Automatic Scenario Generation for Testing and Training Self-driving Cars

Adrien Treuille

Zoox / Carnegie Mellon

Take 1:

Scenario Description Format

Design Constraints

- Drive all simulation modules.

Design Constraints

- Drive all simulation modules.
- Convert from real world -> synthetic data

Design Constraints

- Drive all simulation modules.
- Convert from real world -> synthetic data
- Generate data using an artist

```
entity {
  name: "hero"
  body {
    pose {
      track {
        id: 110100021,
        s: 1.5,
        t: 1.5,
      }
    }
  }
  hero_vehicle {}
}
dispatch_command {
  pose {
    track {
      id: 140100161,
      s: 0.5,
      t: -1.5,
    }
  }
  objective: PICKUP,
}
```

Scenario Definition Format

```
World
- map
   | Synthetic
```

Scenario Definition Format

```
World

- map
   | Synthetic
- entities
```

```
Entities

- body
- behavior
- type
   | Static Obstacle
   | Dynamic Obstacle
   | Hero Vechile
   | etc...
- render properties
```

Scenario Definition Format

World

```
- map
   | Synthetic
- entities
```

Entities

```
- body
- behavior
- type
   | Static Obstacle
   | Dynamic Obstacle
   | Hero Vechile
   | etc...
- render properties
```

Coordinate Systems

```
- global
- intertial
```

Behaviors

```
- stop
- move
- moveTo
```

Recap : Scenario Definition Format

Pros:

- Drive all simulation modules.
- Convert from real world -> synthetic data
- Generate data using an artist
- Use computers to generate a ton of tricky data!

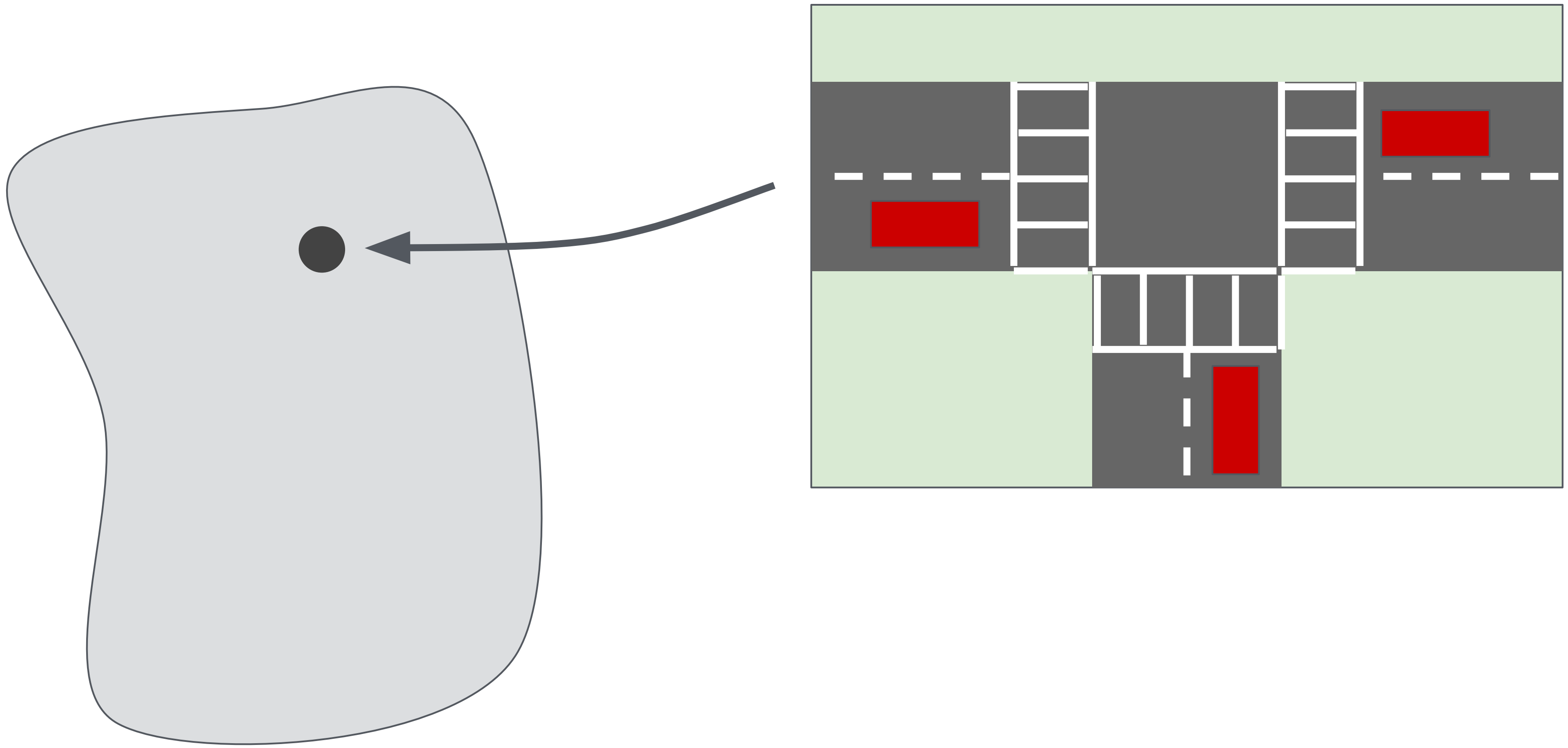
Cons:

- Creating the data is extremely time consuming
- ...**and limiting!**

Take 2:

Scenario Description Language

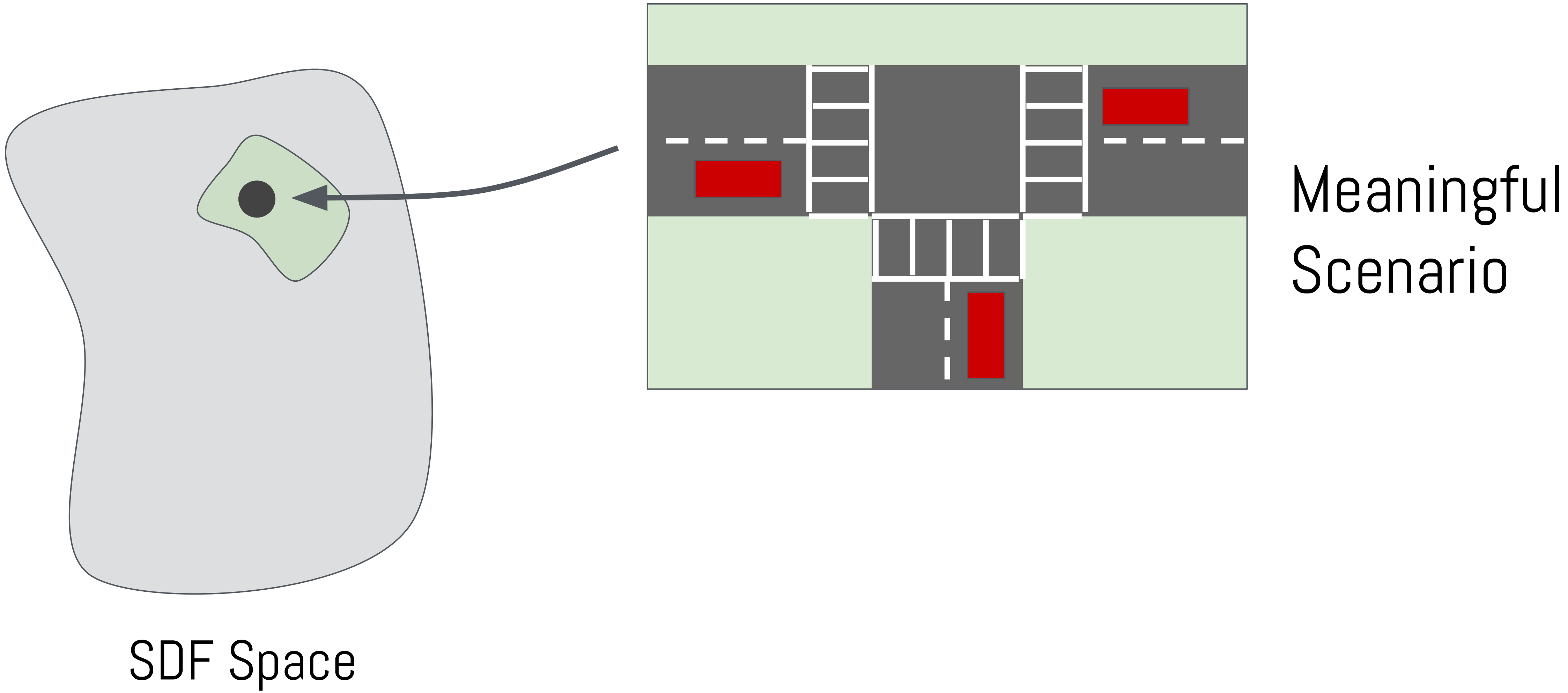
A Combinatorial Perspective



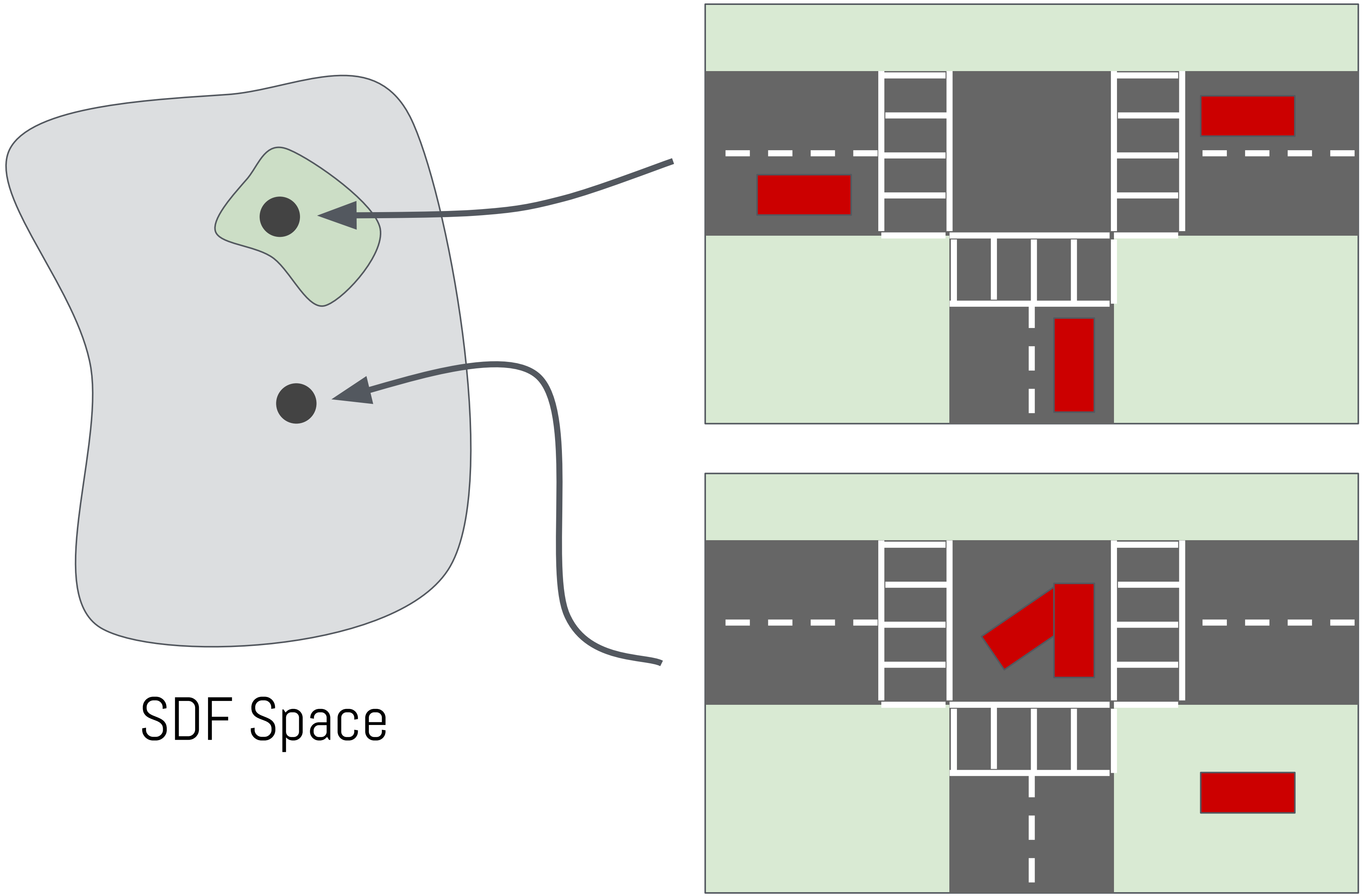
SDF Space

Meaningful Scenario

A Combinatorial Perspective



A Combinatorial Perspective

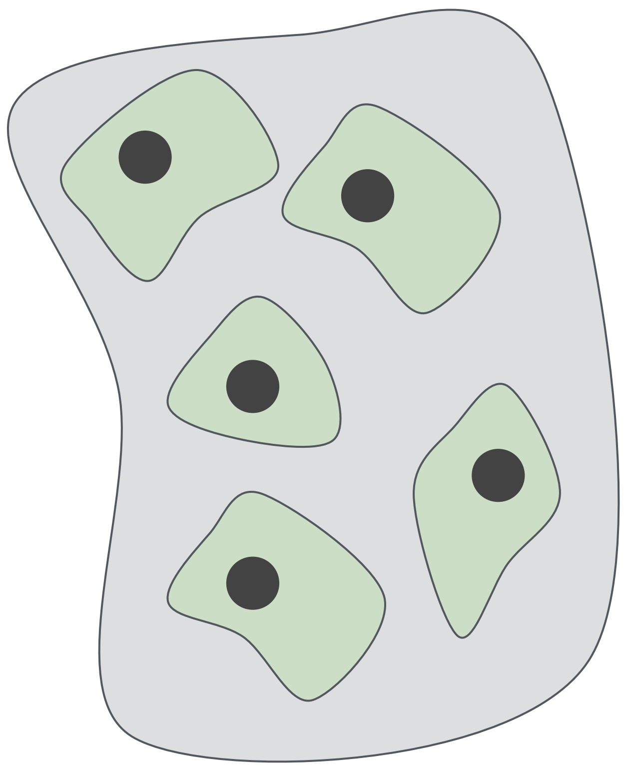


SDF Space

Meaningful Scenario

Meaningless Scenario

A Combinatorial Perspective

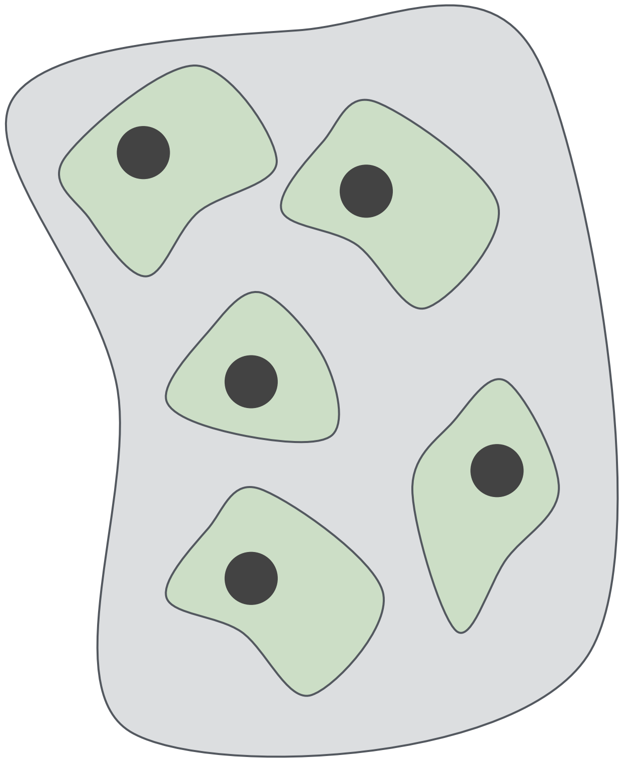


