Optimización de la humanidad de bots de Unreal Tournament 2004 mediante algoritmos evolutivos

Humanness optimization of Unreal Tournament 2004 bots by means of evolutionary algorithms

Álvaro Gutiérrez Rodríguez[1], Antonio M. Mora[2], Antonio J. Fernández-Leiva[1]

[1]Departamento de Lenguajes y Ciencias de la Computación,Universidad de Málaga (España).
alvarogutirodri@hotmail.com,afdez@lcc.uma.es
[2]Depto. Arquitectura y Tecnología de Computadores, Universidad de Granada, España.
amorag@ugr.es

Resumen. Este artículo presenta varias hibridaciones de dos de los mejores bots de la competición Botprize en su edición de 2014. Dicha competición busca el bot con el mejor comportamiento humano jugando al famoso First Person Shooter, Unreal Tournament 2004. Para ello se evaluaron a los participantes mediante un Test de Turing aplicado a videojuegos. El trabajo parte del código fuente de MirrorBot (el ganador) y NizorBot (el subcampeón) y los combina en dos enfoques distintos, con el objetivo de obtener un bot que pueda mostrar el mejor comportamiento humano. Además, se ha realizado una versión evolutiva de Mirrorbot, que ha sido optimizado mediante un algoritmo genético. Tanto los bot originales como las hibridaciones creadas, han sido evaluadas mediante un Test de Turing aplicado a videojuegos cuyos resultados muestran que la versión evolutiva de Mirrorbot supera a la versión original en cuanto a su comportamiento, que es más creíble. Además, una de las hibridaciones obtuvo un nivel de humanidad bastante alto de acuerdo a los resultados de los tests.

Abstract. This work describes three different approaches looking to get the best Bot (autonomous agent) for the First Person Shooter Unreal Tournament 2004. To this end, different hybridizations of the two best bots of the Botprize Competition 2014 are created. This competition decided the humanness level of the bots by means of a Turing test conducted on human judges. Thus, the proposal considers the source code of MirrorBot (winner) and NizorBot (second) and combines the best parts of their behaviour, in order to create a better human-like agent. Then, a Genetic Algorithm is applied to improve MirroBot's behavioural parameters. All the approaches have been tested by human judges, by means of a Turing test for bots, based on a third person assessment (i.e. videos showing gameplay). According to the results, the evolutionary version of MirrorBot is the best implementation. One of the hybridizations also obtained a very good humanness level.

1. Introducción

Muchos de los videojuegos modernos se diseñan para provocar sentimientos intensos en el jugador. Con este fin, muchas veces se incluyen personajes no controlables (NPCs), los cuales intentan empatizar con el humano mostrando comportamientos y sentimientos humanos. A fin de cumplir ese objetivo actualmente se empiezan a implementa inteligencias artificiales (IAs) centradas en la humanidad, que incluyen, por ejemplo, un conjunto de reglas para guiar sus acciones al igual que lo haría una persona. Este enfoque también es conocido como *credibilidad* de los NPCs (o *Bots*).

Así, el objetivo final es que ese jugador debería pasar un Test de Turing [1] dentro del juego, lo que supondría que un jugador humano pensaría que está jugando contra otros humanos. En este sentido, hace algunos años surgió la *2K Botprize Competition*. Ésta competición tiene como objetivo encontrar al Bot con el mejor comportamiento humano para el shooter en primera persona (FPS) Unreal Tournament 2004 [2], o con su acrónimo UT2K4. Este juego, en su modo *DeathMatch* (combates entre dos o más bots intentando derrotar a los oponentes y sobrevivir), se ha considerado el mejor escenario para la competición, en el que evaluar los bots mediante una versión del test de Turing adaptada al juego. En la última edición de la competición, 2014, el primer y segundo bot consiguieron unos niveles de humanidad muy interesantes. Dichos campeones son, respectivamente, Mirrorbot [3] y Nizorbot [4].

Este artículo continua un trabajo previo [5], en el cual ambos bots fueron analizados en profundidad, identificando sus puntos fuertes, así como sus puntos de mejora. En ese estudio se sugirieron diversas formas de optimizar y combinar ambos bots, de modo que, en el presente artículo, se han aplicado algunas de esas propuestas. Por tanto, aquí describimos dos posibles hibridaciones, las cuales combinan las mejores partes de cada bot con diferentes características del otro. Además, también se propone una mejora sobre el ganador de la competición, Mirrorbot, mediante la aplicación de algoritmo evolutivo para optimizar sus parámetros de comportamiento.

Los tres bots resultantes han sido evaluados mediante un test de Turing público online, en el cuál votantes anónimos han juzgado la humanidad de los bots visualizándolos en diferentes vídeos de partidas uno contra uno.

2. Estado del arte

El objetivo de crear personajes con un comportamiento humano creíble en videojuegos consiste en mostrar un comportamiento similiar al de las personas, incluyendo características como la personalidad, emociones, empatía o movimientos casi reales. Normalmente hay interacción entre los personajes en el juego, por lo que, el objetivo sería mostrar la ilusión de que esos jugadores virtuales son controlados por un humano [6]. Este es un desafío importante en los juegos actualmente, porque, de realizarse, ayudaría a mejorar sobre manera la inmersión del jugador en el juego y, en consecuencia, su satisfacción[7].

