1099 Simulador de conducció enfocat a la investigació

Memòria del projecte de final de carrera corresponent als estudis d'Enginyeria Superior en Informàtica realitzat per

Lluís-Pere de las Heras Caballero

i dirigit per

Felipe Lumbreras Ruiz i
Daniel Ponsa Mussarra.

Bellaterra, Setembre de 2009

Els sotasignats,

Felipe Lumbreras Ruiz i Daniel Ponsa Mussara

Professors de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFIQUEN:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en

Lluís-Pere de las Heras Caballero

I per tal que consti firma la present.

Signat:

Bellaterra, Setembre de 2009

A l'Alba, la Mireia i els meus pares

Agraïments a:

Felipe Lumbreras i Daniel Ponsa
per fer possible aquest projecte

TAULA DE CONTINGUTS

1 INTRODUCCIÓ, MOTIVACIÓ I OBJECTIUS

1.1 INTRODUCCIÓ

Els videojocs com a entreteniment comercial fa més de 35 anys que existeixen. Donada la seva acceptació social, la indústria ha patit un creixement tecnològic i econòmic molt gran, poc comparable a d'altres sectors de l'entreteniment. La ràpida evolució de la tecnologia de computació ha permès que cada cop els videojocs siguin més complexes, buscant apropar-se el més possible a la realitat i maximitzar la interactivitat amb l'usuari final. A més, la gran quantitat de beneficis ingressats per aquesta industria han afavorit el desenvolupament de nous motors gràfics encarregats de proporcionar un realisme insospitat fins ara.

Aquests motors gràfics moderns ja no només s'encarreguen de generar les escenes del joc de forma ràpida i realista sinó que, a més, van acompanyats dels motors físics encarregats del desenvolupament de les restriccions físiques dels objectes del joc. Els motors físics d'avui dia simulen acuradament les condicions naturals del món real. La implementació d'aquesta física realista permet que, un mateix motor gràfic/físic, pugui ser utilitzat per fer videojocs tan diferents com un simulador de cotxes o un joc "Shoot'em up" (disparar en primera persona). Aquesta conjunció del motor gràfic i físic forma el que s'anomena motor del videojoc (en Anglès, "game engine").

D'altra banda, la necessitat de dissenyar i provar robots abans de construir-los ha fet que la indústria de la robòtica hagi vist en la simulació una important eina per dissenyar i experimentar amb robots de manera ràpida, segura i econòmica. Poder dissenyar un robot de forma virtual permet que aquest pugui ser provat, experimentat i millorat abans de la seva creació i així, ser construït quan existeixen garanties del seu bon funcionament en el món real. El principal problema ha estat que, per dissenyar i provar un robot de forma virtual, s'ha de disposar d'un motor gràfic i físic que representi de forma acurada i realista les especificacions del món real. La implementació d'un nou motor gràfic i sobretot físic és complexa, econòmicament elevada i a més, fa que els enginyers robòtics no puguin centrar totalment els seus esforços en aspectes relacionats amb aquesta ciència.

És aquest mateix motiu el que fa confluir a aquests dos mons. Els motors gràfics i físics creats per la indústria dels videojocs són cada cop més potents, realistes i formals. La indústria de la robòtica veu la possibilitat d'utilitzar aquests motors per

crear els seus propis simuladors d'una manera econòmica i ràpida. Això ha fet que actualment existeixin simuladors de robots molt potents i realistes muntats sobre motors gràfics i físics d'alguns dels videojocs més populars d'avui dia.

Aquest projecte és centrat en el desenvolupament d'un simulador de conducció que permeti generar seqüències de dades útils per la validació de sistemes avançats d'assistència a la conducció (a continuació referits com ADAS, de l'anglès "Advanced Driving Assistance System"). El nostre treball pren com a referent l'estratègia emprada per la comunitat de recerca robòtica, on s'ha mostrat molt productiu l'aprofitament dels avanços de la industria dels videojocs a l'hora de desenvolupar un simulador.

1.2 MOTIVACIÓ

El desenvolupament de sistemes d'assistència a la conducció és, actualment, una de les àrees de recerca de més interès per les companyies automobilístiques. Aquests sistemes coneguts com ADAS naixen amb la finalitat d'incrementar la seguretat i el confort dels conductors. Normalment, s'utilitzen sensors per percebre i modelar l'entorn de circulació del vehicle i, a partir de la informació rebuda, els ADAS assisteixen al conductor per tal d'evitar i/o resoldre situacions de perill.

El grup de recerca en ADAS del Centre de Visió per Computador (CVC) porta tot un seguit de projectes relacionats amb aquest àmbit. La base del seu desenvolupament es fonamenta en el processament de seqüències d'imatges rebudes de càmeres instal·lades en vehicles, així com de les dades obtingudes per d'altres sistemes sensorials. El mètode habitual de treball requereix d'una fase prèvia d'adquisició de dades, on els sensors recullen tota la informació d'un vehicle mentre aquest circula per un recorregut determinat. Aquesta informació és utilitzada per dissenyar i implementar un primer sistema prototip. El nou prototip és validat posteriorment a partir de les dades adquirides en nous recorreguts del vehicle, o bé usant-lo en temps real dins del mateix entorn de conducció que havia servit per dissenyar el sistema.

La problemàtica d'aquesta metodologia de treball és que, per tal de mesurar de manera objectiva el rendiment del sistema implementat, cal denotar d'alguna manera quin hauria de ser el seu comportament òptim. En general, això s'aconsegueix contrastant les dades sobre l'entorn inferides pel sistema a cada instant amb aquelles que el descriuen de forma precisa (per exemple, si els vehicles observats es detecten correctament, si la seva posició i velocitat 3D estimada és la correcta, etc). Aquestes seqüències de dades que defineixen l'entorn de manera exacta (d'ara endavant GT, de l'anglès "Ground Truth") comparades amb les obtingudes pel sistema serveixen per

quantificar l'error del prototip. Generalment però, el GT és adquirit de forma manual, el que requereix de moltes hores d'anotació, és costós, subjecte a l'error humà, i en molts casos les dades només poden ser establertes de manera aproximada i, per tant, la definició de l'entorn ja no és exacte.

Per tal d'evitar tots aquests problemes, aquest projecte tracta el desenvolupament d'un simulador que ha de permetre la generació de simulacions realistes en un entorn de conducció i, a més, ha de ser capaç de registrar simultàniament i a cada instant la resposta dels sensors instal·lats en un vehicle i el GT. D'aquesta manera, el simulador facilitarà el procés de validació d'ADAS.

Un cop esdevé que es vol crear aquest simulador, s'estudia treure'n profit dels passos fets per la comunitat de recerca robòtica en el desenvolupament dels seus simuladors. De fet, els vehicles de gamma alta actuals són robots autònoms encoberts ja que disposen de sensors i actuadors desenvolupats dins l'àmbit de la robòtica. Com els simuladors robòtics són basats en videojocs actuals, el simulador de conducció disposaria d'una veracitat de les condicions físiques virtuals i d'una alta qualitat gràfica.

A més, després d'haver fet l'estudi sobre la viabilitat del projecte s'obtenen les següents reflexions:

1. És factible.
2. Hi ha una manca de simuladors de conducció enfocats a la investigació de caràcter lliure.
3. L'ús d'un simulador de robots per crear un de conducció permet desenvolupar una eina molt potent.
4. La majoria de simuladors de conducció enfocats a la investigació actuals disposen d'una qualitat gràfica molt pobre. L'ús d'un simulador de robots que utilitzi el motor gràfic d'un videojoc modern atorgaria una gran qualitat gràfica al simulador.

De totes aquestes reflexions anteriors es conclou que el desenvolupament del projecte a partir d'un simulador de robots serà extremadament útil pel CVC a l'hora de validar els seus projectes relacionats en ADAS.

1.3 OBJECTIUS

L'objectiu principal d'aquest projecte és dissenyar un simulador de conducció a partir d'un simulador de robots existent amb la finalitat de que pugui ser utilitzat en el camp de la investigació.

El simulador que es pretén desenvolupar en aquest projecte ha de ser una eina eficaç en la validació de sistemes ADAS. Per acomplir-ho, aquest haurà de ser capaç de crear entorns realistes de conducció, permetrà recrear el moviment virtual d'un cotxe de forma natural i a més, serà capaç de desar automàticament les dades obtingudes pels sensors instal·lats al vehicle juntament amb el GT. A més, el simulador haurà de disposar d'una interfície que permeti a l'usuari crear diferents tipus de simulacions i controlar-les.

Un altre objectiu del projecte és que el simulador desenvolupat tingui un caràcter genèric, és a dir, que pugui ser utilitzat com a base de modificacions i millores fetes pels enginyers per tal de dur a terme les seves investigacions.

Per tal d'assolir els objectius generals s'hauran d'acomplir els objectius específics següents:

1. Recerca i elecció d'un simulador de robots on es pugui implementar el comportament realista d'un cotxe.

2. Creació d'un entorn realista simple pel qual un vehicle pugui circular.

3. Disseny i creació d'un vehicle pilot que es mogui de forma natural dins l'entorn de simulació.

4. Creació d'una interfície d'usuari on es seleccionin les característiques de simulació desitjada i on pugui ser controlada.

5. Implementar i instal·lar una càmera en el vehicle per tal de poder generar imatges que puguin ser processades posteriorment per un ADAS basat en Visió per Computador (VC).

6. Implementar un sensor capaç de percebre de forma exacte el posicionament i orientació (GT) del cotxe dins l'entorn de simulació.

7. Mostrar a través de la interfície les dades obtingudes pel sensor en temps real.

8. Desar les dades del sensor de forma automàtica a disc per fer viable la validació de sistemes ADAS.

9. Desar a disc automàticament les imatges generades per la càmera instal·lada al vehicle.

La suma de totes aquestes funcionalitats formen l'aplicació **IOCS** – ***Investigation Oriented Car Simulator*** – el simulador de conducció enfocat a la investigació desenvolupat en aquest projecte.

1.4 LA MEMÒRIA

Aquesta memòria té com objectiu il·lustrar quins han estat els passos que s'han seguit per poder crear IOCS, el simulador de conducció enfocat a la investigació.

Com el projecte té un caràcter de continuïtat, la memòria no només és un resum del treball realitzat sinó que a més, serveix de guia per aquells futurs enginyers que vulguin millorar el simulador per tal d'aproximar-lo al seu camp d'investigació. És per això que, durant el transcurs d'aquesta memòria, es faran cites al codi del programa contingut en el CD annexat a aquest document.

Per tal de facilitar la instal·lació i l'ús del l'aplicació creada en aquest projecte, el manual d'usuari de IOCS és adjuntat a l'apèndix d'aquest document.

A continuació s'enuncien els capítols que formen aquest document juntament amb una breu descripció del contingut de cadascun.

1. ***Introducció, Motivació i Objectius.***

En aquest capítol es descriu quina és la motivació que ha impulsat a aquest projecte, els objectius que es volen aconseguir i l'estructura d'aquest document.

2. ***Simuladors de Conducció.***

En aquest capítol es presenten els diferents tipus de simuladors de conducció actuals classificats segons la funció per la qual han estat creats. Es presenta una breu conclusió de quina ha de ser la funcionalitat d'un simulador de conducció enfocat a la investigació.

3. ***Simuladors de Robots.***

En aquets capítol es fa una síntesi de quins són els possibles simuladors de robots per desenvolupar IOCS i es presenta el que més s'ajusta a les necessitats del projecte.

4. ***USARSim.***

En aquest capítol es resumeixen les principals característiques del simulador de robots seleccionat com a base del projecte, anomenat USARSim. També es ressenyen les principals virtuts del simulador que permeten el desenvolupament del projecte.

5. ***Construcció d'escenaris a USARSim.***

En aquest capítol es descriu quins són els passos que cal seguir per a construir un escenari vàlid a USARSim. Es mostra l'escenari desenvolupat com a entorn de conducció on es duran a terme les simulacions.

6. ***Creació de vehicles a USARSim.***

En aquest capítol s'explica el procés seguit per a la construcció d'un cotxe d'estètica realista a USARSim.

7. ***Extensions a USARSim.***

En aquest capítol s'exposen les funcionalitats afegides a USARSim: implementació del moviment del vehicle, implementació del sensor de dades GT i la instal·lació dels dispositius sensorials al vehicle.

8. ***Desenvolupament de IOCS.***

En aquest capítol es descriu el procés de desenvolupament i les funcionalitats de IOCS. També s'exposen les decisions de disseny més importants preses durant la seva implementació.

9. ***Conclusions i treball futur.***

En aquest capítol es resumeixen les conclusions obtingudes en el desenvolupament d'aquest projecte així com les possibles extensions i utilitats futures del simulador de conducció.

2 SIMULADORS DE CONDUCCIÓ

2.1 INTRODUCCIÓ

Per tal de desenvolupar un simulador de conducció enfocat a la investigació, s'ha realitzat una recerca entre els simuladors de conducció existents per determinar quines són les seves principals característiques. Aquesta exploració permet concloure quines són aquelles capacitats que ha d'oferir el simulador per tal de que pugui ser utilitzat en l'àmbit de la investigació.

2.2 TIPUS DE SIMULADORS DE CONDUCCIÓ

Cada cop existeixen més simuladors de conducció de tot tipus de vehicles. Aquests simuladors són eines que permeten reproduir el comportament dels vehicles que simulen de forma virtual sense que l'usuari s'exposi món real. La finalitat d'aquests simuladors és variada i depèn dels objectius pels quals ha estat dissenyat. Segons aquests objectius podem diferenciar entre tres grans grups:

- *Simulador de conducció enfocat a l'entreteniment*
- *Simulador de conducció enfocat a l'aprenentatge*
- *Simulador de conducció enfocat a la investigació*

Per tal de distingir bé entre els tres grups es resumeixen a continuació quines són les principals característiques de cadascun.

2.2.1 SIMULADORS DE CONDUCCIÓ ENFOCATS A L'ENTRETENIMENT

Els simuladors de conducció enfocats a l'entreteniment tenen com a objectiu fer passar una bona estona a l'usuari. Són els que comunament anomenem videojocs de cotxes. Generalment aquests simuladors no busquen un realisme pur en els moviments del cotxe sinó que el més important és generar una alta qualitat gràfica i una exageració en el modelatge d'aspectes com la velocitat, destrucció, etc. La figura

2.1 mostra una imatge del videojoc Gran Turismo 4 The Real Driving Simulator, un videojoc que destaca pel seu realisme gràfic.

Figura 2.1 Escena del videojoc Gran Turismo 4

2.2.2 SIMULADORS DE CONDUCCIÓ ENFOCATS A L'APRENENTATGE

El simuladors de conducció enfocats a l'aprenentatge estan dissenyats per tal de que els usuaris que els utilitzen puguin aprendre a conduir el vehicle sense exposar-se al perill de fer-ho en el món real. L'objectiu d'aquests simuladors és recrear el més realísticament possible la interacció entre la persona i el vehicle. Un bon simulador de conducció enfocat a l'aprenentatge és aquell que és capaç de simular el comportament d'un cotxe, seguint les comandes de l'usuari, tal i com es comportaria en la realitat i en les mateixes condicions.

Segons el Reial Automòbil Club de Catalunya (RACC) [11], aquesta és una eina per aprendre a conduir cada cop més evolucionada. L'augment de la complexitat d'aquests simuladors ha fet que recreïn la conducció d'un vehicle de forma natural. D'acord amb el que diu aquesta institució, aquests simuladors permeten que els estudiants de l'autoescola s'habituïn a l'entorn del vehicle abans d'haver fet cap pràctica de conducció i que, aquella gent que perd la confiança a l'hora de conduir, pugui recuperar-la per tornar a utilitzar l'automòbil.

El simulador TUTOR[12] (figura 2.2) és una eina de formació i aprenentatge per a transportistes de camions i autobusos. El simulador recrea situacions d'emergència per a que el conductor assimili quin és el comportament del vehicle. Aquestes simulacions generades són repetides fins que el conductor aprèn a resoldre-les correctament i rep el vist-i-plau de l'instructor.

Figura 2.2 Simulador de conducció TUTOR

2.2.3 SIMULADORS DE CONDUCCIO ENFOCATS A LA INVESTIGACIÓ

És el tipus de simulador que es du a terme en aquest projecte. La principal diferència amb els simuladors anteriors és que aquests generen un tipus de dades útils per tal de dur a terme investigacions sobre l'esdeveniment simulat. Aquesta informació obtinguda ha de ser el més realista possible ja que generalment és utilitzada per treure conclusions concernents a la realitat. El tipus de informació de la simulació generada dependrà d'allò que s'estigui investigant. Per tal de que aquestes dades reflecteixin el millor possible la realitat és molt important que el comportament del cotxe al simulador sigui del tot realista. Així, no és tant important que el simulador generi una alta qualitat gràfica.

La majoria dels simuladors de conducció enfocats a la investigació són propietat de les empreses automobilístiques. Aquestes empreses utilitzen els seus simuladors per estudiar la introducció de noves millores en els seus vehicles, per exemple sistemes ADAS. La veracitat d'aquests simuladors els permet estudiar el comportament del cotxe amb nous elements abans de que siguin introduïts físicament. Fan servir aquestes eines per investigar noves millores en àmbits com l'eficiència del cotxe, el

confort i la seguretat dels conductors. Un exemple és el simulador THUMS[1] [13][14] de Toyota, que permet simular les repercussions d'un accident a gran velocitat en els habitants d'un vehicle. L'estudi de les lesions produïdes per aquests accidents permet perfeccionar els elements de seguretat activa i passiva dels vehicles.

Però no només les empreses del sector de l'automòbil disposen d'un simulador de conducció enfocat en la investigació. Actualment la policia utilitza aquests instruments per tal de fer la reconstrucció d'accidents de trànsit. Introduint les dades referents a l'accident, el simulador es capaç de reconstruir l'accident i donar a conèixer les causes que el van provocar. La figura 2.3 mostra un simulador de conducció per a la recreació d'accidents.

Figura 2.3 Imatge d'un simulador per a la reconstrucció d'accidents

El RACC també utilitza aquest tipus de simuladors per realitzar les seves recerques en matèria de seguretat. Segons el RACC [6], aquestes eines permeten l'obtenció de dades més complexes i més exactes que la resta de procediments utilitzats. Recentment ha fet servir aquests simuladors per tal de determinar quines són les principals causes que interfereixen en la concentració d'un conductor.

[1] Total Human Model for Safety, simulador creat per Toyota i la Federació Internacional de l'Automòbil, la FIA

2.3 CONCLUSIÓ

Com hem vist, els simuladors de conducció enfocats a la investigació generen un tipus d'informació vàlida pels investigadors. El tipus de dades depèn d'allò que s'està estudiant. No té sentit que el simulador faciliti la informació referent al funcionament del pilot d'una llum interior del cotxe quan l'objectiu dels enginyers és estudiar els sistemes de seguretat activa del vehicle.

Com que el simulador que es desenvolupa en aquest projecte no té marcada una aplicació concreta, més enllà de servir com a eina de validació de sistemes ADAS, haurà de ser capaç d'obtenir el ventall més ampli possible de tipus de dades (p.e. resposta dels sensors del vehicle a l'entorn), i així, poder ser utilitzat per tot tipus d'estudis posteriors.

3 SIMULADORS DE ROBOTS

3.1 INTRODUCCIÓ

Com ja s'ha dit abans, la nostra proposta de simulador de conducció enfocat a la investigació prendrà com a base el treball fet per la comunitat científica orientada a la robòtica. Això comporta fer una recerca exhaustiva per tal de trobar aquell simulador de robots que ens permeti poder desenvolupar el millor simulador de conducció en investigació.

El primer que es fa és analitzar quines són les condicions necessàries que ha de posseir el simulador de robots per tal de que el projecte sigui factible. A continuació es citen els tres punts imprescindibles conclosos d'aquest estudi:

1. Ha de permetre dissenyar robots que es moguin amb rodes i així, permetre la creació d'un cotxe.
2. Ha de permetre obtenir el ventall més ampli de dades referents al comportament del robot.
3. Ha de ser programari lliure (open-source) per tal de poder fer les modificacions necessàries per crear el simulador.

Marcats aquests tres punts determinants que ha d'acomplir el simulador de robots escollit, comença una recerca per tal de trobar el més adient a l'hora de crear el simulador de conducció.

