AMETHYST:
THE GEM IN THE RUST

RUST NYC OCTOBER MEETUP 2018

LUCIO FRANCO

TOPICS

- What is a game engine?
- What is Amethyst?
- How does Rust help?

WHAT IS A GAME ENGINE?

- Collection of libraries
- Provides support for windowing, rendering, input and more
- Large code base with many moving parts
- Engine components are often tightly coupled
- Hard to parallelize

WHAT IS AMETHYST?

- Core engine that glues everything together
- Collection of crates (modules)
- Powered by an Entity Component System (ECS) model
- Data oriented/Data driven
- Parallelism at its core
- Focused on reusability and clean interfaces

DATA-ORIENTED

- Programming paradigm
- Exploits modern hardware
- Pipelining
- Modularity
- Parallelism

DATA-DRIVEN

- Software design style
- Easier hot reloading
- Easier prototyping
- Scales better
- Better organization of game logic

THE PAST

- Created by Eyal Kalderon (@ebkalderon) in early 2016
- Started growing rapidly in early-mid 2016
- Specs was born (**S**pecs **P**arallel **ECS**)
- Shred was born shortly after (**Sh**ared **Re**source **D**ispatcher)

THE PRESENT

- Growing as the #1 Rust all-purpose modular game engine
- 33.7K Lines of Code
- 14 crates in the main repo, with many more externally
- A book! (still a work in progress)
- 22 Examples
- Supports Windows, MacOS and Linux

THE PRESENT

- Rendering powered by gfx pre-II
- Controllers and gamepads
- glTF - “JPEG of the 3D world”
- Networking/Multiplayer
- UI
- 3D and 2D Animation
- Parallel ECS (Specs)
- Input abstractions
- Configuration loading through RON
- Asset loading with hot-reloading
- State manager

AND MORE!

THE FUTURE

- New renderer that is built off of Vulkan like APIs (gfx-hal and ash)
- More robust networking
- Editor
- Scripting
- WASM and WebGL
- iOS and Android support
- REPL

HOW DOES RUST HELP?

PARALLELIZATION

- Entity Component Systems
- Efficient usage of Vulkan and Metal
- Framegraphs
- More writing code, less tracing data races and segfaults

COMPILER

- Type and safety checking
- Trait composition
- Type inference
- Bugs

CARGO

- Build
- Run
- Test
- Benchmark
- Docs
- Dependency management

EXAMPLES

HELLO, WORLD!

```
impl EmptyState for Example {
    fn on_start(&mut self, _: StateData<()>) {
        println!("Begin!");
    }

    fn on_stop(&mut self, _: StateData<()>) {
        println!("End!");
    }

    fn update(&mut self, _: StateData<()>) -> EmptyTrans {
        println!("Hello from Amethyst!");
        Trans::Quit
    }
}
```

RUST OBJECT NOTATION (RON)

```
Scene( // class name is optional
    materials: { // this is a map
        "metal": (
            reflectivity: 1.0,
        ),
        "plastic": (
            reflectivity: 0.5,
        ),
    },
    entities: [ // this is an array
        (
            name: "hero",
            material: "metal",
        ),
        (
            name: "monster",
            material: "plastic",
        ),
    ],
)
```

