Máquinas de Estados:

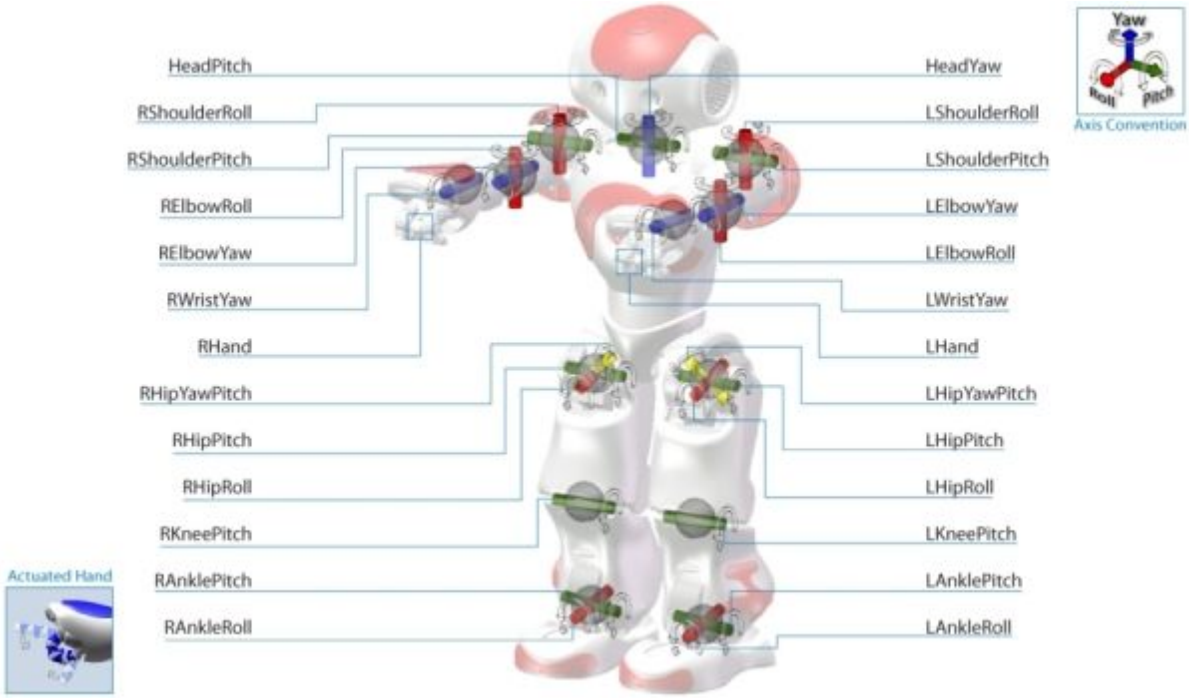
Una máquina de estados es un sistema que puede ser síncrono o asíncrono, este se conforma de diferentes pasos para llegar a una solución, estos pasos se llaman estados. Cada estado tiene una tarea o ejecuta alguna acción mientras está en proceso, para que un estado pueda ser llamado es necesario que algún tipo de señal, active el cambio de estado, de lo contrario se queda en el estado, en el que se encuentra.

Esta señal o esta acción pueden ser múltiples cosas, un sensor, un timer, o algún otro dispositivo que pueda reaccionar a algún estímulo en específico.

Se comienza en algún estado, y normalmente no importa cuál sea el inicial, sin embargo para ciertas aplicaciones se deben comenzar en algún orden en específico.

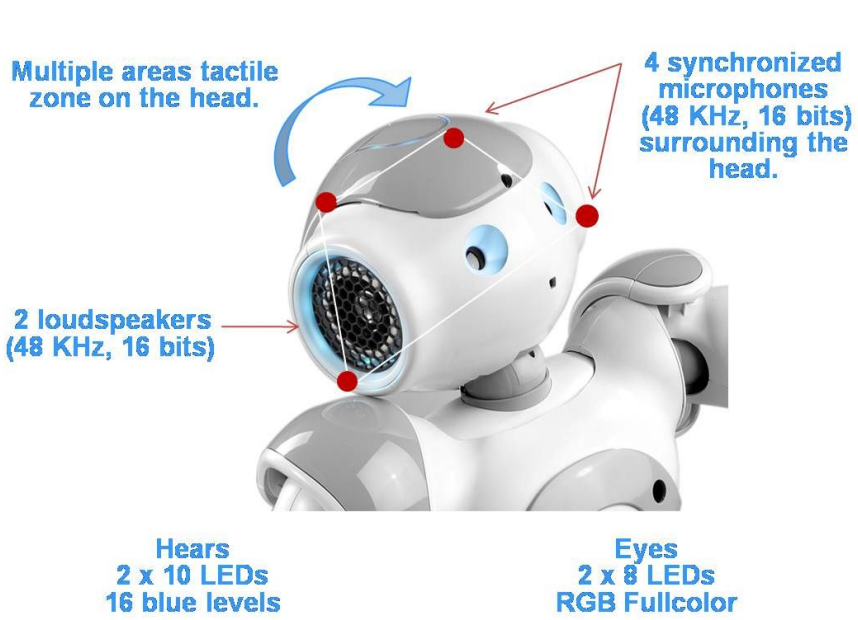
En esta práctica se trabajará con el Robot NAO para poder crear un juego, similar al de Simón Dice, utilizando el concepto de máquina de estados para establecer cada uno de los pasos para su ejecución.