3.2 ESTUDI

A continuació es fa un resum dels **principals** simuladors de robots estudiats per a poder crear el simulador de conducció enfocat a la investigació. També es destaquen quins són els seus principals avantatges i inconvenients a l'hora de permetre crear el nostre simulador:

- ***Contact Dynamics Simulation for Space Robotics Applications*** [10]**.** Simulador de robots espacials.
  Es tracta d'un simulador massa específic que faria que el disseny d'un vehicle "típic" fos molt costós.

- ***Robot simulation, collisions and contacts*** [9]**.** Simulador de robots humanoide.
  El simulador està dissenyat específicament per crear robots que emulin el comportament humà. No està dissenyat per la creació de robots amb rodes tot i que no seria impossible.

- ***SARGE*** [8]**.** Són les inicials de *Safe and Rescue Game Environment.* És un entorn gràfic per a la simulació d'operacions de recerca i rescat amb robots. Utilitza el motor d'un videojoc modern per tal de crear l'entorn. Conté interfícies de controls de robots integrades capaces d'obtenir en temps real les dades obtingudes des del robot.
  En aquest simulador es dissenyen robots sobre rodes cosa que es podria transportar al disseny de cotxes. El fet d'estar construït sobre un videojoc fa que generi una alta qualitat gràfica a la vegada que una física realista.El preu d'una llicència per a desenvolupar aplicacions sobre Windows utilitzant el motor gràfic d'aquest simulador és de 1.499$, uns 1.061€ (a data de 22/08/2008).

- ***USARSim.*** Són les inicials de *Unified System for Automation and Robotics Simulation.* És un simulador de robots utilitzats per posar-los a prova en situacions de rescat en escenaris urbans. Permet simular robots movibles sobre rodes, articulats, submarins i aeris. Utilitza el motor gràfic i físic d'un videojoc comercial.
  USARSim compleix totes les condicions necessàries. Utilitza un motor gràfic d'un joc modern. El motor físic és extremadament realista i a més, es tracta de programari lliure. Només s'hauria de comprar el videojoc sobre el qual està dissenyat, uns 9,95€ aproximadament (a data de 22/08/09).

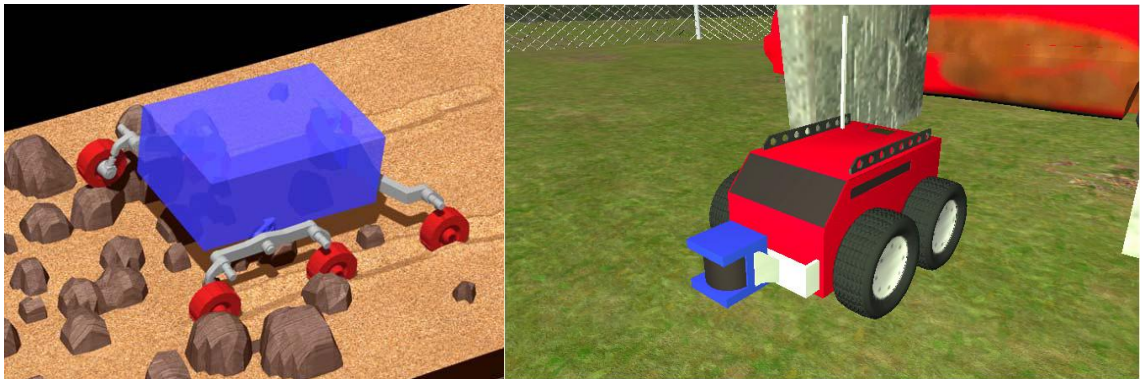


Figura 3.1
A la esquerra: Imatge de Contact Dynamics Simulation for Space Robotics Applications
A la dreta: Imatge del simulador de robots SARGE

3.3 ELECCIÓ

Un cop fet l'estudi dels diferents simuladors robòtics, l'elecció es redueix entre dos possibles candidats de qualitats similars.

- El simulador de robots ***SARGE***.
- El simulador de robots ***USARSim.***

És en aquets moments on comença un anàlisi en profunditat dels dos candidats per tal d'escollir-ne un. Es tenen en compte tots els detalls de cadascun, generals i específics, i s'intenten contrastar. L'escolliment erroni pot fer que el projecte no pugui aconseguir tota la funcionalitat que es vol aconseguir a priori. Aquesta decisió marca el devenir del projecte i per tant s'ha d'estar completament segur quan es produeixi l'elecció. La taula 3.1 mostra les característiques d'un i altre simulador.

Característiques	**SARGE**	**USARSim**
Creació d'un cotxe possible	Sí	Sí
Open-Source	A mitges	Sí
Motor Gràfic	*UnrealEngine* 2 / Unity3d [26]	*UnrealEngine* 3
Motor físic	nVidia PhysX	Karma Engine
Cost de llicència	Motor gràfic: 1061€	Motor gràfic: 9,95 €
Llenguatge de programació	Similar a Java Script i C#	*UnrealScript* (Similar a C++)
Documentació simulador	Completa	Completa
Documentació motor gràfic	Completa	Parcial
Suport institucional	-	IEEE, NIST

Taula 3.1 Mostra les principals característiques de SARGE i USARSim

Un cop feta la recerca i contrastada tota la informació es resol que, es pot aconseguir el millor rendiment realista d'un cotxe tant gràficament com físicament a un menor cost en tots el sentits amb la utilització d'***USARSim***. Els detalls específics dels avantatges d'utilitzar USARSim per desenvolupar el projecte són definits en el capítol següent.

4 USARSIM

4.1 INTRODUCCIÓ

En aquest capítol es fa un resum de les principals característiques del simulador de robots USARSim. La majoria de les especificacions són extretes del seu manual corresponent a la versió 3.1.3 d'aquest software [1]. Aquesta informació es completa amb citacions d'altres estudis fets sobre el simulador.

La informació presentada en aquest capítol és analitzada en l'apartat 4.6 on presenten els principals avantatges que proporciona aquesta eina a l'hora d'implementar el simulador de conducció.

4.2 QUÈ ÉS USARSIM?

USARSim - *Unified System for Automation and Robotics Simulation* – és un simulador lliure (open-source) d'alta fidelitat que permet dissenyar, crear, controlar, estudiar i posar a prova robots terrestres, aeris i submarins.

El simulador està dissenyat sobre el motor d'un videojoc comercial. Aquest fet fa que disposi d'un alt realisme físic i gràfic a un baix cost. A més, USARSim pertany al NIST, l'Institut Nord-americà d'Estàndards, i és recolzat per la institució IEEE, l'Institut d'Enginyers Elèctrics i Electrònics, on actualment desenvolupa la seva competició de robots virtuals *Robocup* [4][5].

A continuació s'exposen les principals raons per les quals USARSim és un dels referents actual en la simulació de robots i és tan utilitzat arreu del món, tant per professionals com per aficionats:

- *Codi i programari lliure (open-source software)*. No cal pertànyer a cap institució de robòtica ni de disposar d'un gran capital econòmic per utilitzar el simulador.

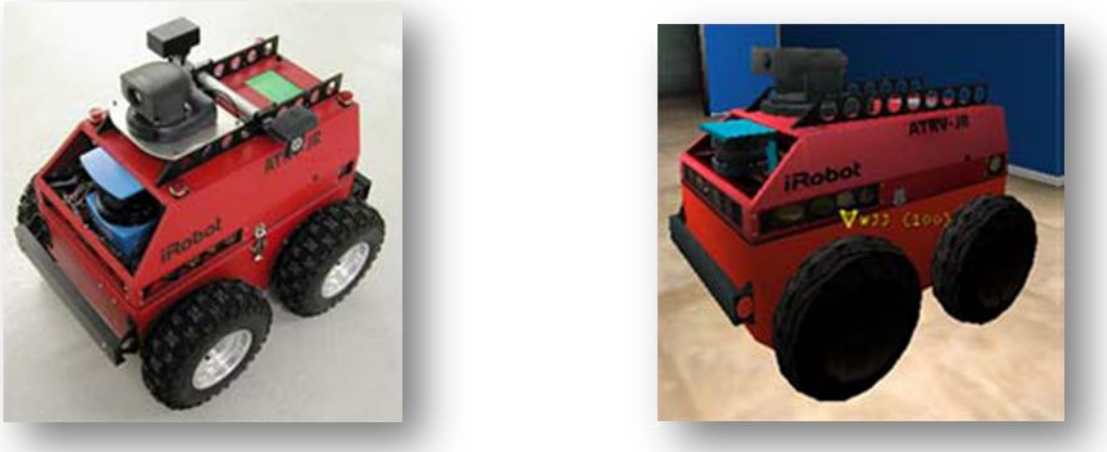
- *Utilitza el motor físic d'un videojoc comercial.* Això permet als usuaris la creació d'entorns de simulació realistes de manera fàcil i pràctica.

- *És dissenyat sobre el motor d'un videojoc de forma independent.* Aquest fet permet que USARSim s'adapti a possibles millores del motor sobre el que funciona. Els motors de videojocs avancen ràpidament, i per tant, USARSim també. La figura 4.1 mostra la qualitat gràfica del simulador.

- *La validació ho és tot.* Els desenvolupadors poden oblidar crear els seus propis simuladors i centrar-se en experimentar i validar les seves creacions.

- *Un suport important.* El fet de que el simulador estigui controlat i validat pel NIST assegura als investigadors la veracitat de les simulacions realitzades a USARSim.

Com a exemple, la extraordinària potència d'aquest simulador i les seves amplies possibilitats són utilitzades per DARPA[2] en l'*Urban Challenge* [7][15] que cada any celebra des de 2006. Es tracta d'un concurs de cotxes conduits de manera autònoma on els concursants han de sotmetre els seus vehicles a un seguit de proves en situacions de tràfic real. DARPA utilitza USARSim per estudiar la creació d'un cotxe sense pilot d'ús quotidià i/o militar en un futur no massa llunyà.



Figutra 4.1
A la dreta: *El robot ATVJr al món real.*
A l'esquerra: *El robot ATVJr simulat a USARSim.*

[2] Defense Advanced Research Projects Agency, és l'agència d'investigació tecnològica de l'exèrcit Nord-americà.

4.3 HISTÒRIA I EVOLUCIÓ D'USARSIM

La idea de substituir robots per persones en tasques de rescat, extinció d'incendis i operacions militars per minimitzar la possible pèrdua vides humanes no és nova. Però, com aquestes emergències no es donen relativament sovint i donat l'alt risc que comporten, no es dóna l'oportunitat d'utilitzar robots poc experimentats que poden dur al fracàs l'operació. A més a més, pel desenvolupament complert d'aquests robots s'han d'invertir altíssimes quantitats econòmiques, l'alt risc al que estan sotmeses aquestes màquines fan que aquestes inversions sovint no siguin fructuoses.

Tot això fa que l'experimentació prèvia amb robots abans de sotmetre'ls a emergències d'aquesta mena sigui primordial. Un simulador que posés a prova els robots en situacions de risc permetria estudiar i optimitzar el seu comportament en situacions reals en la fase del seu disseny, i d'aquesta manera, reduir la despesa econòmica (figura 4.2). És així com sorgeix USARSim.

Figura 4.2 Imatge d'USARSim. Mostra la col·laboració entre robots en emergències

USARSim és creat com simulador d'operacions de rescat urbanes *USAR*[3] desenvolupat per estudiar la interacció entre humans i robots (HRI, de l'angès Huma-Robot Intreaction) i la coordinació d'un grup de robots treballant en equip.

El seu desenvolupament començà el 2002 a la Universitat de Pittsburgh amb col·laboració del Carnegie Mellon University (a continuació referit com a CMU). Aquesta primera versió funcionava sobre el motor gràfic *UnrealEngine* 2 [21] del videojoc Unreal Tournament, propietat de l'empresa Epic Games [25]. Els robots només podien ser controlats utilitzant RETSINA [3], un software multi-agent i, en aquells principis, USARSim només permetia utilitzar un petit nombre de robots amb rodes als que se'ls hi podia adherir pocs tipus de sensors. Aquests robots només podien ser simulats en tres escenaris creats pel NIST [2]. L'objectiu era poder comparar el comportament dels robots virtuals amb els reals i per fer-ho, es va crear una rèplica exacta dels escenaris tant en la realitat com en el simulador (apartat 4.4.2.1).

El comportament tant realista que es va aconseguir dels robots virtuals va fer que el 2005 la Robocup comencés a utilitzar aquesta plataforma per celebrar la seva competició en format virtual. Aquell any, el projecte d'USARSim passà a mans de NIST, qui va reorganitzar el codi base del programa tot mantenint majoritàriament l'estructura implementada per la Universitat de Pittsburgh i el CMU.

Any rere any, amb el suport i control d'institucions tan importants com NIST i IEEE, USARSim ha anat evolucionant molt ràpidament. Actualment utilitza el motor gràfic *UnrealEngine* 3 i el motor físic *KarmaEngine*, tots dos molt avançats tecnològicament. En la seva ultima versió 3.31, el simulador ofereix 15 tipus de sensors diferents i més de 23 escenaris virtuals.

El 2007 canvia el nom per *Unified System for Automation and Robotics Simulation* cosa que no fa canviar el seu acrònim original, USARSim. Això va esdevenir com a conseqüència de que el simulador ja no era principalment utilitzat pel que s'havia concebut.

[3] Inicials d'Urban Search and Rescue que descriu el procediment complert d'operacions de rescat.

4.4 CÓM FUNCIONA USARSIM?

4.4.1 ARQUITECTURA

L'arquitectura interna del simulador es pot separar en dues parts molt diferenciades. A la figura 4.3 es veu clarament aquest fet.

- Client.
- Servidor.

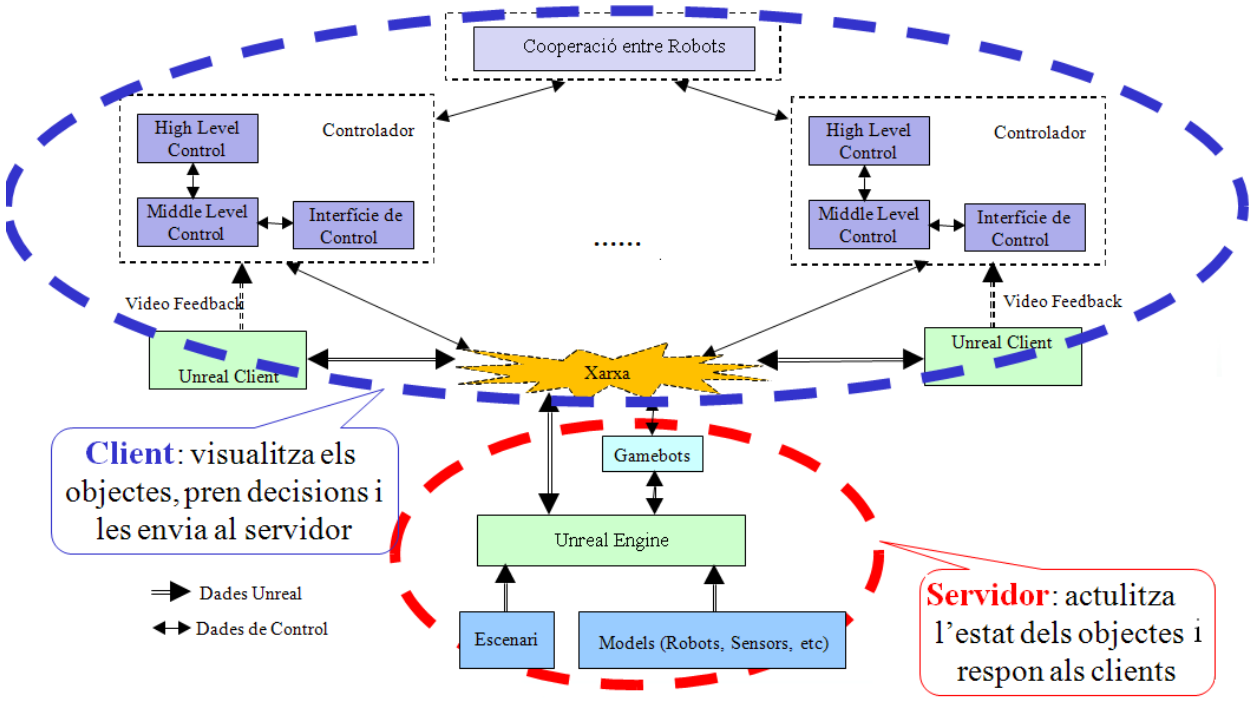


Figura 4.3 Esquema del funcionament d'USARSim

El Sector Client:

El sector *Client* és on l'usuari pot dur a terme les seves investigacions. Comprèn els clients o *UnrealClients*: **executables** del videojoc Unreal Tournament 2004 que USARSim utilitza per a renderitzar l'entorn simulat; i els controladors o interfícies creades pels usuaris. Tots els clients envien les ordres en forma de missatges al *Servidor* a través de la mateixa xarxa, cosa que permet que varis clients puguin actuar sobre la mateixa simulació.

El Sector Servidor o UnrealServer

El sector *Servidor* és l'encarregat de rebre la informació dels clients i de respondre a través de la xarxa. El *Servidor* envia a tots els clients la mateixa informació. Està composat pel motor del videojoc (*UnrealEngine*), els programes gestors del pas de missatges entre els dos sectors (Gamebots), els escenaris de simulació, els robots i tots els objectes simulats.

A continuació es descriuen més detalladament tots els elements que formen els dos sectors en els que està dividit USARSim.

4.4.1.1 UNREAL ENGINE 3

UnrealEngine 3 [21] és el motor utilitzat per USARSim. És propietat d'Epic Games i va inclòs en el videojoc Unreal Tournament 2004 [22]. És necessària la compra del joc per tal de disposar del motor i, per tant, fer servir l'USARSim. Aquest joc té un preu aproximat de 9.95€ (a data de 22/08/09) en qualsevol botiga de videojocs del país.

El motor del joc no només inclou un motor gràfic molt potent sinó que també, va acompanyat d'un motor físic anomenat *KarmaEngine* [16] per obtindre una realitat física avançada. A més, utilitza un llenguatge interpretat anomenat *UrealScript* [29] que permet als seus usuaris crear els seus propis *mods*[4]. A partir d'aquest llenguatge interpretat els desenvolupadors creen tots els objectes, anomenats *Actors*, i els seus controls. El motor també inclou *UnrealEd* [28], una eina de disseny 3D que permet la creació de manera fàcil d'escenaris.

[4] De l'anglès *modification*, és una modificació d'un videojoc feta per els seus usuaris generalment amb l'objectiu d'incrementar la jugabilitat.

4.4.1.2 ELS GAMEBOTS

El protocol de comunicacions utilitzat per l'*UnrealEngine* és propietari i totalment transparent per l'usuari del simulador. Això vol dir que l'accessibilitat per part d'altres aplicacions és molt complicada. Per tal de poder facilitar la comunicació amb el motor, els investigadors en aquesta matèria van introduir unes modificacions en forma de *patch*[5] a l'Unreal Tournament i van crear els Gamebots

Els Gamebots obren un pont de connexió entre el motor del videojoc i les aplicacions d'usuari. Utilitzen una estructura en forma de socket TCP/IP per tal de dur a terme l'intercanvi de missatges en ambdós sentits, tant del *Servidor* al *Client* com del *Client* al *Servidor.*

4.4.1.3 ELS CONTROLADORS

Els controladors són aplicacions pertanyents al sector del *Client* dissenyades i implementades pels usuaris d'USARSim per dur a terme l'estudi desitjat del robot. S'encarreguen de connectar amb el *Servidor* i tot seguit d'enviar l'ordre de carregar el robot en la simulació. Un cop el robot ha estat creat, el controlador escolta pel socket de connexió per on rep la informació procedent de l'*UnrealServer* i envia les comandes per controlar el robot. Aquesta arquitectura client/servidor és la que permet afegir més d'un robot simultani a la simulació, però com cada robot necessita un socket per comunicar-se, per cada robot el controlador haurà d'obrir una nova connexió amb el servidor.

Existeixen tres controladors ja creats que faciliten la feina a l'usuari:

- MOAST
- Pyro
- Player

Aquests programes integren tots els requisits de comportament necessaris per qualsevol controlador comentats anteriorment. A més, faciliten de forma interactiva la resolució de certs aspectes relacionats amb l'obtenció de la informació provinent del servidor, les comandes de control i la integració de sensors als robots.

[5] Anomenat comunament *parche* en castellà és una actualització d'un programa que generalment serveix per introduir-hi noves prestacions.