Sin embargo, evaluar el nivel de humanidad que un jugador virtual exhibe es bastante complejo, ya que, normalmente, se trata de una medida subjetiva. Así, una manera de evaluar esa 'credibilidad' podría ser mediante el test de Turing [1], o más concretamente, mediante su adaptación al ámbito de los videojuegos [6], como propone el campeonato internacional 2K Botprize Competition (ver sección 3).

Sin embargo, modelar un comportamiento humano es una tarea harto difícil, ya que no se puede limitar a seguir un conjunto de estados predefinidos o una fórmula matemática. Por lo que, las soluciones habituales [8] se enfocan en simular acciones habituales en los jugadores, tales como la efectividad media y alta en el juego, cometer en cierta medida errores inesperados de vez en cuando, tomar decisiones diferentes incluso en las mismas condiciones (con un factor estocástico), y mostrar algún tipo de .^{em}oción".

Los FPS son unos de los escenarios más utilizados para realizar el test de Turing aplicado a videojuegos, en consecuencia, han surgido un gran número de propuestas de agentes con comportamiento humano en este ámbito. Por ejemplo, SOAR Bot para Quake, presentado por Laird [9] en 2000, el cual modeló un comportamiento humano a través de una arquitectura cognitiva. Choi et al. en [10] mejoró esta arquitectura y la aplicó a un agente autónomo para el juego Urban Combat. Dicha versión mejorada era capaz de usar su conocimiento y aprender, por medio de recuerdos, cosas como determinadas habilidades o una lista de objetivos priorizados que el agente debe lograr.

Sin embargo, el entorno más extendido para crear bots con comportamiento humano en los FPS ha sido el juego Unreal Tournament 2004 (UT2K4). Algunas propuestas en este ámbito han aplicado una varicación de Algoritmos Evolutivos (EAs). Por ejemplo en [11], los autores implementaron técnicas evolutivas y co-evolutivas, o el sistema basado en reglas evolutivas presentado en [12]. Schrum et al. [13] consideraron la combinación de AEs con redes neuronales artificiales (ANNs), con el fin de aprender a jugar como un humano imitando partidas grabadas de jugadores. Por último, la propuesta de Soni y Hingston [14] intentaba imitar el comportamiento humano aplicando también redes neuronales artificiales.

Finalmente, los autores del trabajo [15] presentaron un bot que modelaba el comportamiento de un jugador español experto. Incluía dos niveles de FSM, en la que el estado primario definía el nivel

superior de comportamiento del bot (como atacar o huir), mientras que los estados secundarios (o subestados) pueden modificar este comportamiento para alcanzar objetivos necesarios o inmediatos (como conseguir items de salud o un arma poderosa que está cerca de la posición del bot). Este enfoque fue igualmente optimizado mediante un AE.

Como ya se ha mencionado, este artículo intenta ir un paso más allá en la creación del mejor bot con comportamiento humano, y lo hará mediante dos hibridaciones de los mejores bots en la última competición de Botprize (de acuerdo a la clasificación) y también mejorando al ganador de esa competición aplicando un método evolutivo.

3. Botprize competition: un test de Turing para bots

Este test es una variación del clásico test de Turing en el que un humano juzga las interacciones de los jugadores dentro de un mundo virtual (un juego), de modo que debe distinguir si esos actores son humanos o bots. Este test apunta a avanzar en los campos de la Inteligencia artificial y computacional en videojuegos, ya que un bot capaz de pasar este test podría considerarse como excelente, y podría incrementar enormemente la calidad del juego, desde el punto de vista del jugador. A su vez, el test también demuestra que el problema de la IA para videojuegos está lejos de estar resuelta.

El test de Turing para bots se han planteado considerando un videojuego multijugador en el cual el bot tiene que cooperar o luchar contra otros jugadores (humanos o bots), tomando las mismas decisiones que tomaría un humano. Esto fue transformado en una competición internacional con las siguientes características:

- Se llevan a cabo partidas en modo *Deathmatch* de diez minutos de duración.
- Idealmente habría tres participantes: un jugador humano, un bot y un juez.
- El bot debe simular ser más humano que el jugador humano y ambos reciben puntuación independiente.
- Los tres participantes no pueden distinguirse desde fuera (cada uno con un nombre y apariencias aleatorios).
- Los bots no pueden tener poderes omniscientes como en otros juegos. Sólo pueden reaccionar ante el mismo estímulo (mediante sensores) que reaccionaría un humano.

