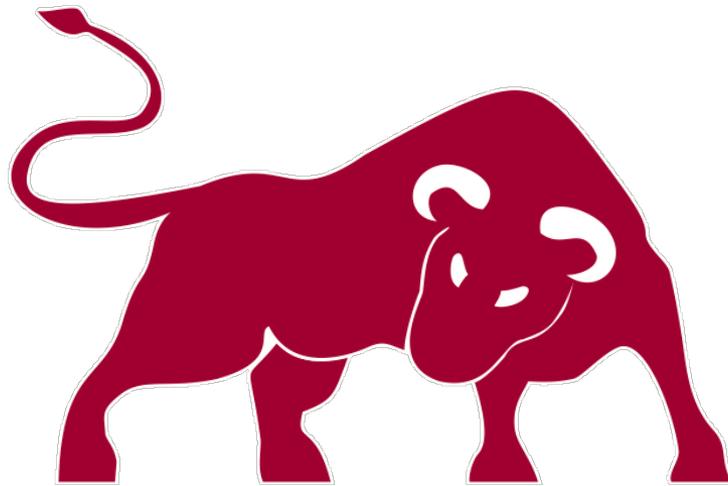


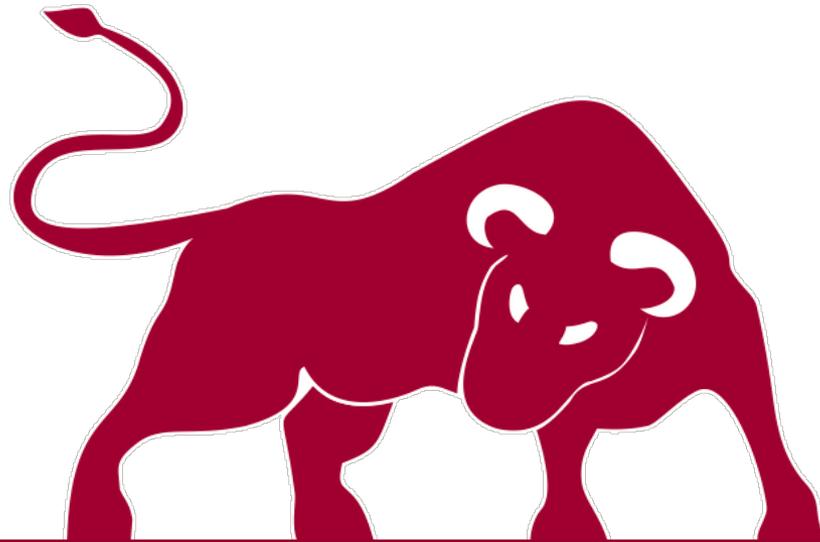
Torocó



Ignacio Bettosini - Agustín Clavelli

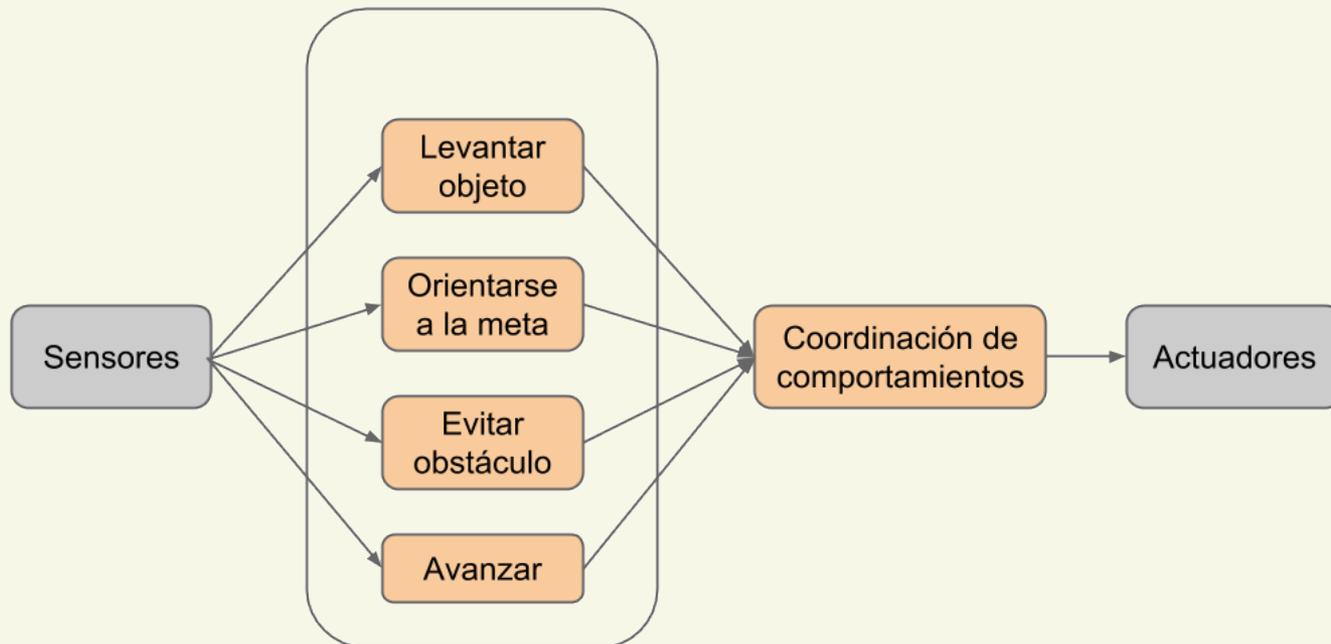
Facultad de Ingeniería - Universidad de la República

1. Arquitectura



Arquitectura Subsumption

- Comportamientos concurrentes.
- Intercambian datos entre sí y con el hardware.
- Desarrollo incremental.



Módulos

Comportamientos

- Procesan datos en paralelo.

Devices

- Permiten acceder a sensores, actuadores, y sistemas externos (visión, GPS, otros robots).