4.4.2 COMPONENTS DE LA SIMULACIÓ

4.4.2.1 ESCENARIS DE SIMULACIÓ

Com ja s'ha parlat anteriorment, en els seus inicis USARSim només disposava de tres escenaris dissenyats pel NIST. Aquets escenaris van ser desenvolupats a partir del projecte Reference Test Facility for Autonomous Mobile Robots (figura 4.4). Actualment però, la introducció del software *CAD*[6] *UnrealEd* inclosa en *UnrealEngine* 3 ha permès als usuaris dissenyar els seus propis escenaris.

Figura 4.4
A l'esquerra: Red Arena dissenyada per NIST
A la dreta: Red Arena virtual a USARSim

Aquesta eina permet dissenyar mapes a partir d'editors de creació en 3D com Studio MAX o Blender. A partir de la geometria desenvolupada en aquests editors, *UnrealEd* s'encarrega de calcular les primitives de col·lisions dels obstacles, genera els efectes de llum de l'escenari, crea els efectes especials i adapta l'escenari desenvolupat per a que sigui generat per l'*UnrealEngine*.

[6] Acrònim de Computer Aided Design, disseny assistit per ordinador.

4.4.2.2 SENSORS

Els sensors són una part molt important en la robòtica. Els robots autònoms prenen les seves decisions a partir de la informació rebuda pels sensors que incorporen. USARSim inclou varis sensors agregables als robots.

- **Sensors de propietat.** *Encarregats de donar informació de les propietats del robot com per exemple el nivell de bateria.*
- **Sensors d'estimació de posició.** *Són sensors que aproximen la posició i l'estat del robot dins de l'entorn simulat, per exemple el sensor de velocitat.*
- **Sensors de percepció.** *Transmeten la informació obtinguda respecte a objectes relatius al robot, per exemple el sonar.*

Tots els sensors són configurables. Quan un sensor és annexat a un robot, les seves especificacions poden ser especificades modificant les classes programades en *UnrealScript*. Per exemple, es pot modificar el rang de percepció d'un sensor sonar a partir del atribut *Range* de la seva classe.

4.4.2.3 ROBOTS

Els robots són creats a partir de classes en *UnrealScript* on és implementat el comportament de cadascun i el conjunt de malles estàtiques que els hi donen forma.

Les classes que defineixen el robot estan formades per les especificacions físiques i tècniques que el descriuen. El comportament del robot ve definit per les propietats anomenades *KParams* que són interpretades pel *KarmaEngine*.

Les malles estàtiques (*Static Meshes*) són objectes creats en editors gràfics 3D i transformats per *UnrealEd*, que són utilitzats pel motor físic per accelerar la generació gràfica en el videojoc. Generalment, aquestes *Static Meshes* són utilitzades per construir el xassís del robot i d'altres parts com per exemple les rodes. El xassís i les diferents parts del robot s'uneixen mitjançant un sistema d'articulacions i parells de torsió simulades.

4.5 VALIDACIÓ DE MILLORES

Un aspecte molt rellevant d'USARSim és el manteniment de l'aplicació i la introducció de millores. Per tal de mantenir el realisme aconseguit fins ara a l'hora de garantir els mateixos resultats en el traspàs d'un robot simulat al món real, aquelles millores que vulguin ser introduïdes en el simulador han de passar per un procés de validació.

Un cop s'ha desenvolupat en el simulador qualsevol tipus de sensor, robot, algorisme de control, etc, haurà de passar dues proves, una en el simulador i l'altre en el món real. Un cop s'hagin fet aquestes dos experiments es compararan els resultats i es contrastarà la informació obtinguda. Si la millora que es vol introduir es comporta de la mateixa manera en ambdós casos, i amb el consentiment de NIST, serà validada i es pujarà a la pagina web del projecte per tal que sigui disponible per a tots els usuaris d'USARSim.

4.6 USARSIM I EL SIMULADOR DE CONDUCCIÓ

Els principals avantatges a l'hora de fer servir USARSim per crear el nostre simulador, més enllà de que es tracta d'un simulador open-source i que disposa d'un motor de videojoc excel·lent, són descrits a continuació:

- *Estructura Client/Servidor.* Aquesta estructura, que utilitza un socket simple en la comunicació entre la part del control d'usuari amb el motor del simulador, facilita l'obtenció de tot tipus de informació generada per l'*UnrealEngine* referent al comportament del cotxe en la simulació. A més, també permet, de manera poc complicada, enviar les comandes de moviment del cotxe en sentit contrari (funcionalitat de *Controlador*, apartat 4.4.1.3).

- *Llenguatge interpretat UnrealScript.* Els sensors i els robots estan implementats en aquest llenguatge orientat a objectes. L'*UnrealEngine* interpreta el codi de tots els objectes simulats en cada instant d'actualització de l'escena. El fet de que es tracti d'un llenguatge semblant a C++ facilita la implementació del cotxe i els sensors corresponents per tal d'enviar la informació necessària a la interfície de control.

- *UnrealEd.* Aquest editor 3D simple permet la creació d'escenaris realistes traient profit d'altres editors (com Blender o Studio MAX). A més, també és qui construeix les malles estàtiques pertanyents al xassís i les rodes dels robots. La utilització d'aquest editor simplifica la creació d'un escenari i un cotxe realista estèticament.

- *Obtenció d'Imatges.* Existeix la manera de poder captar les imatges generades pel client accedint al seu *buffer*[7] de renderització. Les imatges són obtingudes en un format on queda especificat el color de cada píxel. Això pot ser molt valuós en aspectes relacionats en el camp de la Visió per Computador.

El principal desavantatge a l'hora d'utilitzar USARSim és que cap dels *Controladors* de robots existents que utilitza són transportables per controlar un cotxe. El motiu és que aquests controladors utilitzen comandes de moviment específiques dels robots validats en USARSim. La modificació d'aquestes comandes és inaccessible per l'usuari i, a més, no permeten l'obtenció d'informació rebuda pels sensors creats per l'usuari.

Tot això fa que s'hagi de construir un *Controlador* que generi comandes de moviment del nou vehicle i que pugui obtindre tot tipus de informació referent al comportament d'aquest.

L'encarregat de dur a terme les funcionalitats que desenvolupen els *Controladors* serà implementada per IOCS. Aquest gestionarà tota la informació rebuda del Sector *Servidor* i enviarà les comandes de control del moviment del vehicle.

[7] És la paraula provinent de l'anglès que s'utilitza per descriure un espai abstracte on es guarden dades en programes de computació.

5. CONSTRUCCIÓ D'ESCENARIS A USARSIM

5.1 INTRODUCCIÓ

Com que la creació d'un escenari on dur a terme les simulacions no era un dels objectius plantejats a priori, es va optar per utilitzar un dels *nivells*[8] ja desenvolupats en l'USARSim.

En les primeres proves del disseny del cotxe desenvolupat es va utilitzar DM-ARDA, un dels escenaris fets servir en la RoboCup del 2008 (figura 5.1). Però el problema va estar en que els creadors d'aquest mapa havien implementat efectes externs, com el vent i la neu, que influïen en el comportament inicial del cotxe.

Figura 5.1 Imatge del mapa DM-Arda_250

Després d'estudiar el funcionament d'*UnrealEd*, el qual no té res a veure amb d'altres editors 3D, es va optar per dissenyar un escenari el més simple possible per tal de desenvolupar el comportament realista del cotxe.

[8] De anglès "level". Cada un dels escenaris en els que es desenvolupa un videojoc. En el cas d'aquest projecte: Escenari de simulació.

En aquest capítol es fa un breu resum del funcionament d'*UnrealEd*. Aquells investigadors que vulguin ampliar els seus coneixements en aquesta aplicació i tenint en compte que no existeix un manual del que guiar-se, la plana web enunciada a continuació els pot servir de gran ajuda:

http://www.3dbuzz.com/

5.2 UNREALED

L'editor de l'Unreal Tournament, *UnrealEd*, és l'eina que permet la creació d'escenaris vàlids pel joc Unreal Tournament 2004. Aquesta eina permet als usuaris d'aquest videojoc dissenyar nous nivells on dur a terme les seves partides. Però, tot i que pot semblar una eina molt potent només mirant el seu entorn de treball mostrat en la figura 5.2, aquest editor és simple, inestable i requereix l'ús d'altres editors per fer escenaris realistes.

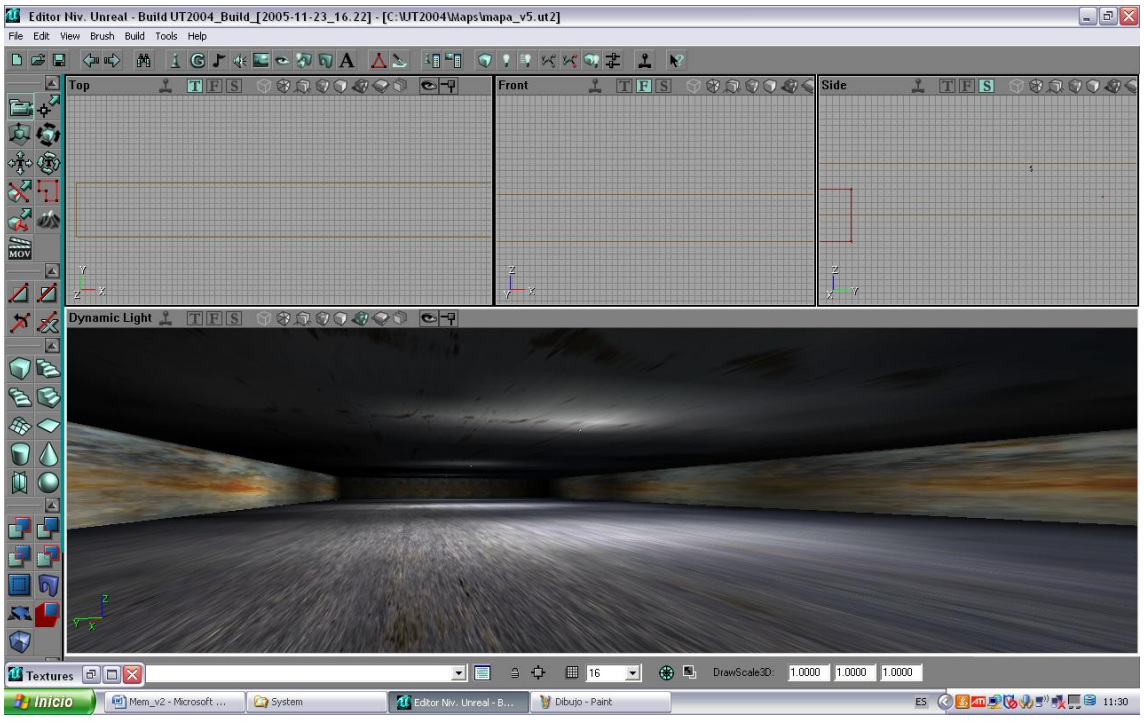


Figura 5.2 Entorn de treball de l'*UnrealEd*

El principal objectiu d'aquest editor 3D és fer de pont entre els editors avançats, com 3D Studio MAX i Blender, i el videojoc. *UnrealEd* només permet la creació de figures geomètriques de l'ordre de complexitat d'un con o d'una esfera. Totes aquelles geometries que necessitin un mínim modificació de les seves característiques, com per exemple edició de cares, han de ser dissenyades en un altre programa per ser importades posteriorment.

UnrealEd és una eina eficaç a l'hora d'ajudar al motor gràfic del videojoc a calcular la incidència de la il·luminació sobre els objectes estàtics de l'escenari. Un altre dels avantatges importants d'aquest editor és que permet *precompilar* el nivell durant el seu disseny, donant a l'usuari la possibilitat d'introduir-s'hi per tal d'observar com és en el videojoc el nivell que està dissenyant.

Els escenaris fets a partir d'aquesta eina han de complir tres requisits per tal de que puguin ser interpretats correctament pel motor del videojoc:

- *El nivell ha de ser tancat.* És a dir, ha d'estar envoltat per un cub o esfera que embolcallin l'escenari. Moltes vegades, a aquestes figures se'ls hi agrega una textura per tal d'obtenir un fons realista.

- *Ha de contenir com mínim una llum.* Per tal de que l'escenari sigui carregat pel joc cal posar-hi com a mínim una llum, encara que aquesta sigui el més feble possible i no permeti la distinció de cap objecte dins el videojoc.

- *Ha de disposar com a mínim d'un marcador PlayerStart.* Aquest marcador senyala quina és la posició inicial del personatge en la partida del videojoc. En cas que hi hagi més d'una, el joc és qui decideix on comença la partida. En el cas del simulador, aquest marcador indica la posició i orientació inicial de la càmera en la simulació.

Un dels problemes més insòlits que proporciona el treball amb l'*UnrealEd* succeeix quan es desa el nivell creat en format exportable per modificar-lo en un altra eina. L'editor modifica la posició del punt de referència origen o centroide [0,0,0] del que s'obtenen totes les distàncies relatives als vèrtex. L'aplicació calcula un centroide nou per facilitar la feina de càlcul pel motor gràfic, i així, modifica la posició relativa de cada punt de l'escenari. La figura 5.3 mostra com afecta aquesta modificació de l'escenari.

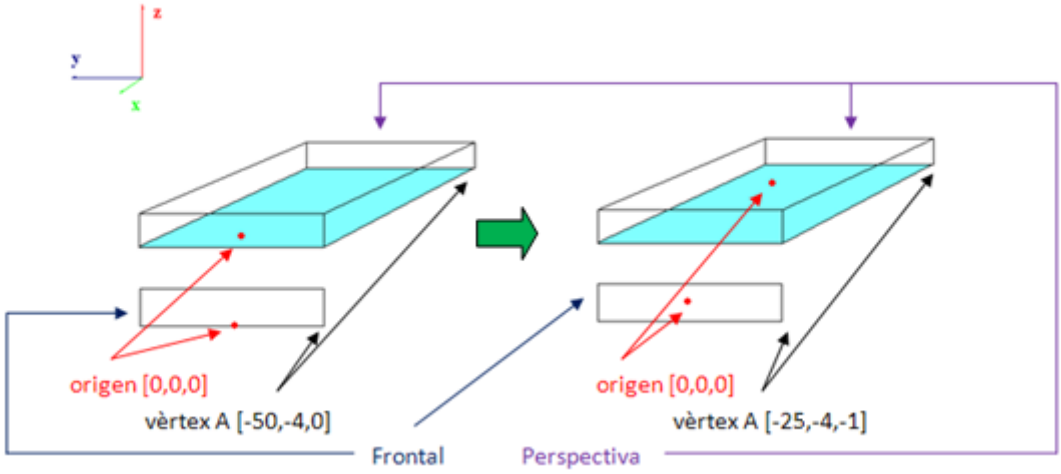


Figura 5.3 Canvi realitzat per *UnrealEd* del Centre Origen de l'escenari.

Si aquest fos l'únic problema seria fàcil de resoldre, però aquesta dificultat en el canvi de centroide cal sumar-hi la interpretació interna que fa el motor gràfic de les magnituds de distància (diferent a les d'*UnrealEd*). Aquest problema fa molt difícil determinar quina ha de ser la posició inicial del cotxe en l'escenari o, pitjor encara, com es veu en el capítol següent, fa dificilíssim determinar quina és la posició relativa del cotxe on ha d'anar ubicada la seva roda. Això s'ha de solucionar provant diferents posicions directament en el joc fins a obtenir la posició desitjada.

5.3 DISSENY DE L'ESCENARI DE SIMULACIÓ

Per tal de que la construcció de l'escenari en el que tindran lloc les simulacions sigui més simple i funcional possible, es decideix crear un cub allargat que emuli un "pàrking", amb el terra pla i suficientment llarg per aconseguir una gran velocitat amb el vehicle.

A la figura 5.4 es mostra l'escenari *mapa_v5* creat de manera ràpida i simple que ha estat utilitzat com a base de proves per a la construcció del vehicle.

Figura 5.4 Imatge de l'escenari mapa_v5

6. CREACIÓ DE VEHICLES A USARSIM

6.1 INTRODUCCIÓ

En aquest capítol es descriu breument com s'ha aconseguit la creació d'un cotxe d'estètica realista pel simulador.

El disseny del vehicle es pot separar en dues fases:

- Edició 3D del xassís i de les rodes.
- Unió de les rodes amb el xassís.

6.2 EDICIÓ 3D DEL XASSÍS I DE LES RODES

6.2.1 DISSENY 3D DEL XASSÍS

L'edició del xassís s'ha dut a terme mitjançant el programa Autodesk 3ds Max 8. A la figura 6.1 es pot veure com queda el vehicle sense textures després de la seva edició.

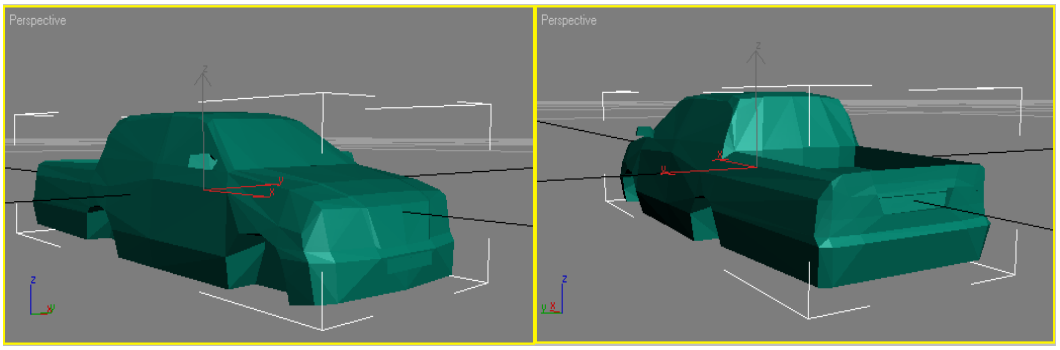


Figura 6.1 Imatges de la perspectiva del cotxe creat.

Per a que el cotxe pugui ser importat per l'*UnrealEd*, i així, ser suportat per l'*UnrealEngine*, ha d'estar format en la seva totalitat per polígons triangulars. De no ser així, el videojoc dóna un error d'interpretació de malles.

Un cop el xassís és importat a l'*UnrealEd*, aquest passa a ser considerat com a *StaticMesh*. Les malles estàtiques són figures geomètriques en que els seus vèrtexs no varien la posició relativa de l'origen de l'objecte. D'aquesta manera, el motor guarda a memòria una sola vegada la posició relativa dels seus vèrtex, i així, aconsegueix no repetir els càlculs innecessaris.

Com es pot veure a l'apartat *7.3 Implementació de la classe Cotxe* el factor d'escala que determina la mida dels objectes pot ser modificat posteriorment. Aquest fet fa que no s'hagi de buscar mesures de volum realistes en la fase d'edició.

6.2.2 APLICACIÓ DE LA TEXTURA AL XASSÍS

Figura 6.2 El xassís amb la textura de *Sedan*

La textura del xassís creat ha estat obtinguda de l'automòbil anomenat Sedan, una rèplica del Nissan Primera del '97, ja creat a USARSim. Cal dir que aquest vehicle va ser dissenyat, sense ser acabat, per algun dels usuaris de USARSim. Ha calgut modificar sensiblement la textura per aconseguir que s'adaptés al xassís (figura 6.2).

6.2.3 DISSENY DE LES RODES

El disseny de les rodes va seguir el mateix procés que el xassís. Dissenyades gràficament a partir de 3DS Max 8, la seva textura és extreta d'un cotxe d'USARSim anomenat *Hummer.* A la figura 6.3 es pot veure en *UnrealEd* una de les quatre rodes amb la textura ja aplicada.

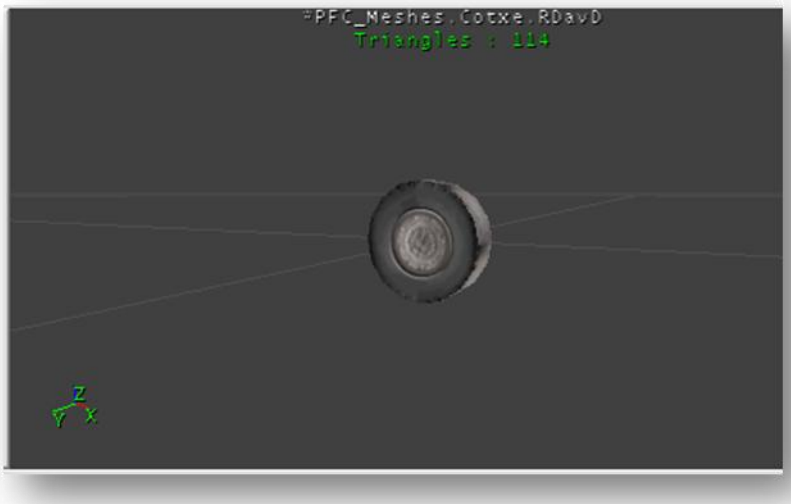


Figura 6.3 Roda dreta del davant

6.3 UNIÓ DE LES RODES AL XASSÍS

USARSim utilitza el sistema abstracte anomenat *Mission Package* per unir el conjunt de peces que formen un robot.

