A Fuzzy Spatial Relationship Graph for Point Clouds Using Bounding Boxes

Andrew R. Buck[a], Derek T. Anderson[a], James M. Keller[a], Robert H. Luke III[b], and Grant Scott[a]

[a]Electrical Engineering and Computer Science (EECS) Department, University of Missouri, Columbia, MO, USA

[b]US Army DEVCOM C5ISR Center, Fort Belvoir, VA, USA

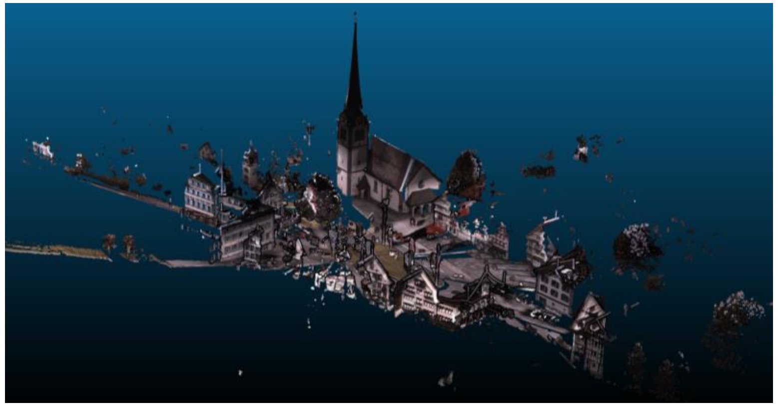
FUZZ-IEEE 2021

Presented by Andrew Buck

University of Missouri

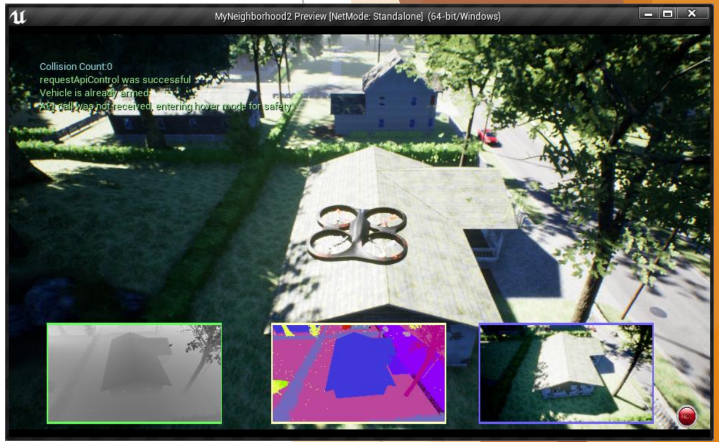
Motivation

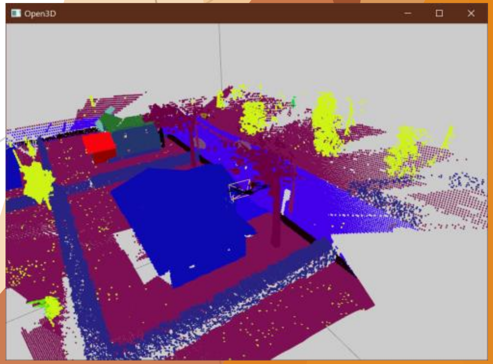
- 3D scenes have a lot of information!
  - How should it be represented?
  - What are the relationships between objects?
- Point clouds give raw 3D data
  - Easy to acquire as raw data from LIDAR or depth camera
  - Files can be huge! Voxels can help sometimes…
  - How to represent the important aspects of the scene?
- Semantic segmentation can identify individual objects
  - How to store and query spatial configurations?
  - Use bounding boxes to represent important objects
  - Compute relationships between bounding boxes



http://www.semantic3d.net/

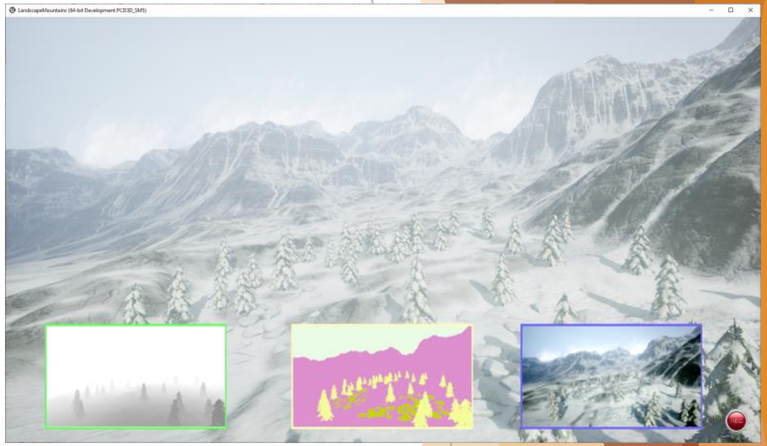
https://github.com/microsoft/AirSim

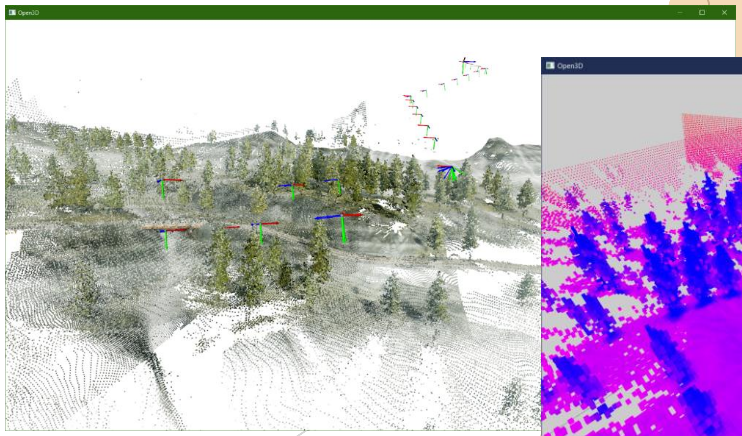


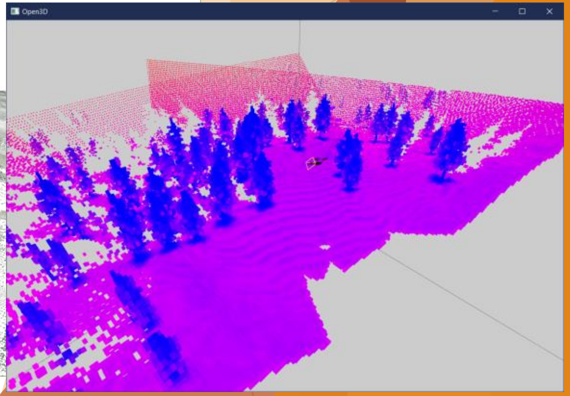


Applications

- Ultimately, we would like AI systems to have an interpretable understanding of their environment
  - This can help design and communicate intended behaviors
  - Make an AI agent act more like a human
- Unmanned Aerial Vehicles (UAVs)
  - Small embedded systems need to respond in real-time
  - Require minimal overhead and processing
- Human robot interaction
  - Use natural language to communicate
  - Mobile computing devices (AR/VR headsets) with limited streaming bandwidth



