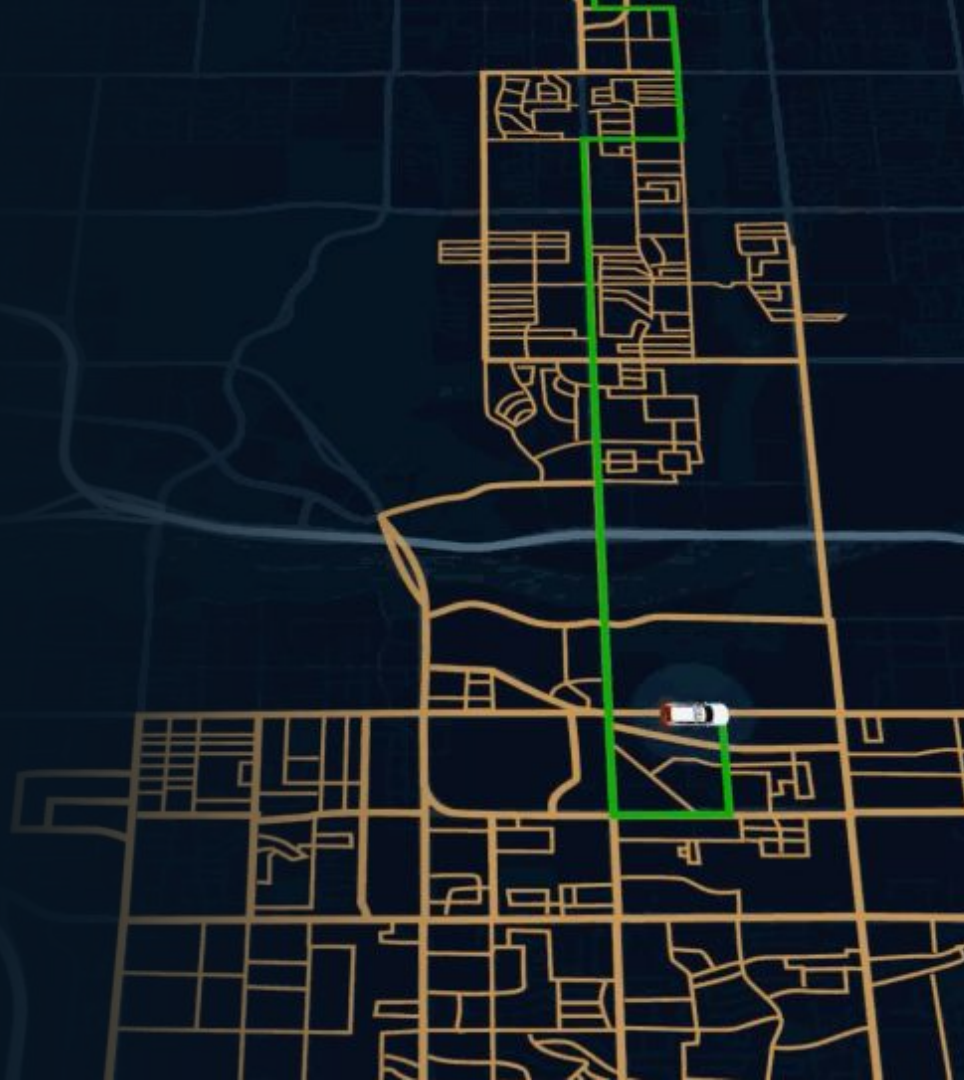
Lustre at Uber ATG

Nick Cobb and Jinshan Xiong
ATG Infrastructure

ATG Mission

Self-driving transportation for everyone and everything.



Why Self-Driving?

Self-driving matters for **the world**

Save lives. Save time. Save space.

