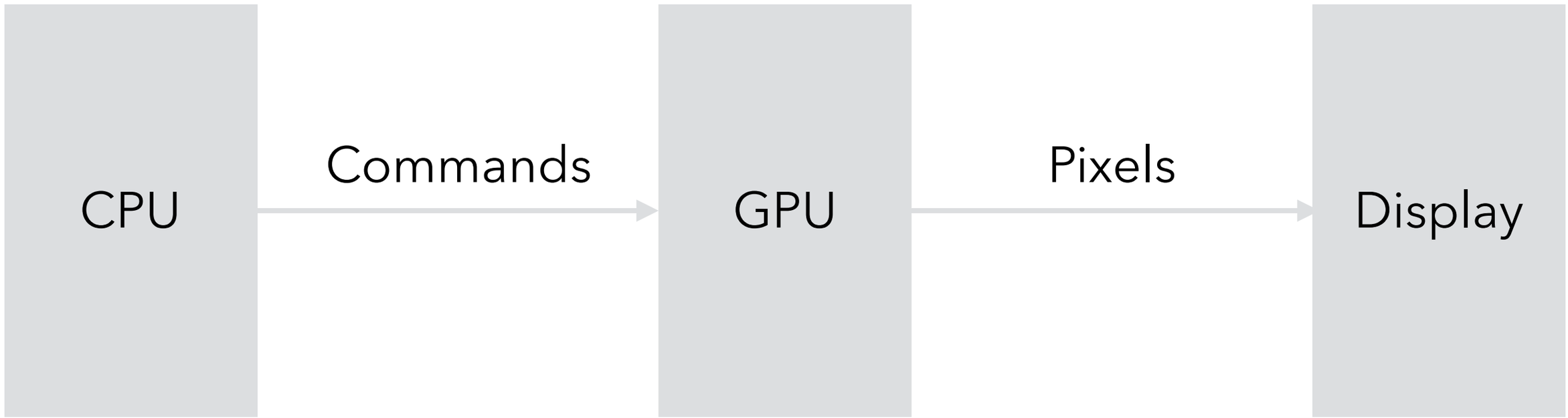
Let’s Fix OpenGL

Adrian Sampson, Cornell


Mozilla Developer Network
Mozilla Developer Network
mozilla.org
https://developer.mozilla.org/en-
Google
Feedback
HW
mozilla
About MDN
Geolocation
Drag & Drop
File API
Web Workers
and more...
AJAX
WebSockets
Offline Cache
Local Storage
IndexedDB
Firefox Add-ons
Mozilla Applications
FIREFOX FOR DEVS
APPS AT MWC 2012
27 February - 1 March
Barcelona
Inspect
div.wrap
section#c
section#h
div#prom
a
h2
3D
HTML
Style


VIVE


●○○○○ T-Mobile LTE 4:45 PM
Bloomberg L.P.
Bloomberg
Bigger Than Hollywood: The Numbers Behind Video Gaming
10:17 PM PDT August 4, 2015


CPU
Commands
GPU
Pixels
Display

Rendering Pipeline
programmable & fixed-function stages

C, C++,
JavaScript

GLSL

GLSL

CPU

Vertex
Shader

Fragment
Shader

vertex positions

pixel colors

Vertex Shader

```
in vec4 position;
in float dist;
out vec4 fragPos;
void main() {
   fragPos = position;
   gl_Position =
     position + dist;
}
```

Fragment Shader

```
in vec4 fragPos;
void main() {
   gl_FragColor =
     abs(fragPos);
}
```

CPU “Host Code”

setup

```
static const char *vertex_shader =
  "in vec4 position; ...";
static const char *fragment_shader =
  "in vec4 fragPos; ...";

GLuint vshader = glCreateShader(GL_VERTEX_SHADER);
// ... more boilerplate ...
glLinkProgram(program);

GLuint loc_dist =
  glGetUniformLocation(program, "dist");
```

render a frame

```
glUseProgram(program);
glUniform1f(loc_dist, 4.0);
// ... assign other "in" parameters ...
glDrawArrays(...);
```

#1 Shader languages are subsets of supersets of C.

#1 Shader languages are subsets of supersets of C.

`struct T { ... }`

declares a typed named:

A. `struct T`

B. T

Like C++

#1 Shader languages are subsets of supersets of C.

