Calibrate good times! The tools and methods to get top-quality robot data.

Roland Meertens
Sam Pfeiffer

Why monitor data quality?

- At each company we worked we found a LOT of issues with the data
- Everyone wants to collect data, but the data should also be good from the start
- For machine learning purposes your data should be good and consistent
- Vibe coding is not going to solve your problems, taking a good look at your data will!
- This presentation contains a subset of common issues!

About us

Sam Pfeiffer, PhD

Robotics Software Engineer, playing with robots for over 15 years with robots of all kinds. But my favorite are humanoid robots!

Fun fact

Love climbing, haven't climbed a skyscraper!

Follow him on:

github.com/awesomebytes

Senior Robotics Engineer @ Humanoid

sammypfeiffer@gmail.com

Roland Meertens

Always working on robotics and machine learning projects! Currently working on end-to-end learning for self-driving cars

Fun fact

Takes photos of foxes in his backyard! Follow my fox on Instagram @Maple_and_her_friends

Follow him on:

LinkedIn!

Tech Lead @ Wayve

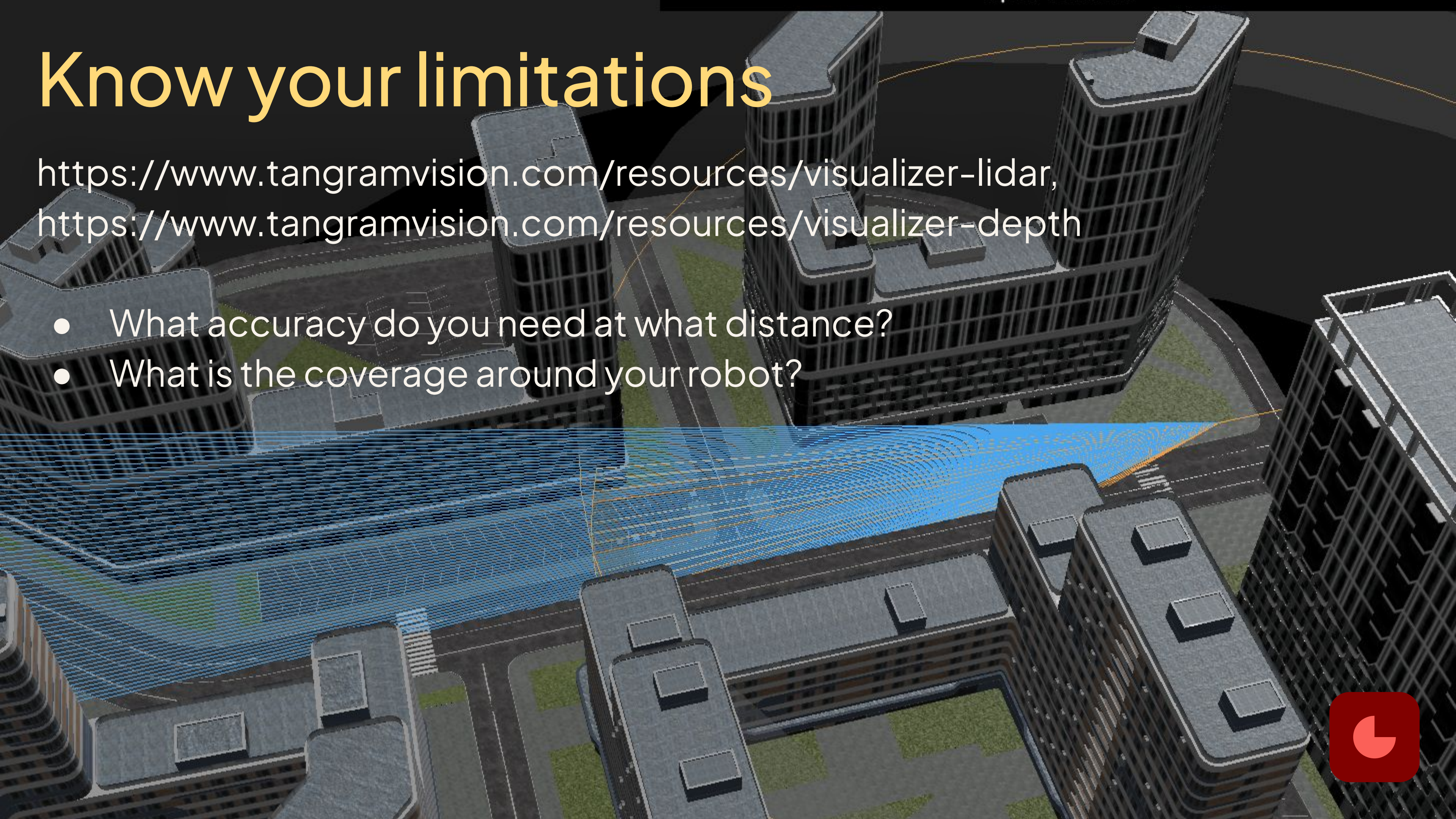
rolandmeertens@gmail.com

Research and plan your sensors

Know your limitations

https://www.tangramvision.com/resources/visualizer-lidar,
https://www.tangramvision.com/resources/visualizer-depth

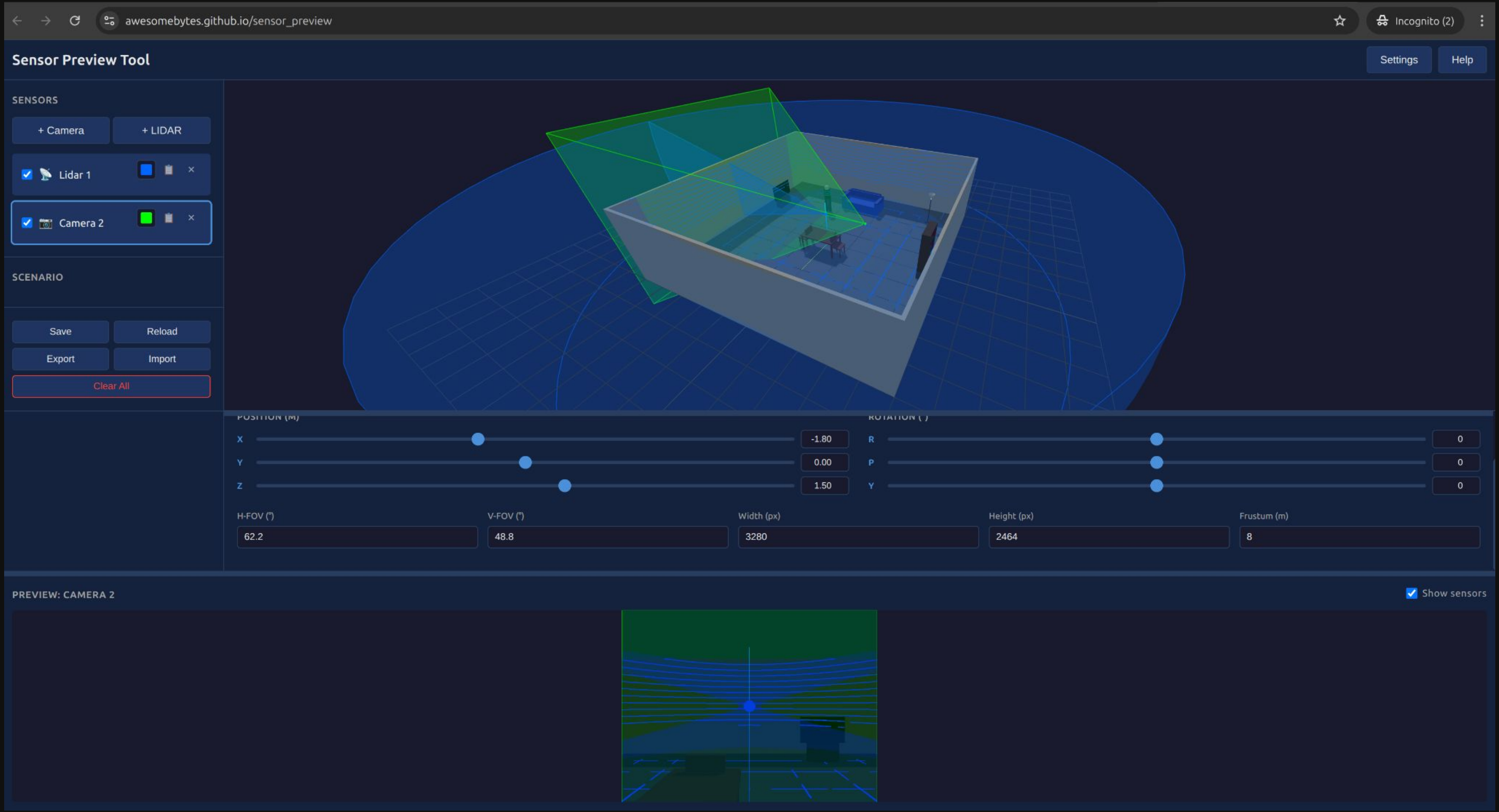
- What accuracy do you need at what distance?
- What is the coverage around your robot?