Self-driving matters for **Uber**

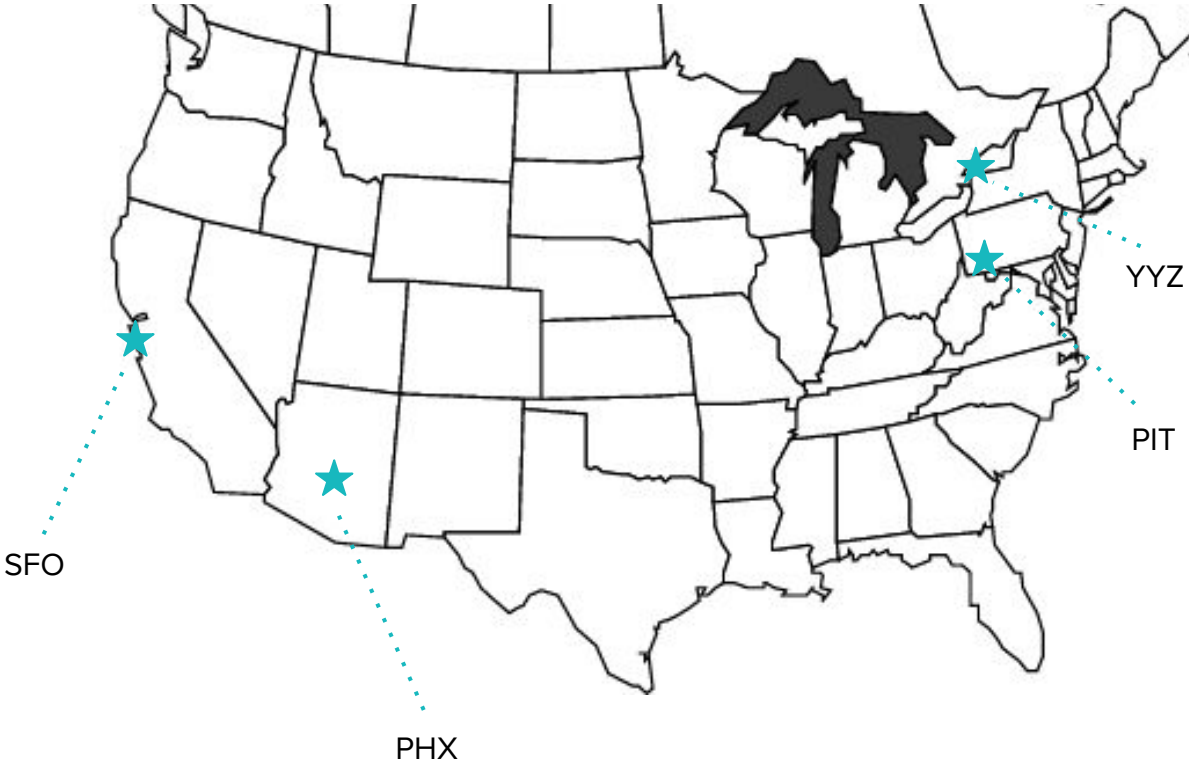
Providing safe, reliable, cost effective transportation is our priority.

Uber matters to **self-driving**

Our network allows us to scale self-driving globally.