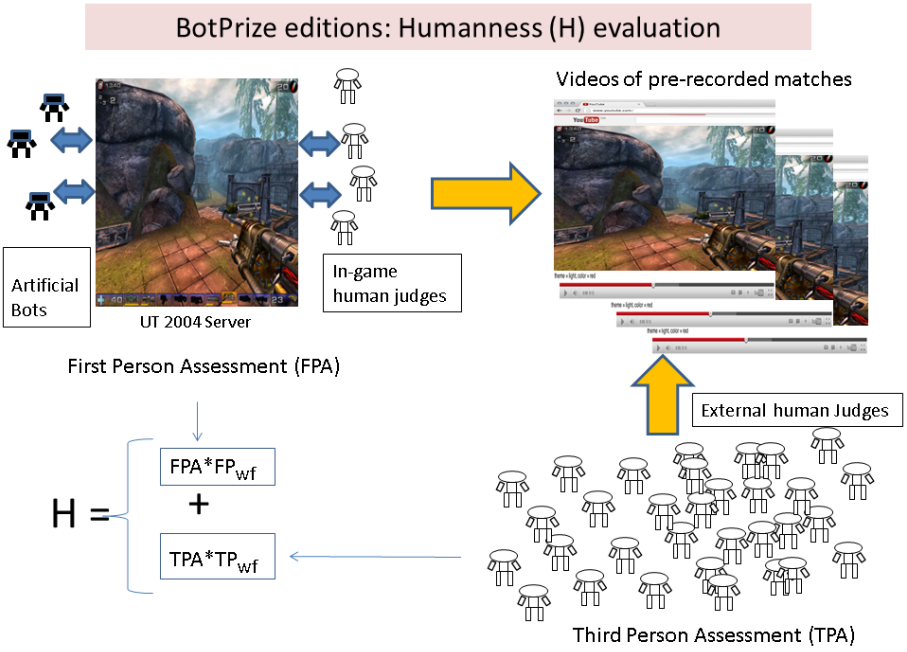


Figura 1: Esquema competición Botprize

En 2008 se celebró la primera competición de 2K Botprize (Botprize desde ahora), en el que UT2K4 fue considerado cómo el entorno ideal para este test. Los participantes debían crear bots con comportamientos humanos usando librerías externas para interactuar con el juego mediante conexiones TCP (Pogamut [16]).

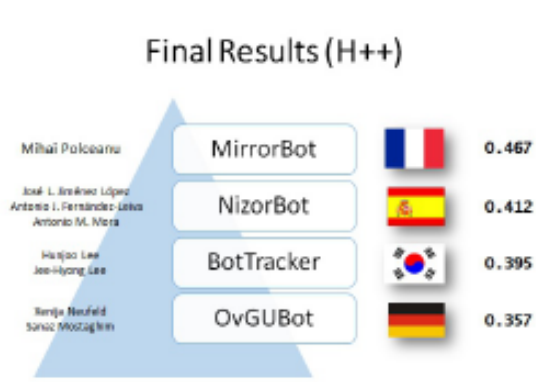
En las primeras ediciones de Botprize (de 2008 a 2011) las puntuaciones de los bots no eran capaces de superar a los jugadores humanos. En cualquier caso, la máxima puntuación de humanidad alcanzada por los jugadores humanos fue sólo 41,4 %. Esto demuestra las limitaciones del test (o de la competición), ya que evaluar incluso el comportamiento humano es una tarea complicada.

Los dos primeros bots en la edición de 2014 de Botprize fueron Mirrorbot [3] (el cual también ganó en la edición de 2012), creado por Mihai Polcenau; y una propuesta de J.L. Jiménez y dos de los autores de este estudio, Nizorbot [4].

En la competición original, un número de jueces que participaron directamente en las partidas fueron responsables de las evaluaciones de humanidad de los bots, haciendo una evaluación en primera persona (FPA). En la edición de 2014, se incluyó una evaluación en tercera persona (TPA) por medio de la participación (externa) de jueces vía una plataforma de *crowdsourcing*. La humanidad (H) fue evaluada siguiendo la siguiente fórmula:

$$H = (FPA * FP_w f) + (TPA * TP_w f) \qquad (1)$$

Dónde $FP_w f$ y $TP_w f$ son factores de ponderación (rango entre [0.0,1.0]) para FPA y TPA respectivamente. en concreto, para la edición de 2014, $FP_w f = TP_w f = 0{,}5$. Los resultados dicha edición se muestran en la Figura 2.



BotName	FPA	TPA	H++
Xenija	0.17139763	0.8235294	0.4974635
MirrorBot	0.20164771	0.7333333	0.4674905
Player	0.19328127	0.6315789	0.4124301
tmchojo	0.17757519	0.6470588	0.4123170
NizorBot	0.11821633	0.7058824	0.4120493
BotTracker	0.20070203	0.5909091	0.3958056
CCBot	0.06214746	0.7058824	0.3840149
Juan_CVC	0.12372294	0.6190476	0.3713853
OvGUBot	0.10545765	0.6086957	0.3570767
ADANN	0.08351664	0.4761905	0.2798536

Figura 2: Resultados de la botPrize 2014. Las celdas amarillas son los mejores bots mientras que las azules son los humanos.

Cómo se puede ver, Mirrorbot estuvo muy cerca de pasar el test de Turing propuesto en esa edición, la cual usó un nuevo y duro sistema de evaluación. No llegó a tener el valor necesario para poder ser considerado humano (0,5) aunque estuvo relativamente cerca de hacerlo. Nizorbot también mostró una muy buena actuación terminado en la segunda posición, obteniendo un factor de humanidad también alto.

4. MirrorBot y Nizorbot

En esta sección se presentarán los dos bots considerados como base para las hibridaciones de este estudio.

