



# EDM Blender Plugin

Official manual

Blender plugin for exporting 3D models into the EDM format used in DCS World. Enables artists and modders to prepare assets for the simulator with streamlined export workflows and integration support.



**Contents**

Plugin Installation .....2

Core Functionality of the Blender EDM Exporter .....3

General Scene Requirements .....4

Layers.....5

Auxiliary Objects: Bounding and User Box .....7

Collisions.....8

Texture Format for Export ..... 10

Material Setup..... 10

Listing Input Values in the Material ..... 12

Glass Material ..... 13

Animation ..... 14

Animation of objects/empties ..... 17

Bone Animation ..... 18

Two ways to export bone animations .....20

Applying Transforms .....22

UV animation.....26

Lighting .....28

Lamps.....28

Fake Lights (BANO-light points) .....30

Rabbits Lights.....35

Emission Materials (glow) .....37

Damage Material (Texture damage).....44

Deck Material.....48

LightMap (Unic AO) .....50

LightMap Animation (Changing LightMap) .....51

Mirror in cockpit .....53

Numbers.....54

Opasity Value Argument .....55

Water Mask .....56



## Plugin Installation

Recommended Blender versions (LTS versions): **3.6, 4.2, 4.5!**

Download from: [https://files.eagle.ru/mods/edm\\_blender\\_plugins/](https://files.eagle.ru/mods/edm_blender_plugins/)

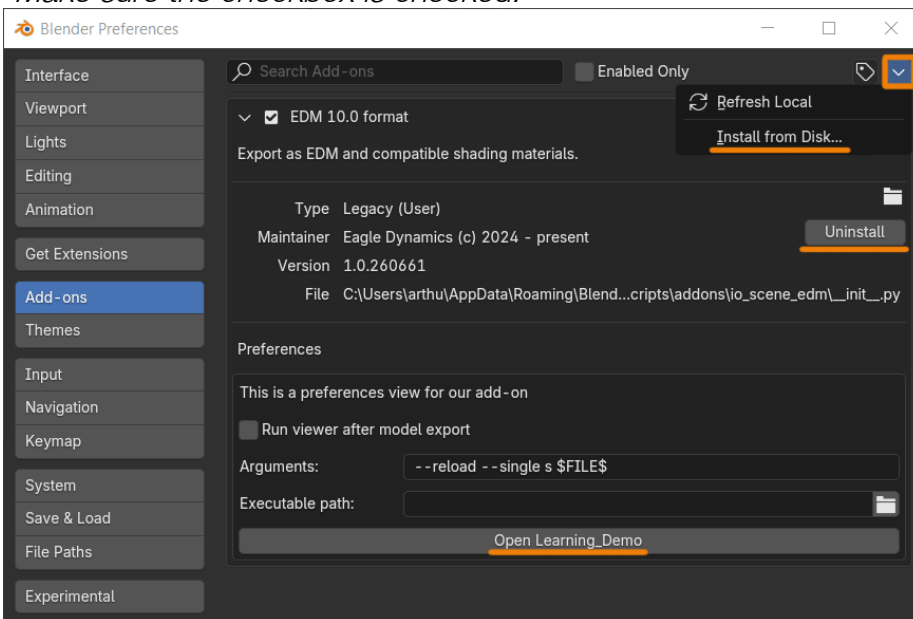
Or go to the Releases section of the <https://github.com/EagleDynamics>.

Plugin installation in Blender is performed in the standard way.

Through the menu Preferences → Add-ons → Install From Disk.

Removal is done in the same menu: in the add-on's own tab, press the Uninstall button.

*\*Make sure the checkbox is checked.*



*\*\*Learning Demo scenes are located in the add-on folder - examples for trainings.*



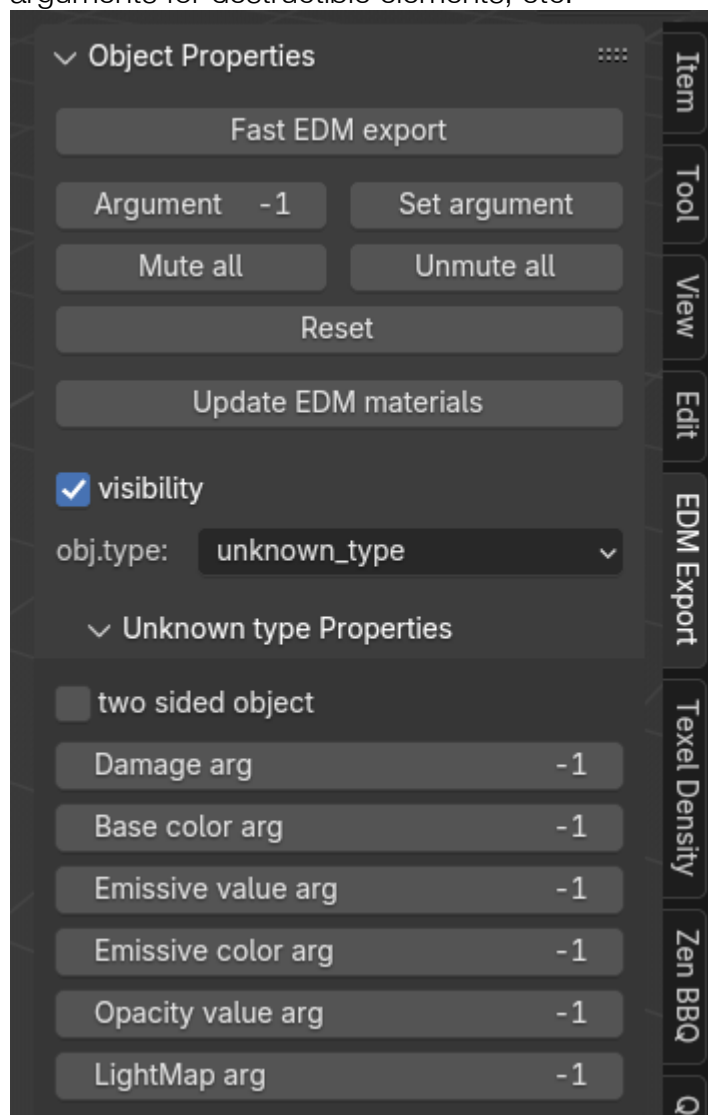
C:\Users\\*\*\*User\AppData\Roaming\Blender Foundation\Blender\3.6\scripts  
 \addons\io\_scene\_edm\Learning\_Demo



## Core Functionality of the Blender EDM Exporter

The EDM Export panel, located in the N-panel, provides essential tools for preparing models and animations for export.

It allows you to: assign argument numbers to animations, specify object types (e.g. connectors, collisions, etc.), configure visibility animations, enable two-sided rendering, set damage arguments for destructible elements, etc.



**Argument** — Selects the active argument for animation playback

**Set Argument** — Disables all animations except the one matching the selected argument

**Mute All** — Turns off all animations

**Unmute All** — Enables all animations

**Reset** — Clears all animations and sets the timeline range from frame 0 to 200

**Update EDM Materials** — Refreshes materials in the scene (useful when new versions are available)

**Visibility** — toggle object visibility on/off

**Obj.Type** — select object type (BoundingBox / Collision / Connector, etc.)

**two sided objects** — enable double-sided geometry rendering



**Damage arg** — damage animation argument **Base colour arg** — argument for animating the Base Colour in the material.

**Emissive value arg** — argument for animating emission strength

**Emissive colour arg** — argument for animating emission colour

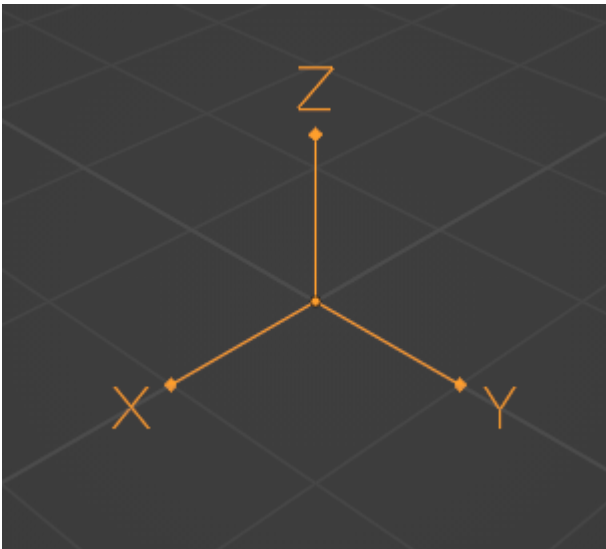
**Opacity value arg** — argument for animating material transparency multiplier

**LightMap arg** — argument for animating LightMap texture switching

## General Scene Requirements

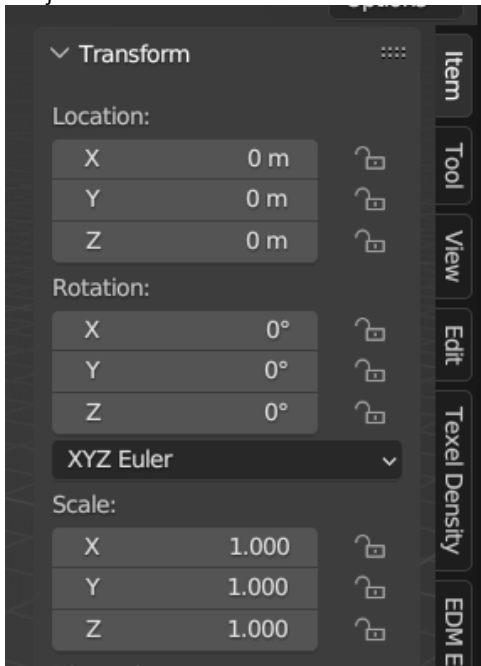
File names should use Latin alphabet letters. Spaces should be replaced with underscores «\_».

The model should be oriented with the nose along the X-axis.

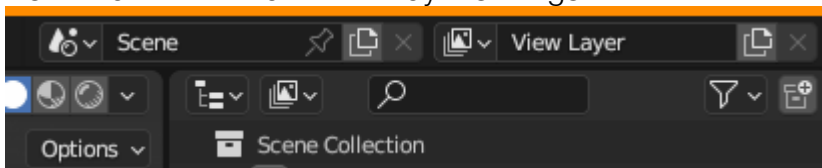




Objects must have their scale and rotation "reset."



Use default Scene and View layer settings.



Export is carried out in Eevee mode.

It is recommended to enable the Wrangler Addon. (installed by default)

## Layers

All scene objects should be organized into layers (scene collections):

The layers window is called the Outliner.

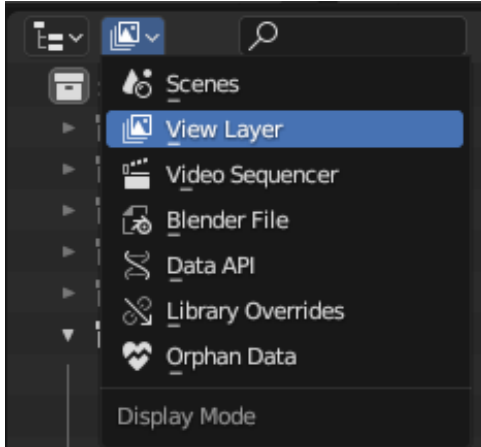


## Eagle Dynamics SA | Plugin Requirements

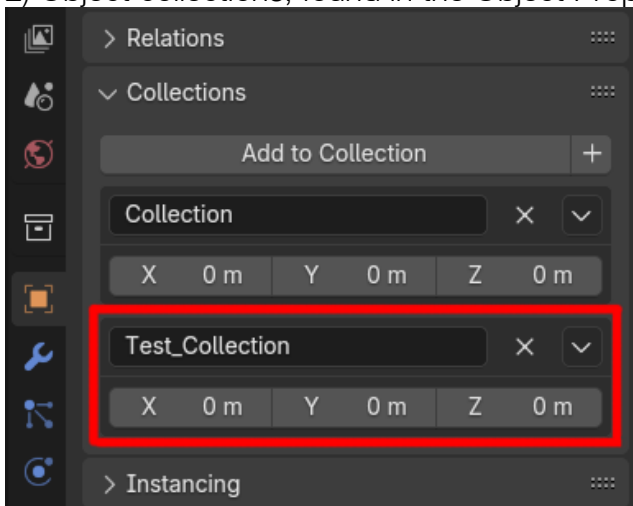


In Blender, there are two types of collections:

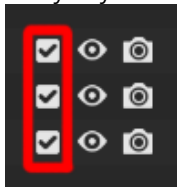
1) Scene collections, shown in the Outliner under View Layer (referred to simply as scene layers). These are considered during export only;



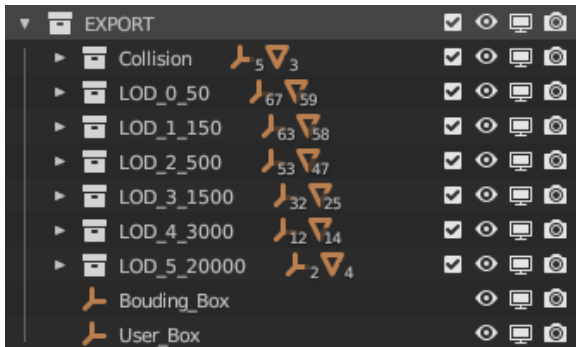
2) Object collections, found in the Object Properties tab. These are not considered during export.



Only layers with an enabled checkbox are exported.



When you create a root layer for export in the scene (name it, for example, Export). Inside it, create layers for the model's LODs and a Collision layer for collisions. Name them according to the LOD (as indicated in the example).



Distribute the model's LODs across these layers. For LOD layers, specify the display distance in meters using an underscore (e.g., LOD\_0\_50).

Helpers (Dummies) used for animation should be unique for each layer.

### Avoid confusion in the hierarchy!

It is crucial to avoid confusion in the hierarchy! In other words, LODs must not overlap or have Parent/Child relationships!

"If this issue occurs, a *'Walker'* error will appear. In that case, you need to remove the dependency.

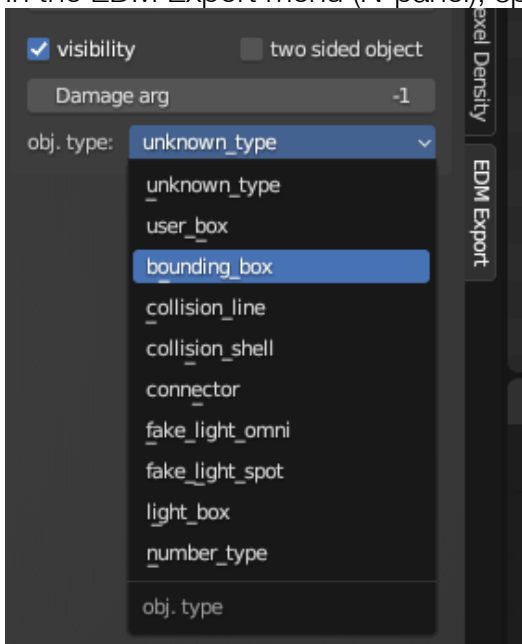
```
walker.destroy()
```

```
^^^^^^
```

## Auxiliary Objects: Bounding and User Box

Create a new Empty Cube in the root layer "Export" and name it **"Bounding\_Box"**.

In the EDM Export menu (N-panel), specify the object type as Bounding Box.







Adjust the size and position of the Bounding\_Box to the model so that it covers its volume with a margin, including during animation (for example, raising a cannon).



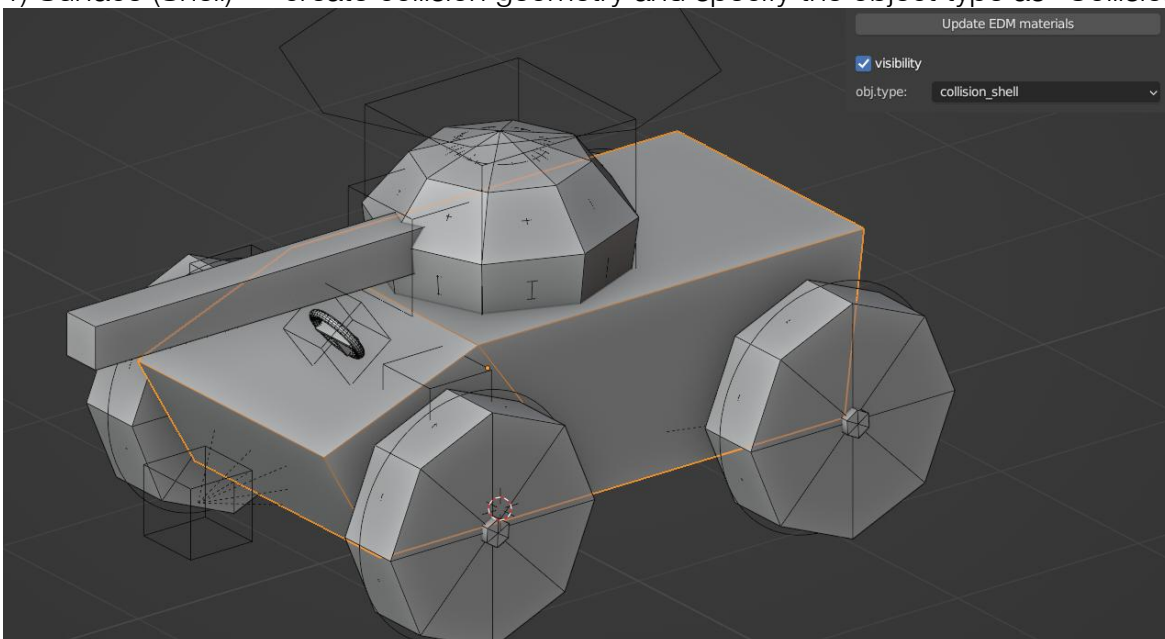
Do the same for the User Box, which should be smaller and simply cover the model.

## Collisions

There are two types: **Shell** for surface collision and **Line** for line collision (e.g., used for aircraft landing gear collision).

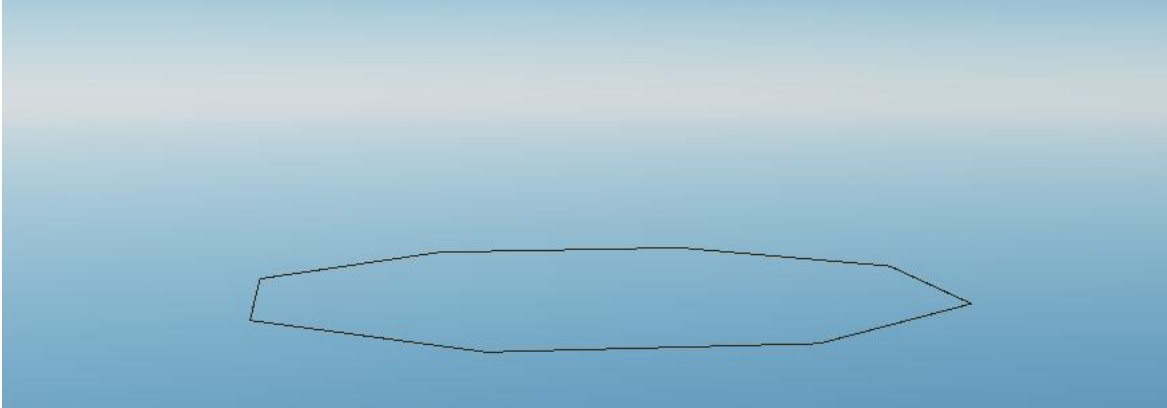
Create a Collision layer in the scene containing all collision objects, each with its own dummies replicating the model's animations.

1) Surface (Shell) — create collision geometry and specify the object type as "Collision Shell."





2) Line — an object with a line of vertices, specify the object type as "Collision Line."



In some cases, it is required that collisions have the same names (for example, parts of a wing divided into sections), but Blender does not allow identical names for different objects. Therefore, during export, the names are truncated at the dot. That is, the collision names: **"TANK"**, **"TANK.001"**, **"TANK.002"** will become the same **"TANK"** after export. This applies to both Collision Shell and Collision Line.



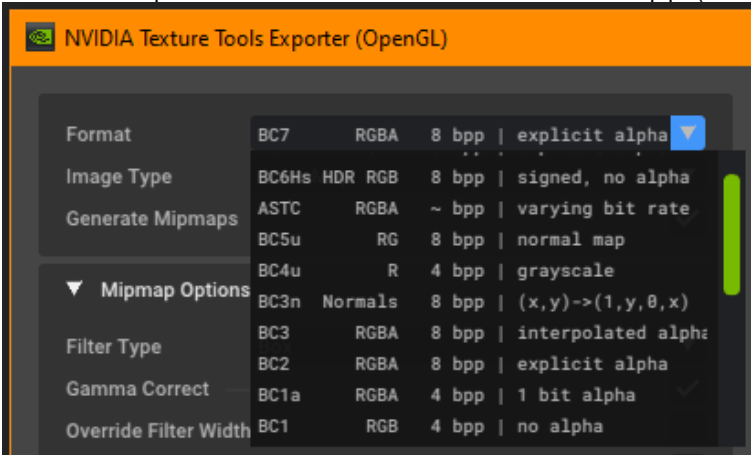
## Texture Format for Export

To optimize the game engine's performance, textures must be converted to DDS format, which requires installing the DDS plugin for Photoshop.

Textures without an alpha channel are saved in BC 1 format.

Textures with an alpha channel are saved in BC 3 format (can also be saved in BC 2 without alpha channel gradation).

