Universidad
Rey Juan Carlos

Motor de físicas mejorado para simulador robótico basado en tecnologías web

TRABAJO FIN DE GRADO

NATALIA MONFORTE RODRÍGUEZ

JOSÉ MARÍA CAÑAS PLAZA

OCTUBRE 2020

Índice

Introducción

(a) Roomba

(b) Thermomix

(c) Tello

(d) Tesla

(e) Robot DaVinci

(f) Robots LEGO

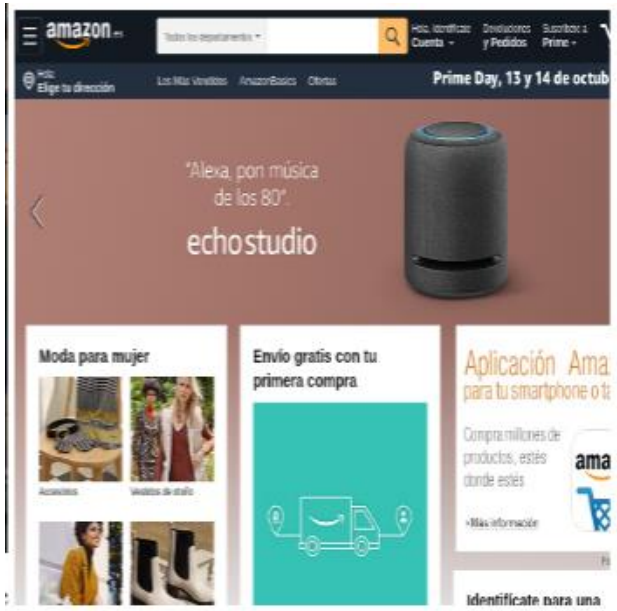
Robótica

La robótica es la disciplina que estudia la creación de máquinas automatizadas capaces de recrear comportamientos humanos o animales en función del software que lleven incorporados.

Un robot presenta dos partes bien diferenciadas:

- Hardware: se encuentran sensores, actuadores y ordenadores.
- Software: donde reside la inteligencia

Tecnologías web



- Aplicaciones web

- Modelo cliente – servidor -> Protocolo HTTP

- Tecnologías de *frontend:*
  - *HTML5*
  - *CSS3*
  - *JavaScript*

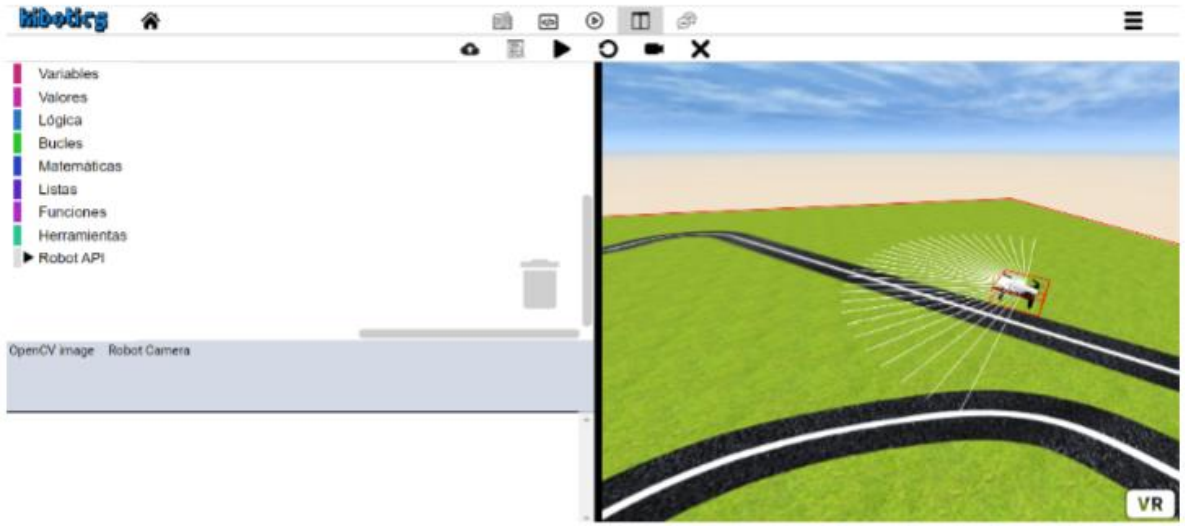
- Tecnologías de *backend:*
  - *node.js*
  - *Django*
  - *Spring*

Docencia robótica

Robótica + tecnologías web ⟶ docencia robótica

Educación *STEM*

- *Lenobotics*
- *LEGO education*
- *OpenRoberta*
- *iRobot*
- *Scratch*
- *Kibotics* ⟶ soporta *Python* y *Scratch*



