)
            elif (CVar.nPage == _PAGE_LINE):
                Page_Line(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)
            elif (CVar.nPage == _PAGE_MOTOR_TEST):
                Page_MotorTest(nNum0, nNum1, Event_Dn0, Event_Dn1,
Event_Up0, Event_Up1, Btn0, Btn1)
            elif (CVar.nPage == _PAGE_OFFSET):
                Page_Offset(nNum0, nNum1, Event_Dn0, Event_Dn1, Event_Up0,
Event_Up1, Btn0, Btn1)
            elif (CVar.nPage == _PAGE_OFFSET_DIALOG1):
                Page_Offset_Dialog_1(nNum0, nNum1, Event_Dn0, Event_Dn1,
Event_Up0, Event_Up1, Btn0, Btn1)
            elif (CVar.nPage == _PAGE_OFFSET_DIALOG2):
                Page_Offset_Dialog_2(nNum0, nNum1, Event_Dn0, Event_Dn1,
Event_Up0, Event_Up1, Btn0, Btn1)
```