SDF Space

How can we discover (iterate over) just meaningful scenarios?

Design Constraints

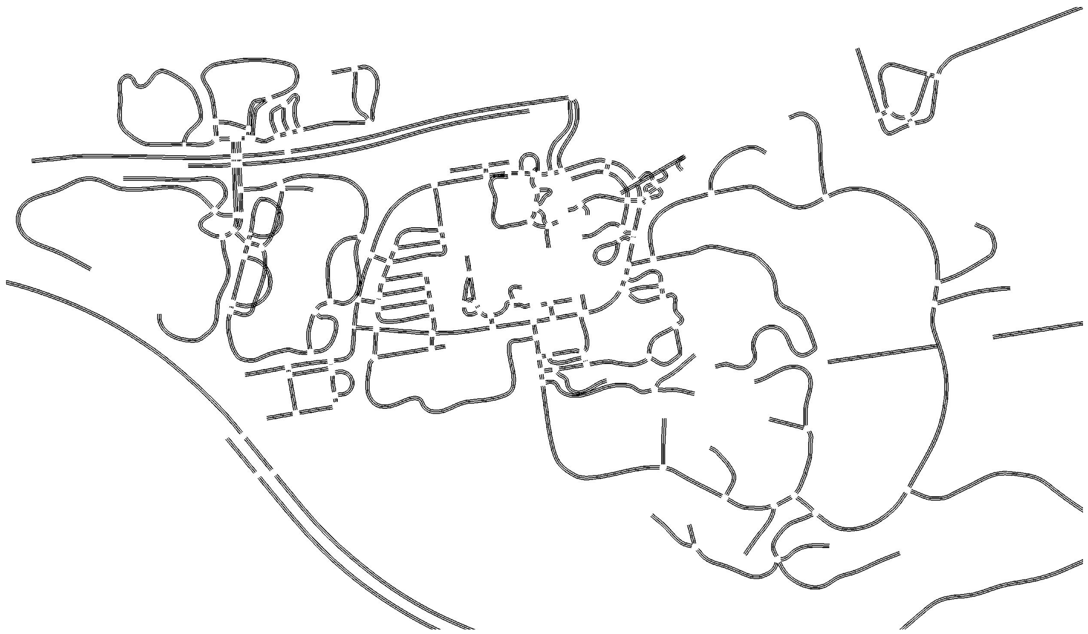
- Understandable to Product Managers / Regulators
- Compiles to SDF
- Combinatorial in Nature



SDF Space

Design Constraints

- Understandable to Product Managers / Regulators
- Compiles to SDF
- Combinatorial in Nature
- Works on real maps.



