3D Localization in Autoware.Auto Using Normal Distribution Transform

Yunus Emre Caliskan
Apex.AI ®

© 2020 Apex.AI, Inc.

01 Localization Using External Sensors

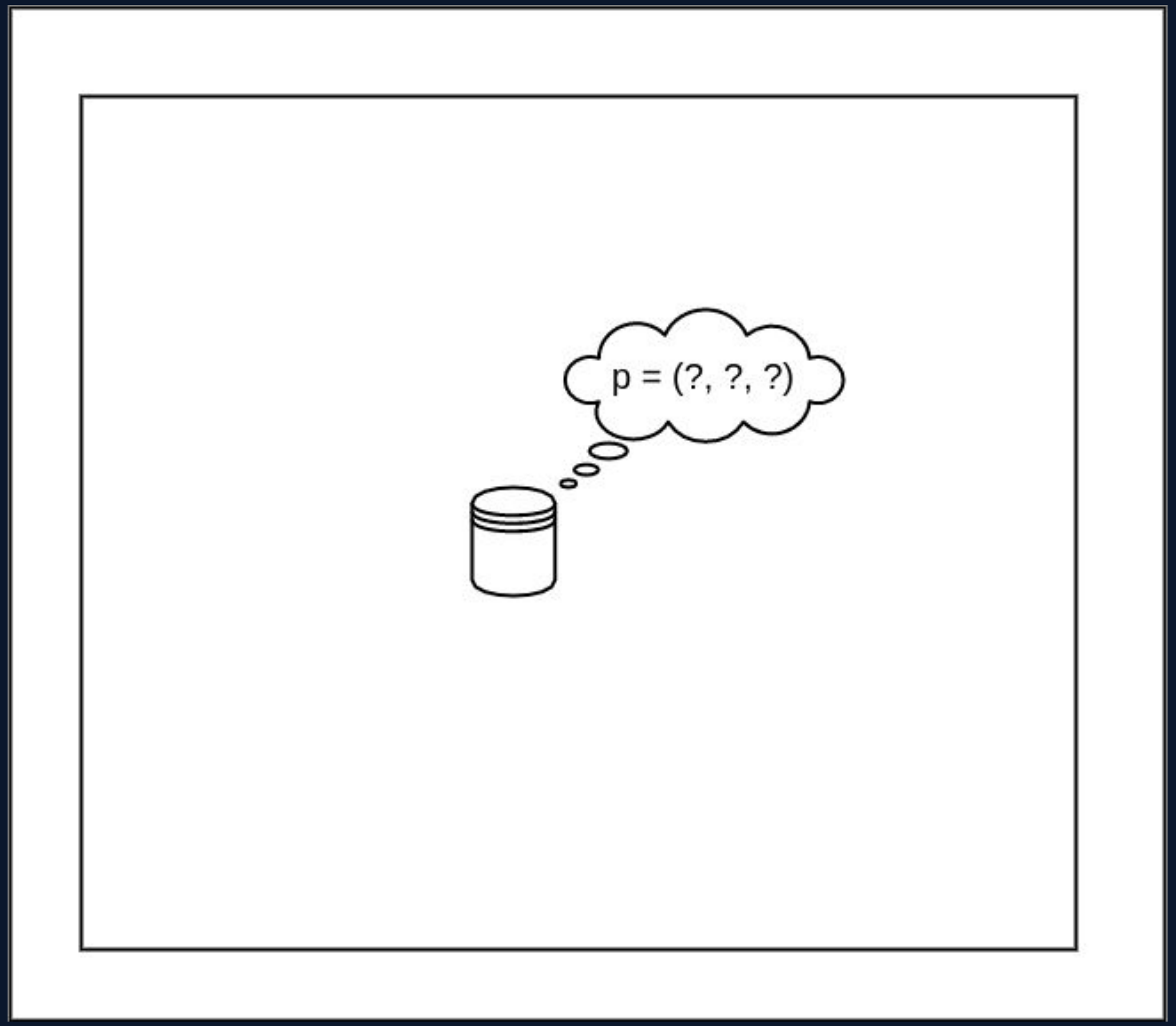
© 2020 Apex.AI, Inc.

Relative Localization: Recap

In order to navigate safely, a mobile robot needs to be informed on where it is located with respect to a static reference point.

This reference point could be:

- A landmark with a known location
- Origin of a map
- Center of the world



How to localize? → By sensing and measuring!

Internal sensors:

- IMU
- Wheel encoder

External sensors:

- GNSS
- Camera
- Lidar

© 2020 Apex.AI, Inc.

Lidar Based Relative Localization

Why have lidar based localization?

- GNSS is subject to limitations:
  - Subject to blockage: Not applicable indoors.
  - Subject to attacks: interference/spoofing.

- Lidar sensors are more precise and robust in sensing the 3D environment compared to other environmental sensors like camera and radar.
  - Robust scale estimation is a challenging problem for camera based techniques.

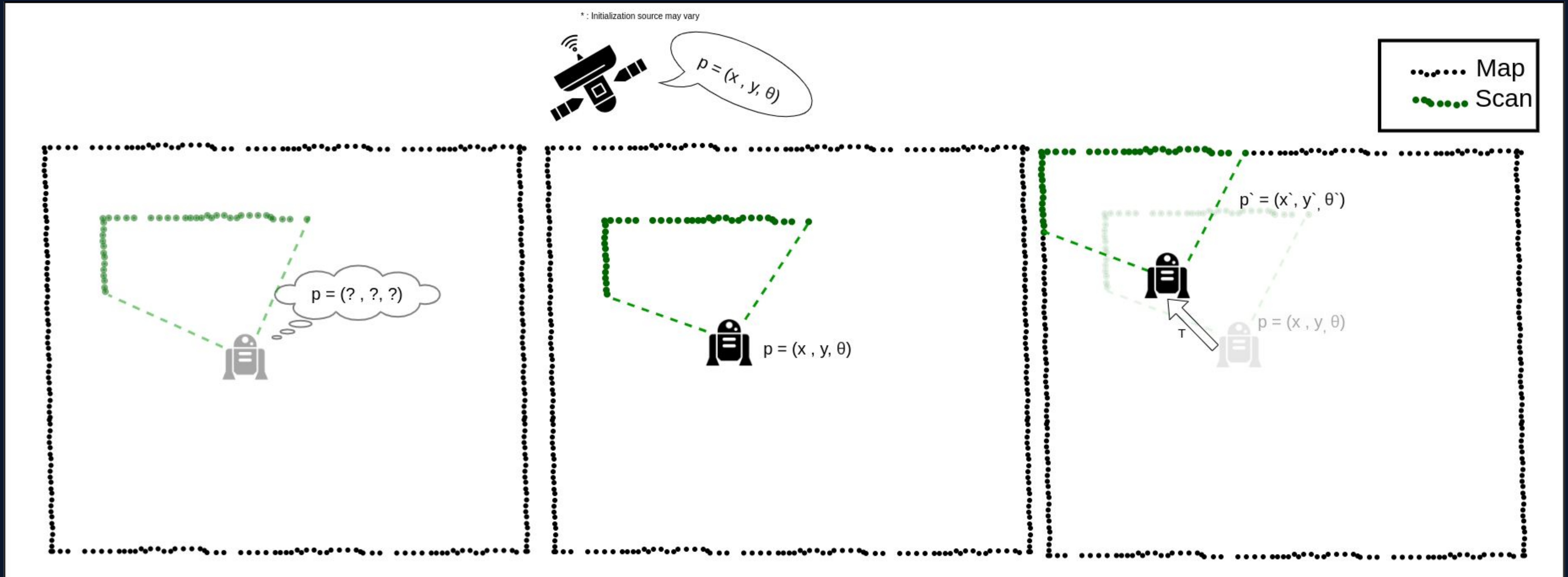
- Lidar scan matching techniques are proven to be quite precise tools for localizing against high resolution point cloud maps.

