end

Anton Gerdelan <gerdela@scss.tcd.ie>
Trinity College Dublin

today

- some thoughts about your next year
- what to learn next (more advanced stuff)
- what to expect from the field in the future
- a short tech talk on 2d rendering

interviews

- Keep your game project to talk about
  - Know about limitations, possible optimisations
  - Put some tidy, simple code on Github etc.
- Project enthusiasm

interviews

- Make big bucks of OpenGL 4+ experience
- Prepare for ridiculous speed coding expectations
- Internships can be insightful / meet and greet
- Jams (especially collaborative)
- Know the answer to that bit hack slide in Texturing

research

- here at GV2
  - Graphics, [Computer] Vision, Visualisation
  - Perceptual studies
  - Simulations
- potential supervisors
- think about the benefits of overseas study
  - ask about good supervisors in area elsewhere

research and/or jobs

- Find someone good in your area
- Make sure the group is friendly
  - Commitment of several years
- Think/discuss/argue for a solid idea before-hand
- Make sure that someone has funding available

next steps in graphics

coming back to do more later

- Don't worry if pace was too fast
- Got the gist of modern 3d vector graphics
- Took me about a year - OpenGL+shaders
- KISS
- Use/try a bigger range of tools

advanced topics

- WebGL and mobile devices
- geometry and tessellation shaders
- compute shaders
- UBOs, texture storage, streaming data buffers
  - “approaching zero driver overhead” talks
- Normal maps, texture projection shadows
- Radiosity, ambient occlusion, global illumination

Future Graphics

- more GPU cores and memory
- glNext - ? simple API with no binding state machine?
- Better culling and clipping algorithms
- Metal, Mantle, alternative APIs
- Direct3D 12 – supposed to be much more efficient

Requested Topic

2d Rendering

Hardware-accelerated 2d Rendering

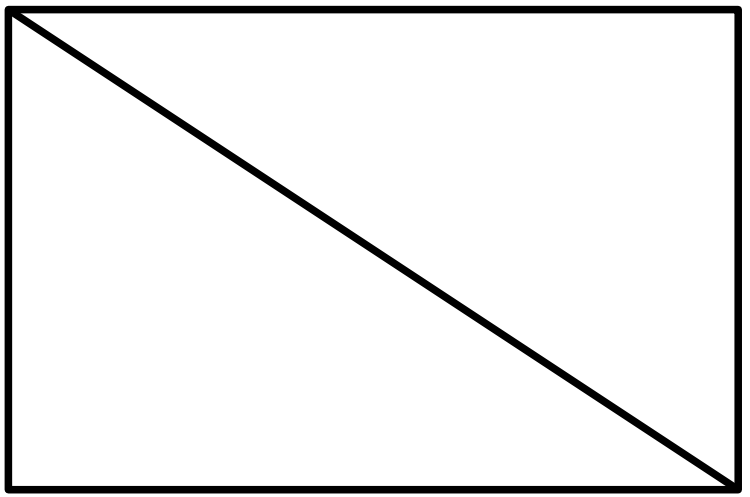
- Modern graphics for sprites is way harder than old-fashioned direct pixel-writing
- Advantages of using vector graphics for 2d?
- Is there an easy way to do this?

Sprites

- `gl_Position` **Z value** is zero for everything
- Draw in **back-to-front order** where possible
  - No need for depth buffer
  - Easy to work out transparency / colour blending