4.1. Mirrorbot

MirrorBot [3] fue desarrollado en 2012 específicamente para la competición de 2K Botprize, en la que resultó el ganador, y fue de nuevo inscrito en la edición de 2014 ganado nuevamente. Está basado en dos módulos principales de comportamiento:

- *Módulo por defecto:* usado frecuentemente para navegar a través del mapa, recogiendo ítems y armas, disparando a enemigos y esquivando sus ataques. Está compuesto por varios submódulos. Los tres principales permiten al bot apuntar automáticamente hacia los enemigos, calculando trayectorias y anticipándose a la ruta del enemigo; navegar aplicando una versión modificada de la navegación por grafo, en la cual añade algún tipo de ruido o distorsión al movimiento, con el fin de ocultar su desplazamiento; y finalmente, disparar al enemigo más apropiado teniendo en cuenta el tipo de arma, la cantidad de daño que producirá y la distancia hacia él.

- *Módulo espejo:* el cual se activa únicamente cuando un enemigo es considerado como no agresivo (todo oponente es considerado así por defecto, a no ser que dispare al bot). La razón es que probablemente ese enemigo sea un juez (humano) dentro del juego (FPA). Cuando el comportamiento espejo está activado para un objetivo, MirrorBot empezará a grabar todas las acciones a bajo nivel del oponente: apuntado, movimiento, disparo, salto, si está agachado y el arma que lleva. Todo esto se almacena en una secuencia de frames la cual reproducirá MirrorBot en el mismo orden. Se invierte la orientación y el movimiento, manteniendo siempre una distancia constante al objetivo. Adicionalmente, se añade un retardo a la secuencia con el fin de engañar al juez que esté mirando al bot.

4.2. Nizorbot

Nizorbot [4] está basado en la idea del ya citado ExpertBot [15], el cual modela el comportamiento de un jugador humano experto usando dos niveles de máquina de estados finitas (FSMs).

ExpertBot está formado por dos capas: la primera es la capa cognitiva, encargada de controlar la FSM teniendo en cuenta las estimulaciones del entorno (mediante sensores). Decide la transición entre estados y subestados usando un sistema experto y una base de datos como conocimiento adquirido. La segunda capa es la capa reactiva, la cual no realiza ningún tipo de razonamiento y sólo actúa en cada momento durante la partida.

Nizorbot es una mejora sobre ExpertBot al que se le ha aplicado un Algoritmo Evolutivo Interactivo (IEA) [17], en el cual expertos humanos guían la optimización de los parámetros del bot con el fin de obtener uno con un comportamiento humano. La idea básica es dejar que los expertos descarten las soluciones candidatas (es decir, los individuos) que actúan subjetivamente de forma menos humana que otras.

Más específicamente, cada **individuo** en el IEA era un cromosoma con 26 genes, dividio en seis bloques de información. Cada bloque representa el comportamiento de una característica específica del bot: distancia, selección de arma, prioridad de arma, perfil, riesgo y tiempo.

La **función fitness** para evaluar los individuos es una combinación de las muertes de enemigos (*frags*), el número de muertes propias y la cantidad de daño hecho y recibido por el bot. La función recompensa los individuos con un balance positivo (más frags que muertes) y un alto número de frags. Además, los individuos que causan una gran cantidad de daño a los enemigos son también recompensados, aunque tengan un peor balance.

La evaluación de un individuo consiste en asignar los valores del cromosoma a los parámetros de comportamiento de la IA de NizorBot y ejecutar una partida 1 vs 1 entre NizorBot y un bot estándar de UT2K4 en su dificultad máxima. Una vez que pasa el tiempo fijado para la partida, se recogen las estadísticas del individuo durante la misma y se evalúan aplicando la función fitness.

Respecto a los operadores genéticos, cómo mecanismo de selección se utiliza una ruleta de probabilidades, y un valor 5 de elitismo. Se utiliza un cruce uniforme, por lo que cada gen de un descendiente tiene la misma probabilidad de pertenecer a uno de sus padres.

La interacción del experto durante el juego se llevó a cabo en ciertos puntos específicos de la evolución, en los que el experto debía realizar un TPA (viendo un vídeo del bot) e identificando aquellas características específicas (como la distancia de selección, selección de arma, etc) que considerase más humanas en el bot. De modo que aquellos bloques asociados a las características seleccionadas se bloquearían y no serían alterados por los operadores genéticos durante la evolución. Esto afecta al resto de la población cuando este individuo combina y esparce su información genética. Esta interacción guía la búsqueda hacia la obtención de individuos con un comportamiento más humano.

5. Bot híbridos

En este artículo se proponen dos bots como hibridaciones de MirrorBot y NizorBot. Con respecto a este último, hemos usado los parámetros del mejor individuo obtenido en el trabajo previo [4], después de todo el proceso de evolución interactiva.