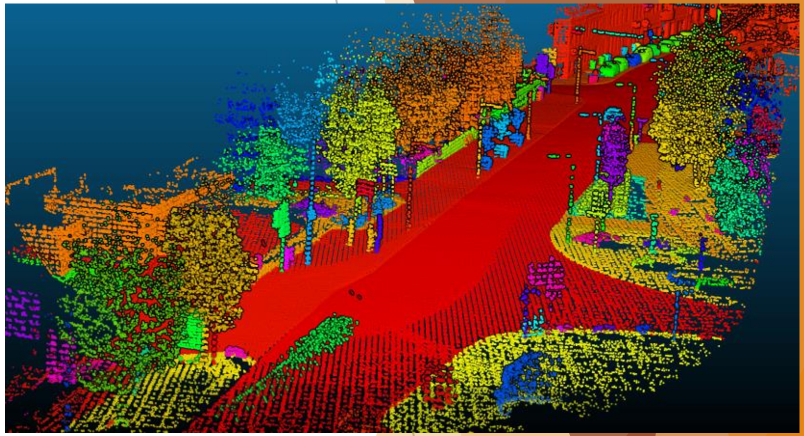




University of Missouri

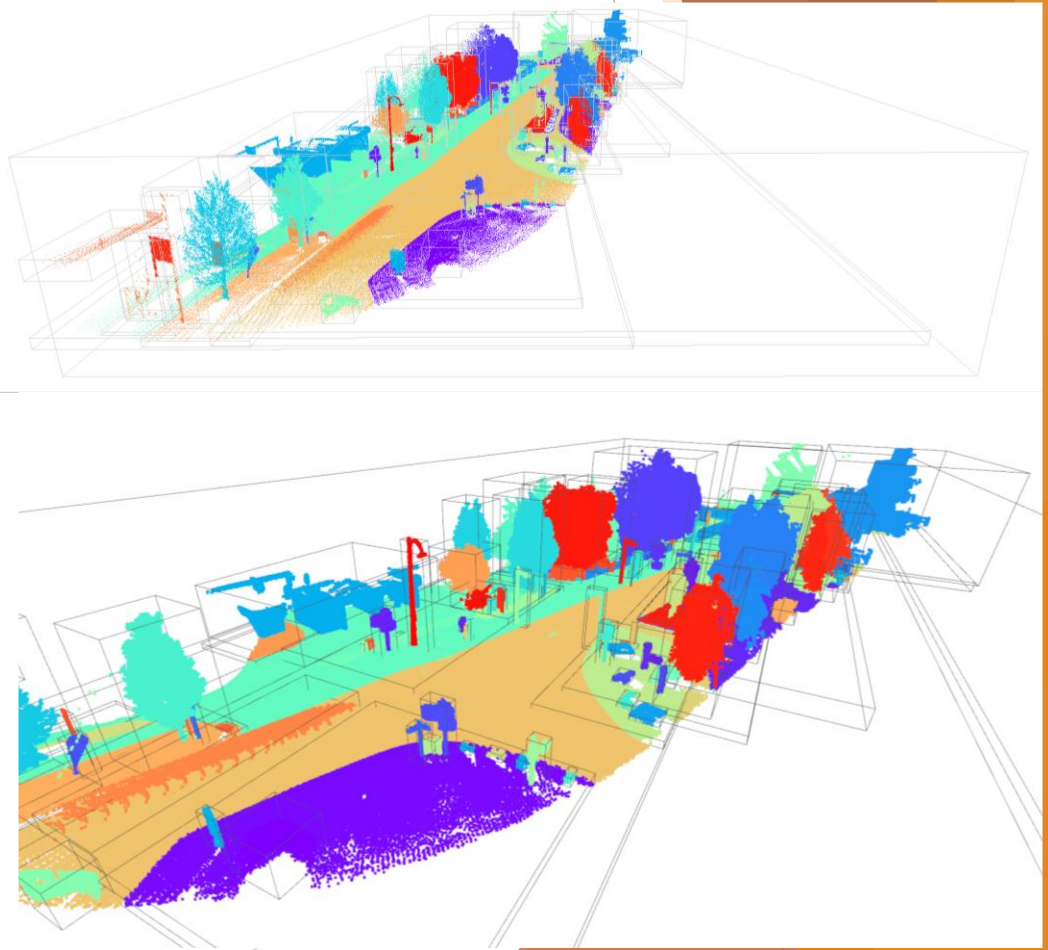
Benchmark Datasets

- We assume that a semantic segmentation of the scene can be acquired
  - Many recent works for labeling and segmenting point clouds
- Focus here on ground truth, hand-labeled benchmark datasets
  - Each object instance is given a unique ID
  - Easy to find individual objects or categories
- Looking at the NPM3D benchmark suite
  - https://npm3d.fr/paris-lille-3d
- Outdoor street scene (static)
- Chose this dataset because it has ground truth segmentation
  - Objects are identified by class and an instance ID
  - 50 classes organized in a hierarchal ontology



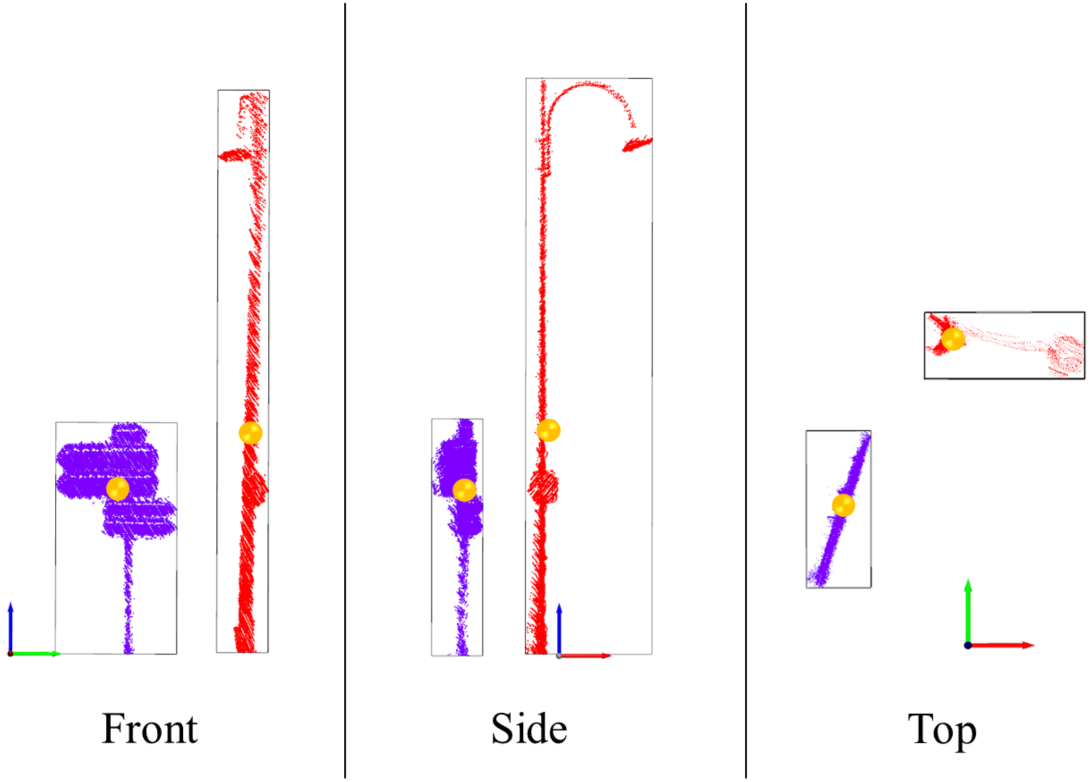
Example Scene

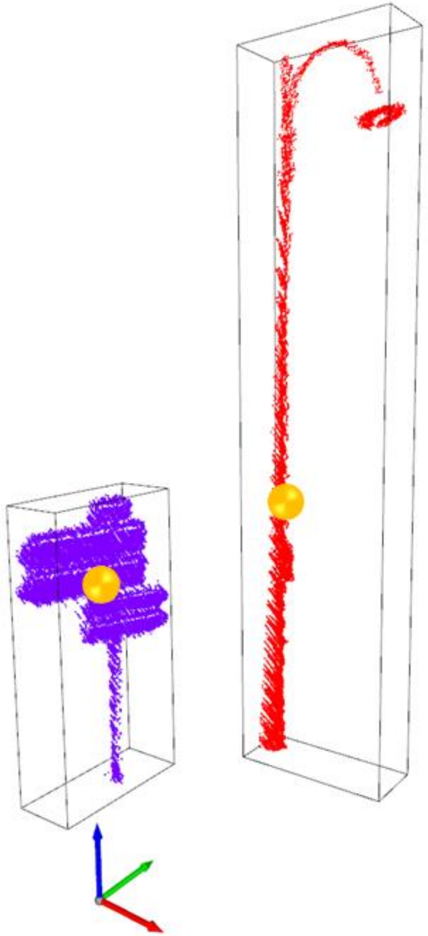
- We chose to look at an example region with ~10,000,000 points and ~100 labeled objects
- How to compute the spatial relationships between objects?
- Each object can be shown with an axis-aligned bounding box (trivial to compute)



Bounding Box Representation

- Consider the relationship between these two objects.
- We can easily compute the bounding boxes and centroids.