Declarations inside conditions

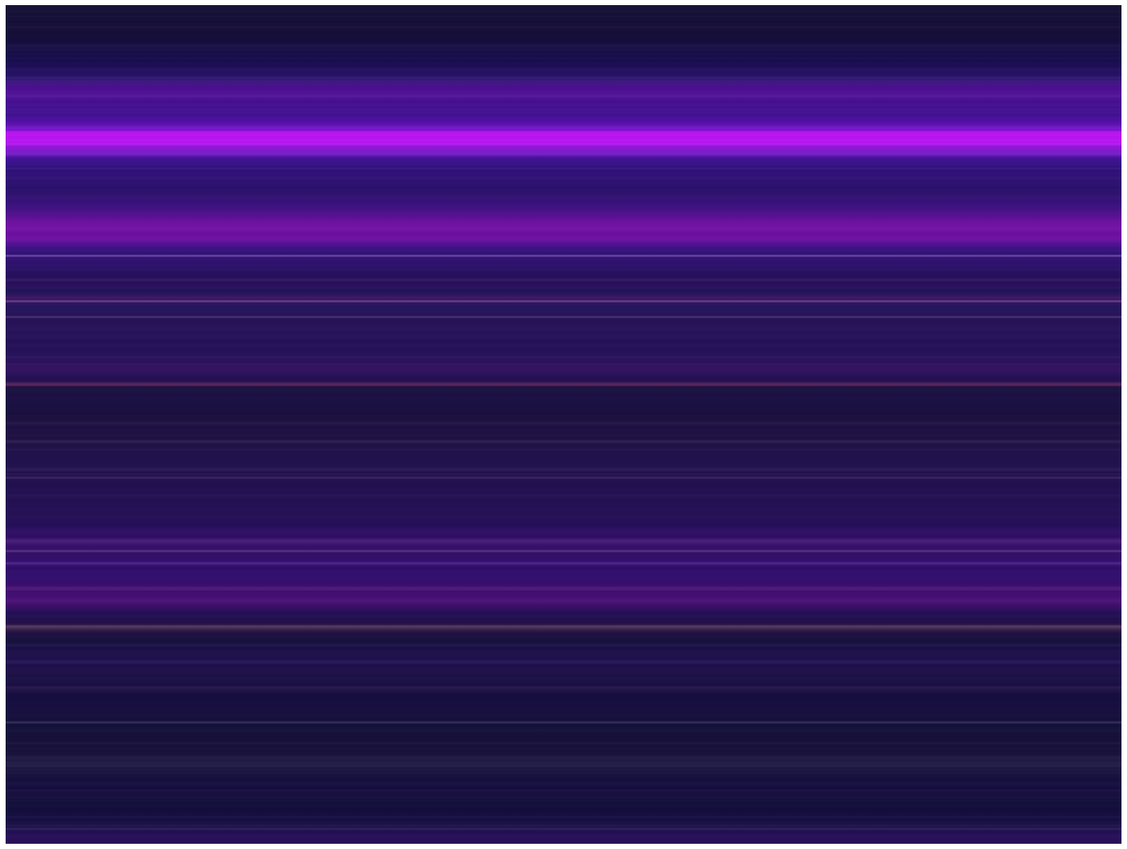
`if (bool b = ...) { ... }`

are allowed.