Cada bot híbrido combina las mejores partes de cada uno de los bots de referencia con una parte complementaria del otro. En concreto son:

- **MIRZorBot**: este bot considera las mejores partes de MirrorBot, es decir, el módulo de navegación (selección de objetivo, pathfinding, apuntado y movimiento) y la habilidad de imitar (módulo espejo). Ya que ha sido considerado la clave del comportamiento humano de MirrorBot después de un profundo análisis. El apuntado y movimiento parecen estar condicionados por la percepción humana durante el juego. El apuntado no es siempre perfecto. El módulo espejo añade un comportamiento no esperado (pero muy natural).
  Estos módulos y submódulos han sido incluidos dentro de la FSM de NizorBot, el cual añade también el sistema experto de selección de arma (muy eficiente, ya que fue diseñado por un humano experto que tuvo en cuenta todos los parámetros relacionados con las armas). Además, la división en dos estados primarios y secundarios es muy parecida a las prioridades que tiene un humano durante la partida.

- **NIZRorBot**: este bot usa casi toda la estructura interna de Nizorbot, pero al que se le ha incluido el módulo de navegación de MirrorBot ya que fue la parte más débil del BotExperto inicial [15]. Así, NIZRorBot hace uso del sistema de navegación con todos los trucos que se implementaron en MirrorBot, incluyendo una mejora de la malla de navegación de Pogamut (con un sistema óptimo de esquiva de obstáculos) y un sistema propio de RayCasting basado en 24 rayos: 16 horizontales (detectan obstáculos, ítems, armas, enemigos) y 8 verticales (desde el centro del bot a 45º, para detectar desniveles o grandes caídas).

6. MirrorBot Evolutivo

Además de los dos bots híbridos, en el trabajo presentamos una mejora del MirrorBot inicial, a la que llamaremos **EVOMirBot**, basada en una optimización de parámetros por medio de un Algoritmo Genético clásico (GA) [18].

El objetivo es mejorar el comportamiento general del bot original con el fin de mostrar un perfil más ofensivo (siendo más agresivo), ya que algunas veces éste se esperaba a las acciones del oponente y no reaccionaba correctamente. Es decir, no actuaba como lo haría un humano (moviéndose alrededor del enemigo y disparando, por ejemplo). Para tal fin, se han identificado y extraído los parámetros codificados a modo de constantes de los que depende el comportamiento del bot, con el fin de componer un cromosoma o individuo para el GA.

De modo que cada individuo del GA es un cromosoma formado por 12 genes, en concreto:

- *Gen 1* (valor inicial 8136): tiempo que el bot está imitando, una vez se ha elegido a un oponente para ser imitado. Se mide en milisegundos y está dentro del rango [1000, 10000].

- *Gen 2* (valor inicial 2982): tiempo que considera que tiene que transcurrir para 'olvidar' a un enemigo. Medido en milisegundos dentro del rango [1000, 9000].

- *Gen 3* (valor inicial 7): nivel de agresividad de un enemigo. Si el valor es superior a este gen, se descarta el oponente para ser imitado (será demasiado agresivo para ser un juez humano). Valor en [1, 10].

- *Gen 4* (valor inicial 115): tiempo de retardo cuando el módulo espejo está reproduciendo las acciones del enemigo. Este valor es muy importante para mejorar la impresión que da el bot al ser observado por el contrincante. Valor en milisegundos en el rango [0, 1000].

- *Gen 5* (valor inicial 842): tiempo transcurrido desde la última acción del oponente al ser imitado. Medido en milisegundos entre [0, 500].

- *Gen 6* (valor inicial 5): votos que debe recibir un oponente para ser imitado. Si recibe más votos que este valor, será imitado. Valor dentro del rango [1, 7].

- *Gen 7* (valor inicial 2000): distancia media entre el bot y el candidato rival a ser imitado. Es la distancia que considera segura para observar al contrario sin ser atacado. Valor en el rango [1200, 2000].
- *Gen 8* (valor inicial 8211): tiempo que considera a un enemigo como némesis, es decir, el bot lo atacará siempre que lo detecte. Valor en milisegundos en el rango [1000, 9000].
- *Gen 9* (valor inicial 2142): tiempo que tiene que transcurrir limpiar la lista de sus némesis. Valor en milisegundos en el rango [1000, 5000].
- *Gen 10* (valor inicial 300): tiempo de disparo descontrolado (o pseudo-aleatorio). Utilizado para mostrar un comportamiento inesperado de vez en cuando. Valor en milisegundos en [100, 500].
- *Gen 11* (valor inicial 3500): distancia considerada como lejana en relación al enemigo. Valor en [500, 5000].
- *Gen 12* (valor inicial 600): distancia considerada como corta respecto al enemigo. Valor en [100, 800].

Esta lista muestra, como referencia, los valores iniciales que tenía MirrorBot. Como se puede ver, los rangos han sido definidos bastante amplios, para que nos permita obtener una gran variación de MirrorBot al ser evolucionado.

La función fitness definida es la siguiente:

$$f(f_r, d, dmgG, dmgT) = ((f_r * 50) - (d * 5)) + (dmgG - dmgT/10) \qquad (2)$$

Dónde f_r es el número de frags que ha obtenido el bot, d es el número de muertes del bot, $dmgG$ es el daño total producido por el bot y $dmgT$ es el daño total recibido por el bot. Al igual que en NizorBot, esta función recompensa bastante a los individuos con balances positivos, es decir, más frags que muertes propias y más daño producido que recibido, con el objetivo citado anteriormente de conseguir un perfil más ofensivo.

