Normal Map Seams

Many times after rendering out a normal map in Xnormal, applying it to the low res mesh and looking it in Cryengine you have normal map seams like in the picture below. In this tutorial we will explain what they are, why they happen and what to do to avoid them.

As common logic the low poly mesh that is being used to render the high poly box is exactly the same size as the high poly mesh, but xNormal uses the low poly mesh as its rendering cage and there lies the reason why these errors occur.

The way xNormal bakes the normal map, the rays that are cast are coming in a perpendicular direction from the low poly mesh, meaning that is only capturing what it displays in the low poly mesh. And as we mentioned before the low poly mesh is literally acting as a cage.

To help you visualize the process we have prepared the image below.

As you can see the pink lines represent the low poly faces while the green lines represent the high poly faces, so the low poly mesh is sitting exactly on top of your high poly box.

The yellow arrows represent the rays coming from the low poly mesh during rendering in xNormal.

In the image below you we displayed what actually got captured in the normal map and how it looks on the model.
There multiple options to solve this, you should choose which one is the most effective for the particular situation.

Using a custom render cage in xNormal

xNormal offers the option to use a custom cage that is created and then exported from the modeling package.

It is not the most practical and quickest solution since it requires a lot of time in tweaking the cage, especially on larger complex shapes to get the desired effect, so it is advisable to keep it as an option and not the rule.

To do that you need to duplicate your final low res object and scale it down so, in this case, the edges of the low res cube are right in the middle of the beveled edges on the high res cube.

For more complex assets you should use the "Push" modifier in 3dsMax.

With the push modifier you can specify a more exact setting which works best in most cases plus it can be easily undone or modified.
Scale down low res cube with the "Push" modifier so it's edges hit exactly at the midpoint of the bevels.

Low res cube has been scaled inwards so it's corners are hitting exactly the mid point of the bevel.
Now rays will hit it during rendering and create a correct map on the corner.
The second way to fix that is by adding a bit more of geometry by adding bevels to your low res asset where it's beveled on your high res asset. It will be more costly geometry wise, but it will give you a smoother and better silhouette.

This will also speed up the process compared to the option below, since the rays are being caught correctly due to the bevel offering extra geometry.
How to lay out UVs based on smoothing groups to get a clean normal map using xNormal

To get the best results when rendering out a normal map when using xNormal it is advisable to split the low poly asset into different smoothing groups and have for each smoothing group it's own UV island.

The benefits are that normal maps will render cleaner and in some cases will provide a greater re-usability.

Having a single smoothing group across the entire low poly asset and having all the UV's connected means that during rendering of the normal map xNormal has to conform the high poly details, meaning it has to compensate during the bake because the normal directions are often offset from the low poly compared to the high poly since it has less geometry to work with and the normals of the low poly are averaged.

Assigning several smoothing groups to it and splitting it up in accordance to it's own UV islands means that it is needing lesser compensation. It is important to keep some space between them. If they're too close it will generate some rendering issues.

The downside is that by splitting it to it's own UV island leads more expensive UV spacing. By doing this it can add greatly to the production time but the final result is better and cleaner since it's leading to lesser visual errors.

As a rule of thumb and to compensate for speed it's best to have one UV island per smoothing group.
UV’s are all connected

Surfaces are not rendered out flat and look puffed since it has to compensate for the averaged normals in the low res.
Here are the results after assigning multiple smoothing groups and splitting up the UV islands according to it.

- Shading errors in the low res bake make it look wrong and dirty.
- Each surface has its own shading group assigned.
- Each UV island is split up to its own shading group and have some space apart between them.
Surfaces are flat and bevels are looking correct. No rendering issues occur.

No Shading errors. All surfaces look clean and like they supposed to.