

Curso de Especialización en Sistemas Embebidos

TESTING DE SOFTWARE

12va Cohorte 2020

Trabajo práctico 2.

Pruebas de sistemas y aceptación.

“Implementación de aprendizaje por

refuerzo en un Robot”

Alumno:

Pablo Daniel Folino

Docentes:

Esp. Ing. Alejandro Permingeat

Esp. Ing. Esteban Volentini

Índice

**1.** [**Historial de cambios**](#HistorialdeCambios) 2

**2.**  [**Introducción**](#Introduccion)3

**3.** [**Elección del subsistema a probar**](#ElecciónSubsistemaProbar)3

**4.** [**Pruebas de Sistema**](#PruebasSistema)3

* 1. [Requerimientos](#Requerimientos). 4
  2. [Técnica de Testing](#TécnicaTesting) 7
     1. [Analizar la descripción de la función](#AnalizarDescripciónFuncion) 7
     2. [Establecer situaciones de prueba](#EstableceSituacionesPrueba) 7
     3. [Establecer casos de prueba lógica](#EstablecerCasosPruebaLógica) 8
     4. [Establecer casos de pruebas físicas](#EstablecerCasosPruebasFísicas) 9
     5. [Establecer controles](#EstablecerControles) 11
     6. [Establecer la situación de inicio](#EstablecerSituaciónInicio) 12
     7. [Ensamblando el script de prueba](#EnsamblandoScriptPrueba) 13

**5.** [**Pruebas de Aceptación**](#PruebasAceptación) **13**

1. Historial de cambios

|  |  |  |
| --- | --- | --- |
| **Fecha** | **Versión** | **Cambio** |
| 08/11/2020 | 1.0 | Versión 1-Generación del documento. |
| 30/11/2020 | 1.1 | Se corrige documento en general y agregan ítems. |
|  |  |  |
|  |  |  |
|  |  |  |

1. Introducción

En el presente documento se detallarán 5 ensayos a Nivel Sistema y 2 Ensayos de Aceptación de distintos módulos de software, perteneciente al firmware del Robot en donde se implementarán técnicas de Aprendizaje por Refuerzo (A.R.).

Como fuentes de información se utilizan las historias de usuario y las especificaciones de requerimientos del Proyecto “Implementación de Aprendizaje por Refuerzo en un Robot” [1] del trabajo final de la Especialización de Sistemas Embebidos-FIUBA, como así también “Especificaciones de Requerimientos de Software” [2], y “Master Test Plan”[3].

El Robot consiste en una plataforma móvil tipo triciclo, y en él se instalan distintos sensores y actuadores, módulos que ayudan a la navegación y al posicionamiento del mismo. A grandes rasgos el “ROBOT\_RL” consta de los siguientes módulos de software:

* Módulo acelerómetro.
* Módulo control de motores.
* Módulos de comunicaciones WIFI-Bluetooth.
* Módulo de posicionamiento (GPS).
* Módulo sensor de ultrasonido.
* Módulo de administrador de energía.
* Módulo principal.
* Módulo encoder.

1. **Elección del subsistema a probar**

Se decide probar el subsistema de Control de Motores, debido a que es uno de los más importantes del proyecto y por lo tanto resulta primordial que el mismo cuente con un buen diseño de testing, a los fines de poder garantizar su correcto funcionamiento al momento de implementarlo.

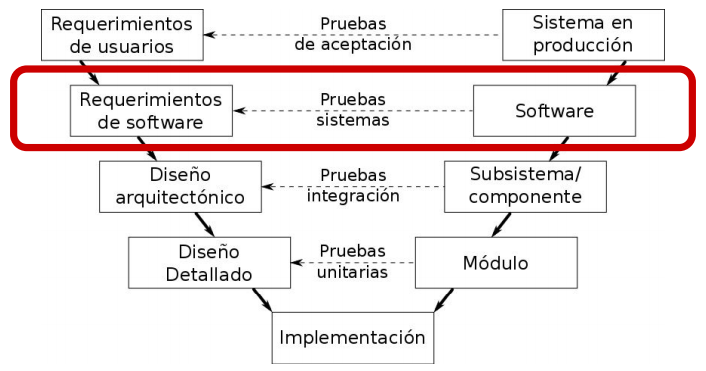
1. **Pruebas de Sistema**

Se decide probar el subsistema de Control de Motores, debido a que es uno de los más importantes del proyecto y por lo tanto resulta primordial que el mismo cuente con un buen diseño de testing, a los fines de poder garantizar su correcto funcionamiento al momento de implementarlo.