El *Mission Package* és un fitxer ".ini" que es fa servir com a contenidor de les fraccions que formen els robots. El fitxer està estructurat en un conjunt de seccions, on cada secció és un robot i dins s'especifiquen les diferents parts físiques que el conformen, les càmeres, els sensors, etc, que són unides a partir d'articulacions.

Una articulació és una representació abstracta d'unió de dues parts del robot: la part pare i la part filla. El pare és la part física *master* on va adherida la part filla *slave*. Tots els moviments del pare repercuteixen directament sobre la part filla però no a l'inrevés. L'articulació defineix els graus de llibertat de moviment que té la part filla sobre el pare. D'asqueta manera, es poden obtenir fins a tres graus de llibertat

especificant els eixos sobre els que es pot moure la peça. Les restriccions de l'angle màxim de moviment sobre l'eix són especificats a les classes pertanyents als objectes (veure l'apartat *7.3 Implementació de la classe Cotxe*)

El format de les articulacions és el següent:

<Nom><Classe><Escala><Tipus><PosicioP><EixP1><PSCOF><EixF1>

On:

Nom:	És el nom de l'articulació.
Classe:	És la classe en *UnrealScript* de la part filla.
Escala:	És el factor d'escala de la peça filla.
Tipus:	És el tipus d'articulació.
PosicióP:	La posició de la peça respecte el pare.
EixP1:	Eix del pare sobre el que es pot moure la peça filla.
PSCOF:	Posició de la peça filla respecte el seu origen.
EixF1:	Eix de la peça sobre el que és movible.

Per integrar les rodes (*slave*) al xassís (*master)*, cal saber la posició dels quatre forats dissenyats per ubicar les rodes al xassís. Donat el problema anunciat al capítol anterior, no hi ha manera de saber quina és la posició relativa d'una part de l'objecte. Per tal de saber on es troben els quatre forats per les rodes no hi ha altra alternativa que posar heurísticament nombres aproximats en el camp **PosicióP** i, a poc a poc, acostar-se a la localització exacta i desitjada. La figura 6.4 mostra la inclusió de la primera roda al xassís.

Figura 6.4 Adheriment de la primera roda al vehicle

A continuació es determinen quins són els eixos sobre els quals les rodes han de tenir possibilitat de mobilitat. La figura 6.5 mostra els eixos relatius a les rodes i al xassís per poder concretar quins són aquells que han de determinar els graus de llibertat.

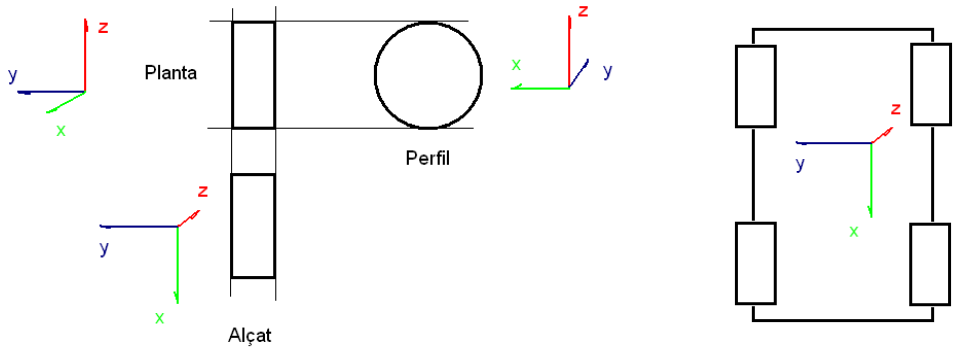


Figura 6.5 Planta, perfil i alçat de les rodes i alçat del cotxe

Com la roda directriu ha de poder fer dos moviments: rotacional, per fer avançar el cotxe, i direccional, per fer que giri; són dos els eixos de la roda que marcaran el moviment. USARSim però, també necessita saber quin són els eixos del pare (en el nostre cas el cotxe) que conflueixen amb els eixos del fill (roda). Són quatre els eixos a especificar: dos per la roda i dos pel cotxe.

L'eix que genera la rotació de les rodes és l'eix Y, tant de la roda com del xassís, i l'eix que genera el moviment direccional és Z, també en ambdós casos. Així, l'articulació de la roda davantera dreta queda expressat com:

JointParts=(PartName="RightFWheel",PartClass=class'USARModels.CotxeRodaDre',
DrawScale3D=(X=1.0,Y=1.0,Z=1.0),Parent="Cotxe",JointClass=class'KCarWheelJoint',
ParentPos=(Y=0.74,X=1.52,Z=-0.17),ParentAxis=(Z=1.0),ParentAxis2=(Y=1.0),
SelfPos=(Z=-0.0),SelfAxis=(Z=1.0),SelfAxis2=(Y=1.0))

Aquesta expressió és inclosa en la secció *Cotxe* creada en el *Mission Package.* Les expressions d'unió de la resta de rodes poden ser consultades al fitxer USARBot.ini inclòs al codi contingut en el CD annexat a aquest document.

7 EXTENSIONS A USARSIM

7.1 INTRODUCCIÓ

Aquest capítol explica quines han estat les modificacions necessàries fetes a USARSim per tal d'acomplir els objectius marcats a priori.

El desenvolupament d'aquestes millores s'ha dut a terme mitjançant el llenguatge programació interpretat i orientat a objecte de l'*UnrealEngine*: l'*UnrealScript*. Tot i ser un llenguatge força similar a C++, consta de certes peculiaritats específiques que s'han hagut d'aprendre prèviament per realitzar aquesta tasca.

Les extensions d'USARSim que s'han dut a terme s'enuncien a continuació i són detallades en aquest capítol:

- Implementació de la classe *MovimentCotxe.*
- Implementació de la classe *Cotxe.*
- Implementació de les classes de les rodes.
- Implementació i instal·lació dels sensors.

Tot i que aquestes millores introduïdes a USARSim manquen de plena funcionalitat per si soles, són el veritable fonament d'aquest projecte. L'aplicació IOCS és qui posteriorment se'n aprofita de totes elles per fer realitat el simulador de conducció enfocat a la investigació. (Capítol *8 IOCS*)

El codi de totes les classes és inclòs al CD que acompanya a aquest document.

7.2 IMPLEMENTACIÓ DE LA CLASSE *MOVIMENTCOTXE*

Una de les condicions indispensables a l'hora d'escollir el simulador de robots on poder crear el moviment realista d'un cotxe va ser que s'hi poguessin desenvolupar robots sobre rodes. USARSim permet implementar el comportament de vehicles d'aquest tipus a partir de classes implementades en *UnrealScript*.

Tots els robots amb rodes disponibles a USARSim són vehicles que es mouen utilitzant el sistema *Skid Steer System*, és a dir, utilitzant rodes de tracció independents respecte les de l'altre costat. Aquest tipus de vehicles es mouen aplicant tracció sobre les rodes d'un costat mentre que les de l'altre resten quietes, amb l'objectiu de girar; o traccionant les de les dues bandes per anar recte. La figura 7.1 mostra la màquina excavadora anomenada *Skid Loader*, clar exemple de vehicle que desenvolupa aquest tipus de moviment.

Figura 7.1 Màquina excavadora Skid Steer Loader

Evidentment, un cotxe no es mou tal i com ho fa aquesta màquina, per tant, s'ha implementat una nova *classe* que gestioni el comportament realista del cotxe.

Un cotxe es capaç de modificar la seva direcció mitjançant la variació de l'angle de gir de les rodes davanteres. Aquest tipus de moviment és definit pel Principi d'Ackerman, que determina quin és l'angle de les rodes directrius òptim per traçar una corba. Segons aquest principi, l'angle de gir de cada roda directriu forma un angle recte amb el radi del viratge quan l'eix del darrere també ho és (figura 7.2). Actualment però, els cotxes no utilitzen aquest sistema ja que no té en compte els efectes de la *dinàmica*[9] sobre el cotxe.

[9] Part de la física que estudia el moviment de les coses amb relació a les causes que el produeixen

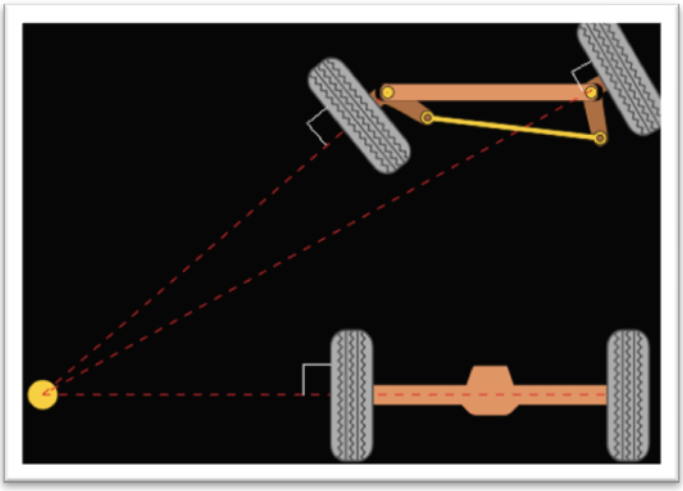


Figura 7.2
Angle descrit per les rodes directrius respecte el centre de la corba segons el principi d'Ackerman (Modificació de la imatge de Wikipedia [27])

Tot i que USARSim conté una classe anomenada *AckermanSteeredRobot* que intenta definir el moviment d'un vehicle a partir del principi d'Ackerman, la funcionalitat que descriu és incorrecta. El principal problema és que tal i com està implementada les rodes giren en temps diferents i de forma inestable. Per aquest mateix motiu, s'ha implementat la classe *MovimentCotxe* que defineix correctament el moviment cinemàtic d'un vehicle. El sistema de gir que implementa és conegut en el món de la competició de vehicles com a *Less Ackerman Angle*, és a dir, que les rodes descriuen el mateix angle sobre l'eix.

La classe *MovimentCotxe* no només és l'encarregada de gestionar el moviment de les rodes directrius del vehicle, sinó que també gestiona el moviment rotacional o de tracció de les rodes motrius i manté rectes les rodes posteriors. La figura 7.3 mostra l'algorisme principal d'aquesta classe.

Totes les comandes de control sobre el cotxe introduïdes per l'usuari al Contorlador (veure apartat 4.4.1.3) són rebudes i gestionades per la classe. En el desenvolupament d'aquesta tasca ens és molt útil l'estructura de dades creada per USARSim, anomenada USARBot. Aquesta estructura és creada per cada robot en escena, és accessible per totes les classes en execució i escolta en tot moment les comandes enviades per un Contorlador. Com el motor *UnrealEngine* actualitza l'estat de totes les classes i estructures a cada instant d'execució, si l'usuari envia una comanda de moviment, serà ràpidament interpretada per la classe *MovimentCotxe*.

El valor de modificació de l'angle de gir de les rodes directrius així de com la velocitat són enviades pel *Controlador*. Aquests valors són rebuts mitjançant

comandes que són interpretades per la classe. De fet, *MovimentCotxe* és capaç de gestionar dos tipus de comandes:

- Comanda normal o pura.
- Comanda normalitzada.

Comanda normal o pura:

La comanda normal o pura és aquella en que el controlador envia els valors de moviment en les unitats utilitzades pel motor físic, és a dir, en radiants (rad) per definir l'angle de gir, i radiants per segon (rad/seg) per la velocitat de rotació de les rodes.

Comanda Normalitzada:

La comanda normalitzada conté valors compresos en el interval [-100,100] per qualsevol tipus de moviment. Abans de ser interpretada pel motor físic cal realitzar una operació de conversió de tipus.

La gestió del moviment a baix nivell és implementada per les classes *GroundVehicle* i *KRobot* ja definides per USARSim, de les quals *MovimentCotxe* hereta les seves funcionalitats.

Procediment ProcessCarInput de la classe MovimentCotxe

1. Per cada roda del vehicle:
   - 1.1 Obtenció del seu tipus (directriu, motriu, pasiva, etc);
2. Per cada instant d'execució del motor;
   - 2.1 Si rep una comanda de moviment llavors;
     - 2.1.1 Si es una comanda de velocitat;
       - 2.1.1.1 Modifica la velicitat del vehicle;
     - 2.2.1 Sinó:
       - 2.2.1.1 Modifica l'angle de gir de les rodes;

Figura 7.3 Algorisme principal de la classe MovimentCotxe

7.3 IMPLEMENTACIÓ DE LA CLASSE *COTXE*

La *classe Cotxe* hereta la funcionalitat de la *classe MovimentCotxe* i és l'encarregada d'especificar les característiques tècniques del vehicle. Característiques com:

- Quantes rodes té el cotxe.
- Quines rodes giren i quin és el seu angle màxim.
- Quines són les rodes de tracció.
- Quina és la malla estàtica que actua com a xassís (creada a l'apartat 6.2).
- Quin és el factor d'escala del xassís
- Quines són les seves dimensions i pes.
- La velocitat de gir de les rodes.
- La velocitat màxima que pot aconseguir el vehicle.
- Etc.

Aquestes propietats s'especifiquen utilitzant els KParams, paràmetres interpretats pel motor físic Karma per calcular el comportament cinemàtic del cotxe. El problema d'aquests KParams és que no existeix cap glossari on s'especifiqui la funcionalitat de tots ells. Com són interpretats pel motor físic i aquest és transparent a l'usuari, no existeix una referència formal. Tot i que Epic Games ha publicat una llista exposant quins són els valors que han de comprendre alguns d'ells [16], en falten de molts d'altres.

Els valors assignats a cada paràmetre han estat avaluats un per un al principi, i conjuntament després, per tal de poder simular un moviment cinemàtic del cotxe realista. Així per exemple, si la fricció lateral del les rodes del vehicle amb el terra és massa baixa, el cotxe es desplaçarà relliscant per aquest com si ho fes sobre el gel, però, si per el contrari, la fricció lateral és massa alta, el cotxe no serà capaç de girar les rodes. D'aquest exemple es conclou que l'estabilitat del cotxe recau en un balanç de les seves especificacions.

La taula 7.1 fa un resum d'alguns dels KParams més importants l'hora de definir el comportament del cotxe. El camp *Valor òptim* que mostra la taula conté aquells valors que han estat calculats per tal d'aconseguir simular el comportament cinemàtic del cotxe implementat en aquest projecte. Això no vol dir que serveixin de referència per a la creació d'altres robots, vehicles, etc.

KPARAM	Descripcció	Valor òptim
DrawScale	Factor d'escala pel qual UnrealEngine dibuixa l'StaticMesh pertanyent al vehicle	4.762
SteerSpeed	Velocitat de gir de les rodes del vehicle	Sí
KMass	Massa relativa del vehicle respecte el volum	1.0
KInertiaTensor(0)	Dificultat de l'objecte a donar voltes sobre si mateix al voltant de Z	0.01
KInertiaTensor(3)	Dificultat de l'objecte a donar voltes sobre si mateix al voltant de X	0.02
KInertiaTensor(5)	Dificultat de l'objecte a donar voltes sobre si mateix al voltant de Y	0.04
KLinearDamping	Quantitat de força aplicada a reduir el moviment linear del Actor causant l'arrossegament transaccional.	0.5
KCOMOffset	Posició del centre de massa de l'objecte relatiu al centre del StaticMesh	(X=0.044,Y=0,Z=0)

Taula 7.1 Resum de paràmetres que descriuen les característiques tècniques del vehicle

7.4 IMPLEMENTACIÓ DE LES CLASSES DE LES RODES

Les rodes, així com tots els objectes dinàmics que formen una escena en *UnrealEngine* són objectes pertanyents a una classe que defineix el seu comportament. Per a la creació de les rodes del cotxe ha fet falta el desenvolupament de dues classes diferents: *CotxeRodaDre* i *CotxeRodaEsq*.

Si bé és cert que les rodes del davant i les del darrera no defineixen el mateix tipus de moviment: ja que les rodes posteriors no giren; el control total del moviment de les quatre rodes és definit i gestionat per la classe *MovimentCotxe*.

D'altra banda, cada classe objecte o *Actor* té associada una malla estàtica que especifica quina és la geometria física de l'objecte quan és carregat pel motor del videojoc. Cada *Static Mesh* està composta per una geometria 3D, formada per cares, arestes i punts; i per una textura. Si tenim en compte que aquesta textura, encarregada de donar una aparença realista a la roda, té el dibuix d'una llanta només en un dels costats, veiem clarament que necessitem dos tipus d'*StaticMesh*. Una servirà per les rodes de la dreta i l'altre per les de l'esquerra; totes dues deixant veure la llanta i donant un toc més realista al cotxe. La figura 7.4 mostra aquest fet.

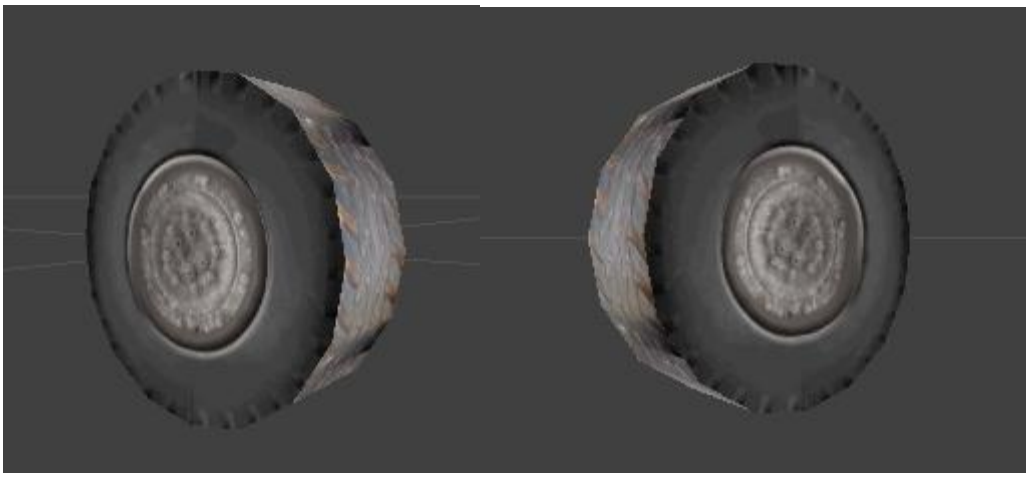


Figura 7.4
A l'esquerra: *StaticMesh* de les rodes davantera i posterior esquerres.
A la dreta: *StaticMesh* de les rodes davantera i posterior dretes.

La única diferenciació entre les dues classes es troba, per tant, en el camp que associa l'*StaticMesh* a cada una:

Per la roda esquerra:
StaticMesh=StaticMesh'PFC_Meshes.Cotxe.RDavE'

Per a la roda dreta:
StaticMesh=StaticMesh'PFC_Meshes.Cotxe.RDavD'

7.5 IMPLEMENTACIÓ I INSTAL·LACIÓ DE SENSORS

7.5.1 IMPLEMENTACIÓ I INSTAL·LACIÓ DE *SENSORPOSICIÓ*

Tal i com ja s'ha explicat en el capítol primer d'aquest document, una de les condicions necessàries que ha d'acomplir el simulador per ser una eina eficaç en la validació de sistemes ADAS és que sigui capaç d'obtenir les dades GT de la simulació.

Una de les grans virtuts que té USARSim és que els sensors validats instal·lables als robots descriuen el mateix comportament que els reals. Això és una avantatge molt gran a l'hora de provar algorismes de control de robots a partir de la informació rebuda pels sensors que, com els de la realitat, són incapaços de descriure

perfectament l'entorn de simulació degut a les deficiències pràctiques d'adquisició de dades.

El GT són seqüències de dades que descriuen exactament l'entorn i per tant, els sensors inclosos per USARSim no són útils per a desenvolupar aquesta tasca. Per aquest mateix motiu s'implementa la classe *SensorPosició*, capaç d'obtenir a cada instant d'execució d'*UnrealEngine* la posició i orientació del cotxe.