La evaluación de un individuo se realiza dando los valores del cromosoma a la IA de MirrorBot y ejecutando una partida Deathmatch 1 vs 1 contra NizorBot en UT2K4 durante 1 minuto. Una vez finaliza la batalla, se calcula el valor para la función fitness considerando la actuación del bot.

Como mecanismo de selección se ha utilizado una ruleta de probabilidad, considerando una probabilidad proporcional al valor de fitness obtenido. Además, se ha llevado a cabo una política de reemplazo estacionaria, en la que sólo el peor individuo es sustituido en cada generación. Finalmente, se ha aplicado un cruce uniforme y una mutación que genera un valor aleatorio dentro del rango de valores posibles para el parámetro.

7. Experimentos y resultados

En esta sección se analizarán los resultados obtenidos, primero respecto al enfoque evolutivo de MirrorBot y después por medio de una evaluación en tercera persona a modo de test de Turing.

7.1. Optimización evolutiva

En este experimento el llamado EVOMirBot ha sido obtenido como una optimización de MirrorBot. La configuración de los parámetros del algoritmo evolutivo ha sido: 20 individuos, 50 generaciones, 1/12 de probabilidad de mutación. Las evaluaciones se han llevado a cabo en combates 1 vs 1 contra Nizorbot y con una duración de 1 minuto, siempre en el mapa DM-TrainingDay (frecuentemente usado en las competiciones de UT2K4). Se han ejecutado diez iteraciones del algoritmo.

La evolución del fitness de todas las iteraciones se muestra en la Figura 3.

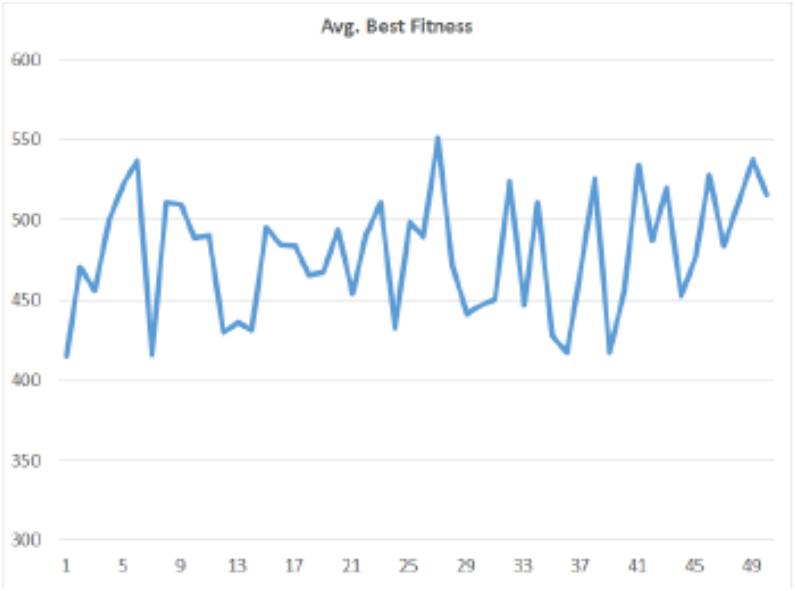


Figura 3: Evolución de los mejores fitness medios por generación considerando 10 ejecuciones.

Como se puede ver, hay una tendencia de mejora en los mejores fitness medios a lo largo de las generaciones. Sin embargo, es un poco leve, debido a al ruido natural del problema [19], esto es, un individuo puede ser evaluado como bueno en un combate, pero el mismo bot puede obtener peores resultados en otro combate. Esto sucede debido al alto componente pseudo-estocástico presente en las batallas, ya que los resultados no dependen por completo de nuestro bot, también se tienen en cuenta las acciones del enemigo, sobre las que no tenemos ningún control, puesto que no serán deterministas.

De modo que, hemos elegido como el EVOMirBot definitivo un individuo de las últimas generaciones. No fue el único con el valor de fitness más alto, pero fue el bot que, bajo nuestro punto de vista, tenía el mejor comportamiento humano (tras revisar varios combates de cada uno de los bots candidatos). Los valores optimizados obtenidos de este bot se muestran en la Tabla 1.

Gn1	Gn2	Gn3	Gn4	Gn5	Gn6	Gn7	Gn8	Gn9	Gn10	Gn11	Gn12
6231	5634	1	25	612	2	1560	3194	2880	447	3398	245

Tabla 1: Genes de EVOMirBot final.

Observando los resultados podemos remarcar que la agresividad de MirrorBot ha sido incrementada en EVOMirBot. Por ejemplo, el *Gen 2* (cambió de 2982 a 5364) hará que el bot recuerde durante más tiempo a su nemesis, el *Gen 8* (tiene como valor 3194 en lugar de 8211) reducirá el tiempo en que considerará a otro como némesis, y el *Gen 12* (cambió de 600 a 245) tiene un valor más pequeño que antes, por lo que el bot tendrá un estilo de combate más cercano que antes.