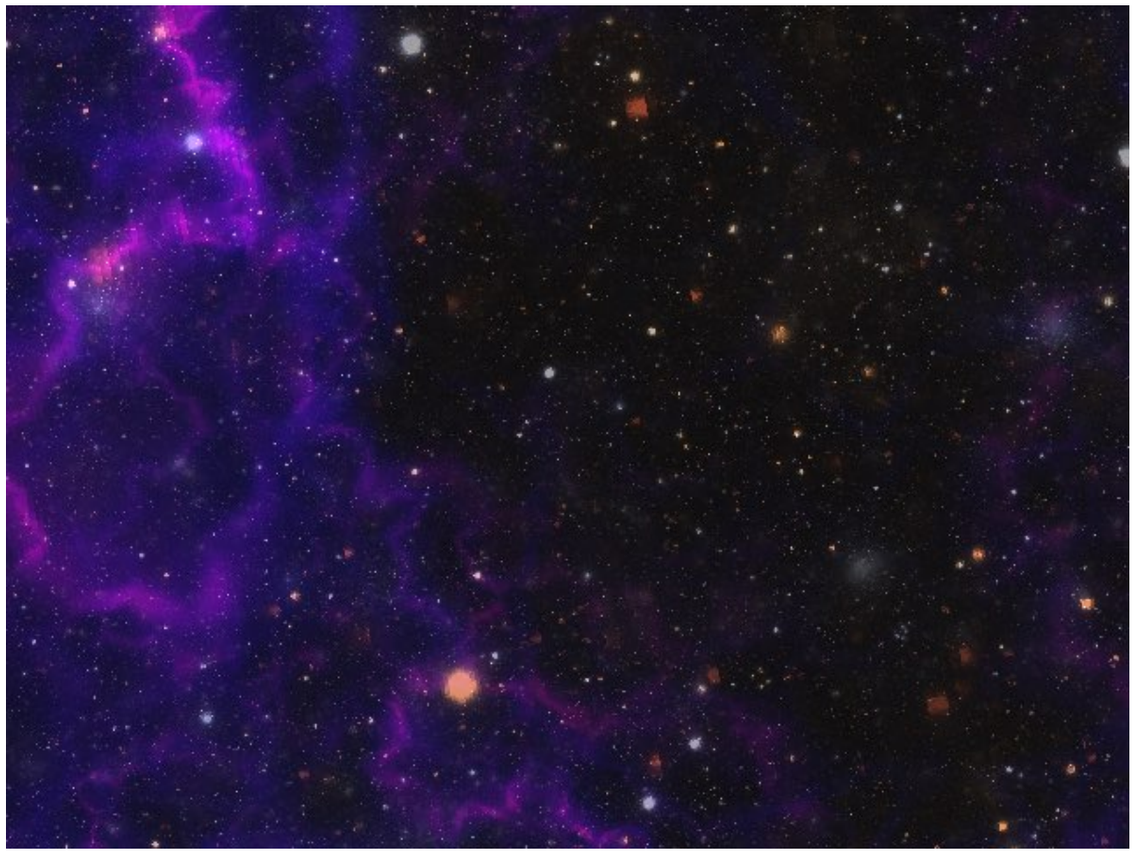
true

false

Like C



ARM Mali GPU on Android.



Correct output.

“It looks like there was indeed an obscure register allocation bug in the driver...” —ARM

from “Bugs can be beautiful," by Alastair Donaldson. https://medium.com/@afd_icl/65b93c5c58f9
c.f. “Metamorphic Testing for (Graphics) Compilers,” by Alastair Donaldson and Andrei Lascu.
http://www.doc.ic.ac.uk/~afd/homepages/papers/pdfs/2016/MET.pdf

#1 Shader languages are subsets of supersets of C.

⇒ Apply work on language extensibility.

#2 Loose CPU-to-GPU and stage-to-stage coupling.

```
GLuint loc_dist =
  glGetAttribLocation(program, "dist");
```

CPU setup

```
glUniform1f(loc_dist, 4.0);
```

CPU render

```
in float dist;
```

vertex shader

#2 Loose CPU-to-GPU and stage-to-stage coupling.

```
GLuint loc_dist =
  glGetAttribLocation(program, "dist");
```

CPU setup

```
glUniform1f(loc_dist, 4.0);
```

CPU render

```
in float dist;
out vec4 fragPos;
```

```
fragPos = position;
```