University of Missouri

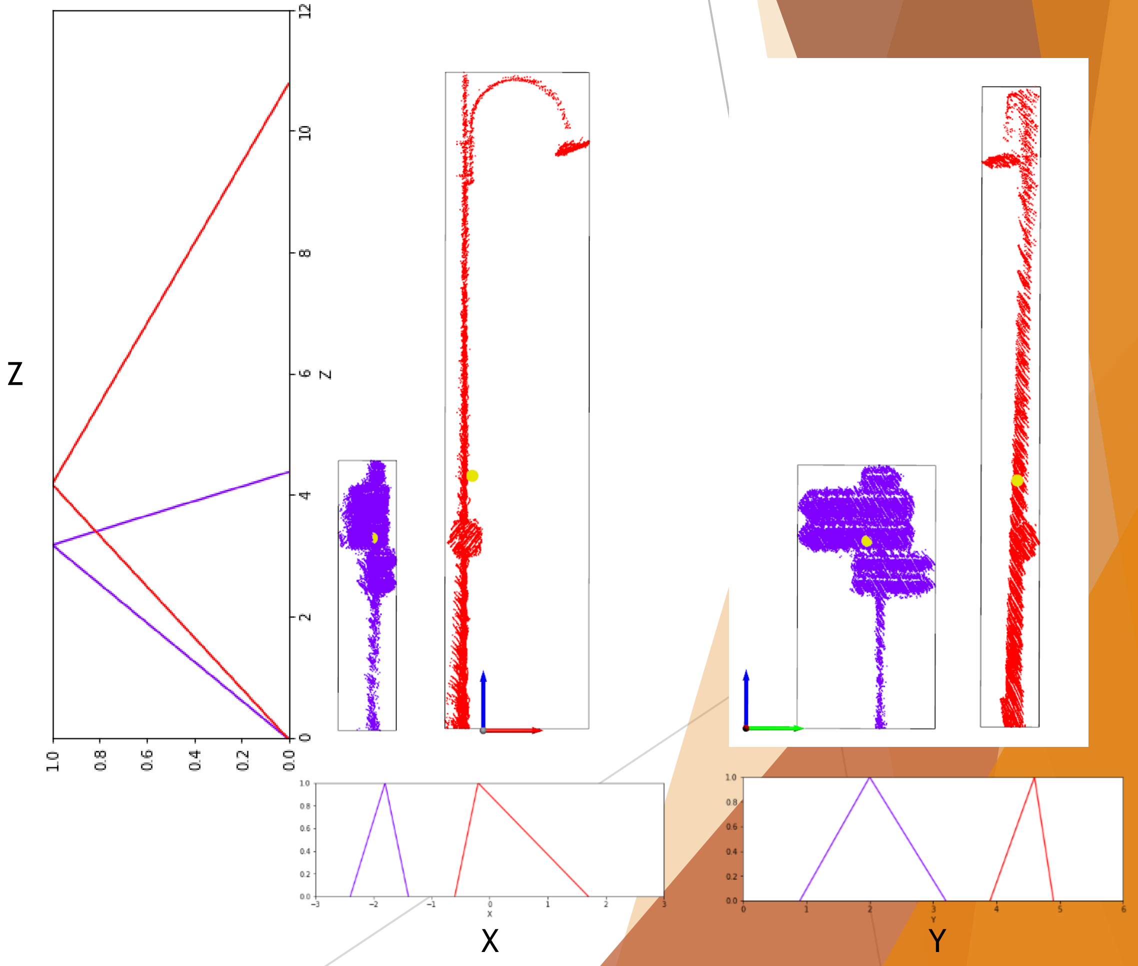
Triangular Fuzzy Numbers

- Along each dimension, we define a triangular fuzzy number (TFN) from the minimum and maximum extents of the bounding box and the object centroid.
- Store 9 values for each object

$$A = \text{Tri}(a_1, a_2, a_3), \qquad B = \text{Tri}(b_1, b_2, b_3)$$

Object Bounds and Centroid: [min, centroid, max]

	X	Y	Z
Signpost	[-2.4, -1.8, -1.4]	[0.9, 2.0, 3.2]	[0, 3.2, 4.4]
Light pole	[-0.6, -0.2, 1.7]	[3.9, 4.6, 4.9]	[0, 4.2, 10.8]



Bounding Box Distance

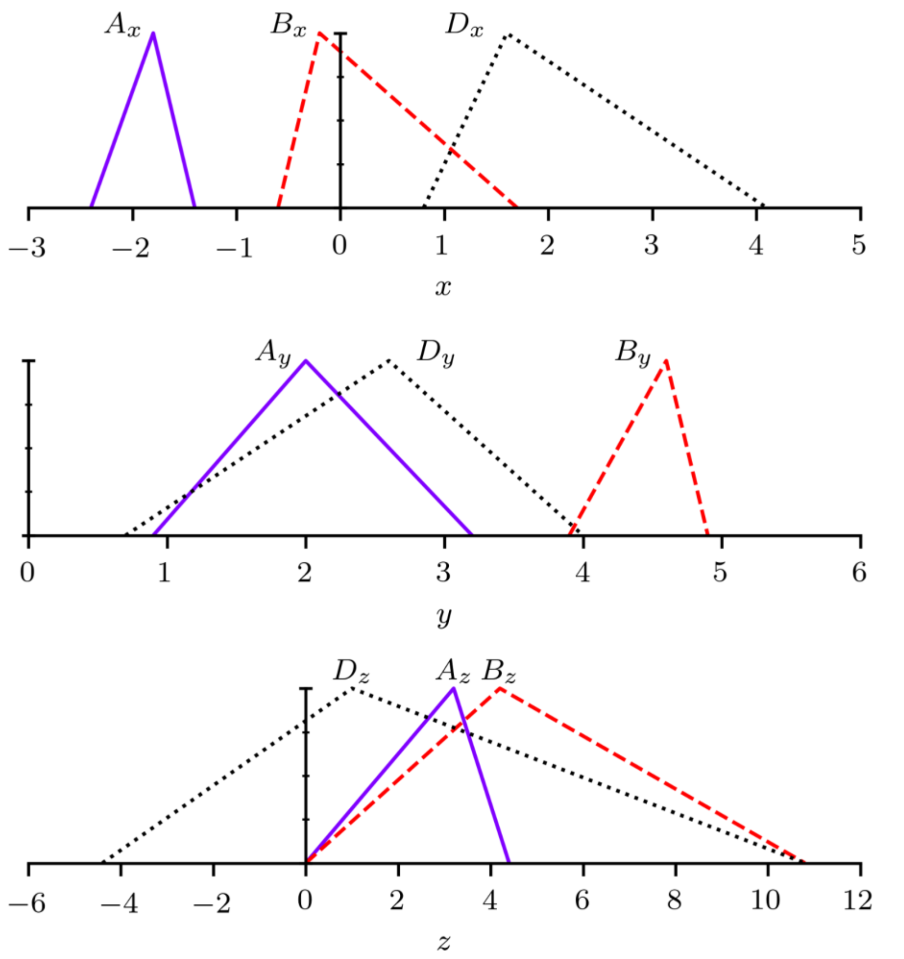
- Using fuzzy arithmetic, the difference between the two objects is computed along each dimension as a new TFN.
- This represents the minimum, maximum, and average distance between objects A and B in each dimension.
- (can be negative)

$$\boxed{A - B = \mathrm{Tri}(a_1 - b_3, a_2 - b_2, a_3 - b_1)}$$

$$\begin{aligned} A_x &= \mathrm{Tri}(-2.4, -1.8, -1.4) \\ A_y &= \mathrm{Tri}(0.9, 2.0, 3.2) \\ A_z &= \mathrm{Tri}(0.0, 3.2, 4.4) \end{aligned}$$

$$\begin{aligned} B_x &= \mathrm{Tri}(-0.6, -0.2, 1.7) \\ B_y &= \mathrm{Tri}(3.9, 4.6, 4.9) \\ B_z &= \mathrm{Tri}(0.0, 4.2, 10.8) \end{aligned}$$

$$\begin{aligned} D_x &= B_x - A_x = \mathrm{Tri}(0.8, 1.6, 4.1) \\ D_y &= B_y - A_y = \mathrm{Tri}(0.7, 2.6, 4.0) \\ D_z &= B_z - A_z = \mathrm{Tri}(-4.4, 1.0, 10.8) \end{aligned}$$