Limitations:

- Relies on the existence of an accurate map.
- Processing 3d data can be expensive depending on the algorithm.
- Local algorithms relies on the assumption of a somewhat accurate initial guess.

© 2020 Apex.AI, Inc.

Lidar Based Relative Localization



1. Pose unknown 2. Pose initialized 3. Pose corrected using scan matching

Given:
- Map $\mathcal{M}$
- Initial pose estimate p
  - Only for local algorithms
- scan S

Find:
- Best alignment: $\mathtt{T}$
- Corrected pose estimate $p` = \mathtt{T}\ p$

How?
- ICP [Besl & McKay, 1992]
- NDT [Biber & Straßer, 2003]
- Others

© 2020 Apex.AI, Inc.

Lidar Scan Matching

- Treat the alignment as a minimization problem

- Minimize alignment error E.

What is E?

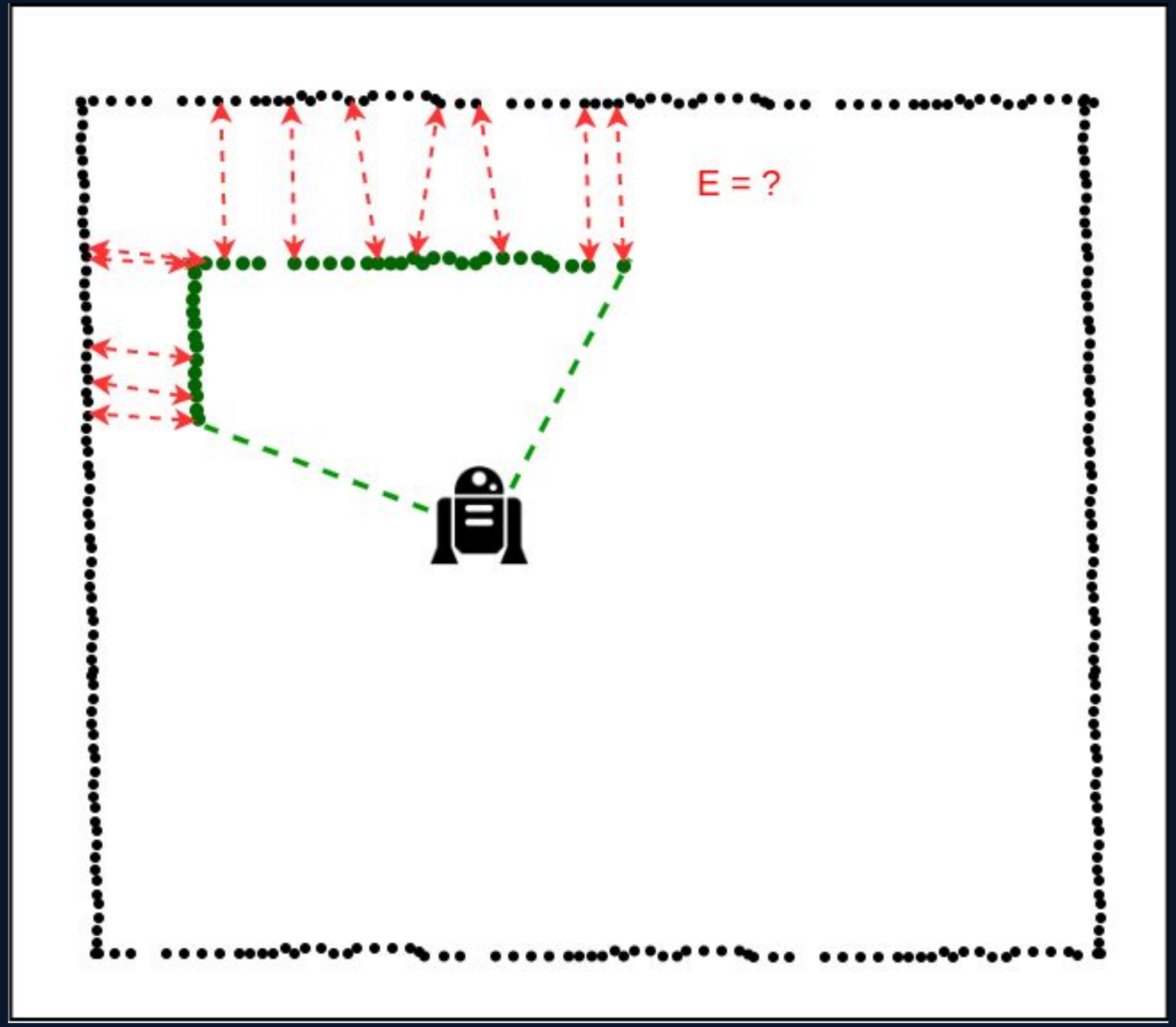
Plain `Iterative closest points` (ICP) algorithm:
- Given the points in the scan S and the map M:

$$x_s \in S \land x_m \in M$$

- Define the Error term E:
  - Closest point-to-point distance:

$$E = \sum_{i=1}^{N_p} (T x_{s_i} - x_{m_i})^2$$

- Find the T that minimizes E



Cons*:
- More sensitive to initial estimate and the level of alignment than NDT [1] [2]
  - Bad initial alignment often leads to getting stuck at a local optimum. [2]
- Benchmarked to be less accurate than the base(P2D) NDT method in varying scenarios. [1] [3]

*: depends on the variant

[1]: [Magnusson et al., 2015]
[2]: [He et al., 2017]
[3]: [Magnusson et al., 2009]

© 2020 Apex.AI, Inc.

02 Normal Distribution Transform

© 2020 Apex.AI, Inc.