vertex shader

```
in vec4 fragPos;
```

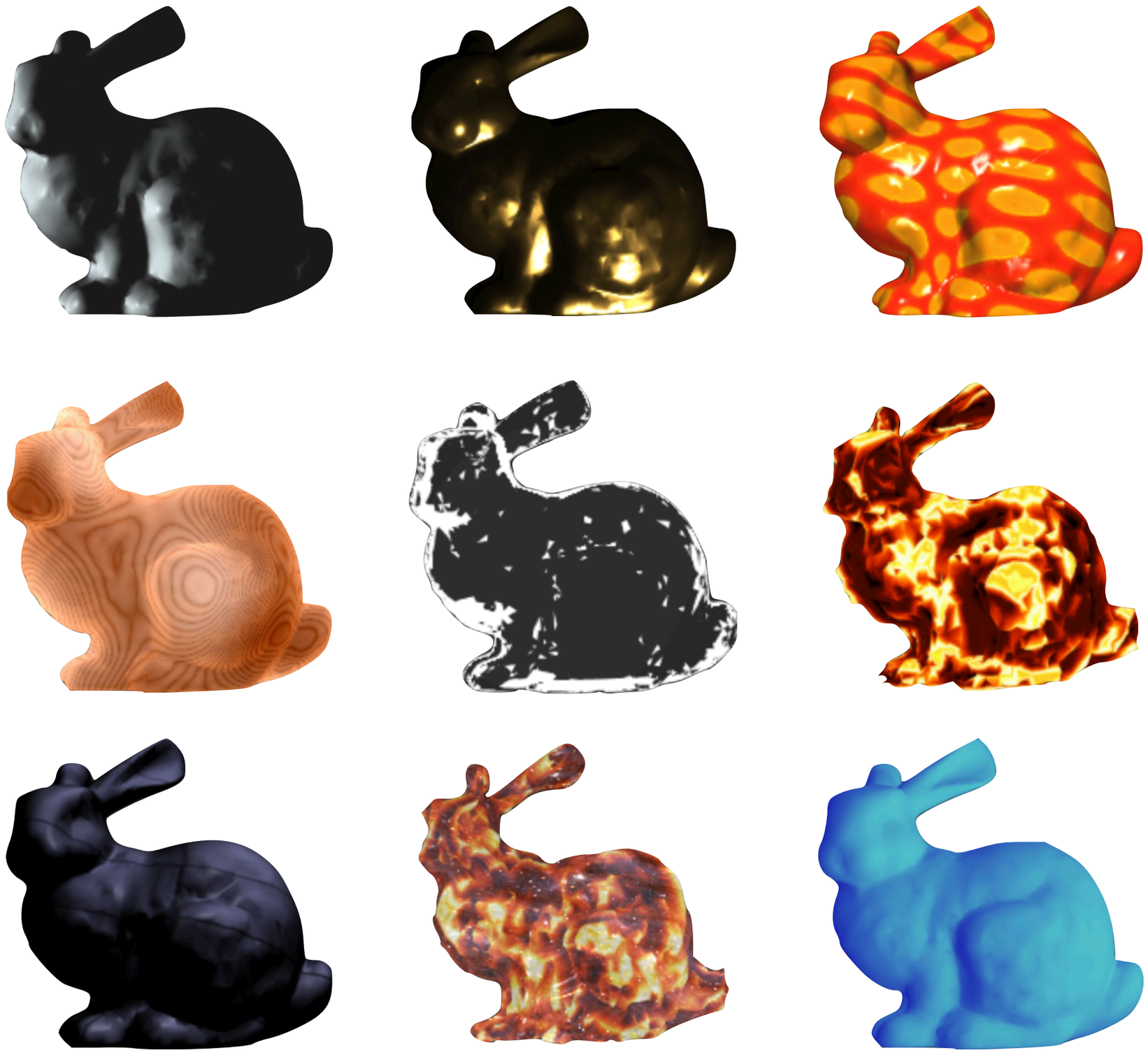
fragment shader

#2 Loose CPU-to-GPU and stage-to-stage coupling.

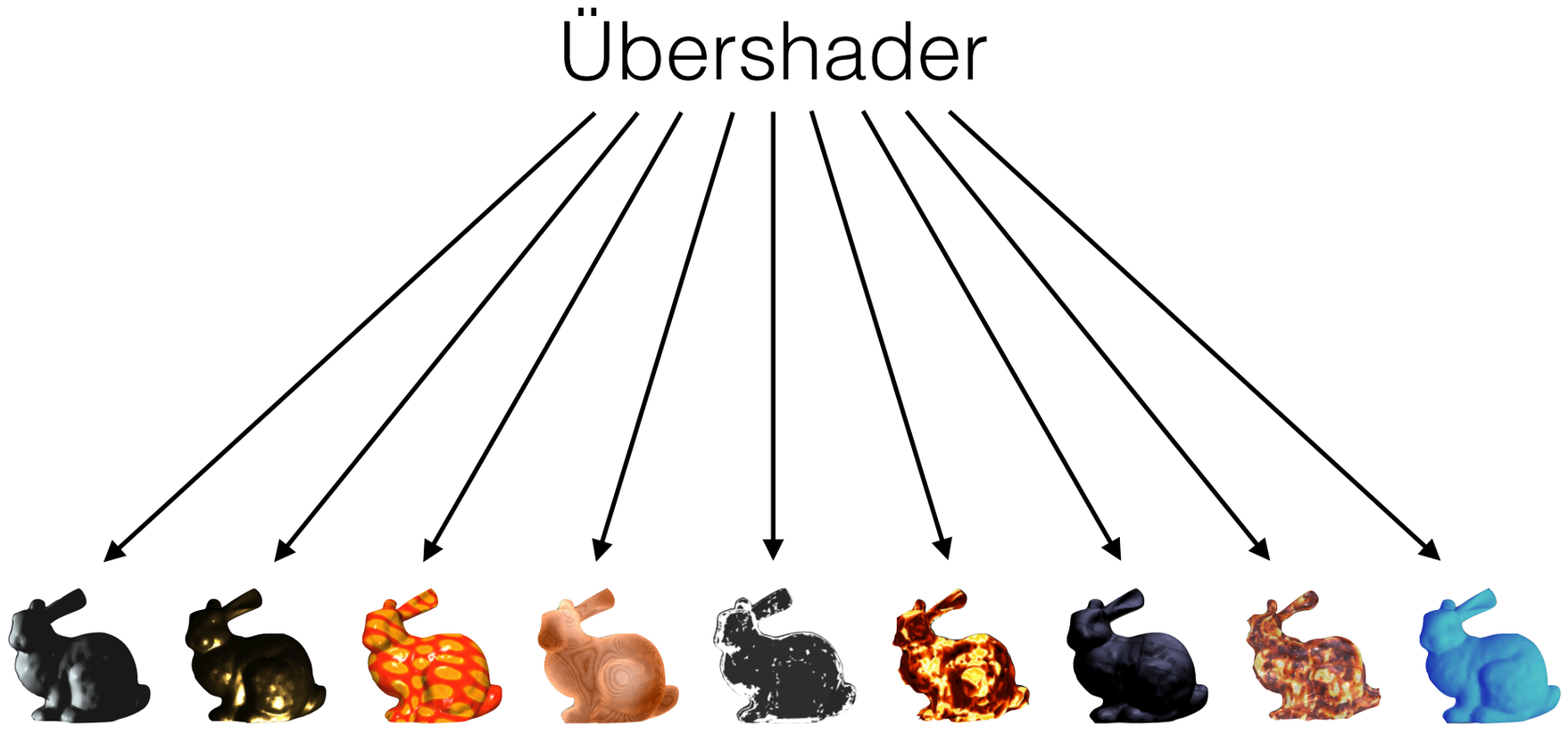
⇒ Cross-language static analysis for the short term.