It’s not only about the sensor

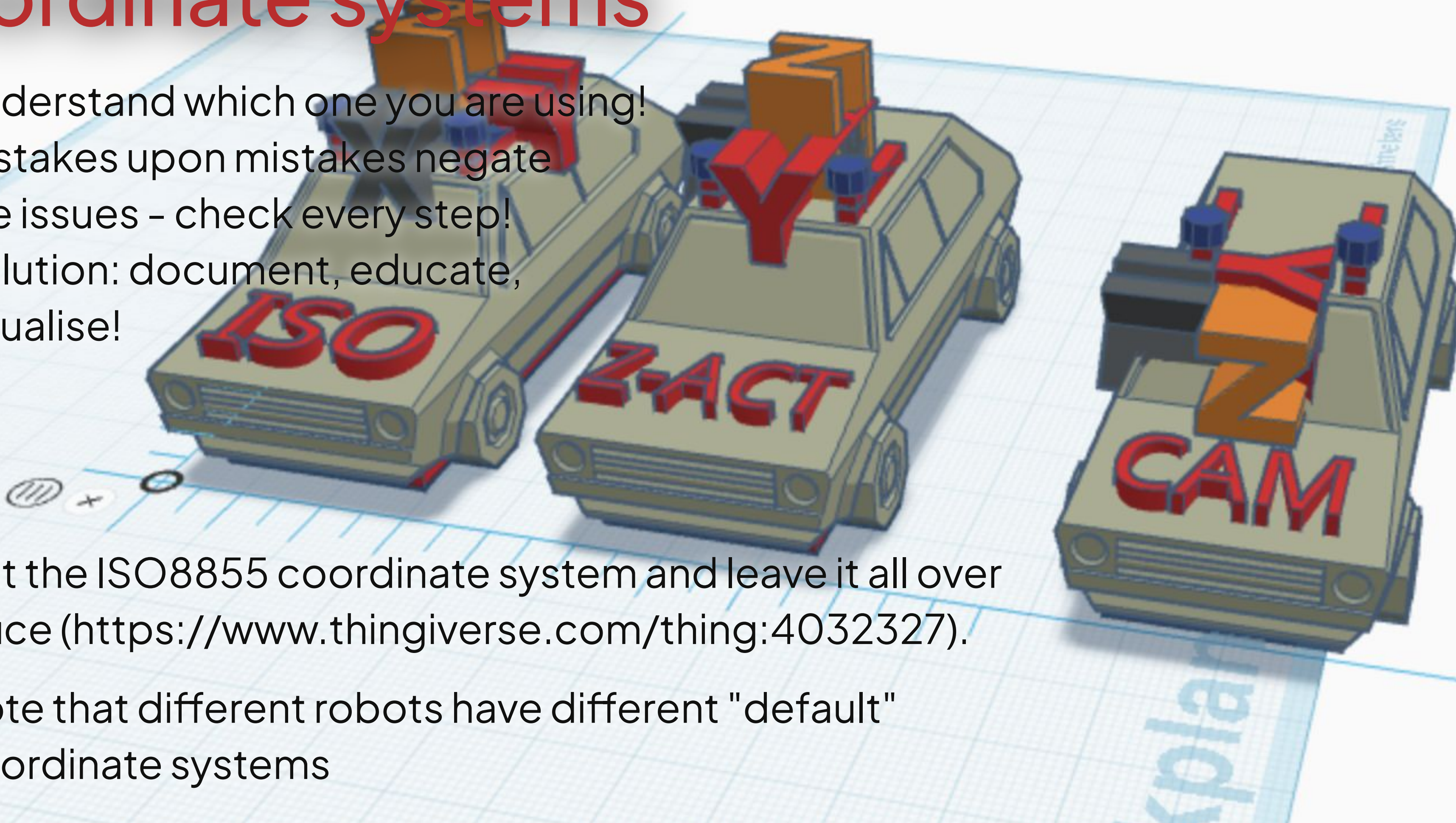
- Interfaces
  - USB: can be quite problematic (flakiness of connection, bandwidth limits)
  - Ethernet: implies the network stack (extra CPU usage and delay)
  - MIPI CSI: implies dedicated hardware (short cabling)
  - GSML: very expensive hardware
  - CAN: low bandwidth, CPU usage
- Driver quality
  - Is it open source?
  - Is it efficient?
- Data format
  - Usable out of the box?