Sites

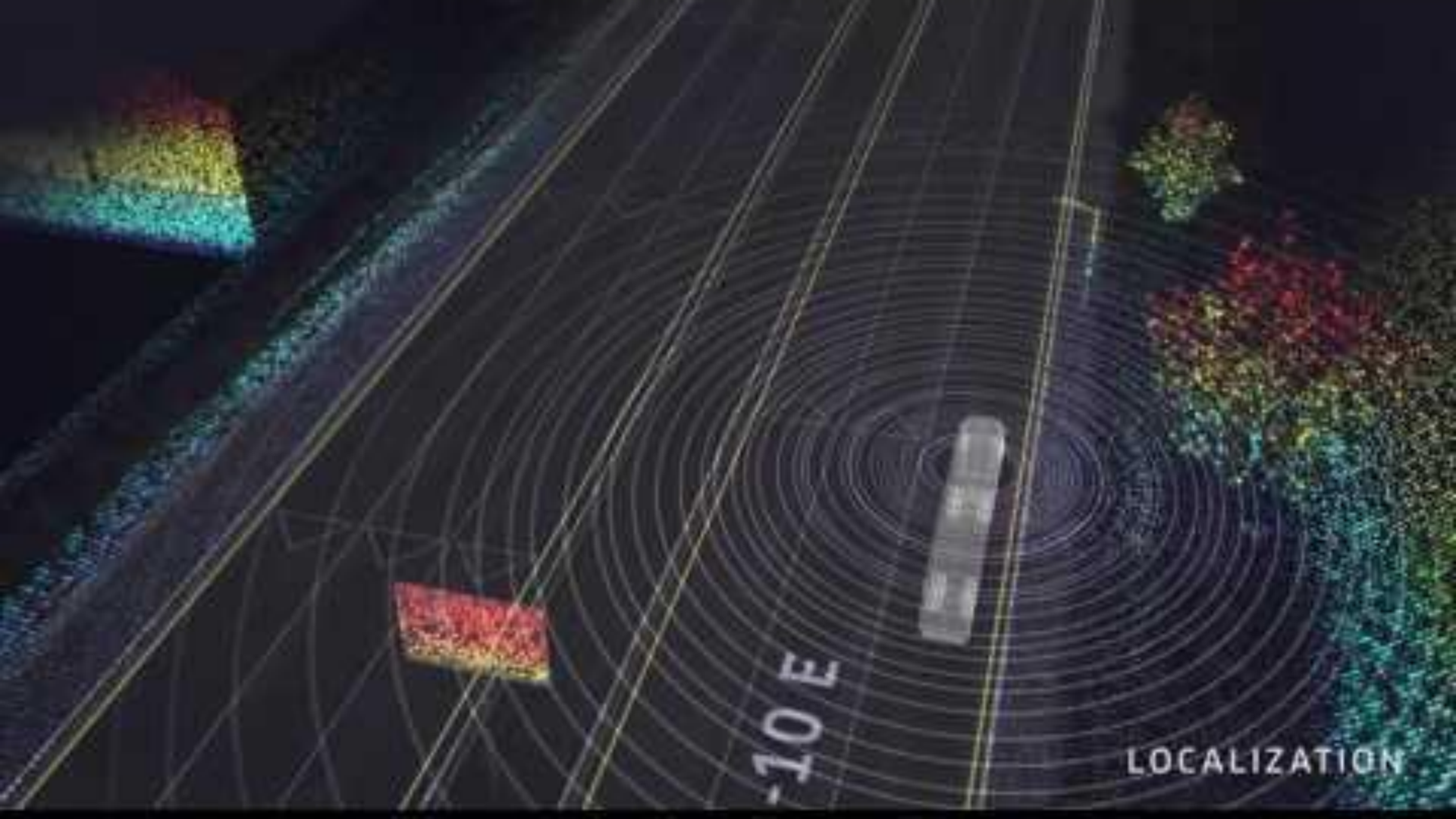
1500+ Total employees



Team Overview

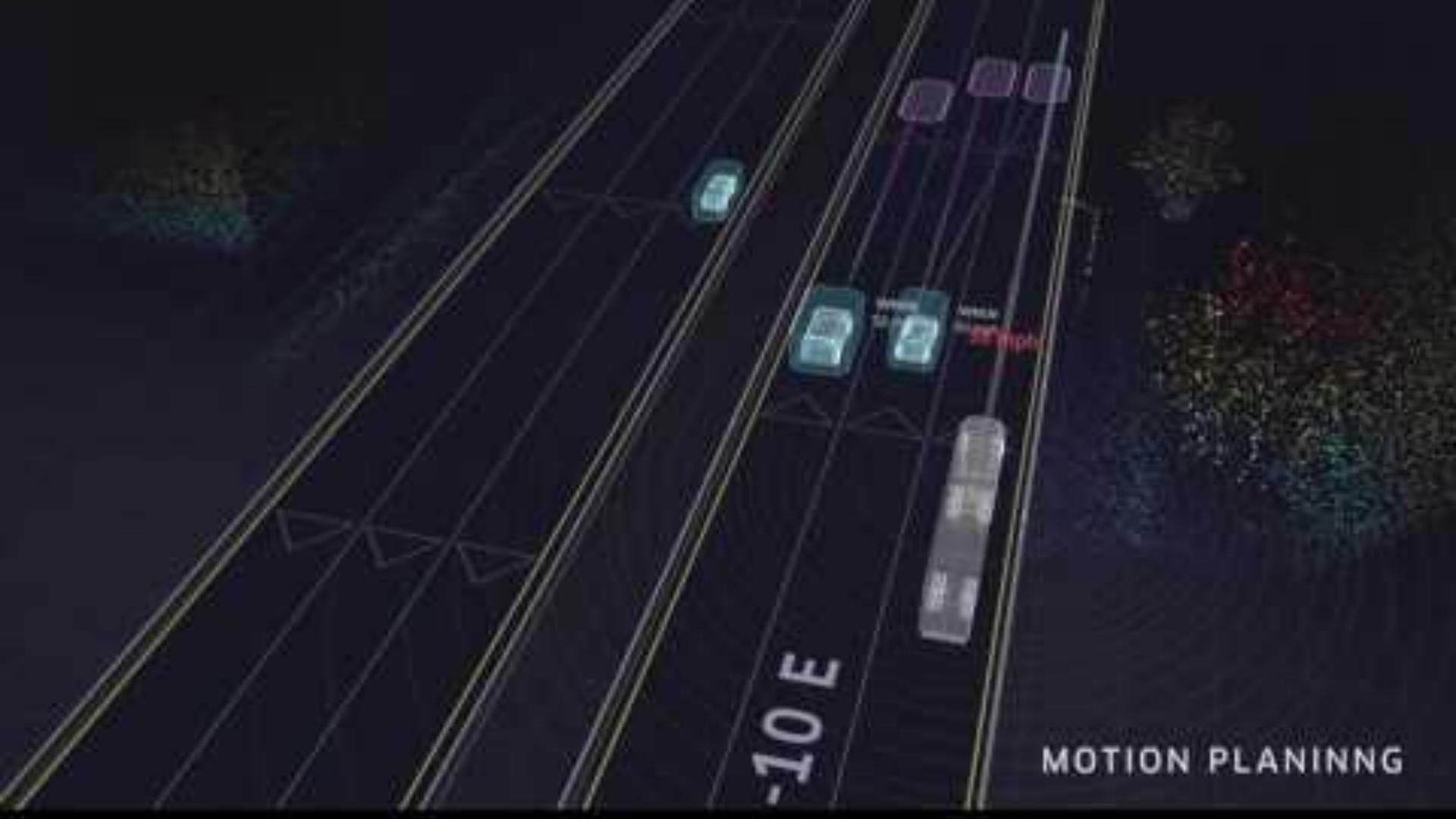
Software	Building a scalable self-driving system for cars and trucks through unique functional groups
Mapping	Collecting and utilizing real-world mapping data via high density maps and Uber Maps
Hardware	Designing, prototyping, and integrating hardware into OEM vehicles that can be produced at scale
Vehicle Programs	Building relationships with the world's top OEMs and Tier 1 suppliers to partner in self-driving innovation and integrate with ATG technology
Operations	Maximizing self-driving vehicle utilization and learnings through real-world testing and passenger operations
Offline Testing	Testing the software stack using real-world and test scenarios
Safety	Define and improve better than human performance from Self-Driving Vehicles