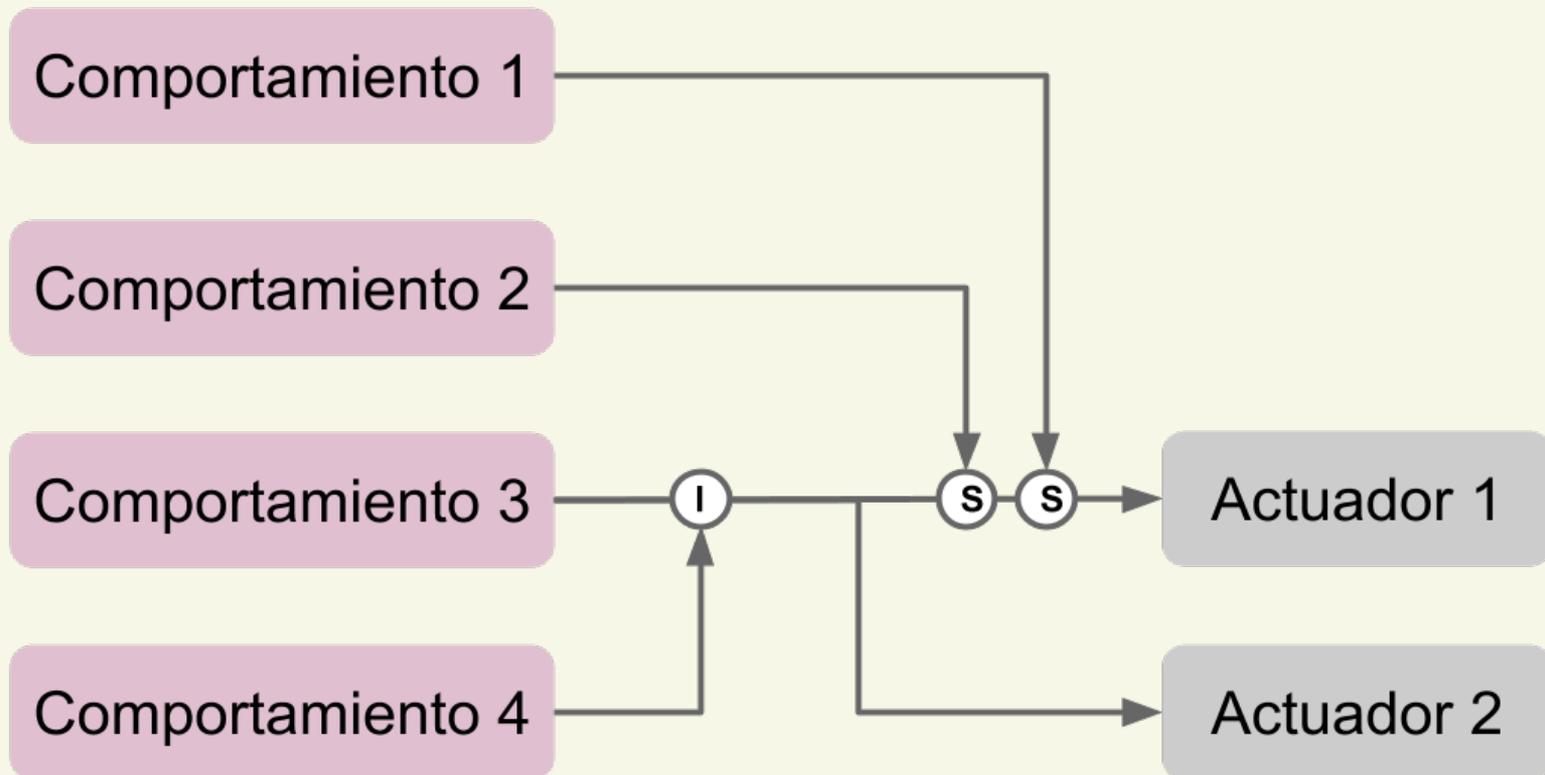


Comunicación entre módulos

- Uso de eventos:
 - Enviar eventos con datos adjuntos.
 - Esperar por un evento individual.
 - Registrarse a un evento de manera permanente.
- Comportamientos y devices son indistintos.

Comunicación entre módulos (2)

- **Supresión:** Lista de eventos para cada entrada.
- **Inhibición:** Lista de eventos para cada salida.



Comunicación entre módulos (3)

Entradas y salidas

- Los módulos definen entradas y salidas.
- Desconocen emisores y receptores.
- La interconexión se define en el programa principal, o sea por fuera de los comportamientos.
- Ventajas: Módulos desacoplados y reutilizables.

Funcionamiento de Torocó

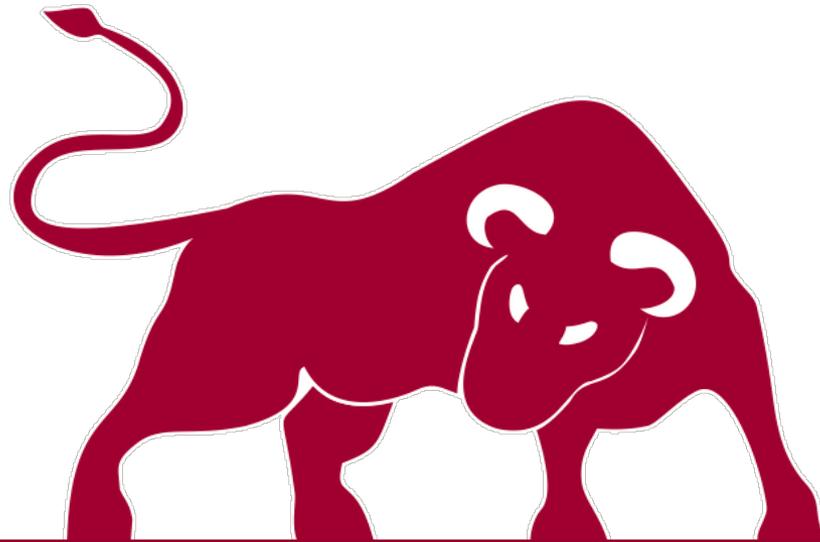
Despachador

- Se encarga de reenviar eventos, aplicando inhibición y supresión.
- Pseudocódigo:
Por cada evento de salida,
 Si la salida no está inhibida,
 Por cada receptor,
 Si no está suprimido,
 Enviar evento al receptor.

Consideraciones

- Comportamientos sencillos, complejidad en interconexiones.
- Conducta compleja con código compacto.
- Planificador no expropiador.