Where to place the sensors



Coordinate systems

- Understand which one you are using!
- Mistakes upon mistakes negate the issues - check every step!
- Solution: document, educate, visualise!



3D Print the ISO8855 coordinate system and leave it all over the office (https://www.thingiverse.com/thing:4032327).

- Note that different robots have different "default" coordinate systems

Timestamps are important

- Is your reference time common in between sensors?
  - Ntpdate, chrony and friends
  - Is your reference time correct? Are you sure we are in 1 Jan 1970? **The world started on a Thursday!**
- Triggering sensor reads
  - Great! Worth it? Need it?
- Decide and understand what your timestamp represents
  - Trigger time
  - Receiving time
  - Publishing time
  - Recording time
- CHECK ALL YOUR TIMESTAMPING CODE! EARLY ON! RECHECK IT!

Get to know your sensors

Calibrate

- Intrinsics: camera model + distortion — needed for geometry and consistency.
- Extrinsics: sensor↔sensor and sensor↔body — needed to fuse and to debug.
- Log raw (or raw-enough) AND store calibration/config with the dataset.
- Undistort/canonicalize deliberately (decide where in the pipeline this happens).

Nice tools for intrinsic calibration!
- Generate calibration patterns: https://calib.io/pages/camera-calibration-pattern-generator
- Calibrate any camera by showing a pattern from another screen: https://calibdb.net/

LiDAR point clouds

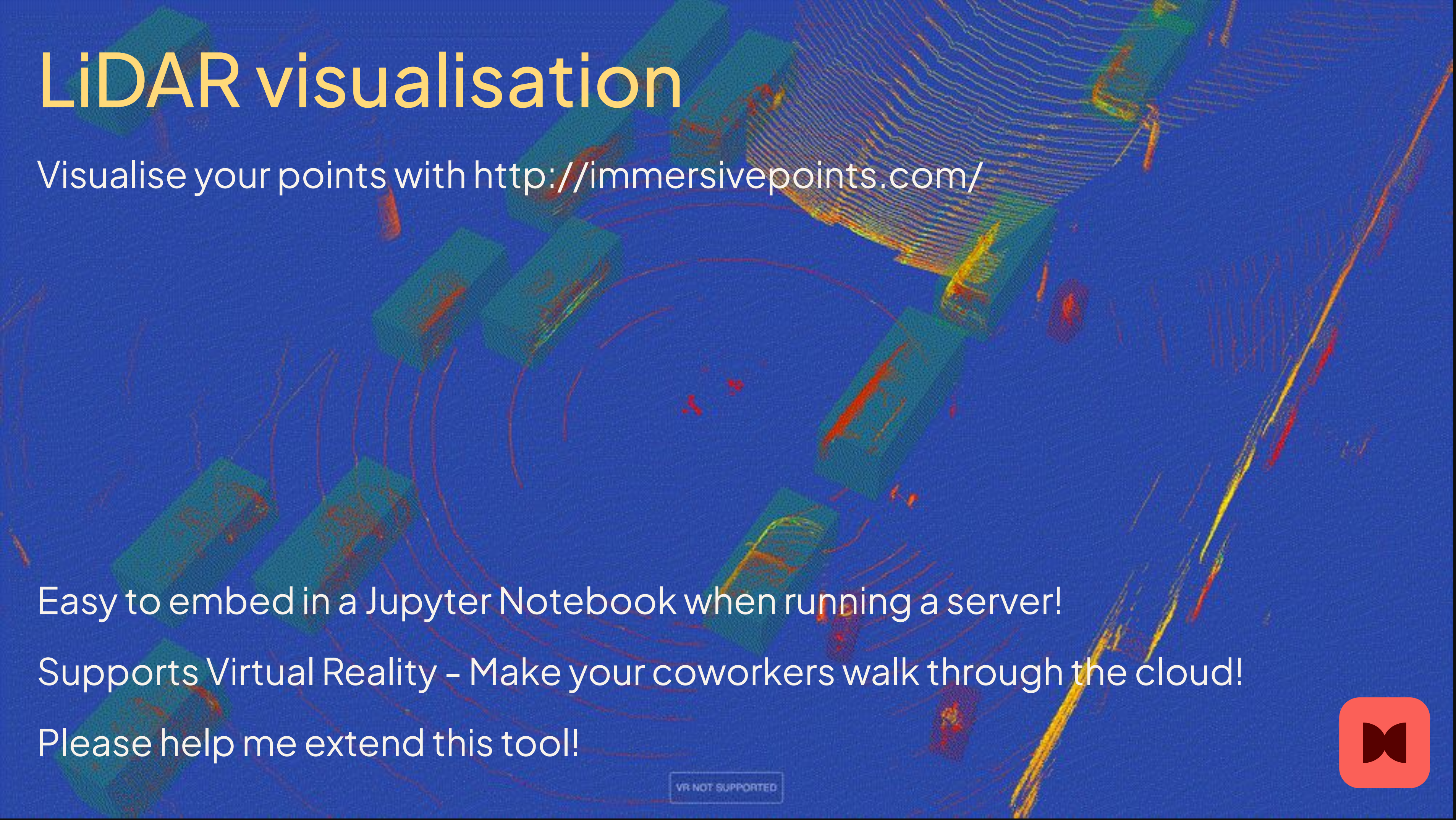
- Generally very accurate (2mm error range), but there are scenarios where it's not...
- Hard to visualise a 3D representation on a 2D screen - so hard to capture 'errors'
- Hard to predict 'errors' as familiarity with LiDAR is needed.
- Common issues:
  - Bad lidar to vehicle/camera calibration
  - Bad lidar to lidar calibration
  - Dirty / blocked lidar not detected
  - Unrealistic expectations of what a LiDAR can do (point density / point spacing)
    - Tangram lidar visualiser (https://www.tangramvision.com/resources/visualizer-lidar)