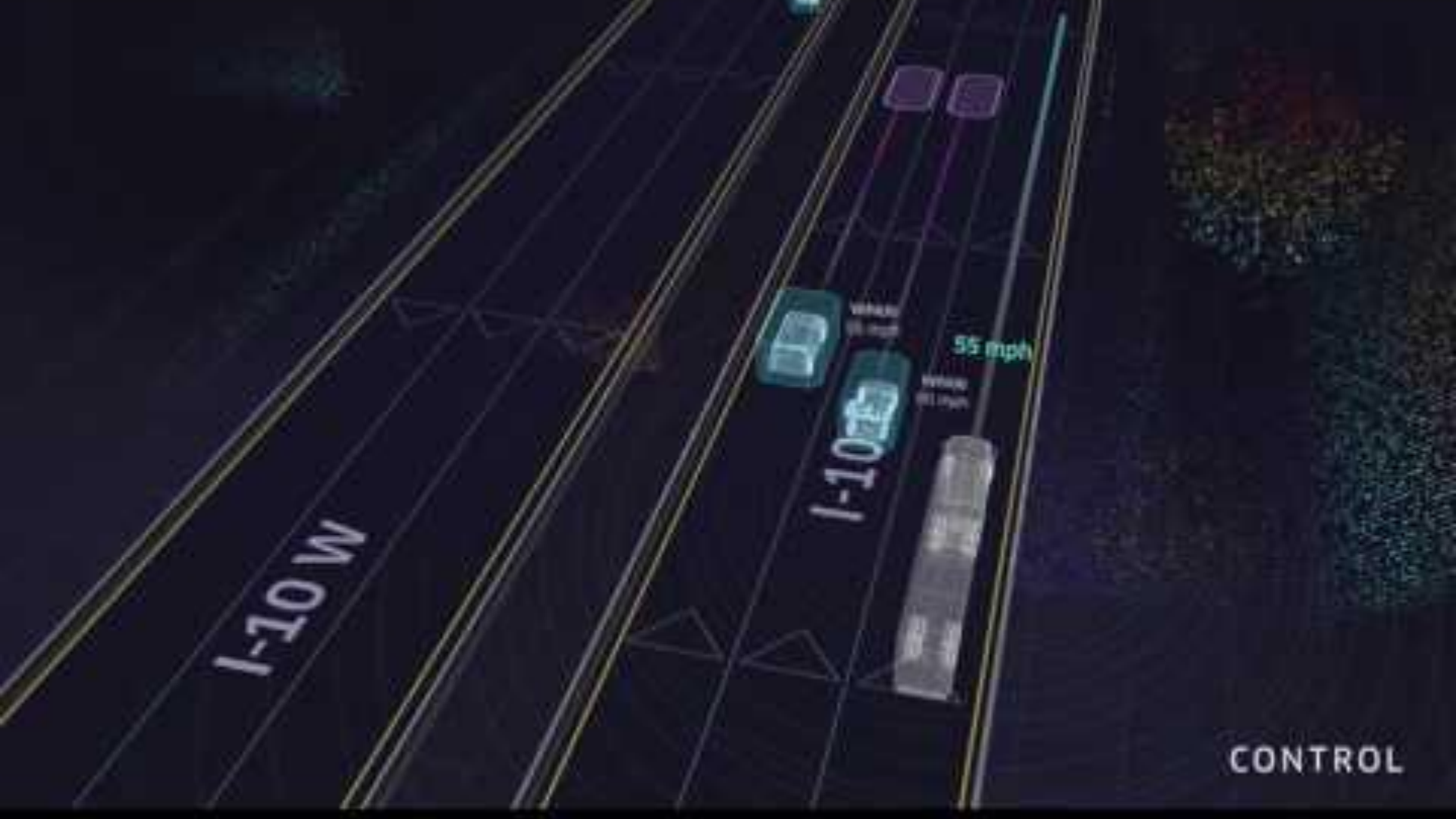

-10E
MAP


LOCALIZATION


LIDAR
CAMERA
RADAR
67 mph
55 mph
PERCEPTION


I-10 W
I-10 E
PREDICTION


-10 E
MOTION PLANINNG


I-10 W
I-10
55 mph
CONTROL


Almono

Software teams and functions

- Every ATG team uses or analyzes vehicle data
- Data storage
- Lustre at ATG
    - Why Lustre?
    - Workloads
    - Architecture
    - Findings
    - Recommendations

Lustre

Looking for new file system to replace NAS

- ”Our traditional NAS was reaching its breaking point”
- “Evaluated Ceph, GlusterFS, Lustre, and pNFS, but Lustre was the best fit for out needs”
    - “Provide POSIX semantics”
    - “Achieve very high I/O rates”
    - ”Hold well under heavy load”
- Jobs are eventually spending more time on processing data than waiting on I/O

Why was Lustre chosen in the first place?

File sizes are variable:

- Logs are typically hundreds of gigabytes
  - Data collected by vehicles on the road
  - Immutable after written, read by most of jobs
- Executable binaries
  - Software releases with 50,000 files and multi GB in total size
- Tiny files, extracted from AV logs for machine learning
  - Millions of files, most of them are several KB
- Good thing: all of them are written once and read-only afterwards

Workload at Uber

Lustre doesn’t support small I/O well:

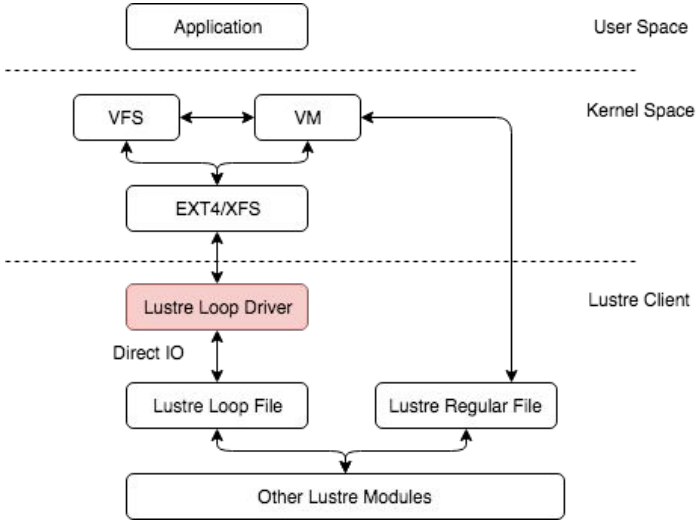
- Big files are stored into Lustre
  - Works extremely well because Lustre provides superb I/O bandwidth
- Small files are stored into NAS and exported by NFS
  - It’s become a bottleneck in job pipeline
    - Thousands of computer nodes need to read data from a single node to launch a job
  - Look for solutions to migrate small files into Lustre