⇒ Single-source CPU+GPU languages for the long term.

#3 Massive metaprogramming without hygiene.



#3 Massive metaprogramming without hygiene.



#3 Massive metaprogramming without hygiene.

```
glUniform1b(param_loc, true);
```

```
in bool param;

if (param) {
  ...
}
```

VS.

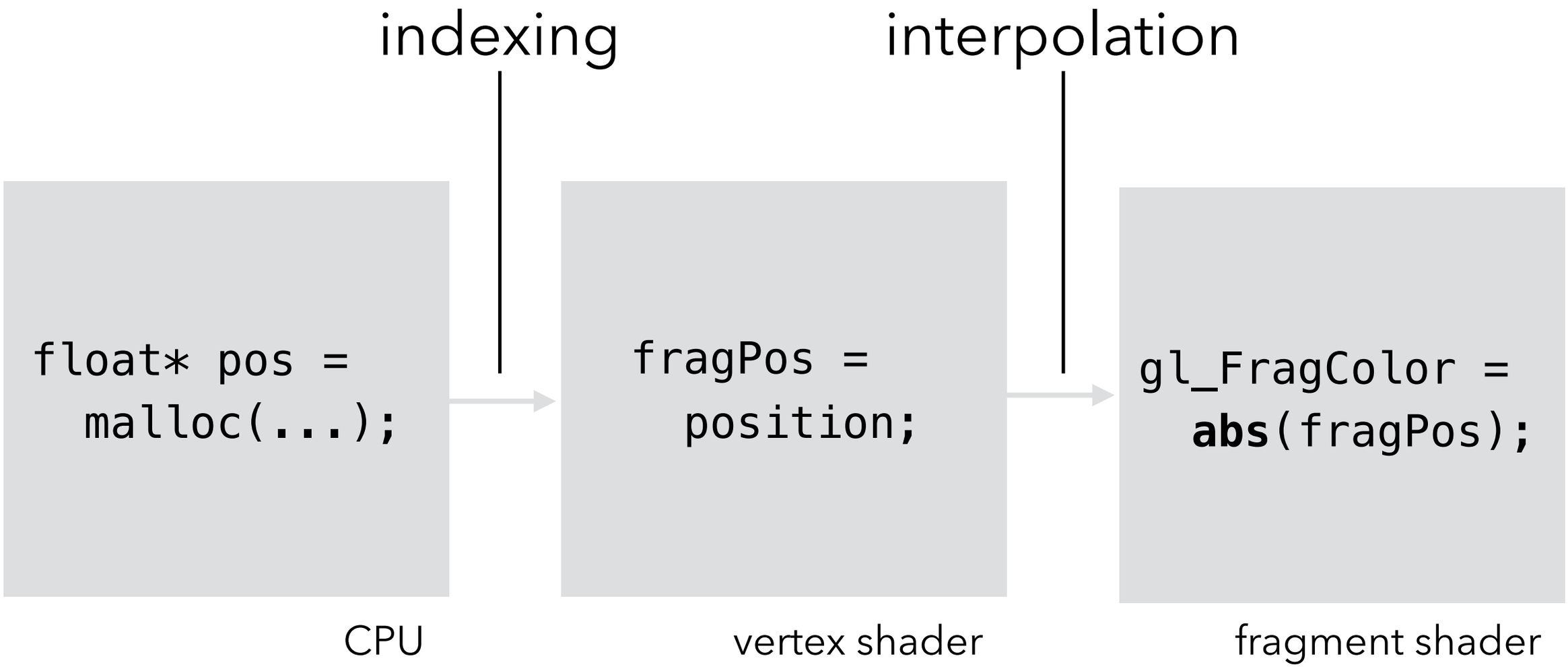
```
glShaderSource(
  "#define _PARAM",
  "...");
```

```
#ifdef _PARAM
...
#endif
```