LiDAR visualisation

- Everyone seems to build their own visualiser
- It's hard to convey details with a plot, as it's 3D information on a 2D screen
- Visualise your points with http://immersivepoints.com/

LiDAR visualisation

Visualise your points with http://immersivepoints.com/



Easy to embed in a Jupyter Notebook when running a server!

Supports Virtual Reality - Make your coworkers walk through the cloud!

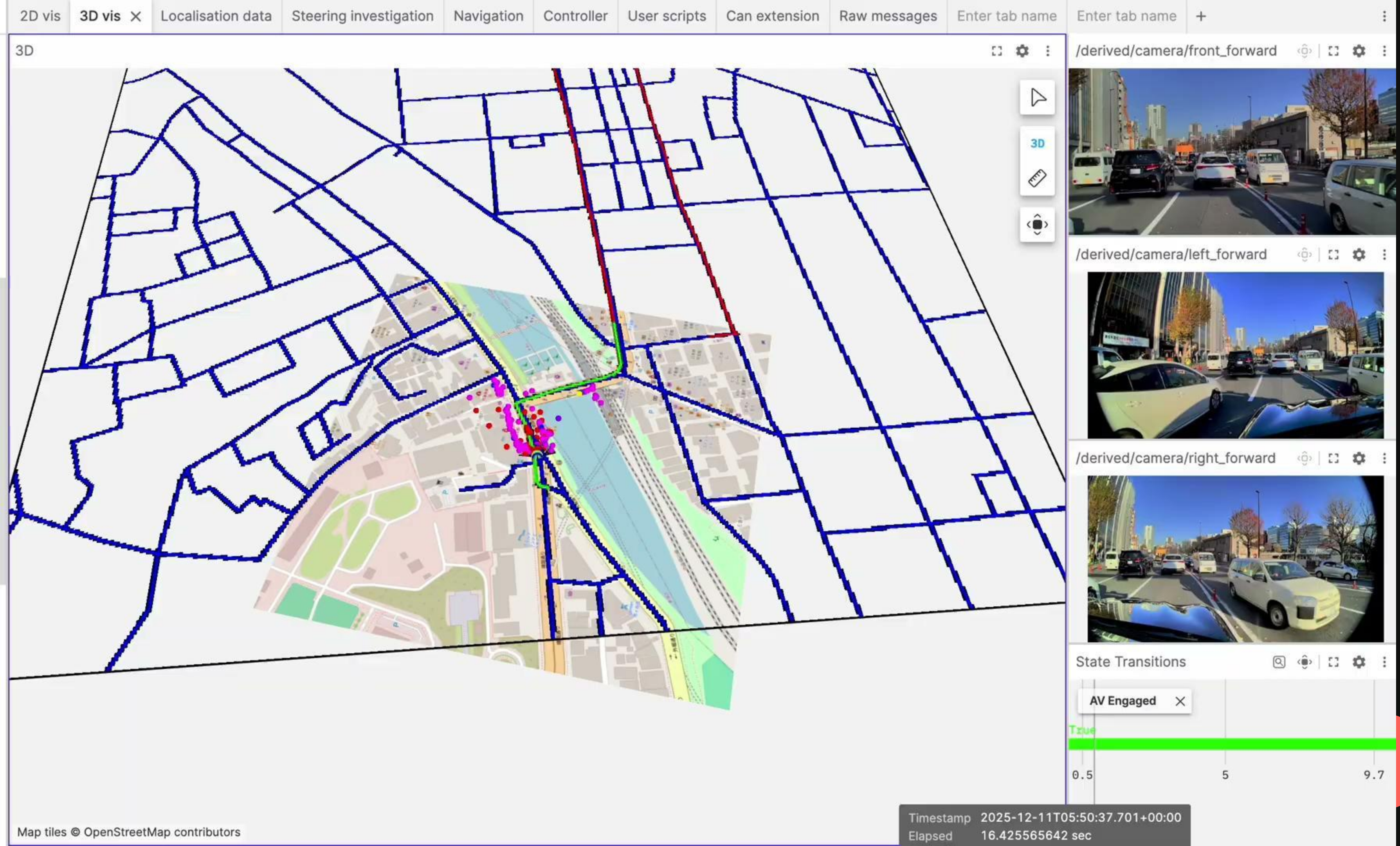
Please help me extend this tool!