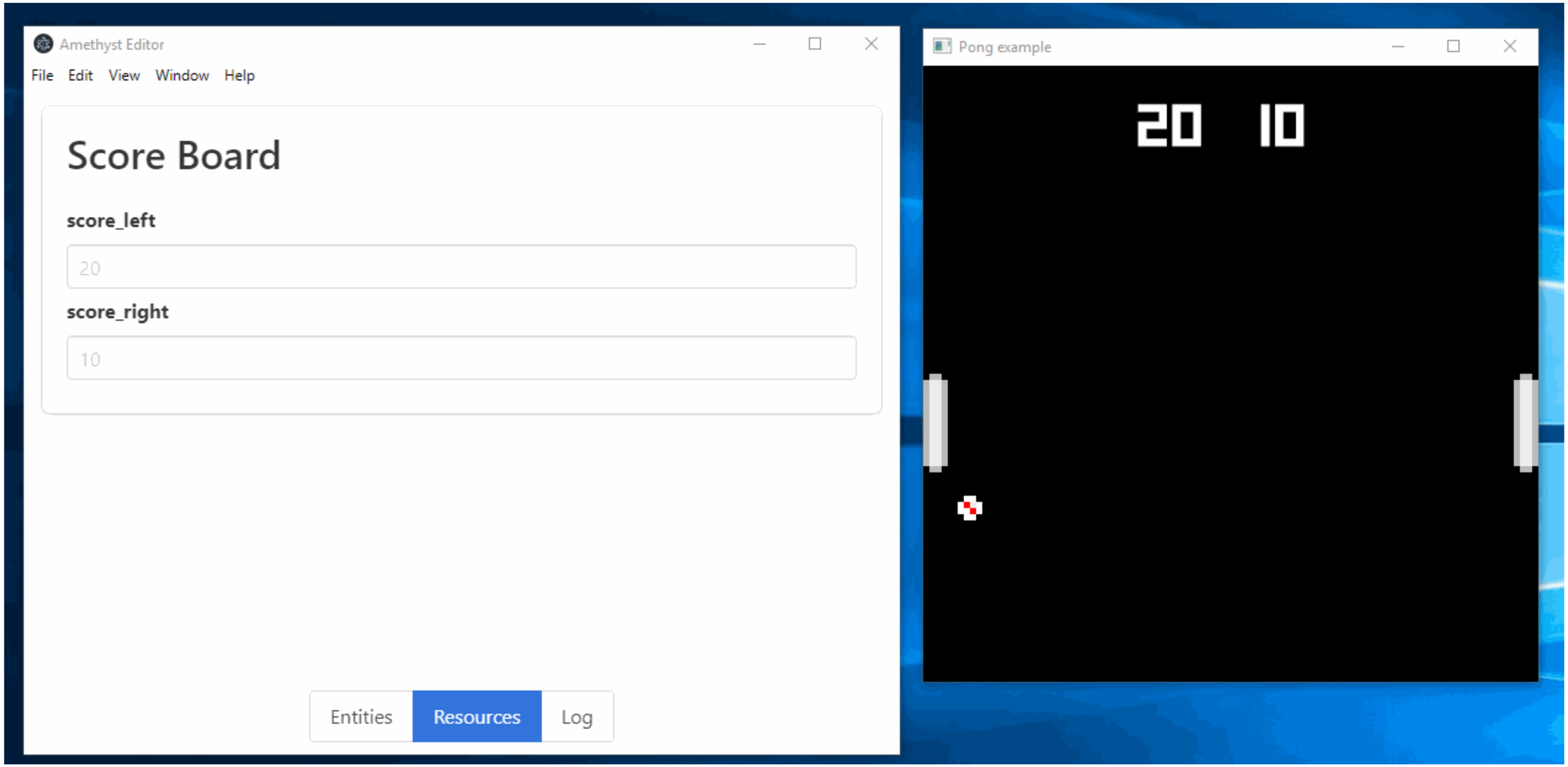
ECS: COMPONENTS

```
struct Vel(f32);

impl Component for Vel {
    type Storage = VecStorage<Self>;
}


#[derive(Component, Debug)]

#[storage(DenseVecStorage)]

struct Pos(f32, f32, f32);
```