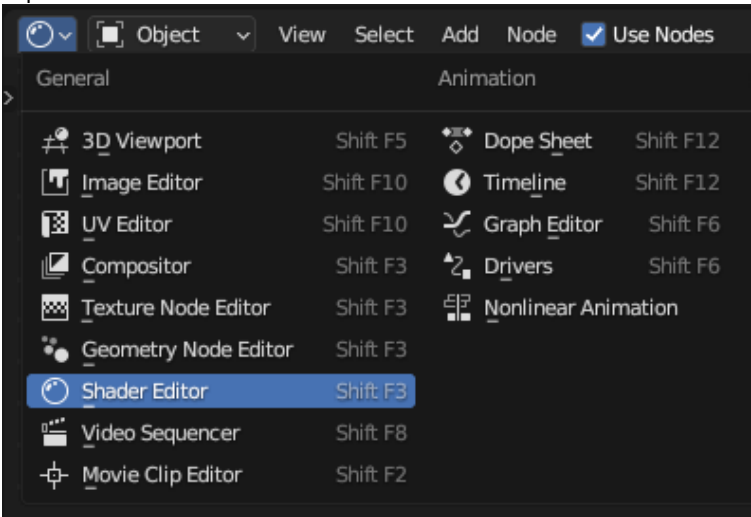
Normal Map in BC 5 format — 8.8.8. RGB 24bpp (uncompressed format).



!!! Attention: dots are not allowed in texture names (i.e. they can only be before the extension). In extreme cases, they can be replaced with a comma.

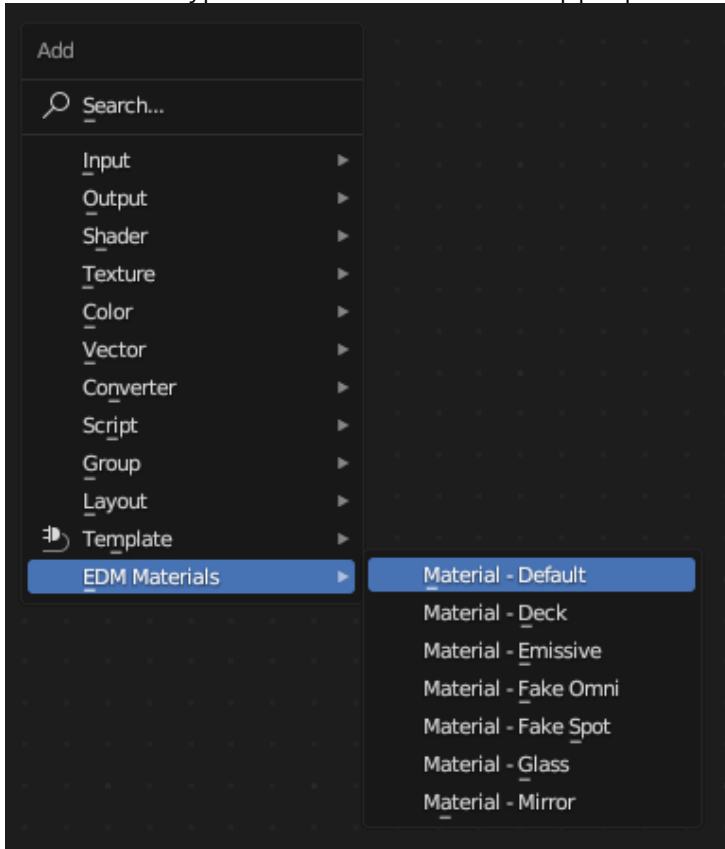
## Material Setup

Open the Shader Editor window.

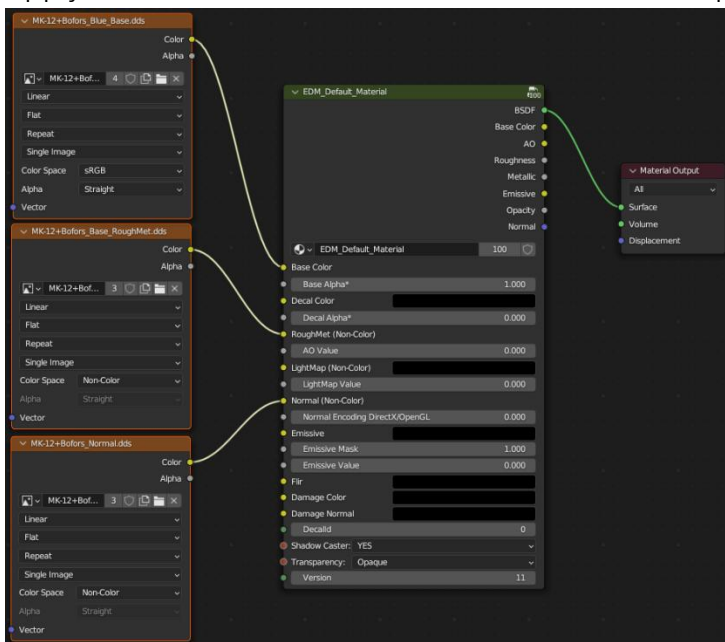




For different types of materials add the appropriate "Nod" from EDM Materials (Shift + A)



Apply textures to the material as shown in the example:



*\* All inputs and outputs in the material group are labelled.*



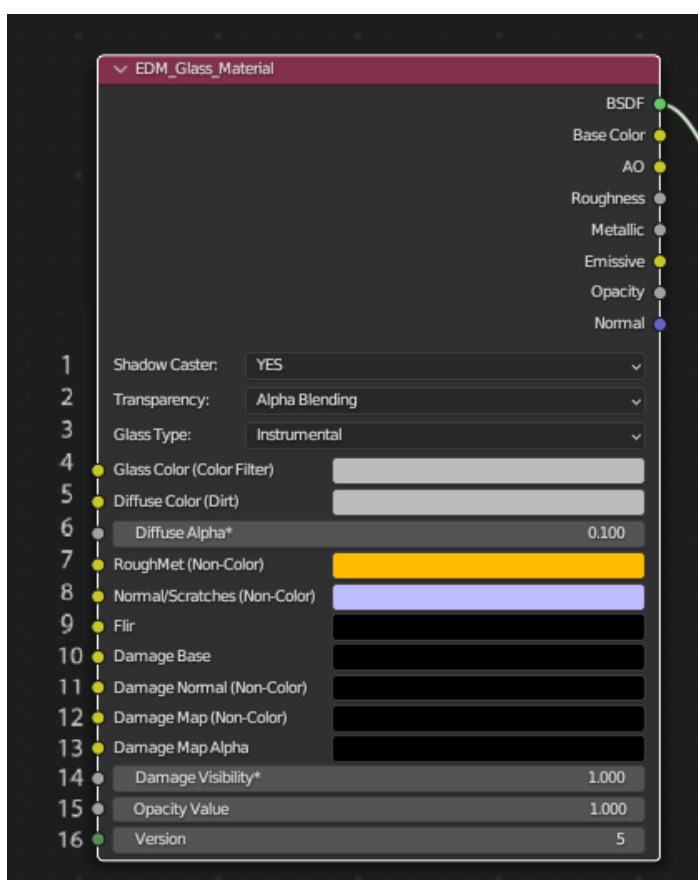
\* Inputs (Non-color) — require switching the Color Space mode of the texture (this is only needed for correct display in the viewer and does not affect export).

\* The right side of the group — outputs for displaying separate texture layers (Roughness, Metallic, etc.)

\* Enable the Node Wrangler add-on for convenience.

\* At the bottom of the group — additional material settings: material transparency, shadows, decal ID.

## Listing Input Values in the Material



\* Some input values are only for visualization in the Blender viewer and do not participate in export: (AO Value, LightMap Value, Normal Encoding)

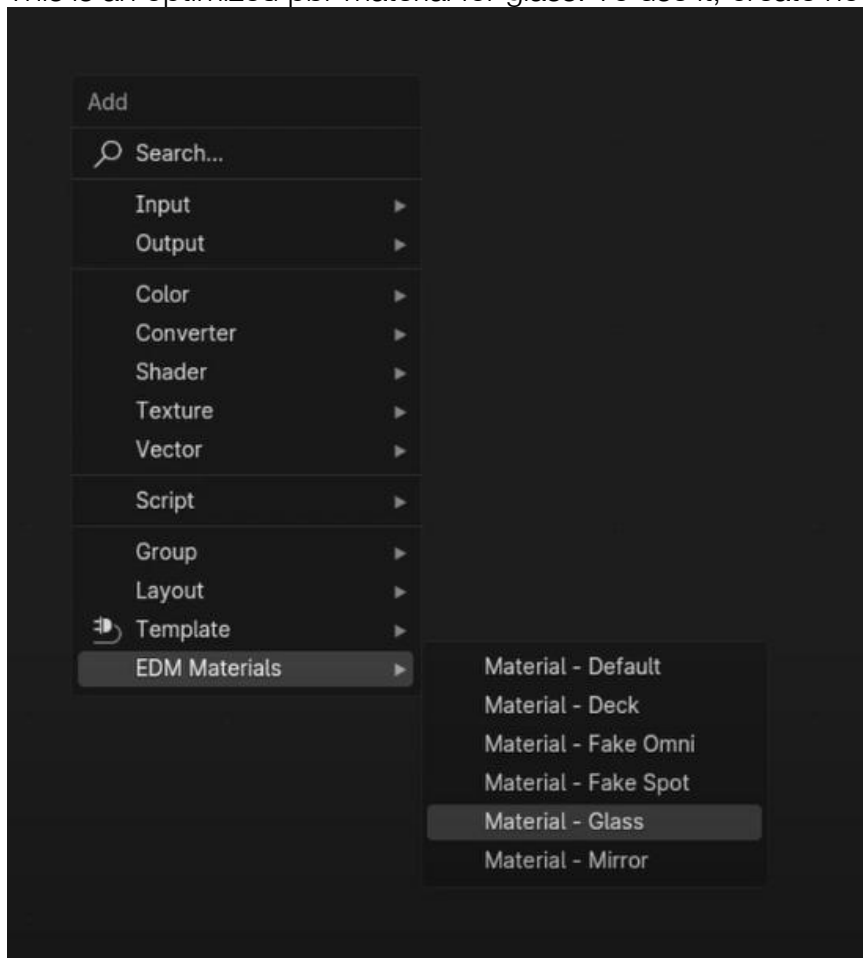
- 1 — Base/Albedo;
- 2 — Alpha Base/Albedo- to display transparency (does not affect export);
- 3 — Albedo of Decal (second layer);
- 4 — Alpha of Decal (does not affect export);
- 5 — RoughnessMetallic texture;
- 6 — Visibility of ?? from RoughMet texture;
- 7 — LightMap (unic ??) texture;
- 8 — Visibility of LightMap;
- 9 — Normal texture;
- 10 — DirectX/OpenGL switcher for Normal (0-DirectX, 1-OpenGL);
- 11 — Emission texture;
- 12 — Emission texture mask;



- 13 — Emission strength regulator;
  - 14 — FLIR texture;
  - 15 — Damage texture map;
  - 16 — Damage Normal map;
  - 17 — Map/Mask of Damage;
  - 18 — Alpha channel of Damage Map (does not affect export);
  - 19 — Damage display scale;
  - 20 — Decal layer identifier (needed to correctly display the order of drawing transparent objects);
  - 21 — Opacity Value - transparency multiplier (used to adjust and animate the strength of transparency);
  - 22 — Material version;
- "Shadow caster" — shadow rendering mode.
- "Transparency" — material transparency mode.

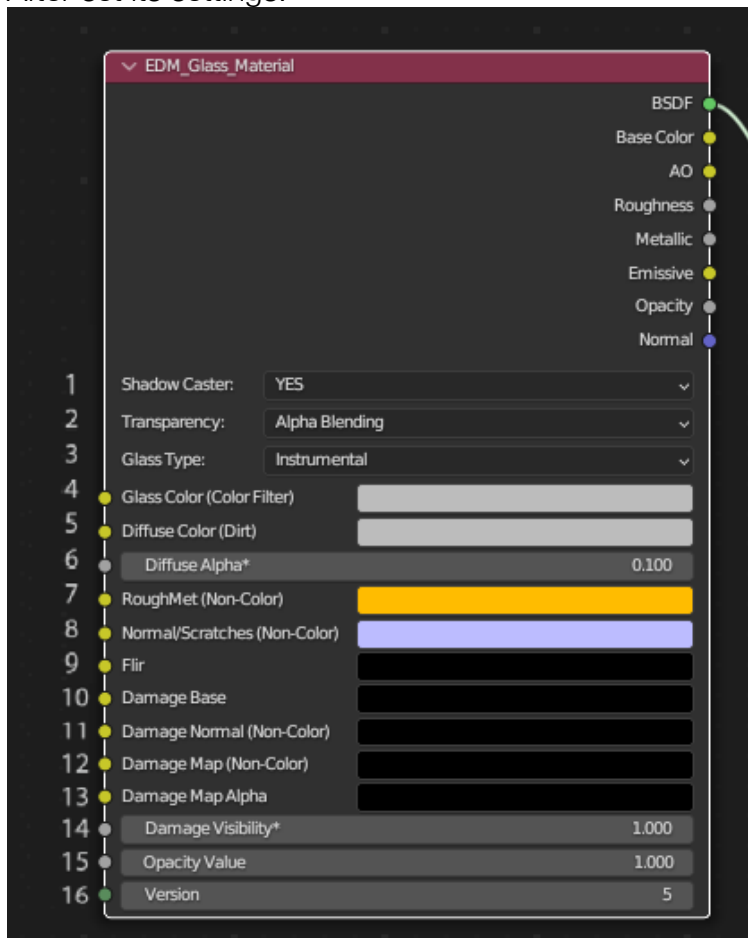
## Glass Material

This is an optimized pbr material for glass. To use it, create node Material - Glass.





After set its settings:



- 1 and 2-shadow and transparency settings (as in the default material);
  - 3 — glass type selection: Cockpit - for cockpit glass, Instrumental - for everything else;
  - 4 — Glass Colour or colour filter is the albedo of the glass (not dirt or streaks, but the colour of the glass itself);
  - 5 — Diffuse Colour(Dirt) -the colour of what covers the glass (mostly dirt and streaks), you need to connect the alpha channel in 6;
  - 7 — RoughMet texture;
  - 8 — Normal/Scratches - mainly used to display scratches (displayed in the game, not displayed in the viewer);
  - 9-16-Flir and damage textures, etc. They work the same way as in the default material.
- Export and check in the viewer;

## Animation

**Action** — is an animation block, a set of keyframes grouped under a single name and assigned to a specific object.

**Argument** — is an action with a numeric prefix in its name, which determines how the animation is triggered in-game.

**Example:** "2\_TowrRottion" — the number 2 is the argument ID, and "TowerRotation" is a user-friendly



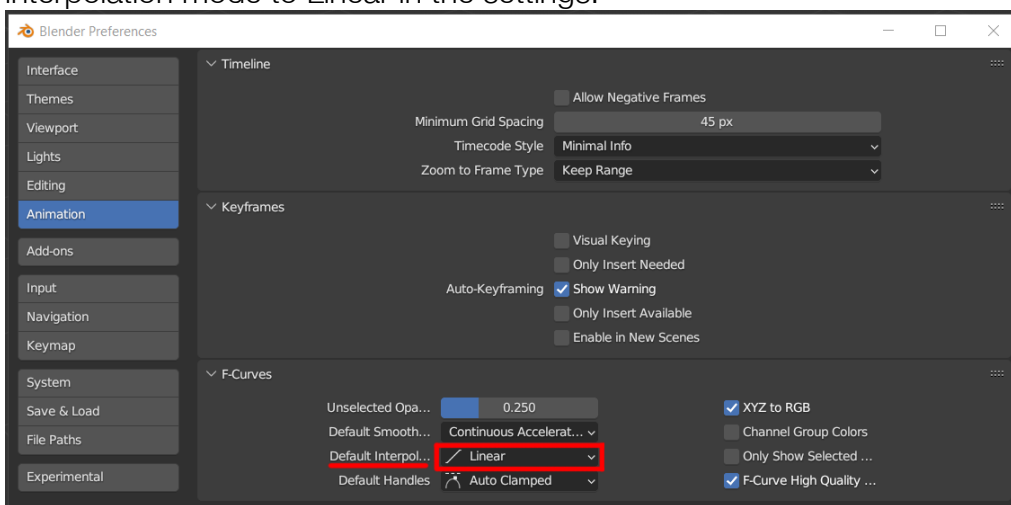
label, does not affect export.

Basics:

!!! If the argument number is missing from the name, the animation will not be exported.

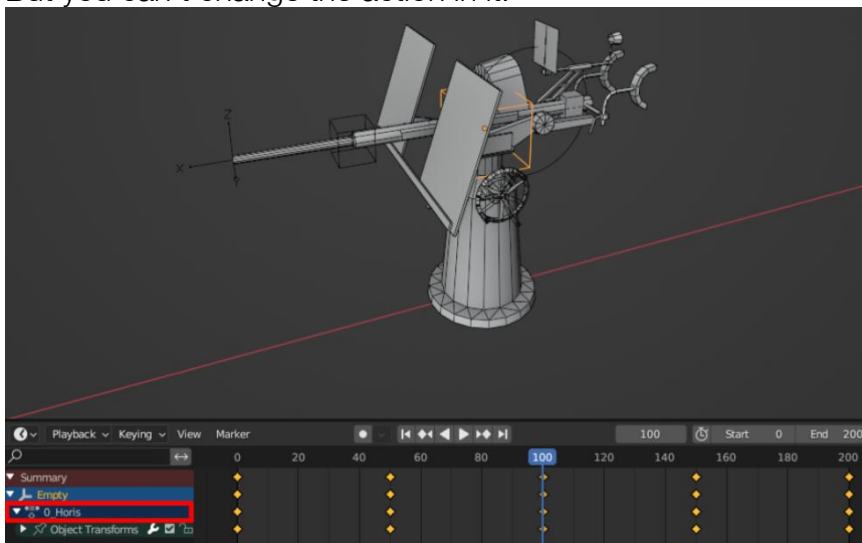
Blender uses a timeline range from 0 to 200, where frame 100 represents the neutral/default position. Pressing Reset in the EDM Export panel automatically sets this range. It is recommended to press it before starting to work with animation.

Interpolation. To correctly display the animation "as in the engine", you need to set the animation interpolation mode to Linear in the settings.



Several windows are used to work with animation:

1) The Timeline window is used to work with animation and specify an argument in the action name. But you can't change the action in it.

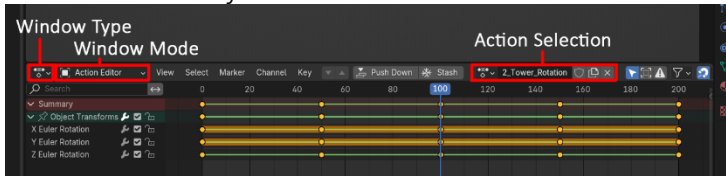


*\*It is open by default and is quite suitable for creating simple animation.*

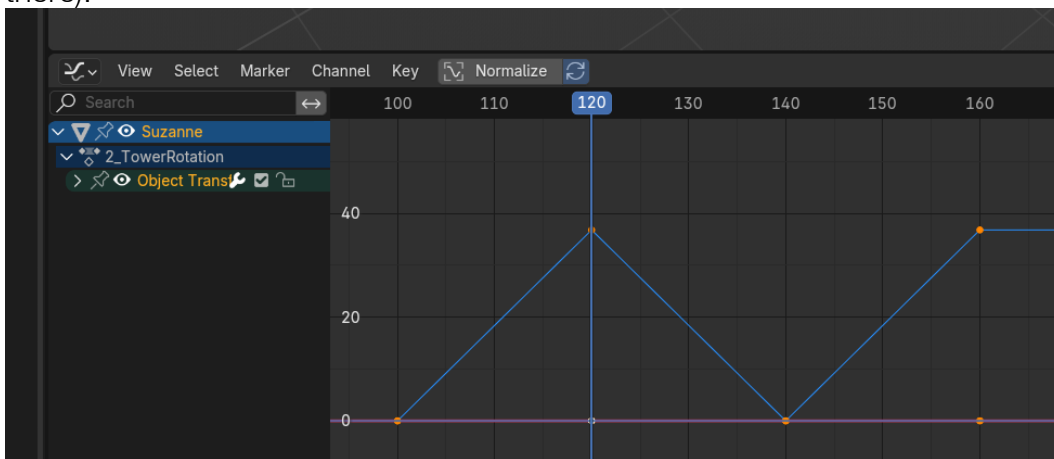




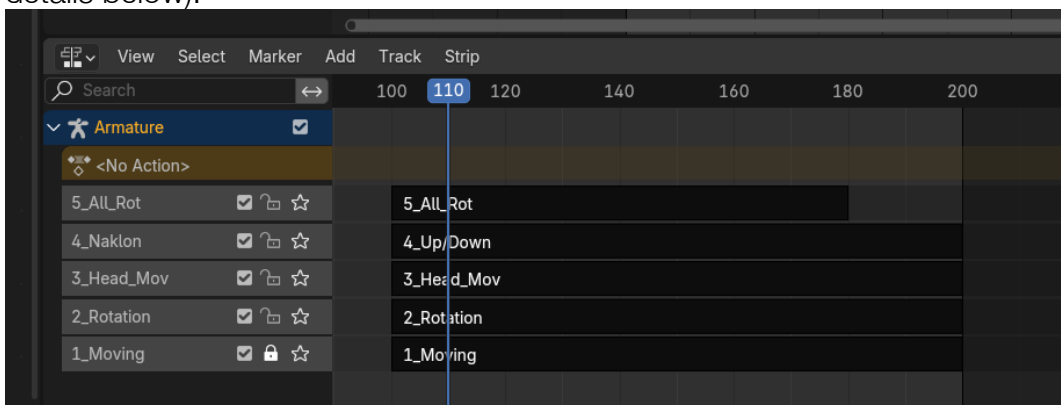
2) The DopeSheet window of the Action Editor type is also convenient for working with animation, and also allows you to switch actions.



3) Graph Editor (analogous to CurveEditor from 3DsMax) — needed to edit interpolation curves between keyframes. The engine uses only "Linear" interpolation. Therefore, changing and setting up any other modes can only be useful when baking animation. (F3 to search, enter "Bake Action" there).