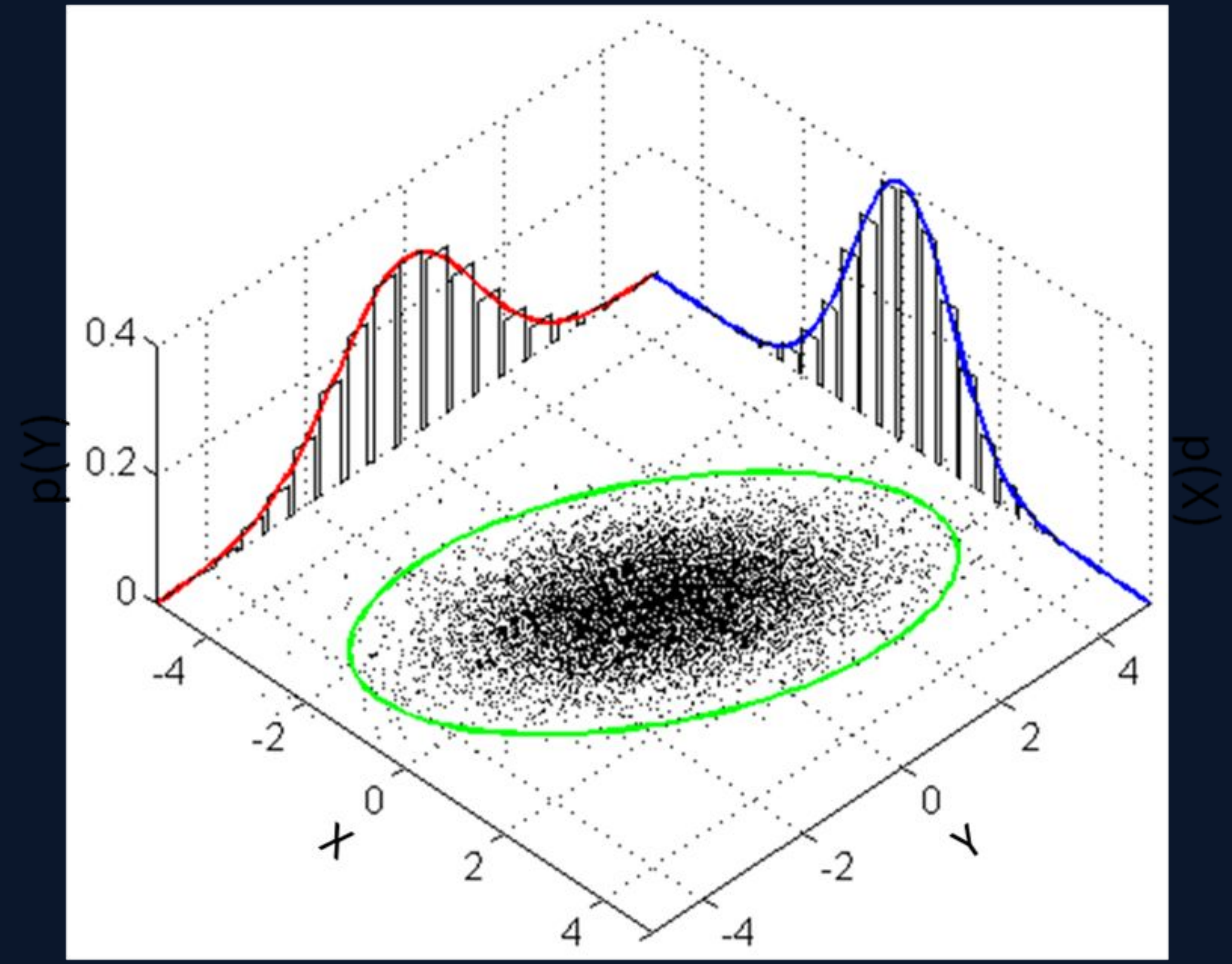
Multivariate Normal Distribution

Probability density function:

$$f(x) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} e^{\frac{-1}{2}(x-\mu)\Sigma^{-1}(x-\mu)^T}$$

Where μ is the mean of distribution and

Σ is the covariance of the distribution.