Interfaz de programación en *Kibotics* de un ejercicio en *Scratch*

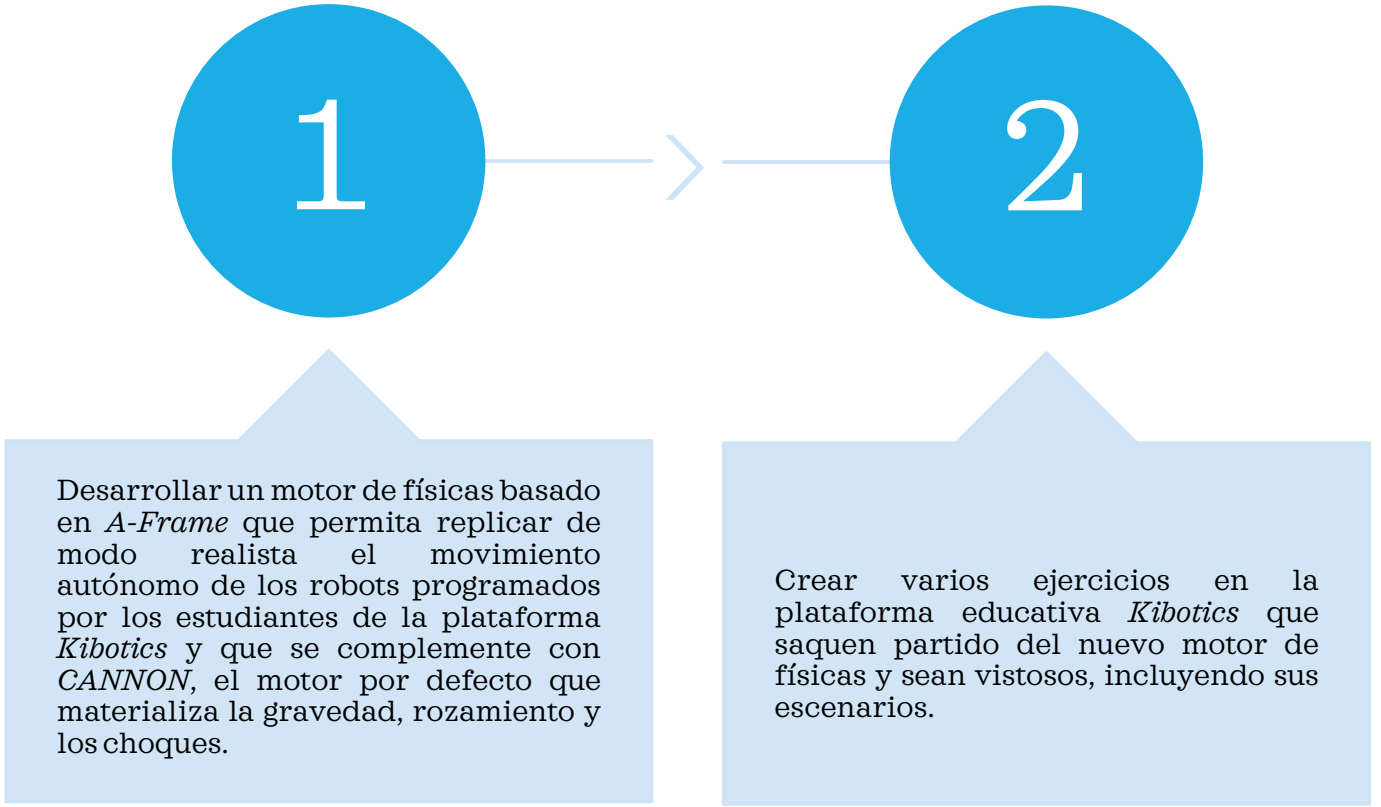
Programación con bloques *Scratch*

(a) piBot (b) mBot (c) Drone Tello (d) Fórmula 1

Robots soportados en la plataforma *Kibotics*

Objetivos

Objetivos


1
2
Desarrollar un motor de físicas basado en *A-Frame* que permita replicar de modo realista el movimiento autónomo de los robots programados por los estudiantes de la plataforma *Kibotics* y que se complemente con *CANNON*, el motor por defecto que materializa la gravedad, rozamiento y los choques.
Crear varios ejercicios en la plataforma educativa *Kibotics* que saquen partido del nuevo motor de físicas y sean vistosos, incluyendo sus escenarios.

Herramientas


kibotics


Javascript
JS


HTML
5

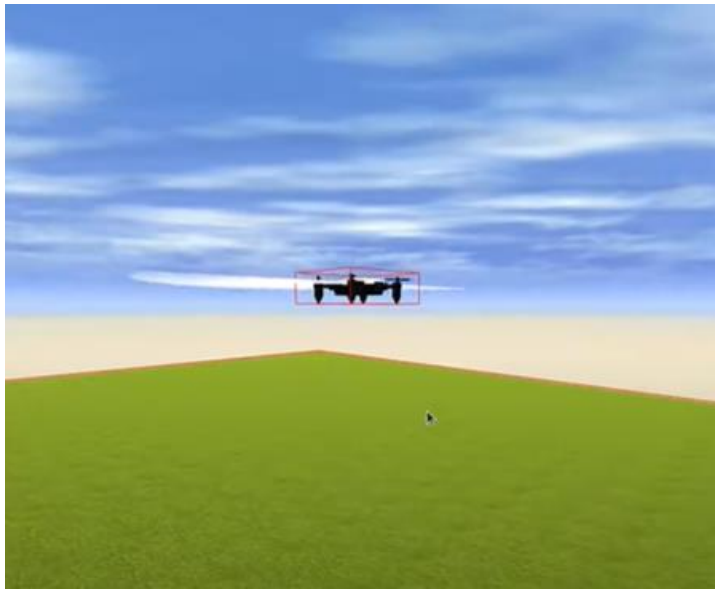

JSON

Mejora de las físicas en *WebSim*

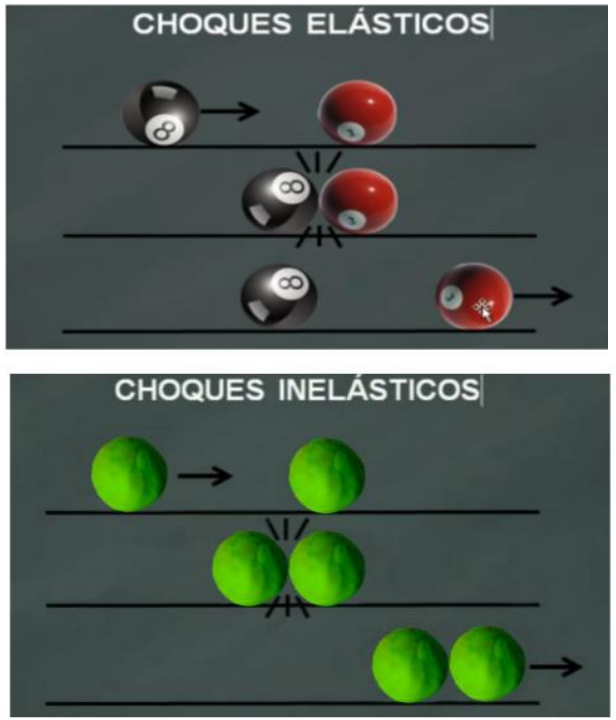
Estudios previos: motor de físicas por defecto en *A-Frame*

- *CANNON* es el motor por defecto de *A-Frame*

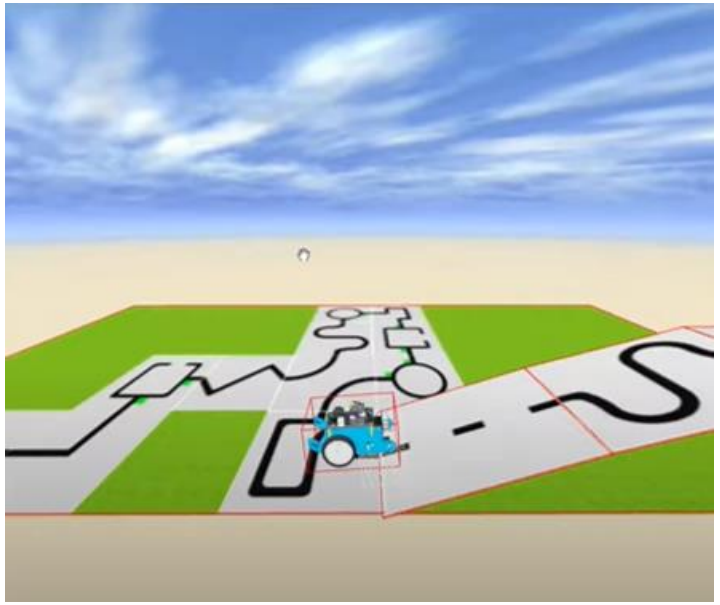
Gravedad



Colisiones



Fricción



Motor de físicas actual para robots en *WebSim*

- Las físicas implementadas no recreaban un movimiento realista y no eran suficientes para lo que se tiene en un robot real.