Overall Distance

- The overall distance is computed as the Euclidean norm of the differences along each axis using fuzzy arithmetic.

$$
\begin{aligned}
D_x &= \mathrm{Tri}(0.8, 1.6, 4.1)\\
D_y &= \mathrm{Tri}(0.7, 2.6, 4.0)\\
D_z &= \mathrm{Tri}(-4.4, 1.0, 10.8)
\end{aligned}
$$

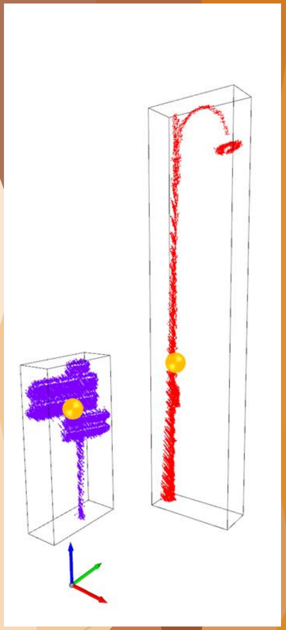
$$
\begin{aligned}
D_x^2 &= \mathrm{Tri}(0.64, 2.56, 16.81)\\
D_y^2 &= \mathrm{Tri}(0.49, 6.76, 16.0)\\
D_z^2 &= \mathrm{Tri}(0.0, 1.0, 116.64)
\end{aligned}
$$

$$A^2 = \mathrm{Tri}\left(a_{\min}, a_2^2, \max\{a_1^2, a_3^2\}\right),$$

$$a_{\min} = \begin{cases} \min\{0, a_1^2, a_3^2\}, & \text{if } a_1 \le 0 \le a_3 \\ \min\{a_1^2, a_3^2\}, & \text{otherwise} \end{cases}$$

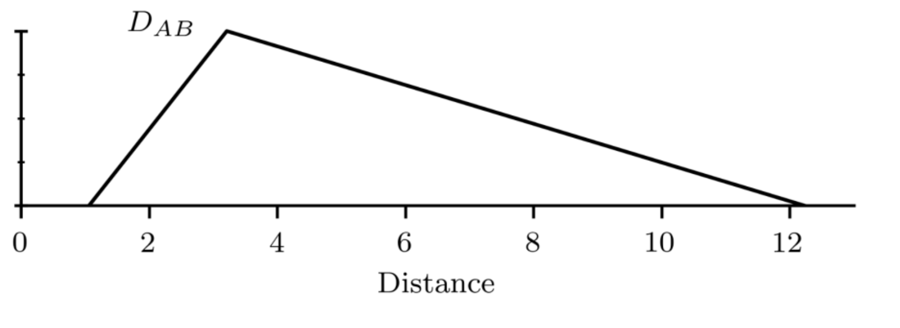
$$A + B = \mathrm{Tri}(a_1 + b_1, a_2 + b_2, a_3 + b_3)$$

$$\sqrt{A} = \mathrm{Tri}(\sqrt{a_1}, \sqrt{a_2}, \sqrt{a_3}), \qquad 0 \le a_1 \le a_2 \le a_3$$



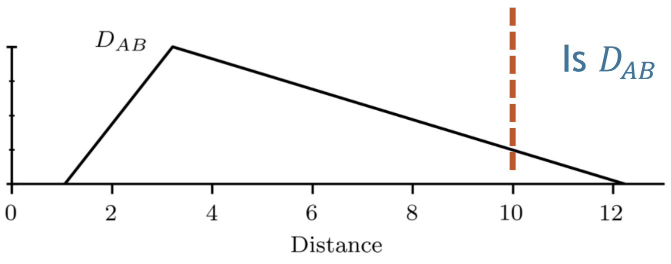
$$D_x^2 + D_y^2 + D_z^2 = \mathrm{Tri}(1.13, 10.32, 149.45)$$

$$D_{AB} = \sqrt{D_x^2 + D_y^2 + D_z^2} = \mathrm{Tri}(1.06, 3.21, 12.22)$$

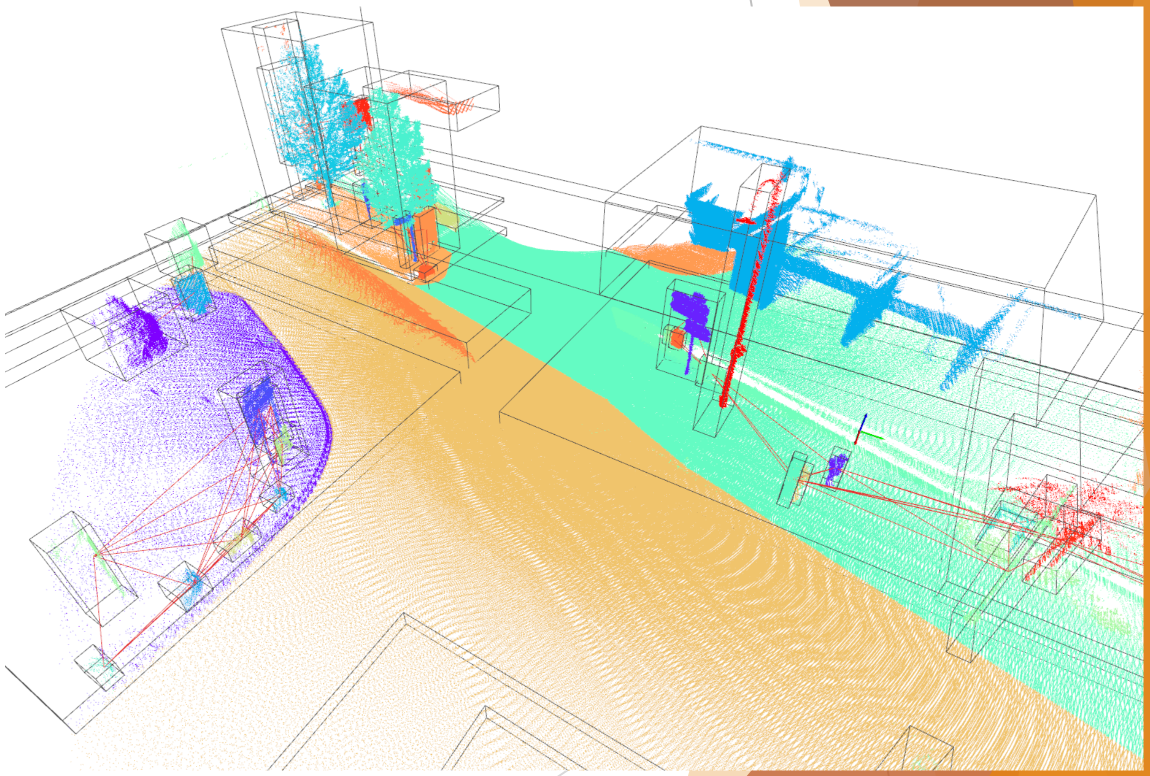


Spatial Relationship Graph

- Knowing the distances between objects lets us define a spatial relationship graph over a scene to show how objects are connected.
- We'll add an edge between two nodes (objects) if the distance between them is less than some threshold d.
- So, we need a way to determine if a triangular fuzzy number represents a distance that is less than d.



Is D_{AB} less than 10?