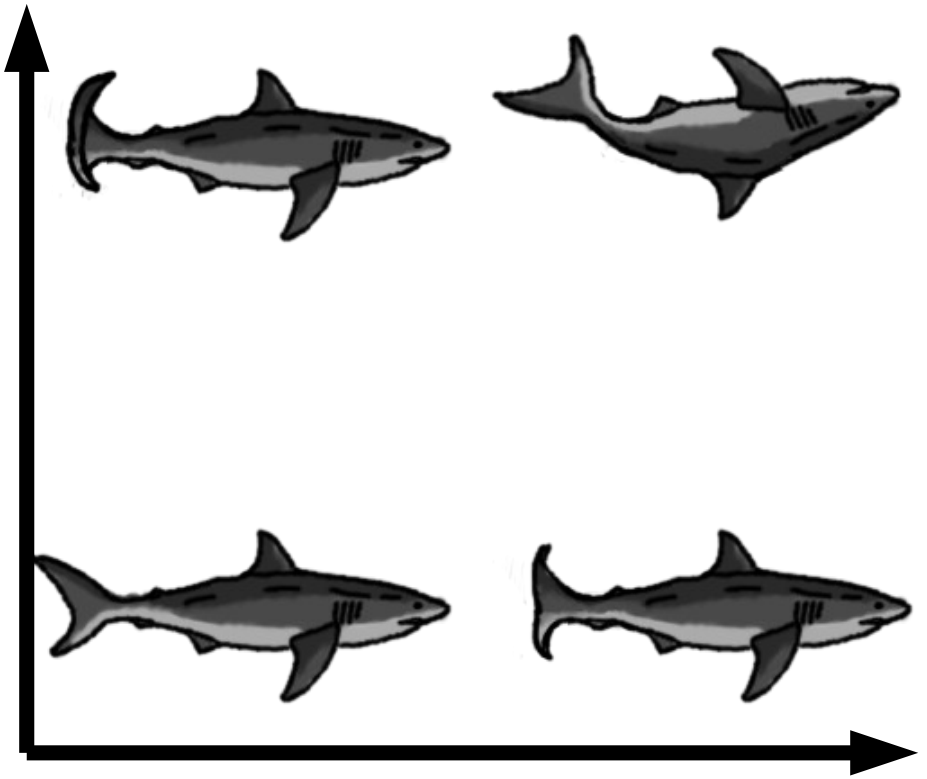
Sprites

- Each sprite is a 2d quad (2 triangles)
  - Could use attribute-less rendering
  - Scale to size
  - Use a model matrix or `uniform vec3` to move
  - Sample a texture



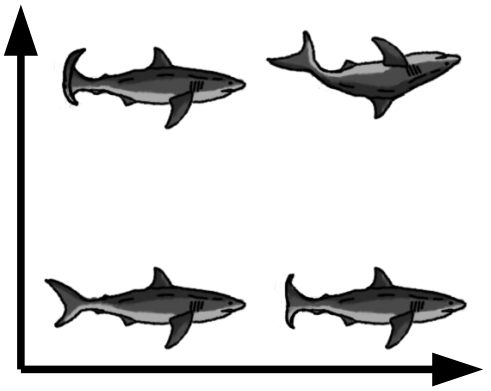
2d Animations

- Could switch textures
  - High overhead
- Better to use an atlas
  - Divide image into grid
  - Padding **Q. why?**
- **anim num → tex coords**
- Time between frames
- Repeat
- **DEMO**



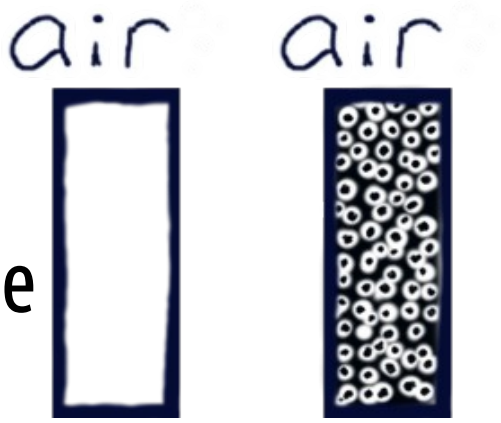
2d Animations

```
void change_sprite (int sprite_index) {

  const int num_cols = 2;
  const int num_rows = 2;

  int col = sprite_index % num_cols;
  int row = num_rows - 1 - sprite_index / num_rows;
  float s = (float)col / (float)num_cols;
  float t = (float)row / (float)num_rows;
  glUseProgram (sp);
  glUniform2f (st_offset_loc, s, t);

}
```

