Preface:

There are many ways to do this, if you dislike AI, then skip that part and create the 3 base images you need by hand, this is just the simplest way I can find to do this.

Requirements:

Just Gimp.

https://www.gimp.org/

Created by : Aftertaste

Section 1 – Generating Your Base Images with Gemini

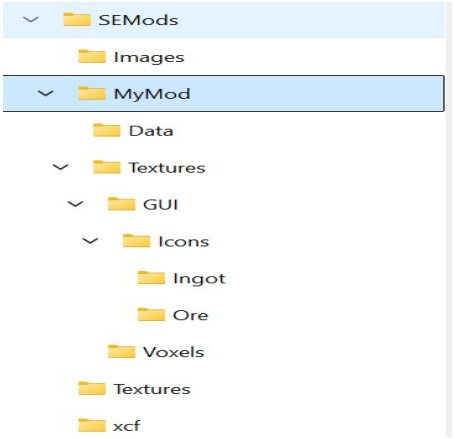
🎯 Goal

We're going to create two starting images:

1. A seamless 2048×2048 texture for your voxel's surface.
2. A PNG with the ore and bar icons.

These will serve as the artistic foundation for everything else we do in GIMP.

Setup for this project, create a Directory SEMods



---

1. Open Gemini

- Go to https://gemini.google.com.
- Sign in with your Google account.
- Start a new chat — Gemini opens a text box at the bottom where you type prompts.

---

2. Create the Voxel Texture

Type or paste the following prompt:

```
A seamless 2048×2048 image texture for Space Engineers voxel materials:
otherworldly red tinged ore surface with translucent radioactive spots.
High detail, flat lighting.
```

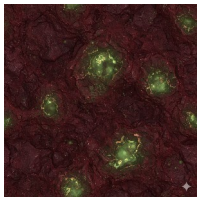
✅ Explanation

- *Seamless* means the edges will line up when the game tiles the texture.
- *2048×2048* is the standard texture size for voxels.
- *Flat lighting* prevents baked-in shadows that would look wrong under game lights.

When Gemini finishes:

1. Right-click the image → **Save image as…**
2. Save it to

   `SEMods/Images/Thorite.png`



---

3. Create the Ore and Bar Icons

Type this second prompt:

```
A PNG with a white background, showing two icons for Space Engineers:
one of an otherworldly red tinged ore (radioactive glowing cluster),
and one of a metal bar made from that ore.
Each centered in its own 512×512 area.
Realistic style, clear transparency.
```

✅ Explanation

- White background lets you control the transparency for the game's inventory UI.
- 512×512 gives enough detail before we scale them down later.

When Gemini finishes:

1. Right-click each image or the combined image → **Save image as…**
2. Save it to

   `SEMods/Images/Thorite_icons.png`

---

4. Verify Your Images

Before opening GIMP, double-check:

File	What to Look For
`Thorite.png`	Looks tileable — edges line up if repeated. No visible shadows.
`Thorite_icons.png`	White background two clear shapes.

If these look correct, you're ready to move on to **Section 2 – Preparing the Texture in GIMP**.

SECTION 2. PREPARING THE TEXTURE IN GIMP

Goal of this section:
We're going to load the base texture in GIMP, set up our working layers, and build all the maps Space Engineers needs.

4.1. Open the base texture in GIMP

1. Open GIMP.
2. File → Open…
3. Open: SEMods/Images/`Thorite.png`

4.2. Scale it to match Space Engineers

Space Engineers voxel textures are typically 2048x2048.

In GIMP:

- Image → Scale Image…
- Width: 2048
- Height: 2048
- Interpolation: Cubic
- Click Scale.

4.3. Save the project file

File → Save As…
Save as:
SEMods/xcf/`Thorite`_Working.xcf

From this point forward, always save changes to this .xcf file. This is your master.

SECTION 5. JUMPING INTO CREATING THE IMAGES

5.1. Create a working copy of the base color

We do not want to destroy the original layer.

