Creació d'un bessó digital d'un robot

Raúl Avellaneda Caballero

Resum– Una de les principals tendències tecnològiques durant els darrers anys es troba en la creació de bessons digitals, que son una replica virtual d'un producte al qual s'incorporen dades que poden ser captades a través de sensors.
Paraules clau– Bessó digital, CoppeliaSim, robot, simulació, parametrització, recollida de dades

Abstract– One of the main technological trends during the last years is the creation of digital bessons, which are a virtual replica of a product to which are incorporated data that can be captured through sensors. .
Keywords– Digital twin, CoppeliaSim, robot, simulations, parameterization, data recolection

1 Introducció - Context del treball

Un bessó digital és un model virtual dissenyat per simular el comportament amb precisió d'un objecte físic. L'objecte en estudi produeix diverses dades sobre diferents aspectes del rendiment del objecte com poden ser, per exemple, la temperatura, la velocitat a la que es desplaça o l'energia que consumeix. Després, aquestes dades es transmeten a un sistema de processament per aplicar-les al bessó digital.

Amb aquestes dades, el bessó digital es pot utilitzar per executar simulacions, estudiar problemes de rendiment que pugui tenir i generar possibles millores, amb l'objectiu de generar informació valuosa que després s'aplica al objecte físic original [1].

El principal avantatge de l'ús de bessons digitals és la possibilitat de crear models analítics amb els quals es puguin predir el comportament de l'objecte en qüestió davant d'uns possibles canvis.

Es desenvolupen amb tres principals objectius [2]:

- Crear un prototip de bessó digital abans de crear el producte final per veure com es comportaria.
- Crear una instància bessona digital una vegada que s'hagi fabricat el producte, es pot fer servir un bessó digital per fer proves en diferents condicions o escenaris.
- Recopilar informació de diferents proves per tal de determinar quines capacitats té el producte.

Actualment, on més s'utilitzen aquests bessons digitals són en vàries industries amb diferents aplicacions i propòsits. Per exemple, en fàbriques s'utilitza per agilitzar la producció i reduir possibles errors. Sobretot on més està emergent és a la industria 4.0 (indústria intel·ligent), una industria on gairebé tot està automatitzat, connectat i globalitzat.

Avui en dia s'utilitzen les capacitats dels bessons digitals de moltes maneres diferents. En els sectors automobilístic [3] i aviació [4] s'han convertit en eines totalment essencials per dissenyar i fabricar productes nous. En el sector de salut [5], investigadors cardiovasculars creen bessons digitals de cors humans per a diagnòstics clínics, educació i entreteniment.

Els bessons digitals poden tenir moltes formes, però l'essència és que tots capturen i utilitzen dades del món físic per fer la simulació de la part física [6].

Una investigació recent de MarketsandMarket [7] suggereix que els esforços per crear-los està creixent: el mercat de bessons digitals tenia un volum de 3.1 bilions de dòlars en 2020 i està previst que arribi a 48.2 bilions de dòlars en el 2026, en part per la demanda creixent en el sector sanitari durant la pandèmia.

En general, les empreses s'estan adaptant molt ràpidament a la utilització dels bessons digitals, encara que és un procés que no es pot completar de la nit al dia, és a dir, la implementació de bessons digitals és una transformació completa de l'empresa. Les companyies que vulguin apostar per aquesta tecnologia hauran de resoldre reptes com [8]:

1. Monitoritzar i digitalitzar processos de la indústria que exigeixen una estricta arquitectura d'integració i automatització industrial.
2. Augmentar la capacitat dels sistemes d'emmagatzemament, gestió i anàlisis de dades actuals per gestionar el volum requerit pels bessons digitals.
3. Començar a fer representacions digitals a la gestió de múltiples copies digitals simultànies amb major capacitat per avaluar escenaris alternatius.

• E-mail de contacte: raul.avellaneda@autonoma.cat
• Mención: Enginyeria de Computadors
• Treball tutoritzat per: Lluís Ribas-Xirgo (Microelectrònica i Sistemes Electrònics)
• Curs 2021/2022

1.1 Objectius

El principal objectiu d'aquest projecte és aconseguir un bessó digital que ens permeti, a partir de un conjunt de dades recollides del robot físic, simular amb la màxima precisió possible el comportament real del robot físic.

En el cas particular d'aquest projecte, el model digital s'ajustarà al comportament del robot físic mitjançant un paràmetre per ajustar la velocitat mitjana.

El cas ideal és que el robot, el primer cop que realitza una instrucció no ajustarà la velocitat perquè no té dades suficients per ajustar-la, però guardarà la informació que s'ha obtingut amb el robot real. Per tant, quan es torni a executar la mateixa instrucció, el bessó digital hauria d'agafar la informació emmagatzemada del robot real de les vegades que ha executat la instrucció en qüestió per poder ajustar la velocitat a la del robot real.

El projecte parteix d'un model existent que es fa servir en les assignatures de Sistemes encastats de les Enginyeries Informàtica i de Dades, que inclou eines de registre de dades i un model de robot diferent del que es farà servir en aquest treball.

Així doncs, aquest projecte té els objectius específics següents:

1. **Creació de la base de dades** on es guardaran els paràmetres recollits de les diferents proves del robot físic.

2. **Configuració del model digital** per a que funcioni, de la manera més precisa respecte al robot físic, amb les dades recollides de la base de dades.

3. **Creació del model digital**: es parteix d'un model digital el qual s'haurà de modificar diferents parts per a que funcioni com el robot físic.