Aquest sensor pregunta a cada instant quina és la posició i orientació del vehicle dins l'entorn de simulació i envia les dades a través de la xarxa per a que puguin ser rebudes pel *Controlador*. Posteriorment, aquesta informació pot ser molt útil en la validació d'ADAS. (p.e. es pot determinar quina és la velocitat del cotxe a cada instant, la distància exacte respecte certs objectes en l'entorn, etc).

La instal·lació dels sensors al vehicle es du a terme de la mateixa manera que les rodes s'uneixen amb el xassís. Al fitxer que descriu el *Mission Package* anomenat "USARBot.ini" s'indica quin és el nom de la classe del sensor a incorporar i quin és el punt respecte el centre del vehicle on es vol instal·lar.

7.5.2 INSTAL·LACIÓ DE LES CÀMERES

Les càmeres implementades a USARSim són sensors instal·lats als vehicles. Aquests sensors descriuen un punt origen i una orientació ens els que l'*UnrealEngine* disposa la seva càmera de visió de l'escena.

Les càmeres són adherides al cotxe seguint mateix procés que els sensors. Només es diferencien en el fet que les càmeres a més de la posició, cal indicar també quina és la seva orientació. D'aquesta manera, es poden agregar fins a quatre punts de visió simultània per vehicle. Aquests punts de visió no cal que estiguin en contacte amb el xassís del cotxe, simplement són ubicats a una distància constant del punt origen. En la figura 7.5 es pot veure la visió simultània de les quatre càmeres que porta equipades el vehicle creat.

Figura 7.5 Vista de les quatre càmeres equipades en el cotxe

A la figura 7.6 es pot veure la representació en 3D d'una càmera realista situada en uns dels punts de visió definits en el cotxe.

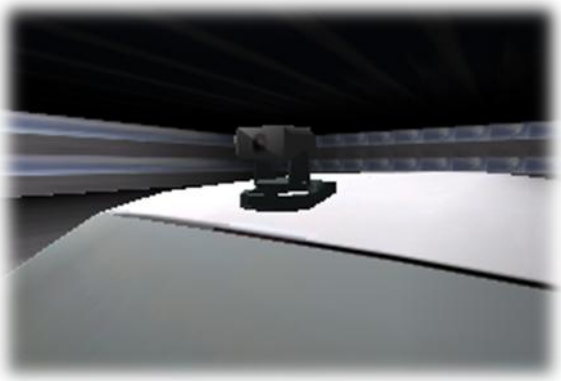


Figura 7.6 Representació 3D d'una càmera sobre el vehicle

8 DESENVOLUPAMENT DE IOCS

8.1 INTRODUCCIÓ

Un cop fetes totes les modificacions necessàries a USARSim per construir els fonaments del projecte, es desenvolupa l'aplicació IOCS encarregada d'obtenir els beneficis d'aquestes extensions realitzades i donar forma al simulador de conducció enfocat a la investigació.

A la fase d'adquisició d'informació per a la implementació del programa, es descobreix SimpleUI. Aquesta aplicació és un Controlador d'USARSim implementat en C++ sobre Windows capaç, simplement, d'obtenir les imatges de l'*UnrealClient* com a retroalimentació de vídeo. Utilitzant la llibreria Detours de Microsoft [18], aquesta interfície accedeix al "back-buffer" de DirectX[10] [23] d'on les extreu en format *bitmap*[11].

Tot i tenir una funcionalitat molt limitada que no permetia acomplir cap dels objectius marcats a priori, SimpleUI va ser d'especial interès pel fet de poder obtenir imatges directament de l'*UnrealClient*. Aquesta capacitat podria ser de gran utilitat en el processament d'imatges per part d'un ADAS basat en Visió per Computador.

Així doncs, a partir d'aquest petit projecte naix IOCS, un programa implementat en C++ que incorpora les funcionalitats que s'enuncien a continuació:

- ***Funcionalitat de Controlador d'USARSim*.** Rebuda d'informació dels sensors del vehicle i enviament de les comandes de control de moviment.
- ***Protocol d'arrencada i aturada automàtic d'USARSim.***
- ***Càrrega de l'escenari escollit per l'usuari.*** Tot i tractar-se d'una part pertanyent al *UnrealServer* i, en principi, inaccessible per l'usuari d'USARSim, IOCS permet a l'investigador seleccionar l'escenari de simulació.
- ***Interfície per a la creació i control de la simulació per part de l'usuari.***
- ***Presentació de la informació dels sensors en temps real a través de la interfície.***

[10] Tecnologia utilitzada per Microsoft Windows en la gestió gràfica, generalment referent a videojocs

[11] Format en el que es guarda la informació de cada píxel de la imatge

- ***Emmagatzematge ordenat i automàtic a disc de la informació rebuda dels sensors.***
- ***Adquisició i emmagatzematge automàtic de les imatges.***
- ***Sistema de Simulacions Automàtiques.***

En aquest capítol es descriuen breument quines han estat les principals decisions de disseny i com s'han implementat en el desenvolupament de IOCS. El codi complet d'aquest programa és adjuntat al CD annexat a aquest document.

8.2 IOCS COM A CONTROLADOR D'USARSIM

El primer objectiu un cop fetes les modificacions a USARSim va ser donar plena funcionalitat de Controlador a IOCS. Per fer-ho, calia que l'aplicació fos capaç connectar mitjançant un socket amb el sector *Servidor* d'USARSim i, implementar un protocol d'arrencada i d'aturada de les dues parts que formen que el formen: l'*UnrealServer* i l'*UnrealClient*.

8.2.1 CONNEXIÓ AMB EL SERVIDOR

En la inicialització del programa es crea un socket encarregat de gestionar l'intercanvi de missatges entre l'aplicació i USARSim. Aquest socket gestiona el pas de missatges en ambdós sentits: del programa al *Servidor* i del *Servidor* al programa. En un sentit, els sensors envien les seves dades obtingudes al programa i, en l'altre, l'aplicació envia les comandes de moviment del cotxe introduïdes per l'usuari.

La figura 8.1 mostra el pas de l'arquitectura original d'USARSim a l'estesa, a partir de la funcionalitat de IOCS com a *Controlador* i les extensions fetes amb anterioritat.

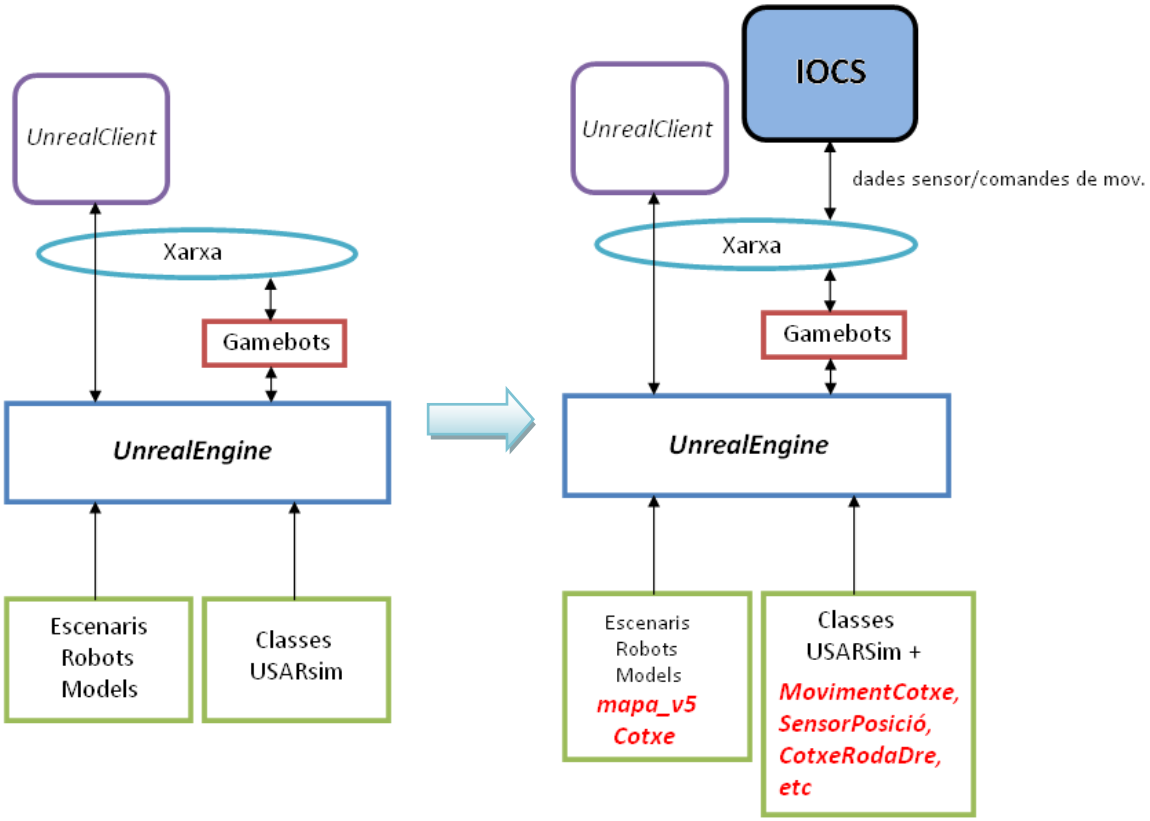


Figura 8.1

A l'esquerra: Estructura original d'USARSim

A la dreta: Estructura estesa d'USARSim amb IOCS com a *Controlador*

Informació rebuda pel programa:

1. La primera informació rebuda és sempre enviada per l'*UnrealEngine*. El motor del videojoc envia un cop la informació com si es tractés d'una partida de Unreal Tournament 2004. La informació està compresa pel tipus de partida, el nivell i el límit de temps de que es disposa (0 per temps il·limitat).

   "NFO {Gametype BotDeathMatch} {Level mapa_v4} {TimeLimit 0}"

2. A partir d'aquest moment el *SensorPosició* (capítol *7.5.1 Implementació i instal·lació de SensorPosició*) envia les dades de posició i orientació del vehicle. A continuació es mostra un exemple de missatge rebut pel sensor posició:

"SEN {Type SensorPosicio} {Name SP} {Location 0.07,1.00,0.30} {Orientation 3.14,1.40,3.14}"

3. La informació del *SensorPosició* és intercalada amb les dades generades per USARSim. Aquesta informació és útil per saber quin és el tipus de vehicle simulat, el temps de simulació, l'angle de gir de les rodes del vehicle, l'activació i intensitat dels llums del cotxe i si la visió de la simulació es realitza des de les càmeres del cotxe (1) o des de l'escenari (-1) :

"STA {Type GroundVehicle} {Time 17.22} {FrontSteer -0.0041} {RearSteer 0.0000} {LightToggle False} {LightIntensity 0} {Battery 1199} {View -1}"

Informació enviada pel programa

El programa és l'encarregat de gestionar el moviment del vehicle segons les instruccions de l'usuari. Per fer-ho, manté a cada instant quin és l'angle de gir i la velocitat de tracció de les rodes en la simulació. Quan l'usuari introdueix un canvi en el moviment del cotxe, el programa calcula els nous valors en l'angle de gir i velocitat del vehicle. Tot seguit, aquests valors són enviats pel socket i, posteriorment, interpretats per la classe *MovimentCotxe* a USARSim, qui realment s'encarrega del moviment del cotxe. L'estructura de la comanda és expressada a continuació:

DRIVE {Speed <valor de velocitat>**} {FrontSteer** <angle de gir>**}**

8.2.2 PROTOCOL D'ARRENCADA I D'ATURADA AUTOMÀTIC

Per tal d'evitar que l'usuari hagi d'executar i aturar l'*UnrealServer* i l'*UnrealClient* cada cop que vol dur a terme una simulació, IOCS gestiona un protocol d'arrencada i, evidentment, també d'aturada.

Quan l'usuari inicia l'aplicació, el programa executa el *Servidor* i guarda la seva identitat (*processId*). Un cop aquest ha acabat el seu procés d'inicialització, IOCS executa el *Client* desant també la identitat. Quan l'usuari tanca l'aplicació, es realitza una cerca a la llista de processos actius del sistema operatiu a partir de les identitats de cada un per tal d'aturar-los.

8.3 INTERFÍCIE DE CREACIÓ I CONTROL DE LA SIMULACIÓ

Per facilitar la feina a l'usuari a l'hora de poder crear i controlar les simulacions es decideix fer una interfície simple en forma de diàlegs de Windows. La figura 8.2 mostra les dues finestres de les que consta IOCS: Menú d'Inici i Interfície de Control.

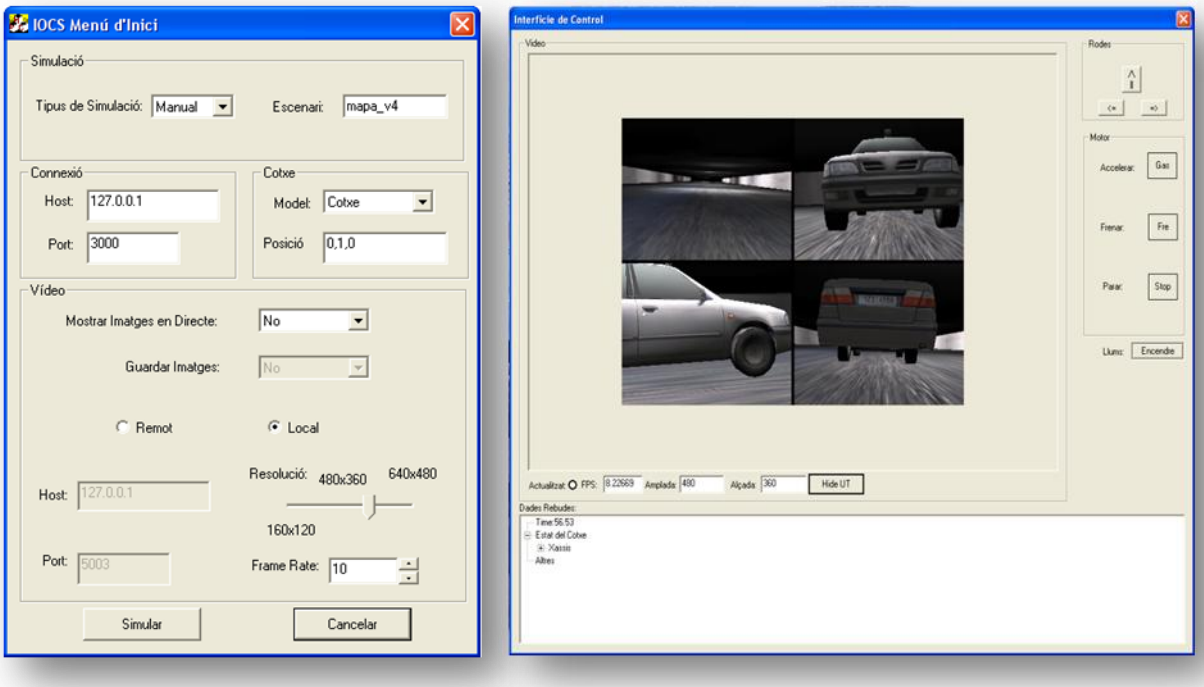


Figura 8.2

A l'esquerra: Finestra Menú d'Inici
A la dreta: Finestra Interfície de Control

Menú d'Inici

Aquesta finestra fou dissenyada amb l'objectiu de facilitar a l'usuari l'elecció de les característiques de la simulació. En aquest diàleg es poden escollir paràmetres com: el tipus de simulació, l'escenari, el cotxe, la posició inicial d'aquest, els paràmetres de visualització de vídeo, etc; totes elles explicades detalladament al manual d'usuari de l'aplicació contingut en l'apèndix primer d'aquest document.

En el desenvolupament d'aquesta finestra es van dur a terme algunes decisions de disseny importants. A continuació se'n descriuen tres:

- *Camp d'elecció d'Escenari com a text lliure*. L'usuari pot escriure el nom de l'escenari en el que vol realitzar la simulació en comptes de seleccionar entre un nombre limitat. Això és implementat així perquè es preveu que els investigadors testin els seus sistemes en varis entorns de simulació diferents. D'aquesta manera, si l'usuari crea un nou escenari on dur a terme la simulació, l'aplicació és capaç de carregar-lo automàticament.

- *Modificació d'UnrealServer.* Com s'ha parlat en el capítol 4.4, l'escenari és carregat pel sector *Servidor* que no és modificable per l'usuari. Per solucionar aquest fet, quan l'investigador introdueix el nom d'un escenari per a la simulació: primer es comprova l'existència d'aquest i després es sobreescriu el fitxer d'inici d'*UnrealServer* (temporalment poc costós). A continuació, mitjançant el protocol d'arrencada, IOCS carrega el nou *Servidor* per generar l'escenari de simulació desitjat.

- *Paràmetres de Vídeo en format Estàndard.* Les càmeres utilitzades en els projectes de Visió per Computador del CVC utilitzen una resolució estàndard de 640x480 i un *frame-rate* major a 20 imatges per segon. Aquestes possibilitats són integrades a IOCS per tal d'obtenir característiques de vídeo compatibles amb els projectes en el món real. Però, com IOCS pot ser utilitzat per d'altres persones que potser no necessiten les mateixes propietats de vídeo, és l'usuari qui decideix la resolució (de 160x120 fins a 640x480) i el *frame-rate* (de 1 a 37 imatges per segon).

Interfície de Control

La Interfície de Control permet a l'usuari de IOCS dur a terme el control de la simulació. Des d'aquest diàleg l'investigador controla el moviment del vehicle, observa en temps real les dades rebudes dels sensors, les imatges i condicions de vídeo que s'obtenen.

Dues de les decisions de disseny més importants en el desenvolupament d'aquesta interfície van ser les següents:

- *Possibilitat de veure la simulació directament des de l'UnrealClient sense obtenir les imatges.* En la figura 8.1 es pot veure com en la Interfície de Control es mostren les imatges rebudes del *Client*. Aconseguir-ho fa disminuir el rendiment de simulació. Per això, l'usuari pot desactivar la visió i guanyar en eficiència de computació. La simulació pot ser seguida directament des del *UnrealClient* tal i com mostra la figura 8.3.

- Un dels objectius prioritaris definit en els primers passos d'aquest projecte va ser aconseguir que el simulador fos capaç de mostrar la informació obtinguda de la simulació en temps real. A la part de sota de la Interfície de Control es mostren en temps real els valors GT rebuts dels sensors instal·lats en el vehicle. Valors com: el temps de simulació, posició i orientació del vehicle, angle descrit per les rodes davanteres, etc.

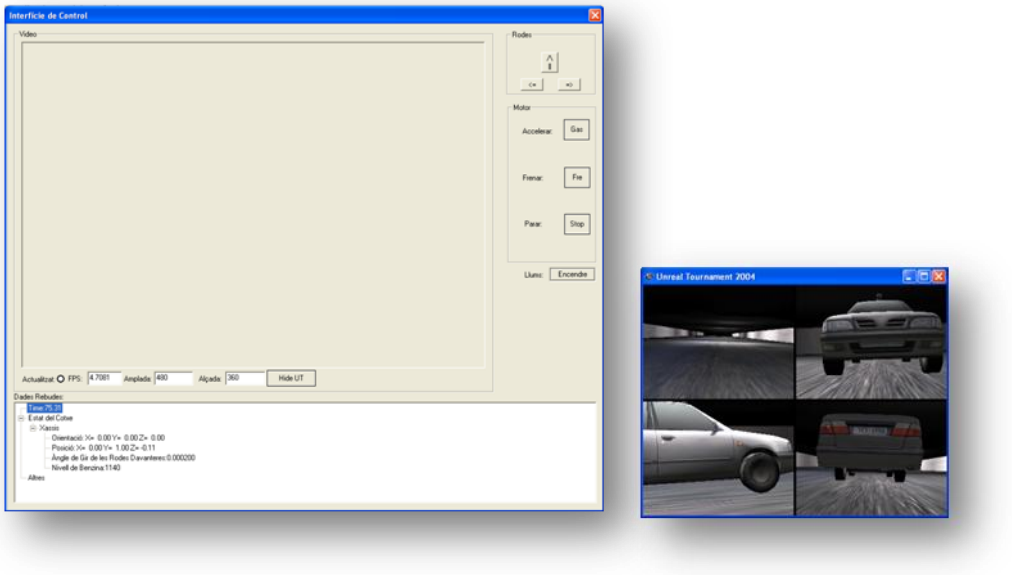


Figura 8.3

A l'esquerra: La Interfície de Control no mostra les imatges de la simulació

A la dreta: L'usuari pot seguir en temps real la simulació des de l'*UnrealClient*

8.4 FUNCIONALITATS AFEGIDES PER A VALIDACIÓ EN ADAS

Arribats aquest punt, el projecte satisfeia molts dels objectius marcats a priori. Tot i això, amb la finalitat de que IOCS fos una eina que realment facilités el procés de validació en ADAS, s'introdueixen noves funcionalitats al simulador.

