Critical Review of IFC

Anton Gerdelan - gerdela@scss.tcd.ie

- Lecturing modern 3d graphics
- Been using WebGL for building visualisation
- Need to triangulate and extrude CAD plans for 3d viewing
- AutoCAD etc. file formats are proprietary (closed)
- IFC standard pushed as interchange format for this

3d Mesh Formats

- Huge range and history of 3d file formats
- Most common are originally proprietary (Autodesk, Microsoft, Wavefront, etc.)
- Specialised tasks; animation, 3d modelling, motion capture
- Most are plain-text format
- Some contain indexed data to avoid repeats.

3d Mesh Formats

Tasks of a 3d mesh format:

1. Must provide an **array of triangulated vertex positions** - in consistent winding order
2. Ideally provides an **array of surface normals**
3. Ideally provides an **array of texture coordinates**
4. Ideally provides an **array of tangents**
5. Must be able to readily export/import/convert to and from common tools
6. Must be data-size efficient (especially online)
7. Must be parsing-time efficient
8. Should be easy to write exporters/importers for; well defined and simplified

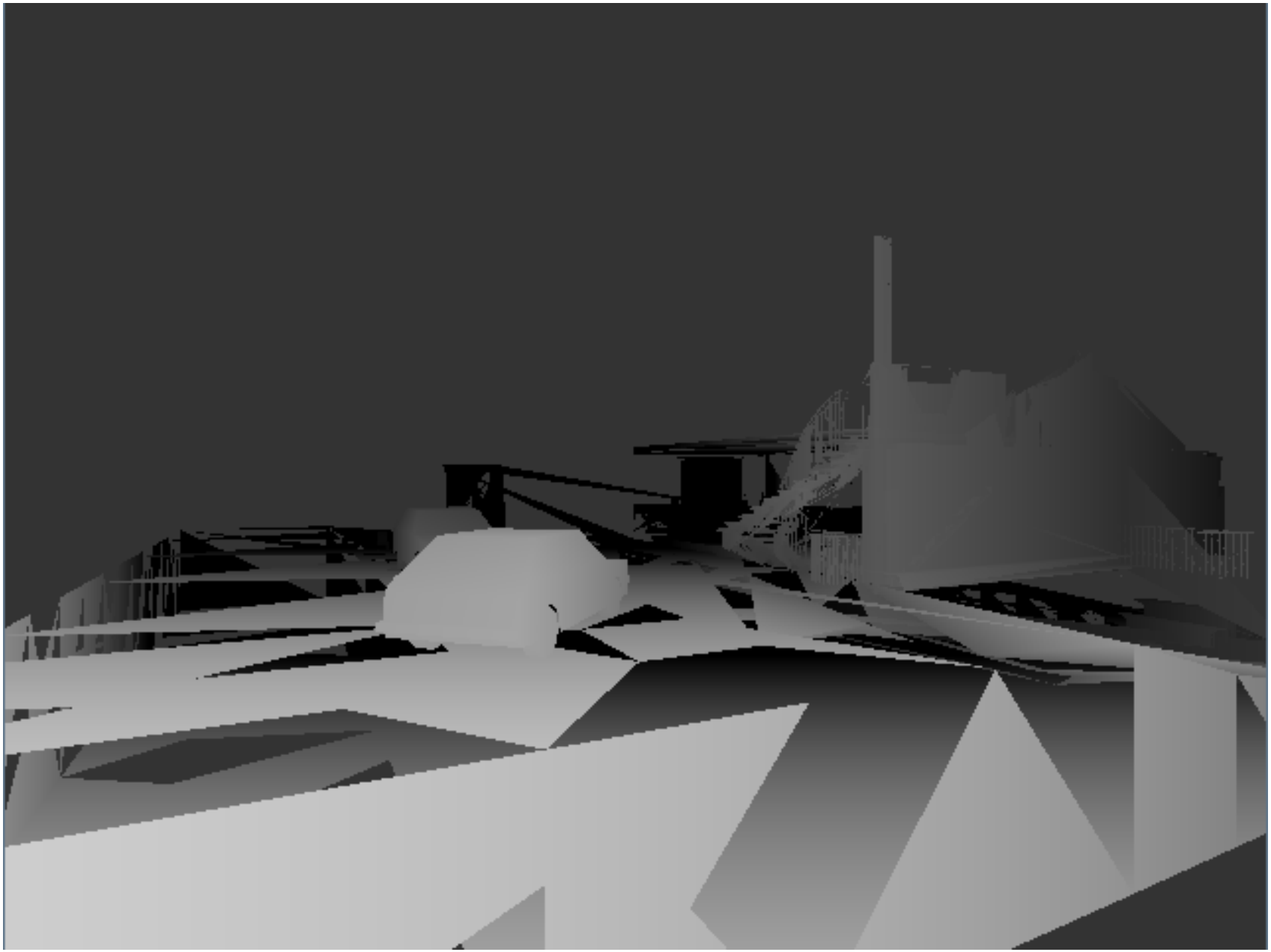
3d Mesh Formats

- All extant formats are horribly designed
- Difficult to find reliable importers/exporters for any format
- Very few are simple enough to write a parser for by hand
- Simple data $\rightarrow$ complex file format $\rightarrow$ simple data
- For other projects I ended up making a custom file format and converter