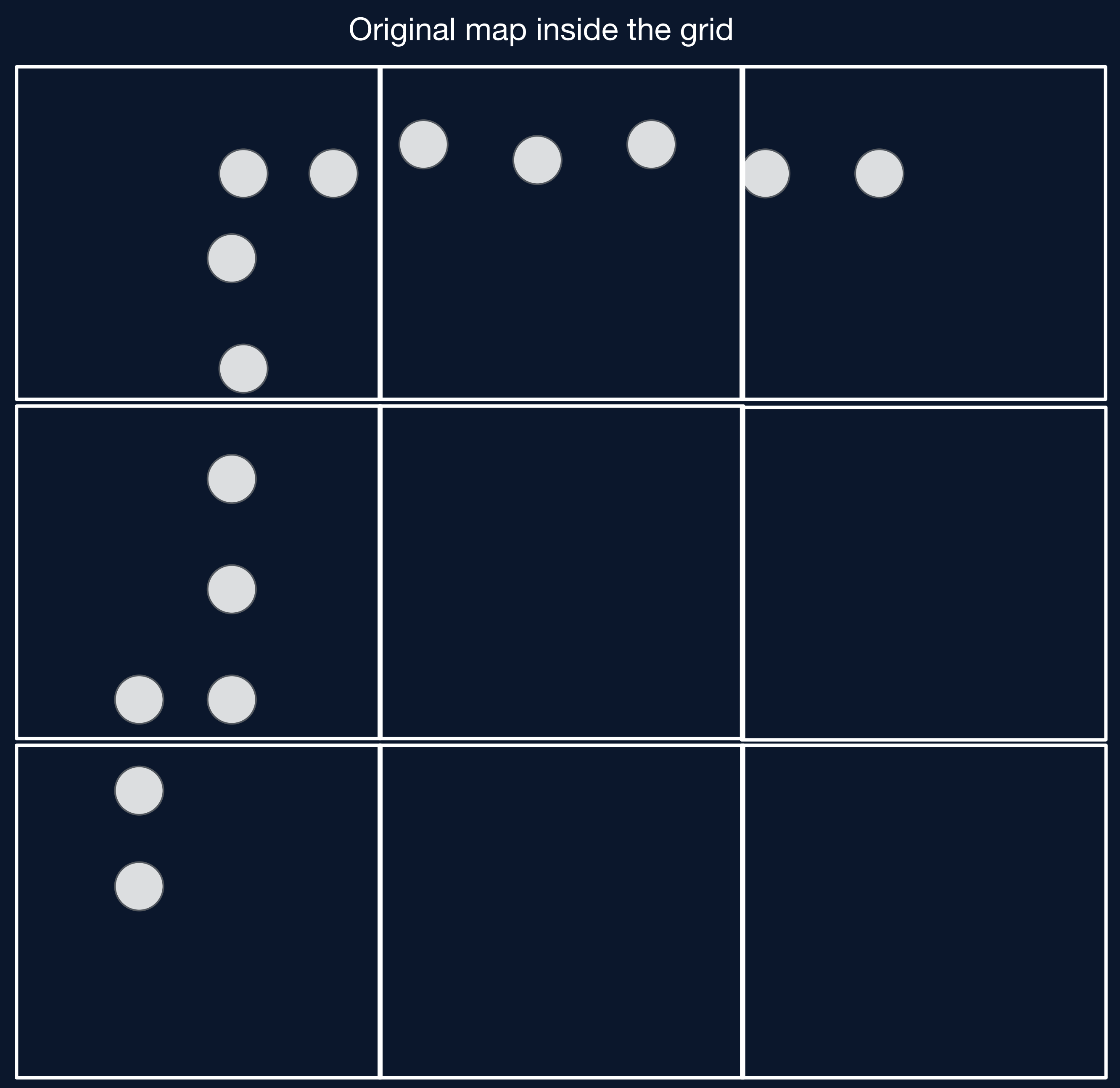


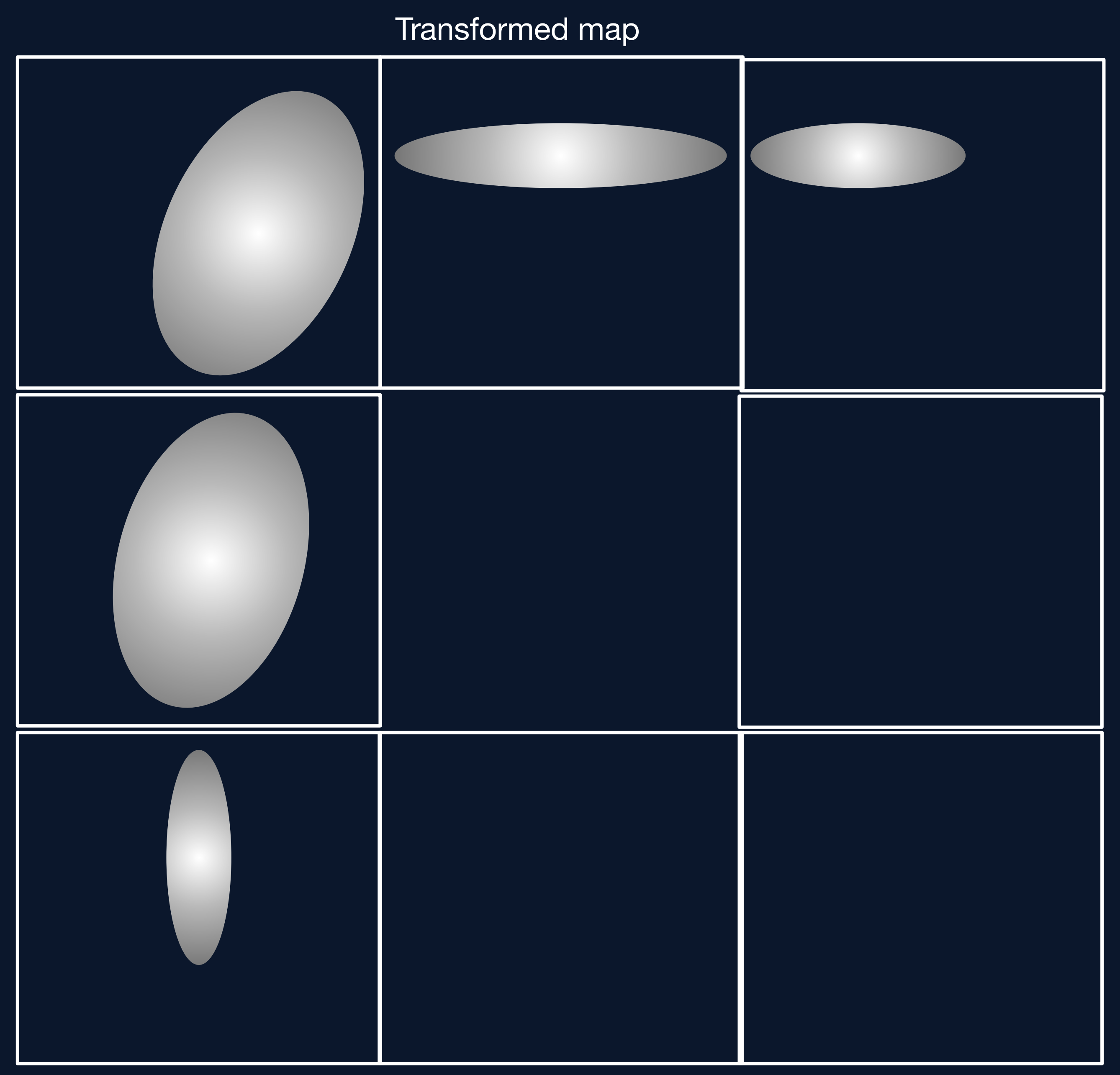
[1] Sample 2D gaussian distribution

[1]: https://en.wikipedia.org/wiki/Multivariate_normal_distribution

© 2020 Apex.AI, Inc.

2D Normal Distribution Transform [Biben & Straßer, 2003]





- Divide the map into a grid.
- Each cell represents a normal distribution of points.
- Mean and covariance is computed using each point \`x\` in a cell with \`N\` number of points:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} x_i$$

$$\Sigma = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \mu)(x_i - \mu)^T$$

© 2020 Apex.AI, Inc.

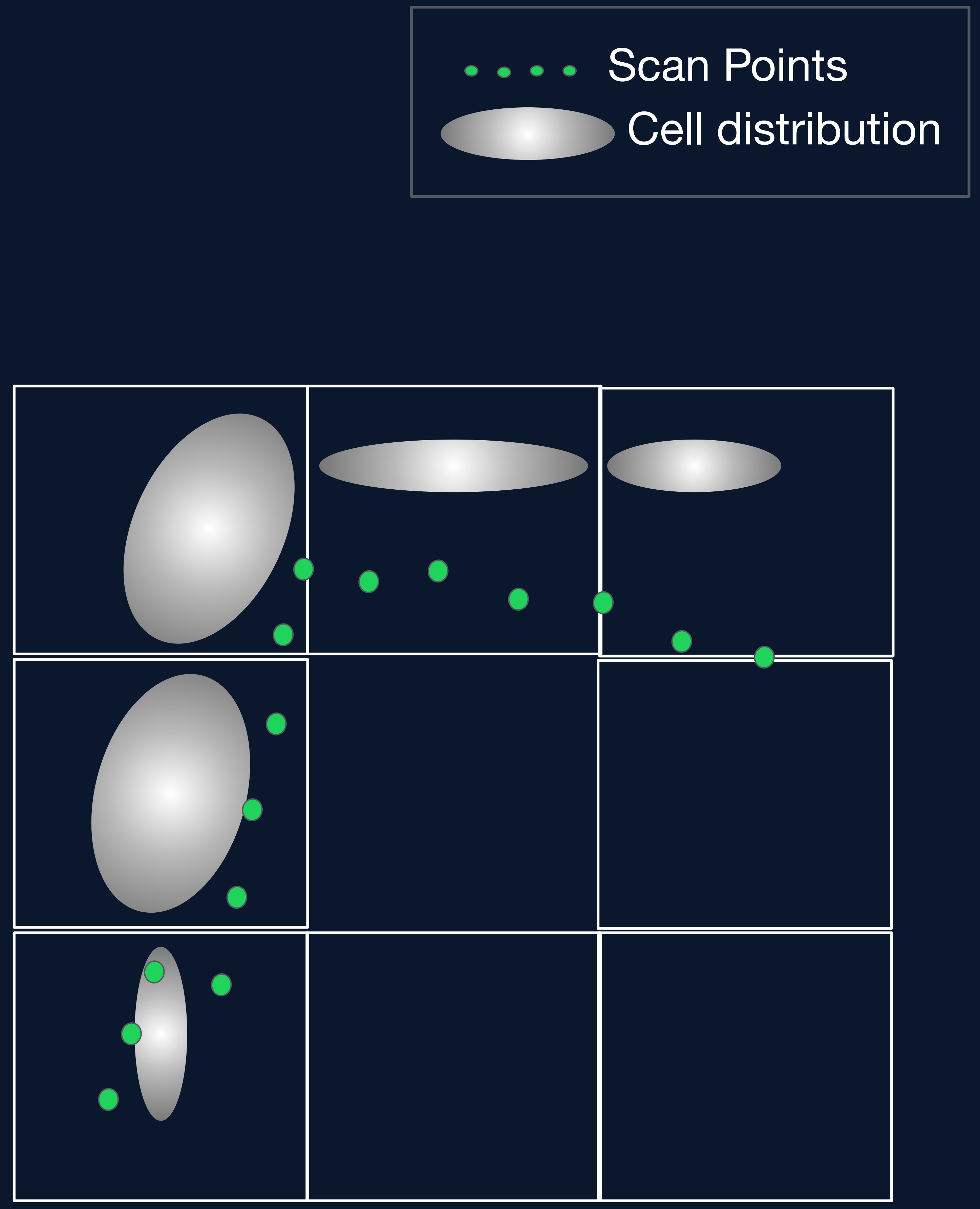
NDT Alignment Problem

$$p(x) = \frac{1}{c} e^{\frac{-1}{2}(x-\mu)\Sigma^{-1}(x-\mu)^T}$$

❖ Probability density function p(x) of a cell C for a point `x` is correlated with the probability that the scan point belongs to the cell C.