2. Ejemplo

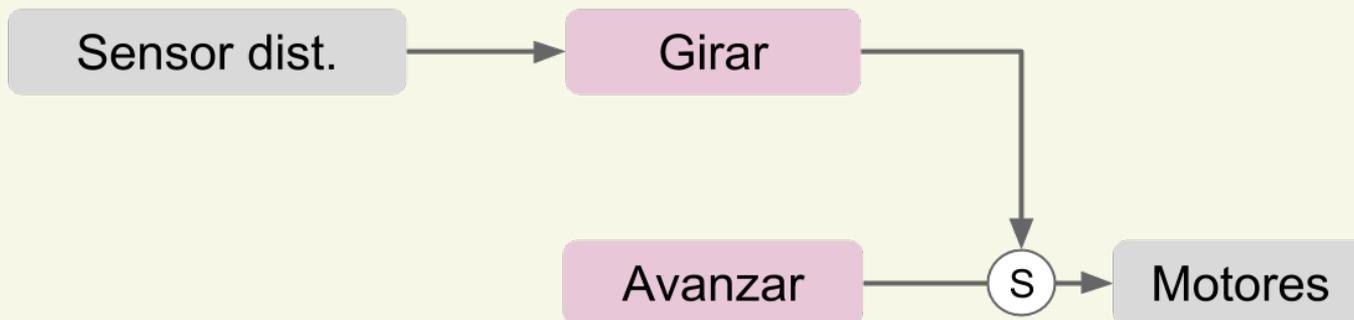


Estructura de carpetas

Nombre	Tamaño	Tipo
 behaviors	4,1 kB	carpeta
 deviceloaders	4,1 kB	carpeta
 nixio	4,1 kB	enlace a ../../nixio
 toribio	4,1 kB	enlace a ../../toribio
 toroco	4,1 kB	enlace a ../../toroco
 toribio.conf	98 bytes	documento de texto sencillo
 nixio.so	117,5 kB	enlace a ../../nixio.so
 main.lua	631 bytes	script en Lua

Robot que evita obstáculos

- **Hardware**
 - Sensor de distancia
 - Motores de propulsión
- **Arquitectura**
 - Si no hay obstáculos delante -> Avanzar.
 - Si hay obstáculos delante -> Girar.



Interfaz: Restablecer eventos

Opción 1: Reenviar manualmente

El comportamiento de avanzar reenvía el evento constantemente. Poco eficiente y elegante.

Opción 2: Eventos fijos

Cuando el evento de avanzar es suprimido y luego se levanta la supresión, Torocó reenvía el evento a los receptores.

Interfaz: Comportamientos

avanzar.lua

```
local coroutine1 = function ()  
    toroco.set_output {velocidad = {90, 90}}  
end  
  
return coroutine1
```

Interfaz: Comportamientos

girar.lua

```
local handler1 = function (evento, hay_objeto)
    if hay_objeto then
        toroco.set_output {velocidad = {-90, 90}}
    else
        toroco.unset_output ()
    end
end

local trigger1 = toroco.trigger (input.distancia, handler1)

return trigger1
```

Interfaz: Programa principal

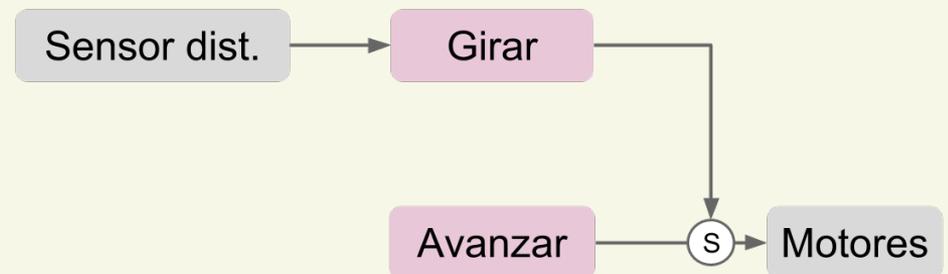
```
toroco.load_behavior (behavior.avanzar, 'behaviors/avanzar')
```

```
toroco.load_behavior (behavior.girar, 'behaviors/girar')
```