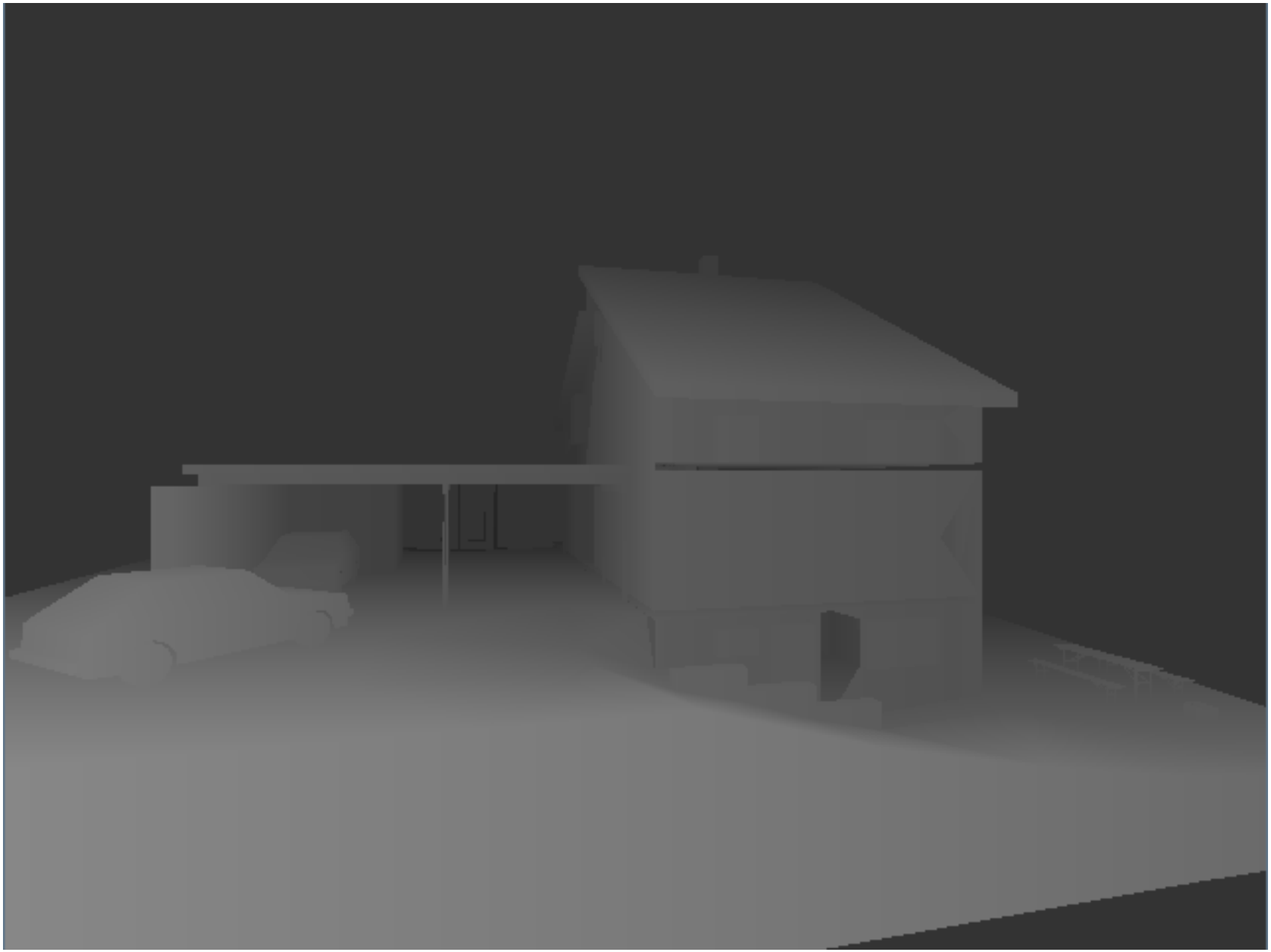
IFC Data Format

- Does not present arrays - presents re-used prefabs in separate coordinate spaces
- "Some assembly required"
- Maybe suited older graphics pipeline, not suitable for modern rendering
- Automatic parsing takes up to 1 minute (should be 1 or 2 seconds).
- Weak rules ('winding' conventions etc.) results in errors
- Inevitable **manual clean-up required** in 3d modelling tool.