- If a high number of points in a scan has a high probability of being in an occupied cell in the map grid, we can assume a good alignment.
- Hence the following sum must be maximized for an optimal alignment:

$$\sum_{i=1}^{N_s} e^{\frac{-1}{2}(T(x_{s_i}, p)-\mu_i)\Sigma_i^{-1}(T(x_{s_i}, p)-\mu_i)^T}$$



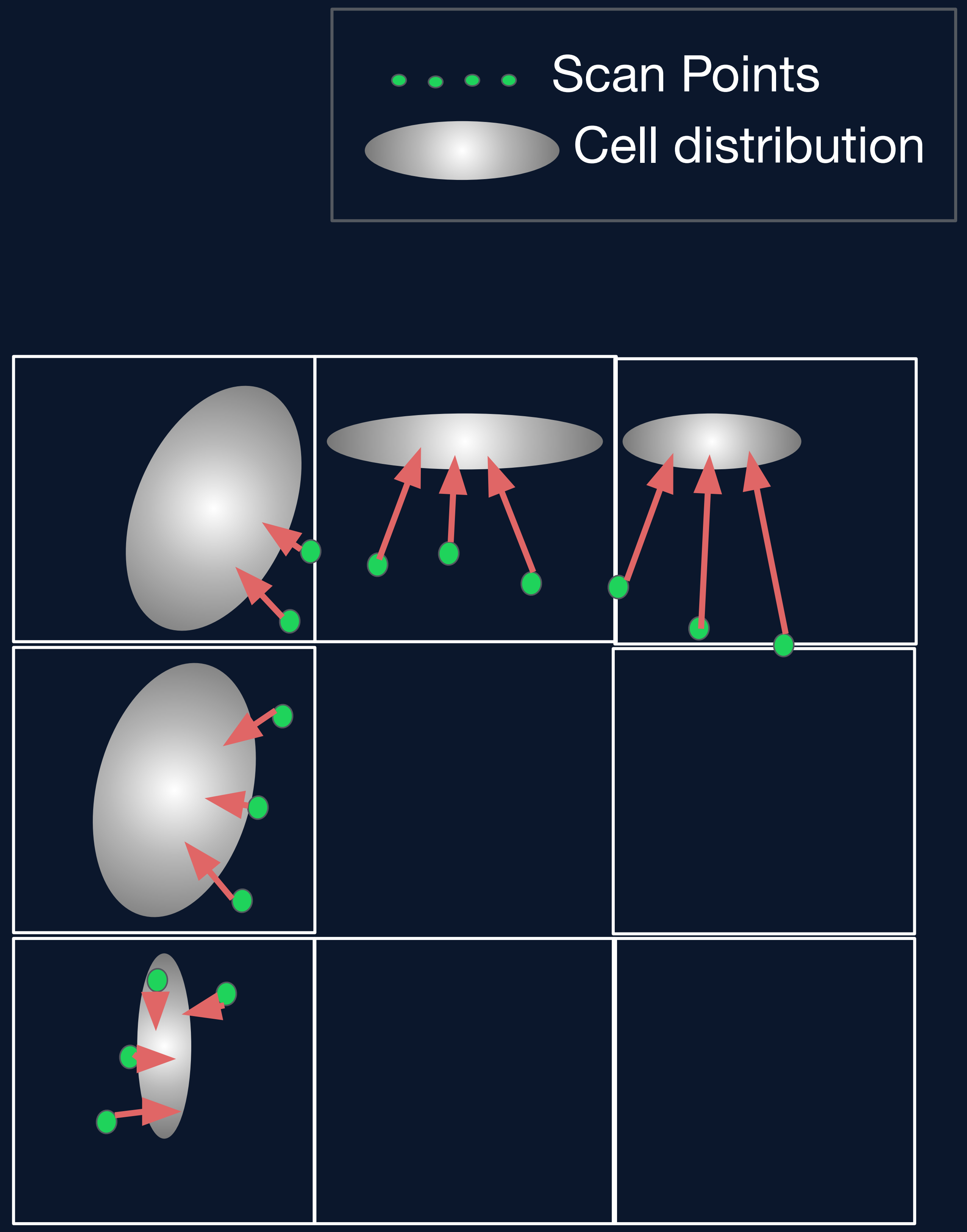
Where $T(x_{s_i}, p)$ is the transformation function that transforms a point x_s from sensor frame to the map frame using the pose p. μ_i and Σ_i are mean and covariance of the cell the transformed point.

© 2020 Apex.AI, Inc.

NDT Alignment Optimization

$p_{2d} = (x,\ y, \theta)$ $p_{3D} = (x,\ y,\ z,\ \phi,\ \theta,\ \psi)$

$$score(p) = -\sum_{i=1}^{N_s} e^{\frac{-1}{2}(T(x_{s_i}, p)-\mu_i)\Sigma_i^{-1}(T(x_{s_i}, p)-\mu_i)^T}$$



Now that we have a minimization problem, we can select an appropriate optimization technique!

Example: Newton's method: $\Delta p = H^{-1} g$

Where H is the hessian and g is the gradient of *score(p)*

- Either directly use direction Δp using fixed step length.
- Or compute a step length for Δp using a line search algorithm.
  - In autoware.auto, an algorithm proposed in [More & thuente, 1994] is used.

© 2020 Apex.AI, Inc.

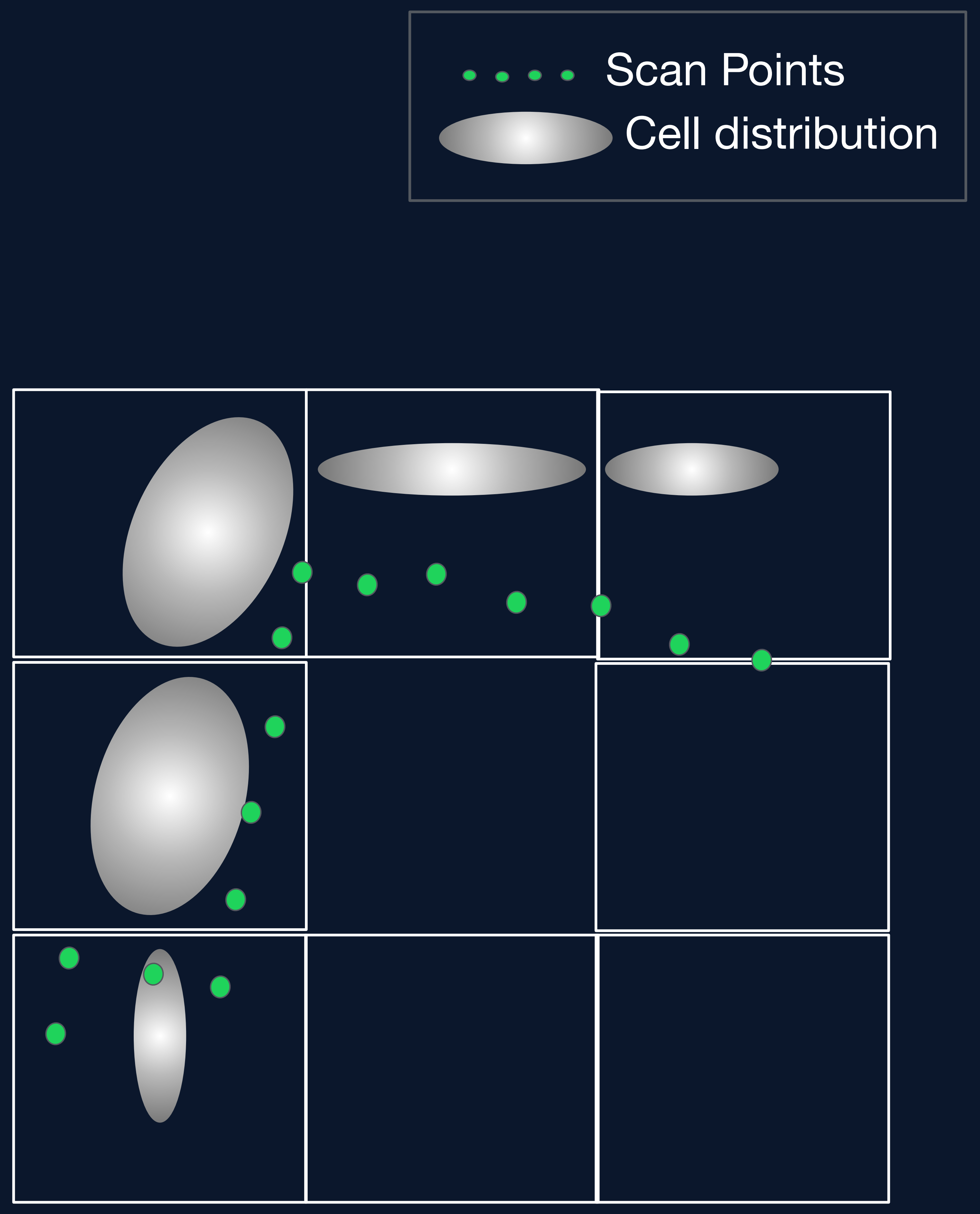
NDT Algorithm [Magnusson, 2009]