Steps:

1. Right-click your current layer → “New from Visible”
2. Rename the new layer: ColorRGB
3. Hide the original imported layer by turning off its eye icon
4. Keep ColorRGB visible and selected

ColorRGB is the main color of your voxel material. This will be used for two of the final textures: _AxisXZ_cm and _AxisY_cm.

SECTION 6. MAKING THE SUPPORT MAPS

Now we generate the other maps the game uses.
We will make:

- MetalnessMask (for reflectivity)
- HeightMap (for depth)
- NormalMap (for lighting bumps)
- GlossMask (for shine tightness)
- CavityAO (for shadow depth)
- DistanceLod (for far-away fallback)

We’ll do them one at a time.

6.1. MetalnessMask

Purpose:
This tells the game what parts of the texture act like shiny metal vs dull rock. In Space Engineers:

- White = metallic / reflective
- Black = matte / rock
- Gray = in between

For ore, you usually want mostly dark gray maybe slightly brighter in highly polished areas.

How to make it:

1. Duplicate the ColorRGB layer.
    - Right-click ColorRGB → Duplicate Layer.
    - Rename the duplicate: MetalnessMask.
2. With MetalnessMask selected:
    - Colors → Desaturate → Desaturate…
    - Mode: Luminance → OK.
3. Now adjust levels to push it darker overall:
    - Colors → Levels…
    - Input black: ~100
    - Input white: ~130
    - OK.

This gives us a mostly dark grayscale.

Later: this will become the alpha channel for the _cm texture.

You can experiment, different images will produce different results.

6.2. HeightMap

Purpose:
HeightMap is a grayscale map where bright means “raised” and dark means “recessed.” We use this to build depth and shading.

How to build it from the same base (no drawing):

1. Duplicate ColorRGB again.
    - Right-click ColorRGB → Duplicate Layer.
    - Rename it: HeightMap.
2. Hide everything except HeightMap (turn off all other eyes).
3. With HeightMap active:
    - Colors → Desaturate → Desaturate…
    - Mode: Luminance → OK.
4. Slight blur to smooth noisy pixels:
    - Filters → Blur → Gaussian Blur…
    - Size X: 1.5
    - Size Y: 1.5
    - Filter: Auto (FIR or IIR is fine)
    - Abyss policy: Clamp
    - OK.
5. Increase local depth (but keep it gentle, ice should be smooth):
    - Colors → Levels…
    - Input black: 60
    - Input white: 180
    - OK.

Now HeightMap is a clean grayscale that gently shows pits and ridges.

6.3. NormalMap

Purpose:
The NormalMap is what tells the game “pretend there are bumps here so light reacts.” This is what makes the material look 3D in-game without actually changing the voxel geometry.

We generate this automatically from HeightMap. You do not draw this by hand.

Steps:

1. Make sure HeightMap is visible and active, then copy the layer and name it NormalMap.
2. Filters → Generic → Normal Map…
3. Settings:
    - Scale: default (this controls how strong the bumps are)
    - Tileable: Just like main image is
    - Invert Y: toggle this if in preview the pits look like they’re popping out instead of going in.
4. Click OK.
5. Hide HeightMap, keep NormalMap visible.

Check direction:
To verify the normal is facing the right way:

- Put NormalMap above ColorRGB.
- Change NormalMap blend mode to “Overlay” or “Soft Light.”
- If cracks look sunken → correct.
- If cracks look like they’re raised and bulging → redo Normal Map with “Invert Y” enabled.
- After checking, set NormalMap blend mode back to Normal and hide ColorRGB again.

NormalMap will later become the RGB of the _ng texture.

6.4. GlossMask

Purpose:
GlossMask controls how sharp (smooth) or dull (rough) the reflections are.

- White = smooth, wet, tight highlight
- Dark = rough, frosty, scattered highlight

For metal ore, you usually want mostly light gray. Some patches can be a bit darker.