    - **Canviar servomotors** del model digital inicial a motors lineals. Aquestos es comporten de manera diferent, per tant, s'haurà de reconfigurar el model per a que torni a funcionar amb els nous motors.
    - **Canviar el sensor sonar per un sensor lidar.** La principal diferència entre aquests sensors és que el sensor lidar funciona a una velocitat molt superior perquè no té la mateixa limitació física que el sensor d'ultrasons, en altres paraules, la velocitat de la llum es molt més ràpida que la velocitat del so, factor que permet al sensor lidar agafar mostres més ràpidament.
    - **Canviar l'estructura** o afegir textures al model digital per a que s'assembli al màxim al robot físic.

1.2 Metodologia

La metodologia que s'ha escollit per poder fer el projecte de manera clara i precisa és *Kanban* [9]. Aquesta metodologia es considerada *agile*, és a dir, fomenta la planificació adaptativa, l'entrega continua i la millora contínua.

Consisteix en un tauler virtual amb diferents columnes que representen les etapes del treball, on hi ha diferents tasques del projecte (anomenades 'targetes kanban') que avancen a través de les diferents etapes del treball.

Per poder dur a terme aquesta metodologia, s'utilitzara l'eina Trello[10], que ens permet crear aquest tauler virtual amb 3 columnes, una per objectius que encara no s'han realitzat, una per objectius els quals estem treballant però no s'han finalitzat i una última per objectius finalitzats.

A més, es programaran reunions conjuntament amb el tutor del treball per fer un correcte seguiment i per comprovar que els resultats es van assolint.

2 Estat de l'art

Actualment, hi ha eines molt potents a l'hora de crear bessons digitals. Una de les més potents és l'Ansys Twin Builder[11], que és una solució oberta que permet crear, validar i implementar un bessó digital, reduint potencialment a la meitat el temps requerit per crear un model de producte precís.

Alternativament, hi ha CoppeliaSim, un simulador molt personalitzable on gairebé cada iteració de una simulació es pot definir per l'usuari mateix [12]. Això ens permet crear bessons digitals gràcies a que es basa en una arquitectura distribuïda de control, on cada objecte de la simulació pot ser controlat per un script, una API remota o una solució personalitzada. Al nostre treball utilitzarem aquest entorn per fer el bessó digital d'un robot. Aquest robot és capaç de moure's mitjançant uns motors DC i un sensor lidar que detecta obstacles i calcula la distància entre aquests i el mateix robot.
En el nostre treball, utilitzarem una sèrie de scripts per a fer funcionar tant el robot virtual com el real. La diferència està en que el model virtual estarà dins del propi Coppelia Sim, i el model real s'haurà de connectar per a poder executar-los. Els scripts a Coppelia Sim es caracteritzen per tenir una estructura predefinida (Fig. 1).

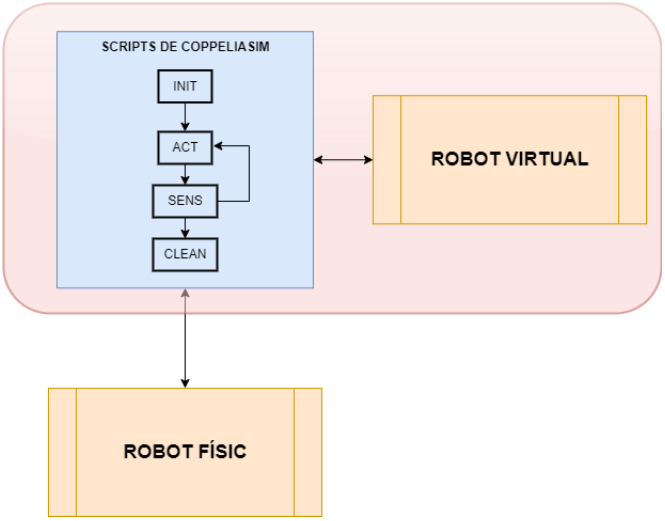


Fig. 1: Estructura del model de simulació

Els scripts que s'utilitzaran per fer el model digital haurà de respectar aquesta estructura també. Per tant, constaràn de cuatre parts diferents que s'encapsulen dins d'unes funcions les quals el seu nom comencen amb 'sysCall_':

1. **'function sysCall_init()':** son accions a realitzar abans d'iniciar la simulació, com pot ser per exemple obrir una consola per a mostrar informació general, definir funcions auxiliars per treure informació en la consola o obrir 'tubes' requerits per a la comunicació entre els diferents scripts.

2. **'function sysCall_actuation()':** son accions que es realitzen en la fase d'actuació de cada iteració de la simulació. Per exemple, per aplicar els valors de sortida de la màquina d'estats als dispositius actuadors del sistema com poden ser els motors.

3. **'function sysCall_sensing()':** son accions a realitzar en la fase de sensat en cada iteració de la simulació. Per exemple, la lectura de les dades d'entrada o l'execució del comportament corresponent al estat actual de la màquina d'estats.

4. **'function sysCall_cleanup()':** son accions a realitzar al finalitzar la simulació com pot ser tancar tots els 'tubes' de comunicació.

En relació al model de simulació pel qual partim inicialment, es tracta d'un robot virtual amb una arquitectura del software dividida en diferents capes (Fig. 2). Cada capa realitza una funció diferent del robot i es comuniquen entre elles amb missatges.

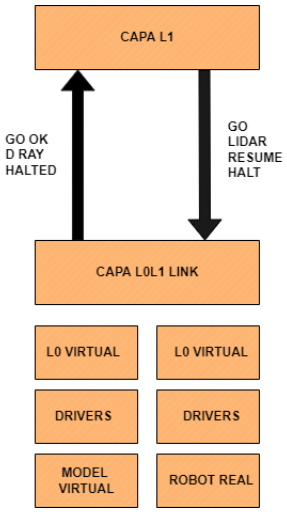


Fig. 2: Estructura del model de simulació

La capa L1 és la capa superior d'aquesta arquitectura. Aquesta implementa una màquina d'estats Mealy per realitzar la funció de seguidor de camins (Fig. 3), la qual serà utilitzada per l'emissió de instruccions.