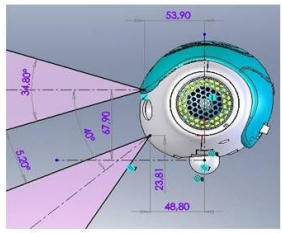
Robot NAO



Simon Dice, el objetivo del juego es poder presionar la secuencia más larga posible de botones. Para esto el juego irá generando aleatoriamente un botón a presionar.

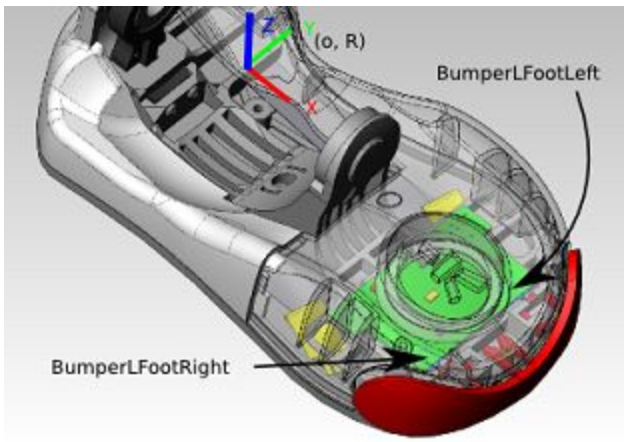
En la versión NAO Dice, se tendrán 6 variables de entrada y 2 variables de salida..



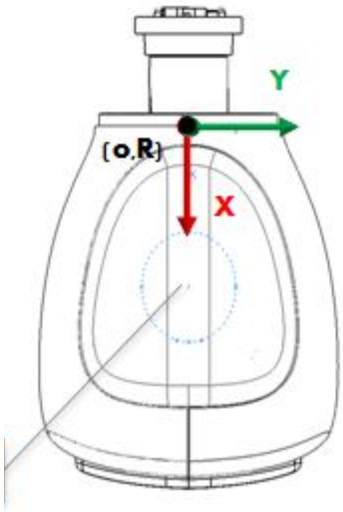


Las variables de entrada serán los 6 botones diferentes, lo cuáles son:

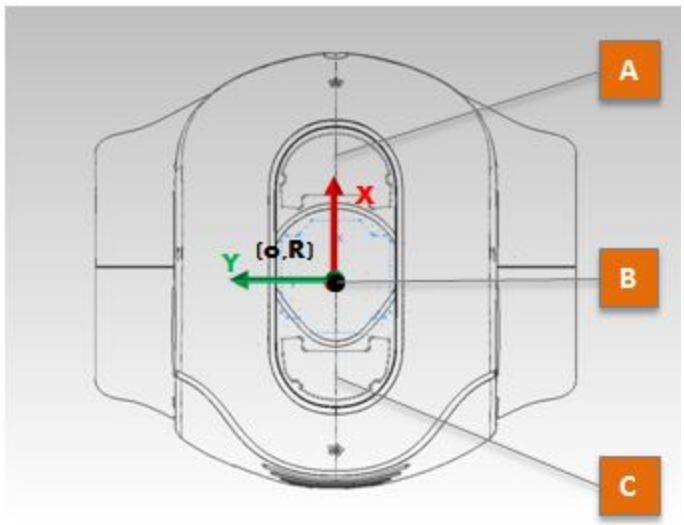
Bumpers izquierdo y derecho Ubicados en los piés.



Táctiles izquierdo y derecho ubicados en las manos.



Táctiles frontal y posterior ubicados en la parte superior de la cabeza.