Transform the dense point cloud map into an NDT grid

1. Discretize the map into cells.
2. Compute mean and covariance for each cell.

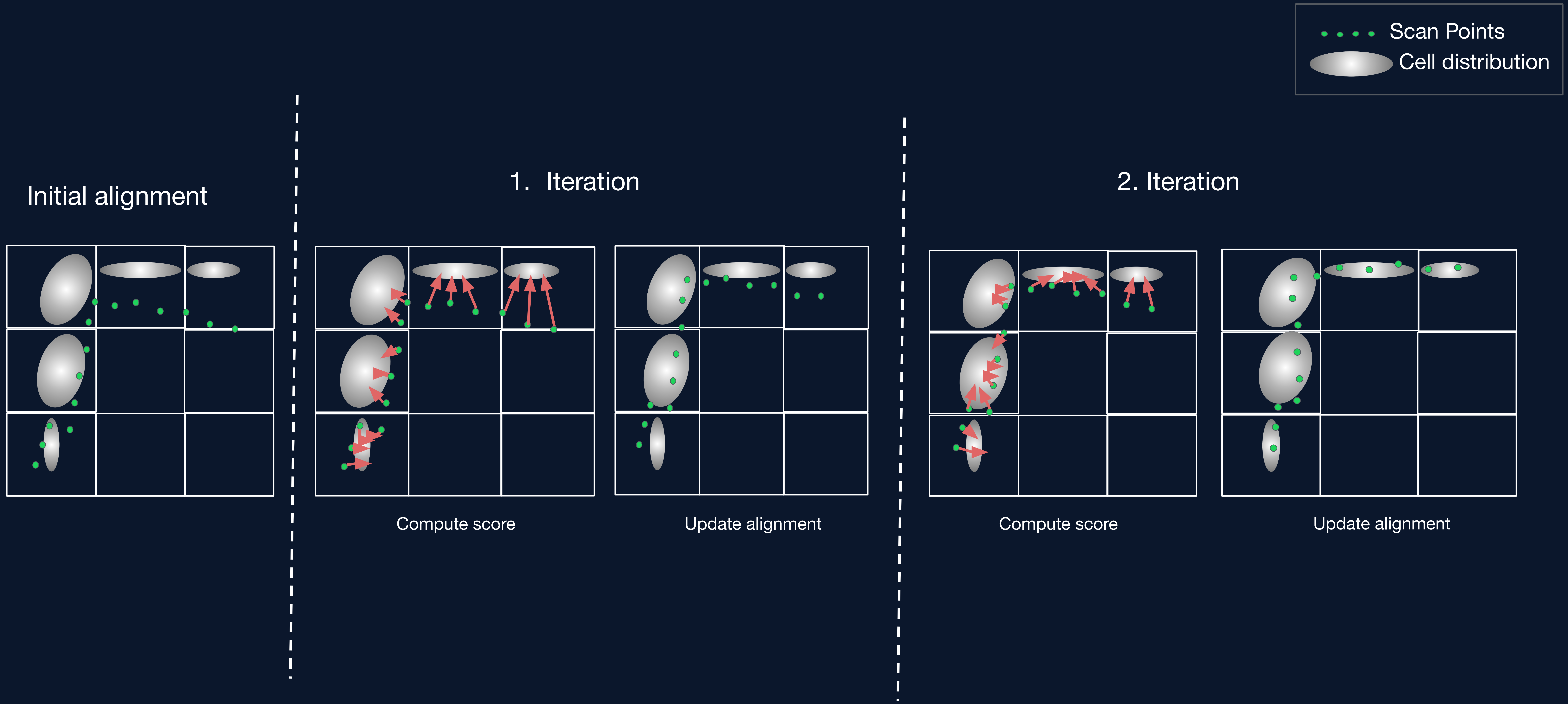
For each received scan:

1. Generate an initial pose estimate p to start the optimization.
2. Start optimization:
   a. Compute *score*(p), *H* and *g* for p.
   b. compute Δp using Newton's method.
   c. Estimate a step length α using line search.
   d. Update p: $p = p + \alpha \Delta p$
   e. Repeat until converged or iteration limit is reached
3. Publish the final value of p.



© 2020 Apex.AI, Inc.

NDT Algorithm



© 2020 Apex.AI, Inc.