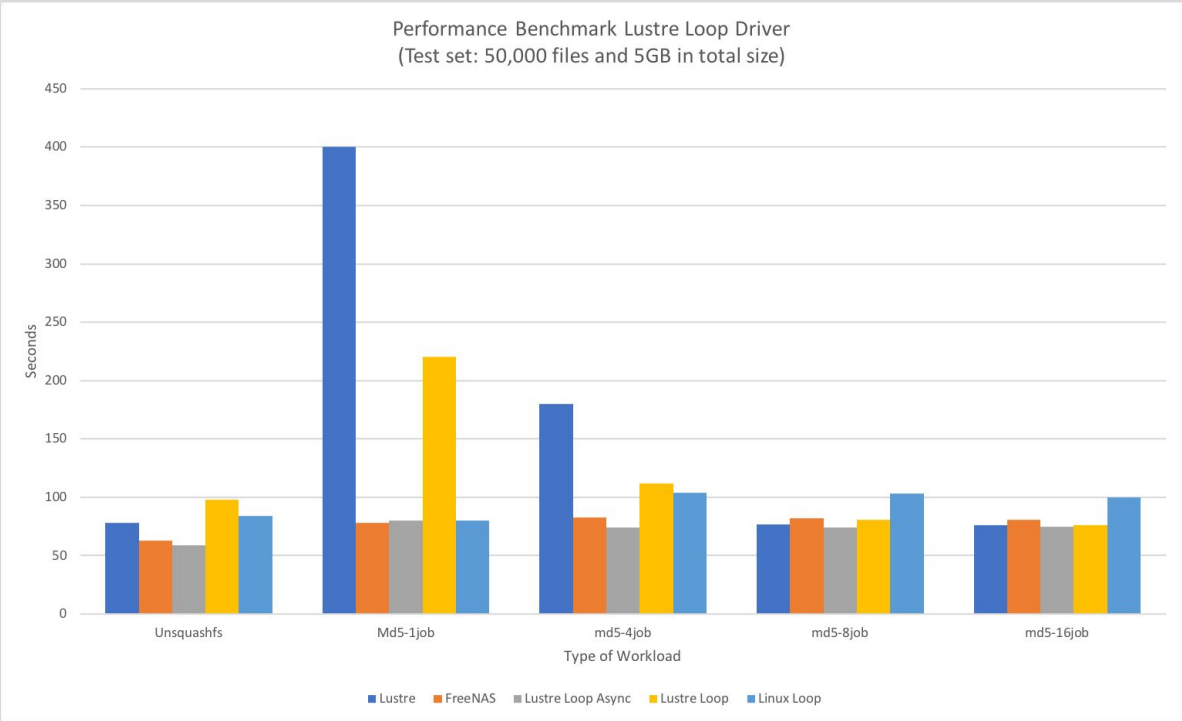
Storage Architecture

Solution for Small Files

Enhanced version of LLITE loop device:

- Revive Lustre loop device
  - It was removed from Lustre 2.7
  - It turned out Linux loop can only support 512 sector size
- Enhanced it
  - Direct + async I/O boost performance
- Ideal solution for Write-Once-Read-Most workload
- No expensive Lustre Open/Close RPC for small files





- Test set: 50,000 files, 5GB in total
- Squashfs for writing test
- Md5sum for reading test:

  find <dir> -type f | parallel -j <job> md5sum

LLITE Loop Performance

Features Needed:

- Data on MDT
  - Improve performance for small files
  - Must do read on open
- File Level Redundancy
  - Distribute read workload across multiple servers
  - Address the problem of 3000 nodes to access the same file at the same time
- Compound RPC
  - Multiple small RPCs should be combined into a large RPC on the import level

Enable AI on Lustre