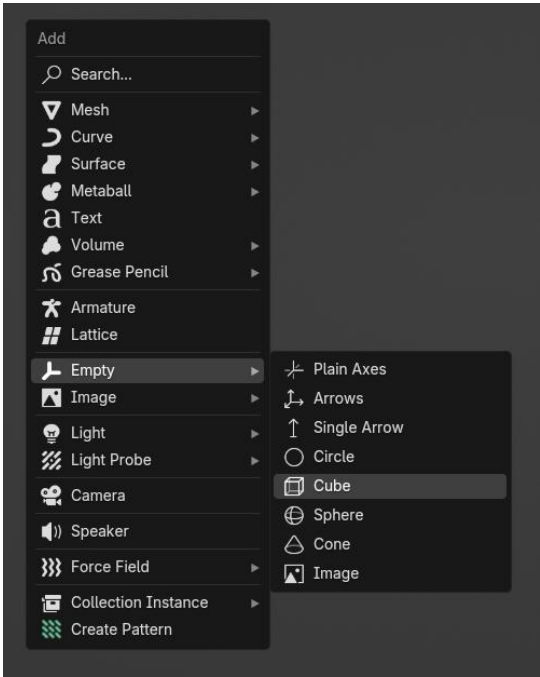
4) NonLinear Animation or NLA Editor — for exporting different animations on one Armature. (More details below).



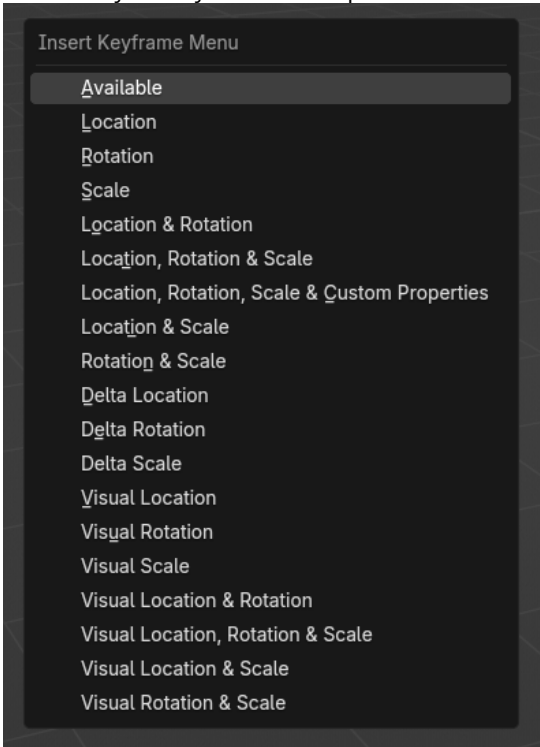


## Animation of objects/empties

It is recommended to animate models using empty objects. One animation per object.



After modifying the object's position, rotation, or scale, press K to insert a keyframe, and then select the key data type: **Location/Rotation/Scale** — to change the corresponding transforms. Available - adds keys only for those parameters that have already been created earlier.



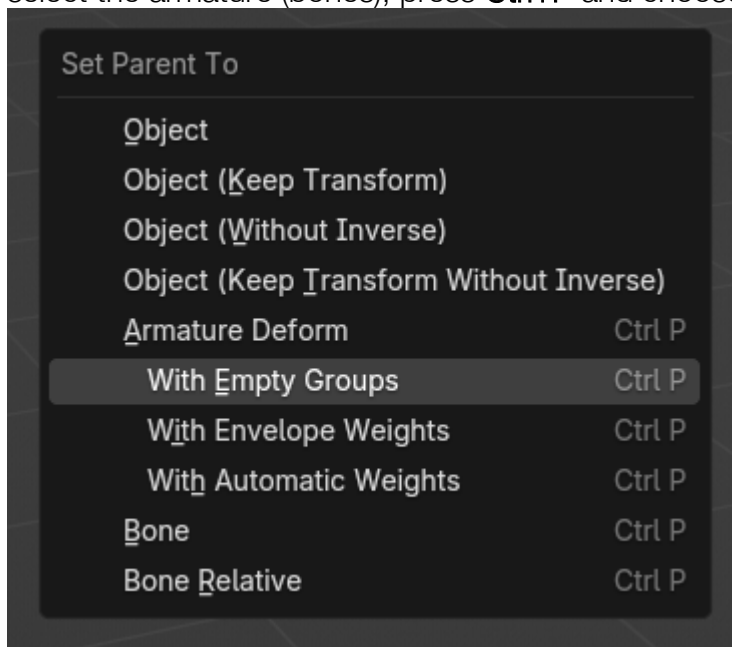


*\*For example: if you've already created rotation keyframes and then use Available, Blender will only add new rotation keys — position and scale will remain unchanged, since they haven't been keyed yet.*

!!! Don't forget to include the argument number at the beginning of the action name. Without it, the animation will not be exported.

## Bone Animation

Bone animation begins by attaching the object to the armature: select the object, hold Shift and select the armature (bones), press **Ctrl+P** and choose the parenting mode.



**With Empty Groups** — creates empty vertex groups; weights must be assigned manually.

**With Automatic Weights** — vertex weights are calculated automatically.

After parenting, adjust and verify the weights to ensure the animation works correctly.

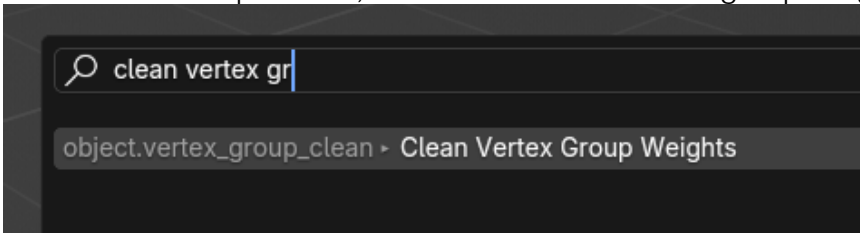
!!! Important engine limitation: no more than four bones can influence a single vertex.

If this rule is violated, the following error appears: "Skin vertex has more than 4 influencing bones"

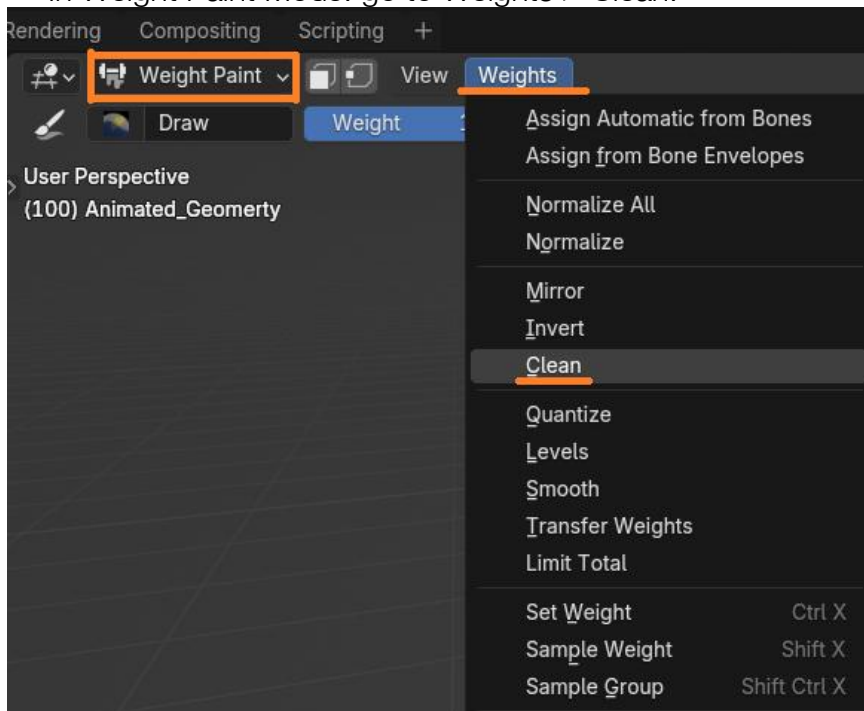


Solutions:

- 1) Remove extra influences manually. Edit the vertex groups directly to delete unnecessary bone weights.
- 2) Use automatic weight cleaning:
  - In Edit Mode: press F3, search for: "Clean vertex group weights"



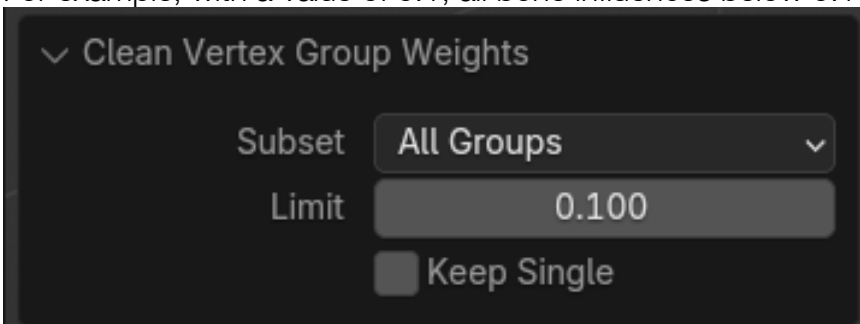
— In Weight Paint Mode: go to Weights > Clean.



**Cleaning settings:** enable All Groups.

Set a weight limit threshold.

For example, with a value of 0.1, all bone influences below 0.1 will be removed.



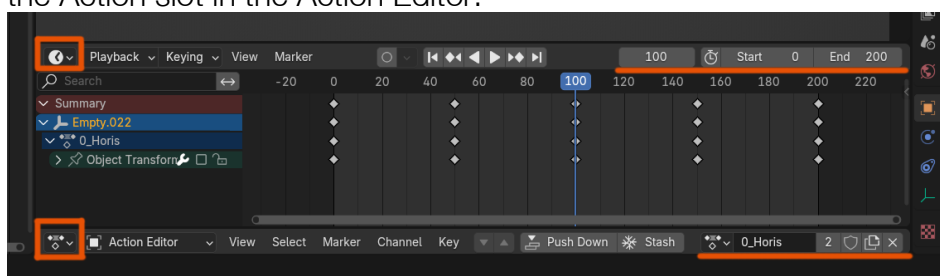


## Two ways to export bone animations

Bones can be exported either with a single animation — through a simple Action (like with a regular object), or with multiple animations — using the NLA Editor.

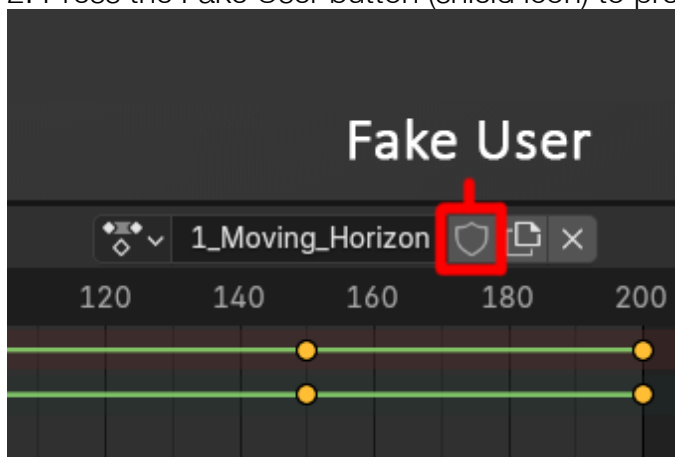
1) Exporting a single animation — the one located in the Action Slot in the Action Editor. The export is performed in the standard way. It is important not to forget to set the argument number in the Action name.

For convenience, you can use both windows simultaneously: **Timeline** and **DopeSheet** in Action Editor mode. This way you have access to the playback frame settings and to the Action slot in the Action Editor.



2) Exporting multiple animations on a single Armature — you need to use the NLA Editor.

1. First, create an animation (Action) and specify the argument number in its name.
2. Press the Fake User button (shield icon) to prevent the Action from being deleted.

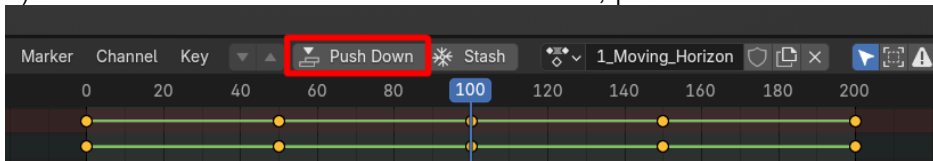


!!! Without "Fake User", Blender may delete the action upon restart, as it automatically removes unused actions for cleanup/optimization.

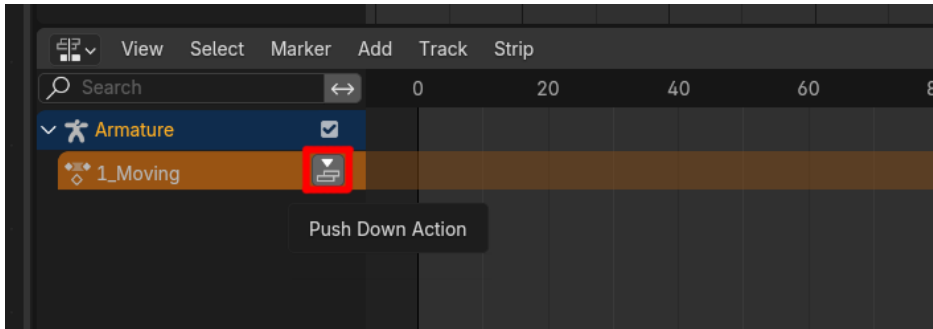
3) Open the **NLA Editor (Nonlinear Animation window)** — it displays a list of tracks/actions you want to export.



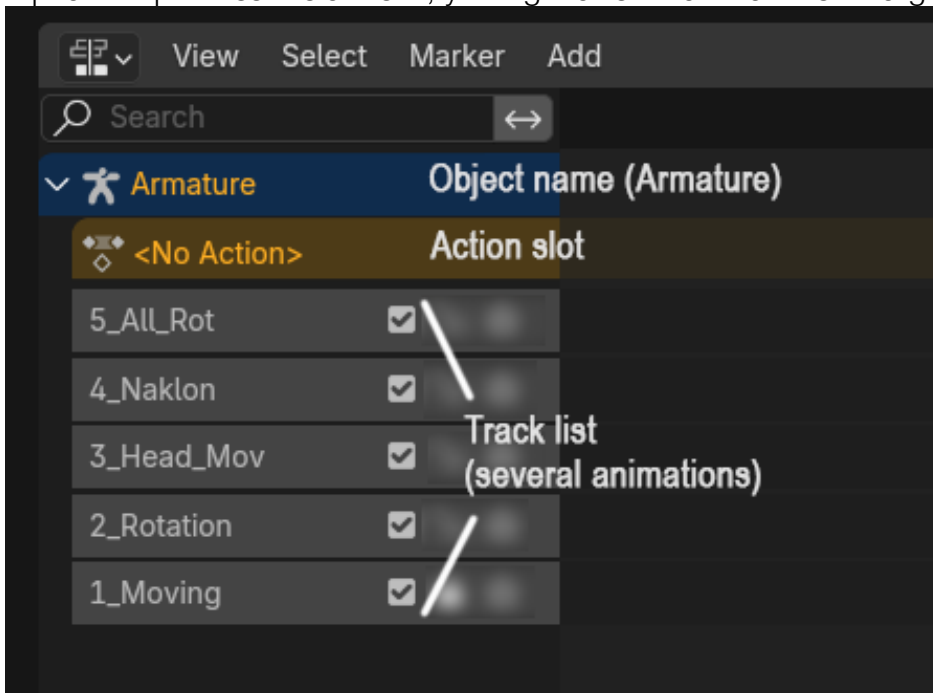
4) To add the created action to the NLA list, press Push Down in the Action Editor window.



Or in the NLA Editor window — Push Down Action.



Make sure Action appears in the NLA track list. Create the next animation and repeat the process. As a result, you'll get a list of animations — arguments.



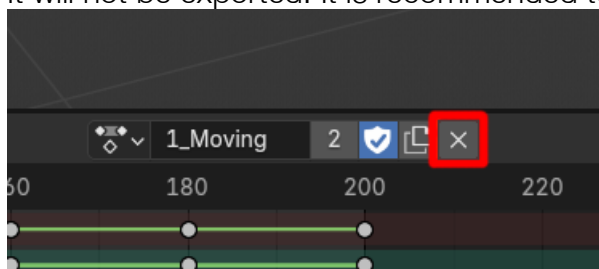


To test, disable all checkboxes and enable only the one for the desired argument (checkboxes do not affect export).

Make sure that all Actions have an argument number specified. To change it, you may need to remove the Action from the Track list, rename the Action itself in the Action Editor, and then add it back to the track list.

To delete an unnecessary track from the NLA list — right-click and choose Delete Tracks.

**Important!!!** When multiple animations are present in the NLA Editor, it no longer matters which Action is selected in the Action Slot of the Action Editor. It will not be exported. It is recommended to clear it (set it to “X”).



**!!! When exporting bones, a Bounding Box is required.**

Done! Export and check.

## Applying Transforms

**!!! Important Conditions for Exporting Bones with Animation**

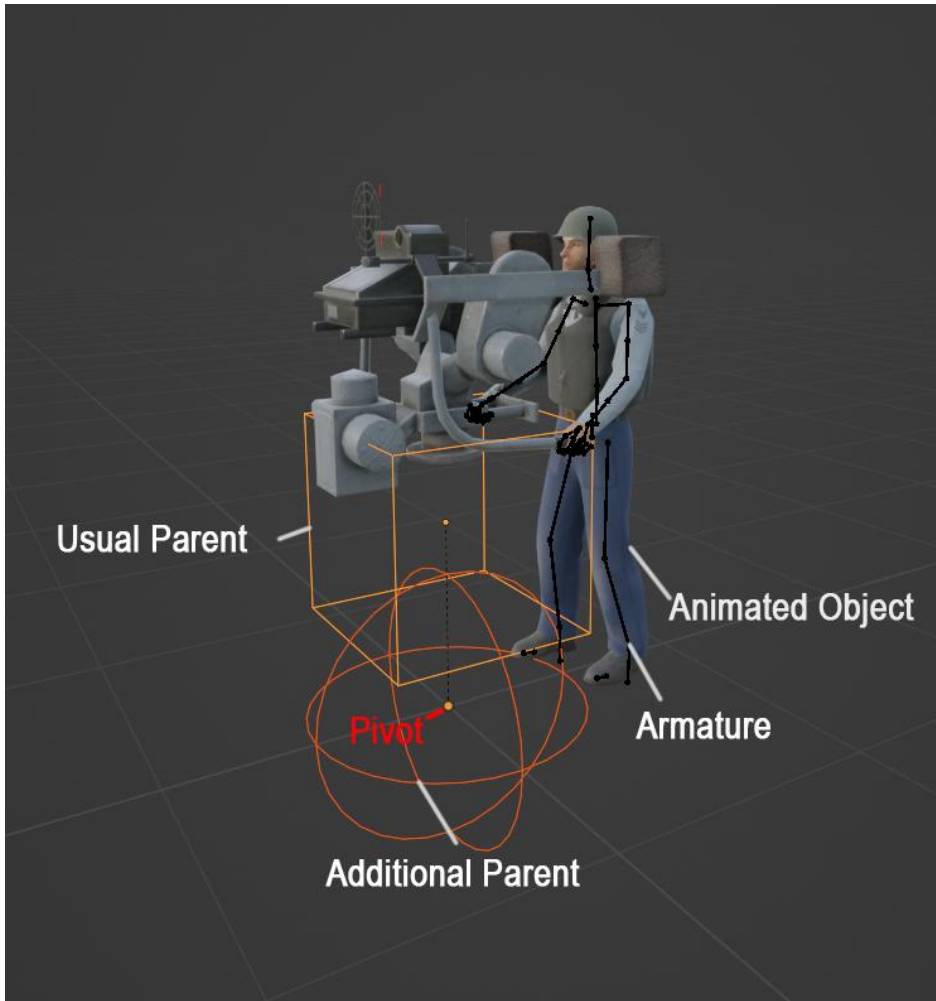
Due to differences between Blender and the game engine, problems may occur when exporting bones with animation.

To avoid them:

- 1) Apply Rotation and Scale transforms, and align the Pivots in the same place for both the Armature and the Animated Object it belongs to.
- 2) If the Armature has a Parent (for example, an Empty controlling the rotation of a gun), whose Pivot cannot match the Armature's Pivot (since it defines the rotation point):
  - You need to create a new Empty — the “Additional Parent.”
  - Its transforms must also be applied, and its Pivot must match the Pivot of the Armature.



In other words, the standard scheme — hierarchy with bones looks like this:



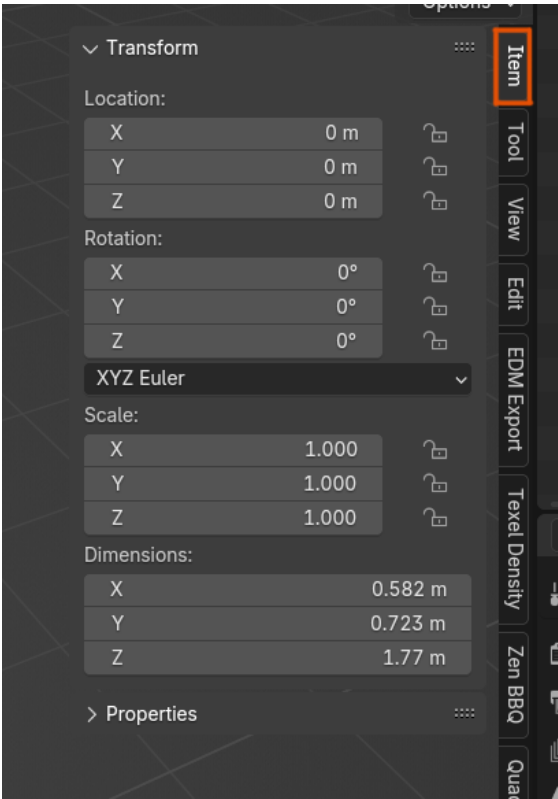
Usual Parent (with its own Transform and Pivot) > Additional Parent — Armature's Empty (has Transform identical to Armature) > Armature/Bones > Animated Object (has Transform identical to Armature).