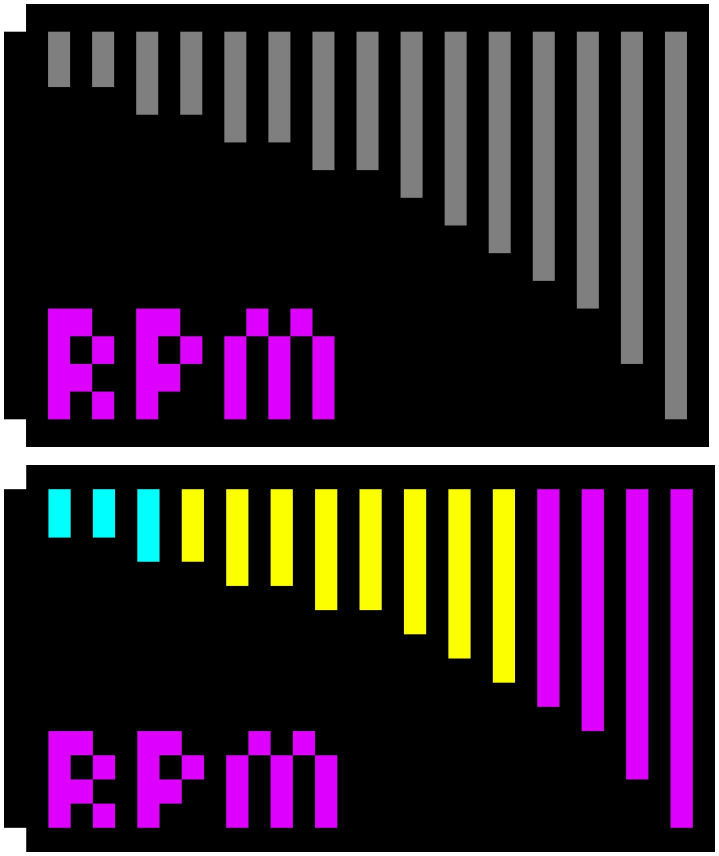


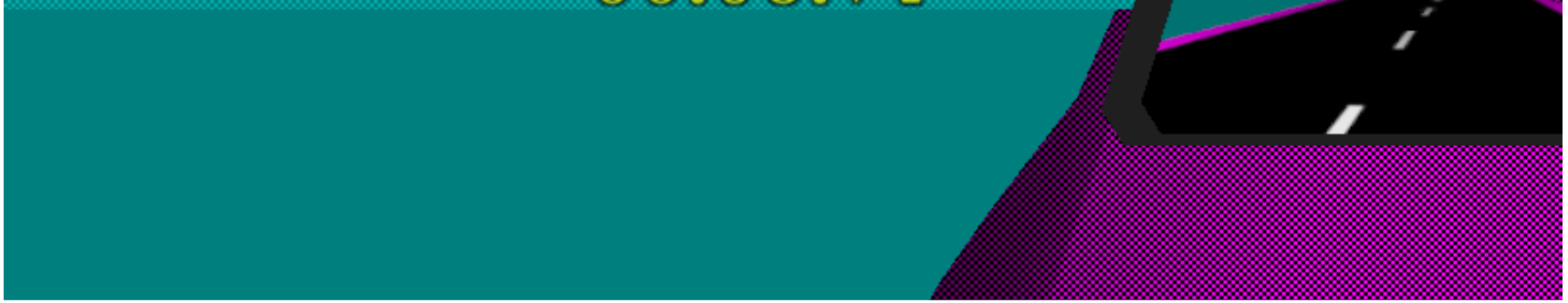
- Scale texture coordinates down to 1/4
- Offset by +0.25 for second row or column
- I also flipped **t** so index 0 was top row