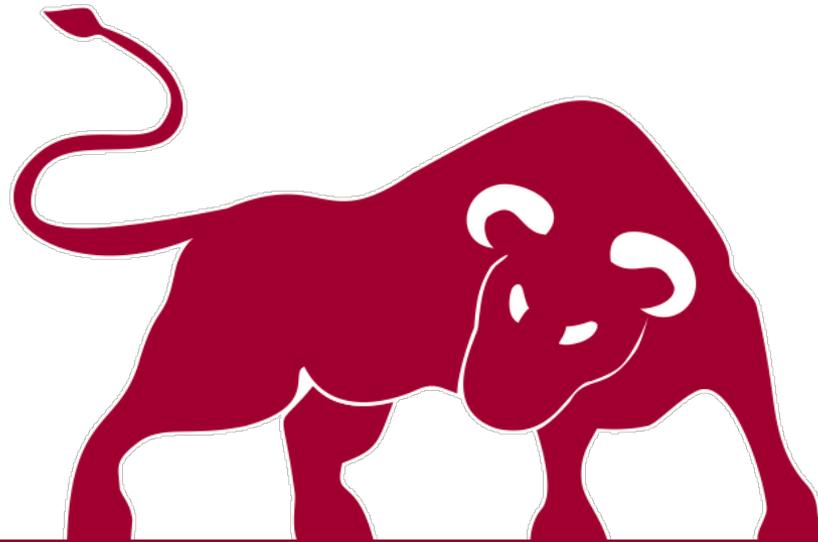
```
toroco.set_inputs (behavior.girar, {  
    distancia = device.distance.update  
})
```

```
toroco.set_inputs (device.motors, {  
    set_speed = {  
        behavior.avanzar.velocidad,  
        behavior.girar.velocidad  
    }  
})
```