Calibrate your senses

Cross-visualisation

Try to visualise sensors with respect to each other! For example:

- Plot lidar points on camera, bounding boxes on both
- Project camera down to the ground - do camera projections "flow over" in each other? Does it align with map-features?
- Does the map on your robot align with the actual world?
- Having good ROS transforms makes this trivial!
- Roland loves Foxglove, Sam loves RVIZ
  - FoxBox/LichtBlick (https://github.com/lichtblick-suite/lichtblick) - A clone of FoxGlove before they went private.
  - Rerun: https://github.com/rerun-io/rerun - committed to stay open source.


2D vis
3D vis
Localisation data
Steering investigation
Navigation
Controller
User scripts
Can extension
Raw messages
Enter tab name
Enter tab name
3D
3D
/derived/camera/front_forward
/derived/camera/left_forward
/derived/camera/right_forward
State Transitions
AV Engaged
True
0.5
5
9.7
Map tiles © OpenStreetMap contributors
Timestamp 2025-12-11T05:50:37.701+00:00
Elapsed 16.425565642 sec

Calibrate your senses

- Consider how "golden" your "golden data" actually is
  - Why is it golden?
  - Is it better in ONE aspect than the one you had before?
  - What does it avoid?
  - What does it incorporate?
  - What does it fix?
  - Does it break anything else?
  - If it was perfect, why would it be?

Easy tricks!

Making simple checks early can save a lot of hassle later

- Monitor time between samples - do you drop images somewhere?
- Monitor start and end of runs - do you start at the place you ended the previous run?
- Visualise what data you record
  - Are your assumptions still correct - objects are visible when you expect it.
  - Are you sensors clean / working / attached / right side up...?
  - Is that sensor the sensor you think? E.g. right/left

Conlusion

- 🕵️ Visualise: make failures obvious (plots, overlays, point clouds, VR if needed), look for the detail.
- 🤖 Automate: cheap checks (blur, FPS, drops, drift, skew, occlusion).
- 🦅 Visualise again: dashboards + spot checks to avoid automating the wrong thing, look for the trend and the outliers.
- 🏷️ Version artifacts: configs + calibrations + datasets + code revisions.

Tools you want to check out

- Sam's sensor preview app: https://awesomebytes.github.io/sensor_preview
- 3D points in VR: immersivepoints.com
- 3D Print the ISO8855 coordinate system: https://www.thingiverse.com/thing:4032327
- FoxBox/LichtBlick (https://github.com/lichtblick-suite/lichtblick) - A clone of FoxGlove before they went private.
- Rerun: https://github.com/rerun-io/rerun
- Lidar planner: https://www.tangramvision.com/resources/visualizer-lidar