

Rückkonvertierung TPF2 -> TPF1 I : Einleitung, abwärtskompatible Dateien, Meshes, Animationen, Gruppen

Um Modelle von TPF1 zu TPF2 zu konvertieren, gibt es mehrere automatisierte Methoden, die inzwischen hinreichend bekannt sein dürften.

Wenn man nun aber ein Modell von TPF2 gerne in TPF1 hätte, dann muß man es manuell umbauen. Das ist bei einfacheren Modellen mit nur wenigen Animationen nicht über die Maßen kompliziert. Alle Sonderfälle und Möglichkeiten kann ich hier nicht beleuchten, aber einen groben Überblick werde ich zu geben versuchen. Dieser Überblick richtet sich an Leute, die grundsätzlich mit der Struktur eines Mods vertraut und für die Keyframes, Transformationsmatrizen, Meshes und Ähnliches nichts völlig Neues sind; wenn ich hier die Grundlagen auch noch erklären würde, würde die Länge des Texts explodieren. Des weiteren rate ich dringend dazu, für die ersten manuellen Konvertierungen kein komplexes Modell zu wählen: bei Flugzeugen zum Beispiel ist die Animation des Einziehfahrwerks oft äußerst verschachtelt und die Konvertierung erfordert ein gewisses Verständnis für die Materie.

Ich werde es hier einmal am Beispiel des MAN-Haubendiesels beschreiben, der sowohl in 2 als auch in 1 vorhanden ist.

1. Abwärtskompatible Dateien

Nicht alle Dateien benötigen eine Nachbearbeitung und man kann sie sowohl in TPF1 als auch in TPF2 verwenden.

Die Verzeichnisstruktur ist bis auf wenige Details identisch. Es ist also möglich, das Modverzeichnis aus TPF2 en bloc an den entsprechenden Ort in TPF1 zu kopieren, dann ist die grundlegende Struktur schon mal vorhanden.

Die mod.lua, die Materialdateien (.mtl) und alle .msh.blob-Dateien brauchen keine Änderungen. Die Texturdateien, ob .tga oder .dds, sind ebenfalls für TPF1 und 2 gleich. Wenn das Modell als Asset hergerichtet ist, ist die .con ebenfalls grundsätzlich abwärtskompatibel. Hier gibt es eine Einschränkung: in TPF2 ist bisweilen eine eigene Assetversion ohne Lichter oder drehende Teile integriert. Dies ist in TPF1 nicht nötig, da es hier weder verwischte Propeller noch animierte Lichter gibt. Deswegen ist es möglich, daß man die .con hier minimal anpassen muß, um nicht die Asset-, sondern die normale Version des Modells anfordern zu lassen.

2. Die Meshdateien (.msh)

Hier sitzt der erste große Unterschied. In TPF1 werden Details eines Meshs in dieser Datei definiert, die in TPF2 in der großen .mdl zusammengefaßt sind. Relevant sind hier die Animationen und vor allem die Anforderung des Materials. Vergleichen wir einmal eine .msh aus 1 mit einer aus 2.

Code

```
function
return
```

```
data()  
{
```

[illegible]

...und hier dieselbe Datei aus 1:

Code

```
data()  
{
```

<https://www.transportfever.net/lexicon/entry/341-r%C3%BCckkonvertierung-tpf2-tpf1-i-einleitung-abw%C3%A4rtskompatible-dateien-meshes-animat/>

Alles anzeigen

Das Material wird in 1 mit "materials = {"vehicle/truck/MAN415.mtl"}," angefordert. In TPF2 ist eine exakt gleich lautende Zeile in der .mdl enthalten:

Code

```
{  
  
    },  
    },  
    },  
    AN415-Betonmischer.mtl"},  
  
    transf={1, 0, 0, 0, 0, 0, 1, 0, 0, -0, -0, 1, 0, 0, 0, 0, 1, },  
    }
```

Alles anzeigen

Die muß also einfach aus der TPF2- .mdl herauskopiert und an den passenden Ort in der 1-.msh hineinkopiert werden. Den kleinen Block oben "matConfigs = {{0, }}, " dürfen wir auch nicht vergessen - der wird vor "subMeshes=..." eingefügt.

Nicht animierte Meshes sind damit schon fertig angepaßt und können in 1 verwendet werden.