En aquesta funció es defineix 4 possibles comandes que el robot ha de poder emetre a través d'una interfície d'usuari:

1. **Instrucció 'GO':** és l'ordre que el usuari introdueix per donar l'ordre de girar el robot i de fer-ho avançar.

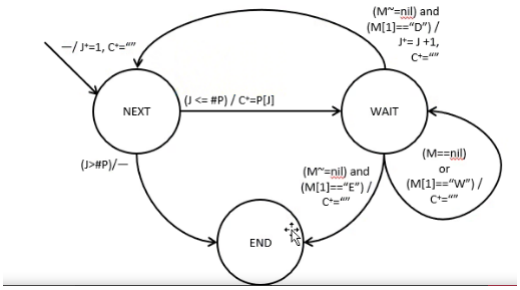


Fig. 3: Màquina d'estats de la capa L1

Els paràmetres que cal indicar són el angle, amb un rang de 0 a 90, i la distància, amb un rang de 0 a 255.

2. **Instrucció "SONAR" (Lidar):** aquesta s'encarrega de donar l'ordre per fer servir el sensor lidar per a la detecció de possibles obstacles.

3. **Instrucció "Resume":** té com a objectiu que el robot torni a funcionar després d'haver-se aturat.

4. **Instrucció "Halt":** atura la instrucció que estigui fent el robot en el moment en que l'usuari utilitza aquesta.

Com s'ha comentat abans, les diferents capes del robot es comuniquen entre elles. La capa L1 rep un missatge de L0 quan s'ha executat bé una instrucció o ha sorgit qualsevol error durant l'execució. També envia un missatge a la capa L0 informant de la instrucció que es vol executar, especificant 3 números en aquest missatge, el primer és per indicar el tipus de instrucció, el segon seria el angle de gir i el tercer és la distància de recorregut (aquests dos últims si és una instrucció GO).

La capa 'L1L0 Link' situada entre la L1 i L0 s'encarrega de la sincronització entre el model virtual i el model real connectat. Funciona amb una màquina d'estats on espera a rebre un missatge, tant de la L0 virtual com de la L0 del robot real.

Aquest missatge acostuma a ser un missatge de que la instrucció de tipus "GO" que s'havia demanat ha sortit bé, i en aquest cas mirarà la diferència que hi ha entre el missatge rebut de la capa L0 virtual i la capa L0 real. Si hi ha poca diferència, la capa "L0L1 link" mostrarà un missatge que els robot virtual va en un temps bo respecte al virtual. Si es rep abans el missatge de la L0 virtual es mostra un missatge de que el robot virtual va avançat al real, i si es en últim lloc mostrarà el missatge de que el robot virtual va per enrere del real. A més, en aquesta capa s'ha afegit la funció de 'logger', que s'encarrega d'obrir un fitxer en format 'csv' per a guardar les dades que rep tant de la capa L0 del robot virtual com de la L0 del robot físic. Aquesta funció permetrà poder crear el bessó digital amb les dades emmagatzemades.

Amb aquesta informació serà la que s'utilitzarà per poder ajustar la velocitat del robot virtual per a que s'adapti, utilitzant un paràmetre nou, que s'anomena 'Speed Ratio'.

A la següent capa ens trobem una divisió en dos, és la capa L0 del robot virtual i del real que actuen de manera separada, però que compleixen la mateixa funció. La funció de la capa L0 es la de rebre un missatge de la capa L1 amb

la comanda que toca i poder-la executar. Funciona amb dues màquines de estats, una per fer funcionar el sensor lidar del robot, tant per la detecció com per al gir del servo motor que té aquest sensor. Després té una segona màquina d'estats que s'encarrega de fer funcionar les instruccions que arriben de la capa L1 i de retornar un missatge amb la informació pertinent.

Com a última capa a nivell de software ens trobem la capa dels controladors, tant del robot virtual com del real. En aquesta capa es configuren els diferents motors i sensors dels robots. En aquesta capa, per exemple, es configura la velocitats del motors DC o l'angle de gir que té el sensor lidar.

L'última capa de l'arquitectura es tracta de l'estructura física de cada robot. Inicialment l'estructura del robot virtual es totalment diferent a l'estructura que té el robot físic.

3 DESENVOLUPAMENT DEL TREBALL

El primer pas és, tot partint de la versió inicial d'un robot virtual, canviar els servomotors pels motors DC, i el sensor sonar per un sensor lidar que és el que té el robot real.

Un cop ja s'hagi realitzat aquests canvis, ja tindrem el nostre model virtual del robot físic, cosa que no significa que sigui el bessó digital que busquem. Per fer el bessó digital es farà un experiment, que consisteix a recollir dades del recorregut del robot real i del virtual per comparar el temps del recorregut amb l'objectiu de ajustar la velocitat del robot virtual per convertir-ho en el bessó digital que busquem.

Com a resultat final, tenim per una banda, una sèrie de paràmetres en una base de dades que s'han recollit de diferents proves utilitzant el robot real, i per altra banda, tenim la simulació del robot virtual utilitzant aquestes dades.