Más información sobre los sensores se puede encontrar en la siguiente liga:
http://doc.aldebaran.com/2-1/family/robots/contact-sensors_robot.html#robot-contact-feet

Software requerido

Python 2.7, este se puede descargar de la página oficial de Python
https://www.python.org/download/releases/2.7/

Pynaoqi 2.7 para la versión 2.1.4, Para la descarga de este Software se necesita licencia y registro en la página de Aldebarán , por esta razón el Instructor le proporcionará el software.

Para la instalación de las librerías de aldebarán , se seguirán las siguientes instrucciones de la página de Aldebaran.

Usuario: garrido.leonardo@gmail.com

contraseña: naoITESM

http://doc.aldebaran.com/2-1/dev/python/install_guide.html

Agregar el PATH de pynaoqi para que Python pueda encontrar las librerías.

Para eso se realizará los siguiente

cd /project-folder-path

```
export PYTHONPATH=${PYTHONPATH}:/path/to/python-sdk
```

Ejemplo

```
cd /home/aguilerapjc/JC/ITESM/nao/projects
```

```
export PYTHONPATH=${PYTHONPATH}:/home/aguilerapjc/JC/ITESM/nao/pynaoqi-python2.7-2.1.4.13-linux64
```

Una vez realizada la instalación del software al hacer la siguientes líneas, dentro de python, no debería de ocasionar ningún error.

Python

```
>>>import naoqi
```

Para esto se realizarán, diferentes estados:

Seis estados para los botones, un estado para cada uno de ellos.

Estos estados deben de considerar el tiempo que se le va a brindar al jugador para responder a cada uno de los botones que se piden antes de declarar que este olvido la secuencia, esto puede ser entre 3 y 5 segundos, y quedará a criterio de las personas realizando la práctica.

Para realizar un timer sencillo en Python se puede hacer de la siguiente manera.

```
import time

start = time.time()

while(end <= “Tiempo que se quiera esperar”):

        something

        end = time.time() - start
```

Un estado de transición, en donde se generará aleatoriamente uno a uno botones que se agregaran para el juego.

Para generar número aleatoriamente se puede utilizar la librería random

```
#Genera un número aleatorio de 1 al 10 con pasos de 1

import random

numero = random.randrange(1,10,1)
```

Información Importante

Direcciones

Acceso a los sensores:

```
memory.getData("FrontTactilTouched")
memory.getData("RearTactilTouched")
memory.getData("HandRightBackTouched")
memory.getData("HandLeftBackTouched")
memory.getData("RightBumperPressed")
memory.getData("LeftBumperPressed")
```

NOTA:Sensores Táctiles

Para mejor detección de los sensores táctiles hacer el mayor contacto posible con el sensor. Para los sensores de las manos es posible agarrar toda su mano.

Desarrollo de la práctica:

Hacer las siguientes modificaciones al código que se brindará para que se ejecute correctamente.

1) Realizar el Diagrama de estados del sistema
2) Se brindará al estudiante un ejemplo del estado 1, Realizar los estados 2,3,4,5 y 6 de los botones.
3) Realizar el estado de espera.
4) En el estado de transición agregar la selección de los estados de los botones añadiendo los if, else, el ifs necesarios para que se pueda llamar adecuadamente a las funciones previamente diseñadas.

   Para esto se puede utilizar esta función

LED será un objeto que se envió previamente por parámetro y que viene de una declaración de ALProxy