- Modelo cinemático donde se asume que los robots adquieren instantáneamente la velocidad ideal que el software del robot ordena. Se asume una aceleración infinita.

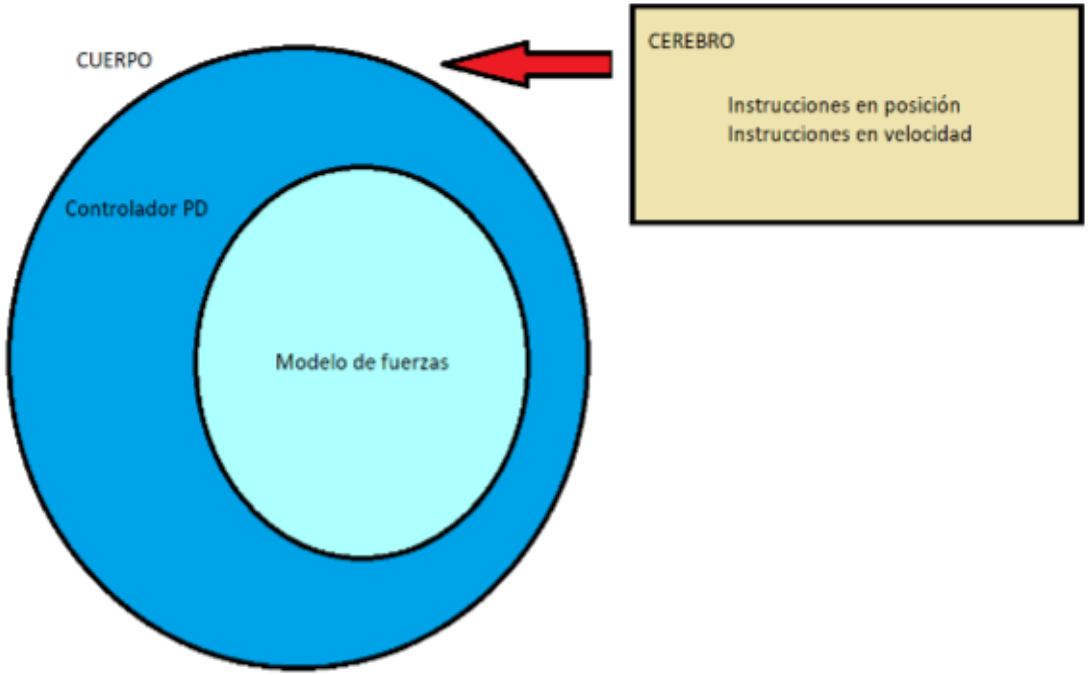
```
updatePosition(rotation, velocity, robotPos) {
    if(simEnabled){
        let x = velocity.x / 10 * Math.cos(rotation.y * Math.PI / 180);
        let z = velocity.x / 10 * Math.sin(-rotation.y * Math.PI / 180);
        let y = (velocity.y / 10);
        robotPos.x += x;
        robotPos.z += z;
        robotPos.y += y;
    }
    return robotPos;
}
```

Nuevo motor de físicas complementario

1. DISEÑO

```
Fuerza robot = Fuerza autónoma + Fuerza gravedad + Fuerza fricción
```

- Fuerzas gravedad y fricción: materializada por *CANNON a* un ritmo marcado por el propio motor *CANNON*.
- Fuerza autónoma: lo materializa nuestro motor complementario a su propio ritmo y teniendo en cuenta las velocidades deseadas que marca en cada instante el código fuente del cerebro programado.



Parámetros del modelo de fuerzas	
mass	Masa del robot
inertia	Momento de inercia del robot
Fmax	Fuerza máxima aplicable
Tmax	Torque máximo aplicable
accelerationMax	Aceleración lineal máxima
angularAccelerationMax	Aceleración angular máxima
linealSpeedMax	Velocidad lineal máxima que puede alcanzar el robot
angularSpeedMax	Velocidad angular máxima que puede alcanzar el robot
Parámetros de *A-Frame*	
restitution	Conservación de la energía cinética en un choque entre partículas
gravity	Gravedad
friction	Fricción (rozamiento estático y dinámico)
linearDamping	Amortiguación lineal (rozamiento dinámico en el movimiento lineal)
angularDamping	Amortiguación angular (rozamiento dinámico en el movimiento angular)

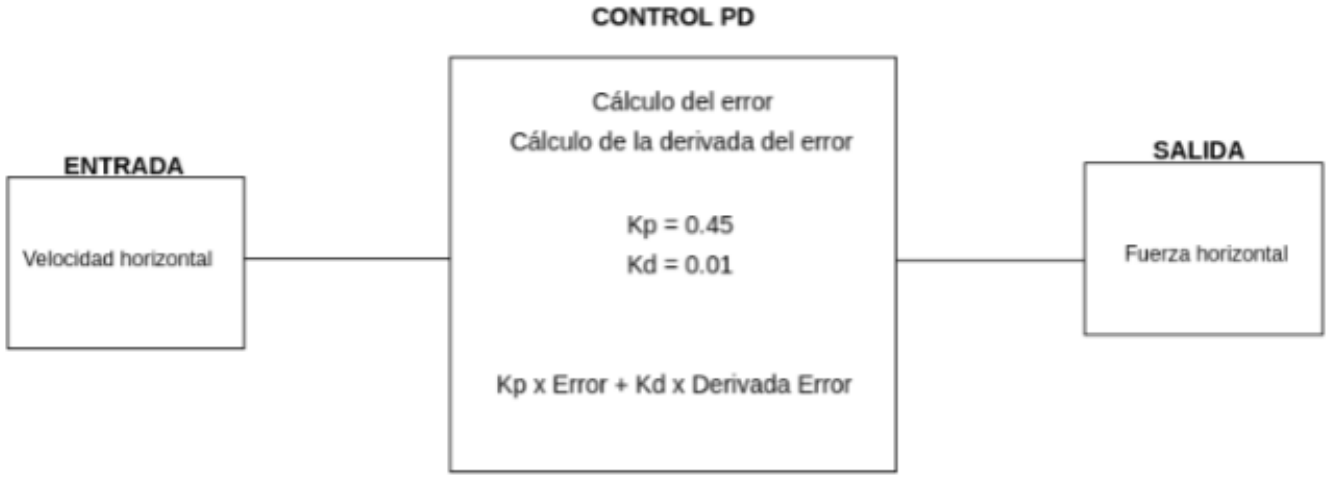
2. MODELO DE FUERZAS

- Es el núcleo del motor de físicas.
- A partir de la definición de la masa y el momento de inercia del robot, se calcula la aceleración o torque a aplicar.
- Parámetros a definir:
  - Fuerza máxima.
  - Torque máximo.
  - Velocidad lineal máxima.
  - Velocidad angular máxima.
  - Masa.
  - Momento de inercia.