Respecto a la habilidad de imitación de MirrorBot, ésta ha sido mejorada o mejor adaptada al juego. Así, por ejemplo el valor del *Gen 1* es mucho más pequeño que antes (6231 en lugar de 8136), esto significa que estará menos tiempo imitando al oponente. El *Gen 3* (valor 1 en lugar de 7) hace que tenga un menor ratio de imitación que antes, pero los oponentes, considerados como bots serán atacados con más frecuencia, lo que es más recomendable en este modo de juego. El nuevo valor del *Gen 5* (612 en lugar de 842) significa que tendrá un retardo más pequeño a la hora de imitar. El *Gen 6* (nuevo valor 2 en lugar de 6) afecta a la probabilidad de elegir candidato a imitar. El nuevo valor del *Gen 10* (447 en lugar de 300) conducirá a EVOMirBot a tener un precisión más errática o aleatoria, lo que aumenta la probabilidad de creer que se trata de un jugador impreciso, más parecido a un humano.

7.2. Test de Turing TPA abierto

En este experimento, los tres nuevos bots (MIRZorBot, NIZRoBot y EVOMirBot) y los originales Nizorbot y MirrorBot han sido evaluados en un test de Turing abierto (al público), en una evaluación en tercera persona (TPA) a través de la página web *http://1-dot-proyecto-tfg.appspot.com*.

Para este fin se grabaron 7 vídeos de 20 segundos y se presentaron a una serie de jueces voluntarios (cualquiera podía acceder a la web). Cada vídeo mostraba parte de una escena de combate entre dos jugadores, pudiendo ser cada uno de ellos o ambos bots o humanos. Así, cada uno de los 5 bots participó en dos vídeos: uno contra otro bot y otro contra un jugador humano.

Después de ver el vídeo, el juez debía decidir sobre el nivel de humanidad mostrado por los jugadores con las opciones: *a) el jugador 1 es humano, b) el jugador 2 es humano, c) ambos son humanos, d) ninguno es humano, e) NS/NC, f) Incongruente*.

El test estuvo abierto durante tres semanas y participaron 61 jueces. Hemos considerado únicamente los votos que recibió cada bot en las opciones *a)* y *c)*. El resto de opciones se obviaron en el cálculo del nivel de humanidad de cada bot. Este valor ha sido calculado como el número de votos recibidos dividido por 122, ya que es el máximo de votos que puede recibir cada bot (61 votos * 2 vídeos en los que un bot está presente). Los resultados se muestran en la Tabla 2.

Bot	Votos como humano	Humanidad
NizorBot	57	46.72
EVOMirBot	53	43.44
MIRzorBot	51	41.80
NizorBot	47	38.52
NIZRorBot	46	37.70

Tabla 2: Resultado del test de Turing TPA.

Cómo puede verse todos los bots han tenido una puntuación bastante alta, lo que es una buena señal de su comportamiento humano.

Podemos remarcar el hecho de que EVOMirBot (aparentemente) ha mejorado con respecto a Mirrorbot. El primero ha obtenido la segunda mejor puntuación, quedando por detrás de NizorBot quién ha 'ganado'. MIRZorBot ha obtenido también buenos resultados, sin embargo, NIZRorBot ha obtenido la peor puntuación. La razón de que haya quedado en último lugar es probablemente por el sistema de raycasting de MirrorBot que entra en conflicto con la elección de objetivo original de NizorBot, lo que ha supuesto un comportamiento en cuanto al movimiento erróneo y, por tanto, los jueces lo han identificado como no humano.

Sin embargo, mirando las cifras completas en los resultados, creemos firmemente que los vídeos podrían ser mejorados. Por ejemplo, aumentando su duración (parecen ser demasiado cortos), o situando el punto de vista en cada uno de los bots, ya que probablemente no es lo mejor evaluar un oponente desde la vista de un bot que se enfrenta a él (se pierde de vista de vez en cuando, por ejemplo). La cuestión es que en este tipo de test abierto es muy importante alcanzar un número adecuado de vídeos con una duración bien ajustada, con el fin de evitar el aburrimiento en los jueces y su abandono.

8. Conclusiones y trabajo futuro

Este artículo ha presentado tres enfoques diferentes de bots con comportamientos humanos para el FPS Unreal Tournament 2004. Todo ellos se han obtenido a partir de una variación/mejora de los dos mejores bots de la competición 2K Botprize (un test de Turing para bots), en su edición 2014: MirrorBot y NizorBot.

Estos son MIRZorBot (basado en MirrorBot con componentes de NizorBot), NIZRorBot (cuya estructura es de NizorBot y el módulo de movimiento de MirrorBot) y EVOMirBot (MirrorBot optimizado evolutivamente).

En los resultados obtenidos hemos analizado la mejora obtenida de MirrorBot mediante un Algoritmo Genético, considerando los nuevos valores de los parámetros y estudiando su influencia en el comportamiento del bot. Se ha visto que se ha obtenido un comportamiento más agresivo del mismo.

Por otro lado, hemos realizado un test de Turing basado en evaluaciones en tercera persona, en las que una serie de jueces voluntarios han revisado una serie de vídeos de bots luchando en el juego y han decidido quién es el humano (si es que hay alguno). Los resultados de este test han ofrecido dos conclusiones principalmente: EVOMirBot parece ser una mejora real de MirrorBot (el hecho de ser más agresivo parece ser que ha convencido en mayor medida a los jueces en sus evaluaciones), Por su parte, MIRZorBot ha obtenido un muy buen nivel de humanidad. NIZRorBot ha tenido los peores resultados, pero la razón podría ser una incompatibilidad entre uno de los módulos con el sistema de raycasting, que ha derivado en problemas en el movimiento y, por tanto, en una evaluación negativa. Este será uno de los puntos de mejora que encararemos en un futuro próximo.