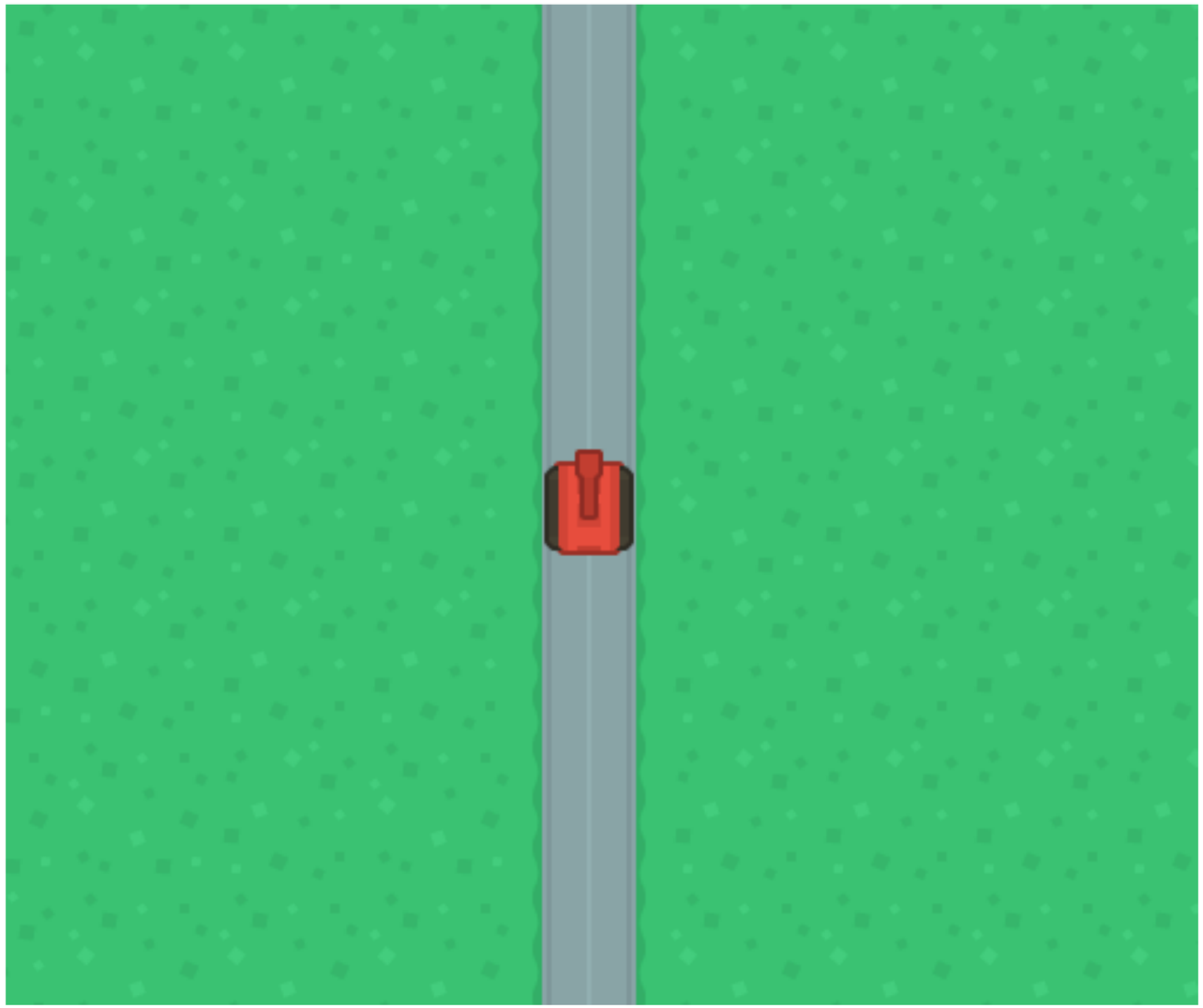
ECS: SYSTEMS

```
struct TransformSystem;

impl<'a> System<'a> for TransformSystem {
    type SystemData = (WriteStorage<'a, Pos>, ReadStorage<'a, Vel>);


    fn run(&mut self, (mut pos, vel): Self::SystemData) {
        // The `.join()` combines multiple components,
        // so we only access those entities which have
        // both of them.
        for (pos, vel) in (&mut pos, &vel).join() {
            pos.0 += vel.0;
        }
    }
}
```

EDITOR


Amethyst Editor
File Edit View Window Help
Score Board
score_left
20
score_right
10
Entities
Resources
Log
Pong example
20 10