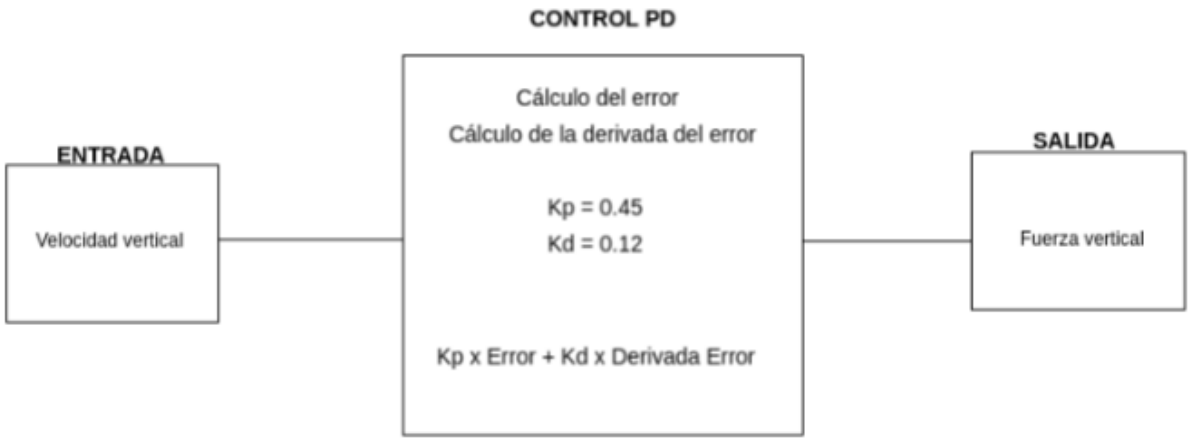
3. CONTROLADOR PD

- Se encarga de la traducción de las velocidades deseadas que le llegan al motor complementario en cada momento del cerebro a la fuerza autónoma a aplicar al robot.

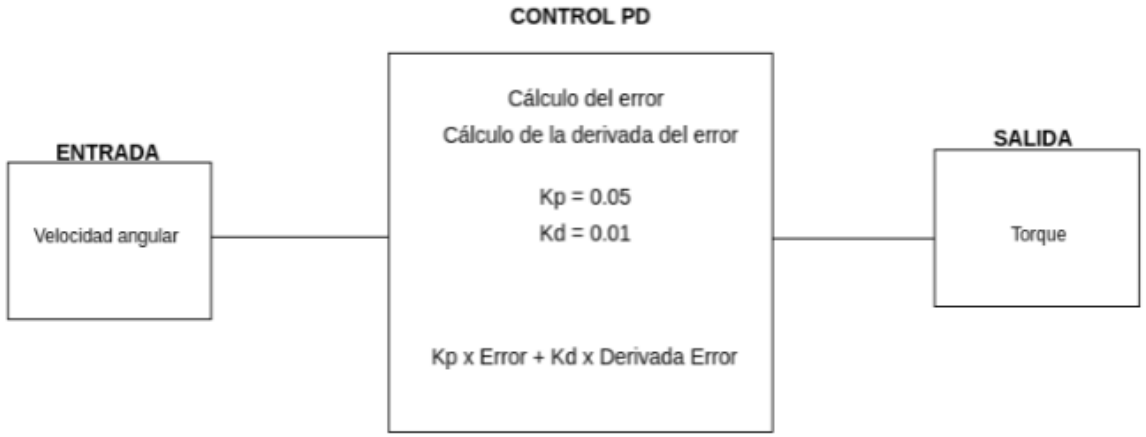
1. Controlador PD en velocidad del plano horizontal



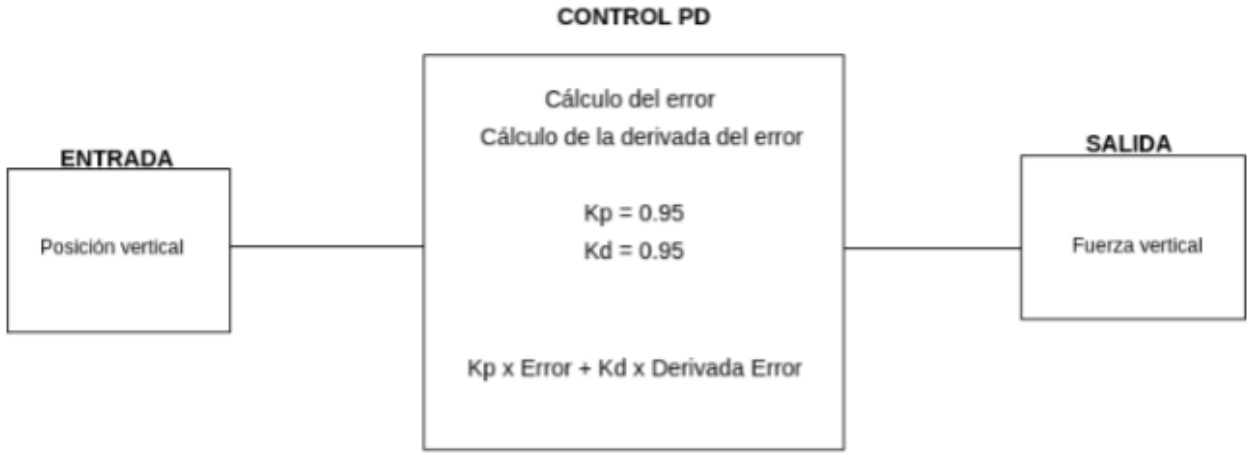
2. Controlador PD en velocidad del eje vertical