Design Constraints

- Understandable to Product Managers / Regulators
- Compiles to SDF
- Combinatorial in Nature
- Works on real maps.

(movies)

Design Constraints

- Understandable to Product Managers / Regulators
- Compiles to SDF
- Combinatorial in Nature
- Works on real maps.
- **Can be short!**

Scenario Definition Language

World

```
- map
   | Synthetic
   | Real World
- entities
```

Entities

```
- body
- behavior
- type
    | Static Obstacle
    | Dynamic Obstacle
    | Hero Vechile
    | etc...
- render properties
```

Coordinate Systems

```
- global
- intertial
```

Behaviors

```
- stop
- move
- moveTo
```

Scenario Definition Language

World

```
- map
   | Synthetic
   | Real World
- entities
```

Entities

```
- body
- behavior
- type
   | Static Obstacle
   | Dynamic Obstacle
   | Hero Vechile
   | etc...
- render properties
```

Coordinate Systems

```
- global
- intertial
```

Behaviors

```
- stop
- move
- moveTo
- follow_road
- follow_entity
```

Scenario Definition Language

World

```
- map
   | Synthetic
   | Real World
- entities
```

Linear Temporal Logic

- **G** : always (globally)
- **F** : in the future
- **R** : for release
- **X** : next
- **U** : until

Entities

```
- body
- behavior
- type
    | Static Obstacle
    | Dynamic Obstacle
    | Hero Vechile
    | etc...
- render properties
```

Conditions

```
- distance(X,Y) < D
- in_region(X,R)
- speed(X) > S
- speed(X) < S
```

Coordinate Systems

```
- global
- intertial
```

Behaviors

```
- stop
- move
- moveTo
- follow_road
- follow_entity
```

Scenario Definition Language

World

- map
  | Synthetic
  | Real World
- entities
- outer_products

Linear Temporal Logic

- **G** : always (globally)
- **F** : in the future
- **R** : for release
- **X** : next
- **U** : until

Entities

- body
- behavior
- type
  | Static Obstacle
  | Dynamic Obstacle
  | Hero Vechile
  | etc...
- render properties