3.1 Recollida i emmagatzematge de les dades

En relació a l'obtenció de les dades, els nostres scripts de CoppeliaSim que implementen el robot virtual s'encarregaran de fer la comunicació amb el robot físic per a la captació de les dades. Per cada instrucció que executin els robots es guardarà:

1. **El temps d'execució:** es guarda informació del temps en el qual s'ha executat les instruccions especificades.
2. **L'angle de gir:** es tracta d'un número de 0 a 90 que és l'angle de gir que han fet els robots.
3. **Distància de recorregut:** és un número de 0 a 255 que defineix la distància que haurà de recórrer els robots després de fer el gir especificat.
4. **El temps d'execució de les instruccions del robot virtual:** es guarda el que ha trigat el robot virtual en realitzar la instrucció de gir i d'avançament en mil·lisegons.
5. **El temps d'execució del robot físic respecte al virtual:** es guarda quin dels dos robots ha acabat abans d'executar les instruccions, i a més, es guarda la diferència de temps que ha hagut amb el virtual.
6. **Missatge:** es guarda un missatge per qualsevol error que pugui aparèixer a les diferents capes del robot, o bé, per a indicar que tot ha anat bé.

Per a la desa de la informació en les simulacions, inicialment es volia connectar una base de dades mySQL directament amb el script LUA de CoppeliaSim. Això no és possible en aquest entorn, per tant vam haver de buscar altres opcions. Les diferents opcions que s'han trobat són les següents:

- **Utilització d'una API remota:** Les API son mecanismes que permeten a dos components software comunicar-se entre ells mateixos a partir d'un conjunt de protocols i definicions. [13] En Coppelia Sim es dona dues possibilitats amb aquestes API[14]:
  - **'Continuous legacy remote API server service':** la aplicació API remota llegirà un fitxer de configuració per inicialitzar els serveis necessaris. Amb aquest mètode, la API remota s'executarà encara que la simulació no estigui executant-se.
  - **'Temporary legacy remote API server service':** és el mètode més utilitzat per a inicialitzar les API remotes. L'usuari controla quan el servei s'inicia o es para a través d'un script de simulació. De totes maneres, el servei es para al final de la simulació si encara segueix executant-se.

  En aquest cas, tindríem vàries opcions per al tractament de les dades.

  1. **Programació d'un software extern:** consistiria en programar una aplicació amb la capacitat de connectar-se a la simulació per rebre les dades i emmagatzemar-les. La principal avantatge seria que es podria processar les dades per a crear el paràmetre del nostre bessó digital, però la programació i la implementació és més complexa.
  2. **Configuració d'una base de dades:** es tracta de configurar una base de dades al nostre host per tal de poder utilitzar les API remotes per connectar-se al script. La principal avantatge és que és més fàcil d'implementar, però es depèn de que el servei de la base de dades funcioni correctament per a la visualització de les dades, a més, que no es poden tractar per a la creació del paràmetre.

- **Creació d'un plugin:** Una altra opció que tenim seria la creació d'un plugin per a Coppelia Sim[15]. Els plugins son complements que afegeixen funcionalitats extres o millores als programes, és a dir, sumen alguna funcionalitat que no té el programa base i funcionen com afegits però no per si mateixos [16]. Per tant, hauríem de crear un plugin que s'encarregues de emmagatzemar la informació de tota les simulacions realitzades per a la posterior parametrització. El principal problema és que la seva implementació és molt complexa, però ens permetria emmagatzemar i processar dades dins del propi Coppelia Sim.
- **Creació d'un arxiu:** És el mètode utilitzat en el nostre treball, consisteix en generar les dades en la simulació

i guardar-les en un arxiu en format '.csv'. La principal avantatge d'aquesta opció és que és molt més visual i fàcil d'implementar al script, a més que no depèn d'un software extern per a l'emmagatzematge de les dades de cada simulació.

3.2 Canvi dels sensors

Una de les primeres tasques que s'han hagut de realitzat és el canvi del sensor sonar que hi havia a la versió inicial del bessó digital per un sensor lidar.

Els sensors sonars utilitzen polsos per detectar distàncies fent ús de ones de só. Aquestes ones reflecteixen en els objectes i tornen al sensor, amb aquesta informació es calcula del obstacle detectat respecte al sensor. El problema d'aquestos que no tenen els lidar és que son més lent degut a la limitació física de la velocitat del só. En canvi, els sensors lidars s'utilitzen polsos de llum per determinar la distància respecte al sensor. Són més precisos que els sonar, i a més, els polsos viatgen a la velocitat de la llum, per tant és molt més ràpid.

El que hem trobat al nostre treball és que aquest sensor sonar ja funcionava com un sensor lidar en el nostre bessó digital, com es pot veure a la capa L0 del software del bessó, es comporta com un sensor lidar amb l'ajuda d'una taula de escaneig on té una sèrie de valors normalitzats per a les diferents mides del sensor per a la detecció d'objectes.

A més, trobem en els diferents components del bessó en el nostre entorn de treball, que aquest sensor es de tipus raig, que és el que s'utilitza principalment en els sensor lidars a diferència dels sonars que utilitzen un tipus con.

Per tant, en la part de canviar el sensor sonar per un lidar no s'han hagut de fer canvis significatius respecte a la versió inicial del bessó.

3.3 Canvi dels motors

En relació als servomotors del bessó digital cal fer una sèrie de canvis per a que funcionin com els motors lineals del robot físic.

En primer lloc, s'haurà d'ajustar els paràmetres dels motors amb els mateixos, és a dir, s'haurà de configurar el parell del motor per a que vagi a la mateixa velocitat que el robot físic.

A nivell de software, en els controladors hi han dues taules per fer la conversió de la velocitat de gir del motor. Per a que funcioni com a un motor lineal també s'haurà de canviar el valor de la segona taula amb les revolucions per minuts del robot real. (Fig. 4)

Per poder fer el canvi d'aquesta taula, s'ha fet un prova amb el robot real on aquest recorria 88cm en línia recta i després calculem les revolucions per minut de la velocitat lineal escollida. S'ha decidit que la distància sigués de 88cm perquè justament el robot físic fa 7 revolucions, per tant es més fàcil i precís a l'hora de calcular les revolucions per minut.

