9. Der KI auf die Sprünge helfen

KI-Verhalten erzwingen

Insbesondere die Nutzung von Fahrzeugen durch KI macht oft Probleme - sowohl im Editor als auch mit Zeus gespawnt.

3Den Enhanced bietet eine große Zahl an Einstellungen dafür, dass sich die KI nicht verselbständigt und dann unberechenbar agiert.

Man kann:

- Fahrer von Fahrzeugen daran hindern, in den "Aware-" oder "Kampfmodus" umzuschalten, wenn Feinde in der Nähe sind: "autocombat" , "autotarget" und "target" deaktivieren!

- den Flucht-Koeffizienten ganz runter setzen - und den Mut/Courage auf Maximum;

- Einheiten auf "achtlos" (Gruppenverhalten) schalten und erst ab einem bestimmten Wegpunkt (Wegpunkt-Attribute) auf "wachsam" schalten;

- Einheiten, die als Fahrer fungieren, auf "gefangennehmen"/ "set captive" schalten (macht sie zivil);

Keine Lust auf 3DEN Enhanced? Selber schuld. Dann müssen die entsprechenden Kommandos aus dem Biki gesucht und in die Init-Boxen der Einheiten und Wegpunkte geschrieben werden.

Fraktionszugehörigkeit erzwingen

1. OPFOR-Gruppe erstellen;

2. Anführer austauschen (von feindlicher Fraktion) und neu gruppieren.

Schon sind z.B. CSAT-Truppen BLUFOR - oder umgekehrt.

Seitenverhältnisse erzwingen

Der globale Befehl

setFriend

kann alle Seitenverhältnisse definieren, auch die zwischen OPFOR und BLUFOR. Er muss serverseitig mit *if (isServer) then {}* in einer init.sqf oder einer initServer.sqf ausgeführt werden. Die Syntax lautet

seite1 ***setFriend*** *[seite2 , Wert];*

die Seitenbezeichnungen lauten *east, west, resistance* und *civilian.* Ist der Wert 1, sind die Parteien global befreundet, ist er null (bzw. <0.6), sind die Parteien verfeindet.

Beispiel:

```
if (isServer) then {
west setFriend [east, 1];
};
```

Aber Achtung: Will man beide Seiten wechselseitig zueinander befreunden, so muss der Befehl wechselseitig ausgeführt werden:

```
if (isServer) then {
west setFriend [east, 1];
east setFriend [west, 1];
};
```

Hier der Link zum Biki:
https://community.bistudio.com/wiki/setFriend

Kapitulation bei bestimmter Gruppengröße

1. Trigger mit Gruppe verknüpfen, Gruppe einen Var.Name geben!

2. Trigger Condition ("unitname" durch Var. Name ersetzen):

({alive _x} count units unitname) < 2

Erläuterung: Der Trigger fragt nun permanent die Gruppengröße ab und reagiert bei < 2. Die Zahl lässt sich natürlich ändern. Kleiner als 2 ist 1, also wird bei nur einem verbleibenden Gruppenmitglied dieses kapitulieren.

3. On Activation ("unitname" durch Var. name ersetzen):

```
{
_x setCaptive true;
_x action ["Surrender", _x]
} forEach units unitname;
```

KI wirft Waffe weg bei Kapitulation

Folgenden Befehl im Editor oder in Zeus in der Init-Box einer bewaffneten Infantrieeinheit ausführen. Gibt sie auf, wirft sie vorher ihre Waffen weg:

```
["ace_captiveStatusChanged", {
params ["_unit", "_state", "_type"];

if (local _unit && {_state && {_type == "SetSurrendered"}}) then {
_unit call ace_hitreactions_fnc_throwWeapon;
};
}] call CBA_fnc_addEventHandler;
```

Unendlich Sprit, Mut, Gesundheit

1. Unendlich Treibstoff.

In die Fahrzeug-Init:

```
_vehicle = (_this select 0);

_h = [_vehicle]spawn
{
while {true} do
{
if ((fuel (_this select 0)) < 0.8) then
{
(_this select 0) setFuel 1;
};
sleep 120;
};
};
```

2. No Surrender:

```
this allowFleeing 0;
```

3. Disable/Enable Damage:

```
this allowdammage True/False;
```

Einen (stabilen) KI-Konvoi machen

Geht mit 3DEN Enhanced ganz gut, aber es gibt Regeln:

1. Konvoi mit Besatzung erstellen. Die Sache simpel halten: nicht zu viele verschiedene Fahrzeugtypen durcheinander, keine unnötigen Kommandos (get out, tut dies, tut das), denn Arma spinnt bei Konvois ziemlich rum. Und nicht zuviele Fahrzeuge (max. 10) dranhängen, geht unter Umständen aufs Netzwerk.