Steps:

1. Duplicate NormalMap.
    - Right-click NormalMap → Duplicate Layer.
    - Rename to GlossMask.
2. Desaturate it:
    - Colors → Desaturate → Desaturate…
    - Mode: Luminance → OK.
3. Adjust Levels to make it mostly light gray:
    - Colors → Levels…
    - Input black: ~100
    - Input white: ~220
    - OK.
4. Blur very slightly:
    - Filters → Blur → Gaussian Blur…
    - Size X: 1.0
    - Size Y: 1.0
    - OK.

Later: we will paste this grayscale into the alpha channel of the normal map export.
This tells the game “this pixel is grey-ish metal ore with some rock, or this pixel is fully the ore.”

6.5. CavityAO

Purpose:
CavityAO becomes the _add texture. It darkens cracks and gives the voxel shadowed depth so it doesn’t look like flat wallpaper.

Steps:

1. Duplicate HeightMap.
    - Rename the duplicate: CavityAO.
2. Make only CavityAO visible.
3. If the deep areas look bright instead of dark:
    - Colors → Invert.
4. Smooth it so it’s not harsh/blotchy:
    - Filters → Blur → Gaussian Blur…
    - Size X: 5.0
    - Size Y: 5.0
    - OK.
5. Increase contrast just a little:
    - Colors → Levels…
    - Input black: ~70
    - Input white: ~170
    - OK.

This should give you a mostly mid-gray image, with darker creases in cracks. That’s perfect.
Later we’ll export this directly as _add.

6.6. DistanceLod

Purpose:
DistanceLod becomes _distance_cm. This is used by the game when you’re far away so the voxel doesn’t shimmer or flicker.

Steps:

1. Duplicate ColorRGB.
    - Rename it: DistanceLod.
2. Make only DistanceLod visible.
3. Blur it heavily:
    - Filters → Blur → Gaussian Blur…
    - Size X: 20
    - Size Y: 20
    - OK.
4. Fade the contrast:
    - Colors → Brightness-Contrast…
    - Brightness: +5 to +10
    - Contrast: -20 to -30
    - OK.

This should look like a very low-detail, soft-colored version of the texture.
That’s correct. That’s what the engine wants at distance.

SECTION 7. EXPORTING EACH MAP FROM GIMP

Important rule:
When exporting a map, ONLY that export layer should have its eye icon on. Turn off every other eye. Whatever you see on canvas is what you'll save.

We will export to PNG first. PNG is easier to work with. We'll convert to DDS in the next section.

We are going to export 4 voxel textures:

- _cm
- _ng
- _add
- _distance

We will also export 3 UI textures:

- thumbnail
- ore icon
- ingot icon

7.1. Export _cm (color + metalness alpha)

The _cm file needs:

- RGB = ColorRGB
- Alpha = MetalnessMask

This file defines:

- visible color (RGB)
- metalness / reflectivity (alpha)

Steps:

A. Build Export_cm

1. Duplicate ColorRGB.
    - Rename it: Export_cm.
2. Make sure Export_cm is visible.
3. Hide everything else.

B. Add metalness as alpha
This is the slightly tricky part.

Older GIMP method tries to paste into “Alpha,” but that can overwrite RGB. The safe approach is:

1. Copy MetalnessMask:
    - Show MetalnessMask.
    - Click MetalnessMask in Layers.
    - Ctrl + A, then Ctrl + C.
2. Go back to Export_cm.
    - Make Export_cm visible and active.
3. Open the Channels panel:
    - Windows → Dockable Dialogs → Channels
4. Right-click in the Channels panel → “New Channel…”
    - This creates a new channel at the bottom (this will become the alpha we export).
5. Select that new channel.
6. Press Ctrl + V to paste.
7. Anchor (Ctrl + H).