Manche Teile sind jedoch animiert. In der oben gezeigten 1-.msh seht ihr einen Block "animations = {...}", der prinzipiell auch im gezeigten Teil der TPF2-.mdl vorkommt. Wenn die 2-.mdl Keyframes benutzt (die sehen so aus: "{time = 0,rot = { 0,0,0},transl = {0,0,0}}"), dann kann der Block einfach 1:1 genommen und aus der 2-.mdl in die 1-.msh hineinkopiert werden (an den Ort, an dem er in der oben gezeigten 1-.msh sitzt!); dann ist nur noch der Verweis aus der 1-.mdl wichtig. Zu dem kommen wir später.

Wenn die 2-.mdl aber wie hier auf eine .ani-Datei verweist, dann ist es etwas komplexer. In diesem Fall müssen wir die 2-.ani öffnen, und die sieht so aus:

Code

```
data()  
{
```

<https://www.transport11veve.net/lexicon/entry/3410643BCckonv766rune-tpf20pf1-0e1766ing-0w643A0rsknopathle-0e11-meshes-5nimat/>

Alles anzeigen

Das scheint zunächst etwas chaotisch, es ist aber nichts anderes als die in TPF1 und 2 bekannten Transformationsmatrizen in einer etwas anderen Sortierung. Hier geht jetzt ein wenig Tipparbeit los. Wir legen uns im zu animierenden 1-.msh diesen Block hier an:

Code

```
    },  
  },  
}
```

Alles anzeigen

...und den füllen wir dann mit den Matrizen aus der .ani. In den Block "keyframes = {...}" kommt dann für jede Transformationsmatrix aus der .ani folgender Block:

Code

```
{  
  transf = { 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, },  
},
```

Hinter "time" kommt der Reihe nach der Wert aus dem "times"- Block der .ani und hinter "transf = " ebenfalls der Reihe nach jede Transformationsmatrix. Die ersten drei Blöcke würden daher so aussehen:

Code

```
{  
  transf = { 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, },  
},  
{  
transf={1,0,0,0,0,0.996,0.087,0,0,-0.087,0.996,0,0,0,0,1},  
},  
{  
transf={1,0,0,0,0,0.985,0.174,0,0,-0.174,0.985,0,0,0,0,1},  
},
```

Alles anzeigen

...und so weiter. Das ist natürlich langwierig, da in 2 die .anis oft sehr lang sind - es ist aber in 1 auch möglich, bei kontinuierlichen Animationen wie rotierenden Wellen nur beispielsweise jeden vierten Block anzugeben. Möglich ist natürlich ebenfalls, wenn man die Animation genau kennt, sie in Keyframes schnell neu selbst zu schreiben, das kann bisweilen etwas schneller gehen.

Wichtig ist, daß wir der Animation einen in diesem Modell eindeutigen Namen geben - unter diesem werden wir sie dann in der 1-.mdl aufrufen.

Damit ist dann das Mesh fertig, in TPF1 verwendbar und für die Animation vorbereitet.

Diese Konvertierung ist natürlich für alle zum Modell gehörenden Meshdateien einzeln durchzuführen.

3. Die Gruppen

Die sind in TPF1 völlig anders organisiert als in TPF2. Sie sitzen im Verzeichnis res\models\group\vehicle\truck (je nach Art des Modells angepaßt!) und enthalten Dinge, die in 2 ebenfalls in der .mdl sitzen.

Vergleichen wir einmal die Gruppe, in der die Trommel des Betonmischers bei TPF2 sitzt, mit der Datei in 1.

In 2 erkennt man die Struktur, die in 1 eine .grp-Datei benötigt, am Stichwort "children=". Das steht gleich hinter den Animationen und sieht in der .mdl so aus:

Code

```

{
    {time = 0, rot = { 0,0,0}, transl = {0,0,0}, },
    {time=500, rot={ 0,0,15}, transl={0,0,0}, },
    {time=1000, rot={ 0,0,30}, transl={0,0,0}, },
    {time=1500, rot={ 0,0,45}, transl={0,0,0}, },
    {time=2000, rot={ 0,0,60}, transl={0,0,0}, },
    {time=2500, rot={ 0,0,75}, transl={0,0,0}, },
    {time=3000, rot={ 0,0,90}, transl={0,0,0}, },
    {time=3500, rot={0,0,105}, transl={0,0,0}, },
    {time=4000, rot={0,0,120}, transl={0,0,0}, },
    {time=4500, rot={0,0,135}, transl={0,0,0}, },
    {time=5000, rot={0,0,150}, transl={0,0,0}, },
    {time=5500, rot={0,0,165}, transl={0,0,0}, },
    {time=6000, rot={0,0,180}, transl={0,0,0}, },
    {time=6500, rot={0,0,195}, transl={0,0,0}, },
    {time=7000, rot={0,0,210}, transl={0,0,0}, },
    {time=7500, rot={0,0,225}, transl={0,0,0}, },
    {time=8000, rot={0,0,240}, transl={0,0,0}, },
    {time=8500, rot={0,0,255}, transl={0,0,0}, },
    {time=9000, rot={0,0,270}, transl={0,0,0}, },
    {time=9500, rot={0,0,285}, transl={0,0,0}, },
    {time=10000, rot={0,0,300}, transl={0,0,0}, },
    {time=10500, rot={0,0,315}, transl={0,0,0}, },
    {time=11000, rot={0,0,330}, transl={0,0,0}, },
    {time=11500, rot={0,0,345}, transl={0,0,0}, },
    {time=12000, rot={0,0,360}, transl={0,0,0}, },

```

```

},

```

```

},

```

```

},
{

```

```

materials={"vehicle/truck/MAN415-Betonmischer.mtl",},

```

```

mesh = "vehicle/truck/MAN415B_Lagerring.msh",

```

```

transf = { 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, },
},
{

```

```

materials={"vehicle/truck/MAN415-Betonmischer.mtl",},

```

```

mesh = "vehicle/truck/MAN415/MAN415B_Lagerring.msh",

```

```

transf = { 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, },
},
},

```

```

.9850,-0.174001000.17400.9850,.414302.147281},
},

```

Alles anzeigen

Das kommt in TPF1 dann in eine Gruppendatei mit der Endung .grp. Der Name kann relativ frei gewählt werden, sollte aber im Spiel einmalig sein. Hier lautet er MAN415B_Trommel.grp.

Code

$\{$

<https://www.transportfever.net/lexicon/entry/341-r%C3%BCckkonvertierung-tpf2-tpf1-i-einleitung-abw%C3%A4rtskompatible-dateien-meshes-a0imat/>

Alles anzeigen

Den großen Block der Keyframeanimationen können wir auch hier wieder 1:1 aus der TPF2-.mdl in die .grp übernehmen; wäre ein Verweis auf eine .ani da, würde es wieder so wie in der .msh gezeigt funktionieren.

Und jeder Block, der in der 2-.mdl ein Mesh oder eine weitere Gruppe verlangt, wird auch in der TPF1-Gruppe benötigt. Die Anforderung eines Meshes heißt in 2 "mesh=...", in 1 jedoch "id = ...". Die Transformationsmatrizen dahinter können 1:1 kopiert werden.

Aus dem Block in der 2-.mdl

Code

```
{
materials={"vehicle/truck/MAN415-Betonmischer.mtl",},
          mesh = "vehicle/truck/MAN415/MAN415B_Lagerring.msh",
transf = { 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, },
},
```

wird also in der 1-.grp

Code

```
{
                                id = "vehicle/truck/MAN415/MAN415B_La
transf = { 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, },
},
```

Und selbiges gilt für jedes andere Mesh, das in der .grp angefordert wird.

Jetzt müssen wir uns noch um die Animation kümmern. In der 1-.grp steht ein Block:

Code

```
    },
},
```

Das bedeutet folgendes: Aus der .mdl wird später eine Animation namens "forever" angefordert werden, die oben definierte Keyframeanimation, die auch oben den Namen "MAN415_TrommelAction" zugewiesen bekommen hat, auslösen soll. In diesem Fall dreht sich einfach nur die ganze Trommel um ihre Achse. Wenn

es Untergruppen oder andere separat animierte Teile in der Gruppe gibt, kann die Animation wesentlich komplexer aussehen, das Prinzip ist aber dasselbe.

Die Transformationsmatrix ganz unten in der 2-.mdl werden wir später auch brauchen, die wird in der 1-.mdl stehen und dort die Gruppe anfordern.

Am einfachsten ist hier, wenn man sich eine einfache .grp wie die hier gezeigte hernimmt und die Elemente aus der 2-.mdl hineinkopiert.

Die .mdl werde ich in einem separaten Artikel behandeln.