* 1. **Requerimientos**

Las pruebas de sistema básicamente tienen la finalidad de probar que se cumplen todos los requerimientos, tal como puede observarse en la Figura 1, por lo que es necesario plantear

primero los requerimientos del subsistema que se desea probar.



Los requerimientos de sistema se desprenden del documento de “Especificación de requerimientos de software” realizados en la materia de Ingeniería de Software, se usará la primera versión de dicho documento. Para el módulo de control de motores se tienen los siguientes ítems:

1. El subsistema de ***Control de Motores*** recibe solamente información del ***Módulo Principal***, y le brinda a éste estados del módulo.
2. Controla dos motores en forma aislada mediante señales PWM, y lee el estado de avance mediante un sensor óptico instalado en cada uno de los ejes (odometría).
3. Los principales requerimientos son:
   1. El software deberá manejar 2 motores con ***PWM*** en forma independiente. [ROBOT\_RL-ER-0001-REQ0004].
   2. El software deberá permitir deshabilitar el módulo ***PWM***. [ROBOT\_RL-ER-0001 -REQ0005]​
   3. El software podrá recibir tramas de control de movimiento, hacia adelante, hacia atrás, giro hacia la derecha e izquierda y parada.[ROBOT\_RL-ER-0001 -REQ0017]
   4. El ciclo de control de los motores no será mayor a 100mseg. [ROBOT\_RL-ER-0001 -REQ0018]​
   5. El valor de velocidad en cada motor contemplará no enviar información en la zona muerta del mismo, para evitar recalentamiento y gasto de energía innecesaria. [ROBOT\_RL-ER-0001 -REQ0019]​
   6. **Técnica de Testing**

Para probar el módulo, se empleará la técnica ECT (Elementary Comparison Test), por lo que es necesario brindar una descripción de la función que se va a probar, que en este caso es la función de setear la velocidad en uno de los motores. A continuación, se muestra un pseudocódigo que representa el comportamiento del submódulo a probar: El valor de velocidad recibido puede ser +/- 255.

velocidad=recibir\_velocidad()

**IF** velocidad > 0

**THEN**

sendido\_del\_motor=ADELANTE

**IF** velocidad<= maxVel **AND** velocidad>=minVel

**THEN**

setear\_velocidad( )

**ELSE**

**IF** velocidad> maxVel

**THEN**

setear\_velocidad\_MAXVEL()

**ENDIF**

**IF** velocidad<minVel

**THEN**

setear\_velocidad\_CERO()

**ENDIF**

**ENDIF**

**ENDIF**

**IF** velocidad<0

**THEN**

sendido\_del\_motor = ATRAS

velocidad=-velocidad

**IF** velocidad<= maxVel **AND** velocidad>=minVel

**THEN**

setear\_velocidad( )

**ELSE**

**IF** velocidad> maxVel

**THEN**

setear\_velocidad\_MAXVEL()

**ENDIF**

**IF** velocidad<minVel

**THEN**

setear\_velocidad\_CERO()

**ENDIF**

**ENDIF**

**ENDIF**

**IF** velocidad=0

**THEN**

setear\_velocidad\_CERO()

**ENDIF**

|  |
| --- |
|  |

**4.2.1** **Analizar la descripción de la función**

velocidad=recibir\_velocidad()

**C1 IF** velocidad > 0

**THEN**

sendido\_del\_motor=ADELANTE

**C2** **IF** velocidad<= maxVel **AND** velocidad>=minVel

**THEN**

setear\_velocidad( )

**ELSE**

**C3**  **IF** velocidad> maxVel

**THEN**

setear\_velocidad\_MAXVEL()

**ENDIF**

**C4**  **IF** velocidad<minVel

**THEN**

setear\_velocidad\_CERO()