A més, com s'ha comentat abans, s'haurà de crear un nou paràmetre als controladors del motor per a que es vagi ajustant a la velocitat del robot físic segons les dades obtingudes anteriorment.

```
PW2speed = function( pw )
    local PWTable = {
        18, 24, 28, 34, 42, 50, 54, 62, 70, 76,
        82, 86, 90, 94, 98, 104, 110, 118, 126, 130,
        138, 146, 152, 156, 162 } -- microseconds
    local speedTable = {
         -40,  -38,  -36,  -32,  -29,  -23,  -19,  -16,  -12,  -10,
          -8,   -5,    0,    5,    8,   10,   12,   16,   19,   23,
          29,   32,   36,   38, 40 } -- rpm
```

Fig. 4: Taula de velocitats dels motors

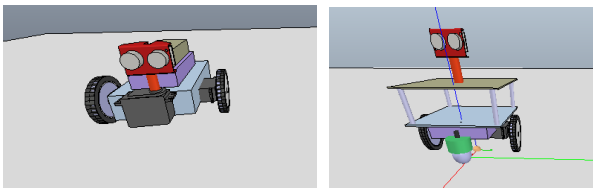
Després de fer moltes proves amb diferents recorreguts i valors del 'speed ratio' s'ha aconseguit ajustar la velocitat del robot virtual a la del robot real utilitzant un valor de 1.15 al paràmetre per a les superfícies totalment planes.

3.4 Canvi de l'estructura física

Les modificacions en l'estructura del bessó digital són essencials. Es parteix d'una versió força diferent al robot real, amb mides distintes i un altre distribució de les diferents parts del robot.

El que hem fet ha sigut redistribuir aquestes parts segons el robot físic i fer-ho a escala real. El més important és aconseguir que les rodes i la distància entre aquestes al nostre bessó digital sigui el màxim precís que a la realitat, ja que d'això depèn bona part la velocitat del robot.

Un altre component molt important a l'hora que sigui el més precís possible es el sensor lidar del bessó digital, aquest també l'hem canviat la mida i la posició per a que funcioni el més semblant possible al sensor del robot físic. Com es veu a la següent imatge el sensor. s'ha posat encara més endarrere, per tant variables com la distància de seguretat s'haurà de canviar per fer-la més gran (Fig. 5).



(a) Robot digital original (b) Robot digital actual

Fig. 5: Comparació entre les diferents versions

4 Resultats obtinguts

Un cop s'ha acabat el bessó digital s'ha procedit a fer una sèrie de diferents proves per mostrar que realment s'adapta adequadament i funciona força semblant com el robot real.

Hi ha un factor que cal tenir en compte, que és l'existència d'una espera programada a l'hora d'executar una comanda de gir i seguidament una altra per fer avançar el robot. Això provoca, com es veurà mes endavant, que el robot virtual aconsegueix estalviar temps respecte al robot físic. En altres paraules, encara que el model del robot virtual i real siguin similars, la relació de velocitats serà molt diferent per aquesta espera programada.

4.1 Dades recollides

S'ha fet una sèrie de proves en diferents entorns per tal de recollir i fer un anàlisis de les dades. En primer lloc s'ha fet diverses execucions amb el robot real en una superfície

totalment plana amb diferents distàncies a recórrer per poder fer un estudi de com evoluciona el 'speed ratio' segons la distancia recorreguda pels robots. La primera execució que s'ha fet ha sigut en una superfície totalment plana, on s'han recollit una sèrie de dades representat a la següent gràfica (Fig. 6). Com es pot veure, hi ha una correlació entre el factor de la velocitat i la distancia recorreguda. Té una tendència creixent molt lineal, això vol dir que a mida que es les distancies son més grans, cal augmentar la velocitat mitjana per a que la diferencia de temps entre el bessó digital i el robot físic sigui mínima.

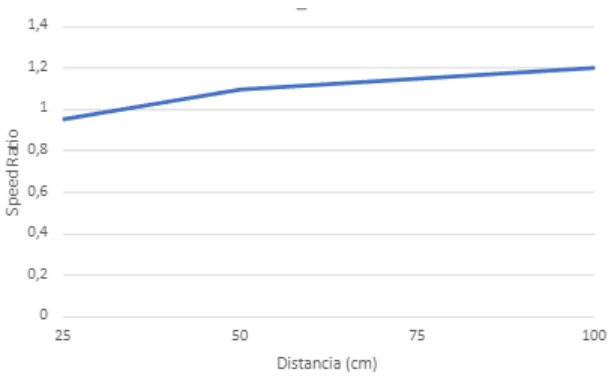


Fig. 6: Relació del speed ratio i temps en superfícies planes

La segona execució que s'ha fet ha sigut en una superfície irregular, on s'han recollit diverses dades de com evoluciona el 'speed ratio' segons la distancia recorreguda. (Fig. 7). En aquest cas, com era d'esperar també s'obté una funció lineal creixent, però aquesta és menys lineal per les imperfeccions de la superfície, que fan el comportament del robot físic una mica més aleatori.
Encara això, es pot veure que el factor de la velocitat és més petit en comparació als valors que hi ha en superfícies planes. El motiu és que aquesta superfície irregular té més fricció, per tant el robot real anirà més lent i no es necessitarà augmentar tant el factor de velocitat com en les superfícies planes.