That is, the Applied Scale and Rotation transforms and identical Pivots must be present on three objects: **Additional Parent**, **Armature/Bones** and **Animated Object**.

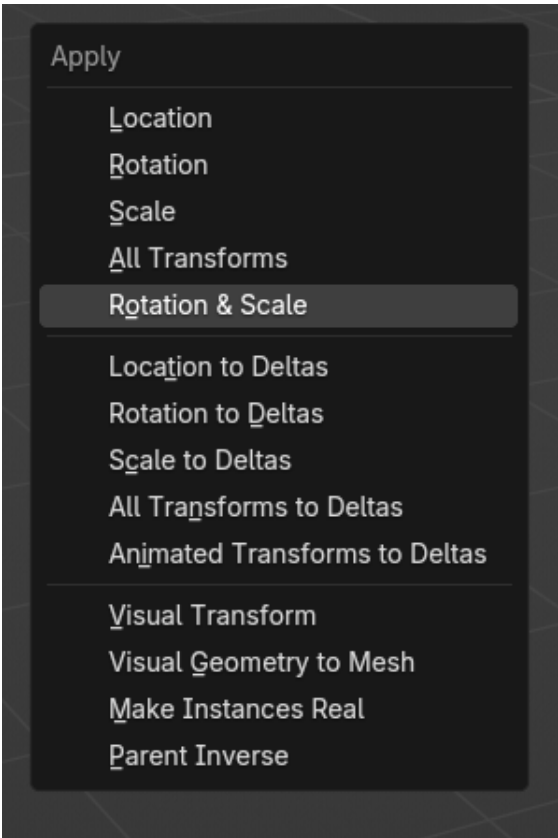




Transforms are checked in the Item tab of the N-panel.

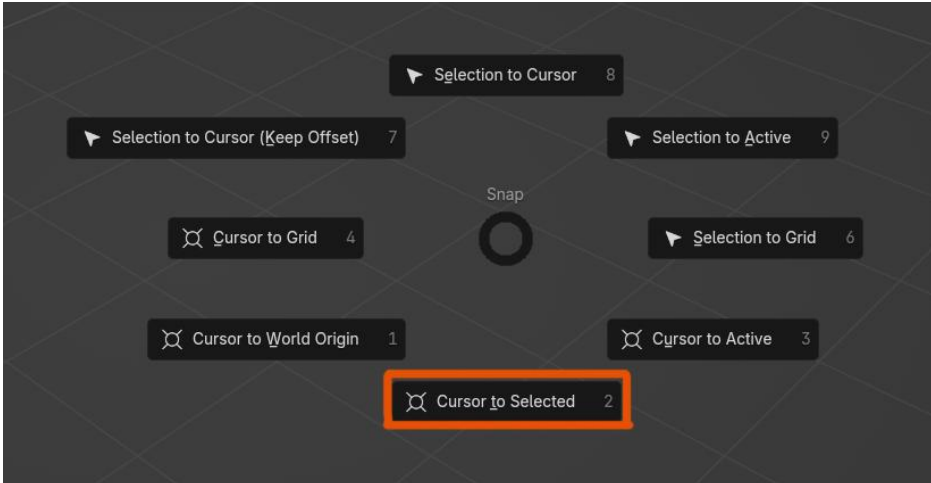


Apply transforms with **Ctrl+A**.

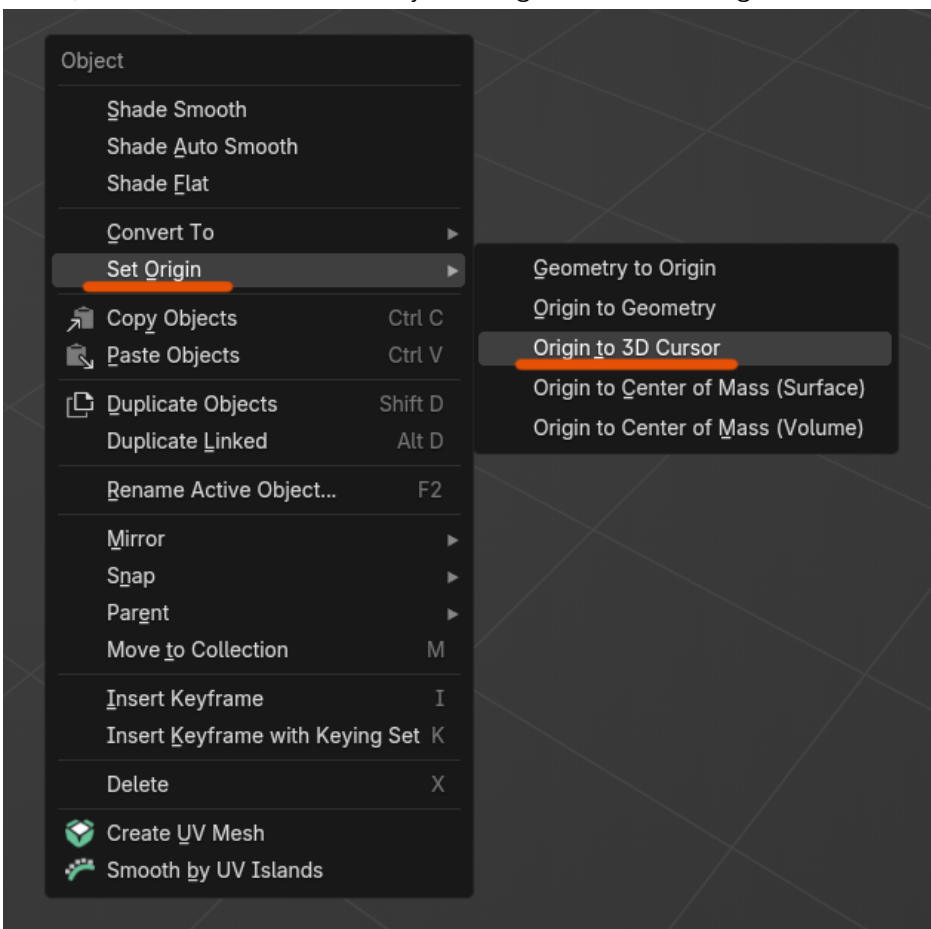




To align Pivots: set the cursor to the Pivot of the bones with Shift+S > Cursor to Selected.



Then, select the Animated Object > right-click Set Origin > to 3D Cursor.





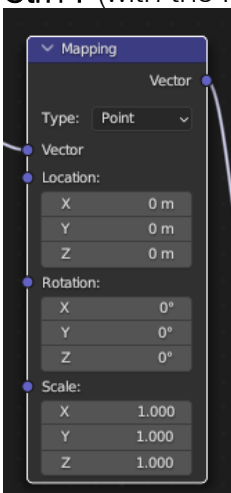
!!! Sometimes **unApplied** Transforms and Pivots placed in different locations can cause problems. From the outside they may look like a frame from a horror movie — this is a transform issue.



!!! For correct operation the Armature must not be animated as an object (only the bones themselves).

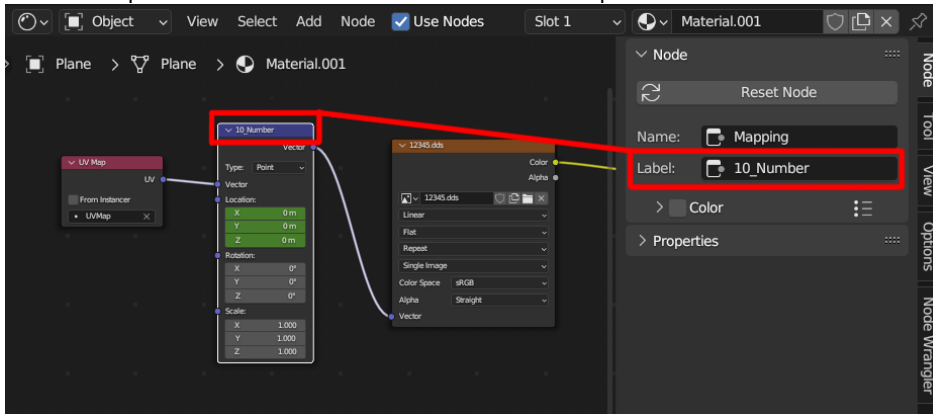
## UV animation

To animate UV offset (texture movement) in the Shader Editor, select the texture node and press **Ctrl+T** (with the Node Wrangler addon enabled).

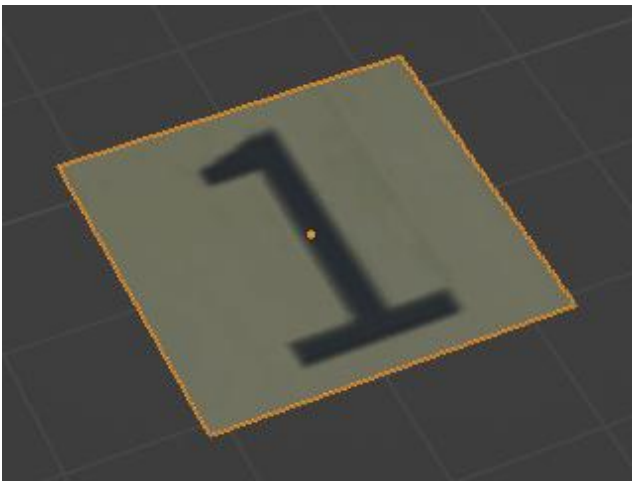
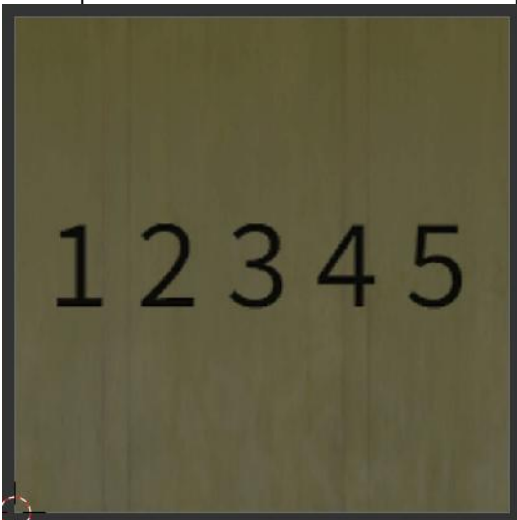




Specify the animation argument in the Label. The node name will change to the argument name with its description. Attach a UV node with the specified UV channel.



Create an animation by changing the Location and saving keys (**key "I"**). Example of a texture atlas and a completed animation.



Roughness Metallic and Normal using the same Mapping node as the Base.



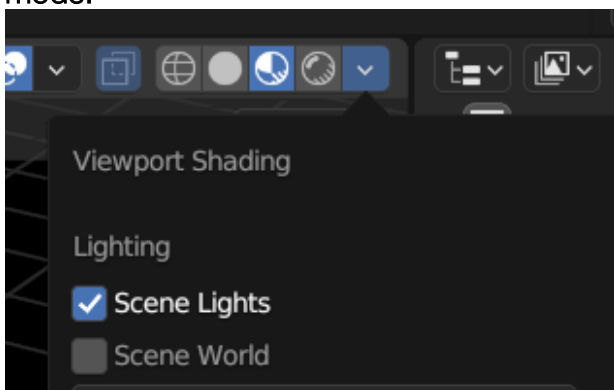
## Lighting

Divided into lamps (providing real light) and fakes (images - light points)

## Lamps

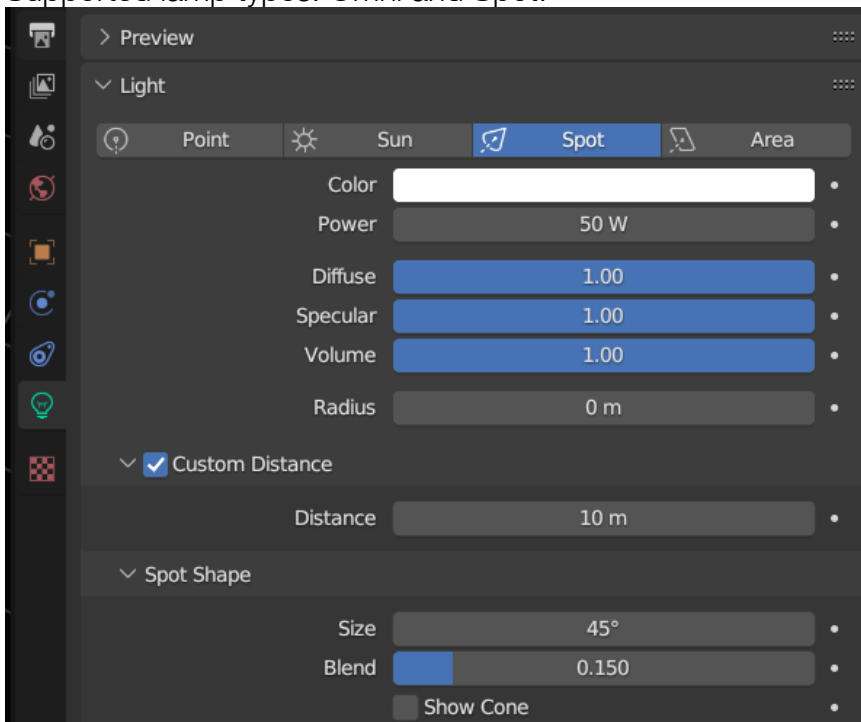
It is recommended to work in **Eevee render mode**.

To adjust the light setup, use **Material View display** mode with **Scene Lights** enabled or **Rendered mode**.



!!! The exporter tries to transfer lamp characteristics to the engine as accurately as possible so that they look the same in Blender, but differences will still exist due to technological rendering differences. Therefore, lighting is set up in Blender first, then adjusted by looking at the viewer.

Supported lamp types: Omni and Spot.





Characteristics used for export:

**Color** — light color.

**Power** — intensity.

**Specular** — ability to cast glare.

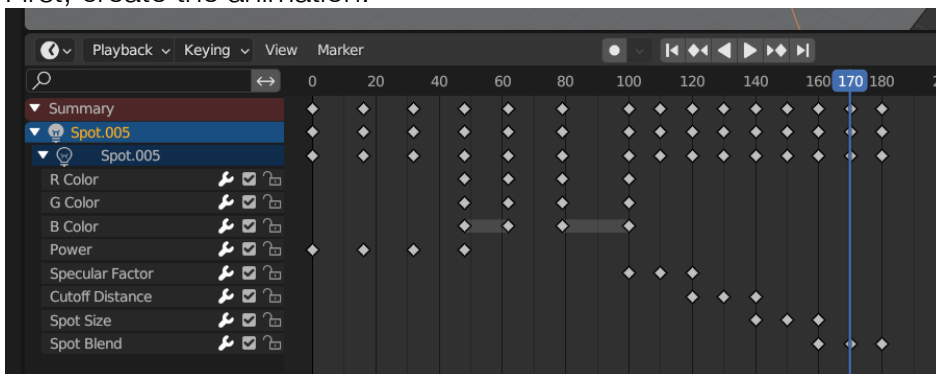
Custom distance - specifies the lamp's working distance (default is 50m if unchecked).

**Spot shape** — used for Spot lamps: Size - outer light cone angle.

**Blend** — softness of the light border (0-hard, 1-soft).

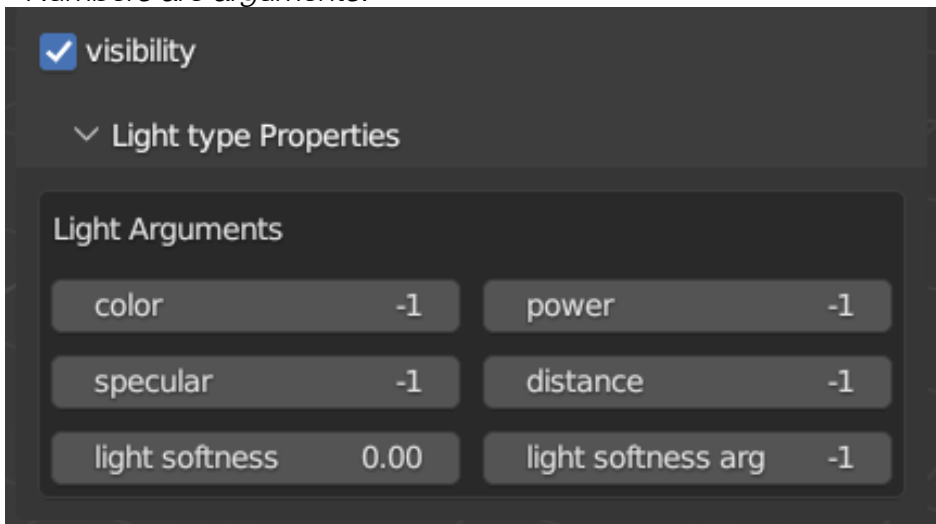
Lamp characteristics animation - performed in the "EDM Export" panel.

First, create the animation.



Then specify its arguments in the N-panel.

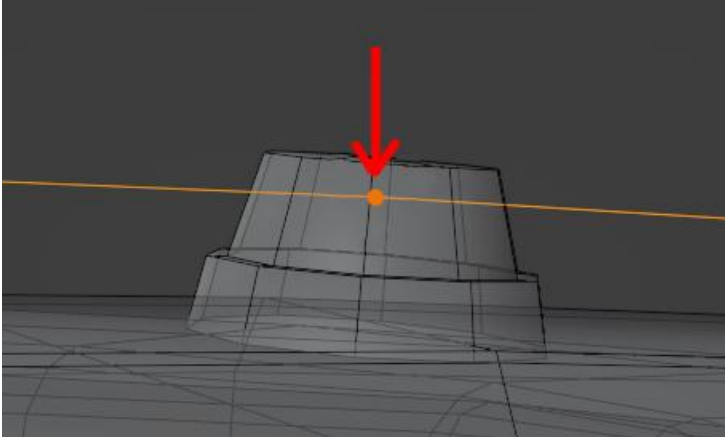
*\* Numbers are arguments.*



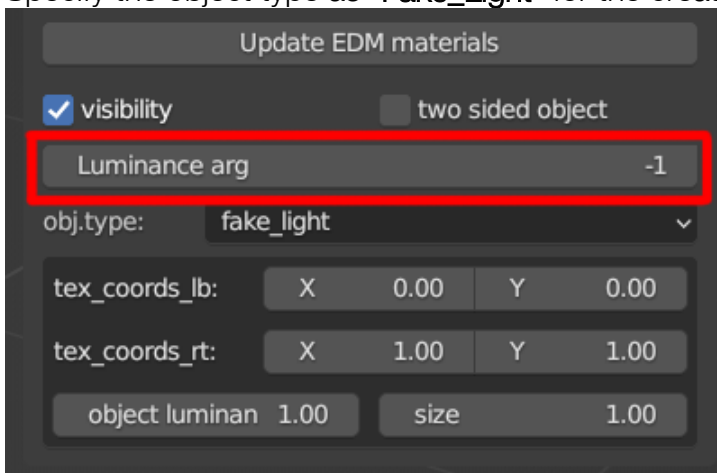


## Fake Lights (BANO-light points)

*\* Implemented using geometry with vertices as light points.*



Specify the object type as **"Fake\_Light"** for the created geometry.



Then specify the edges of the light point texture area inside the atlas (if any):

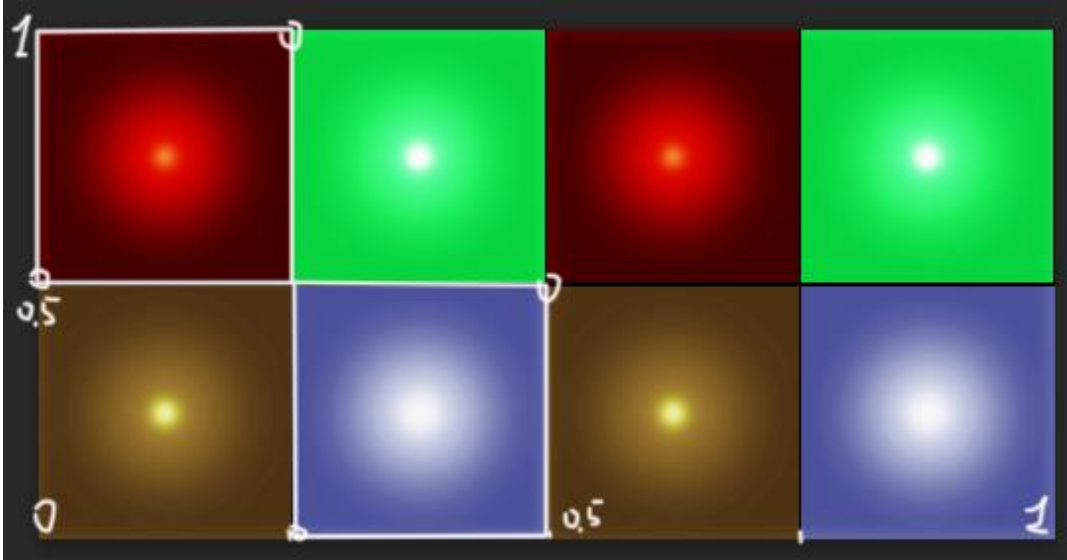
*\*tex\_coords\_lb - lower left edge*

*\*tex\_coords\_rt - upper right edge*

Object luminance — brightness multiplier, also allows to create animation (the argument is specified in the action name).



