Boti ve First Person Shooter hrách

Vysvětlení pojmů z nadpisu:

First Person Shooter

Střílečka z pohledu první osoby. Hráč ovládá postavu, u které vidí jakoby jejíma očima a zároveň jediné co z ní vidí, jsou většinou ruce se zbraní. Příkladem takovéto hry je např. hra Counter Strike, postavená jako rozšiřující mód na enginu hry Half life. Na této hře bude demonstrována problematika botů v tomto referátu.

Mód

Hra, která používá většinu prvků z enginu, ale v některých prvcích se liší. Např Counter Strike je modem enginu Half life, na kterém je postavena i stejnojmenná hra. Rozdíl je v tom, že Half life je vesmírná sci-fi střílečka, odehrávající se ve smyšleném prostředí s nereálnými zbraněmi a upravenou fyzikou, ale Counter Strike je do jisté míry reálná hra, kde jsou hráči rozděleni na dva týmy. Na teroristy a policisty. Tyto dva týmy proti sobě bojují na dvou různých typech map – na mapách s rukojmími, kde je úkolem policistů rukojmí osvobodit a úkolem teroristů jim v tom zabránit a na mapách s bombami, kde teroristé musí bombu položit, spustit a nechat vybuchnout, v čemž jim mají naopak policisté zabránit. Hra končí buďto splněním úkolu daného týmu nebo smrtí všech členů jednoho z týmů.

Bot

Počítačem simulovaný hráč užívaný k tréninku, nebo když není momentálně možnost živých spoluhráčů. Nahrazuje protivníky i spoluhráče. Z této definice vyplývá, že boti jsou používáni především ve hrách pro více hráčů – tzv. on-line multiplayer hrách, jako jsou Counter Strike, Quake, Unreal Tournament, atd.

Zdroje:

http://developer.valvesoftware.com/

http://www.kiv.zcu.cz/ - semestrální práce na téma Multibot

http://scienceworld.cz – článek Umělá inteligence v PC hrách

+ části zdrojových kódů hry Couter Strike a enginu Half life

Co musí bot umět:

Navigace

Základním problémem, který musí každý bot řešit, je navigace a pohyb v herním prostředí. Bot si za tímto účelem musí nejen pamatovat pozici významných prvků v herní mapě (protivník, umístění rukojmí, místo na položení bomby apod.), ale musí být i kdykoliv schopen najít do vybraného místa pokud možno nejkratší cestu. O to se stará tzv. pathfinding, dnes v naprosté většině her založený na algoritmu heuristického prohledávání grafu. Pro jeho efektivní nasazení je potřeba k dané úrovni nejdříve zkonstruovat tzv. navigační síť (angl. navigation mesh), tedy graf, který popisuje odkud kam je možno se v dané úrovní dostat.

Kromě informací týkající se pohybu může navigační síť obsahovat i speciální značky (waypointy) označující místa výhodná pro obranu či naopak pro číhanou na protivníka. V tomto pojetí se z navigační sítě stává komplexní „mentální" reprezentace úrovně, kterou bot využívá nejen k navigaci, ale i k taktickým bojovým rozhodnutím. Aby se dosáhlo její maximální optimalizace, je navigační síť často konstruována přímo designéry úrovní. Manuální konstrukce navigační sítě má ale některé významné nevýhody. Kromě zřejmé pracnosti je to především neschopnost reagovat na dynamické změny prostředí v průběhu hry (prorážení zdí, zasypání chodeb, zhroucení mostů…). V poslední době se proto objevuje snaha tvořit navigační síť automaticky pomocí nástrojů umělé inteligence, což však představuje nelehký úkol.