Conditions

- distance(X,Y) < D
- in_region(X,R)
- speed(X) > S
- speed(X) < S

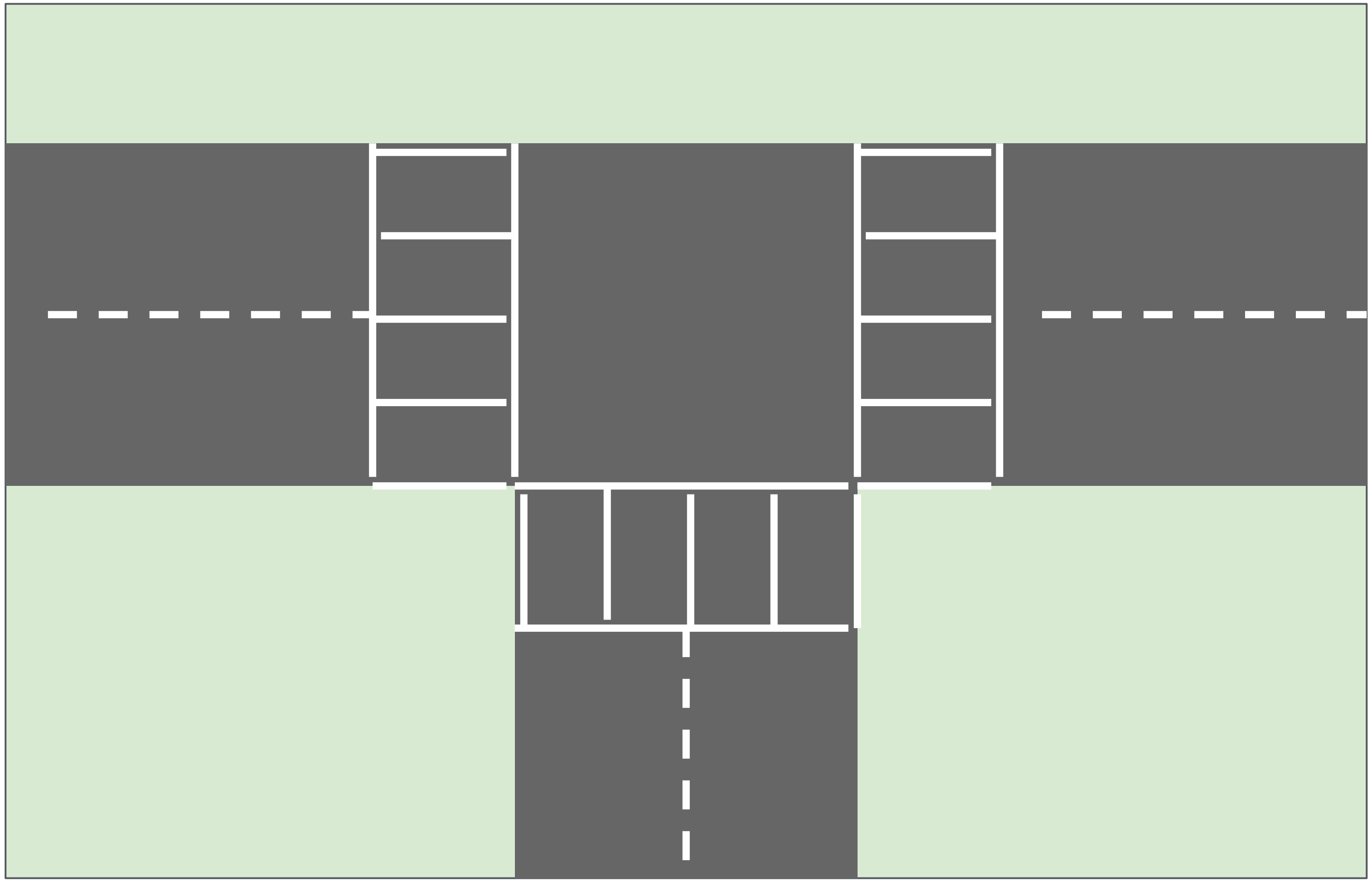
Coordinate Systems

- global
- intertial
- topological

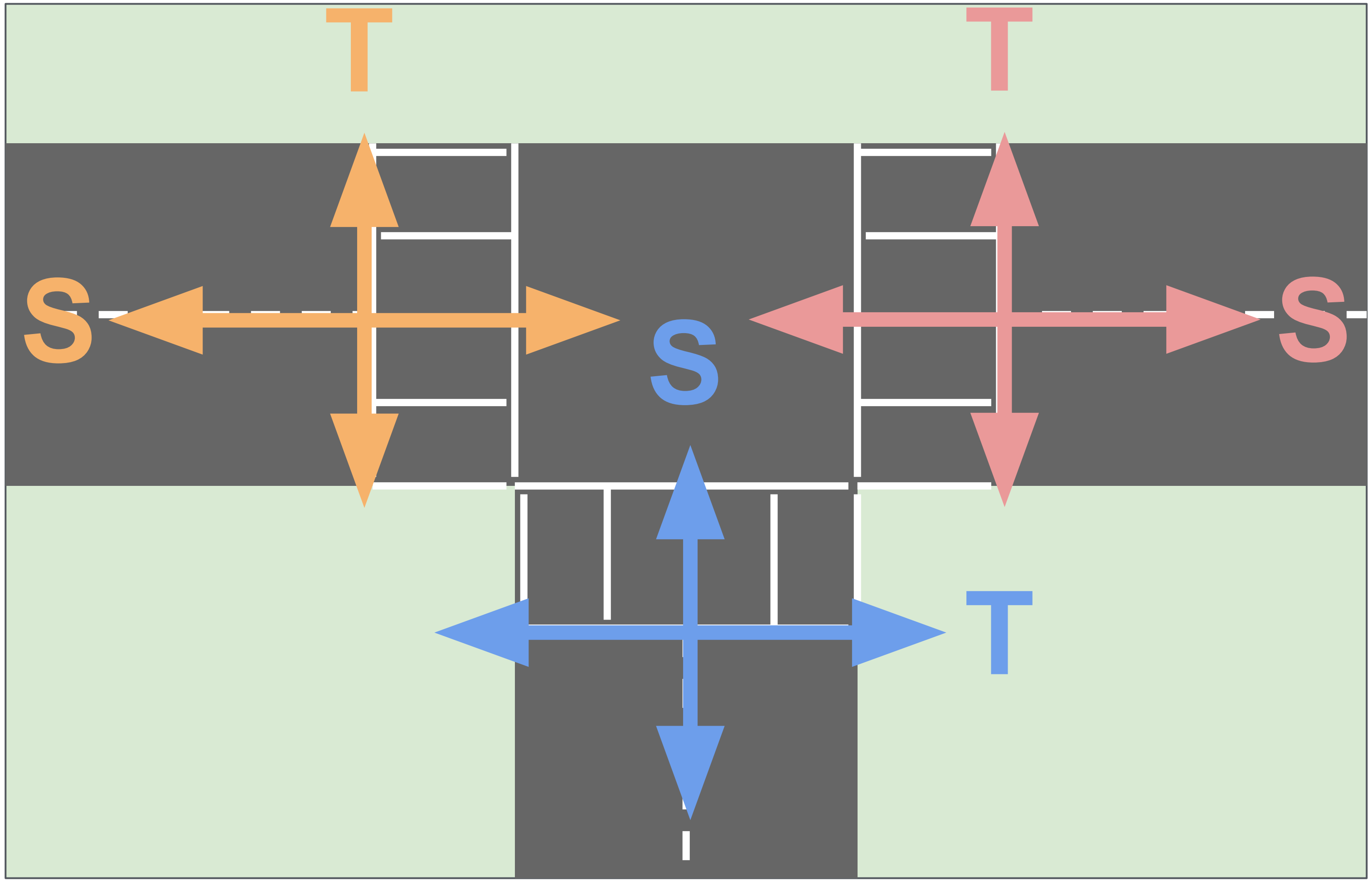
Behaviors

- stop
- move
- moveTo
- follow_road
- follow_entity

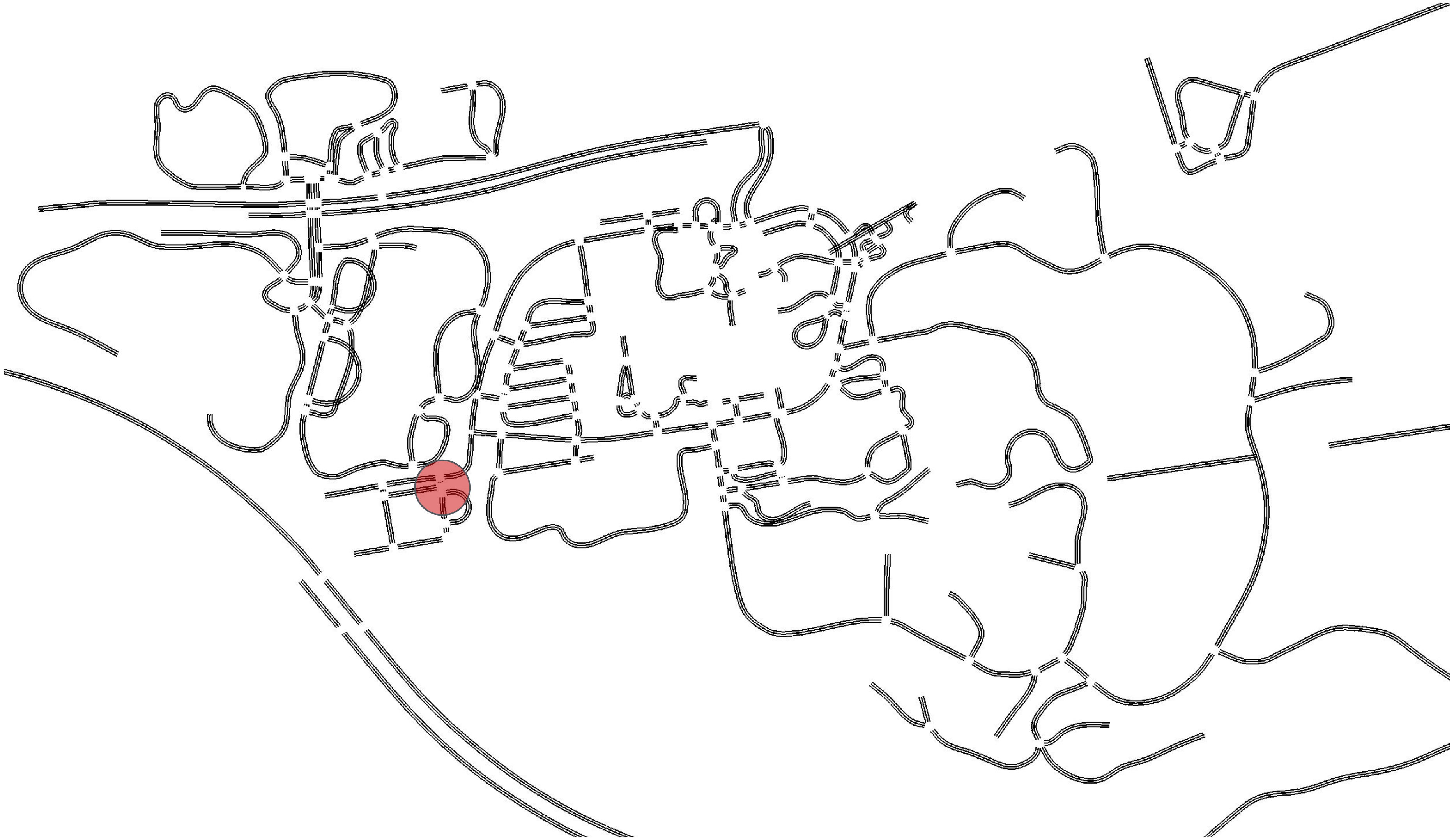
Topological Coordinate Systems



Topological Coordinate Systems



Outer Products



Outer Products

Outer Products

Scenario Definition Language

World

- map
  | Synthetic
  | Real World
- entities
- outer_products

Linear Temporal Logic

- **G** : always (globally)
- **F** : in the future
- **R** : for release
- **X** : next
- **U** : until

Entities

- body
- behavior
- type
  | Static Obstacle
  | Dynamic Obstacle
  | Hero Vechile
  | etc...
- render properties

Conditions

- distance(X,Y) < D
- in_region(X,R)
- speed(X) > S
- speed(X) < S

Coordinate Systems

- global
- intertial
- topological

Behaviors

- stop
- move
- moveTo
- follow_road
- follow_entity

Scenario Definition Language