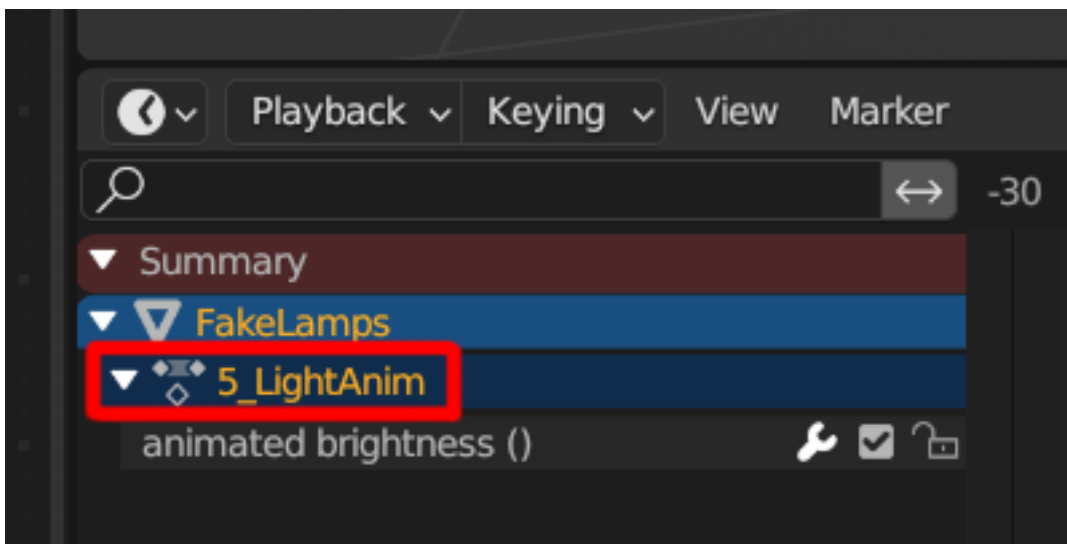
Size — specifies the size of the banner/light in meters.



\* Example of a texture atlas. (the right four have a truncated alpha channel shape and simply look similar in colour).

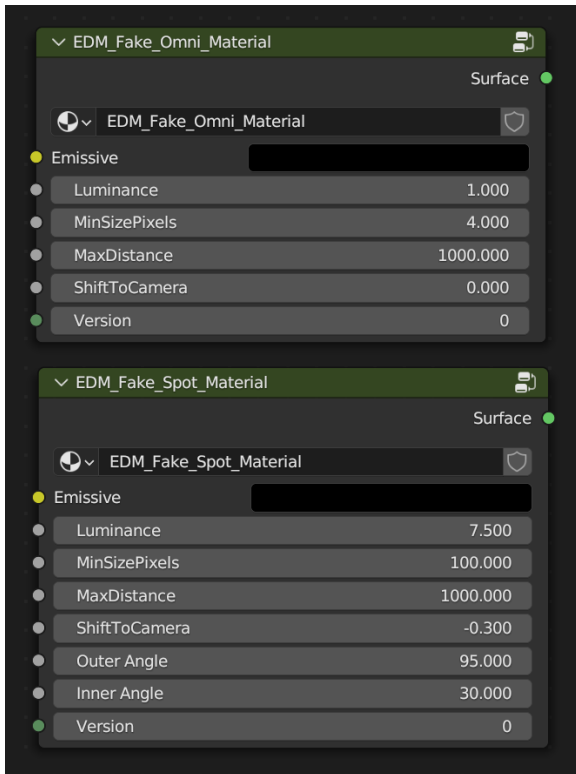
\*\* Example of red "ball" UV `tex_coord_lb: x=0.0;y=0.5; tex_coord_rt: x=0.25;y=1.0`

\*\* Example of blue "ball" UV "ball" `tex_coord_lb: x=0.25;y=0.0; tex_coord_rt: x=0.5;y=0.5`



Create a node in the material: EDM\_Fake\_Omni\_Material or EDM\_Fake\_Spot\_Material.





Specify characteristics inside the group:

**Luminance** — the brightness of the light,

**MinSizePixels** — the minimum size of the light spot when moving away,

**MaxDistance** — the distance after which the light will disappear,

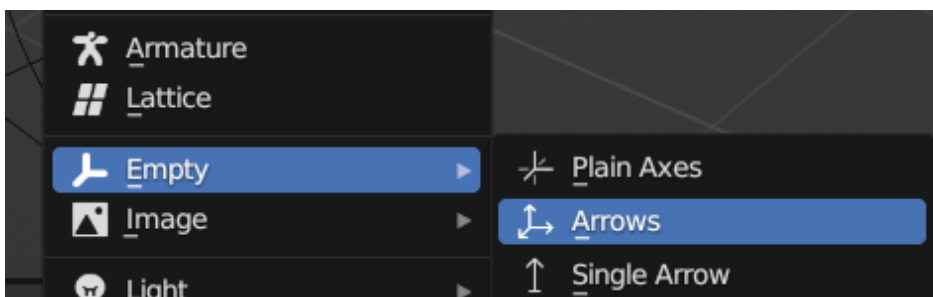
**ShiftToCamera** — shift the light plate towards the camera (to remove noticeable intersection with the geometry),

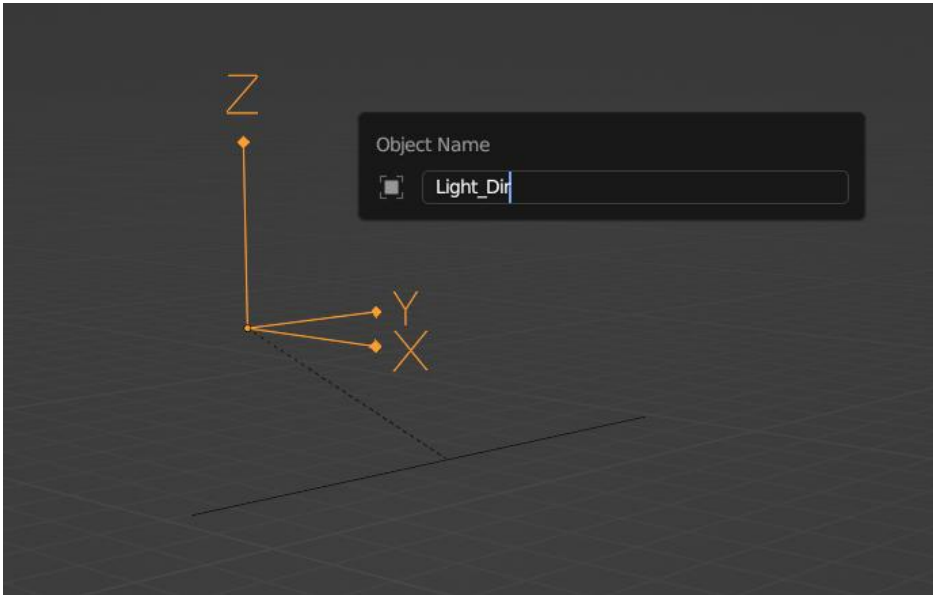
**OuterAngle** — the outer angle of the directional light (spot),

**InnerAngle** — internal.

*\*Add the texture to the Emissive channel in the group.*

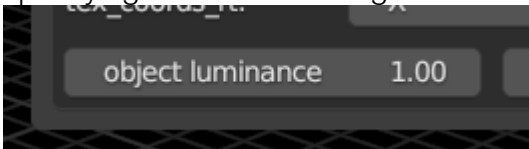
For Fake Spot, you need to set the direction (where it is looking). To do this, assign Empty( Arrows) to fake spot geometry. Lights direction will be defined by direction of X - axis.





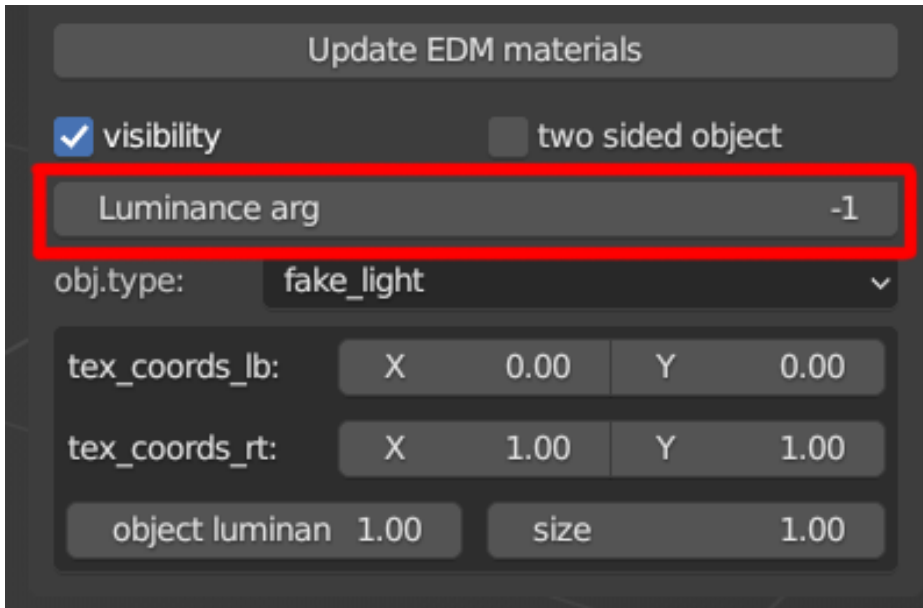
Brightness animation for Fake\_Light (blinking) can be set:

1) For Fake Lights geometry — using **Object luminance** animation in the EDM Export panel and specifying the animation argument in the action.



2) For material — an animation of the Luminance parameter is created and the animation argument is specified in the **Luminance arg** in EDM Export.





*\* If animation is created both in geometry and material, geometry animation takes priority.*

Example of Fake Omni



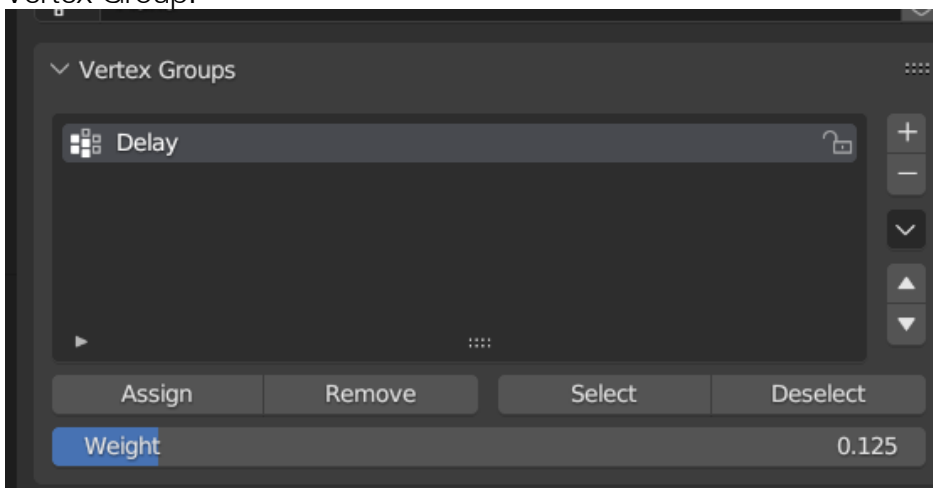


Example of Fake Spot



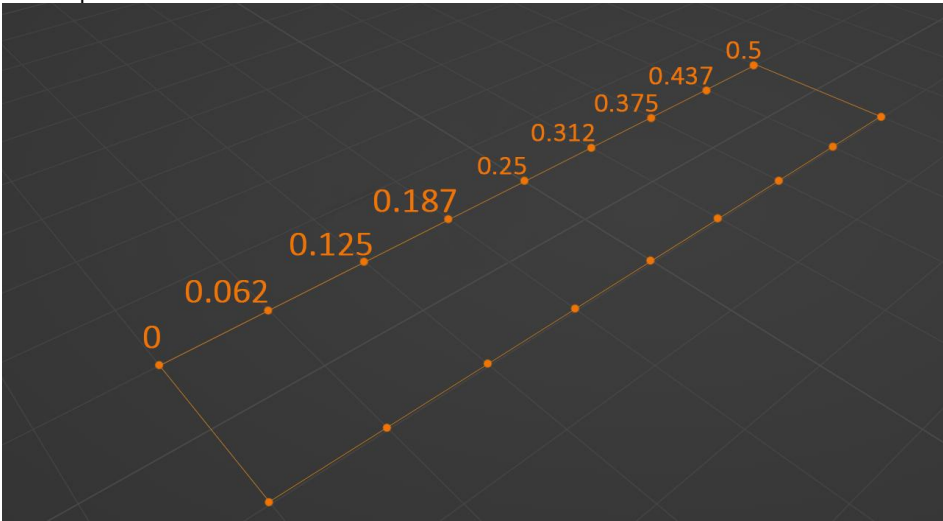
## Rabbits Lights

It is a fake Lights function where brightness changes along a lamp line (a light moves along the line). Used for landing strips on ground and aircraft carriers. Works for Omni and Spot. A general brightness animation (flash) is created and the delay/lag time of the lamps is specified using Vertex Group.





Example:



Each vertex's delay is easily adjusted and checked in the Item menu.

X

Y

Z

Options

▼ Transform

Vertex:

|   |      |
|---|------|
| X | 0 m  |
| Y | -1 m |
| Z | 0 m  |

Global

Local

Vertex Data:

|               |      |
|---------------|------|
| Bevel Weight  | 0.00 |
| Vertex Crease | 0.00 |

▼ Vertex Weights

All

Deform

Other

Delay

0.250

×

Normalize

Copy

> Properties

Item

Tool

View

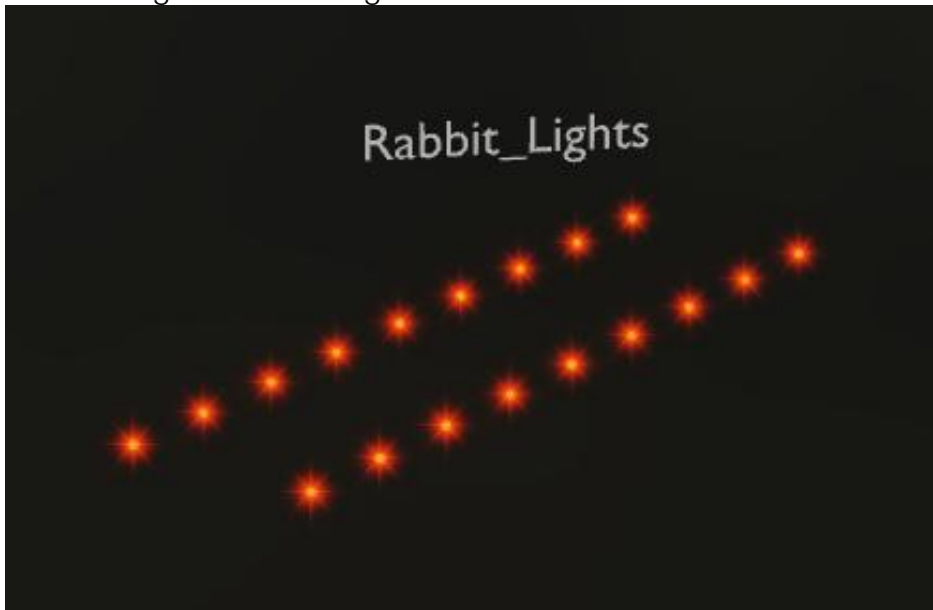
Edit

Texel Density

EDM Export

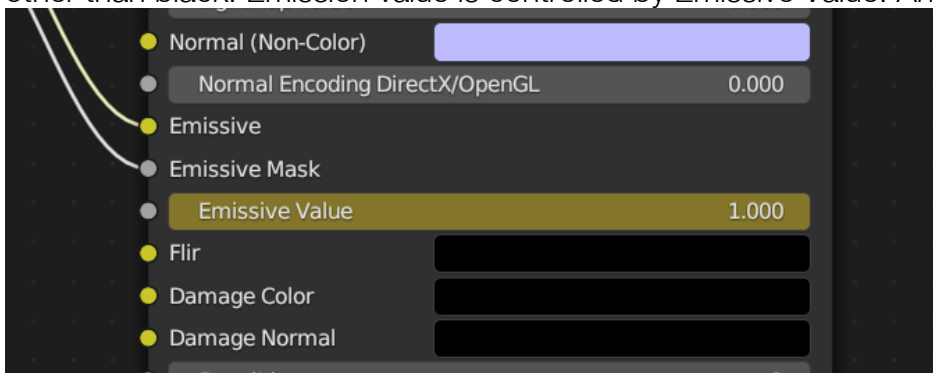


Result — light moves along the line.



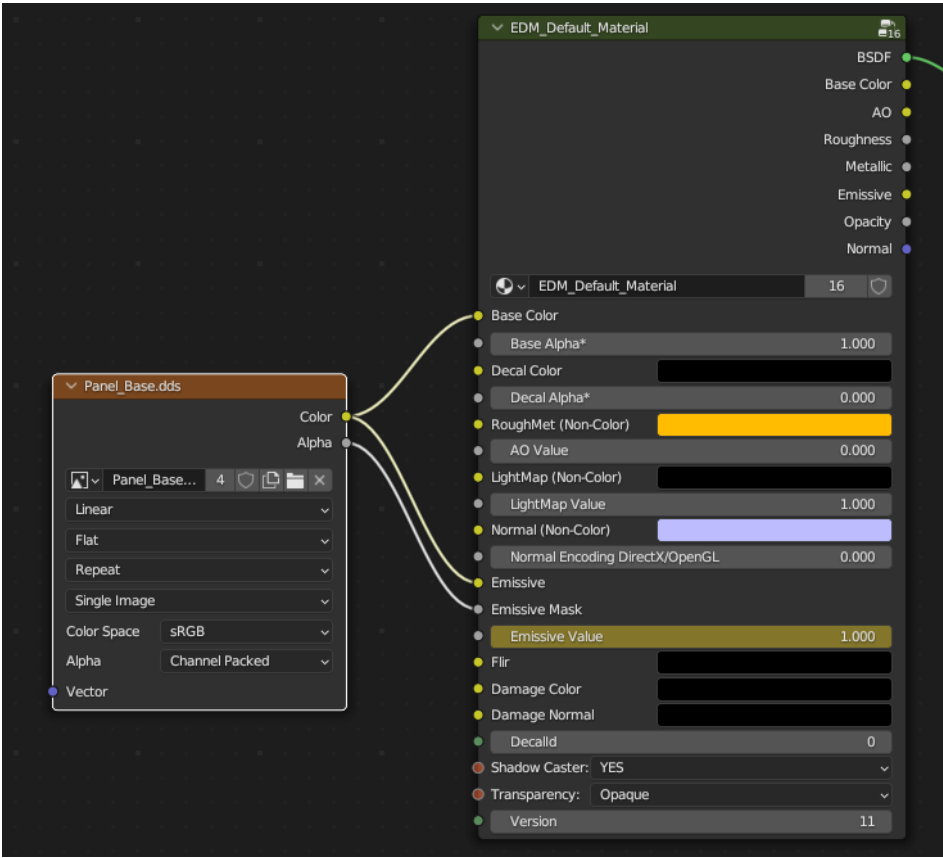
### Emission Materials (glow)

Emission is a property of Default material. It is turned on by changing colour of “Emissive” to some other than black. Emission value is controlled by Emissive Value. Animated value is allowed.



Two main glow options are used:

- 1) **Self Illumination** - default mode — weak lighting. For example, for cockpit elements backlighting. Glow texture usage mode — connect it to the **Emissive channel**. Alpha channel to **Emission mask**.

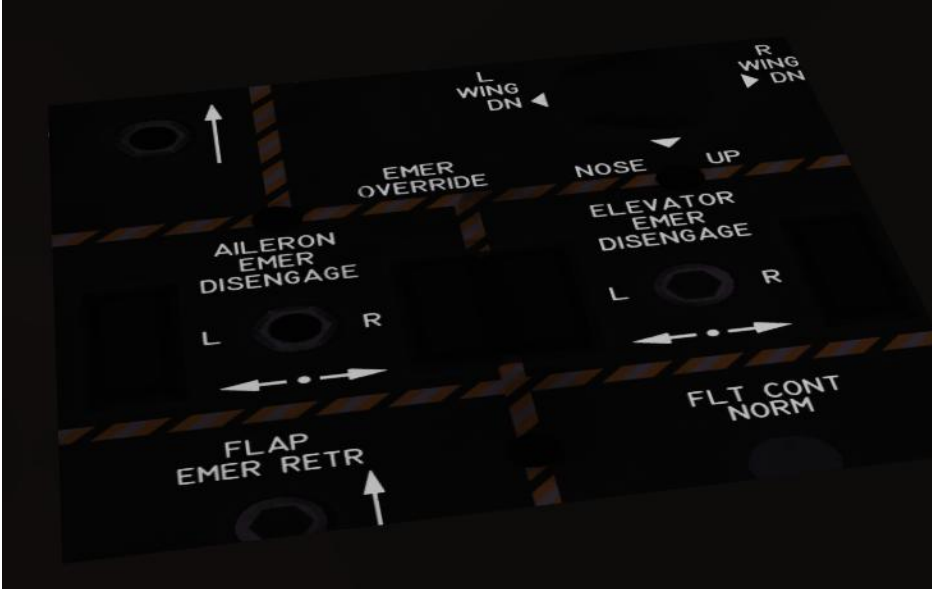


Glow colour comes from the texture, the mask (alpha channel) shows what glows and what doesn't.