GUI Progress Bars

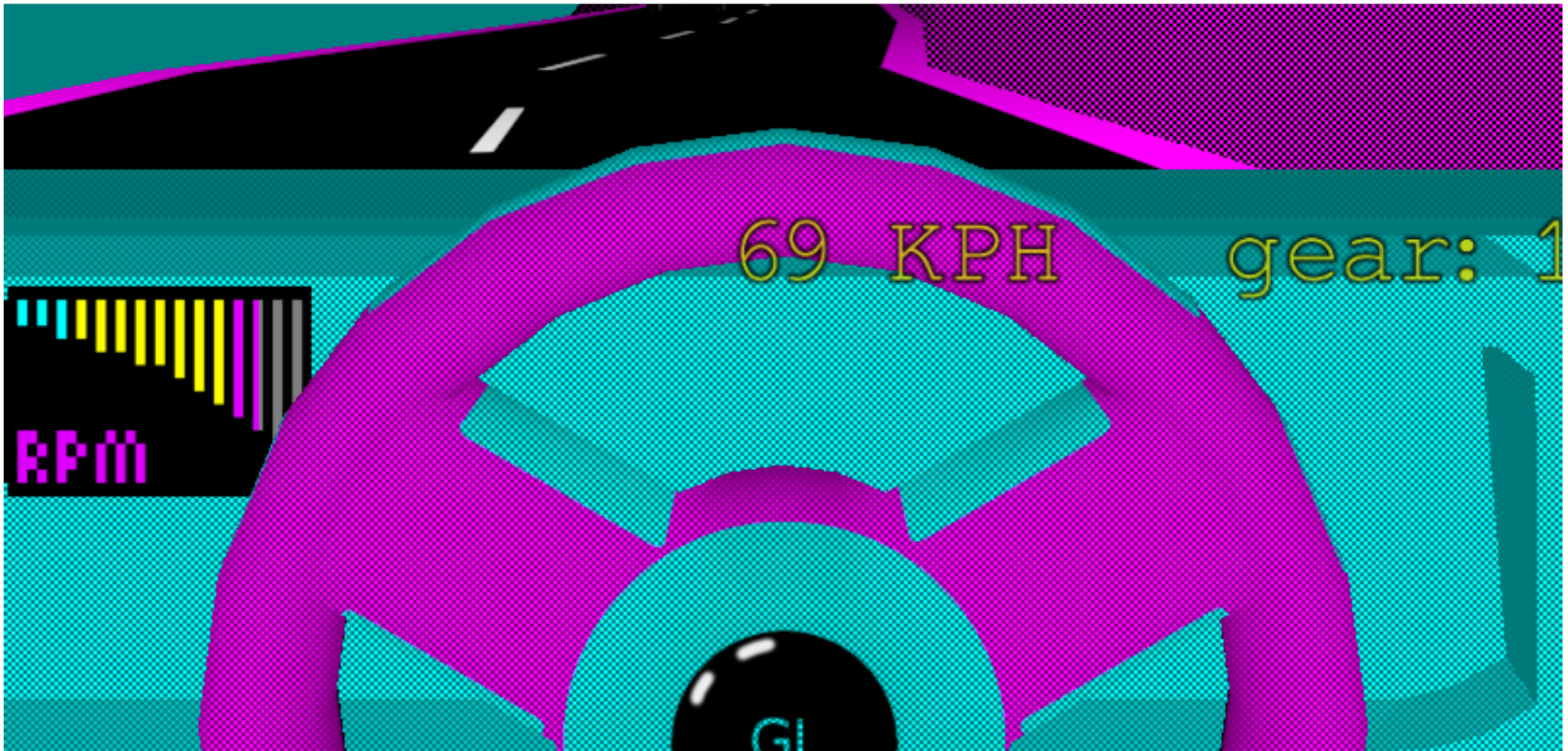


- Also 2d quads
- Draw <u>after</u> the scene
- Empty image
- Full image
- `uniform float factor` 0..1
- Multi-texturing





```
if (st.s < rpm_fac) {
   fc = texture (full_dm, vec2 (st.s, 1.0 - st.t));
} else {
   fc = texture (empty_dm, vec2 (st.s, 1.0 - st.t));
}
```



Alpha Blending

- Transparency in GIMP, Photoshop etc. sets value for pixel alpha
- We have to tell OpenGL what to do with this
- `glEnable (GL_BLEND);`
- `glBlendFunc (GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);`

- Various blend functions exist
  - New pixel colour = (1.0 – a) * previous + a * current
- Can be semi-transparent
- This is not realistic transparency – it's just a mixing function

Alpha Blending

- `glBlendFunc (GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);`
- Various blend functions exist
  - New pixel colour = (1.0 – a) * previous + a * current
- Can be semi-transparent
- This is not realistic transparency – it's just a mixing function

Framebuffer Effects Framework

- Drawn main scene to texture
- Lots of special effects can be added
  - In the old-fashioned pixel-modifying way
- Blur
- Shake
- Rotate, twist, skew, flip
- Animated things: **demo**

Distributing Your Work

- Build in Release mode
- Libraries
  - Release version
  - Static libs
  - **Dependency Walker**
  - VS redistributables "*DLL Hell*" - or just don't use Visual Studio
- Relative paths to data being loaded
- Binary lump / wad /zip of data

Distributing Your Work

- Test on multiple machines
- Provide user options, and min(), fall-backs, or multiple builds of GL
- Find all possible GL and GLSL errors
  - gDEBugger
  - Debug callback
  - Testing
- Apple .app builds
  - Script to create a folder hierarchy

end

- Tomorrow – 11am
  - Prefabs – set up early / kick out non-gfx ppl
  - Have some sort of fall-back for laptop fails
  - Talk one of us through a live demo
  - Impressions
  - Submit report, **videos**, code next week
  - Grade by consensus
- Hope you enjoyed lectures for CS4052!