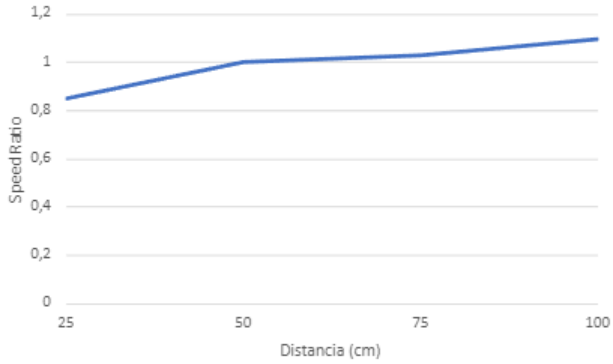


Fig. 7: Relació del speed ratio i temps en superfícies irregulars

Un cop que s'han obtingut aquests resultats s'ha realitzat cuatre execucions, dos en cada superfície d'on s'han obtingut les dades i amb distancies més curtes o llargues. L'objectiu és veure com s'aplica el factor de velocitat en un recorregut real segons les distancies d'aquest. Per fer aquestes execucions s'ha creat un circuit dividit en uns trams que han seguits els robots (Fig. 8).

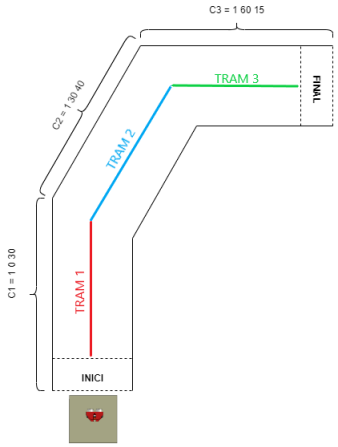


Fig. 8: Recorregut dels robots

4.2 Simulació en superfície plana

En primer lloc s'ha recollit una sèrie de dades per veure el comportament del robot físic sense aplicar el paràmetre del 'speed ratio' sobre superfícies planes. Com es pot observar en la següent taula (Taula. 1) on la columna temps és la diferencia que existeix entre el robot virtual i el real, a mida que les distàncies a recórrer creix, el temps que hi ha entre el robot físic i el bessó digital augmenta considerablement, però a més, ens indica que el robot físic arriba abans que el real en la majoria de casos.

Distancia	Temps
10	0,874 ms
25	-0,075 ms
50	-0,151 ms
75	-0,973 ms
100	-1,625 ms
150	-2,976 ms
200	-4,725 ms

TAULA 1: TEMPS QUE TRIGA A RECÒRRER LES DIFERENTS DISTÀNCIES.

4.2.1 Simulacions amb distàncies curtes

A la següent taula es pot veure quines comandes s'han executat per seguir el circuit mostrat anteriorment, amb el 'speed ratio' que s'ha aplicat a cada tram i el temps que ha trigat en executar-se. (Taula. 2).

Tram	Speed Ratio	Temps entre robots
C1 = 1 0 30	0.9	0,176 ms
C2 = 1 30 40	0.79	0,050 ms
C3 = 1 60 15	0.66	0,100 ms

TAULA 2: RESULTATS OBTINGUTS AMB EL SPEED RATIO APLICAT.

Si fem una comparació entre el temps de cada tram quan s'aplica el 'speed ratio' es pot apreciar la gran diferència i com es capaç d'adaptar-se a la velocitat del robot adequadament segons la distància que cal recórrer o girar, arribant fins a un temps de diferencia de gairebé 0 ms (Fig. 9).

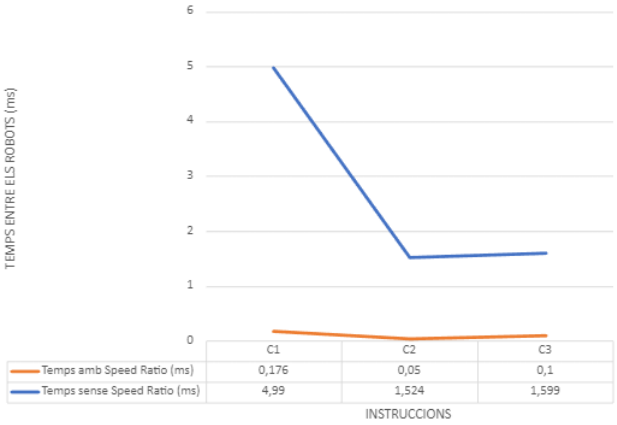


Fig. 9: Comparació dels resultats aplicant el Speed Ratio

4.2.2 Simulacions amb distàncies llargues

En canvi, en una segona prova realitzada amb distàncies més llargues, teòricament el bessó digital hauria d'anar més endarrerit que el robot real abans d'aplicar el paràmetre del 'speed ratio'. Un cop que s'ha aplicat el 'speed ratio' s'han obtingut les següents dades:

Tram	Speed Ratio	Temps entre robots
C1 = 1 0 200	1.17	0,025 ms
C2 = 1 30 200	1.13	0,172 ms
C3 = 1 60 150	1.15	0,075 ms

TAULA 3: RESULTATS OBTINGUTS AMB EL SPEED RATIO APLICAT.

Com es pot veure a l'anterior taula (Taula. 3), el 'speed ratio' ha de créixer amb distàncies grans ja que el robot virtual va més endarrerit que el robot real, el qual es confirma que a distàncies llargues es continua endarrerint més el robot virtual com s'ha explicat anteriorment. A més, afegint un gir a la instrucció, el 'speed ratio' decreix per el temps d'espera programat que té el robot real entre una instrucció de gir i una d'avançament en línia recta. Aquesta espera provoca un alentiment en el robot real, per tant la velocitat del bessó digital ha de disminuir en aquests casos.

Comparant el temps dels diferents trams un cop s'ha aplicat el 'speed ratio', es pot observar que el temps sense el 'speed ratio' és negatiu, per tant al augmentar el paràmetre s'ajusta gairabé a 0 el temps de diferencia entre els dos robots (Fig. 10). Per tant, el bessó digital és capaç d'adaptar-se també a distàncies més llargues un cop es tingui tota la informació prèvia.