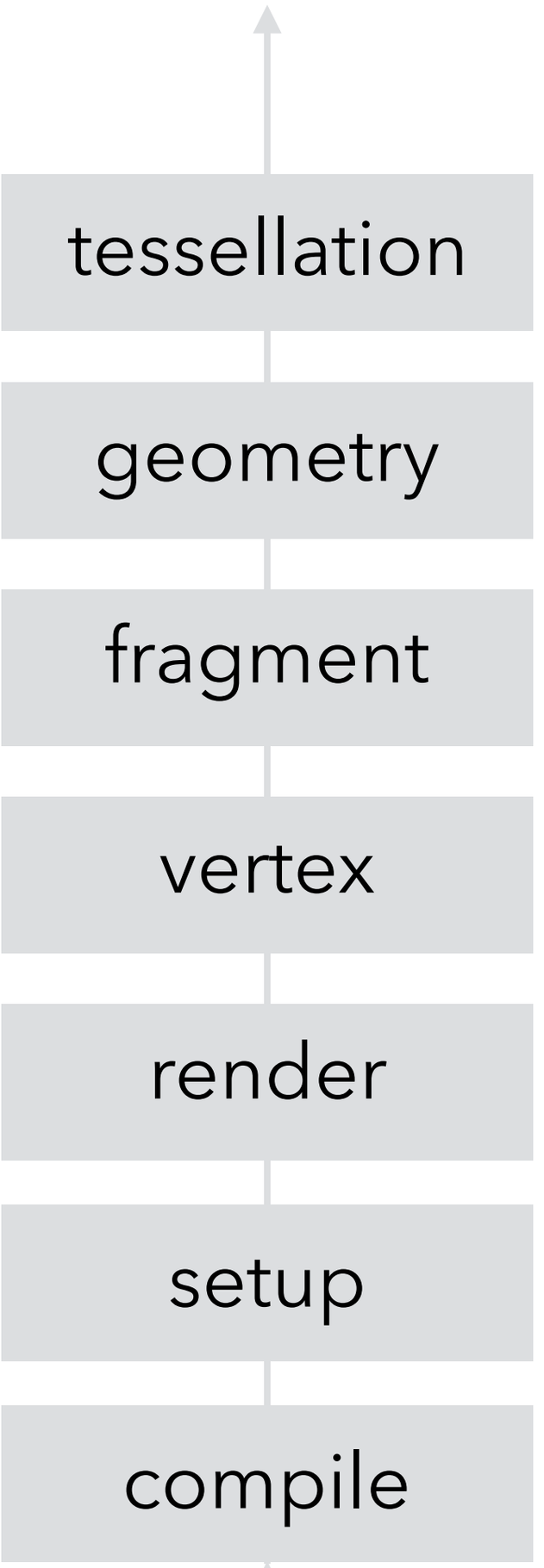
#3 Massive metaprogramming without hygiene.

⇒ Scale up safe metaprogramming tools
to generate thousands of shader variants.

#4 Missing general theory for execution rates.



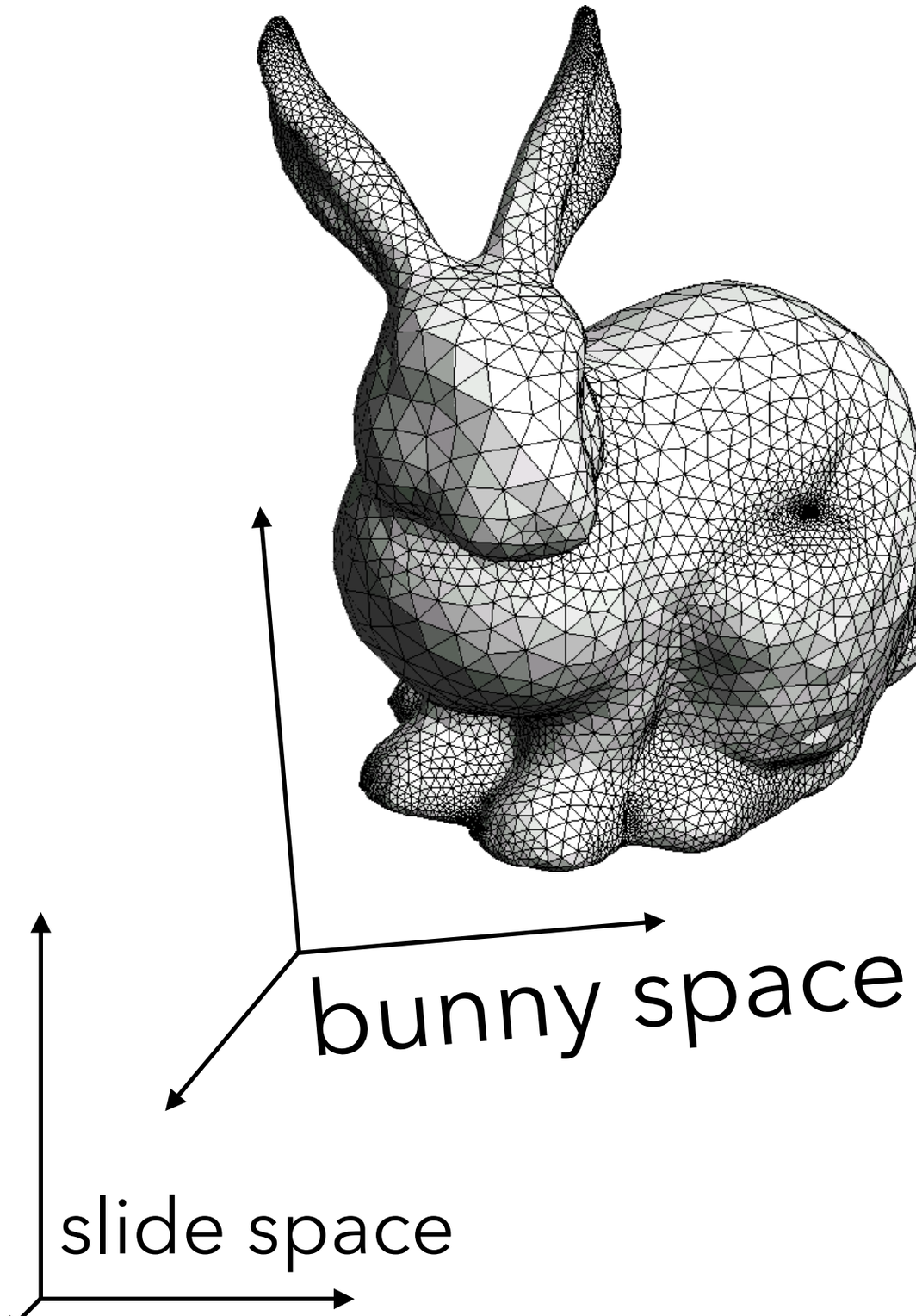
#4 Missing general theory for execution rates.