3. Controlador PD en velocidad angular horizontal (yaw)



4. Controlador PD en posición para la altura



4. TIMING

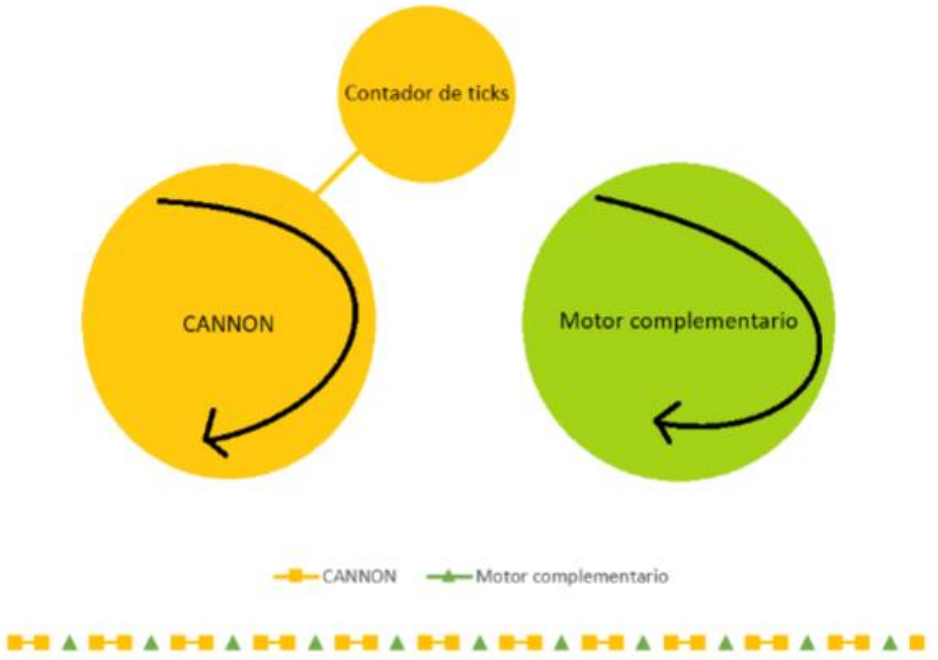
Coexistencia de motores ⟶ Necesidad de combinarlos

```
Aceleración autónoma = iteracionesCANNON x aceleración calculada
```

- ¿Timing del nuevo motor complementario? 20 ms

```
setTimeout(this.auxiliaryPhysics.bind(this), 20);
```

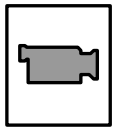
- ¿Timing de *CANNON*? Desconocido.



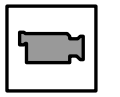
Validación experimental

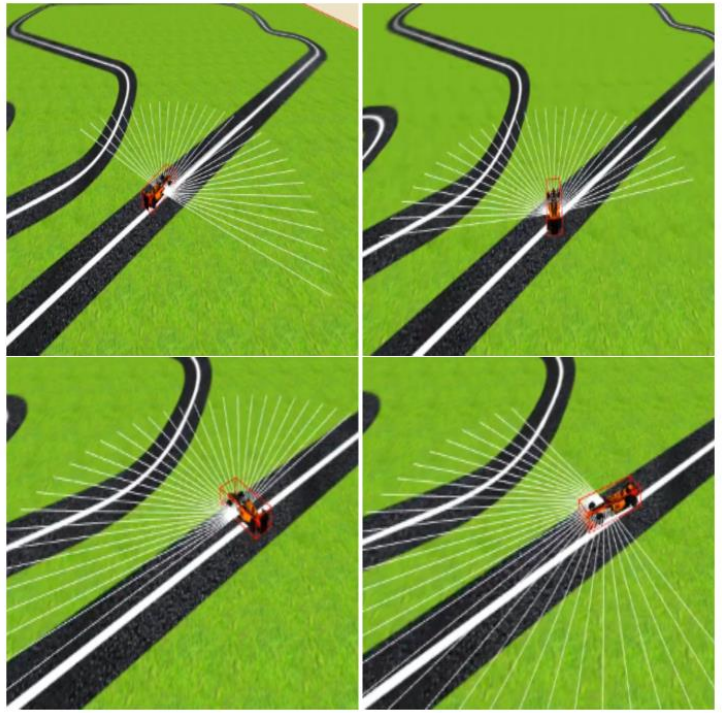
Simulación realista de robots terrestres

mBot subiendo una rampa con una fuerza máxima insuficiente

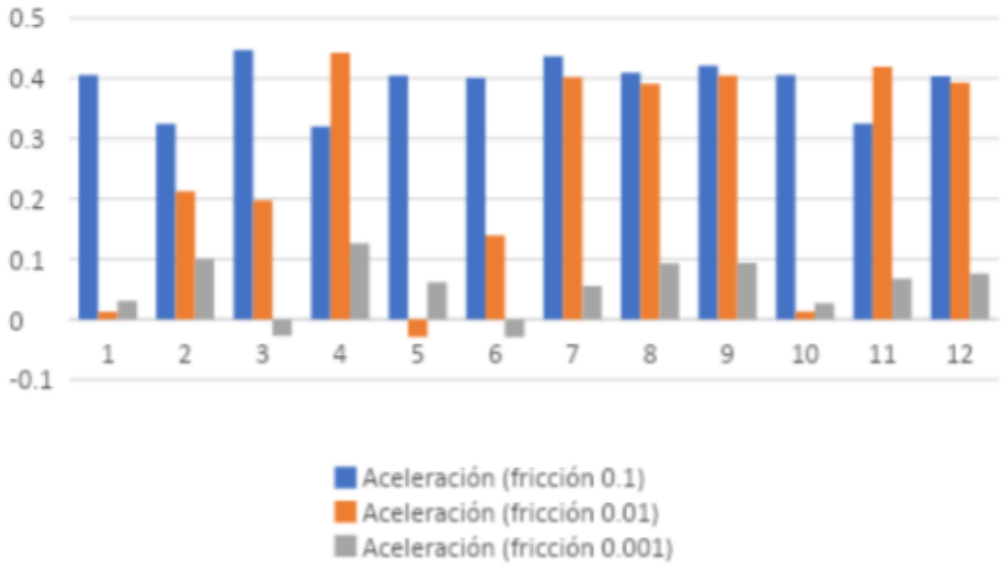


Giro de 90º es un escenario con una fricción muy baja (pista de hielo)