Strengths of NDT localization

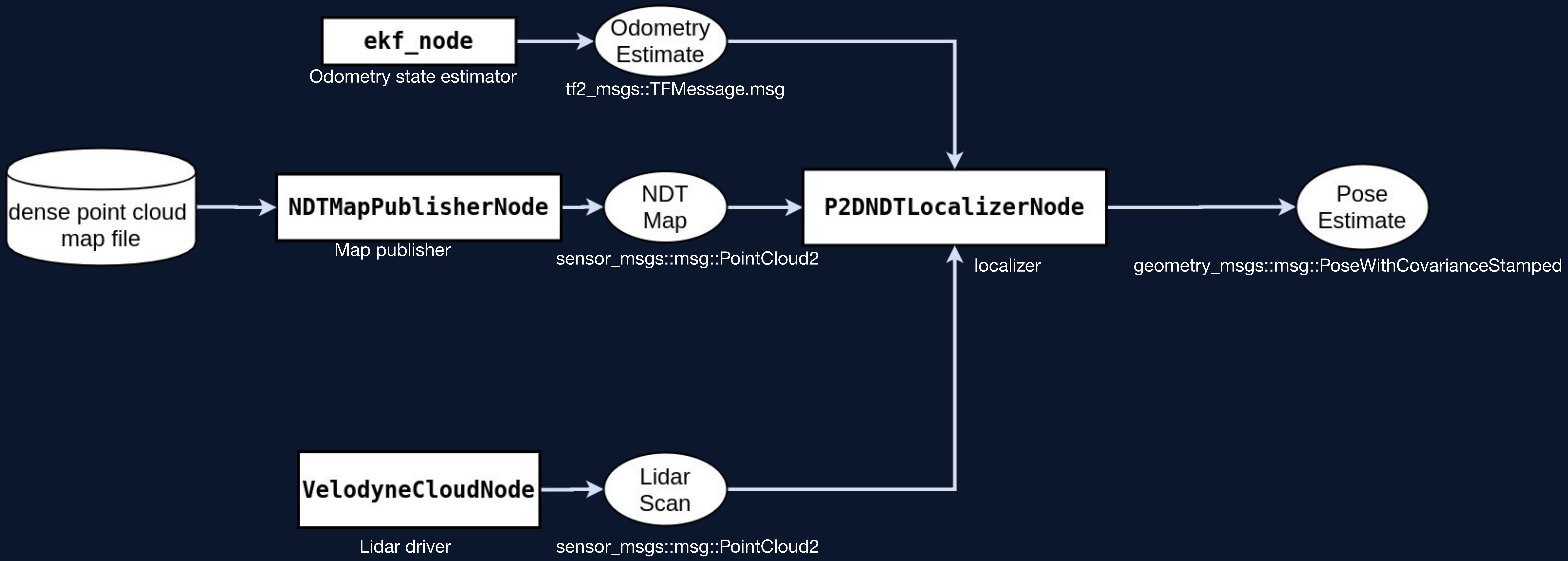
- Map has a sparse representation:
  - Lower memory cost
  - Lower computational cost
- No need to engineer a point-correspondence criterion for alignment
- Normal distribution transform addresses the uncertainty in data.
  - More robust to noisy data.
- Variants allow faster computation and/or better accuracy:
  - D2D NDT: [Stoyanov et al., 2012]
  - NDT with constraints on pose: [Andreasson et al., 2017]
  - See more: https://autowarefoundation.gitlab.io/autoware.auto/AutowareAuto/ndt-literature-review.html

© 2020 Apex.AI, Inc.

03 Autoware.Auto Implementation

© 2020 Apex.AI, Inc.

Autoware Auto NDT Localization Architecture



© 2020 Apex.AI, Inc.

A closer look into the P2DNDTLocalizerNode

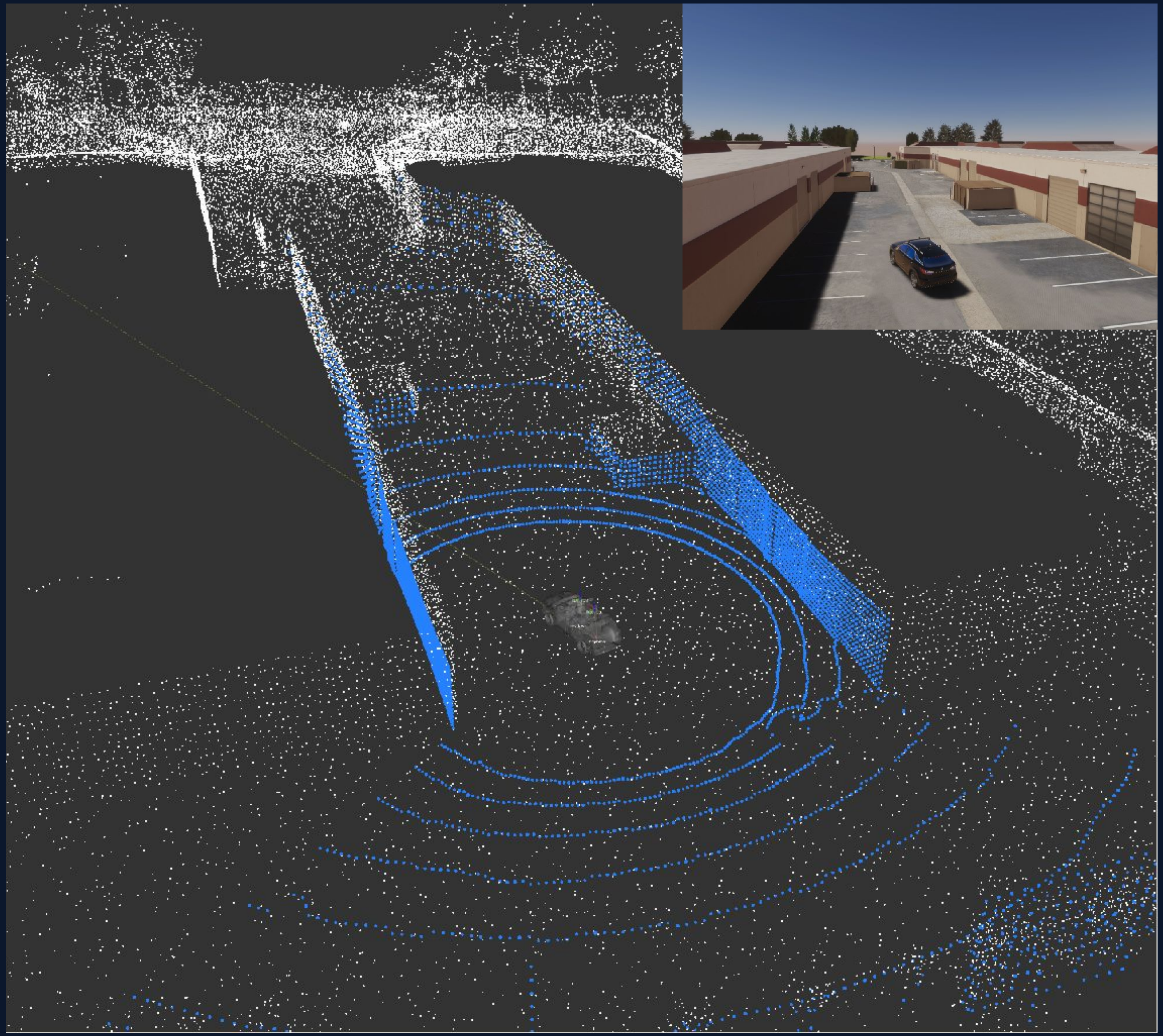
© 2020 Apex.AI, Inc.

NDT Implementation Details

Experimental findings:

- Fixed step-length optimization was not really good at converging in hard alignment problems so `More-thuente` line search algorithm is utilized to improve.

- Cells with co-linear or co-planar points result in singular covariance matrices.
    - Stabilizing covariance matrices before computing the score improves the numerical stability.
    - <u>This stabilization</u> step helped with the convergence characteristics in low information environments with large planar surfaces.



© 2020 Apex.AI, Inc.