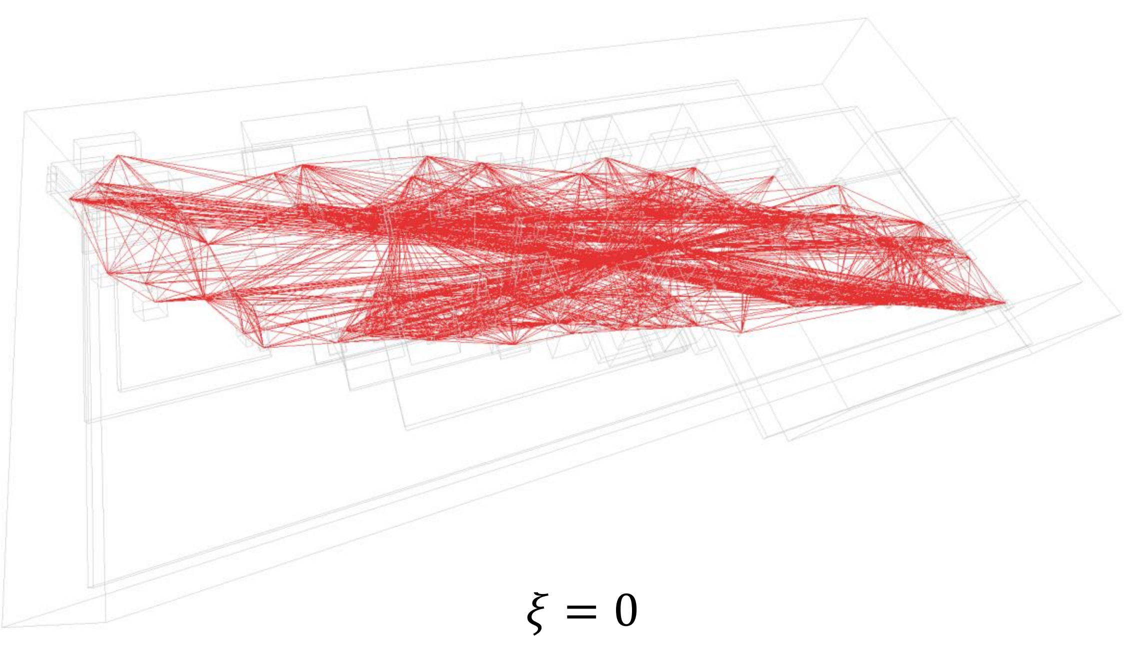
TFN Defuzzification

- Given a triangular fuzzy number $X = \mathrm{Tri}(a, b, c)$, we can defuzzify to a crisp value using an optimism/pessimism parameter $\xi \in [0,1]$.

$$\Gamma(X|\xi) = \begin{cases} a + 2\xi(b-a), & \xi \le 0.5 \\ b + 2(\xi - 0.5)(c-b), & \xi > 0.5 \end{cases}$$

- This gives a way to select the minimum ($\xi = 0$), maximum ($\xi = 1$), or average ($\xi = 0.5$) values of the TFN.
- When ξ is high, it's like complete linkage clustering.
- When ξ is low, it's like single linkage clustering.

$d = 10$ meters



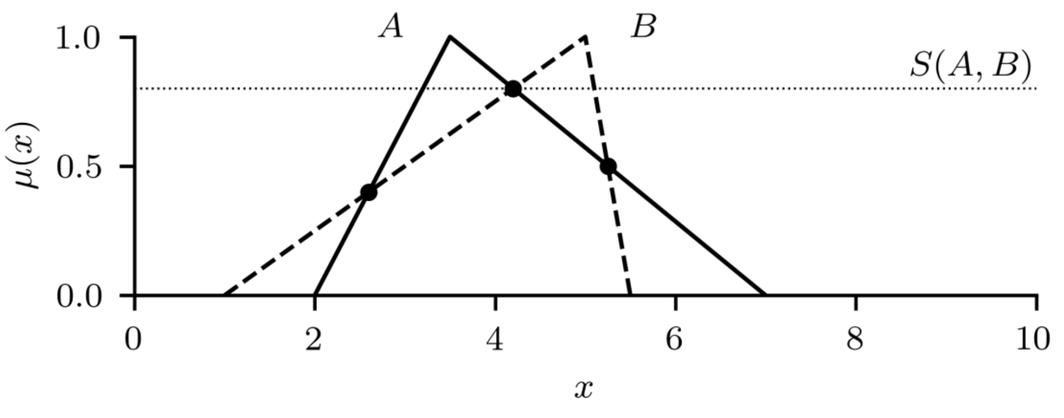
University of Missouri

Distance Queries

- Suppose we want to find all objects that are a certain distance away from a reference object.
- We define the query distance as a TFN $Q = \mathrm{Tri}(q_1, q_2, q_3)$.
- The similarity between two TFNs can be computed as the maximum of their intersecting points.

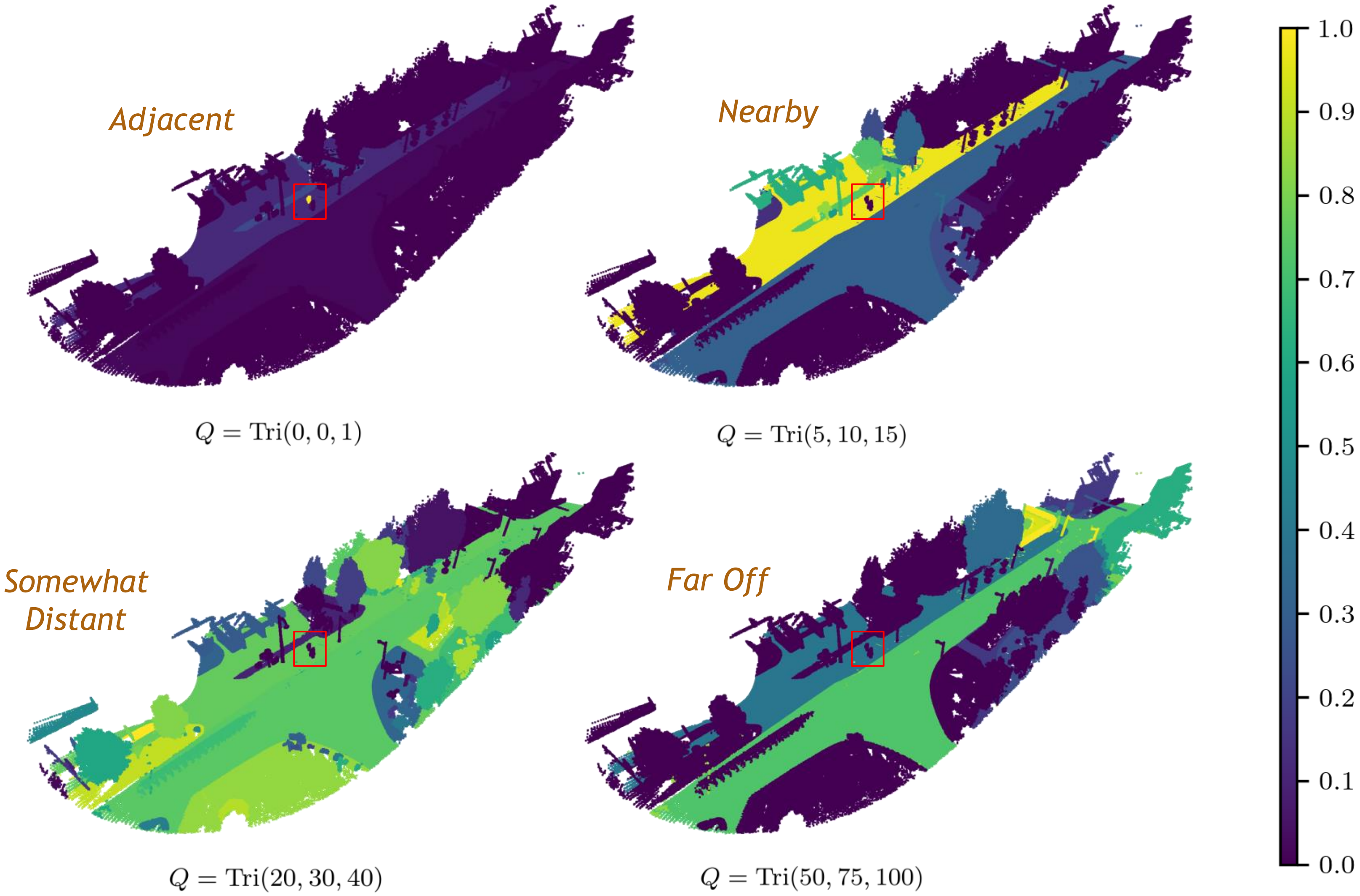
$$\boxed{S(A,B) = \max_{x \in \mathbb{R}} \{\min(\mu_A(x), \mu_B(x))\}}$$

- The distance D_{AB} between objects A and B can be compared with the query distance Q to give the distance similarity $s_{\mathrm{dist}} \in [0,1]$.