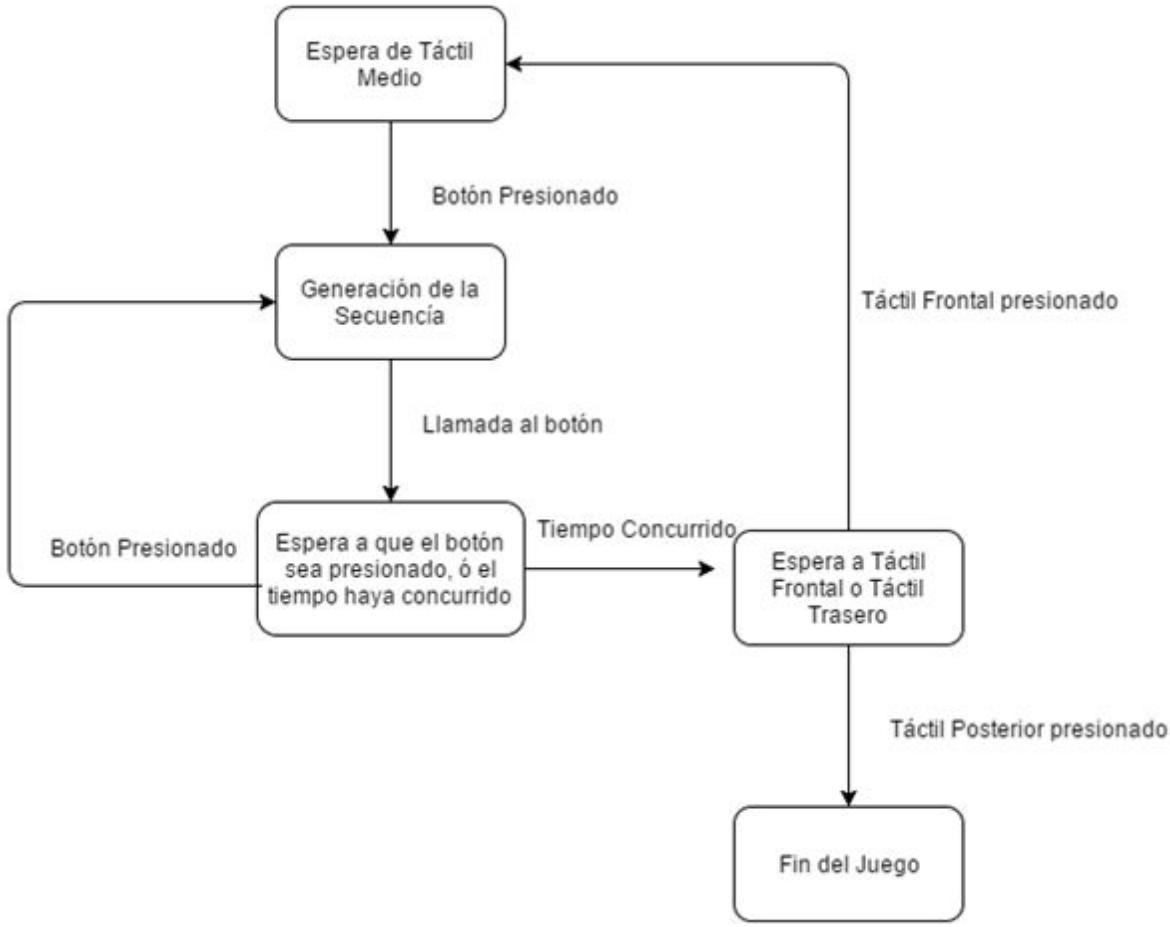
```
LED.fadeRGB('FaceLeds',R,G,B,Time)
```

Donde los valores de R, G, B van de 0 a 1

Time se dará en segundos

1) Diagrama de Estados del Sistema

Ejemplo del Diagrama de un botón, agregar para el diagrama para todo el sistema.



Un estado Inicial el cual esperará a que el jugador inicie o termine el proceso, cada vez que alguien haya perdido.

El programa comenzará con una función main , y otra función que se utiliza para poder acceder a los argumentos de la terminal, estas dos funciones se brindan a continuación.

```
def main(robotIP):

        IP = robotIP

        PORT = 9559

        memory  = ALProxy("ALMemory", IP, PORT)

        motion  = ALProxy("ALMotion", IP, PORT)

        posture = ALProxy("ALRobotPosture", IP, PORT)

        speech  = ALProxy("ALTextToSpeech",IP,PORT)

        LED     = ALProxy("ALLeds",IP,PORT)

              replay=1;

        LED.setIntensity("Eyes",1.0)

              estadoInicial(memory,speech,LED,replay)


if __name__ == "__main__":

    robotIp = "127.0.0.1"

    if len(sys.argv) <= 1:

        print "Usage python motion_walk.py robotIP (optional default: 127.0.0.1)"

    else:

        robotIp = sys.argv[1]

    main(robotIp)
```

Las funciones se deben de insertar previas a estas dos funciones.