Řídicí systém

Zatímco navigaci lze považovat za podpůrný systém, skutečným mozkem bota je jeho řídicí systém. Na základě aktuálního stavu bota, jeho záměrů a situace v nejbližším okolí volí akce, které bot v příštích okamžicích provede. Tyto akce lze většinou rozdělit do několika úrovní – hovoříme proto o hierarchickém řídicím systému. Na nejvyšší úrovni jsou akce, které se váží k dlouhodobějším a strategičtějším rozhodnutím – bot si (ve hře určitého typu) může vybrat, zda bude prozkoumávat aktuální úroveň, doplňovat střelivo, útočit či naopak ustupovat z boje. O úroveň níž bot rozhoduje např. o tom, kterým směrem se v mapě při svém průzkumu vydá, na jaký konkrétní cíl zaútočí a jakou přitom použije zbraň. Nejnižší úroveň se pak týká konkrétních atomických úkonů, tj. např. kdy vystřelit, kam zamířit, kudy uhnout střele protivníka apod. Zdaleka nejčastějším způsobem implementace řídicího systému je určitá obdoba tzv. hierarchického stavového automatu. Základní myšlenka konečných stavových automatů je velmi jednoduchá, navíc jsou výpočetně velmi efektivní. Problémy ovšem nastávají při jejich realizaci, která má obvykle podobu kódu v běžném procedurálním jazyce, nejčastěji přímo v C++. Takto implementovaný stavový automat se totiž při větším počtu stavů a stavových přechodů rychle stává nepřehledným, obtížně rozšířitelným a testovatelným. Další nevýhodou jsou jeho omezené schopnosti adaptace a

plánování. Sílí proto poptávka jednak po specializovaných nástrojích pro implementaci stavových automatů, jednak po jiných způsobech realizace řídicího systému. Slibnou alternativou, rozvíjenou především na akademické půdě, jsou v současné době tzv. plavidlové systémy, v delším horizontu pak i systémy pro cílově orientované dynamické plánování.

Jak bot pro Half life funguje?

Obyčejně mody pro Half life nemohou používat boty, pokud je tvůrci nezapracují přímo do modu (např. Counter Strike). Nedá se tedy použít jeden bot pro více modů. Bot může být přidán do modu, bez toho aniž byste vlastnili zdrojový kód modu. Většina vývojářů modů nezveřejňuje své zdrojové soubory. Přesto lze vlastního bota do některého z modů přidat. Je to díky tomu, že engine Half lifu používá pro herní entity interní strukturu. Ta samá struktura je pak použita i pro entity v modech. Když se hráč připojí k serveru, engine Half lifu vytvoří entitu pro toho hráče a v modu pak volá funkce, které představují akce prováděné s entitou. Boti fungují na stejném principu jen s tím rozdílem, že u nich neexistuje síťové spojení, protože existují pouze na serveru. Je důležité pochopit, jak funguje interface mezi enginem a modem. V enginu Half lifu existuje seznam funkcí, které může mod volat a naopak v modu existuje seznam funkcí, které muže volat engin. Ostatní kód v modu je "samostatný" a nepotřebuje (kromě zmíněných funkcí) žádné spojení s enginem Half lifu. Je možné vytvořit DLL knihovnu, která "sedí" mezi enginem Halflifu a modem, a která přenáší volání funkcí z jedné strany na druhou. Když chce engin Half lifu zavolat nějakou z funkcí modu, zavolá nejprve DLL a to předá volání dále modu a naopak mod volá prostřednictvím DLL knihovny funkce enginu. Pro vlastní naprogramování bota je nutné použít právě tuto knihovnu a to protože v Counter Striku jsou boti jakoby předchystáni, ale samotné nastavování paramaterů probíhá pouze v této knihovně, do původního kódu hry se nezasahuje (nebo jen minimálně).

<u>Konkrétní ukázky některých funkcí</u>

Na začátku hry (každého kola) je provedena funkce BotSpawnInit(), která zinicializuje všechny potřebné struktury, popřípadě bota teprve vytvoří (pokud začíná úplně první kolo). Bot má na svoje akce každou vteřinu k dispozici 30 framů (ve smyslu časového úseku). Na začátku každého framu je spuštěna funkce StartFrame(), která volá pro každého bota funkci BotThink(), což je hlavní funkce inteligence bota. Právě funkce BotThink() se stará o zjišťování stavu bota (živý/mrtvý/zraněný) a provádění na něm závislých reakcí. V případě, že má bot zadány nějaké úkoly (tasky), stará se o jejich provedení atd. S trochou nadsázky se dá říci, že všechny ostatní klíčové funkce se volají v těle této rutiny nebo v tělech jí volaných rutin. Na konci kola je provedena funkce UpdateGlobalExperienceData(), která vyhodnotí úspěšnost bota a nastaví podle toho příslušné dovednosti, např. zaznamená nebezpečná místa, kde byl bot zabit.