University of Missouri

Example Distance Queries



Looking at the heat maps for different distance queries from a person in the scene, outlined with a red box.

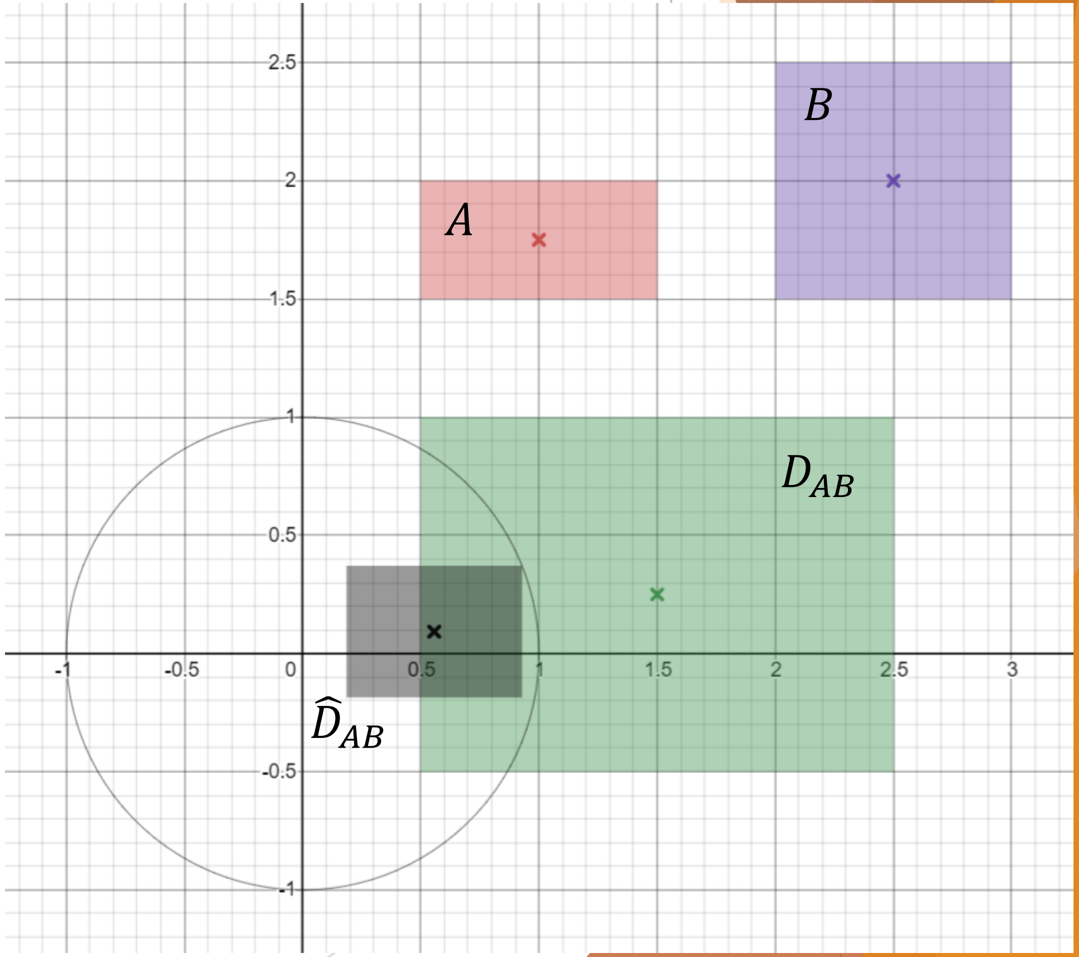
University of Missouri

Normalized Direction

- The axis-aligned difference TFNs D_x, D_y, and D_z encode both the relative ***distance*** and ***direction*** between two objects.
- We can use this to compute how much support there is for the statement "Object B is in direction $\hat{\boldsymbol{u}}$ from Object A," where $\hat{\boldsymbol{u}}$ is a unit vector pointing in the direction of interest.
- First, we need a normalized difference vector, $\widehat{\boldsymbol{D}} = \left[\widehat{D}_x, \widehat{D}_y, \widehat{D}_z\right]$, where $\widehat{D}_x$, $\widehat{D}_y$, and $\widehat{D}_z$ are normalized versions of the computed difference TFNs D_x, D_y, and D_z.

$$\widehat{D}_k = \alpha D_k, \qquad \text{s.t.} \max_{k\in\{x,y,z\}} \widehat{D}_k = 1$$

- Consider a 2-dimensional example…



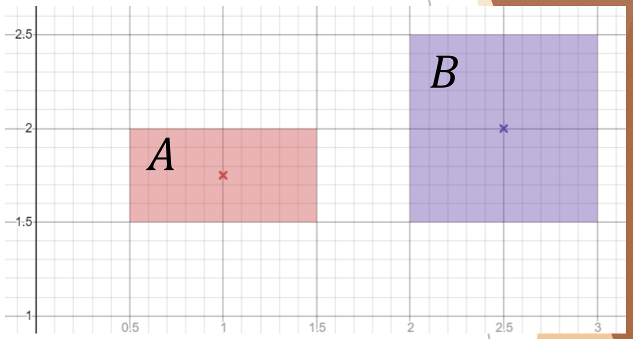
Directional Similarity

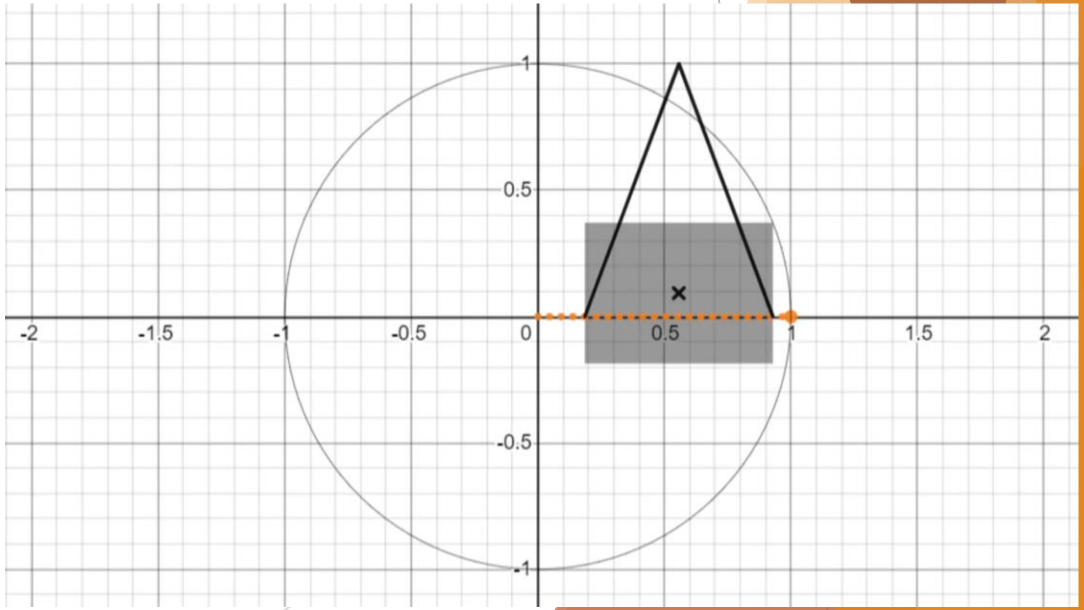
- Given a reference direction $\hat{\boldsymbol{u}}$, the directional similarity to the normalized difference TFN $\widehat{\boldsymbol{D}}$ is the dot product.

$$\boxed{S_{\text{dir}} = \widehat{\boldsymbol{D}} \cdot \hat{\boldsymbol{u}} = \widehat{D}_x u_x + \widehat{D}_y u_y + \widehat{D}_z u_z}$$

- S_{dir} is a TFN bounded in the range $[-1, 1]$.
  - See example…
- To reduce the directional similarity to a scalar value (like distance), we can use the defuzzification parameter ξ and clamp to positive values.