Aquestes noves utilitats del simulador es poden separar en dos grups:

- Gestió de la informació de les simulacions i
- Sistema de Simulacions Automàtiques.

8.4.1 GESTIÓ DE LA INFORMACIÓ DE LES SIMULACIONS

Un investigador que utilitza IOCS amb l'objectiu de validar un sistema d'assistència en la conducció ha de ser capaç de poder estudiar la informació generada per la simulació. L'enginyer pot necessitar saber, per exemple, quines han estat les posicions i orientacions del cotxe a cada instant, o quins els moviments del vehicle en la simulació. A més, és probable que l'usuari necessiti contrastar les dades de la simulació actual amb totes les que ha realitzat anteriorment. Per tal d'acomplir aquest objectiu, IOCS desenvolupa un gestor de informació.

Aquest gestor és capaç d'ordenar i d'emmagatzemar tota la informació generada per cada una de les simulacions (posicions i orientacions del vehicle, totes les imatges de la simulació, etc) . Per fer-ho, el programa crea una directori *Experiments* a la carpeta d'USARSim. Dins d'*Experiments* i per cada simulació, es crea un directori que conté tota la informació obtinguda. El nom d'aquestes carpetes, el contingut de la carpeta *Imatges* i la informació dels fitxers és explicada detalladament en el manual d'usuari de l'aplicació a l'apèndix d'aquest document. La figura 8.4 mostra com es desa la informació de cada simulació.

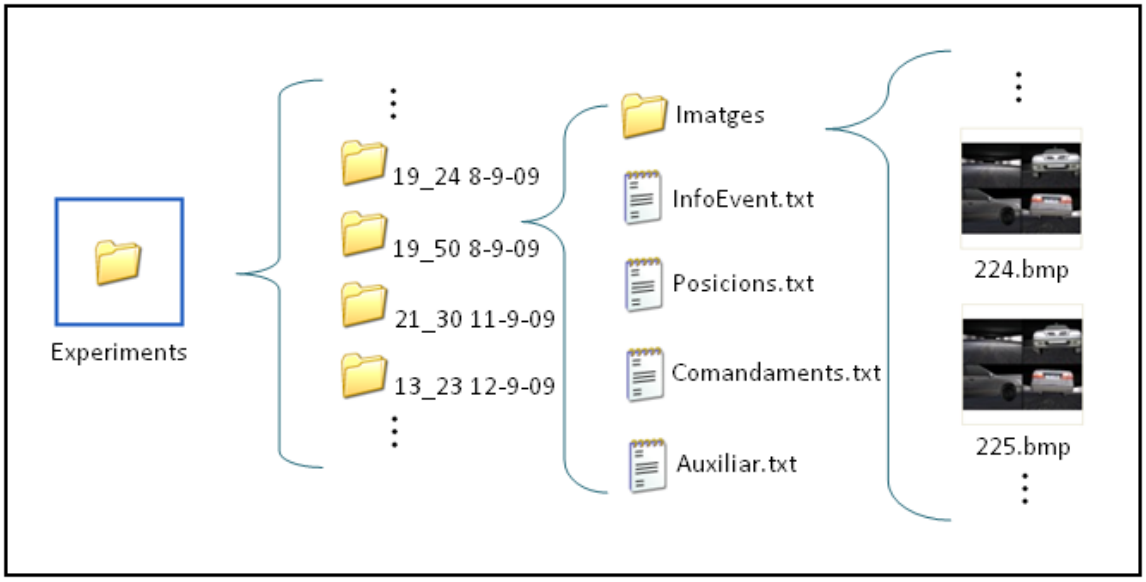


Figura 8.4 Disposició a disc de la informació de les simulacions

8.4.2 SISTEMA DE SIMULACIONS AUTOMÀTIQUES

És evident que la validació de qualsevol tipus de sistema, ja sigui relacionat amb la conducció o no, ha de ser estudiat vàries vegades per extreure'n conclusions determinants. En el món real, la repetició d'experiments en les mateixes condicions mai repeteixen un resultat idèntic. En el cas de la conducció, si un investigador prova un sistema d'assistència a la conducció vàries vegades en el mateix recorregut i amb les mateixes condicions, mai no n'extraurà els mateixos resultats exactes. IOCS incorpora un sistema de repeticions de simulacions automàtic capaç de simular aquestes condicions del món real.

El Sistema de Simulacions Automàtiques, tal i com se'n diu, utilitza la informació dels moviments del cotxe en una simulació anterior per repetir-los en l'actual. Aquests valors són enviats de nou a *l'UnrealEngine* que, donada la seva estructura interna, els interpreta de forma lleugerament diferent. Si bé és cert que la moció del cotxe no és exactament la mateixa, el moviment és extremadament semblant.

A cada simulació el gestor de la informació del programa guarda un fitxer anomenat Auxiliar.txt. Quan l'usuari de IOCS selecciona a *Menú d'Inici* que desitja realitzar una simulació automàtica, el programa obre una finestra per tal que pugui seleccionar el fitxer auxiliar de la simulació que vol repetir. Evidentment, IOCS desenvolupa una gestió d'errors en cas que l'usuari seleccioni un altre tipus de fitxer. Quan l'usuari l'ha seleccionat, mitjançant la utilització de gestió de memòria dinàmica, el programa llegeix quin és el tipus de moviment que ha de seguir el cotxe a cada instant. En l'execució de la simulació, a cada moment, IOCS comprova si el cotxe ha de realitzar algun moviment i, de ser així, envia la comanda oportuna pel socket per a que sigui interpretada de nou per l'*UnrealEngine*.

La creació d'aquesta nova funcionalitat permet extreure'n tres conclusions:

1. La informació que obté *UnrealEngine* és interpretada lliurement a cada instant. Encara que una mateixa comanda de moviment d'un vehicle sigui enviada en el mateix instant de temps i sota les mateixes condicions en dues simulacions diferents (impossible al món real), el motor del videojoc no té perquè interpretar-les igual, depèn de factors interns que intenten emular la realitat. Això és d'extremada importància en la validació en ADAS. Si el sistema és exposat dues vegades en el món real sota les mateixes condicions mai no s'obtindran els mateixos resultats. IOCS és capaç d'emular aquest comportament.

2. El *Sistema de Simulacions Automàtiques* permet estudiar com influeixen les condicions del cotxe i de l'entorn en el comportament del vehicle. Aquest sistema pot realitzar la mateixa simulació en escenaris i amb vehicles diferents. Així doncs, si un dels sistemes en ADAS ha de servir per assistir la frenada d'un vehicle, IOCS permet estudiar la distància de frenada depenent, per exemple, del pes del vehicle o del grau de fregament sobre el terra.

3. Si l'investigador de IOCS volgués repetir exactament una simulació, pot generar el vídeo de qualsevol simulació a partir de les imatges desades automàticament per l'aplicació a disc. La informació serà exactament la mateixa que la de la simulació repetida.

9 CONCLUSIONS I TREBALL FUTUR

En aquest capítol s'exposen les conclusions extretes de la realització d'aquest projecte així com possibles línies de treball futur.

9.1 CONCLUSIONS

El desenvolupament d'aquest projecte ha permès assimilar els avantatges que proporciona la simulació en el progrés de confecció d'un propòsit o designi.

El simulador és una eina molt potent que facilita la presa de decisions importants ja en la fase de disseny d'un projecte. Poder "provar abans de crear" no només permet fer estudis de viabilitat d'idees, sinó que a més, acostuma a garantir els millors resultats al final del treball.

Però, per tal de que un simulador pugui ser utilitzat en el modelatge d'esdeveniments reals i ser enfocat a la investigació, aquest ha de complir dos requisits molt importants: ha de fiançar el màxim realisme possible en allò que simula i, d'altra banda, garantir l'obtenció d'informació necessària per poder extreure'n conclusions de la simulació.

En aquest projecte s'ha dissenyat i implementat un simulador de conducció enfocat a la investigació capaç d'acomplir els dos requisits indispensables enunciats anteriorment. IOCS, tal i com així s'anomena, és una eina nascuda amb l'objectiu de facilitar als investigadors la creació i l'estudi d'esdeveniments virtuals en el camp de la conducció de vehicles.

El fet d'haver estat creat a partir del simulador de robots USARSim, que utilitza el motor d'un videojoc actual, proporciona a IOCS una alta qualitat gràfica en les simulacions i una consistència física veraç, validada per institucions d'investigació en robòtica reconegudes internacionalment.

IOCS permet simular entorns de conducció realistes tot obtenint, simultàniament, les dades provinents dels dispositius sensorials instal·lats en un vehicle i el seu "Ground Truth" respectiu. Per fer-ho realitat, s'ha desenvolupat un automòbil a USARSim, d'estètica i comportament realistes, capaç d'incorporar els sensors creats, aprovats i emprats per la comunitat internacional de recerca en robòtica, així com el sensor GT de posicionament implementat en aquest projecte.

Aquesta funcionalitat del simulador és molt útil per a facilitar el procés de validació de sistemes d'assistència a la conducció. Els ADAS perceben i modelen l'entorn a partir de les dades obtingudes pels sensors instal·lats en un vehicle (sensors disponibles a USARSim), però per a poder quantificar l'error del sistema, són necessàries les dades que defineixen l'entorn de conducció de forma precisa. Com IOCS és capaç de generar i emmagatzemar de forma automàtica aquesta informació, els investigadors podran evitar les llargues hores d'anotacions i imprecisions provocades pel procés manual de validació de sistemes ADAS actual, i guanyar en eficiència i exactitud en la realització d'aquesta tasca.

A continuació, s'enuncien totes les tasques que han estat desenvolupades en aquest projecte per a la creació de IOCS:

- Recerca del simulador de robots per ser utilitzat com a base del projecte. USARsim ha estat seleccionat donades les seves característiques propícies per a la creació del simulador de conducció.
- Disseny i construcció d'un escenari a USARSim on dur a terme les simulacions.
- Creació d'un vehicle d'estètica realista mitjançant l'ús de 3D Studio Max i *UnrealEd*.
- Desenvolupament de les classes en *UnrealScript*, afegides a USARSim, que implementen el moviment realista d'un vehicle.
- Implementació i instal·lació del sensor de posicionament capaç d'obtenir la informació "Ground Truth" de la posició i orientació del cotxe.
- Programació en C++ d'una aplicació que permet a l'investigador seleccionar les característiques de simulació i controlar-la.
- Creació i instal·lació d'un protocol d'arrencada i d'aturada d'USARSim gestionat de forma automàtica per part de l'aplicació.
- Implementació de les comandes de moviment del vehicle accessibles a través de la interfície d'usuari de l'aplicació i interpretades pel cotxe.
- Recepció i visualització en temps real i per la interfície de les dades GT de posicionament, orientació, angle de gir de les rodes i benzina restant referents al vehicle de la simulació.
- Creació d'un gestor de la informació encarregat d'emmagatzemar i ordenar automàticament tota la informació generada per les simulacions.

- Capacitat de guardar les imatges que conformen el vídeo de la simulació amb l'objectiu de que puguin ser processades per ADAS basats en Visió per Computador.

- Creació del *Sistema de Simulació Automàtica* capaç de reinterpretar simulacions realitzades amb anterioritat i facilitar la validació de sistemes en ADAS.

9.2 TREBALL FUTUR

Donades les seves funcionalitats, les possibles línies de treball futur a partir d'aquest projecte són extenses. Els investigadors poden modificar i millorar IOCS per tal de dur a terme les seves pròpies investigacions en entorns de conducció. Les prestacions actuals de l'eina poden ser vistes com la base de diferents aplicacions útils per a cada grup de recerca.

A continuació, es citen algunes investigacions futuribles dutes a terme a partir d'extensions del simulador creat en aquest projecte:

- Es podrien dissenyar algorismes de control en temps real del moviment del cotxe amb l'objectiu, per exemple, de que fos capaç d'esquivar obstacles autònomament. La possibilitat d'equipar el cotxe amb tot tipus sensors: càmeres, sensors de proximitat, sonars, GPS, etc, permetrien rebre les dades de distància dels obstacles a través del simulador. Aquesta informació seria analitzada en temps real per l'algorisme, que modificaria la trajectòria del cotxe per no xocar.

- Actualment es poden introduir varis cotxes a la mateixa simulació utilitzant una aplicació IOCS per cada vehicle. Com que el simulador permet generar els moviments del cotxe de forma automàtica, l'investigador podria encarregar-se del moviment d'un vehicle, i així, veure la simulació des de totes les càmeres dels automòbils de forma simultània i obtindre en temps real la informació generada per tots ells. Serviria, per exemple, per estudiar si el sistema ADAS d'un cotxe, que deté amb seguretat un vehicle en cas de frenada brusca del cotxe que va al davant, detecta la resta de cotxes de manera correcta, frena amb la intensitat adequada per no xocar, etc. És a dir, si el sistema d'assistència funciona correctament.

- Com les càmeres del simulador són dissenyades de la mateixa manera que els sensors d'USARSim, seria factible implementar una càmera que determinés la proximitat dels objectes i renderitzés cadascun d'un color diferent depenent de la seva proximitat.

- Donat que l'obtenció de les imatges per part del simulador és píxel a píxel, es podrien dissenyar LUTs per tal de modificar les imatges d'entrada i, per exemple, crear càmeres amb soroll que emulessin les reals. Donat que el simulador permet veure la simulació directament des de l'*UnrealClient*, l'investigador seria capaç de veure la imatge real i la modificada simultàniament.

- Les especificacions tècniques dels vehicles són modificables. Així, l'usuari de IOCS pot variar de manera fàcil les característiques del cotxe com el pes, la potència, el centre de gravetat, la fricció de les rodes amb el terra, la velocitat de gir de les rodes, etc. Aquest fet permetria estudiar el comportament del vehicle segons les seves particularitats, i definir quins aspectes són determinants en el comportament òptim dels cotxes a la realitat.

I així podríem no acabar mai d'exposar nous casos d'estudi utilitzant el simulador. Tot dependrà de la imaginació i l'enginy dels investigadors que l'utilitzin aquesta eina.

Definitivament podem conclure que, donades les funcionalitats contingudes i les futures possibilitats que permet, IOCS és una nova porta oberta per a la investigació en entorns de conducció a partir de la simulació.

APÈNDIX

IOCS: MANUAL D'USUARI

A INTRODUCCIÓ

A.1 IOCS

IOCS – *Investigation Oriented Car Simulator* – és una eina d'investigació en el camp de la conducció de vehicles a partir de la simulació desenvolupada sobre el simulador de robots USARSim.

Aquesta aplicació té la finalitat de facilitar la validació de sistemes d'assistència en la conducció (ADAS, de l'anglès Advanced Driving Assistance System). És a dir, permet als usuaris estudiar protocols de control de vehicles de forma fàcil, realista, ràpida, segura i, sobretot, econòmica.

IOCS és una programa capaç de simular entorns realistes de conducció adquirint tota la informació que es desprèn dels sensors d'un vehicle. Mitjançant la creació d'una simulació en un entorn real de conducció, l'aplicació s'encarrega de guardar tota la informació necessària pels investigadors de forma automàtica; fins i tot, les imatges de les càmeres del cotxe.

Els cotxes, els sensors, les càmeres, els protocols de control del vehicle, etc; són definits en simulador de robots USARSim. L'usuari és capaç d'implementar les funcionalitats del seu projecte i així, dur a terme la validació del seu propòsit.

Tot això, combinat amb una qualitat gràfica sorprenent i un realitat física simulada avalada per institucions tan importants com NIST i IEEE, fa que IOCS sigui una eina molt potent a l'hora de validar qualsevol tipus de projecte relacionat amb la conducció de vehicles.

A.2 CONSIDERACIONS PRÈVIES

Per instal·lar i executar aquest programa és indispensable disposar d'una còpia del videojoc Unreal Tournament 2004 [22] de l'empresa Epic Games [25]. Aquest joc, a dia de 22/08/2009, es pot adquirir a qualsevol botiga de videojocs d'Espanya rondant un preu de 9,95€ [22 d'Agost de 2009].

Aquest manual està constituït per el capítol d'instal·lació: que explica detalladament quin són els passos que cal seguir per instal·lar l'aplicació amb èxit; i el manual d'ús pas a pas de l'aplicació.

Si us plau, llegeixi detingudament les instruccions d'instal·lació i d'ús d'aquest manual. Per resoldre qualsevol dubte posi's en contacte amb la persona que li ha servit aquest programa.

B INSTAL·LACIÓ

B.1 INTRODUCCIÓ

Per al correcte funcionament de l'aplicació IOCS és necessària la prèvia instal·lació del videojoc Unreal Tournament 2004 i del simulador de robots USARSim.

El procés que ha de seguir la instal·lació és estrictament important. La variació en l'ordre d'instal·lació farà inservible aquesta aplicació. Si disposa d'alguna versió d'USARSim ja instal·lada en el seu ordinador, desinstal·li-la i segueixi l'ordre establert per la figura B.1.

Figura B.1 Ordre d'instal·lació dels programes que conformen el simulador

A continuació es detallen quins són els requeriments, tant Hardware com Software, que ha de complir el seu ordinador per a garantir el funcionament correcte de l'aplicació i, tot seguit, s'indicaran els passos per instal·lar els tres programes que conformen el simulador de conducció.

B.2 REQUERIMENTS HARDWARE I SOFTWARE

IOCS és implementat en el llenguatge C++ i ha estat dissenyat per ser executat sobre el sistema operatiu Windows XP o superior de Microsoft. És necessari instal·lar l'entorn en temps d'execució de Windows Microsoft .NET Framework 3.5 [19] abans de començar la instal·lació de l'aplicació.

IOCS s'executa simultàniament amb el videojoc Unreal Tournament 2004. L'acompliment dels requeriments Hardware d'aquest joc per part de l'ordinador són

extremadament importants per tal de garantir el correcte funcionament de l'aplicació. A continuació s'especifiquen els requeriments recomanats pel fabricant del videojoc [20] pel seu funcionament:

- *Pentium III, AMD Athlon 1.2 GHz o superior.*
- *256 MB de RAM.*
- *5.5 GB de disc dur.*
- *Lector de CD o DVD.*
- *Targeta gràfica de 64 MB.*

A més cal disposar de la versió DirectX 9.0b o superior. A dia de 22/08/09 es pot descarregar la versió DirectX 10 al web:

http://www.microsoft.com/en-us/index.aspx

B.3 INSTAL·LACIÓ D'UNREAL TOURNAMENT 2004

En la instal·lació d'Unreal Tournament 2004 segueixi les instruccions referides en el manual d'instal·lació associat al videojoc.

> ***NOTA:*** Pot instal·lar l'Unreal Tournament 2004 en el directori desitjat sempre que no canviï el nom de la carpeta principal del joc: %UT2004%. Es recomana fer la instal·lació en el directori C:/UT2004 sempre que sigui possible.

Un cop instal·lat amb èxit l'Unreal Tournament 2004 és necessari instal·lar el "Official Unreal Tournament 2004 Patch V3369". A dia 22/08/09 es pot descarregar gratuïtament del web:

http://data.unrealtournament.com/UT2004-WinPatch3369.exe

B.4 INSTAL·LACIÓ D'USARSIM

Un cop instal·lat l'Unreal Tournament 2004 s'han de seguir els passos següents per a la instal·lació d'USARSim:

1 Baixar els fitxers usarsim-2004 del web del projecte USARSim. Actualment l'última versió és USARSimFull_3.37 que data de Maig de 2009. Aquest fitxers són disponibles al web: http://sourceforge.net/project/showfiles.php?group_id=145394

2.1 En el cas que es baixi l'instal·lador automàtic ".exe": executi l'instal·lador i segueixi les instruccions.

2.2 En el cas que es baixi els fitxers comprimits ".zip": descomprimeixi'ls i copiï totes les carpetes dins del directori on hi ha instal·lat l'Unreal Tournament 2004: %UT2004. A continuació, compili USARSim mitjançant el script "make.bat" que es troba al directori %UT2004/System.