2. In 3DEN Enhanced bei allen FAHRERN "Target" und "Autocombat" deaktivieren, "Fleeing Coeff." auf 0, full "Courage" und "Commanding". FAHRER-Verhalten auf "Safe" und "Forced Hold Fire" (Schützen können schießen);

3. Alle Fahrzeuggruppen nacheinander mit vorderstem Fahrzeug in richtiger Reihenfolge gruppieren.

4. Konvoigruppe auf Formation "Linie". Mit der Reihenfolge muss man ein bisschen experimentieren. Schnelle Fahrzeuge am Konvoi-Ende sorgen eher für Probleme.

5. Danach normale Move-WP setzen und Route sinnvoll planen! Jeder Haltepunkt, jede Kurve KANN Probleme machen, vor allem auf Maps mit schweren Steigungen und verbuggten Straßen.

6. Geschwindigkeit des ERSTEN Fahrzeugs auf "Slow" oder per Editor definieren.

7. Dichte Abstände können dazu führen, dass einzelne Fahrzeuge nach einem abrupten Stopp nicht mehr weiterfahren und von der Straße abkommen. Abstände können beliebig definiert werden, sollten aber einheitlich sein:

    if (isServer) then {secondVehicle setConvoySeparation 20};

8. Die Straße muss komplett frei sein! die KI verweigert sonst die Weiterfahrt oder weicht aus.

Link:
https://www.youtube.com/watch?v=3os0EF-xOGk

Diskussion:
https://forums.bohemia.net/forums/topic/216226-ai-convoy-best-practices/?page=1

Getriggerte KI-Wegpunkte (auch mit show/hide-Modul)

Funktioniert super und eignet sich dafür, sowieso eingeplante Einheiten mit dem Editor ins Geschehen zu bringen, ohne Zeus nutzen zu müssen. Schont die Performance der Mission und vor allem den Traffic, weil Zeus-Einheiten dem Zeus-Client "gehören" und dort berechnet werden. Der Datenaustausch sorgt schnell für De-Syncs, wenn es zu viele werden!

1. Einheit setzen, Verhalten definieren;
2. Trigger setzen und einen Var. Name (hier: trg1) geben;
3. Wegpunkt "SCRIPTED" direkt vor der Einheit setzen, danach "MOVE", "SEEK AND DESTROY" oder was auch immer man will;
4. Trigger Activation definieren;
5. Condition des Triggers:

    true && triggerActivated trg1;

Einheit setzt sich bei Triggerauslösung in Bewegung. Eignet sich gut für Einheiten, die vorher nicht ins Spielgeschehen eingreifen sollen.

Sollen die Einheiten vorher **versteckt** sein, bieten sich Show/Hide-Module an. Sieht entsprechender Eintrag hier im Buch.

Einheiten bei Missionsstart verstecken (show/hide-Modul)

Simple Sache:

ZWEI Module der Sorte "Zeigen/Verstecken" (hide/show) platzieren, **BEIDE** mit der Einheit/dem Gruppenführer verbinden (synchronisieren). **EINES** der beiden Module auf "zeigen" schalten, das andere auf "verstecken". Es lassen sich beliebig viele Einheiten mit einem Modulpaar verknüpfen, aber nur ein Trigger pro Modulpaar!

Trigger setzen, gewünschte Bedingungen einstellen und Trigger **nur** mit dem Modul "zeigen" verknüpfen (synchronisieren). Einheit ist versteckt, bis der Trigger aktiviert wurde. Umgekehrt geht natürlich auch.

Man kann den einheiten auch Wegpunkte setzen, die die Einheiten sofort ablaufen, sobald der Trigger ausgelöst wurde.