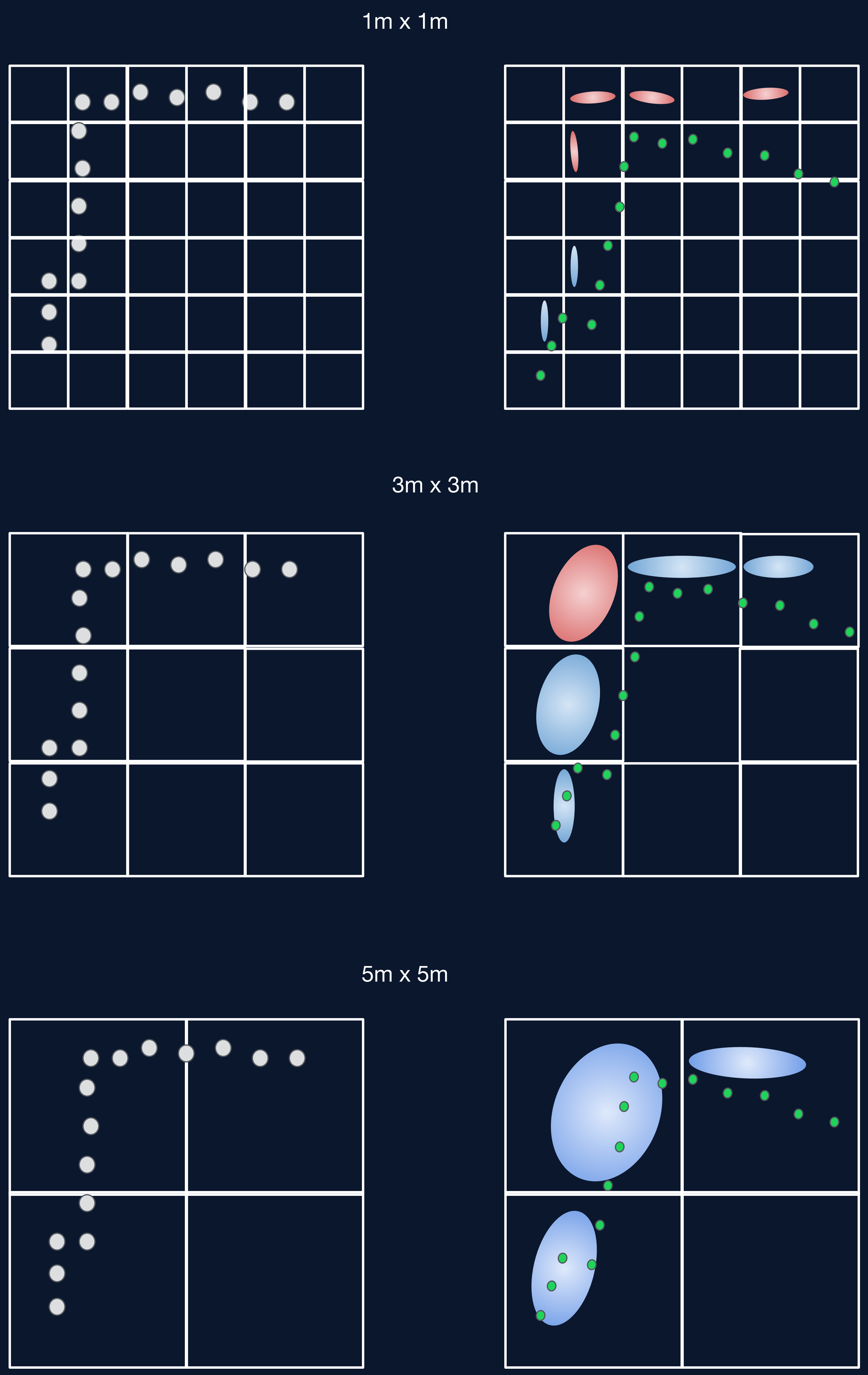
NDT Implementation Details

Effect of parameterization:

- High resolution requires a smaller voxel size.
  - Small voxel size makes the convergence more sensitive on the initial alignment.
  - In higher velocities, the gap between the predicted location (initial guess) and the actual lidar scan may not be close enough to correspond to unoccupied voxels.
- Larger voxels are less sensitive to initial alignment.
  - Lower resolution: less features to match.
- It's hard to find a voxel size that is fit for all scenarios.

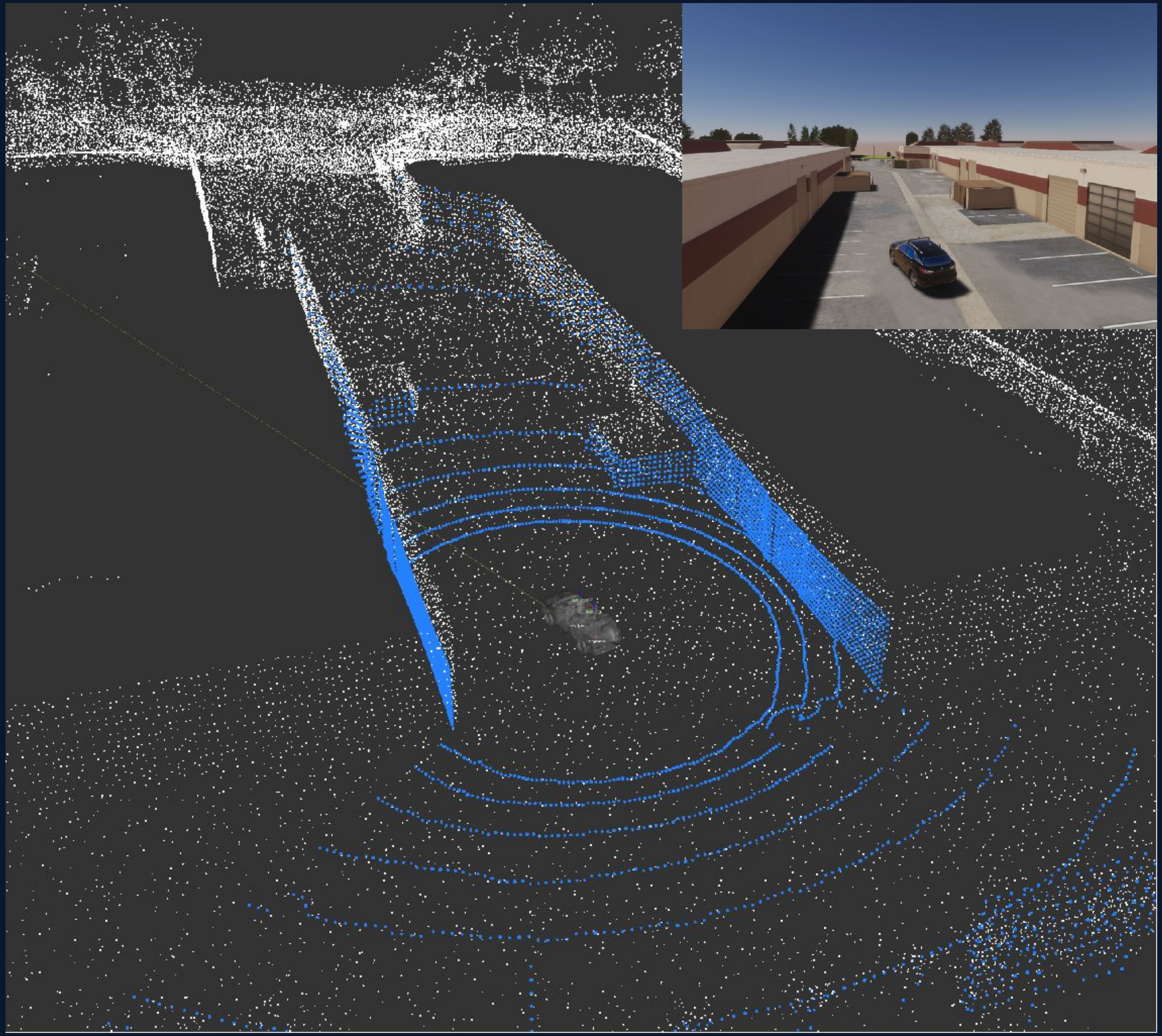
Solution:

- Multi-resolution maps
- Octree maps



© 2020 Apex.AI, Inc.

Live Demo



© 2020 Apex.AI, Inc.