from vcScript import \*

from vcHelpers.Robot import \*

from vcHelpers.Robot2 import \*

#from socket import \*

import socket

import time

import errno

import os

local\_ip = '127.0.0.1'

robot\_ip = '127.0.0.1'

recv\_port = 30001

send\_port = 30002

SIZE = 250

app = getApplication()

comp = getComponent()

robot = comp

conectado = False

err = "";

tool = False

def OnStart():

#Executes before sim clock starts running when play is pressed

global robot, exe, ctr, inmap, doStates

global local\_ip, robot\_ip, recv\_port, send\_port, my\_socket, conectado

#Connection vars

local\_ip = comp.getProperty('LocalIP').Value

robot\_ip = comp.getProperty('RobotIP').Value

recv\_port = comp.getProperty('RecvPort').Value

send\_port = comp.getProperty('SendPort').Value

intentoConexion = 5

while (intentoConexion > 0):

print "ROBOT\_4: Intento de conexion nº: %i" %(6-intentoConexion)

try:

my\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

my\_socket.settimeout(1)

my\_socket.connect((robot\_ip,recv\_port))

print "ROBOT\_4: Éxito en la conexion"

break

except socket.error, exc:

print "ROBOT\_4: Error de conexion : %s" % exc

if (intentoConexion == 1):

return

finally:

intentoConexion = intentoConexion - 1

if robot:

exe = robot.findBehavioursByType(VC\_ROBOTEXECUTOR)[0]

inmap = exe.DigitalInputSignals

inmap.OnSignalTrigger = writeDI #Event that writes inputs from sim to robot

ctr = robot.findBehavioursByType(VC\_ROBOTCONTROLLER)[0]

else:

exe = None

ctr = None

inmap = None

doStates = {}

conectado = True

def OnRun():

#Executes after sim clock starts running

global robot, exe, ctr, my\_socket, conectado, err, tool

#Init receiving socket

#print(conectado)

while conectado == True:

print ('ROBOT\_4: Receiving packets from local IP {0}, port {1}\n'.format(local\_ip, recv\_port))

print ('ROBOT\_4: Sending packets to IP {0}, port {1}\n'.format(robot\_ip, send\_port))

if not robot:

print 'ROBOT\_4: No robot associated to TCP link'

return

delay(0.001)

try:

#Send dummy package so robot picks up IP

my\_socket.send('DUMMY')#, (robot\_ip, send\_port))

except:

pass

mt = ctr.createTarget()

mt.MotionType = VC\_MOTIONTARGET\_MT\_JOINT

mt.UseJoints = True

jv = mt.JointValues

robot1=getRobot()

tool=robot1.SignalMapOut.getInternalPortSignal(1)

#estadopinza = pinza.Value

#estadopinza\_old = estadopinza

#infopinza = "Pinza = " + str(estadopinza)

#my\_socket.sendto(infopinza, (robot\_ip, send\_port))

while True:

delay(0.05)

try:

(data1, addr) = my\_socket.recvfrom(SIZE)

data = ""

data = data1.split(" ")

data = data[0]

#print(data)

#Data from robot to sim (polling)

if 'JOINTS' in data:

readJoints(data, mt)

elif 'DO' in data:

readDO(data)

estadotool = (tool.Value == True)\*1

#if (estadopinza != estadopinza\_old):

#estadopinza\_old = estadopinza

infotool = "Tool=" + str(estadotool)

my\_socket.sendto(infotool, (robot\_ip, send\_port))

#print(infopinza)

#Poll interval

delay(0.01)

except socket.error as error:

#print(errno.ECONNREFUSED)

#if error.errno == errno.ECONNREFUSED:

# print(os.strerror(error.errno))

#else:

#raise

delay(0.01)

if error.errno == 10053:

print("ROBOT\_4: Error "+str(error.errno))

print("ROBOT\_4: Intentando reconectar...")

OnStart()

break

else:

print(os.strerror(error.errno))

#endwhile

print('ROBOT\_4: Cerrando el programa')

def readJoints(data, mt):

#Parse joint values and set simulation joints

global ctr

data = data.split(':')

if len(data) < 2:

return

jv = mt.JointValues

data = data[1]

data = data.split(',')

for i in range(len(data)):

val = float(data[i])

jv[i] = float(val)

mt.JointValues = jv

ctr.moveImmediate(mt)

def readDO(data):

#Parse string data and set simulation output if state has changed

global exe, doStates

data = data.split(':')

if len(data) < 2:

return

try:

port = int(data[0][2:])

value = data[1] == '1'

except:

return

old\_value = not value

if data[0] in doStates.keys():

old\_value = doStates[data[0]]

if old\_value != value:

exe.DigitalOutputSignals.output(port, value)

doStates[data[0]] = value

def writeDI(map, port, value):

#Simulation input state changed, send data to robot

global robot\_ip, send\_port, my\_socket

if not inmap:

return

#Data from sim to robot (event-based)

value\_as\_int = 0

if value:

value\_as\_int = 1

data = 'DI%i:%i' % (port, value\_as\_int)

try:

my\_socket.sendto(data, (robot\_ip, send\_port))

except socket.error as e:

my\_socket = socket.socket()

my\_socket.connect((robot\_ip,recv\_port))