Still a very small language!

Scenario Definition Format

World

```
- map
    | Synthetic
    | Real World
- entities
```

Entities

```
- body
- behavior
- type
    | Static Obstacle
    | Dynamic Obstacle
    | Hero Vechile
    | etc...
- render properties
```

Coordinate Systems

```
- global
- intertial
```

Behaviors

```
- stop
- move
- moveTo
```

Scenario Definition Language

World

- map
  | Synthetic
  | Real World
- entities
- outer_products

Linear Temporal Logic

- **G** : always (globally)
- **F** : in the future
- **R** : for release
- **X** : next
- **U** : until

Entities

- body
- behavior
- type
  | Static Obstacle
  | Dynamic Obstacle
  | Hero Vechile
  | etc...
- render properties

Conditions

- distance(X,Y) < D
- in_region(X,R)
- speed(X) > S
- speed(X) < S

Coordinate Systems

- global
- intertial
- topological

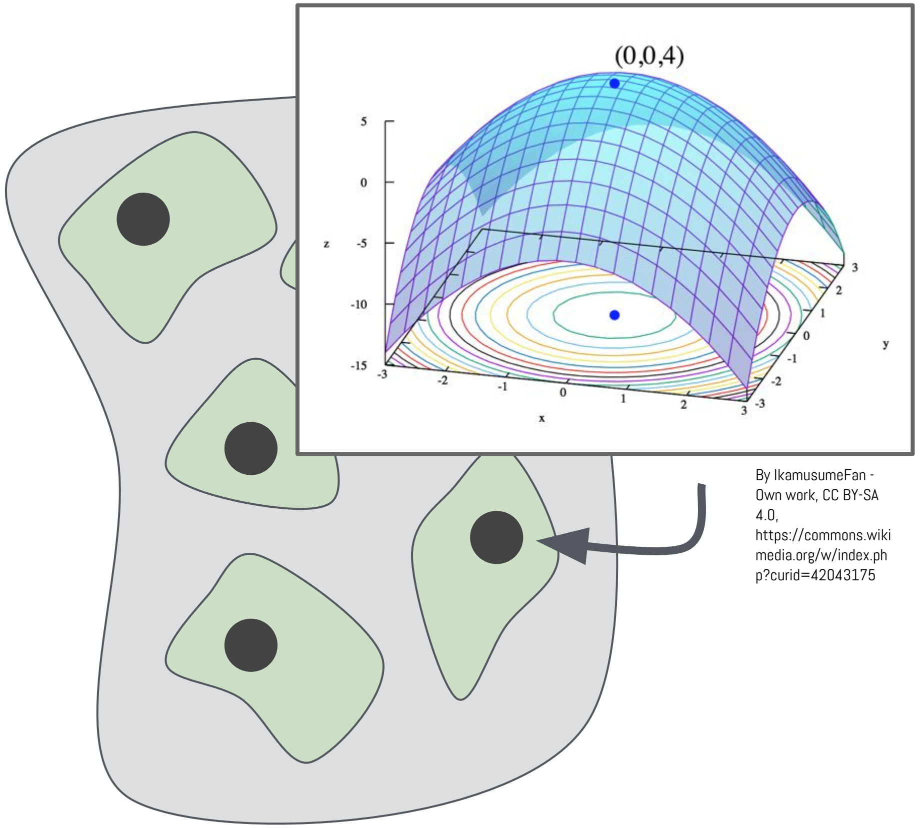
Behaviors

- stop
- move
- moveTo
- follow_road
- follow_entity

Future Work

Future Directions

- Optimization over scenario types.