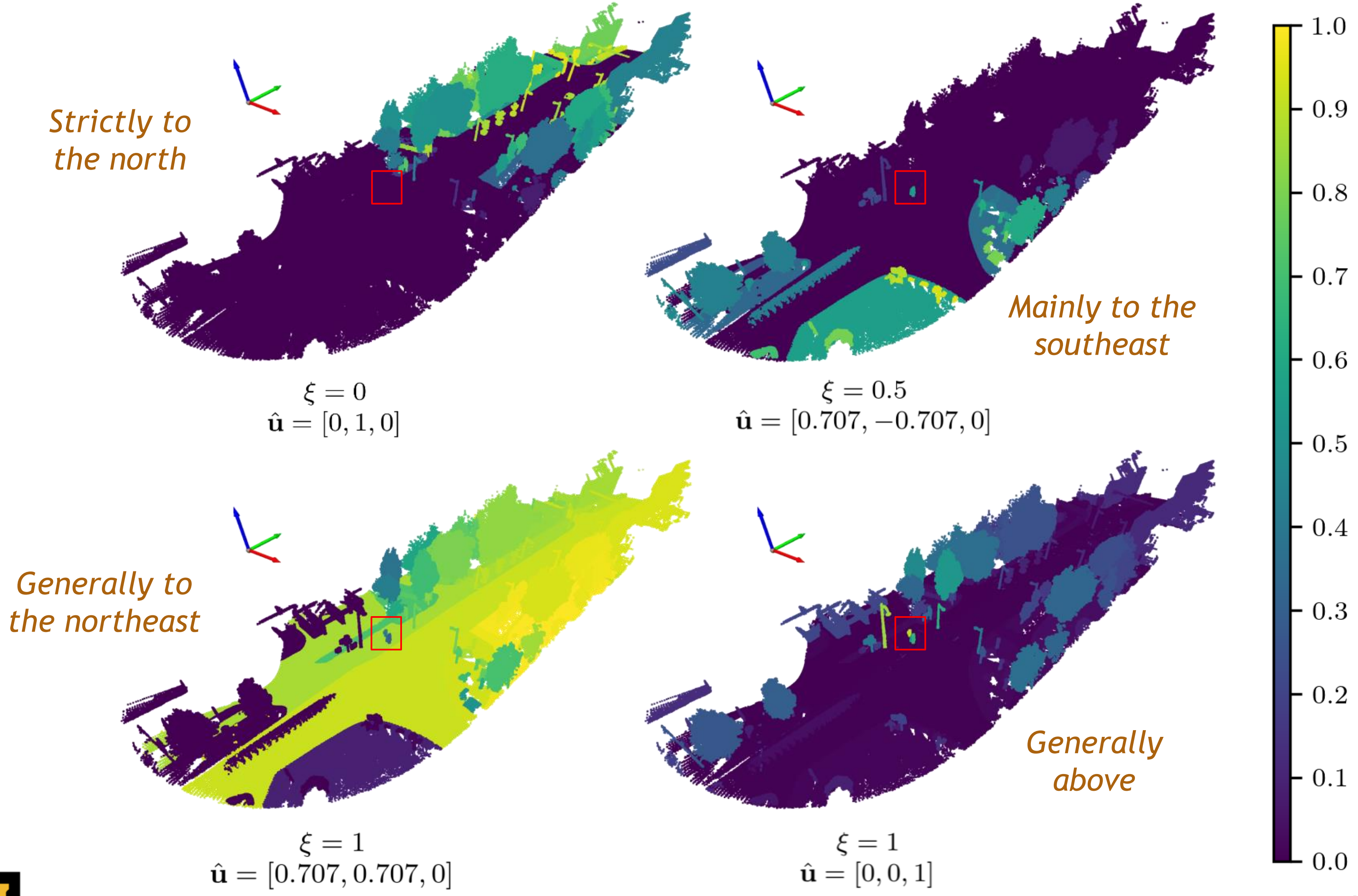
$$\boxed{s_{\text{dir}} = \max\{0, \Gamma(S_{\text{dir}}|\xi)\}}$$





University of Missouri

Example Directional Queries



Looking at the heat maps for different directional queries from a person in the scene, outlined with a red box.

University of Missouri

A Multi-Criteria Framework

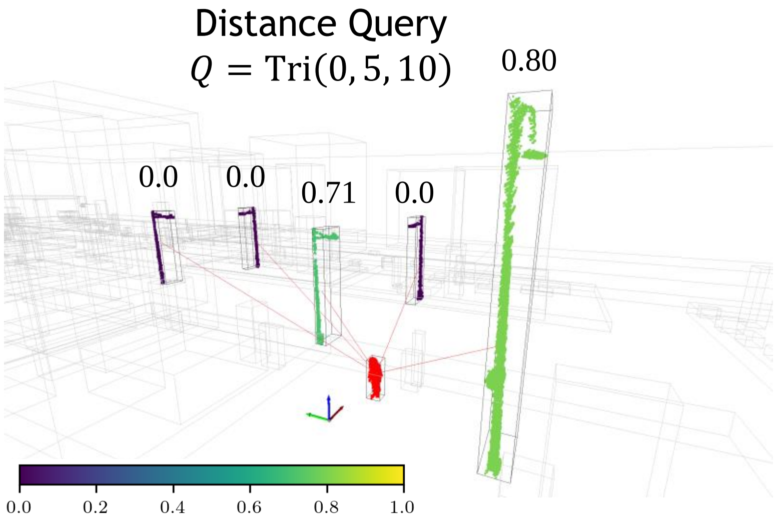
- Distance and direction are two features that can be used to search for objects in a scene.
- Other features might include class type, number of neighbors of a certain type, location in world space, etc.
- Our criteria for object selection is represented by a normalized feature vector $\boldsymbol{s} = [s_1, \dots, s_n]$, where each $s_i \in [0, 1]$ represents the degree to which an object satisfies a particular set of criteria.
- The multidimensional feature vector can be mapped to a single value with a scalarization function $g_\theta(\boldsymbol{s})$, where θ represents the parameterization.

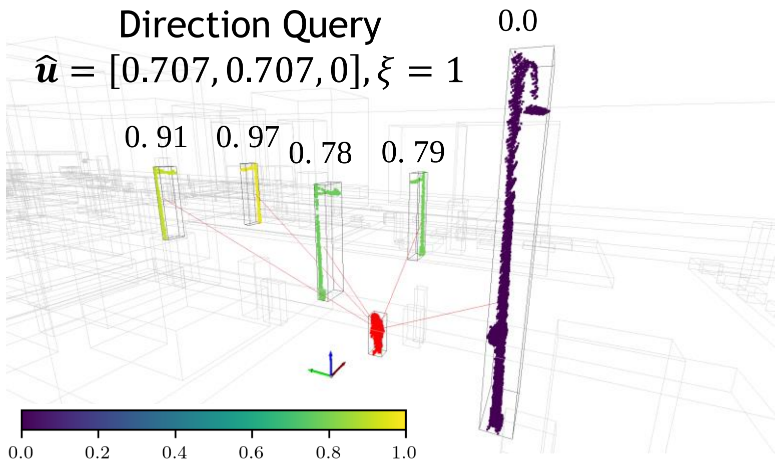
$$g_{\text{avg}}(\boldsymbol{s}) = \frac{1}{n}\sum_{i=1}^{n} s_i$$