B.5 INSTAL·LACIÓ DE IOCS

L'aplicació IOCS està formada per un programa escrit en C++ i tot un seguit de fitxers que han estat afegits i/o modificats a USARSim. La instal·lació de l'aplicació consisteix en copiar i/o sobreescriure els fitxers en els directoris corresponents.

Per a facilitar la feina. La carpeta general del projecte (***Projecte***) conté 7 carpetes a dins. Cada una d'aquestes, excepte la que s'anomena IOCS, té el mateix nom que la carpeta d'USARSim on han d'anar els fitxers que hi conté cadascuna. Per tal que quedi més clar, a continuació s'enuncien les accions que cal fer per a consolidar la instal·lació de IOCS. En cas que el sistema operatiu pregunti si es vol sobreescriure un fitxer, sempre dir que "Sí":

1 Copiar la carpeta **IOCS** continguda a %Install dins de la carpeta %UT2004.

2 Copiar i sobreescriure els fitxers **Cotxe.uc**, **MovimentCotxe.uc**, **GroundVehicle.uc, Hummer.uc** i **SensorPosicio.uc** continguts a %Install/USARBot a la carpeta %UT2004/USARBot/Classes.

3 Copiar i sobreescriure els fitxers **CotxeRodaDre.uc**, **CotxeRodaEsq.uc**, **HummerTireLeft.uc** i **HummerTireRight.uc** continguts a %Install/USARModels a la carpeta %UT2004/USARModels/Classes.

4 Sobreescriure els fitxers **USARBot.ini** i **usar_serv.bat** continguts a %Install/System a la carpeta %UT2004/System.

5 Copiar els fitxers **mapa_v1.ut2, mapa_v2.ut2, mapa_v3.ut2, mapa_v4.ut2 i mapa_v5.ut2** continguts a %Install/Maps a la carpeta %UT2004/Maps.

6 Copiar el fitxer **PFC_StaticMeshes.usx** contingut a %Install/StaticMeshes a la carpeta %UT2004/StaticMeshes.

7 Sobreescriure el fitxer **USARSim_Vehicles_Textures.utx** contingut a %Install/Textures a la carpeta %UT2004/Textures.

NOTA: En el procés d'instal·lació de IOCS **SEMPRE** es **sobreescriuen** els **fitxers**; **MAI** les **carpetes**.

Un cop s'han realitzat aquestes operacions d'instal·lació és necessari compilar tots els fitxers d'USARSim. Per fer-ho, executi el fitxer ***make.bat*** contingut a la carpeta %UT2004/System.

Un cop s'ha completat tot el procés d'instal·lació amb èxit, ja pot executar el simulador de conducció enfocat a la investigació IOCS.

C INSTRUCCIONS PAS A PAS

C.1 INTRODUCCIÓ

Aquest capítol mostra com funciona l'aplicació pas a pas per tal que l'usuari pugui conèixer totes les possibilitats del programa i així, esprémer el màxim rendiment.

Per iniciar l'aplicació executi el fitxer anomenat IOCS.exe contingut al directori %UT2004/IOCS

C.2 L'APLICACIÓ

L'aplicació es divideix en dues finestres principals:

- Menú d'Inici
- Interfície de Control

Els usuaris d'aquest programa especifiquen el tipus de simulació que volen dur a terme modificant els camps de la finestra *Menu d'Inici* de l'aplicació. Aquesta informació és gestionada pel programa per crear l'entorn de simulació desitjat. Durant la simulació, l'usuari és capaç de modificar el comportament del cotxe mitjançant les comandes de moviment de la *Interfície de Control*. A més, des d'aqueta mateixa finestra, l'investigador pot veure en temps real quin és l'estat dels paràmetres del cotxe dins l'escenari de simulació creat. Un cop acabat l'experiment, tota la informació queda guardada en fitxers per tal de que pugui ser estudiada.

A continuació s'expliquen en detall les funcionalitats d'aquestes finestres:

C.2.1 MENÚ D'INICI

Menú d'Inici és el nom de la finestra inicial de l'aplicació encarregada de permetre a l'usuari escollir els paràmetres específics de la simulació que es vol realitzar. A la figura C.1 podem observar que es pot dividir en quatre parts diferenciades:

- Simulació
- Connexió
- Cotxe
- Vídeo

A continuació es descriuen tots amb detall.

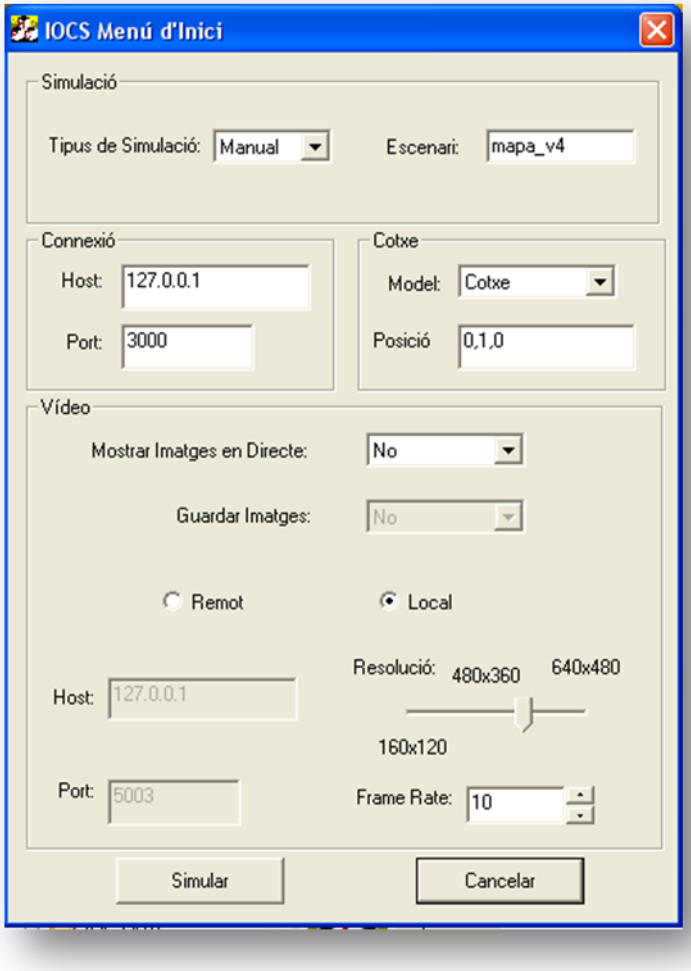


Figura C.1 Finestra Menú d'Inici de l'aplicació IOCS

C.2.1.1 SIMULACIÓ

En el camp *Simulació* del *Menú d'Inici* podem trobar un desplegable que ens permet seleccionar el tipus de simulació desitjat i un camp on introduir el mapa o nivell de simulació.

C.2.1.1.1 TIPUS DE SIMULACIÓ

El desplegable *Tipus de Simulació* permet escollir entre les opcions Manual o Automàtic (figura C.2).

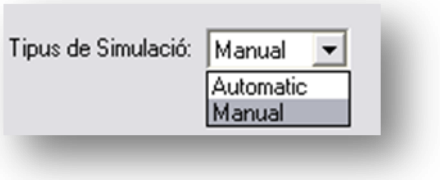


Figura C.2 Desplegable per escollir el tipus de simulació desitjat

Simulació Manual:

Una simulació Manual permetrà exclusivament a l'usuari moure el cotxe mitjançant els comandaments de control en la Interfície de Control.

Simulació Automàtica:

Una simulació Automàtica permet a l'usuari carregar un dels *experiments* duts a terme amb anterioritat i recrear els moviments del vehicle d'aquella simulació de forma automàtica. També permet a l'usuari modificar el comportament del cotxe de forma interactiva mitjançant els comandaments de control.

Si l'usuari selecciona *Automàtic*, un cop hagi pitjat el botó *Seguir* del *Menú d'Inici,* l'aplicació obre la finestra informativa de la figura C.3.

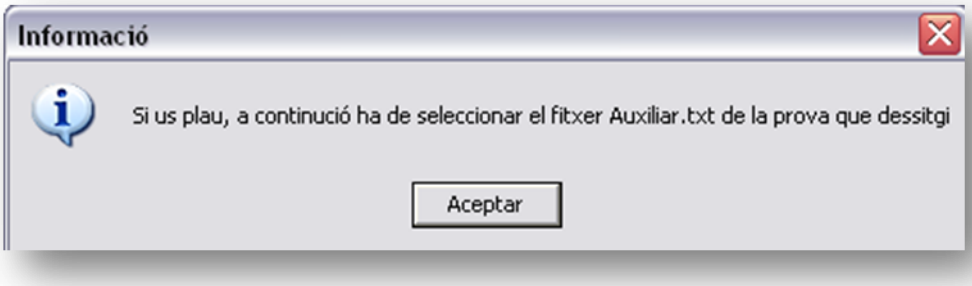


Figura C.3 Missatge Informatiu generat per l'aplicació

Tot seguit, el simulador obre una finestra de selecció de fitxer que serveix a l'usuari seleccionar la simulació que desitgi recrear (figura C.4). Un cop l'hagi escollit ha de seleccionar el fitxer Auxiliar.txt de l'experiment triat (figura C.5).

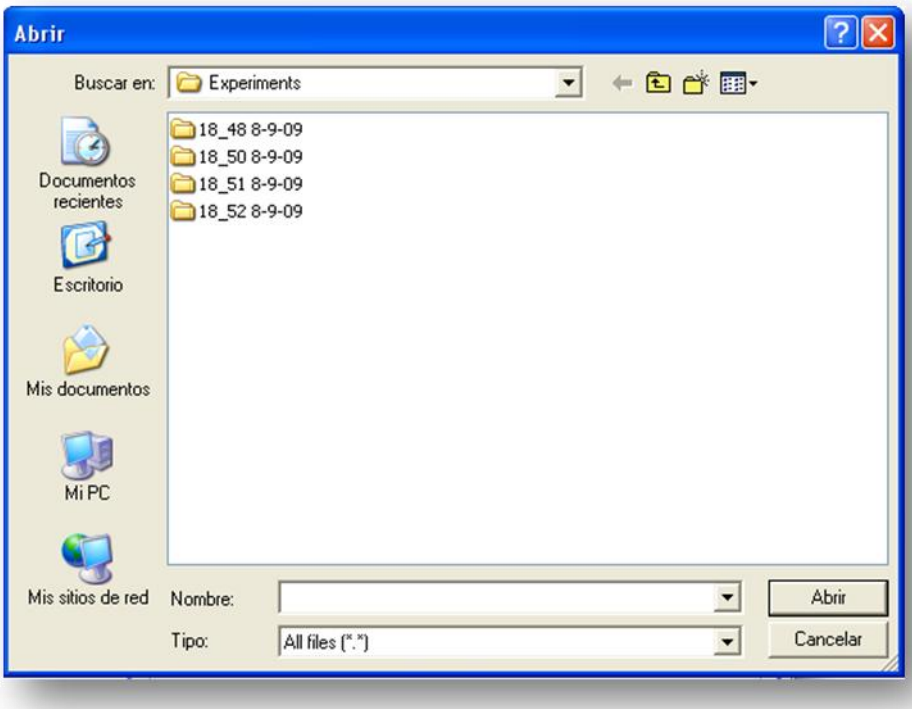


Figura C.4 Finestra de selecció simulació/experiment

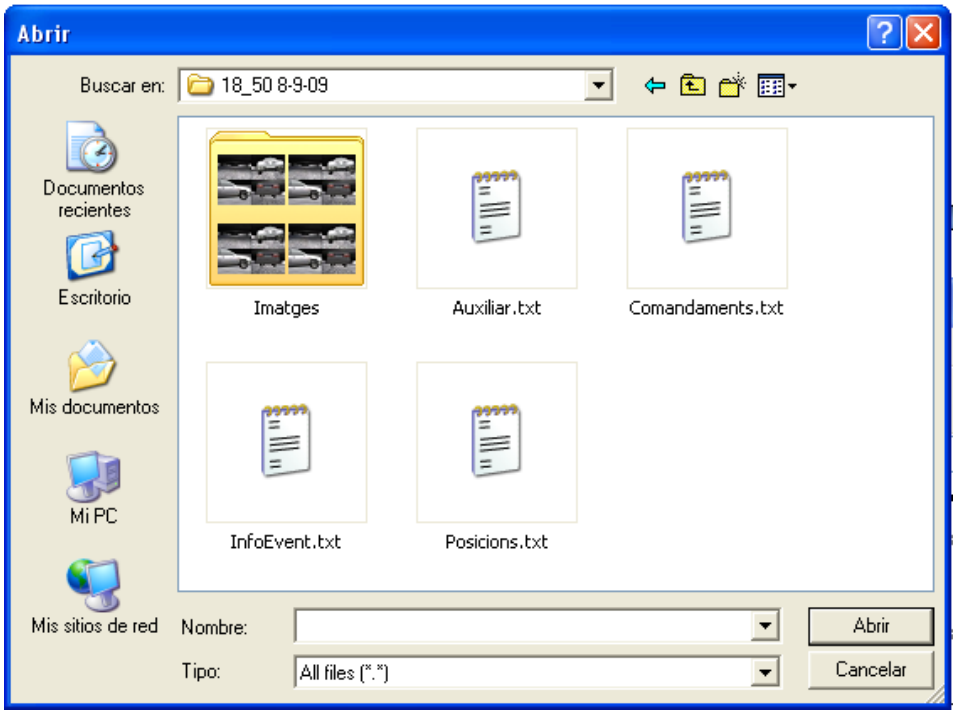


Figura C.5 Finestra de selecció del fitxer Auxiliar

En el cas que l'usuari s'equivoqui i seleccioni un fitxer que no sigui del format d'Auxiliar.txt, el programa llença un missatge d'error i permet a l'usuari tornar a escollir (figura C.6).

Figura C.6 Missatge d'error d'elecció de fitxer

C.2.1.1.2 ESCENARI

Escenari és un text editable on l'usuari escull l'escenari on tindrà lloc la simulació. Per defecte ve escollit el mapa_v3, però l'usuari pot canviar-lo només posant el nom d'aquell que vulgui ser generat. En el cas de que aquest no existeixi, quan es pitja el botó *Simular*, l'aplicació dóna un missatge d'error (figura C.7).

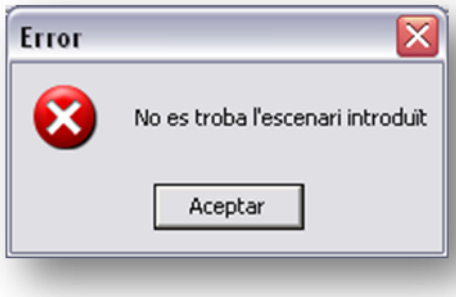


Figura C.7 Missatge d'error per la introducció d'un escenari inexistent

C.2.1.2 CONNEXIÓ

El camp *Connexió* conté dues caixes editables que són *Host* i *Port* (figura C.8).

Host serveix per indicar quina és la direcció IP de la màquina on s'executa l'*UnrealServer*. Per defecte és 127.0.0.1, la direcció de la pròpia màquina "*localhost*".

Port és el número de port pel qual es durà la connexió entre el simulador i el servidor. Per defecte és el 3000.

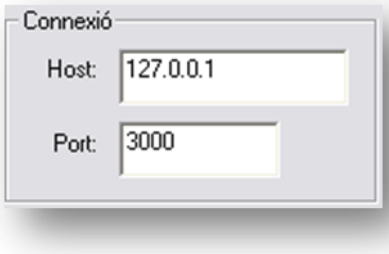


Figura C.8 Camps de Connexió

Si l'usuari s'equivoca a l'hora d'introduir la direcció IP del servidor o el port de connexió, tal i com es veu a la figura C.9, el servidor envia un missatge d'error de connexió.

Figura C.9 Missatge d'error en la connexió

C.2.1.3 COTXE

En el camp *Cotxe* el client pot seleccionar el cotxe amb el que desitgi dur a terme la simulació (figura C.10).

El botó *Model* permet seleccionar el model de cotxe a simular, actualment es disposa de dos: *Cotxe* (model pick-up) i *Hummer* (model 4x4).

La caixa editable *Posició* permet a l'usuari editar el posicionament inicial del cotxe en el mapa.

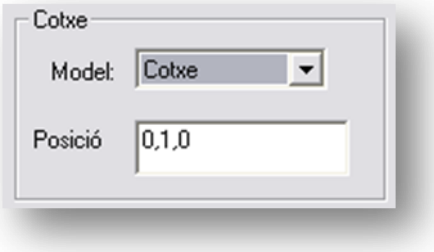


Figura C.10 Camp de selecció del cotxe i la seva posició inicial

C.2.1.4 VÍDEO

El camp vídeo dóna l'oportunitat a l'usuari de seleccionar les condicions de vídeo de les que vol disposar en la simulació (figura C.11) .

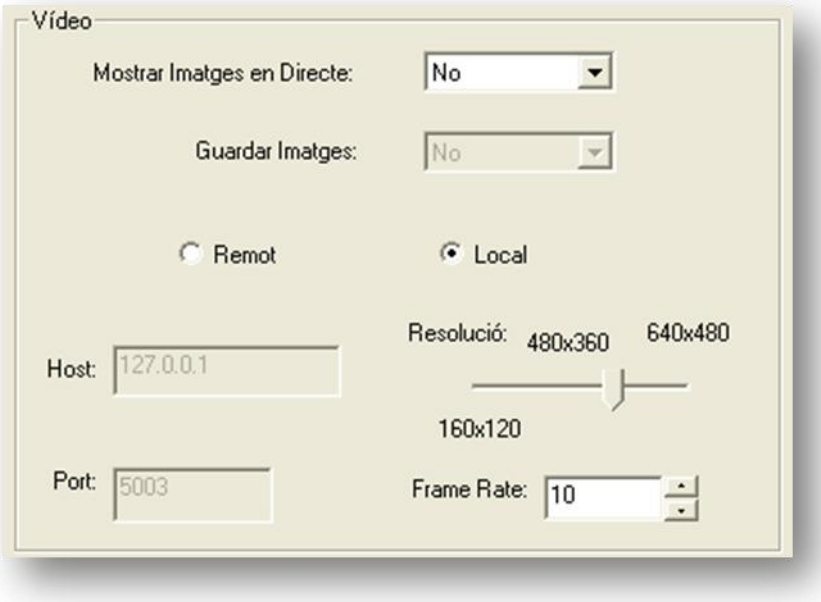


Figura C.11 Camps de selecció de modalitat de vídeo

El camp *Mostrar Imatges en Directe* concedeix la possibilitat d'elegir si es vol que el simulador mostri les imatges obtingudes de l'*UnrealClient*. Es determina la velocitat d'obtenció d'imatges modificant el *Frame Rate* i la qualitat resolutiva movent la barra *Resolució.*

En el cas que escollim "*Sí"* en el camp *Mostrar Imatges en Directe*, modificant el camp *Guardar Imatges,* permetrà guardar automàticament totes les instantànies obtingudes de la simulació*.*

El camp *Local* és seleccionat per defecte ja que el *buffer* per l'obtenció d'imatges es troba a l'ordinador on s'executa el simulador. Si el servidor és remot, l'usuari ha de prémer el botó remot i especificar la direcció IP en *Host* i el *Port* per on s'han de rebre les dades pertanyents a les imatges.

C.2.2 INTERFÍCIE DE CONTROL

Quan l'usuari de l'aplicació ha omplert correctament tots els camps del *Menú d'Inici*, el simulador llença el servidor, l'*UnrealServer*, i tot seguit el client, l'*UnrealClient*. Si la connexió no s'ha pogut dur a terme, el simulador llença un missatge d'error de connexió (figura C.9), però d'altra banda, si tot és correcte el simulador obre la pantalla informativa del funcionament de la Interfície de Control (figura C.12).