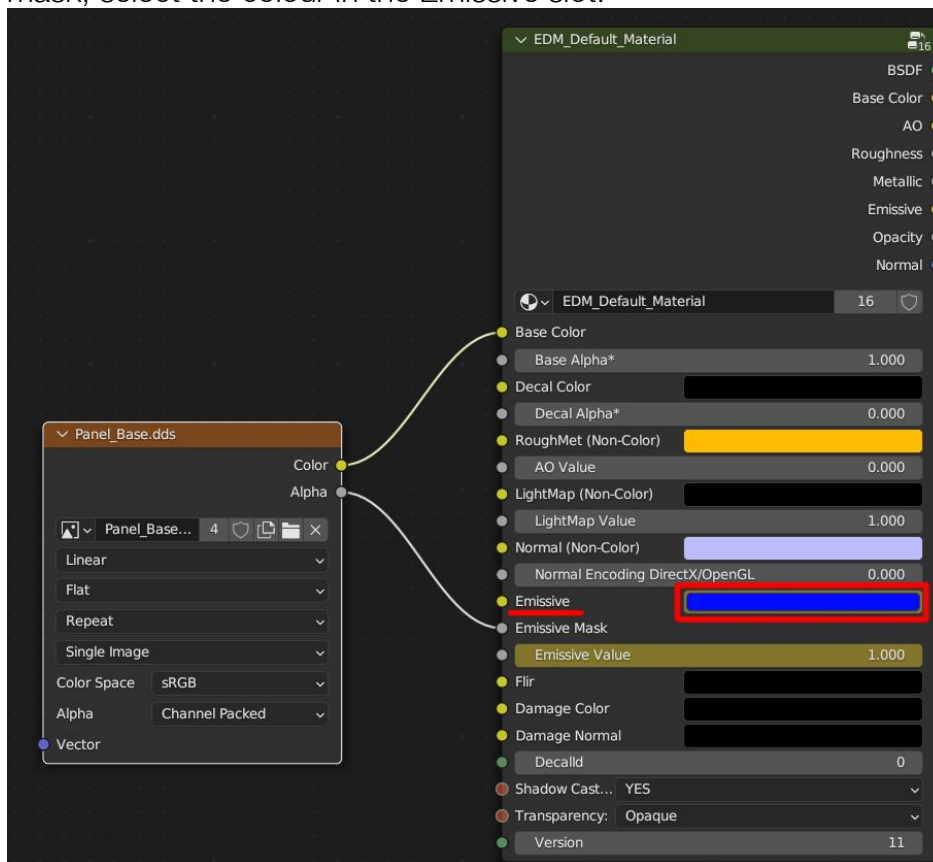




Result — cockpit panel inscriptions are backlit with the selected colour.



The mode of using the selected colour is to connect the alpha channel of the texture. In the Emission mask, select the colour in the Emissive slot.



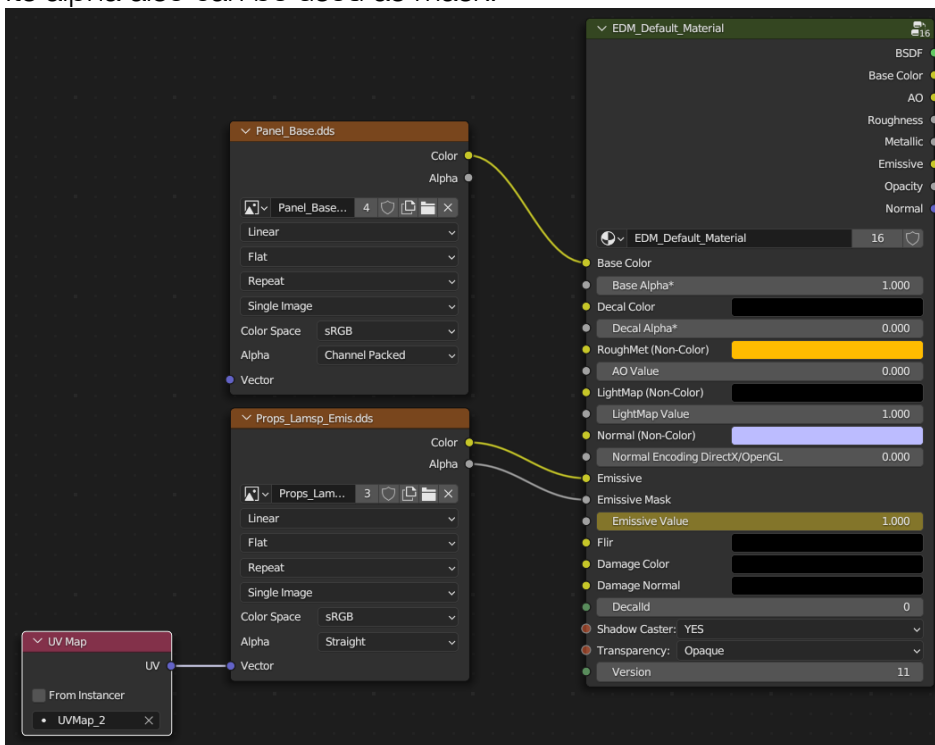




Result: the panel inscriptions is backlit by the selected colour.

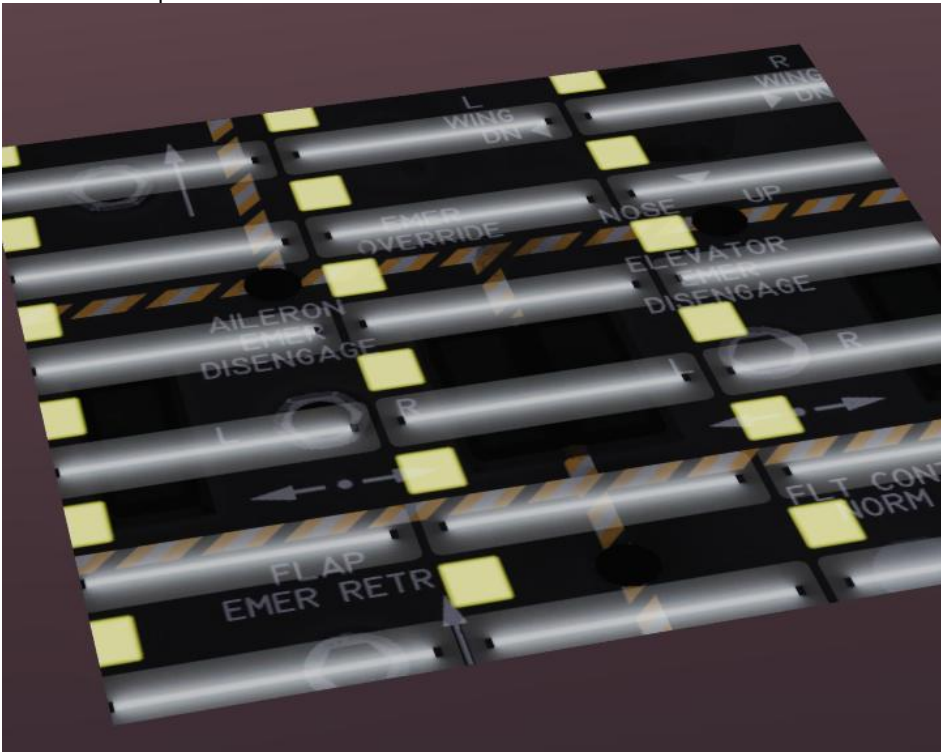


It is possible to set a different emission texture and give it a different UV. Its alpha also can be used as mask.



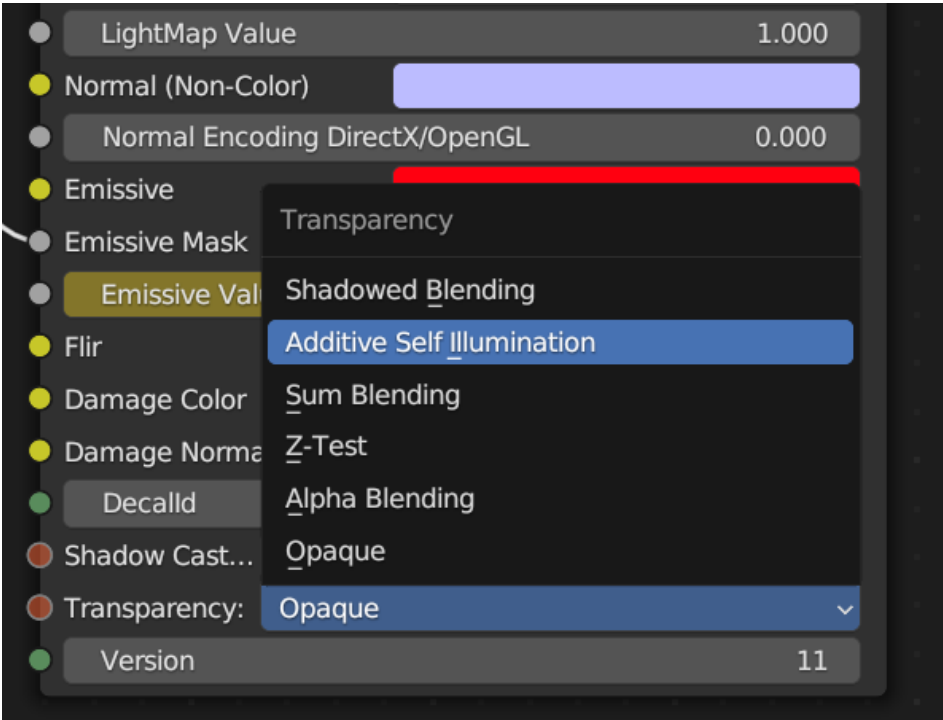


Result: the panel is illuminated with a different texture.



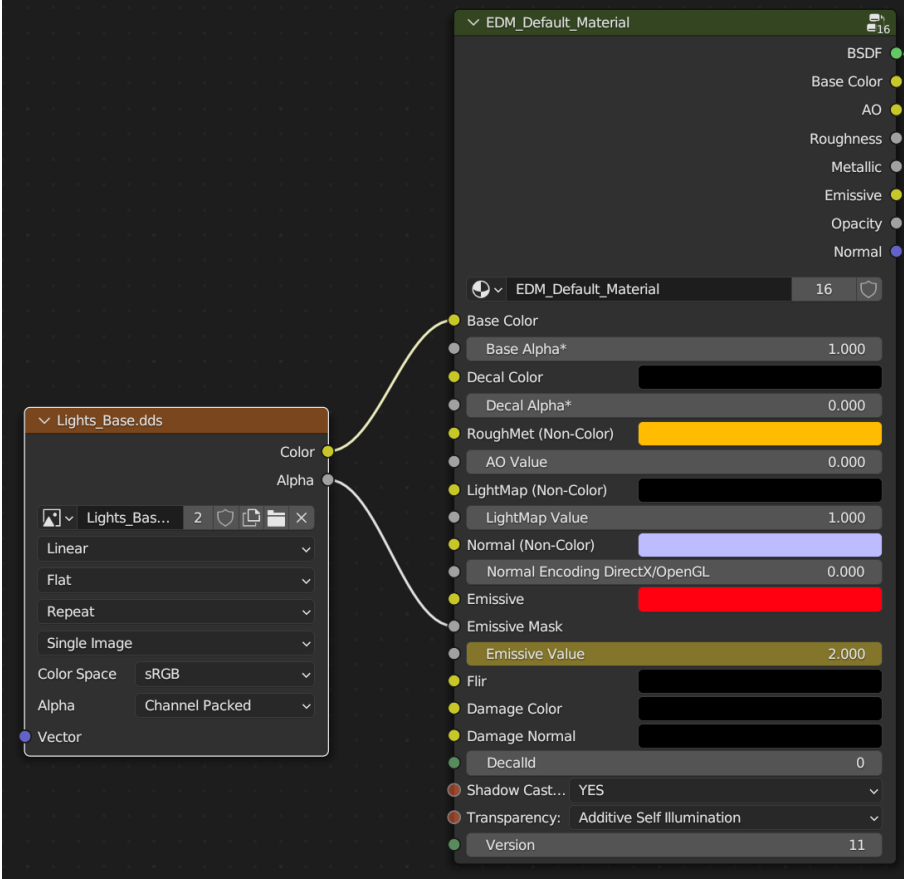
2) Additive Self Illumination - bright glow.

For bright (even blinding) indicators in cockpits, headlights/lamps of equipment, light spots on the deck. Switch to this mode by enabling **Additive Self Illumination** in transparency settings and setting the **Emission Value**.

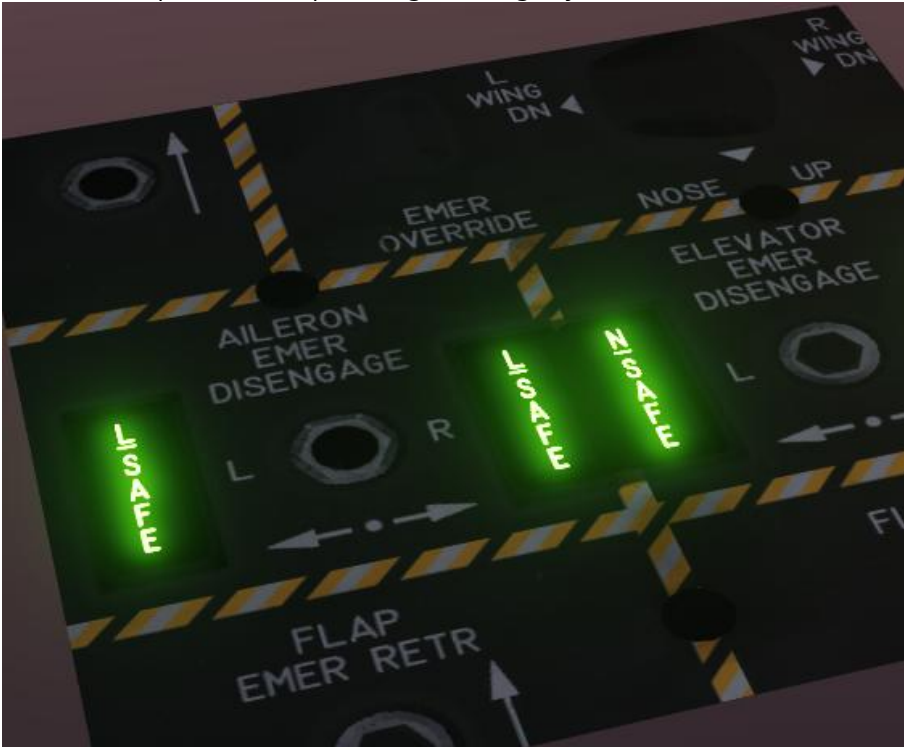




Uses texture from the Base Colour channel and the mask from Emissive Mask.

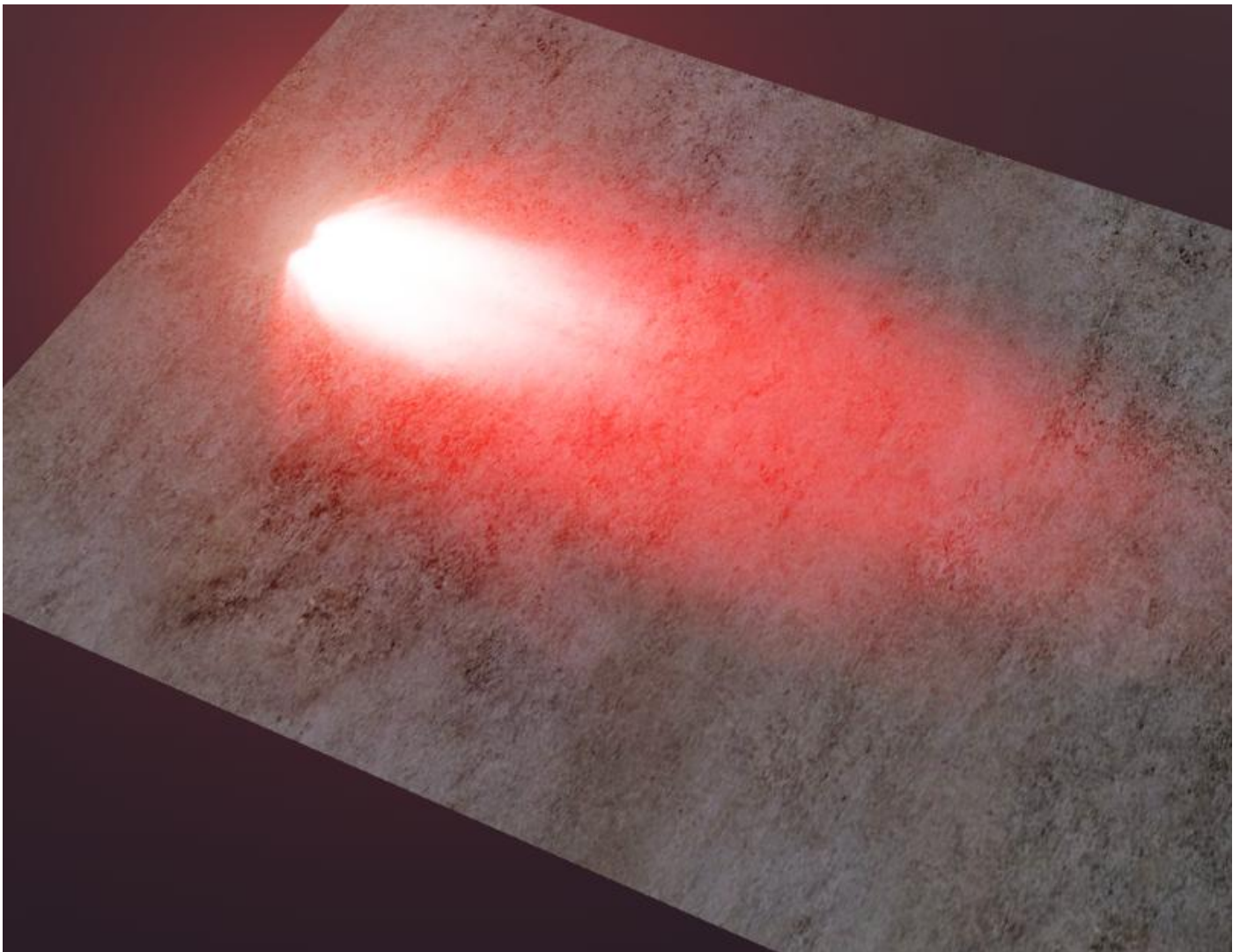


Result: the panel inscriptions glow brightly.



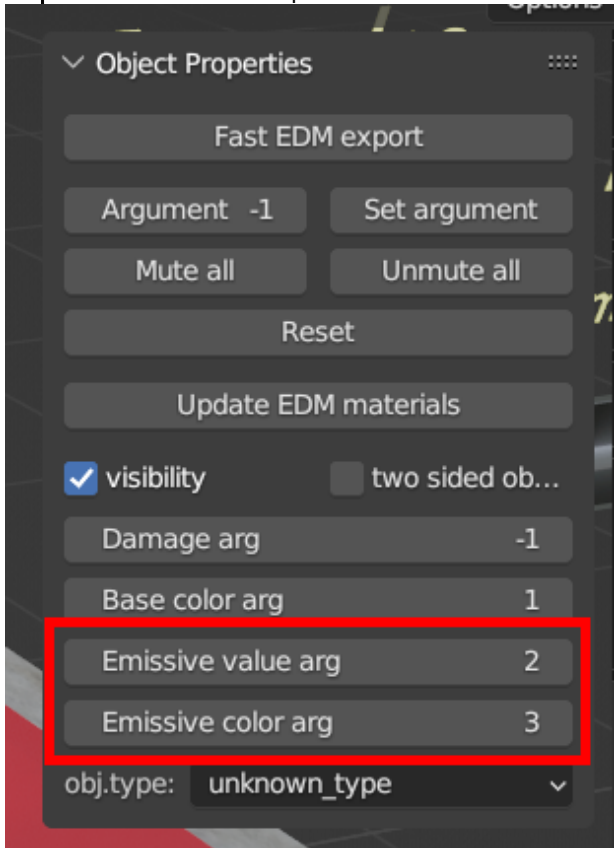


*\*\*Examples of Additive Self Illumination: Headlights/lamps and bright spot of light on the deck.*





The brightness and the selected colour can be animated — create animation keys in the material for the **Emissive Value** and the **Emissive colour (key "I")**, specify the argument number in the **"EDM Export"** tab of the N-panel.

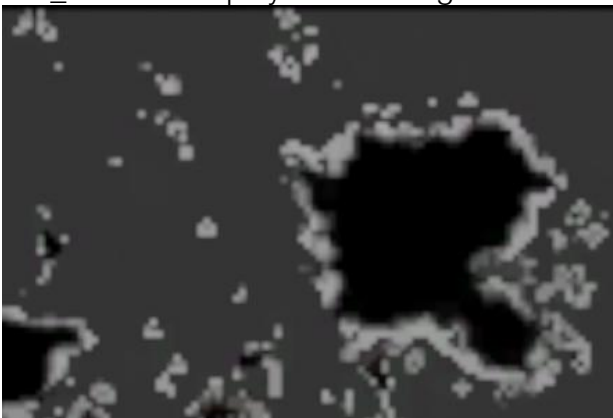


### Damage Material (Texture damage)

Damage material (texture damage) allows creating damage gradation for an object's material using textures. Used to show minor (bullet/shrapnel) and medium damage — holes.

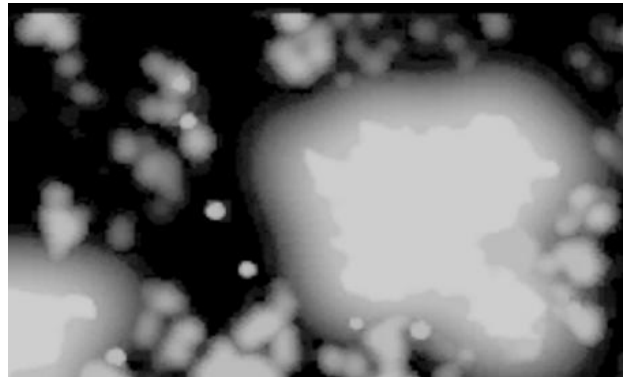
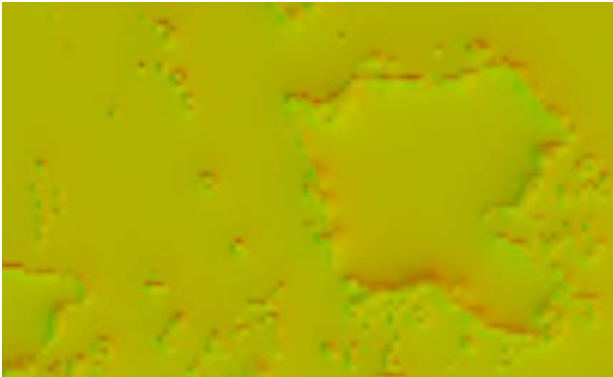
Textures used:

DM\_Base — displays the damage itself





DM\_Normal — normal maps for damage      DM\_map — mask containing damage animation frames.