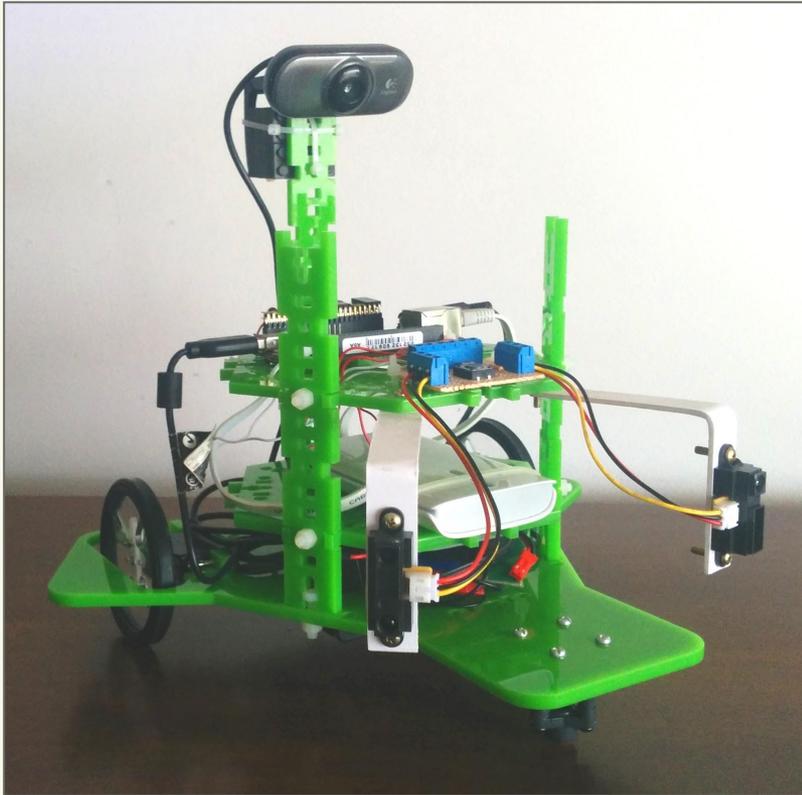
```
toroco.run ()
```



3. Torobot

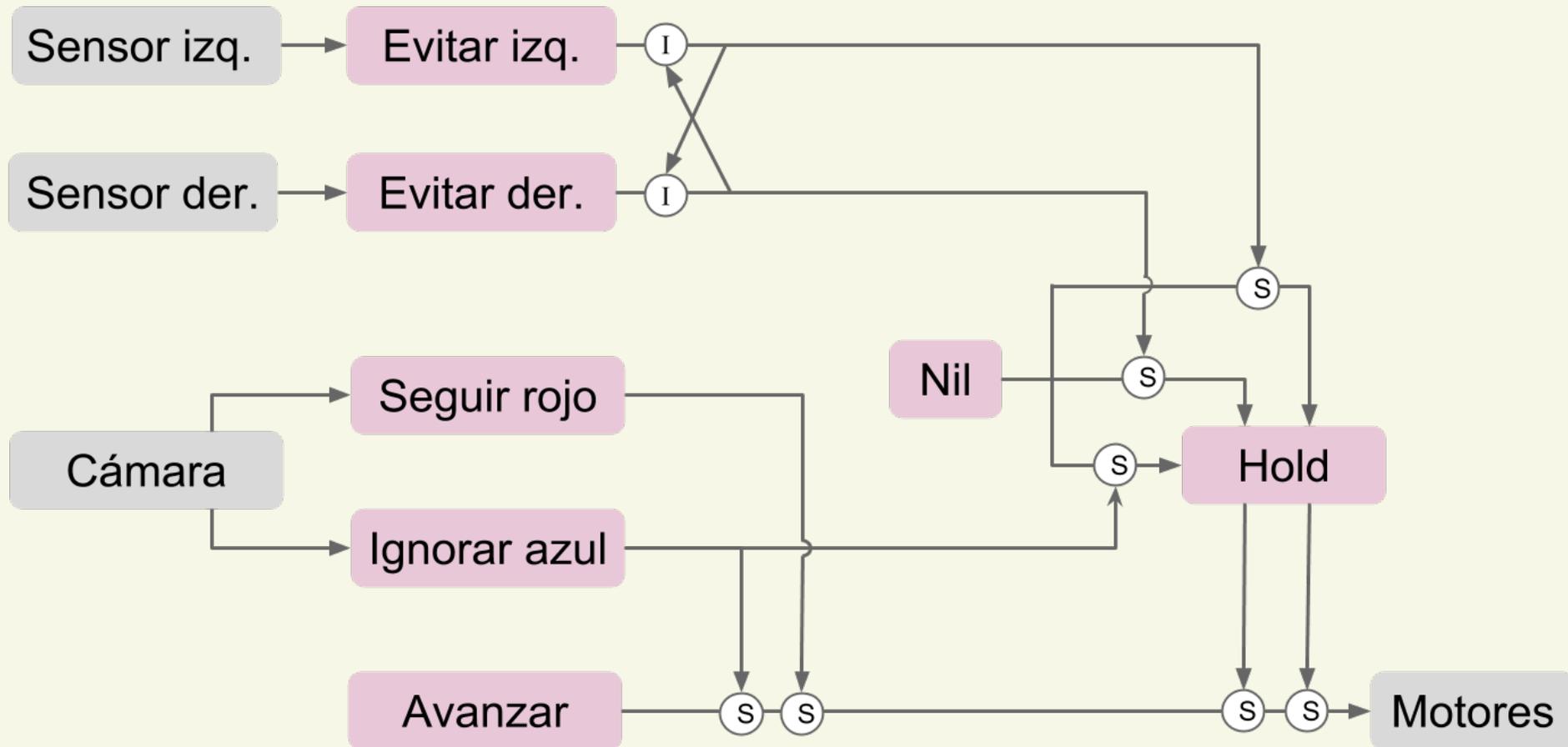


Torobot

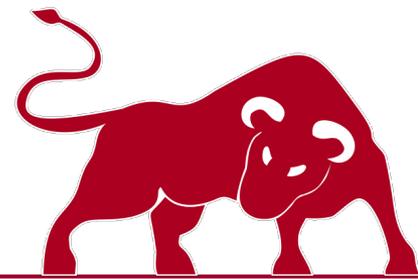


- Motores para avanzar.
- Cámara web para reconocer obstáculos.
- Sensores de distancia para evitar paredes.

Torobot



¿Preguntas?



Proyecto Torocó

www.fing.edu.uy/~pgtoroco