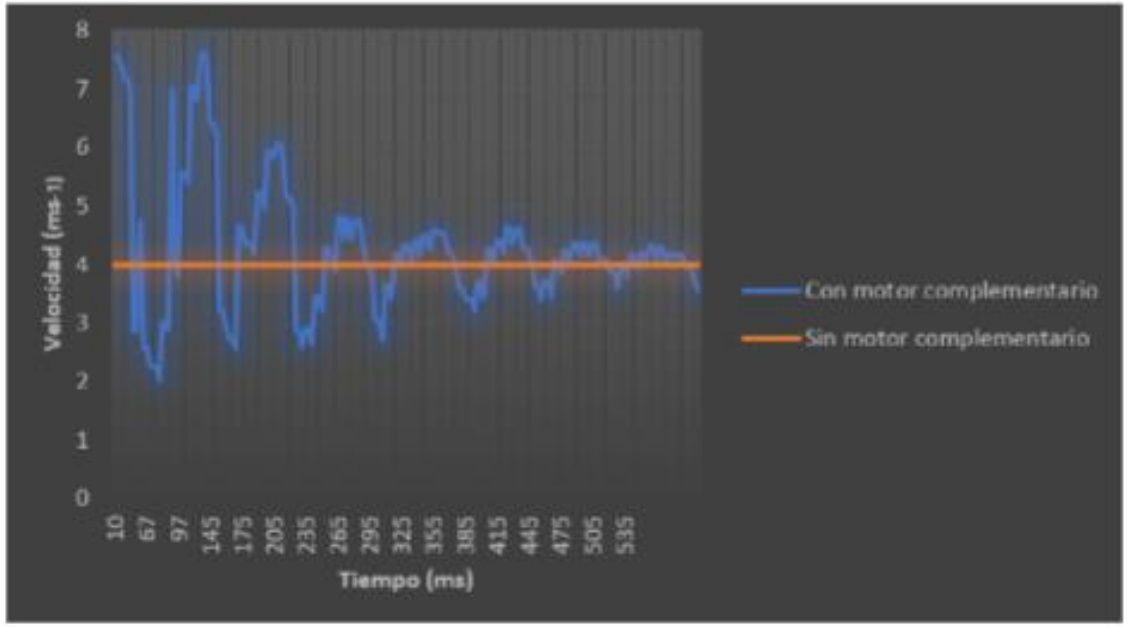


Variación de la aceleración en función de la fricción

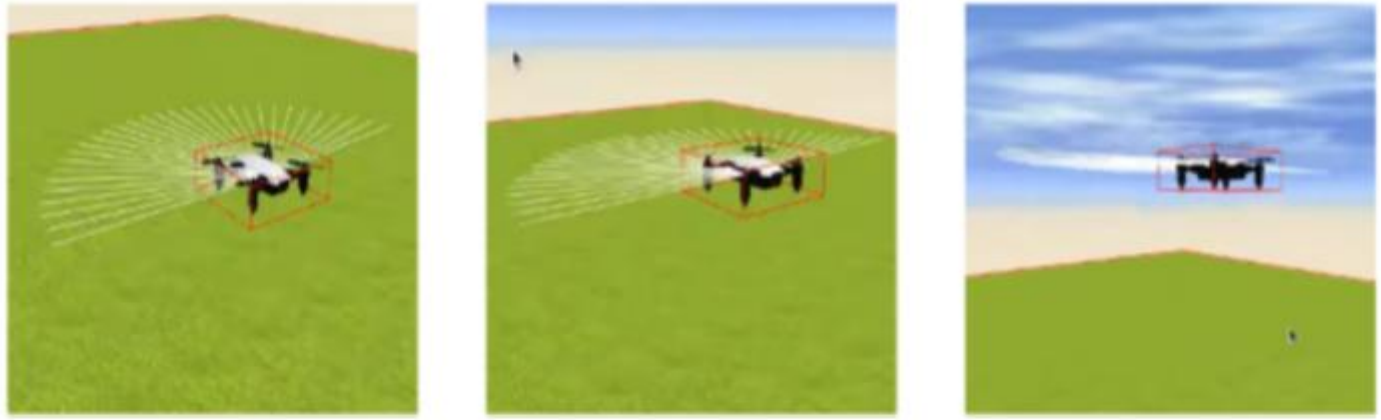


Controlador PD en velocidad del plano horizontal

Tiempo – Velocidad



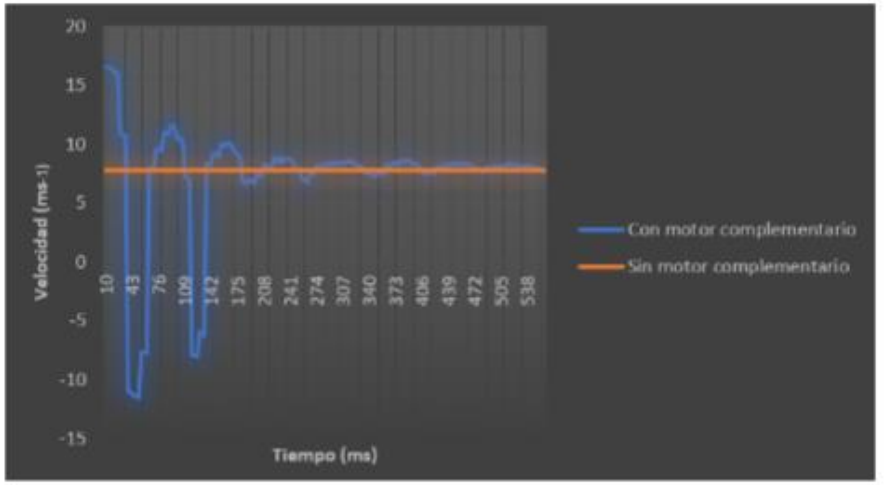
Simulación realista de drones



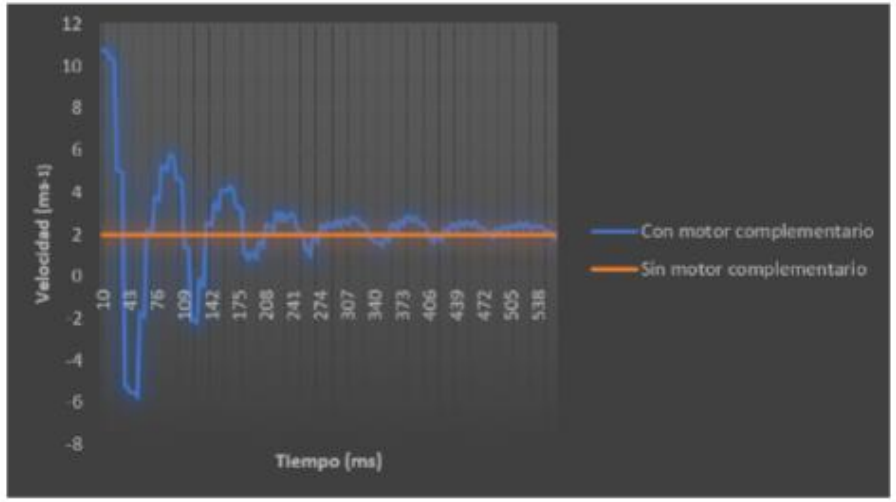
(a) Despegue del drone Tello de 1 Kg

(b) Despegue del drone Tello de 100 Kg

Controlador PD en posición para la altura
Tiempo – posición



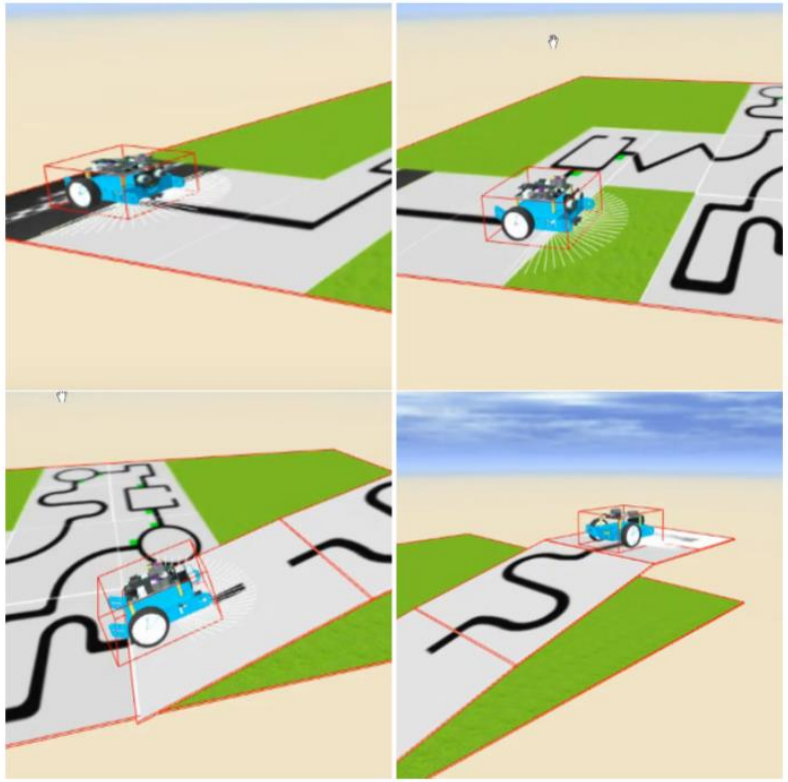
Controlador PD en velocidad del eje vertical
Tiempo – velocidad



Nuevos ejercicios con físicas realistas