**ENDIF**

**ENDIF**

**ENDIF**

**C5 IF** velocidad<0

**THEN**

sendido\_del\_motor = ATRAS

velocidad=-velocidad

**C6** **IF** velocidad<= maxVel **AND** velocidad>=minVel

**THEN**

setear\_velocidad( )

**ELSE**

**C7** **IF** velocidad> maxVel

**THEN**

setear\_velocidad\_MAXVEL()

**ENDIF**

**C8** **IF** velocidad<minVel

**THEN**

setear\_velocidad\_CERO()

**ENDIF**

**ENDIF**

**ENDIF**

**C9 IF** velocidad=0

**THEN**

setear\_velocidad\_CERO()

**ENDIF**

**4.2.2** **Establecer situaciones de prueba**

En esta etapa se confecciona una tabla de comparación para cada condición, ya sea de comparación simple o de comparación compuesta no combinada.

En la tabla se muestra en qué casos se evaluará la condición en cuestión como verdadera o falsa.

|  |  |  |
| --- | --- | --- |
| **Test situation** | **C1.1** | **C1.2** |
| C1 | TRUE | FALSE |
| velocidad | >0 | <=0 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test situation** | **C2.1** | **C2.2** | **C2.3** |
| C2 = (C2.a & C2.b) | 1(11) | 0(10) | 0(01) |
| C2.a velocidad <= maxVel | <= maxVel | <= maxVel | > maxVel |
| C2.b velocidad >= minVel | >=minVel | <minVel | >=minVel |

|  |  |  |
| --- | --- | --- |
| **Test situation** | **C3.1** | **C3.2** |
| C3 | TRUE | FALSE |
| velocidad > maxVel | > maxVel | <= maxVel |

|  |  |  |
| --- | --- | --- |
| **Test situation** | **C4.1** | **C4.2** |
| C4 | TRUE | FALSE |
| velocidad < minVel | < minVel | >= minVel |

|  |  |  |
| --- | --- | --- |
| **Test situation** | **C5.1** | **C5.2** |
| C5 | TRUE | FALSE |
| velocidad | < 0 | >=0 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test situation** | **C6.1** | **C6.2** | **C6.3** |
| C6 = (C6.a & C6.b) | 1(11) | 0(10) | 0(01) |
| C6.a velocidad <= maxVel | <= maxVel | <= maxVel | > maxVel |
| C6.b velocidad >= minVel | >=minVel | <minVel | >=minVel |

|  |  |  |
| --- | --- | --- |
| **Test situation** | **C7.1** | **C7.2** |
| C7 | TRUE | FALSE |
| velocidad > maxVel | > maxVel | <= maxVel |

|  |  |  |
| --- | --- | --- |
| **Test situation** | **C8.1** | **C8.2** |
| C8 | TRUE | FALSE |
| velocidad < minVel | < minVel | >= minVel |

|  |  |  |
| --- | --- | --- |
| **Test situation** | **C9.1** | **C9.2** |
| C9 | TRUE | FALSE |
| velocidad =0 | =0 | >< 0 |

**4.2.3** **Establecer casos de prueba lógica**

En este punto, se deben encontrar los diferentes caminos funcionales en los que cada