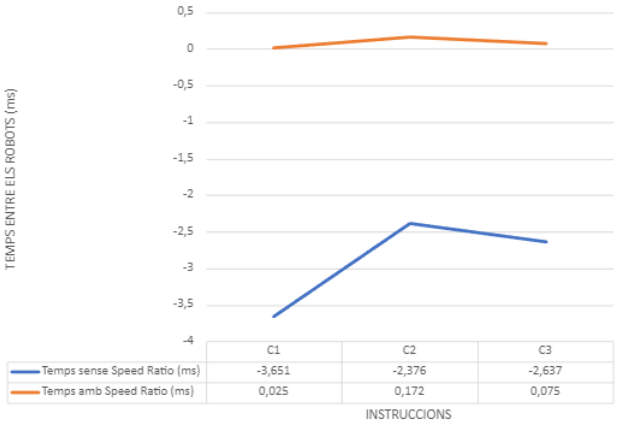


Fig. 10: Comparació dels resultats aplicant el Speed Ratio

4.3 Simulació en una superfície irregular

En aquesta simulació es tractara de fer les mateixes proves que s'ha fet anteriorment però en una superfície més irregular, amb l'objectiu de veure com pot afectar a la velocitat del robot l'entorn on funciona. En primer lloc s'han recollit una sèrie de dades (Taula. 4) sense aplicar el paràmetre del 'speed ratio' per veure com funciona el robot real sobre aquesta superfície .

Distancia	Temps
10	1,025 ms
25	0,950 ms
50	0,426 ms
75	0,599 ms
100	0,076 ms
150	-0,476 ms
200	-0,774 ms

TAULA 4: TEMPS QUE TRIGA A RECÓRRER LES DIFERENTS DISTÀNCIES.

S'han obtingut uns resultats on es mostra que en la majoria de casos el robot físic va més endarrerit que el bessó digital a causa de la superfície, que es més irregular i n'hi ha molta més fricció.

4.3.1 Simulacions amb distàncies curtes

Com a la simulació en superfície plana, la primera simulació realitzada és en el mateix circuit dividit en tres trams diferents. D'aquesta manera podrem veure i comparar les diferències que hi ha en el 'speed ratio' en distàncies llarges i curtes, i a més, podrem fer una comparació entre els resultats dels diferents ambients.

A la següent taula es pot veure quines instruccions se han executat en cadascun dels tres trams del circuit que s'ha fet servir amb el 'speed ratio' que s'ha aplicat i el temps que ha trigat en cada cas . (Taula. 5)

Tram	Speed Ratio	Temps entre robots
C1 = 1 0 30	0.8	0 ms
C2 = 1 30 40	0.73	0,025 ms
C3 = 1 60 15	0.58	0,1 ms

TAULA 5: RESULTATS OBTINGUTS AMB EL SPEED RATIO APLICAT.

En aquest cas, com es pot veure a la següent gràfica (Fig. 11), el temps després d'haver aplicat aquest paràmetre en la velocitat es molt proper a 0ms, per tant, el robot s'ha adaptat correctament al seu entorn i a la distància que calia recórrer en cada moment.

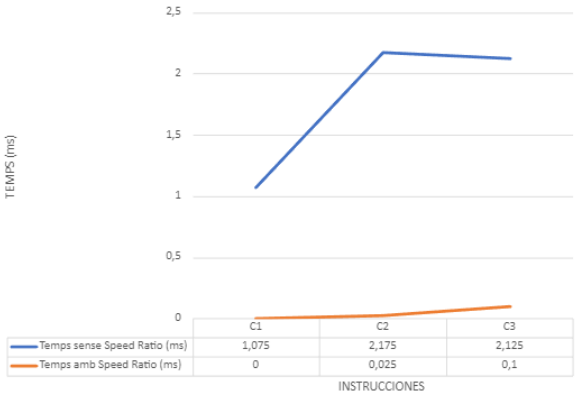


Fig. 11: Comparació dels resultats aplicant el Speed Ratio

4.3.2 Simulacions amb distàncies llargues

En canvi, en una segona prova realitzada amb distàncies més llargues, teòricament el bessó digital hauria d'anar més endarrerit que el robot real, però aplicant el 'speed ratio' s'han obtingut les següents dades:

Tram	Speed Ratio	Temps entre robots
C1 = 1 0 200	1.04	0,077 ms
C2 = 1 30 200	1.01	-0,150 ms
C3 = 1 60 150	0.99	0,125 ms

TAULA 6: RESULTATS OBTINGUTS AMB EL SPEED RATIO APLICAT.

Com es pot veure a l'anterior taula (Fig. 6), el 'speed ratio' ha anat decreixent molt lentament conforme s'ha afegit un angle de gir. Això està provocat per el retràs que provoca aquest gir en la execució de la instrucció. Com que el bessó digital abans d'aplicar el paràmetre anava més endarrerit que el robot real, al aplicar una instrucció més complexa provoca un retràs en el robot real i disminueix aquesta diferència de temps.

Comparant el temps dels diferents trams un cop que s'ha aplicat el 'speed ratio', es pot observar que el temps sense el 'speed ratio' és negatiu. Això vol dir que el robot virtual va més endarrerit que el robot físic, per tant, per ajustar-ho s'ha d'augmentar el 'speed ratio'. Amb aquest paràmetre es manté en un marge acceptable (Fig. 12), molt proper a 0. Per tant, el bessó digital és capaç d'adaptar-se també a distàncies més llargues un cop es tingui tota la informació prèvia.