Sigue-Líneas con rampa

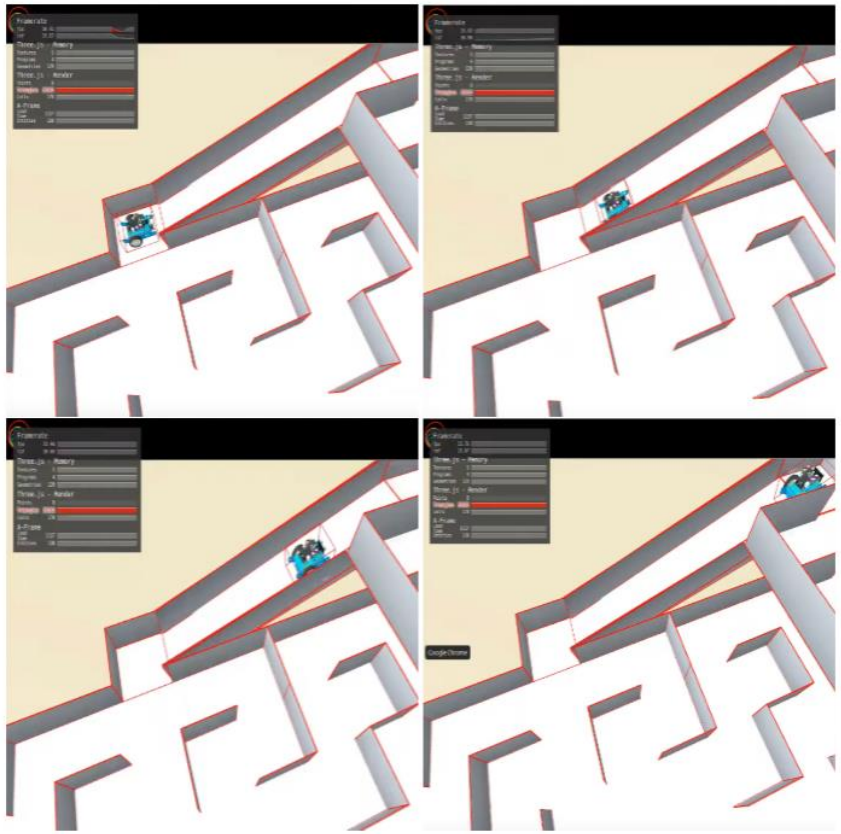
El ejercicio aprovecha las ventajas del motor de físicas complementario en la subida de la rampa.



Parámetros del modelo de fuerzas	
mass	1
inertia	1.3
Fmax	10
Tmax	1
accelerationMax	10
angularAccelerationMax	0.77
linealSpeedMax	10
angularSpeedMax	5
Parámetros de *A-Frame*	
restitution	0.3
gravity	-9.8
friction	0.00003
linearDamping	-1.3
angularDamping	-1.3

Laberinto 3D para mBot

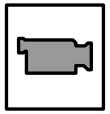
El ejercicio aprovecha las ventajas del motor de físicas complementario en la subida de la rampa.