Now Export_cm has:

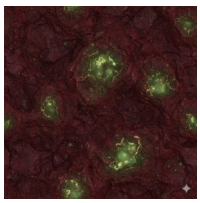
- RGB (its own color)
- A channel containing the MetalnessMask gray

C. Save it

1. Turn off all eyes except Export_cm.
2. File → Export As…
   Save as:
   SEMods/Textures/Thorite_AxisXZ_cm.png
   and also
   SEMods/Textures/Thorite_AxisY_cm.png
   (they can be the same for now)

In-game note:

- AxisXZ_cm is used for top/bottom faces of voxels.
- AxisY_cm is used for vertical faces.
  You can make them different later to add vertical striations.



7.2. Export _ng (normal + gloss alpha)

The _ng file needs:

- RGB = NormalMap (the purple/blue normal map)
- Alpha = GlossMask (the smoothness/roughness map)

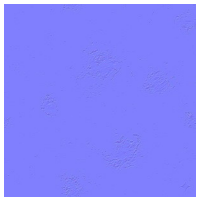
Steps:

1. Duplicate NormalMap.
    - Rename it: Export_ng.
2. Make Export_ng visible and active. Hide all other layers.
3. Copy GlossMask:
    - Show GlossMask, select it, Ctrl + A, Ctrl + C.
4. Go back to Export_ng.
5. Channels panel → Right-click → New Channel…
    - This creates a new channel slot for alpha.
6. Make that new channel active.
7. Ctrl + V to paste the GlossMask into that new channel.
8. Ctrl + H to anchor.

Now Export_ng has:

- RGB = lighting bump directions
- Alpha = how shiny each pixel is

9. Export:
   File → Export As…
   SEMods/Textures/Thorite_AxisXZ_ng.png



7.3. Export _add (cavity / AO depth)

The _add file needs:

- RGB = CavityAO (grayscale depth shading)
- Alpha = not required

Steps:

1. Make only CavityAO visible.
2. File → Export As…
   SEMods/Textures/Thorite_AxisXZ_add.png

What it does:
dark cracks = deeper shadows, lighter ridges = raised material.
This makes the voxel feel embedded in-world, not flat.

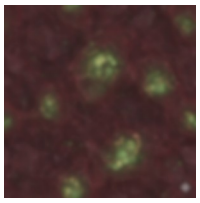
7.4. Export _distance_cm (far-away fallback)

The _distance file needs:

- RGB = DistanceLod (the blurred, low-contrast version)
- Alpha = not required

Steps:

1. Make only DistanceLod visible.
2. File → Export As…
   SEMods/Textures/Thorite_AxisXZ_distance_cm.png



7.5. Export the voxel thumbnail (164 x 12)

This is the tiny “material preview strip” Space Engineers uses to show what the voxel type looks like.

Fastest way to make it:

1. With ColorRGB visible:
   - Right-click → New from Visible
   - Rename it: ThumbnailBase
2. Layer → Scale Layer…
   - Width: 164
   - Height: 12
   - Interpolation: Cubic
   - Scale
3. File → Export As…
   SEMods/Textures/Thorite_thumbnail.png

We now have all the images required to add a new voxel ore to game

We want to go one step further, and add the GUI representations of this ore to the game,

The icon for the ore, and by adding the metal bar icon it would produce after being smelted in the furnace.

The icons given by gemini, will not have a uniformed background color, you will need to replace it by hand or by using some set of filters I am unaware of, for now, I simply open it in paint, trace the icon out using black, and then fill in the rest of the background with black, this way I can see each area that is off color, and replace it with a color we can turn transparent as a whole, once you have the background replaced with black, you can now set transparency correctly.

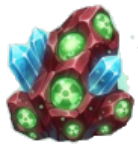
7.6. Export the Ore icon (128 x 128)

Now we switch to the icon sheet.

Open SEMods/Images/Thorite_Icons.png in GIMP.

We'll extract the ore icon first.