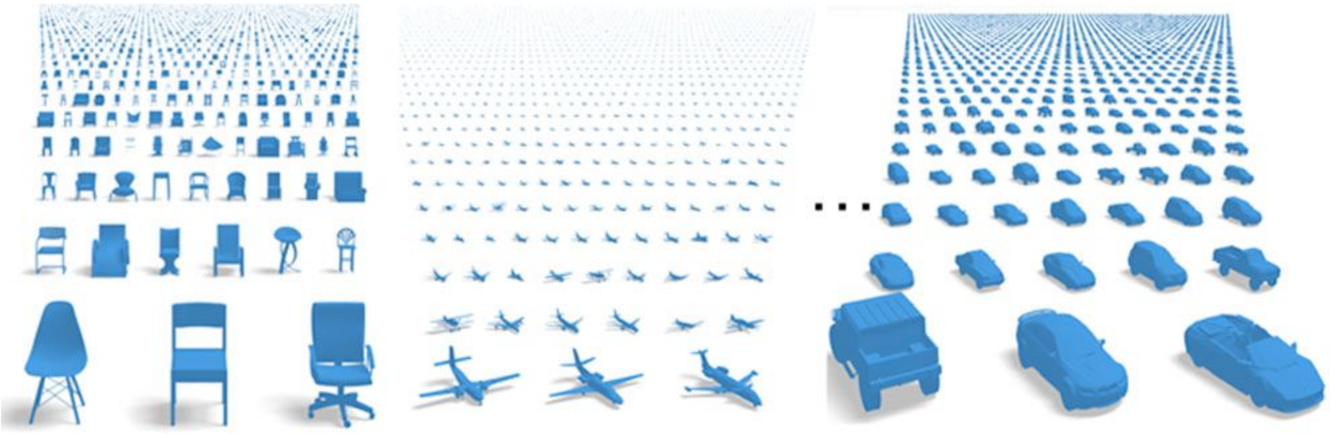
By IkamusumeFan - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=42043175

Future Directions

- Optimization over scenario types.
- Precisely characterize the statistical realism of the scene relative to real data

Future Directions

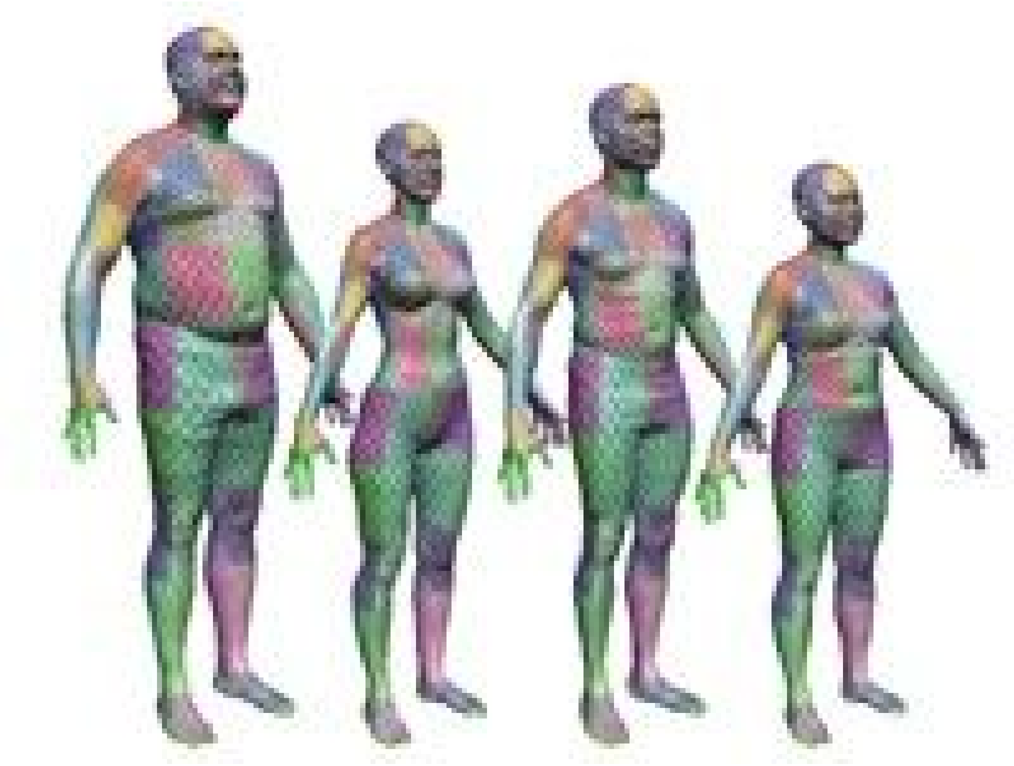
- Optimization over scenario types.
- Precisely characterize the statistical realism of the scene relative to real data
- Big-data geometry creation for maps



[1] Chang et al., ShapeNet: An Information-Rich 3D Model Repository arXiv:1512.03012