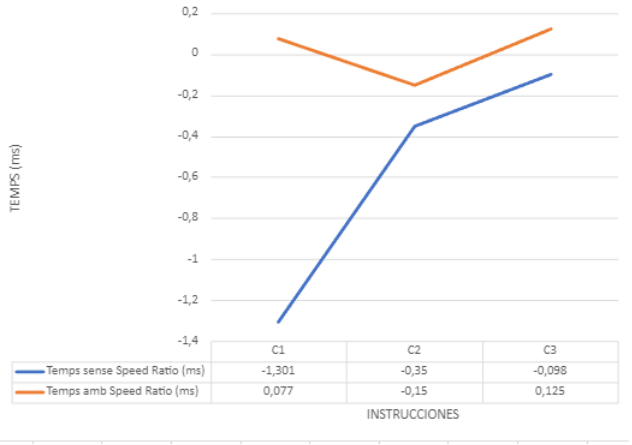


Fig. 12: Comparació dels resultats aplicant el Speed Ratio

5 CONCLUSIONS

L'objectiu inicial era crear un bessó digital que fos capaç d'agafar les dades obtengudes d'un robot real per ajustar la seva velocitat mitjana. Això implicava una sèrie de canvis com l'estructura física, els drivers dels motors i incorporar un emmagatzematge de dades. Encara que tots els canvis es van completar, no s'ha pogut assolir la creació d'una funció per a llegir i calcular el factor de velocitat a aplicar. Per tant, totes les proves fetes han sigut realitzades calculant aquest factor amb diferents proves i posant-lo manualment, fet que ens ha permès arribar a fer diferents proves.

Basant-nos en els resultats que s'han obtingut amb aquest treball podem dir que el bessó digital funciona amb molta precisió encara que no s'ha pogut fer que el bessó digital fos capaç de agafar dades d'altres simulacions per adaptar la velocitat. A més, s'ha pogut comprovar com la distància a recórrer, l'angle de gir i l'entorn on funciona el robot afecta directament a la velocitat del robot, i com amb un paràmetre es pot ajustar aquesta velocitat per a que un funcionament similar entre el bessó digital i el robot real.

Malgrat que la idea de un bessó digital es utilitzar, molt sovint, una gran quantitat de paràmetres per poder ajustar-ho, hem aconseguit ajustar la velocitat del robot real, i a més, ho hem recreat amb mides reals per a millorar els resultats de la simulació.

Finalment, s'ha pogut demostrar que aquest tipus de tecnologia com són els bessons digitals pot arribar a ser molt útil a l'hora de fer simulació en entorns virtuals, i que avui dia, ja és una tecnologia a tenir en compte i que facilita el desenvolupament de noves tecnologies per al nostre dia a dia.

REFERÈNCIES

[1] ¿Qué es un gemelo digital? — IBM. (2020). Ibm.com. https://www.ibm.com/es-es/topics/what-is-a-digital-twin

[2] Arantxa Herranz. (2021, May 26). Digital twins: qué son, para qué sirven y cuáles son los beneficios y problemas de los gemelos digitales. Xataka.com; Xataka. https://www.xataka.com/pro/digital-twins-que-sirven-cuales-beneficios-problemas-gemelos-digitales

[3] de, J. M. (2021, September 27). Coches gemelos: así ayudan a mejorar los vehículos - HackerCar. HackerCar. https://hackercar.com/coches-gemelos-la-solucion-para-que-no-ataquen-tu-vehiculo/

[4] Gemelo digital en la industria aeronáutica - Avantek. (2020, June 4). Avantek. https://avantek.es/video-desarrollo-de-sistemas-electricos-electronicos-y-gemelo-digital-en-la-industria-aeronautica/

[5] Los Gemelos Digitales ya son una realidad en el sector de la salud. (2022, April 27). Distrito E-Health; Distrito e-Health. https://distritodigitalehealth.es/los-gemelos-digitales-ya-son-una-realidad-en-el-sector-de-la-salud/

[6] Tendencias de tecnología 2020. (n.d.). https://www2.deloitte.com/content/dam/Deloitte/pe/Documents/technology/(5)

[7] Digital Twin Market Size Global forecast to 2026 — MarketsandMarketsTM. (2020). Marketsandmarkets.com. https://www.marketsandmarkets.com/Market-Reports/digital-twin-market-225269522.html

[8] Gemelos Digitales: qué son, ventajas y aplicaciones - Iberdrola. (2021). Iberdrola. https://www.iberdrola.com/innovacion/gemelos-digitales: :text=Gartner

[9] Metodología Kanban — Kanban Tool. (2022). Kanbantool.com; Kanban Tool. https://kanbantool.com/es/metodologia-kanban

[10] Trello. (2022). @Trello. https://trello.com

[11] Ansys Twin Builder — Create and Deploy Digital Twin Models. (2021). Ansys.com. https://www.ansys.com/products/digital-twin/ansys-twin-builder

[12] Scripts. (2022). Coppeliarobotics.com. https://www.coppeliarobotics.com/helpFiles/en/scripts.htm

[13] ¿Qué es una API? - Guía sobre las API para principiantes - AWS. (2022). Amazon Web Services, Inc. https://aws.amazon.com/es/what-is/api/

[14] Enabling the remote API - server side. (2022). Coppeliarobotics.com. https://www.coppeliarobotics.com/helpFiles/en/remoteApiServerSide.htm

[15] Plugin tutorial. (2022). Coppeliarobotics.com. https://www.coppeliarobotics.com/helpFiles/en/pluginTutorial.htm

[16] ¿Qué es un plugin y para qué sirve? - El blog de dinahosting. (2021, July 13). El Blog de Dinahosting. https://dinahosting.com/blog/que-es-un-plugin-y-para-que-sirve/