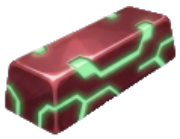
1. Use the Rectangle Select Tool (R).
2. Select around just the ore chunk.
3. Ctrl + C.
4. File → New…
    - Width: 128
    - Height: 128
    - Fill with: Transparency
5. Ctrl + V to paste.
6. Layer → Anchor Layer (Ctrl + H).
7. Image → Crop to Content.
8. Image → Scale Image…
    - Width: 128
    - Height: 128
    - Scale.
9. File → Export As…
   SEMods/Textures/Thorite_ore.png



7.7. Export the Bar icon (128 x 128)

Repeat the same process for the bar / ingot form from that same Thorite_Icons.png.

1. Rectangle Select Tool → select only the bar.
2. Ctrl + C.
3. File → New…
    - Width: 128
    - Height: 128
    - Fill with Transparency.
4. Ctrl + V.
5. Anchor Layer.
6. Image → Crop to Content.
7. Image → Scale Image… to 128 x 128.
8. File → Export As…
   SEMods/MyMod/Textures/GUI/Icons/Ingot/Thorite_bar.png



SECTION 8. CONVERTING PNG TO DDS

Space Engineers expects voxel textures as DDS, not PNG. We'll convert all the PNGs in the Voxels folder using a batch script and Microsoft's texconv.exe (DirectXTex).

8.1. Download texconv.exe
Get texconv.exe from Microsoft's DirectXTex release and put it in:
SEMods/texconv.exe

https://github.com/microsoft/DirectXTex/releases

8.2. Create convert_folder.bat
In SEMods/, create a new file named convert_folder.bat with this content:

```
@echo off
setlocal enabledelayedexpansion
if "%~1"=="" (
    echo Usage: convert_folder.bat path_to_folder
    exit /b
)
set "folder=%~1"
echo Converting all PNGs in: %folder%
for %%f in ("%folder%\*.png") do (
    set "fname=%%~nf"
    echo Processing %%f ...
    set "format=BC7_UNORM_SRGB"
    echo !fname! | findstr /I "ng" >nul && set "format=BC7_UNORM"
    echo Using format !format!
    texconv.exe -f BC7_UNORM_SRGB -m 0 -if LINEAR -sepalpha -y -nologo -nogpu -o "%folder%"
"%%f"
if exist "%folder%\%%~nf.DDS" (
    ren "%folder%\%%~nf.DDS" "%%~nf.dds"
    echo   Output: %folder%\%%~nf.dds
) else (
    echo   !! No DDS created for %%f !!
)
)
echo Done.
pause
```

8.3. Run the conversion
Open Command Prompt:
cd /d path\to\SEMods
convert_folder.bat Textures

This will create .dds versions of:

- Thorite_AxisXZ_cm.dds
- Thorite_AxisY_cm.dds
- Thorite_AxisXZ_ng.dds
- Thorite_AxisXZ_add.dds
- Thorite_AxisXZ_distance_cm.dds
- Thorite_thumbnail.dds
- Thorite_ore.dds
- Thorite_bar.dds

SECTION 9. FINAL FILE PLACEMENT

After conversion:

Put these .dds files in:
SEMods/MyMod/Textures/Voxels/
Thorite_AxisXZ_cm.dds
Thorite_AxisY_cm.dds
Thorite_AxisXZ_ng.dds
Thorite_AxisXZ_add.dds
Thorite_AxisXZ_distance_cm.dds
Thorite_thumbnail.dds

Put these icons in:
SEMods/MyMod/Textures/GUI/Icons/Ore/
Thorite_ore.dds

SEMods/MyMod/Textures/GUI/Icons/Ingot/
Thorite_ingot.dds

SECTION 10. WHAT YOU JUST BUILT

From scratch, with no hand-painting and no art background, you produced a complete voxel material in the same style Space Engineers expects:

1. _cm (color + metalness alpha)

2. _ng (normal + gloss alpha)
3. _add (cavity / AO depth)
4. _distance (far-away fallback)
5. thumbnail (164x12 preview strip)
6. ore icon (128x128, transparent)
7. bar icon (128x128, transparent)
8. proper DDS exports using texconv

In other words:
You can now make your own asteroid ores, ice deposits, alien minerals, etc.

The next PDF will dive into how we add these to game