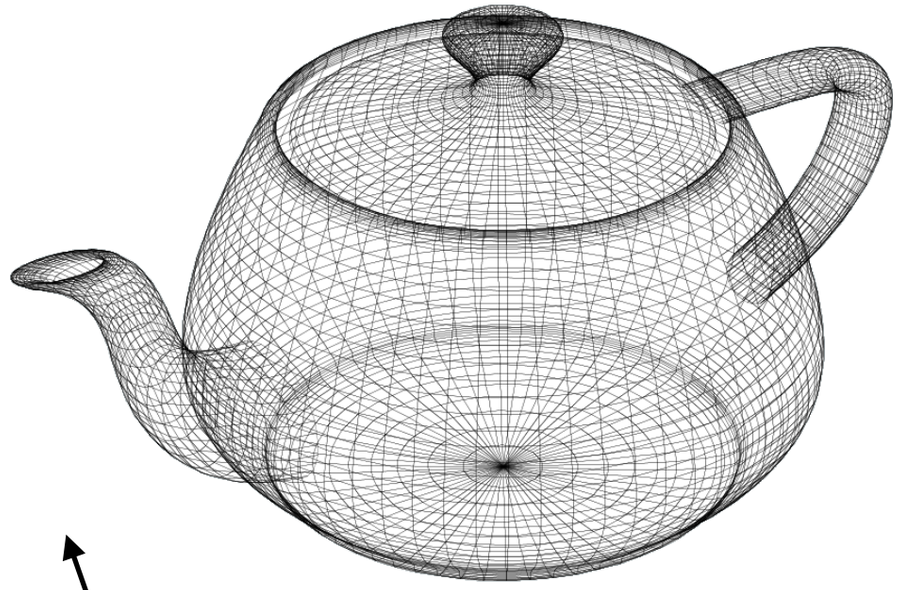


#4 Missing general theory for execution rates.

⇒ Define a core λ_{GPU} to describe programming with rates and a translation from OpenGL.

#5 Latent types for linear algebra.





#5 Latent types for linear algebra.

```
in mat4 bunny_model;
in vec4 position;
```

```
vec4 position_slide = bunny_model * position;
```

#5 Latent types for linear algebra.

```
in mat4 bunny_model;
in vec4 position;
in vec4 normal;
```

```
vec4 position_slide = bunny_model * position;
vec4 normal_slide =   bunny_model * normal;
```

```
normal_slide - position_slide
```

```
normal - position_slide
```

#5 Latent types for linear algebra.

⇒ Use a type system to track the space for each vector.

⇒ Synthesize matrix–vector multiplications
to declaratively obtain a vector in the right space.

#6 Visual correctness.

#6 Visual correctness.

Add aspirational Phong test and its intrinsics

master

Browse files

sampsyo committed on Oct 31, 2015 — 1 parent ac0a7bb — commit 6b2a742d4fa58a160f9add1cf02773c9a773e966

10 months!

Fix wrong variable reference in phong shader

master

Browse files

sampsyo committed on Aug 4, 2016 — 1 parent e86f7b0 — commit f7f3c03fb06ede73c654a492c3b5135f7a9f608b

Showing 1 changed file with **1 addition** and **1 deletion**.