Parámetros del modelo de fuerzas	
mass	1
inertia	1.3
Fmax	10
Tmax	1
accelerationMax	10
angularAccelerationMax	0.77
linealSpeedMax	10
angularSpeedMax	5
Parámetros de *A-Frame*	
restitution	0.3
gravity	-9.8
friction	0.0005
linearDamping	-1.3
angularDamping	-1.3

Laberinto para drone

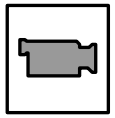
- Con señalización y sin señalización
- Aprovecha el motor de físicas complementario durante el vuelo del drone (controlador PD en velocidad) y mientras que el drone permanece quieto a una cierta altura (controlador PD en posición)

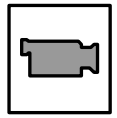


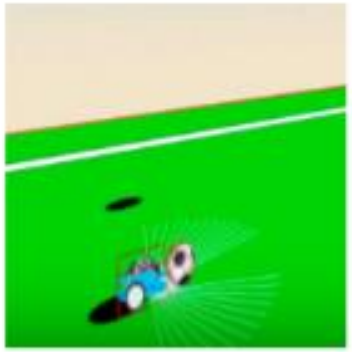
Parámetros del modelo de fuerzas	
mass	1
inertia	1.3
Fmax	10
Tmax	1
accelerationMax	10
angularAccelerationMax	0.77
linealSpeedMax	10
angularSpeedMax	5
Parámetros de *A-Frame*	
restitution	0.3
gravity	-9.8
friction	0.0000001
linearDamping	0.01
angularDamping	0.01

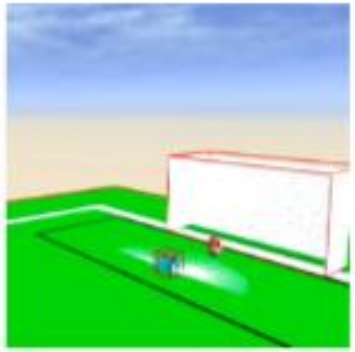
Fútbol competitivo

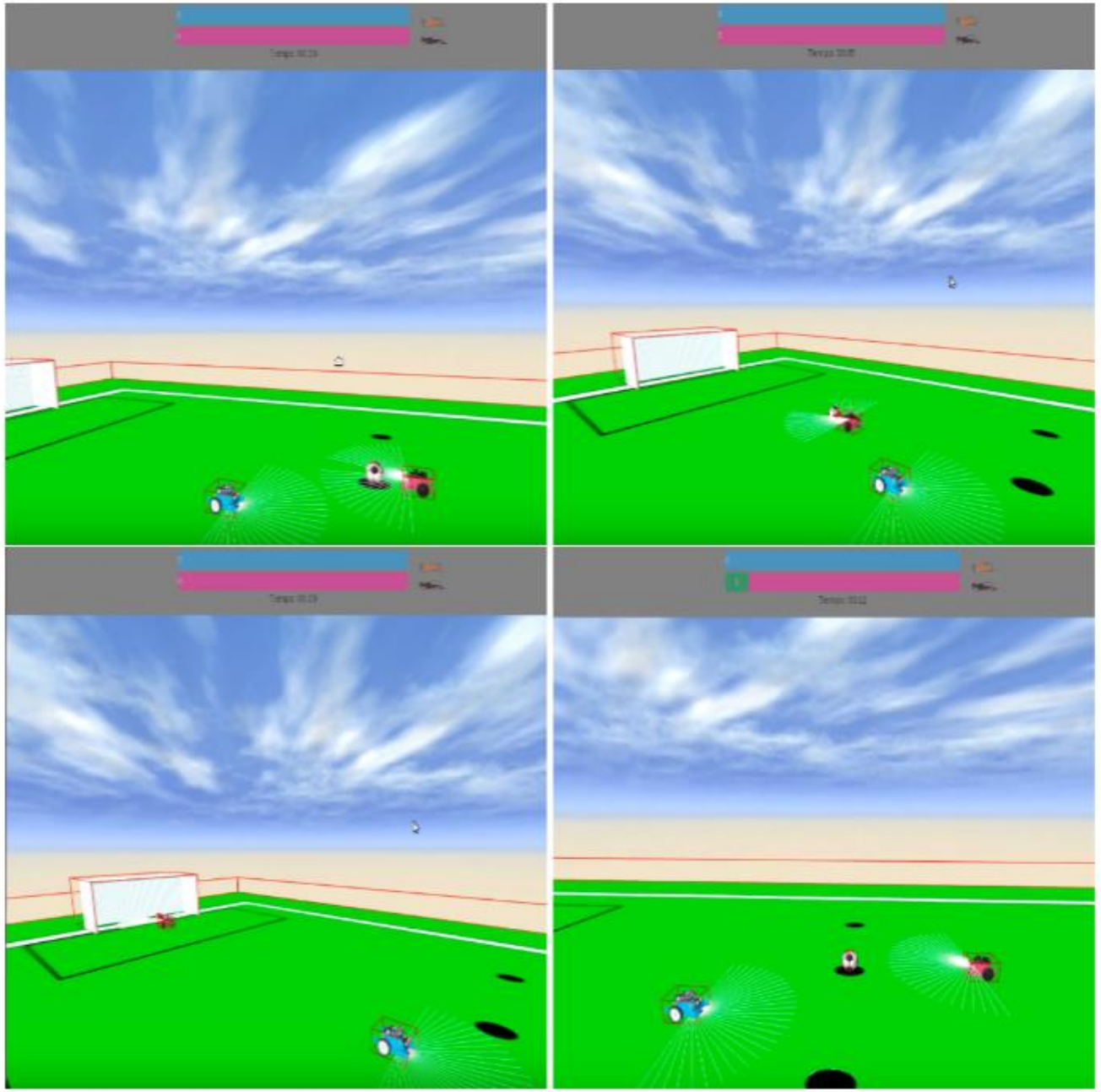
- Ejercicio competitivo uno contra uno
- Cuenta con un evaluador
- Saca provecho de las nuevas físicas tanto al golpear el balón (colisión) como cuando el balón rueda por el suelo (fricción)

Sin el nuevo motor complementario 

Con el nuevo motor complementario 







Parámetros del modelo de fuerzas	
mass	1
inertia	1.3
Fmax	10
Tmax	1
accelerationMax	10
angularAccelerationMax	0.77
linealSpeedMax	10
angularSpeedMax	5
Parámetros de *A-Frame*	
restitution	0.5
gravity	-9.8
friction	0.0005
linearDamping	0.01
angularDamping	0.01

Conclusiones

Conclusiones

- Crear un motor de físicas mejorado para *WebSim* que estuviera basado en tecnologías web y que permitiese dotar al simulador robótico de unas físicas más realista ✓
- Materialización de robots con distinta masa y que recreen un movimiento autónomo realista, con una aceleración máxima limitada y capacidad de control acotada ✓
- Coexistencia con el motor por defecto *CANNON y* que no requiera la modificación de su código fuente ✓
- Crear varios ejercicios en la plataforma educativa *Kibotics* que sacasen partido del nuevo motor de f físicas y fueran más atractivos para los niños ✓

Líneas futuras

- Creación de ejercicios competitivos para cuatro jugadores 
- Exploración del nuevo motor de físicas *ammo.js* y extender la implementación a este motor 
- Adición de efectos de sonido a los ejercicios