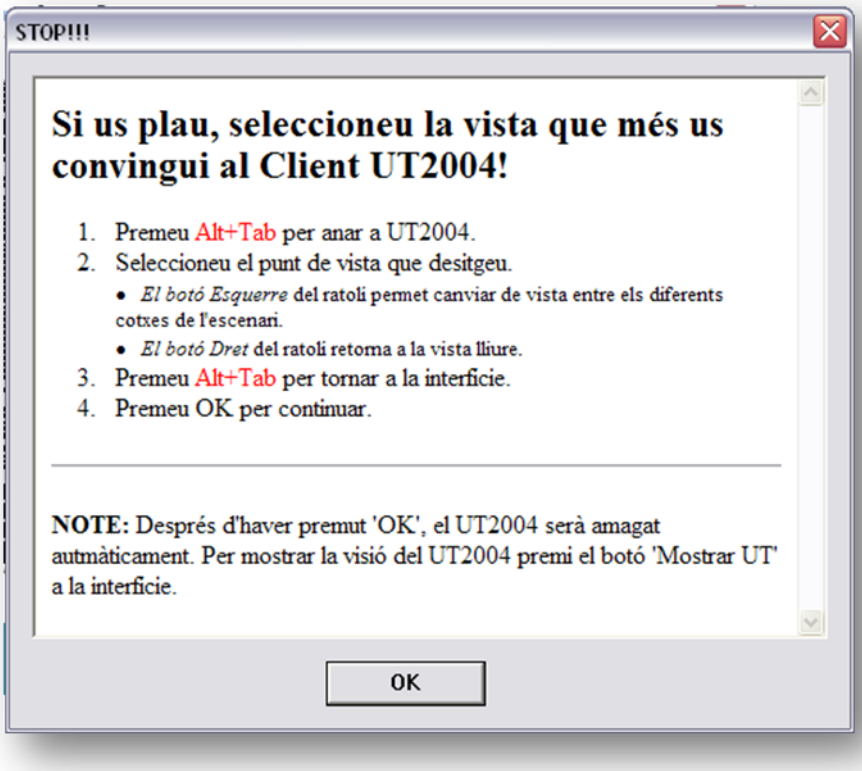


Figura C.12 Nota informativa sobre el funcionament de la Interfície de Control

Quan l'usuari prem *OK* s'obre la Interfície de Control (figura C.13). Aquesta interfície està dividida en tres parts.

- Vídeo
- Comandes de Control
- Dades Rebudes

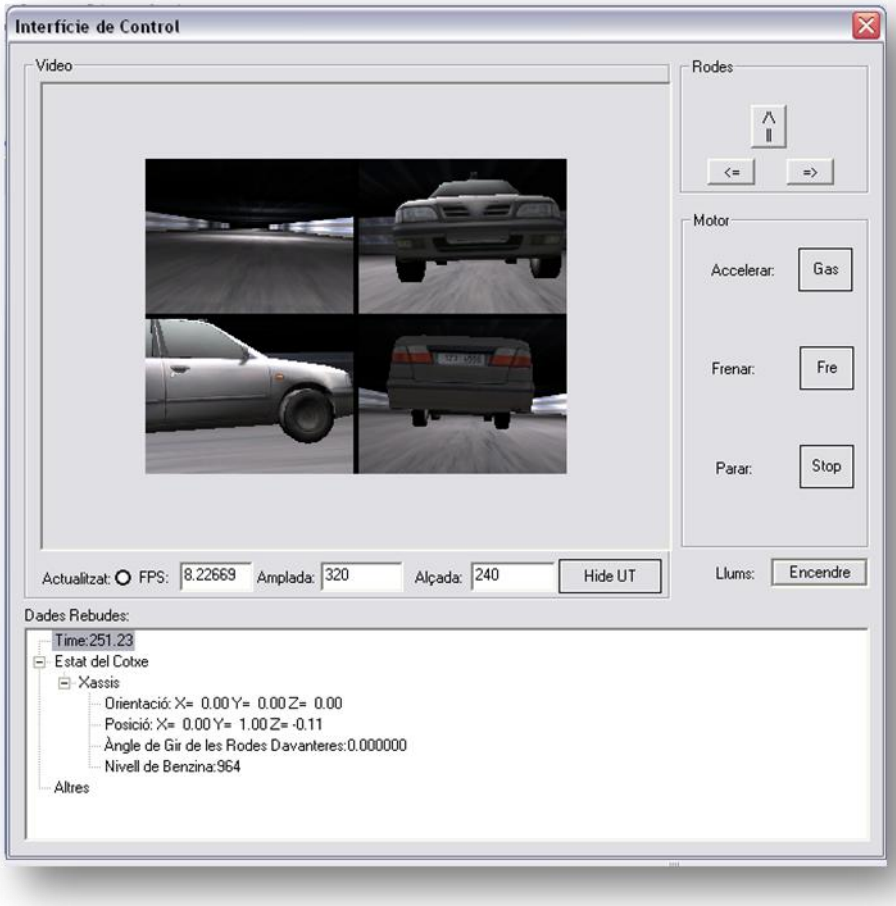


Figura C.13 Interfície de Control

C.2.2.1 VÍDEO

El camp *Vídeo* permet a l'usuari veure les imatges obtingudes del *UnrealClient* en la interfície quan s'escull "*Sí*" en el camp *Mostrar Imatges en Directe* del *Menú d'Inici.* En cas de seleccionar-se "*No",* les imatges no es mostren.

Just a sota de la pantalla de vídeo, tal i com il·lustra la figura C.14, la *Interfície de Control* dóna informació sobre quan són actualitzades les imatges, els *frames* per segon que s'estan obtenint i la seva resolució en forma d'*Amplada* i *Alçada.*

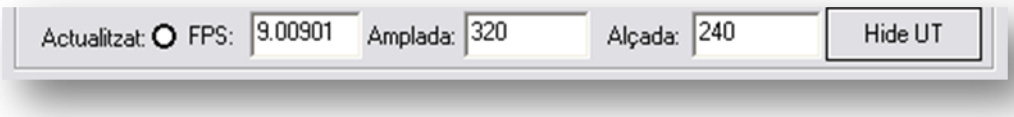


Figura C.14 Panell d'Informació del Vídeo

A més, el sector de vídeo dóna la possibilitat a l'usuari de mostrar l'*UnrealClient* quan vulgui pitjant el botó Show UT / Hide UT. L'*UnrealClient* mostra la simulació tal i com està succeint en la realitat (veure figura C.15).

Figura C.15 *UnrealClient*

C.2.2.2 COMANDES DE CONTROL

Les comandes de control es poden separar en tres grups (figura C.16).

- Control de les rodes, *Rodes.*
- Control del motor, *Motor.*
- Control de llums.

Les comades de control de les rodes permeten modificar el grau de gir de les rodes davanteres del cotxe. Els botons de les fletxes direccionals d'esquerra i dreta fan que el cotxe modifiqui en una *unitat*[12] l'angle de gir de les rodes. La fletxa cap a dalt les redreça.

Les comandes de control del motor permet modificar la tracció de les rodes davanteres del vehicle. Mitjançant el botó per accelerar *Gas* fa que el cotxe augmenti la seva velocitat en una *unitat.* A l'inrevés passa quan es pitja el botó *Fre.* Independentment de quina sigui la velocitat del cotxe, el botó *Stop*

[12] Especificada en el codi del simulador, actualment 3,5º aproximadament

neutralitza la potència del motor del vehicle. Quan la velocitat del cotxe és 0, el botó *Fre* serveix com a accelerador en marxa enrere.

En el cas de que el cotxe disposi de llums incorporades, el botó *Llums Encendre* permet encendre-les i apagar-les.

Figura C.16 Panell de control dels moviments del cotxe

C.2.2.3 DADES REBUDES

La secció de *Dades Rebudes* és on es mostra en temps real la informació relativa al cotxe en la simulació (figura C.17). S'obtenen les següents dades:

- *Time:* És el temps de simulació de l'experiment.
- *Estat del vehicle:* Es mostra quina és la posició, orientació, angle de gir de les rodes davanteres i el nivell de benzina del vehicle.
- *Altres:* Espai reservat per mostrar la informació de nous sensors acoblats als vehicles.

Figura C.17 Informació obtinguda en temps-real

C.3 INFORMACIÓ DE LA SIMULACIÓ

En el directori de Windows on es troben els fitxers pertanyents al videojoc hi ha USARSim, generalment *C:/UT2004/*, hi podem trobar una carpeta anomenada *Experiments.* Aquesta carpeta és l'encarregada de guardar totes les simulacions elaborades per l'usuari. Les simulacions són desades en carpetes amb el nom següent:

HH_MM DD-M'M'-AA on:

HH: és l'hora en que s'ha dut a terme la simulació.
MM: és el minut.
DD: és el dia.
M'M': és el més de l'any.
AA: és l'any.

Així doncs una simulació feta el 23 de Juliol de 2009 a les 10h 59' quedarà guardada en una carpeta amb el nom següent: 10_59 23-07-09.

Dins cada experiment/simulació hi podem trobar un màxim de quatre fitxers en format ".txt" encarregats de guardar la informació generada per la simulació.

- InfoEvent
- Posicions
- Comandaments
- Auxiliar

A més a més, si per la simulació seleccionada en la finestra *Menú d'Inici* vam marcar que volíem guardar les imatges, dins de la carpeta del experiment seleccionat hi trobarem una carpeta Imatges que contindrà totes les imatges extretes de la simulació.

A continuació es mostra la informació que guarda cadascun dels fitxers i de la carpeta Imatges que es poden torbar dins de les carpetes de simulacions.

C.3.1 INFOEVENT

Aquest fitxer, creat per cada simulació, és l'encarregat de guardar la informació bàsica de la simulació respectiva. Té com objectiu facilitar a l'usuari els trets més significatius de l'experiment. La informació continguda és la següent:

- ***Temps de creació**:* Desa la data exacta i en Anglès de la simulació, per exemple, *Thu Aug 13 20:31:07 2009,* simulació creada a les 20h 31' 07'' el Dijous 13 d'Agost de 2009.
- ***Model de simulador:*** Guarda el tipus de simulació que es va fer, Automàtica o Manual.
- ***Nivell:*** Escenari on succeeix la simulació. Per exemple: mapa_v3.
- ***Vehicle utilitzat:*** El cotxe que es va utilitzar en l'experiment.

C.3.2 POSICIONS

És el fitxer que permet a l'investigador veure quin ha estat el comportament del cotxe dins el mapa simulat. Aquest document mostra quina ha estat la posició i orientació en cada instant. Sempre es desa, independentment que el cotxe no s'hagi mogut en tota la simulació. La informació és presenta de la següent forma:

*A l'instant **I** la posició del cotxe és **X,Y,Z** i l'orientació **S,R,L** MR: **J** en temps: **T*** on:

I: És l'*instant*[13] de la simulació {0,1,2,3,...,n} on n és l'últim instant.
X: Posició del cotxe respecte l'eix X del mapa.
Y: Posició del cotxe respecte l'eix Y del mapa.

[13] Fa referència al número de la seqüència d'informació rebuda des del *UnrealServer*. No confondre amb el temps de simulació.

Z: Posició del cotxe respecte l'eix Z del mapa.
S: Orientació del cotxe respecte l'eix X del mapa.
L: Orientació del cotxe respecte l'eix Y del mapa.
R: Orientació del cotxe respecte l'eix Z del mapa.
J: És el número de imatge rebut per el simulador.
T: Temps de simulació en l'instant **I**.

C.3.3 COMANDAMENTS

Aquest fitxer només és creat si l'usuari utilitza algun dels comandaments de control per moure el cotxe. Permet veure quina ha estat el conjunt de d'ordres de moviment produïts l'investigador i en quin instant ha estat cadascuna. Les dades són presentades de forma següent:

*A l'instant **X** el cotxe **Y** una unitat.* On:

X: És l'*instant* de la seqüència en que es rep la comanda.
Y: És el moviment: *accelera, frena, gira a la dreta, etc.*

C.3.4 AUXILIAR

És el fitxer que conté abreviada la informació del fitxer comandaments. Està composat per el tipus de moviment que dibuixa el cotxe en un cert instant de simulació. Aquestes dades són utilitzades pel simulador per llegir de forma prèvia i ràpida quin són els moviments que haurà de realitzar el cotxe en una simulació automàtica. A continuació s'expressen les equivalències de les dades:

ac: accelerar
fr: frenar
dr: girar a la dreta
es: girar a l'esquerre
st: parar del tot
or: orientar les rodes

C.3.5 IMATGES

Aquesta carpeta és creada sempre que l'usuari hagi seleccionat a la finestra *Menú d'Inici* que vol guardar les imatges de la simulació.

Les imatges són desades en ordre correlatiu (1,2,...,243,244,...) segons el frame de la simulació al que pertanyen.

> ***NOTA:*** El procés d'emmagatzematge de les imatges en aquest directori és lent i pot afectar al rendiment de la simulació. Això és debut a la velocitat d'accés als discs durs actuals, molt inferior que la de la memòria principal.

BIBLIOGRAFIA

[1] J. Wang, S. Balakirsky i E. García, "USARSim V3.1.3 A Game-based simulation of mobile robots", NIST, CMU i Pittsburgh University .
http://sourceforge.net/projects/usarsim
[Consultat el 10 de Desembre del 2008]

[2] A. Jacoff, E. Messina i J. Evans, "Experiences in deploying test arenas for autonomous mobile robots," Servei de sistemes intel·ligents del NIST, 2001.

[3] J. Wang, M. Lewis i J. Gennari, "Usar: A game-based simulation for teleoperation," Proceedings of the IEEE International Conference on Systems, Man and Cybernatics, Denver CO, 2003, p. 493–497.

[4] S. Carpin, J. Wang, M. Lewis, A. Birk i A. Jacoff, "High fidelity tools for rescue robotics: results and perspectives," Robocup 2005:Robot Soccer World Cup IX, ser. LNCS, 2006, p. 301–311.

[5] S. Carpin, M.Lewis, J.Wang, S. Balarkirsky and C.Scrapper, "Usarsim: a robot simulator for research and education", IEEE International Conference On Robotics and Automation (ICRA), 2008.

[6] M. E. Gras, M. Planes i S. Font-Mayols, "La distracció dels conductors: un risc no percebut", Fundació RACC i Universitat de Girona, 2008.
http://imagenes.w3.racc.es/uploads/file/1376_adjuntos_cat_distraccions_web_jzq_e769cdca.pdf [Consultat el 12 d'Agost del 2009]

[7] B. Balaguer, "Where Am I? A Simulated GPS Sensor for Outdoor Robotic Applications", First International Conference on Simulation, Programming and Robots Modeling, University of California Merced, 2007.

[8] J. Craighead, "SARGE: A USER-centric Robot Simulator and Training Environment", International Conference on Human-Robot Interaction, Amsterdam 2008.

[9] J. G. Hale, B. Hohl i E. M. Moraud, "Robot simulation, collisions and contacts", 2007.
http://www.robot.uji.es/research/events/iros08/contributions/hale.pdf
[Consultat el 9 de Desembre de 2008]

[10] R. Krenn i G. Hirziner, "Contact Dynamics for Space Robotics Applications", 2008.
http://www.robot.uji.es/research/events/iros08/contributions/krenn.pdf
[Consultat el 9 de Desembre de 2008]

[11] Reial Automòbil Club de Catalunya, “Aprender a conducir con un simulador”, Revista Virtual Mundo Motor, 2001.
http://financiacion.coches.racc.es/index.racc/mod.actualidadHome/mem.FNot/relnoticia.10/relcategoria.13/chk.c2a34db2753f8b436e0718c7736c21ee.html
[Consultat 13 de Febrer de 2009]

[12] Simulador de conducció TUTOR.
http://www.landersimulation.com/index.php?id=27
[Consultat el 10 de Març de 2009]

[13] Toyota Safety, “Total Human Model for Safety, 2008”.
http://www.safetytoyota.com/en-gb/thums.html
[Consultat el 10 de Març de 2009]

[14] FIA Institute, “Human Simulator Helps Racecar Safety”, 2008 .
http://www.fia.com/oldautomotive/issue13/Institute/article3.html
[Consultat el 10 de Març de 2009]

[15] Urban Challenge by DARPA
http://www.darpa.mil/grandchallenge/index.asp
[Consultat el 15 de Febrer de 2009]

[16] J. Gonding i C. Linder “Karma User Refference”, 2003.
http://udn.epicgames.com/Two/KarmaReference.html
[Consultat el 12 de Gener de 2009]

[17] Controlador d’USARSim SimpleUI
http://sourceforge.net/projects/usarsim/files/
[Consultat l’1 de Març de 2009]

[18] Microsoft Research, “Llibreria de Windows Detours”
http://www.research.microsoft.com/sn/detours/
[Consultat l’1 de Març de 2009]

[19] Microsoft, “Modelo de programación .NET Framework”
http://msdn.microsoft.com/es-es/netframework/default.aspx
[Consultat 27 de Febrer de 2009]

[20] UltimateGamer, “Unreal Tournament 2004 System Requirements”.
http://ultimate-gamer.com/ut2004/ut2004.htm
[Consultat el 10 de Juliol de 2009]

[21] Unreal Technology : *UnrealEngine* 2 i *UnrealEngine* 3.
http://www.unrealtechnology.com
[Consultat el 3 d’Abril de 2009]

[22] Unreal Tournament 2004 Comunity.
http://www.unrelatournament2003.com
[Consultat el 3 d'Abril de 2009]

[23] Microsoft, "Explore the features: DirectX10".
http://www.micrisoft.com/en-us/index.aspx
[Consultat el 15 de Març de 2009]

[24] Static Meshes, Malles estàtiques en UnrealEd.
http://wiki.beyondunreal.com/Legacy:Static_Mesh
[Consultat el 5 de Març de 2009]

[25] Epic Games Company.
www.epicgames.com
[Consultat el 2 de Febrer de 2009]

[26] Unity3d game engine.
https://store.unity3d.com/shop/
[Consultat el 12 de Febrer de 2009]

[27] Wikipedia, "Ackerman Steering Geometry".
http://en.wikipedia.org/wiki/Ackermann_steering_geometry
[Consultat el 8 d'Abril de 2009]

[28] Unreal Editor a Unreal Wiki.
http://wiki.beyondunreal.com/Unreal_Editor_overview
[Consultat el 14 de Febrer de 2009]

[29] Unreal Script Reference.
http://unreal.epicgames.com/UnrealScript.htm
[Consultat el 14 de Febrer de 2009]

*Projecte realitzat per en **Lluís-Pere de las Heras Caballero***

Signat.

Barcelona, Setembre de 2009

RESUM

El desenvolupament de sistemes d'assistència a la conducció (ADAS) és, avui dia, una de les àrees de recerca de més interès pel Centre de Visió per Computador. A partir de la informació adquirida per sensors instal·lats en un vehicle, els ADAS assisteixen al conductor per tal d'evitar situacions de perill. La validació d'aquests sistemes però, requereix de l'obtenció "manual" de les dades que defineixen l'entorn de conducció de forma precisa: una tasca costosa i subjecta a l'error humà. Per tal de resoldre aquest problema, en aquest projecte s'ha implementat IOCS, un simulador de conducció creat a partir d'un de robots, capaç de crear entorns realistes de conducció i d'obtenir, simultàniament, les dades sobre l'entorn inferides per un ADAS i les que el descriuen objectivament. Aquesta funcionalitat facilita extremadament el procés de validació actual dels sistemes d'assistència a la conducció.

RESUMEN

El desarrollo de sistemas de asistencia a la conducción (ADAS) es, hoy en día, uno de los campos de investigación con mayor interés para el Centro de Visión por Computador. A partir de la información adquirida por sensores instalados en un vehículo, los ADAS asisten al conductor con el afán de evitar situaciones peligrosas. La validación de estos sistemas requieren de la obtención "manual" de los datos que definen el entorno de conducción de forma precisa: una tarea costosa i sujeta al error humano. Con el fin de resolver este problema, en este proyecto se ha implementado IOCS, un simulador de conducción creado a partir de uno de robots, capaz de crear entornos de conducción realistas y de obtener, simultáneamente, los valores sobre el entorno inferidos por un ADAS y los que lo describen objetivamente. Esta funcionalidad facilita extremadamente el proceso de validación actual de los sistemas de asistencia a la conducción.

ABSTRACT

The development of systems to assist a driver in their driving activity (ADAS) is an area of great interest for the Computer Vision Center. This systems use sensors to perceive and monitor the environment, and then, based on the information obtained from them, assist the driver to avoid unsafe situations. The validation process of these systems usually requires obtaining manually the "Ground truth" values: a difficult task and sometimes, imprecise due to human error. IOCS has been created in this project to avoid this problem. This program is a driving simulator built from a robots simulator that can create realistic driving environments to obtain, simultaneously, the data inferred by an ADAS and also the real values of the environment. This functionality is extremely effective to solve the problems of the current ADAS validation methodology.