Unified | Split

2 dingus/examples/phong.ss — View

```
@@ -12,7 +12,7 @@ def phong(pos: Float3 Array, norm: Float3 Array, model: Mat4, lightpos: Vec3, co
12 12      fragment glsl<
13 13        # Convert to world space.
14 14        var position_world = vec3(model * vec4(pos, 1.0));
15    -      var normal_world = normalize(vec3(model * vec4(pos, 0.0)));
   15 +      var normal_world = normalize(vec3(model * vec4(norm, 0.0)));
16 16        var view_dir_world = normalize(camera_pos - position_world);
```

#6 Visual correctness.

⇒ Apply work on live coding to shorten the debug cycle.

⇒ Apply crowdsourcing to evaluate visual quality.

Application

"Engine"

programming interface

portable hardware abstraction

GPU

Application

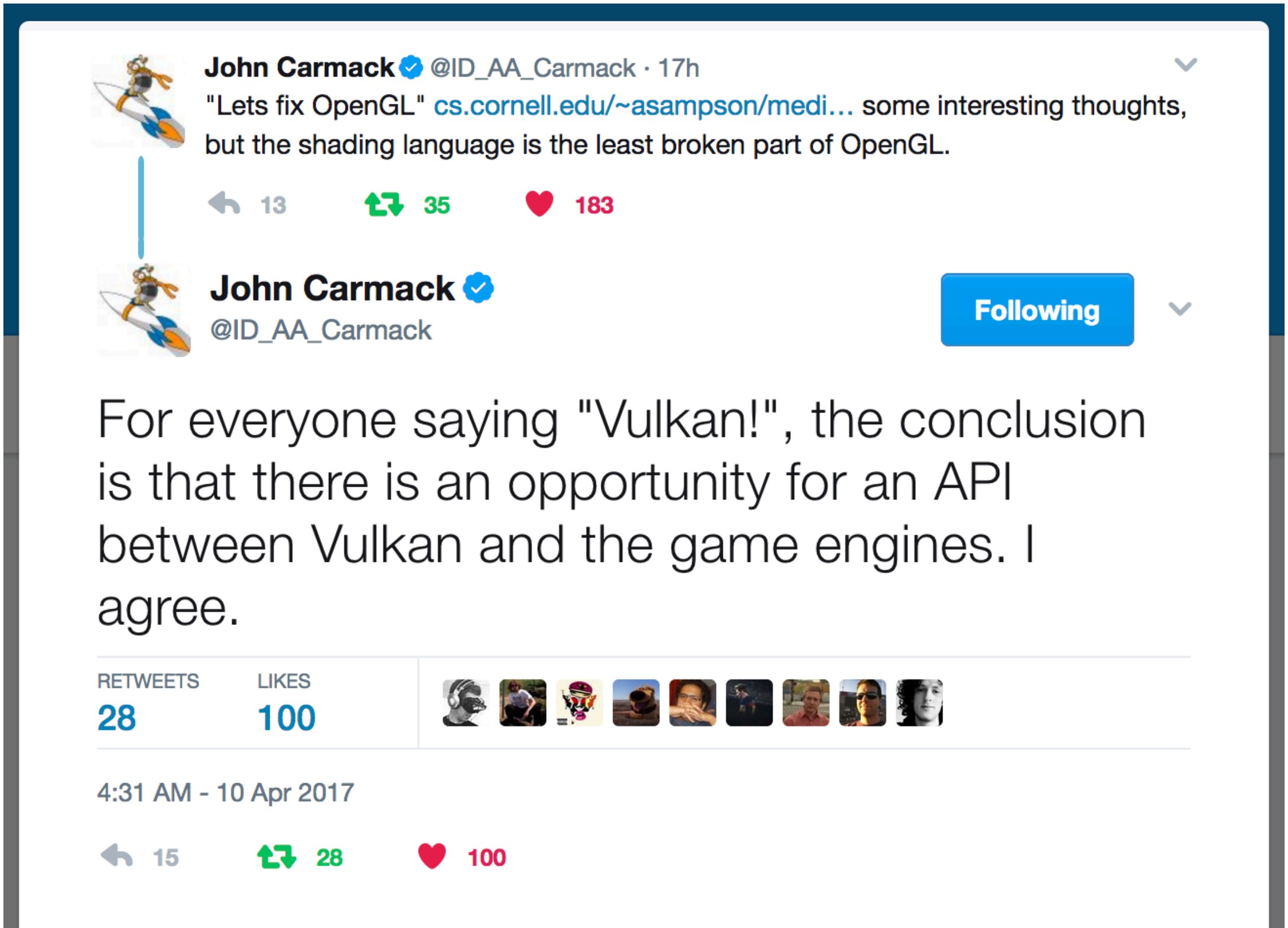
"Engine"

new programming models?

Vulkan

GPU

https://twitter.com/ID_AA_Carmack/status/851397231320150017



Special thanks to:

Kathryn McKinley

Todd Mytkowicz

Yong He

Kayvon Fatahalian

Tim Foley

Pat Hanrahan