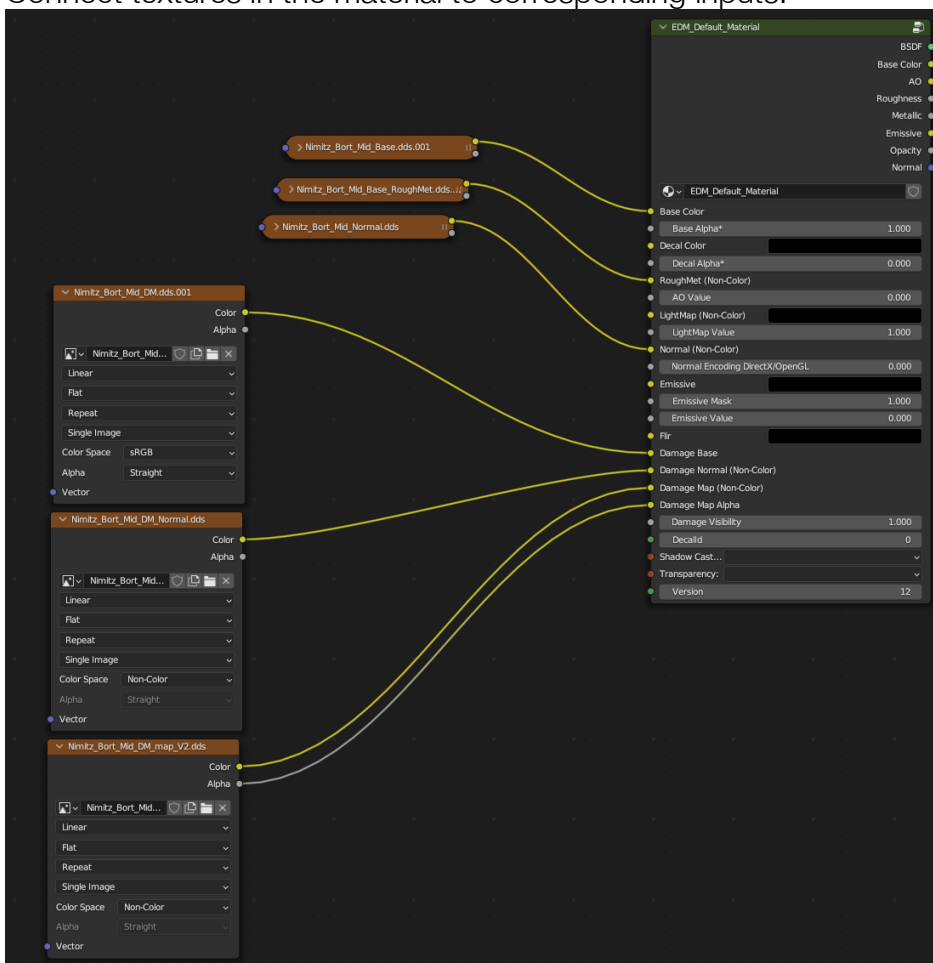


Anything brighter than  $<0.85$  on the mask will be drawn (rendered) as a transparent hole.

!!! Blender does not work with 3D Volume DDS textures previously used to store four damage gradation frames.

Now, BC3 DDS is used, with frames as RGB(1,2,3) layers and Alpha(4).

Connect textures in the material to corresponding inputs:





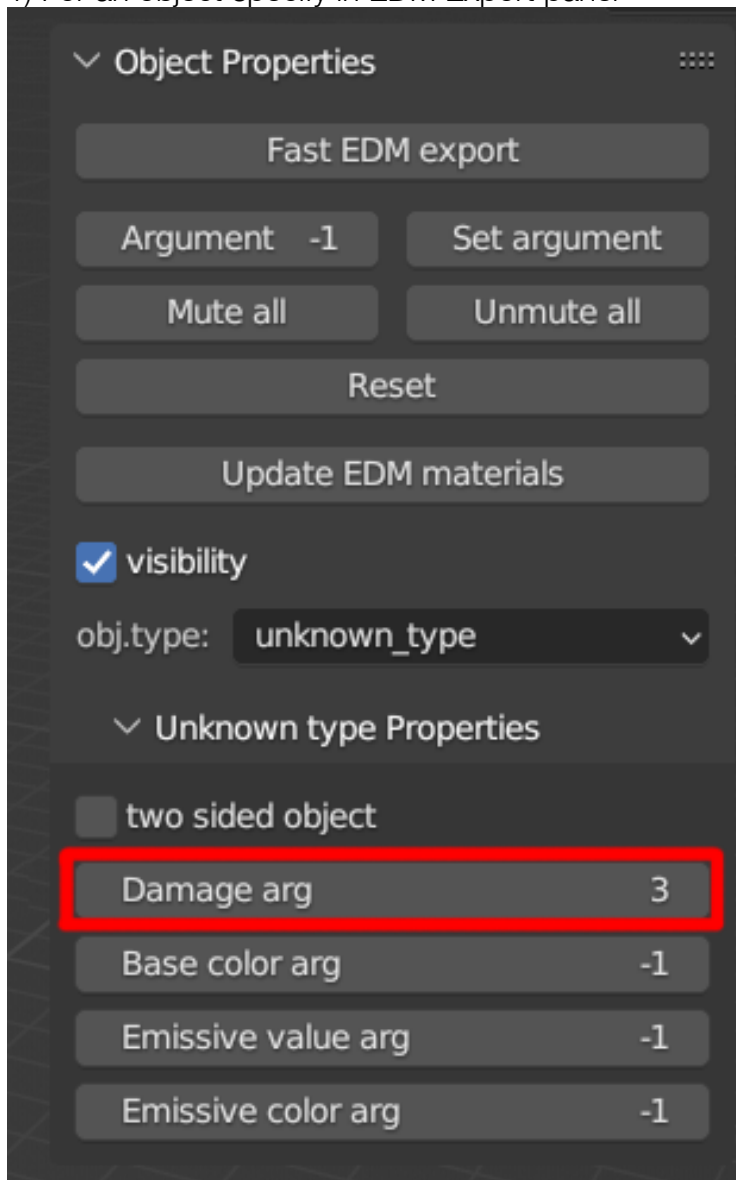


Damage Visibility is needed to show damage in the blender viewport.

*\*\*\* For viewing, connect the DM\_Map alpha, as it contains the fourth damage frame. Necessary for Blender viewing, does not affect export.*

You can set an argument either to an object (the whole thing will damage) or its parts (using vertex groups).

1) For an object specify in EDM Export panel

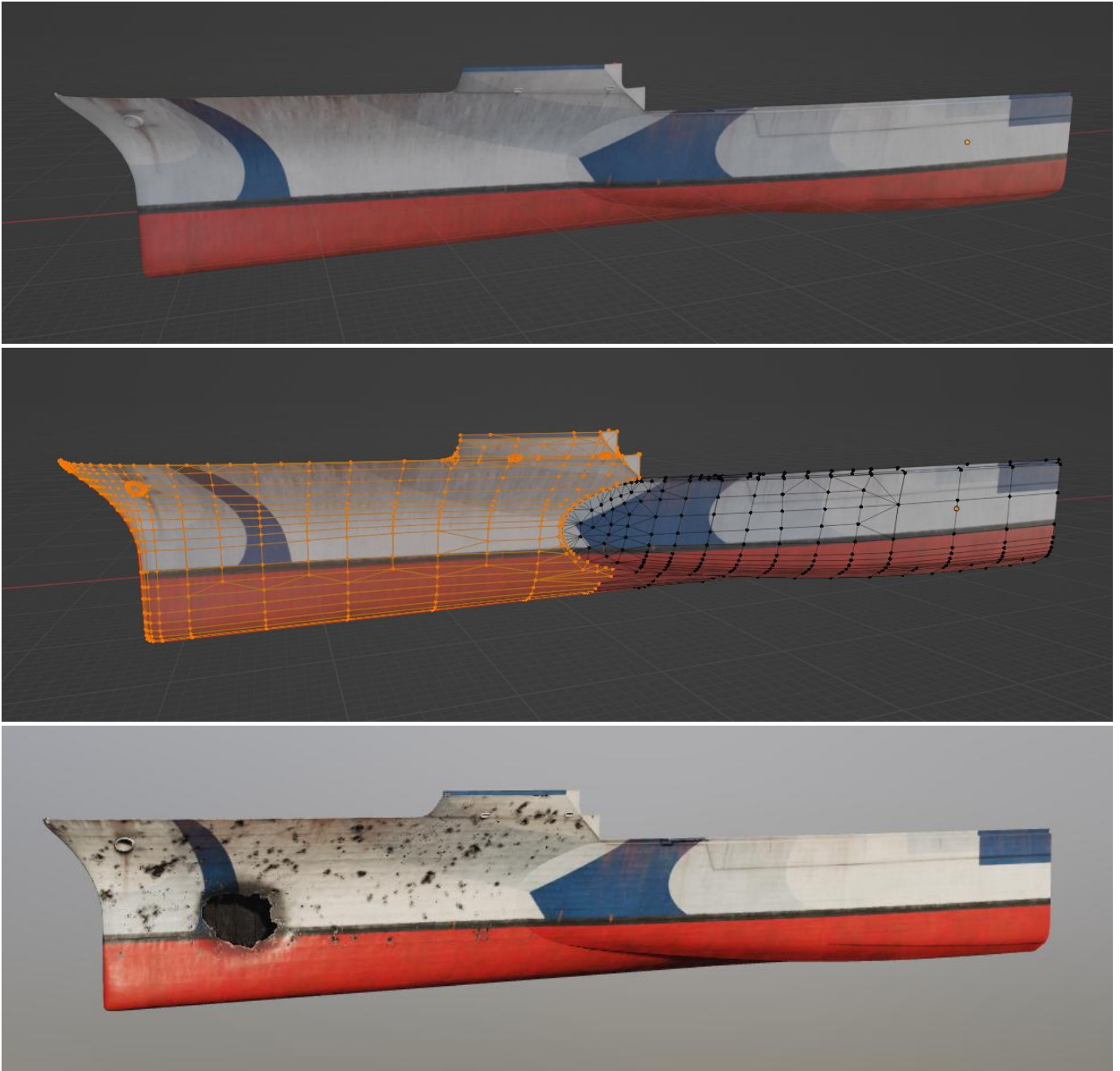


2) For a part of an object — for one object you can assign different damage arguments to different parts. They are indicated using vertex groups.

Create vertex groups in the object with the name: **DMG\_1**, where the number is the damage argument. Assign corresponding geometry parts to these groups. Create several such groups, one for each damage part.



**Example:** one part of the ship's side belongs to the group **DMG\_74**, another to **DMG\_75**. They will “damage” accordingly.



!!! Do not use both options. If vertex groups are used, do not specify Damage\_arg for the object.





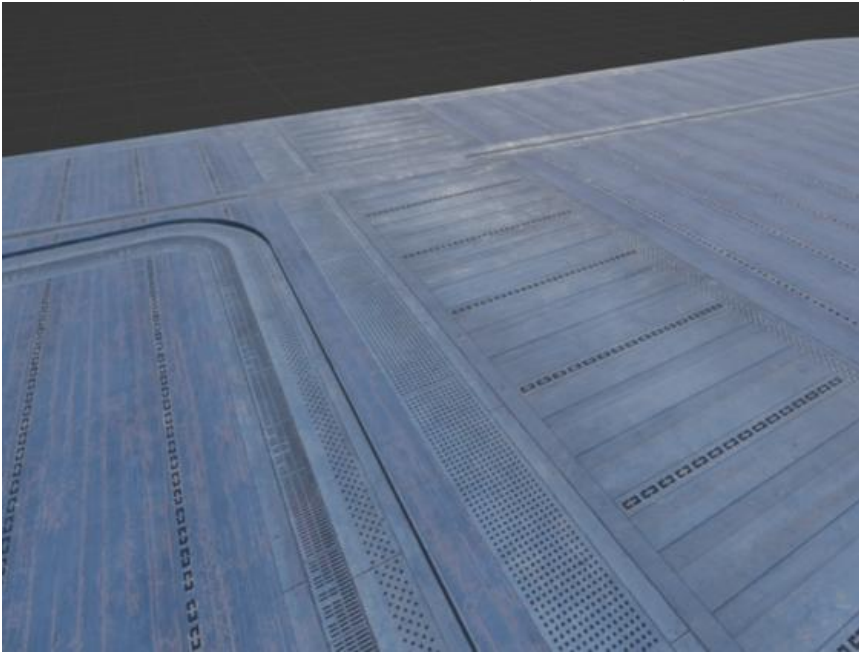
## Deck Material

A complex material for the aircraft carrier's deck. Allows you to create a realistic deck surface that will look good both from a distance and up close, in addition it has the ability to become wet in the rain.

Consists of two layers:

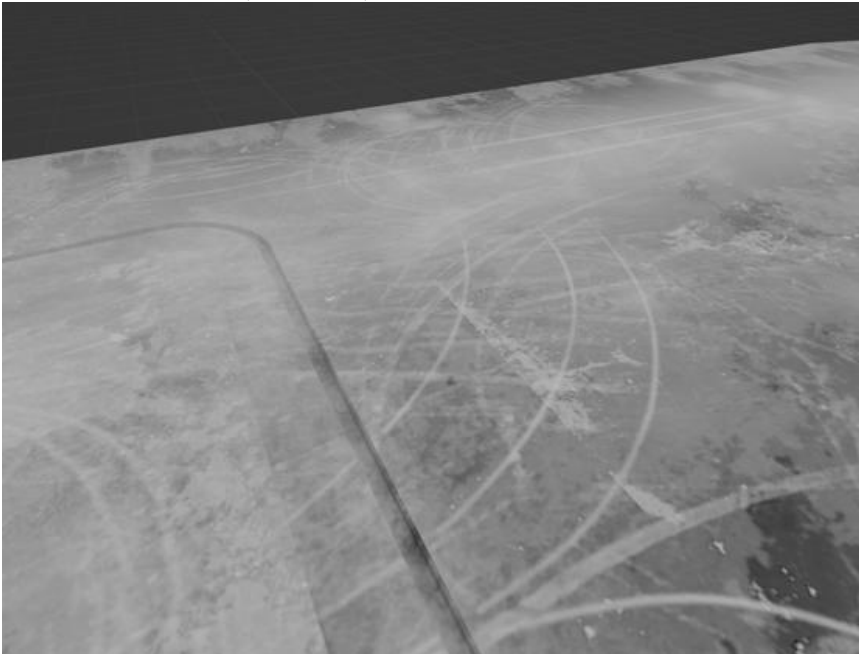
1) Tile (trim) — contains: **Base, RoughMet, Normal.**

It is the deck material itself without dirt, tire marks, etc.



2) Unique - consists of: **Base, RoughMet, Damage and WetMap.**

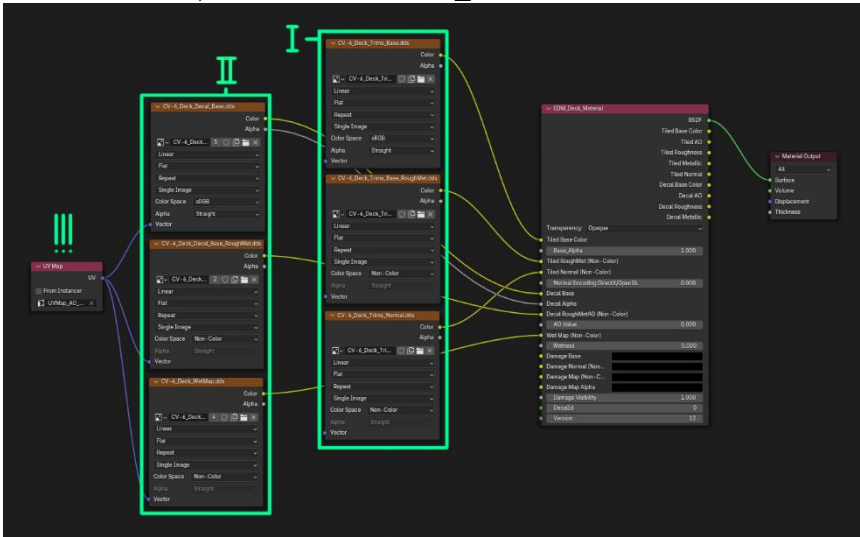
It is a decal of dirt, streaks, etc.



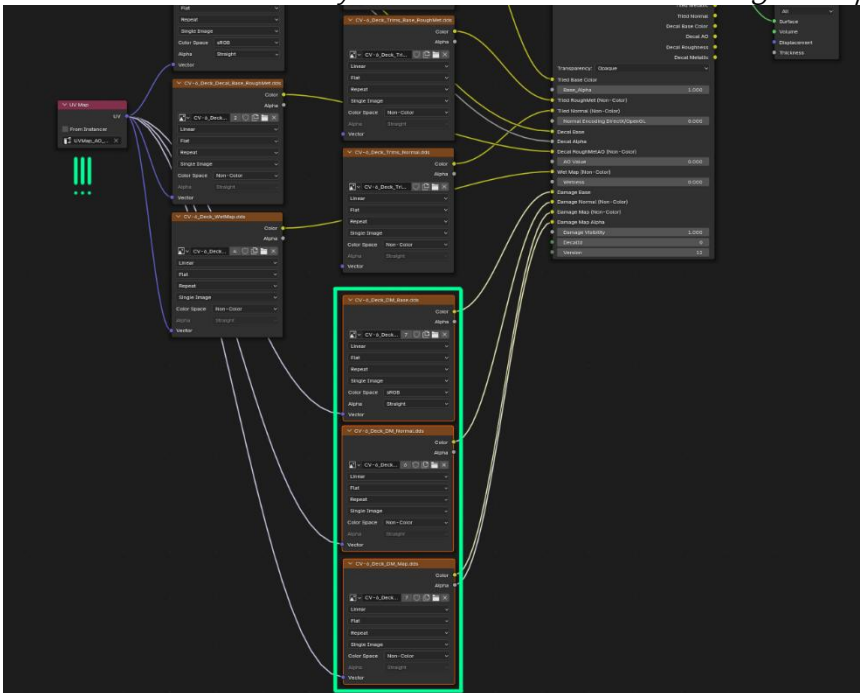
*\*These two layers are mixed using different UV.*



In the material, we create a Deck\_Material node and connect textures to it:

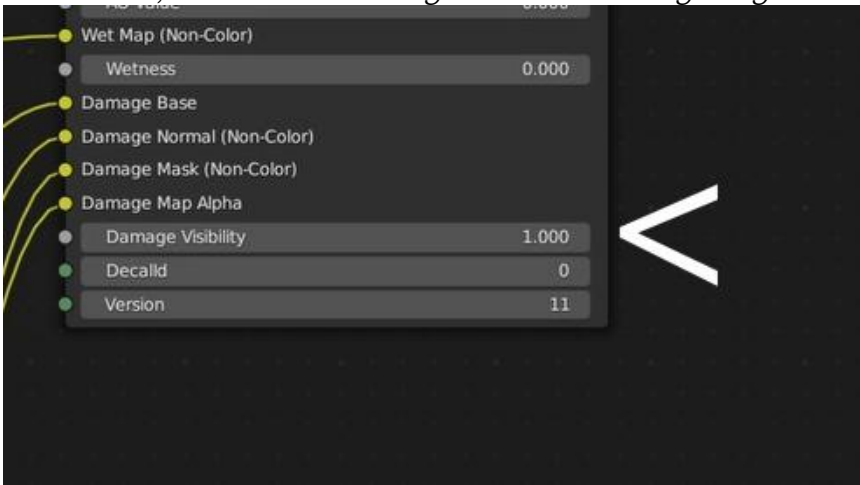


*\*I - textures of the first layer. II - the second. Do not forget to specify the UV Map.*





*\*If available, connect the damage texture. Not forgetting about UV.*



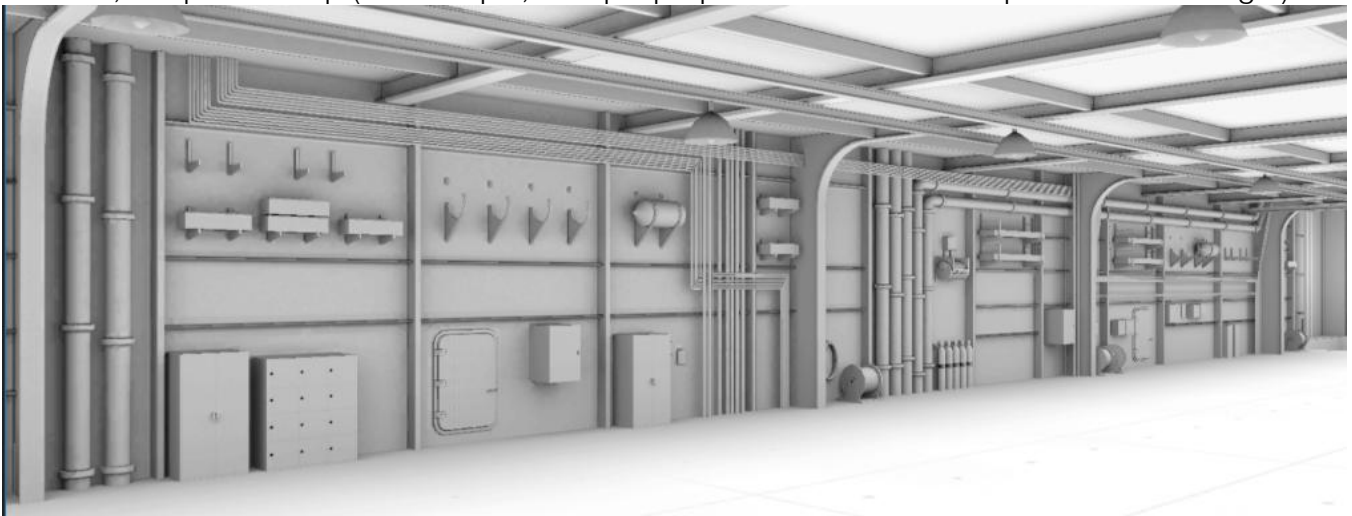
*\*Damage Visibility slider - allows you to check the gradation of damage in Blender viewport.*

Done, let's check.