Úkoly

Boti mohou mít zadány úkoly, které se snaží vyplnit. Například zadání (ve hře) požadavku "Cover me" použitím vysílačky vyústí v uložení zprávy do fronty pro všechny boty v "doslechu". Pokud některý z kolemstojících botů tuto zprávu "pozitivně" vyhodnotí, přidělí se mu úkol TASK_COVERME. Ten pak plní, dokud není splněn nebo pokud mu není zadán další úkol. Všechny zadané úkoly se ukládají do zásobníku, což znamená, že bot vždy plní poslední zadaný úkol. Po jeho dokončení začne plnit předposlední úkol atd. Splnění úkolu může vypadat různě. Například u úkolu TASK_FOLLOWUSER (provádí radio-příkaz "Follow me") je vyhodnocováno (ve funkci BotFollowUser()), zda bot může svého kolegu stále následovat, tzn., jestli je naživu, jestli je v dohledu apod.

Pohyb bota

Bot má několik základních parametrů, které určují jakým směrem a jak rychle se pohybuje. Tyto parametry se v DLL vlastně jen nastavují jejich zpracování a "provedení" si pak zajišťuje přímo engine hry.

Příklad
float f_max_speed
- maximální rychlost bota se zbraní, kterou drží

float f_move_speed
- aktuální rychlost pohybu vpřed (vzad - záporná hodnota)

float f_sidemove_speed
- aktuální rychlost do strany (úkroky)

Obecně se boti pohybují po předem známých waypointech. BOT se vlastně stále pohybuje od waypoint k waypointu. Pouze setkání s nepřítelem nebo následování kolegy může vyústit v "neowaypointovaný" pohyb. BOTi mají předdefinované útočné waypointy, obrané waypointy atd. pro každou mapu. Pokud tyto waypointy pro danou mapu neexistují, není ani možné boty do takové mapy přidat.

Vlastnosti a dovednosti bota

Bot má veliké množství vlastností a dovednosti. Ty jsou sdruženy ve struktuře botabot_t.

Příklady:

float fAgressionLevel
- aktuální úroveň agresivity bota

float fFearLevel
- aktuální úroveň strachu bota

boolean bDead
- určuje, zda je bot ještě na živu

int iTeam
- index týmu

Podle hodnot těchto (a mnoha dalších) atributů bot reaguje na většinu podnětů ze svého okolí, provádí rozhodnutí o svém pohybu, střelbě nebo vyhledává nejbezpečnější úkryt. Většina hodnot se mění skoro neustále s ohledem na mnoho podnětů, i herní engin je nelineární a proměnliví během hry, proto, když vedle sebe postavíme boty dva, bude se každý z nich chovat odlišně.

Shrnutí

Pro přibližnou ukázku toho, jak bot přibližně funguje, máme informací už dost, takže bych údaje z referátu ještě na závěr krátce shrnul. Před samotným programováním bota je potřeba nachystat si navigační mapu, připravit si hru s módem a předpřipravit si dll knihovnu. U bota je třeba naprogramovat jeho existenci, pohyb, vlastnosti a akce, které provádí a to mnoha příkazy, z kterých jsme si ty důležité ukázaly. Pokud by někoho tvorba botů zajímala více a podrobněji, doporučuji stránky pro tvůrce her od firmy Valve http://developer.valvesoftware.com/

Co bude dál

Jaká tedy bude herní AI budoucnosti? Především bude schopna učit se, ať už ze svých vlastních „zkušeností“ nebo z hráčových postupů, a výrazně tak omezí možnost opakovaného využívaní stejné strategie ze strany hráče. Hra si bude vytvářet model zachycující herní styl a preference hráče a tomuto modelu pak přizpůsobí svůj průběh tak, aby hráče ještě více bavila. Použití technik cílově orientovaného plánování umožní herní AI realizovat komplexní strategie šité na míru aktuální herní situaci. Umělá inteligence začne hrát významnou roli i jako hráčův asistent. Dokáže hráči poradit, nebude-li vědět jak dál nebo bude-li se chtít ve hře dále zdokonalit. Adaptivní uživatelské rozhraní bude schopno předvídat úmysly hráče a ulehčí mu od nutných rutinních činností, které se i dnes ve hrách vyskytují. Ve větší míře bude využívána komunikace hlasem.

Herní postavy budou vybaveny moduly pro komunikaci v přirozeném jazyce, což nahradí monotónní předskriptované dialogy volnou komunikací. Bouřlivý rozvoj zřejmě čeká metody reprezentace nelineárních příběhů a tzv. adaptivního vyprávění.

Ač by se mohlo zdát, že se jedná o hudbou vzdálené budoucnosti, velká většina těchto technologií už byla v určité míře použita i v existujících hrách. V nejbližších letech se proto nejspíš máme na co těšit.