situación de cada condición se considera al menos una vez. Para ello, se confecciona una tabla indicando los diferentes valores de cada condición, y hacia qué condición iría a continuación. Y por último en una columna de “Control”, se contabiliza por cuántas veces se pasó por cada situación de prueba.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test situation** | **Value** | **To** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **Control** |
| C1.1 | 1 | C2 |  | x | x | x |  |  |  | 3 |
| C1.2 | 0 | C5 | x |  |  |  | x | x | x | 4 |
|  |  |  |  |  |  |  |  |  |  |  |
| C2.1 | 1 | C5 |  |  | x |  |  |  |  | 1 |
| C2.2 | 0 | C3 |  |  |  | x |  |  |  | 1 |
| C2.3 | 0 | C3 |  | x |  |  |  |  |  | 1 |
|  |  |  |  |  |  |  |  |  |  |  |
| C3.1 | 1 | C4 |  |  |  | x |  |  |  | 1 |
| C3.2 | 0 | C4 |  | x |  |  |  |  |  | 1 |
|  |  |  |  |  |  |  |  |  |  |  |
| C4.1 | 1 | C5 |  | x |  |  |  |  |  | 1 |
| C4.2 | 0 | C5 |  |  |  | x |  |  |  | 1 |
|  |  |  |  |  |  |  |  |  |  |  |
| C5.1 | 1 | C6 |  |  |  |  | x | x | x | 3 |
| C5.2 | 0 | C9 | x | x | x | x |  |  |  | 4 |
|  |  |  |  |  |  |  |  |  |  |  |
| C6.1 | 1 | C9 |  |  |  |  |  | x |  | 1 |
| C6.2 | 0 | C7 |  |  |  |  |  |  | x | 1 |
| C6.3 | 0 | C7 |  |  |  |  | x |  |  | 1 |
|  |  |  |  |  |  |  |  |  |  |  |
| C7.1 | 1 | C8 |  |  |  |  |  |  | x | 1 |
| C7.2 | 0 | C8 |  |  |  |  | x |  |  | 1 |
|  |  |  |  |  |  |  |  |  |  |  |
| C8.1 | 1 | C9 |  |  |  |  | x |  |  | 1 |
| C8.2 | 0 | C9 |  |  |  |  |  |  | x | 1 |
|  |  |  |  |  |  |  |  |  |  |  |
| C9.1 | 1 | End | x |  |  |  |  |  |  | 1 |
| C9.2 | 0 | End |  | x | x | x | x | x | x | 6 |

**4.2.4**  **Establecer casos de pruebas físicas**

Para establecer los casos de prueba físico, se le dan valores reales a la secuencia del punto anterior.

Variables generales:

|  |  |
| --- | --- |
| **Nombre de la variable** | **Valor** |
| maxVel | 250 |
| minVel | 127 |

Caminos lógicos en función de la variable de entrada(velocidad):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Caso de Testeo | velocidad |  |  |  |  |  |  |
| 1 | 0 | C1.2 | C5.2 | C9.1 |  |  |  |
| 2 | 100 | C1.1 | C2.3 | C3.2 | C4.1 | C5.2 | C9.2 |
| 3 | 230 | C1.1 | C2.1 | C5.2 | C9.2 |  |  |
| 4 | 255 | C1.1 | C2.2 | C3.1 | C4.2 | C5.2 | C9.2 |
| 5 | -100 | C1.2 | C5.1 | C6.3 | C7.2 | C8.1 | C9.2 |
| 6 | -230 | C1.2 | C5.1 | C6.1 | C9.2 |  |  |
| 7 | -255 | C1.2 | C5.1 | C6.2 | C7.1 | C8.2 | C9.2 |

**4.2.5** **Establecer controles**

|  |  |  |
| --- | --- | --- |
| **Chequeo** | **Caso de Testeo** | **Situación esperada** |
| **CO1** | 1 | Motor **parado** PWM=0 |
| **CO2** | 2 | Motor **parado** PWM=0 |
| **CO3** | 3 | Motor hacia **adelante** PWM=230 |
| **CO4** | 4 | Motor hacia **adelante** PWM= maxVel = 250 |
| **CO5** | 5 | Motor **parado** PWM=0 |
| **CO6** | 6 | Motor hacia **atrás** PWM=230 |
| **CO7** | 7 | Motor hacia **atrás** PWM= maxVel = 250 |