!!! Blender does not work with 3D Volume DDS textures, which were previously used to store four frames of damage gradation, now BC3 DDS is used, inside which frames are RGB(1,2,3) and Alpha(4) layers.

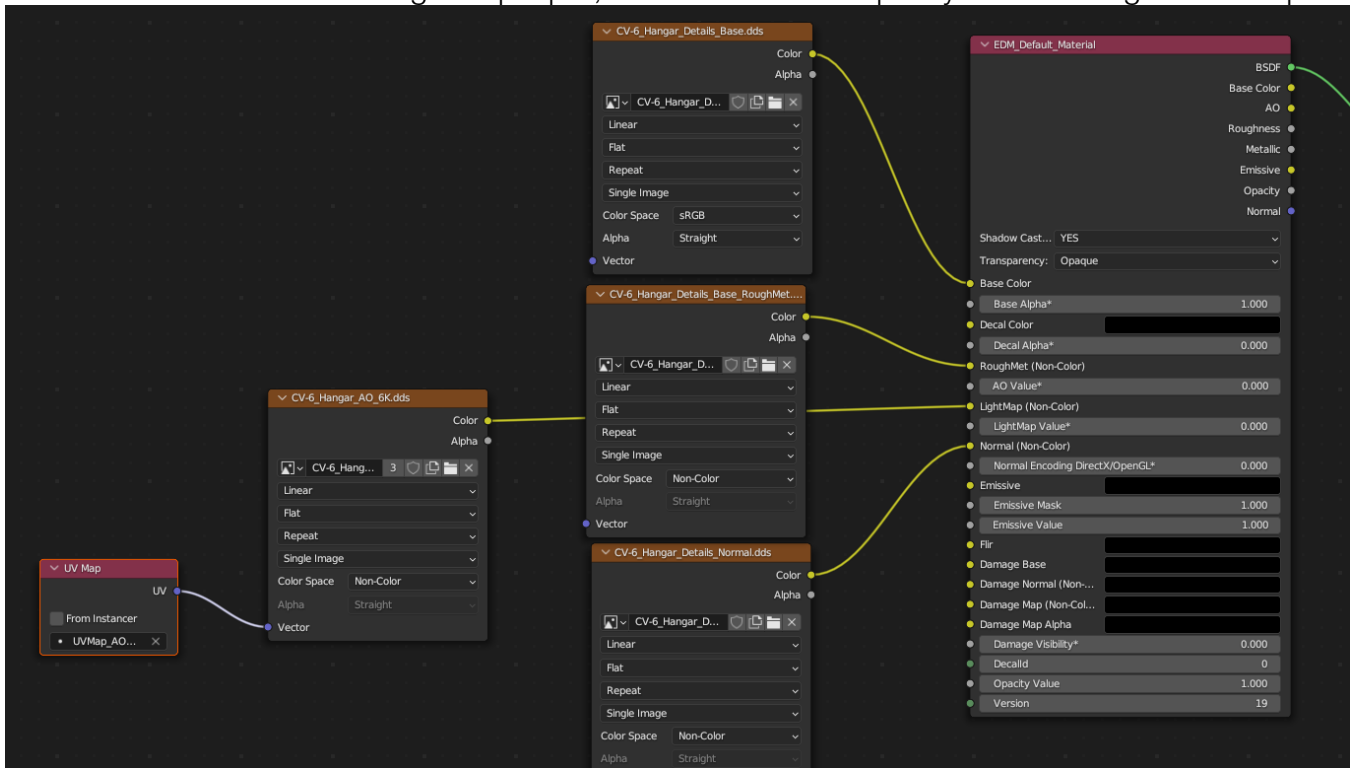
## LightMap (Unic AO)

This is a separate texture that is overlaid "on top" of the AO from **RoughMet** texture using the Darken method (the darkest pixel). It is used to create a unique AO for duplicate objects baked using a different, unique UV map (for example, multiple props located in different places in the hangar).





The connect Texture to the LightMap input, and make sure to specify the UV using the UVMap node.

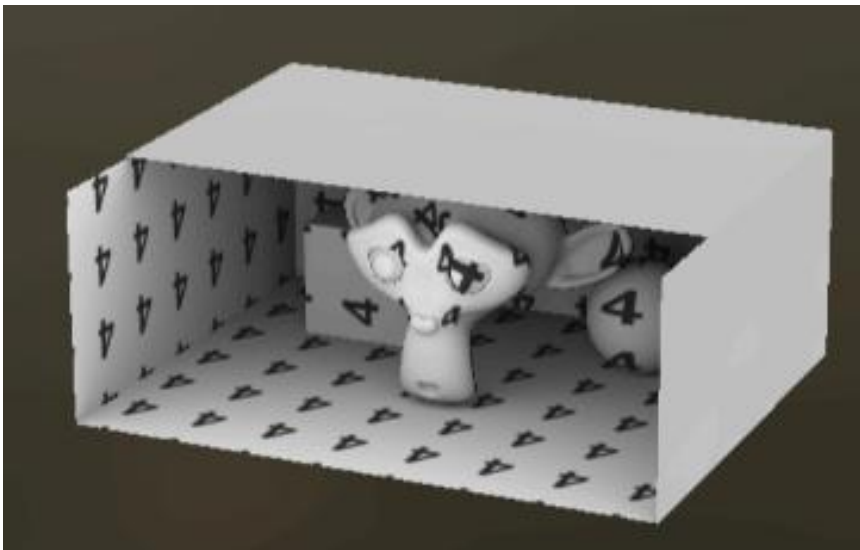
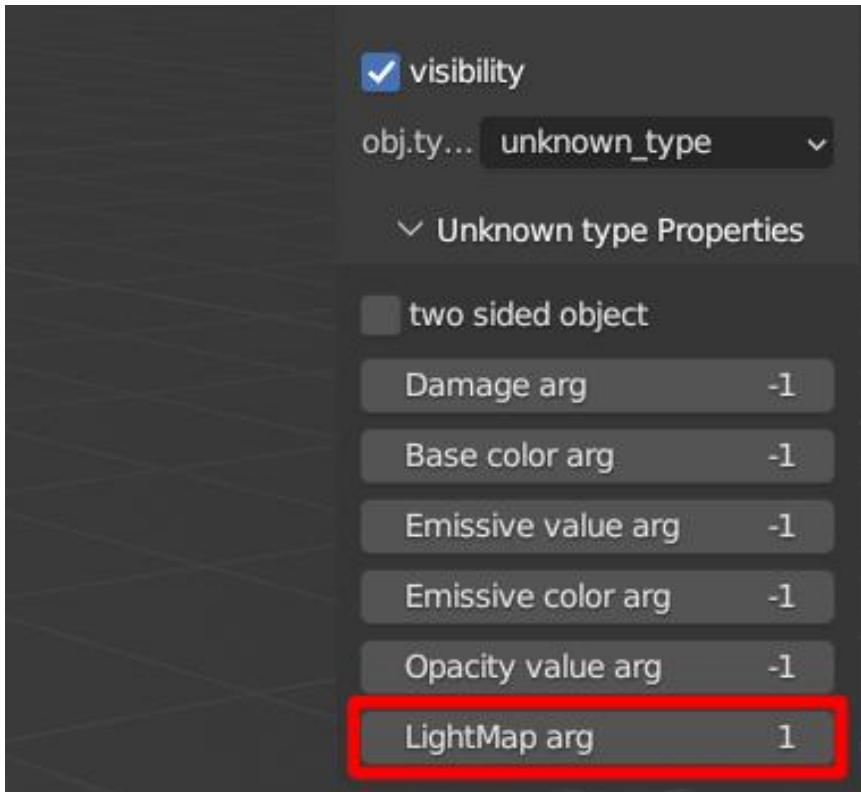


## LightMap Animation (Changing LightMap)

It is possible to change the lightmap by argument. For example, when closing the cockpit door, the lighting in it changes - it becomes darker. To do this, additional "frames" are added to the Lightmap texture in the RGB + Alpha layers (you can add 4 Lightmap frames that will change one after another by argument) and in the object settings in the EDM Export panel, specify "LightMap Arg" - by which it must change.

Mixing occurs by argument:

0.0 - no mixing, 0.25 - mixed with the first frame-(R), 0.5 - the second-(G), 0.75 - the third-(B), 1.0 - the fourth-(Alpha). Check result in the viewer.



\*\*\*Because the Lightmap will be mixed with the AO from the R-channel of RoughtMet using the "Darken" method - the darkest pixel - the AO from RoughtMet should not be darker than the LightMap.

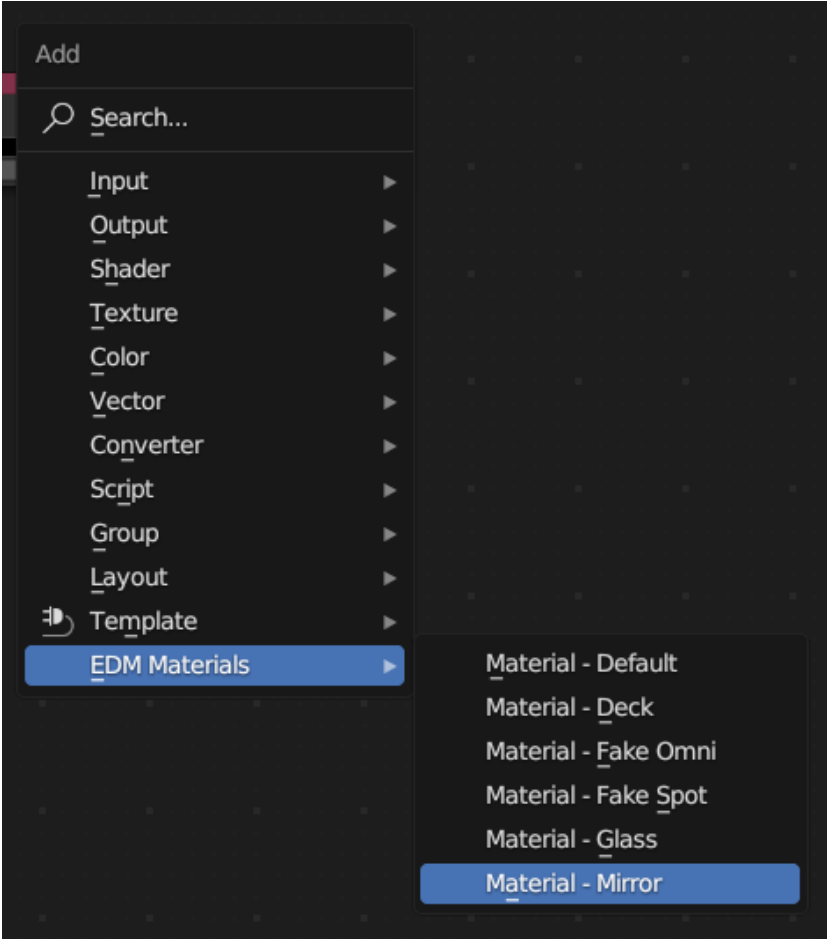
There are two options for using this technology:

- 1) AO from RoughtMet contains only Ambient Occlusion of objects with a small radius (without baked light), different frames of the Lightmap will be mix with it.
- 2) AO from RoughtMet is filled completely with white so as not to affect the baked light from the LightMap.



## Mirror in cockpit

Is set using the Mirror material in the editor.





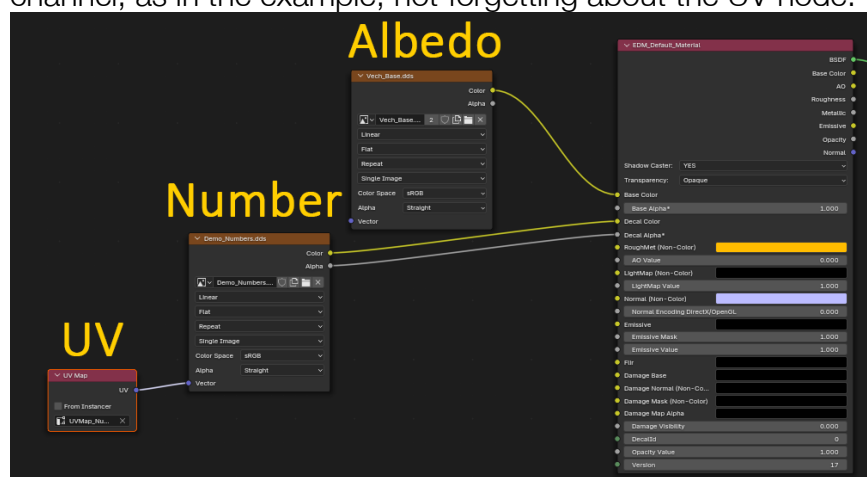


## Numbers

This is a tool for numbers on vehicles in the game, which works by moving UV. The principle is similar to "UV animation" But it is more optimized for this task. In this case, UV is moved only at the decal channel. Create a unique texture with numbers or other signs for it, duplicate a part of the geometry (where the number will be located) and "bulge" slightly forward, create a new UV for drawing the number, in which the UV island is installed on the first sign. (This can be an "empty" sign).



Create a new material for it. As an albedo, you need to use the same texture as the material underneath it, this is necessary for correct display. Connect the texture with numbers to the decal channel, as in the example, not forgetting about the UV node.

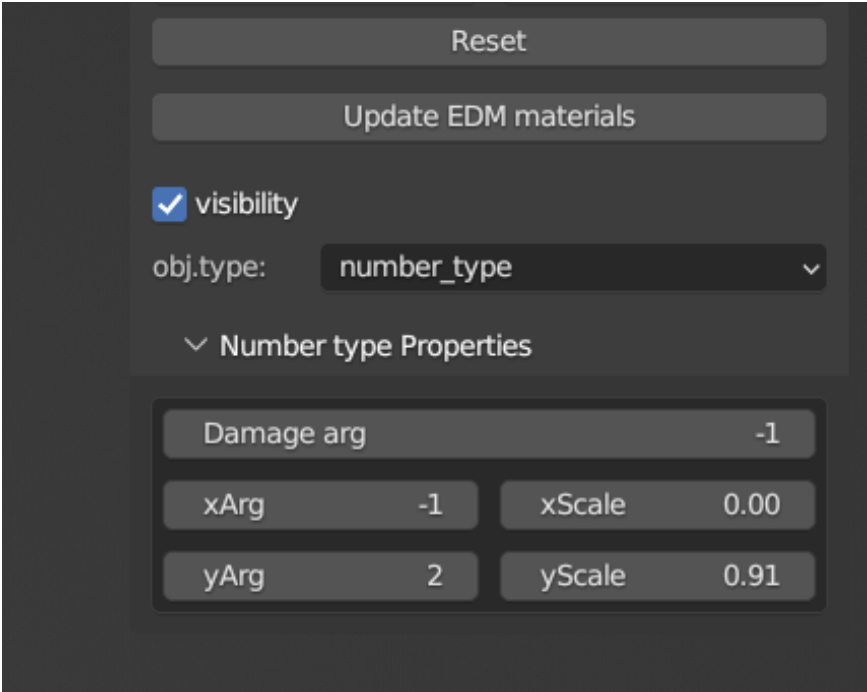


Select the object type "**number\_type**", in the opened chose the argument of the offset along the horizontal X axis - **xArg**, vertical Y - **yArg**.



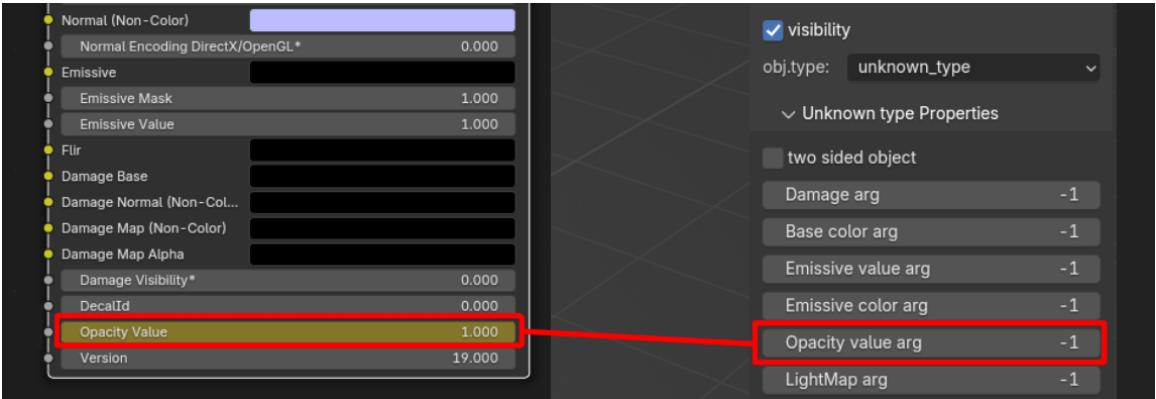
**Argument -1 — disabled.** And the step size (Scale) (from 0 to 1 along the length of the UV). That is the UV offset from the first to the last sign (for this example with 11 characters (10 digits + "empty") the Scale will be equal to 0.91 ( $1/11 \cdot 10$ )). Also, if necessary, chose the damage argument and add damage textures to the material.

Check in the viewer.



## Opacity Value Argument

Opacity Value — a tool for adjusting the transparency of materials. For example, it can be used for smooth appearance/disappearance or simply changing the transparency of objects in Blend mode. It's set using the animation of the "Opacity Value" in the material (l key) and chose the animation argument for the object in the EDM Export panel. In the Alpha Test mode, it adjusts the **"cutoff point 0.5"** up or down.

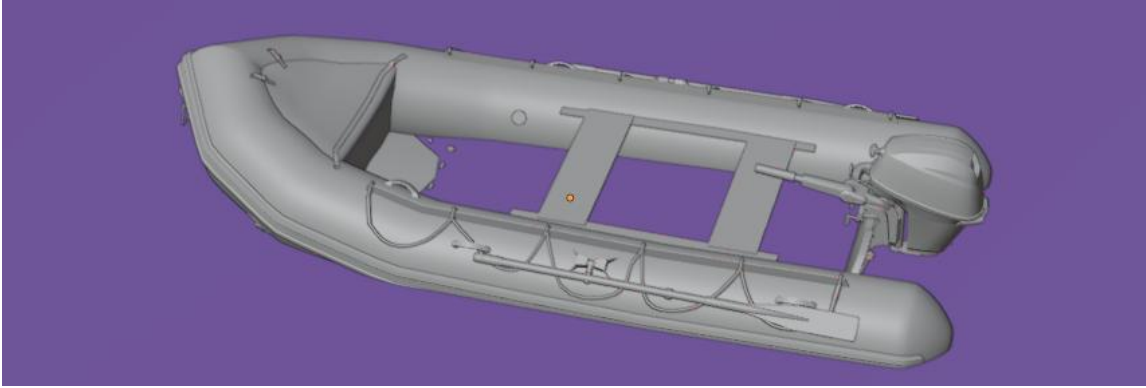




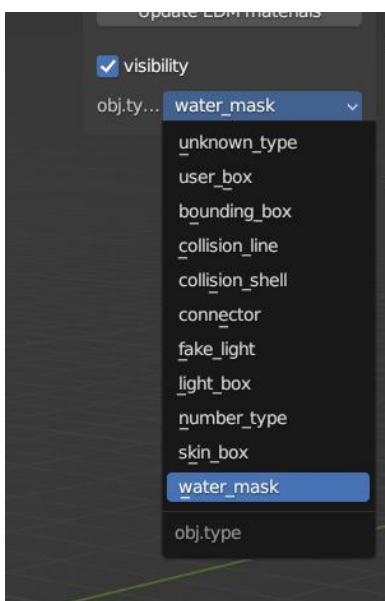
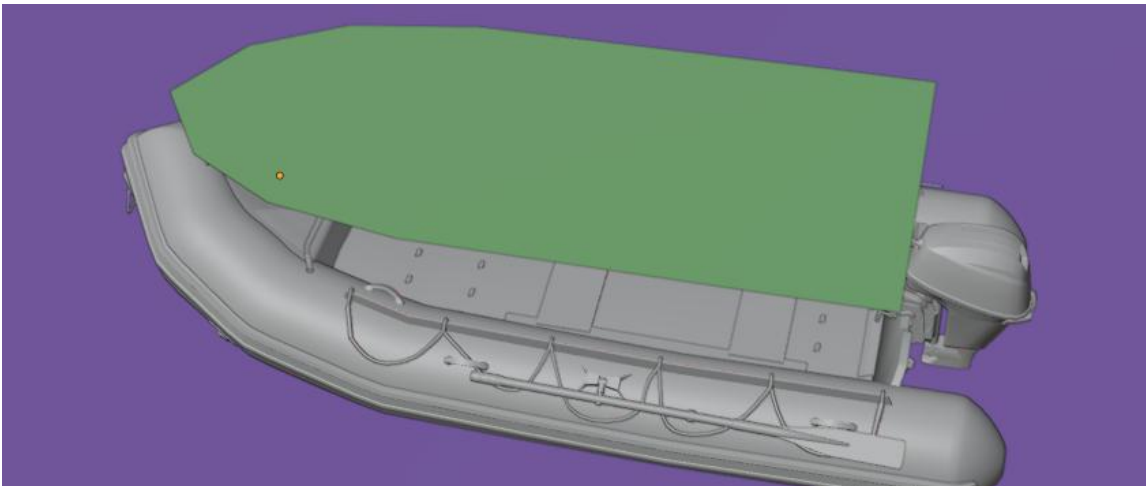


## Water Mask

In some cases, water may show through geometry. For example, in a boat.



To fix this, create a geometry (water cutoff), set it to the bottom of the boat and chose type — **Water Mask**.





as a result, the water should be cut off in the game (approximately as in the screenshot).