$$g_{\min}(\boldsymbol{s}) = \min_i s_i$$

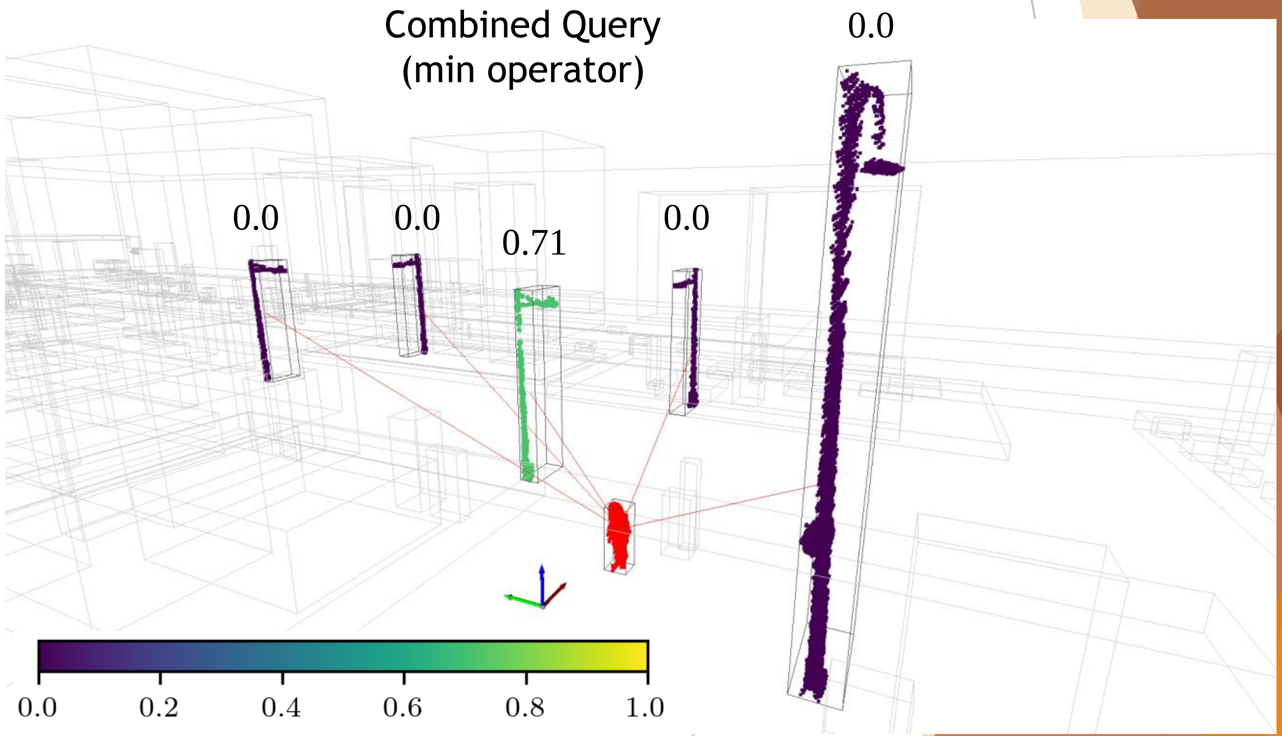
University of Missouri

Example: Multi-Criteria Selection



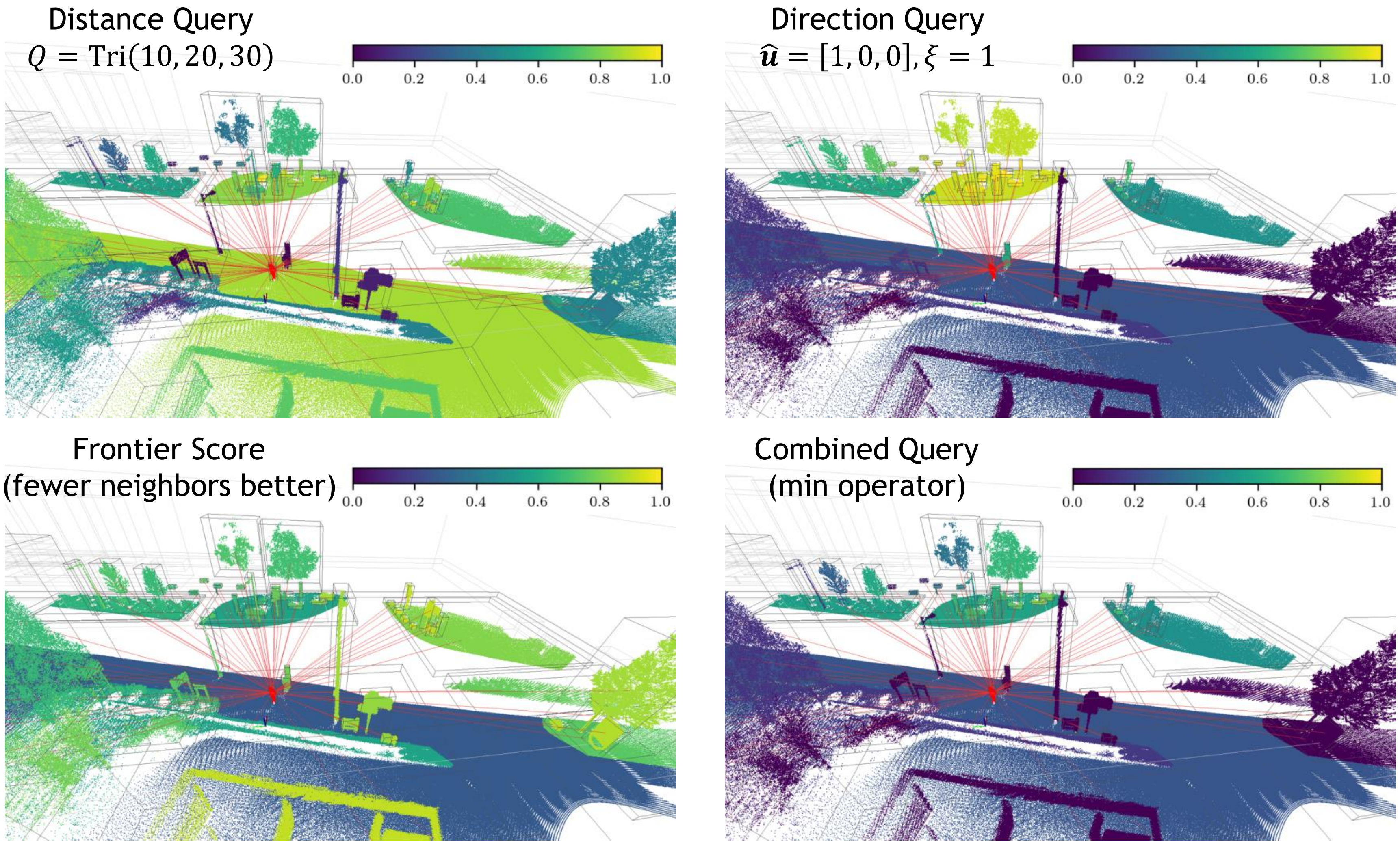


- Consider a person (red) looking to identify a lamp post that is "near" and to the "front-left".



University of Missouri

Example: Choosing an Exploration Target



Consider an agent in the environment choosing its next exploration target.

Want to select an object to interrogate that is
- Nearby
- In the forward direction
- Near the edge of what's already been explored

Conclusions

- We've shown a way to compute distance and directional spatial relationships between objects in a 3D scene represented with axis-aligned bounding boxes.
- These features can be used to construct a spatially attributed graph of the environment and search for objects using multiple criteria.
- Simplifying the representation to bounding boxes instead of full point clouds helps achieve real-time performance on embedded hardware.
- Future directions:
  - Handle dynamic environments and an incremental/updating graph
  - Integrate with semantic segmentation algorithms for point clouds
  - Use a hierarchical representation to represent large compound objects
    - E.g., buildings with windows and doorways, roads and intersections, etc.