**4.2.5** **Establecer la situación de inicio**

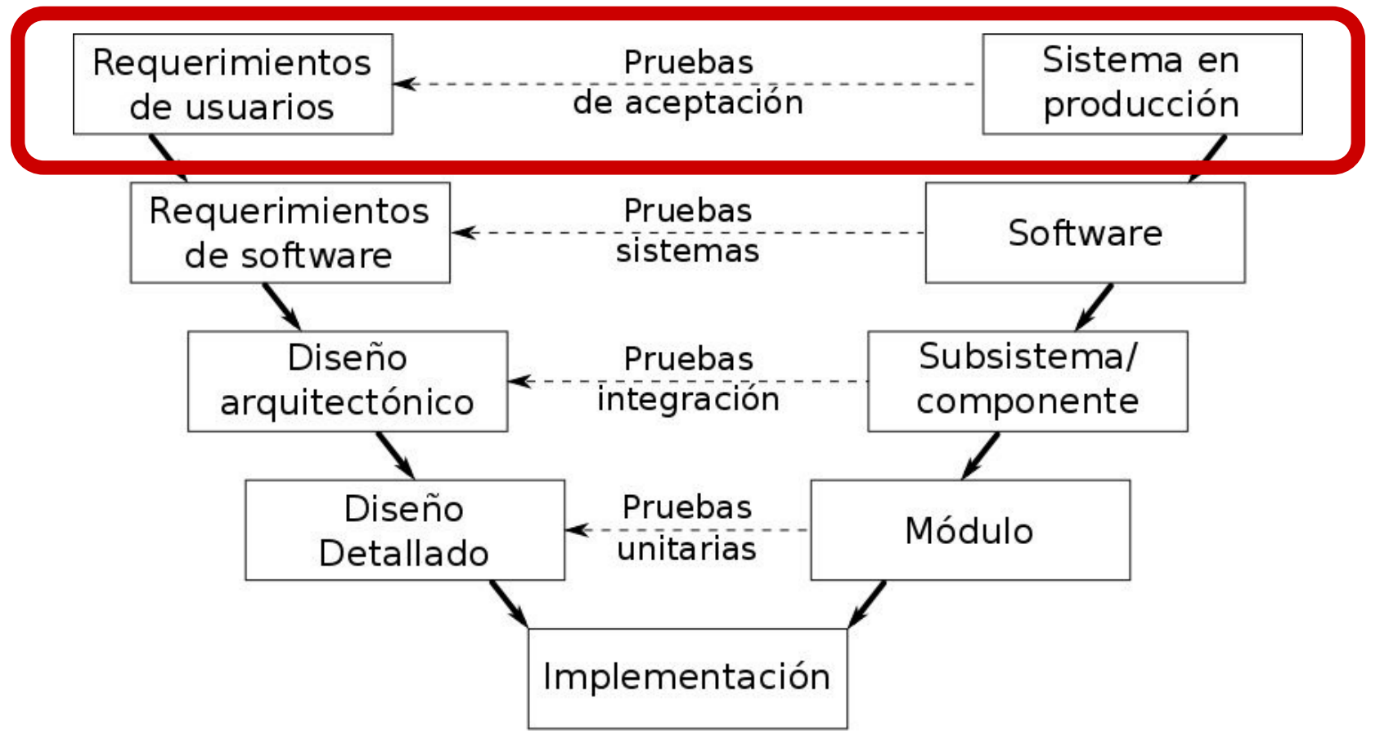
Antes de ejecutar la secuencia de los casos de prueba descripto en el punto 4.2.4, el robot deberá estar en su base de prueba (para que las ruedas no toquen el suelo), estar encendido en el estado de reposo, con las baterías cargadas por encima del 75% y los valores de las variables “maxVel” y “minVel” con los datos suministrados en la tabla del ítem 4.2.4.

**4.2.6** **Ensamblando el script de prueba**

Para el ensamblado de los 7 casos de prueba, se recomienda seguir la secuencia de la tabla 4.2.3, en donde primero se prueban los valores positivos de velocidad y luego los negativos en forma creciente.

**5. P****ruebas de aceptación**

Las pruebas de aceptación se realizan para permitir que un usuario o cliente final acepte el componente o sistema total que se está probando. Se encuentran un nivel por encima de las pruebas de sistema en el modelo V



En general son un subconjunto de las pruebas de sistema, siendo este el caso del componente del submódulo setear la velocidad de los motores. Por lo tanto, las pruebas de aceptación ya quedan cubiertas en las exhaustivas pruebas de sistema planteadas previamente. A modo de ejemplo, una prueba de aceptación razonable, muy cercana al uso que el usuario eventualmente le daría, sería aquella en la que configura la velocidad por debajo de minVel (para que entre el valor en la zona muerta del motor), y el motor no reaccione.

Otra de las pruebas que se puede hacer es configurar una velocidad en ambos motores con el mismo sentido y superior a maxVel, y ver que el móvil cumple con el requerimiento que se consensuó con el usuario en el ítem 4.4.2 del documento del proyecto, en donde dice que “La velocidad que desarrolla el robot debe ser inferior a 1 m/seg”.