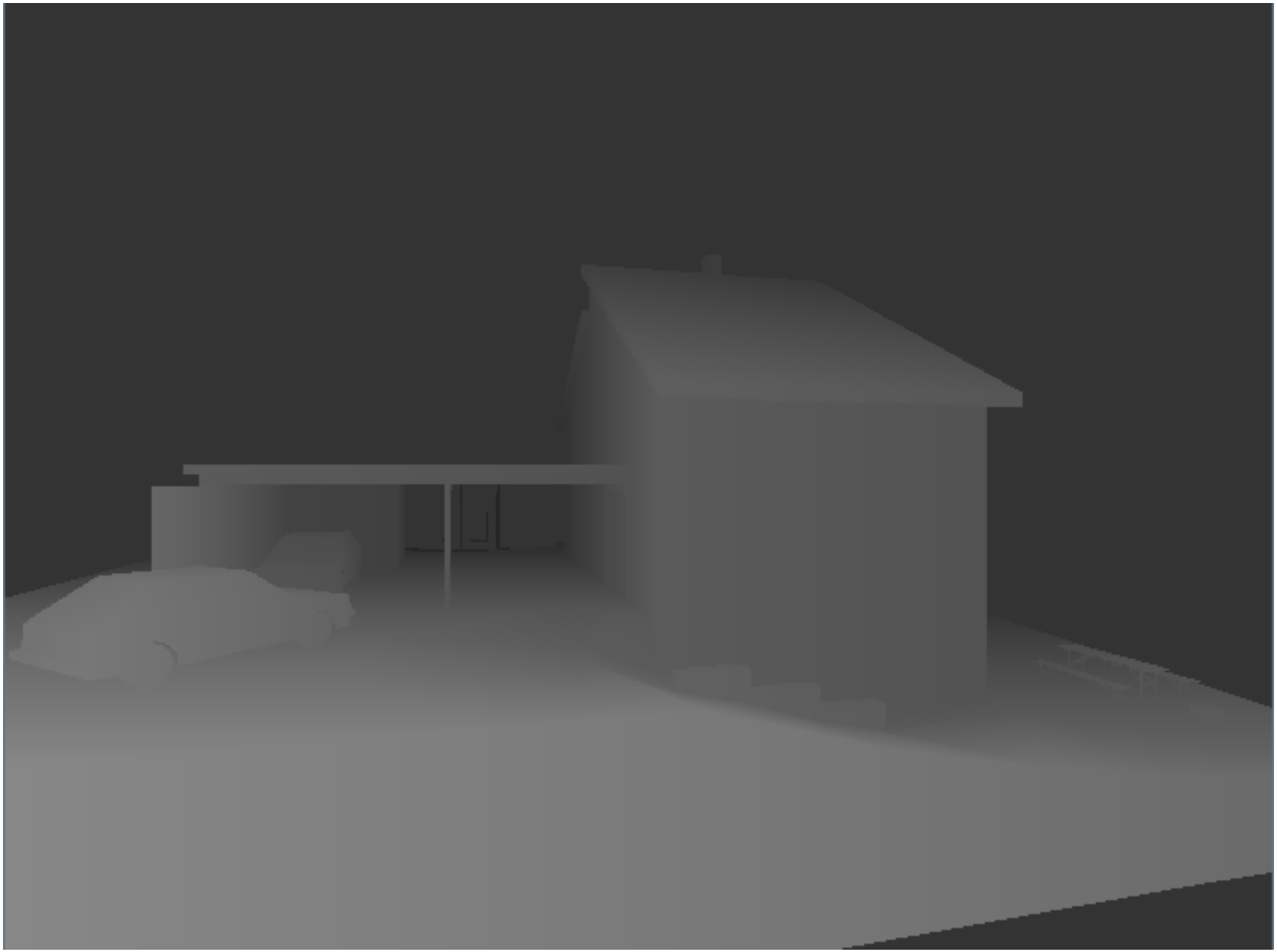
Test Mesh from KIT: Initial Parse



After Manual Clean-Up



After Correcting Winding Order



IFC and STEP

```
#115016=IFCCARTESIANPOINT
    ((0.405495112538337 9,3.440000023841857,0.2529293334484 1)
    );
#115017=IFCCARTESIANPOINT
    ((0.405495112538337 9,3.440000023841857,0.2529293334484 1)
    );
#115018=IFCCARTESIANPOINT
    ((0.407439110577106 7,3.440000023841857,0.2504039640798 57)
    );
#115019=IFCCARTESIANPOINT
    ((0.409779051095247 5,3.420004699528216,0.2504039640798 57)
    );
```

- Format based on "STEP" format
- These lines are 370 bytes for **maximum** 36 bytes of useful data.
- Why are there line numbers? We know the line number in a file.
- Units here are in metres. These claim $\times 10^{-16} m$ precision. Really?

IFC Data Size

File Format	File Size (kB)
.ifc	14600
.obj	1400
.apg	3400

- Converted one of KIT's sample IFC building meshes
- .ifc is $10\times$ bigger than non-optimised (but indexed) Wavefront .obj
- 14MB is far too big for a download, even when zipped
- My non-indexed plain-text format has 4x as much rendering data as .ifc, but much more efficient meta-data

My Opinion

- STEP is amateurish. ISO standards carry 0 weight in graphics community
- **do not use STEP-based IFC or allow it to propagate**
- Follow actual engineering precision standards - don't just make them up!
- Format must follow strict rules regarding winding order, etc. (eliminate ambiguity errors)
- Do not ignore what the graphics community is doing and re-invent the wheel
- Mixing up graphics and other CAD data in a single file is a mistake

Alternatives

- For plain-text: Khronos Group (OpenGL) is building a JSON replacement for COLLADA called glTF `https://github.com/KhronosGroup/glTF/`
- This is an interchange format meaning good support, and easy to read
- i.e. does all the things that IFC/STEP is trying to re-invent separately
- For actual end-use online, why not use a binary encoding?
- Plain text encoding means a 4-byte `float` uses minimum 9-10 bytes (probably more) plus precision issues
- Binary encoding means a 4-byte `float` uses 4 bytes
- Easy to read in C, as well as direct-to-structure in JSON

3d Printing

- Convert CAD file to an interchange format
- Extrude, hand-alter ‘skinny’ bits, close mesh
- I used Blender to do the above.
- Convert to STL (a very simple 3d file format) for printer software

3d Printing