Siehe auch "Getriggerte Wegpunkte" hier im Buch!

Eine Einheit verschwinden lassen

Wenn man Editor-Objekte nicht mehr auf die Karte herumfahren/-fliegen lassen will, weil sie ihren Zweck erfüllt haben, kann man sie automatisch per Script löschen, ohne Zeus starten zu müssen:

1. Fahrzeug setzen, einen Var.Name geben;

2. Folgendes z.B. in die OnAct. eines Fahrzeug-Wegpunkts oder eines Triggers:

```
{
deleteVehicle _x;
} forEach crew fahrzeugname;
deleteVehicle fahrzeugname;
```

3. Man kann auch nur die Crew verschwinden lassen, wenn man die letzte Zeile weglässt. Für das Kommando ist nur wichtig, dass erst die Crew, dann das Fahrzeug gelöscht wird.

4. Der Despawn dauert ein paar Sekunden.

Insertion via Helikopter

Das Script erlaubt es dem Spieler, im Helikopter den Startbefehl zu geben für eine Insertion. Der Helikopter setzt den/die Spieler am gewünschten Punkt ab. Danach fliegt er zum Ausgangspunkt zurück, landet dort und schaltet die Motoren aus. Die addAction kann auch an ein anderes Objekt geheftet werden. Die Aktion ist beliebig wiederholbar.

1. Helicopter Init:

this AddAction ["Titel", {meineVariable = true; publicVariable "meineVariable";}, [], 0, false, true, "", "", 3];

ERLÄUTERUNG: Die Variable ("meineVariable") kann auch anders heißen, denn sie definiert nur die Bedingung der Trigger-Auslösung. Keine Dopplungen mit anderen Scripts, sonst gibt es eine Race Condition! Es handelt sich um die Variable, die per *addAction* auf "true" geschaltet werden kann. Die Variable "public" zu machen, macht sie global. Default-Radius für "AddAction" ist 50 Meter, oben ist er auf 3m reduziert.

2. Erster Wegpunkt, wichtig: DIREKT vorm Helikopter, wird mit Trigger ("Waypoint Activation") verknüpft. Trigger auf REPEATABLE schalten!

Trigger Condition:

meineVariable;

3. Abladeplatz mit Helipad markieren, Wegpunkte draufsetzen (WP und HP überlagern sich, "rasten" ein), Art: "Transport Unload";

WP-Aktivierungsfeld:

this land "get out";

ERLÄUTERUNG: Wegpunkte nicht zu eng setzen; Geschwindigkeit wird vom angesteuerten Wegpunkt definiert, nicht vom überflogenen. Abruptes Abbremsen führt dazu, dass der Helikopter aufsteigt! Mit 3den Enhanced lässt sich die Maximalgeschwindigkeit in den Heli-Attributen genau festlegen. Soll eine Hot LZ angesteuert werden, sollte man KI-Verhalten auf "Careless" stellen und zusätzlich in die Helikopter-Init einfügen: this allowfleeing 0 oder entsprechende Einstellungen in 3den Enhanced vornehmen. "Safe" mit deaktiviertem Autocombat geht aber auch.

4. Wegpunkte zum Ausgangspunkt zurückführen, letzter WP "CYCLE", auf Heim-Helipad setzen. WP vor Cycle hat im Aktivierungsfeld:

vehicle this land "land"; if (isServer) then {meineVariable = false; publicVariable 'meineVariable'; };

Erläuterung: Die Variable wird hiermit zurückgesetzt auf "false", damit trotz cycle-Befehl erst wieder der Trigger ausgelöst werden muss. Achtung: Der Cycle-Wegpunkt muss wieder mit dem ersten Wegpunkt verknüpft sein. Ist er zu nah am Helikopter, verknüpft er sich mit dem Fahrzeug.

Das Ganze lässt sich natürlich auch als Extraction aufbauen. Dafür muss aber die addAction (für die WP-Aktivierung) dem zu extrahierenden Spieler/Team direkt zur Verfügung stehen. Der Extract.-WP muss vom Typ "GET IN" sein. Am besten macht man hierfür zwei addAction-Einträge: Einen zum Anflug und einen zum Abflug, wenn alle drin sind.