Código

El código también puede ser descargada del siguiente enlace:

```
##Code made by Juan Carlos Aguilera Pérez

##For further information can email me at aguilerap.jc@gmail.com

import random

import sys

import motion

import time

from naoqi import ALProxy


#State 1 to 6 have the same way to work


#State 1 Function it will wait for 3 secs to the user to press the Tactil

#If the player does not press it, will return 0 and end the execution,

#else it will return 1 , and continue the game

def state1_Rear_Tactil(memory,LED):

        print("Estado 1 Rear Tactil")

    start = time.time()

        end = 0

        pressed = 0

        while(memory.getData("RearTactilTouched") <> 1.0 and end <  3.0):

            end = time.time()-start

            pressed = 1


        if(end >= 3):
```

```
            pressed = 0
        if(pressed == 0):
            print("Game Over")
        LED.fadeRGB('FaceLeds',1.0,0,0,1.0)
        LED.fadeRGB('FaceLeds',1.0,1.0,1.0,0.5)
        return 0
        else:
            print("Great")
            LED.fadeRGB('FaceLeds',0,1.0,0,1.0)
        LED.fadeRGB('FaceLeds',1.0,1.0,1.0,0.5)
            return 1


##########################################################################
##############Add here the functions of the states that are missing#################
##########################################################################


#Wait function used to wait between the say and the call of the function
#Allows the user to be prepared to press any sensor
def wait05_State():


#State that generate the random numbers and call the sensor methods
def TransitionState(memory,speech,LED):
    on = 1
    state = random.randrange(1,6,1)
```

```
stateArr = [0]*10
counter = 0
state = random.randrange(1,6,1)
stateArr[counter] = state
#Use of ALTextToSpeech proxy to say the sequence to follow
while(on == 1):
    speech.say("Listen to the secuence")
    for x in xrange(0,counter+1):
        if(stateArr[x] == 1):
            speech.say("Rear Tactil")
            wait05_State()
        elif(stateArr[x] == 2):
            speech.say("Front Tactil")
            wait05_State()
        elif(stateArr[x] == 3):
            speech.say("Right Hand")
            wait05_State()
        elif(stateArr[x] == 4):
            speech.say("Left Hand")
            wait05_State()
        elif(stateArr[x] == 5):
            speech.say("Right Foot")
            wait05_State()
        elif(stateArr[x] == 6):
```

```
            speech.say("Left Foot")
            wait05_State()


    #Call the state for each case
for x in xrange(0,counter+1):
        actState = stateArr[x]
        if(actState == 1):
            on = state1_Rear_Tactil(memory,LED)
            if(on == 0):
                break
        ################################################
        #############Add the states that are missing #######
        ##########Those state could be added by elif()######
        ################################################
        else:
            print("State Error")


    state = random.randrange(1,6,1)
    counter +=1
    if(counter >= 10):
        speech.say("Congratulations you WIN!!!")
        LED.rasta(2)
        InitialState(memory,speech,LED,0)
    counter = 0;
```

```
    else:
        LED.fadeRGB('FaceLeds',1.0,0,0,1.0)
        speech.say("So sad, You Lose")
        LED.fadeRGB('FaceLeds',1.0,1.0,1.0,0.5)
        InitialState(memory,speech,LED,0)
        counter = 0;

    stateArr[counter] = state

def InitialState(memory,speech,LED,replay):
    #replay = 1 Play Game
    #replay = 0 Waiting for an answer of the player
    #replay = 2 Player dont want to play anymore, finish the game
    if(replay == 1):
    speech.say("Touch middle tactil to start the game")
        print("Presione sensor tactil medio para comenzar")
        while(memory.getData("MiddleTactilTouched") <> 1.0):
            1+1

        print("El Juego comenzara ahora")
        TransitionState(memory,speech,LED)
    elif(replay == 2):
        speech.say("Game Over")
        print("Juego Terminado")
```

```
    main(0)

    else:

        speech.say("To play again touch front Tactil, else touch rear tactil")

        print("Si desea jugar de nuevo presione el Tactil Frontal, de lo contrario presione Tactil
trasero")

        while(replay == 0):

            if(memory.getData("FrontTactilTouched")):

                replay = 1

                InitialState(memory,speech,LED,replay)

            elif(memory.getData("RearTactilTouched")):

                replay = 2

                InitialState(memory,speech,LED,replay)


def main(robotIP):

    # 1.5

        # Create proxy to ALMemory

    ##Robot IP received as a parameter on Terminal

        IP = robotIP

        PORT = 9559

    #Initialization of Proxies(Aldebaran API's objects)

        memory  = ALProxy("ALMemory", IP, PORT)

        motion  = ALProxy("ALMotion", IP, PORT)

        posture = ALProxy("ALRobotPosture", IP, PORT)

        speech  = ALProxy("ALTextToSpeech",IP,PORT)
```

```
        LED     = ALProxy("ALLeds",IP,PORT)
    replay=1;
        LED.setIntensity('FaceLeds',1.0)
    InitialState(memory,speech,LED,replay)



if __name__ == "__main__":
    robotIp = "127.0.0.1"
    if len(sys.argv) <= 1:
        print "Usage python motion_walk.py robotIP (optional default: 127.0.0.1)"
    else:
        robotIp = sys.argv[1]
    main(robotIp)
```