Otras líneas de trabajo futuras tratarán el ruido en el proceso evolutivo (función de evaluación), con el fin de conseguir una mejora de la tendencia en el fitness. Además, teniendo en cuenta los resultados de las votaciones en el test de Turing, ha habido algunos 'NS/NC' o 'Incongruente', lo que nos lleva a pensar que los vídeos deben ser mejorados, quizás centrándose la vista únicamente en cada bot, con una duración más larga o reduciendo el voto a un sólo bot por vídeo.

Agradecimientos

Este trabajo ha sido apoyado por el proyecto MINECO EPHEMECH (TIN2014- 56494-C4-1-P, 3-P), KNOWAVES (TEC2015-68752) (MICINN and FEDER) y la Universidad de Málaga (Campus de Excelencia Internacional Andalucía Tech). Los autores están muy agradecidos a Mihai Polcenau y José L. Jiménez, autores respectivos de MirrorBot y NizorBot, por proporcionarnos el código fuente de sus bots y su ayuda para el desarrollo de este trabajo.

Referencias

1. Turing, A.M.: Computing Machinery and Intelligence. Mind 59(236) (1950) 433–460
2. http://www.unrealtournament.com/: Unreal tournament (2014)
3. Polceanu, M.: Mirrorbot: Using human-inspired mirroring behavior to pass a turing test. In: Computational Intelligence in Games (CIG), 2013 IEEE Conference on, IEEE (2013) 1–8
4. Jiménez, J.L., Mora, A.M., Fernández-Leiva, A.J.: Evolutionary interactive bot for the FPS unreal tournament 2004. In Camacho, D., Gómez-Martín, M.A., González- Calero, P.A., eds.: Proceedings 2st Congreso de la Sociedad Española para las Ciencias del Videojuego, Barcelona, Spain, June 24, 2015. Volume 1394 of CEUR Workshop Proceedings., CEUR-WS.org (2015) 46–57
5. Polceanu, M., Mora, A.M., Jiménez, J.L., Buche, C., Leiva, A.J.F.: The believability gene in virtual bots. In: Proceedings of the Twenty-Ninth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2016, Key Largo, Florida, May 16-18, 2016., AAAI Press (2016) 346–349
6. Livingstone, D.: Turing's test and believable AI in games. Computers in Entertainment 4(1) (2006) 6
7. Soni, B., Hingston, P.: Bots trained to play like a human are more fun. In: Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on. (2008) 363–369
8. Yannakakis, G., Togelius, J.: A panorama of artificial and computational intelligence in games. Computational Intelligence and AI in Games, IEEE Transactions on (2014) Accepted for publication.
9. Laird, J.E.: It knows what you're going to do: Adding anticipation to a quakebot. AAAI 2000 Spring Symposium Series: Artificial Intelligence and Interactive Entertainment SS-00-02 (2000)

10. Choi, D., Könik, T., Nejati, N., Park, C., Langley, P.: A believable agent for firstperson shooter games. In Schaeffer, J., Mateas, M., eds.: Proceedings of the Third Artificial Intelligence and Interactive Digital Entertainment Conference, June 6-8, 2007, Stanford, California, USA., The AAAI Press (2007) 71–73

11. Priesterjahn, S., Kramer, O., Weimer, A., Goebels, A.: Evolution of humancompetitive agents in modern computer games. In: IEEE World Congress on Computational Intelligence 2006 (WCCI'06). (2006) 777–784

12. Small, R., Bates-Congdon, C.: Agent Smith: Towards an evolutionary rule-based agent for interactive dynamic games. In: IEEE Congress on Evolutionary Computation 2009 (CEC'09). (2009) 660–666

13. Schrum, J., Karpov, I., Miikkulainen, R.: Ut2: Human-like behavior via neuroevolution of combat behavior and replay of human traces. In: Computational Intelligence and Games (CIG), 2011 IEEE Conference on. (2011) 329–336

14. Soni, B., Hingston, P.: Bots trained to play like a human are more fun. In: IEEE International Joint Conference on Neural Networks, IJCNN'08. (2008) 363–369

15. Mora, A.M., Aisa, F., García-Sánchez, P., Castillo, P.A., Guervós, J.J.M.: Modelling a human-like bot in a first person shooter game. IJCICG 6(1) (2015) 21–37

16. http://pogamut.cuni.cz/main/: Pogamut - virtual characters made easy — about (2014)

17. Takagi, H.: Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation. Proceedings of the IEEE (9) (2001) 1275–1296

18. Goldberg, D.E.: Genetic Algorithms in search, optimization and machine learning. Addison Wesley (1989)

19. Mora, A.M., Fernández-Ares, A., Merelo, J.J., García-Sánchez, P., Fernandes, C.M.: Effect of noisy fitness in real-time strategy games player behaviour optimisation using evolutionary algorithms. J. Comput. Sci. Technol. 27(5) (2012) 1007–1023