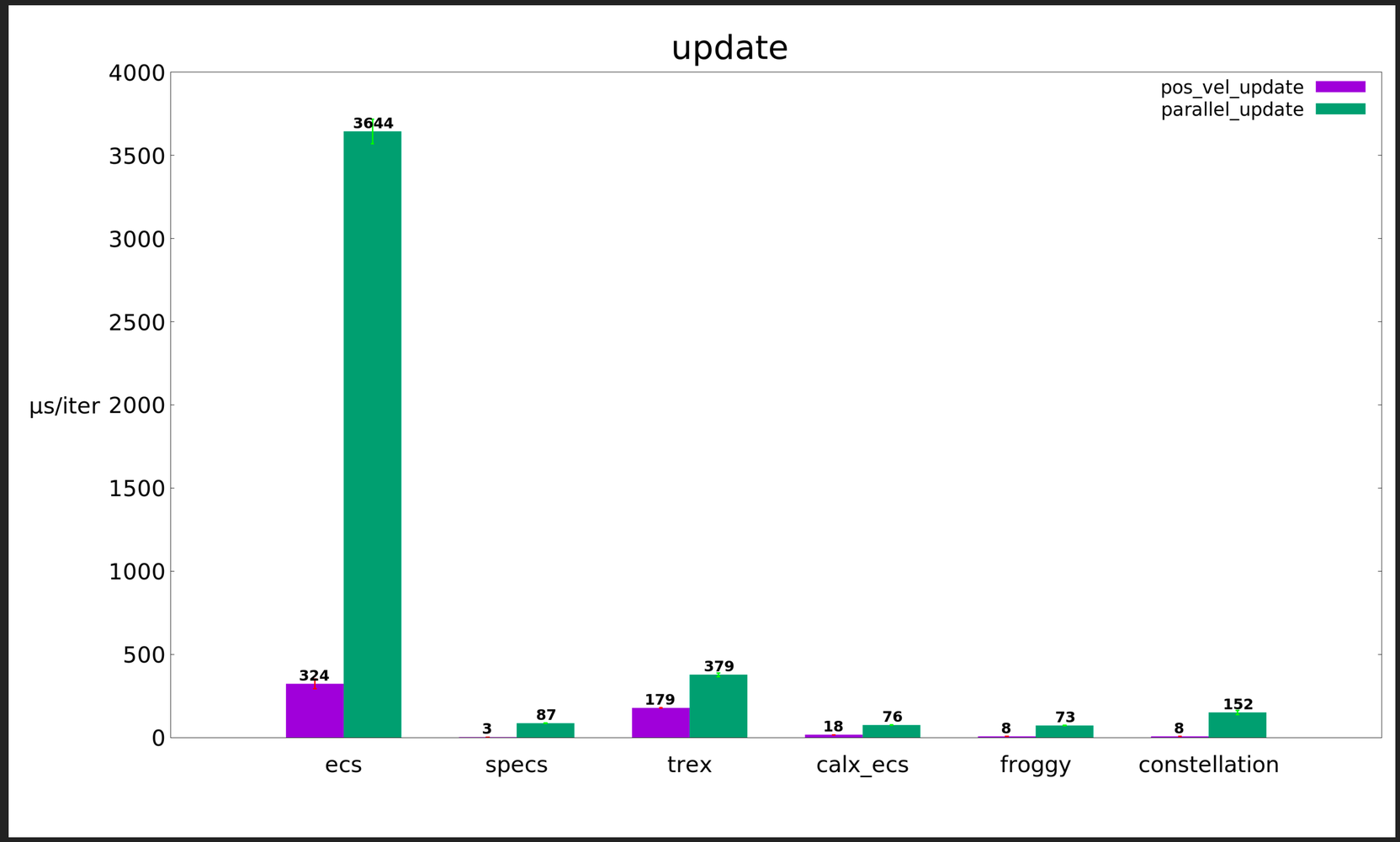
ANIMATIONS



BUNNYMARK

- cart/amethyst-bunnymark
- In can render 109,800 bunny sprites
- Godot3 max is 60,120
- This is not totally fair, but is promising
- A lot of room for improvement

ECS BENCH


update
pos_vel_update
parallel_update
µs/iter
4000
3500
3000
2500
2000
1500
1000
500
0
324
3644
3
87
179
379
18
76
8
73
8
152
ecs
specs
trex
calx_ecs
froggy
constellation

AND MORE!
COME JOIN US!

https://amethyst.rs

https://github.com/amethyst

https://discord.gg/GnP5Whs

QUESTIONS?