Future Directions

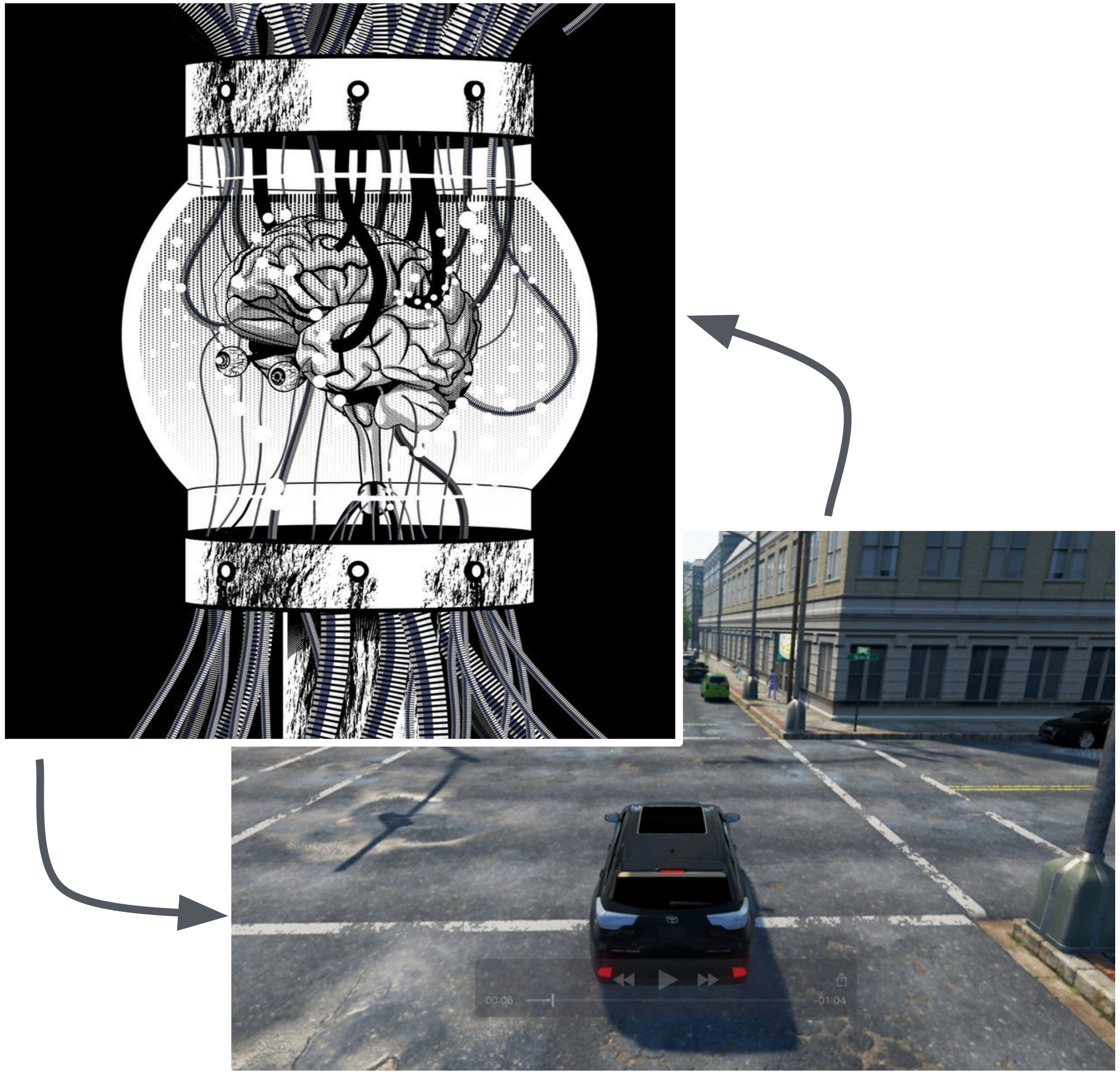
- Optimization over scenario types.
- Precisely characterize the statistical realism of the scene relative to real data
- Big-data geometry creation for maps
- Studying various kinds of variation



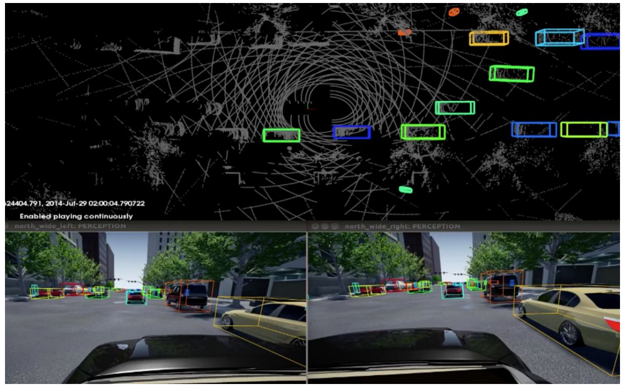
Allen, et al. SIGGRAPH 2003

Future Directions

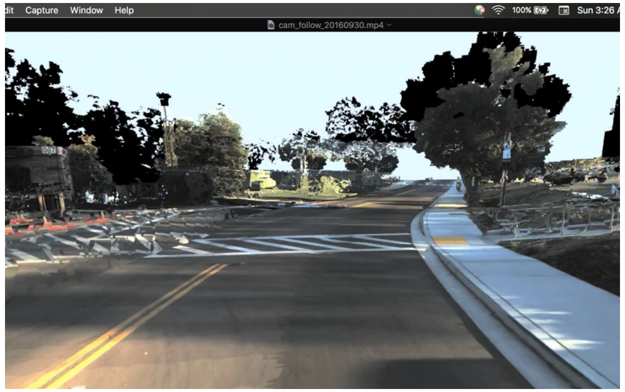
- Optimization over scenario types.
- Precisely characterize the statistical realism of the scene relative to real data
- Big-data geometry creation for maps
- Studying various kinds of variation
- Using neural nets to validate the accuracy of data simulation



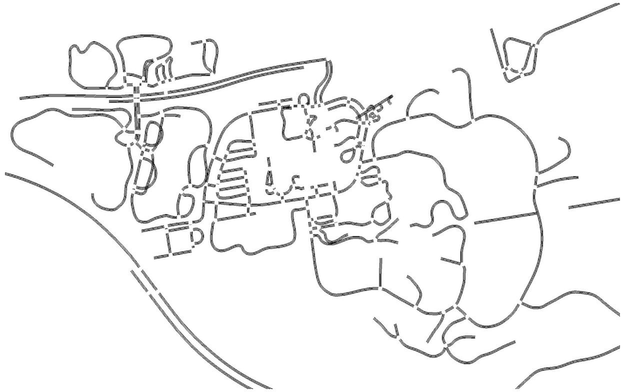
In Short...



Multi-modal synthesis



Captured Camera Data



Thousands of Scenarios

Thank you!