Finding Probabilistic Nearest Neighbors for Query Objects with Imprecise Locations

Yuichi Iijima and Yoshiharu Ishikawa
Nagoya University, Japan

Outline

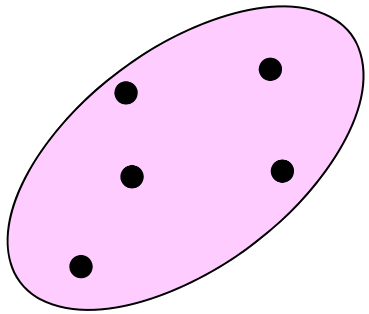
- Background and Problem Formulation
- Related Work
- Query Processing Strategies
- Experimental Results
- Conclusions

Imprecise Location Information

- Sensor Environments
  - Measurement Noise
  - Frequent updates may not be possible
    - GPS-based positioning consumes batteries

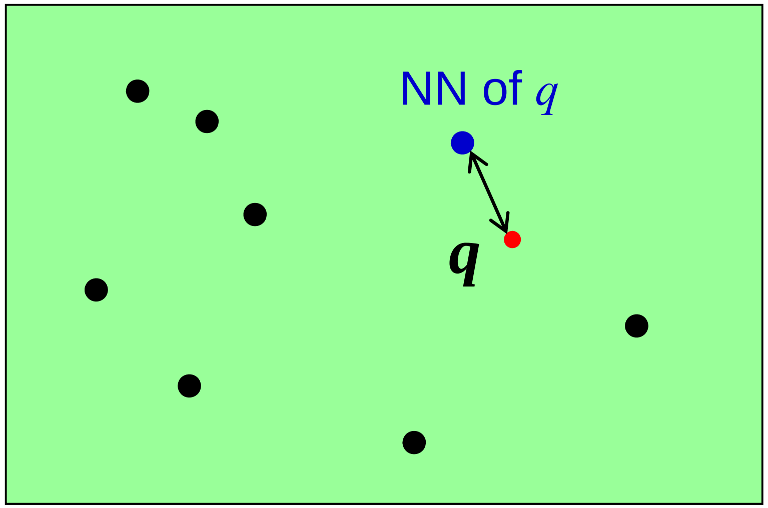
- Robotics
  - Localization using sensing and movement histories
  - Probabilistic approach has vagueness

- Privacy
  - Location Anonymity



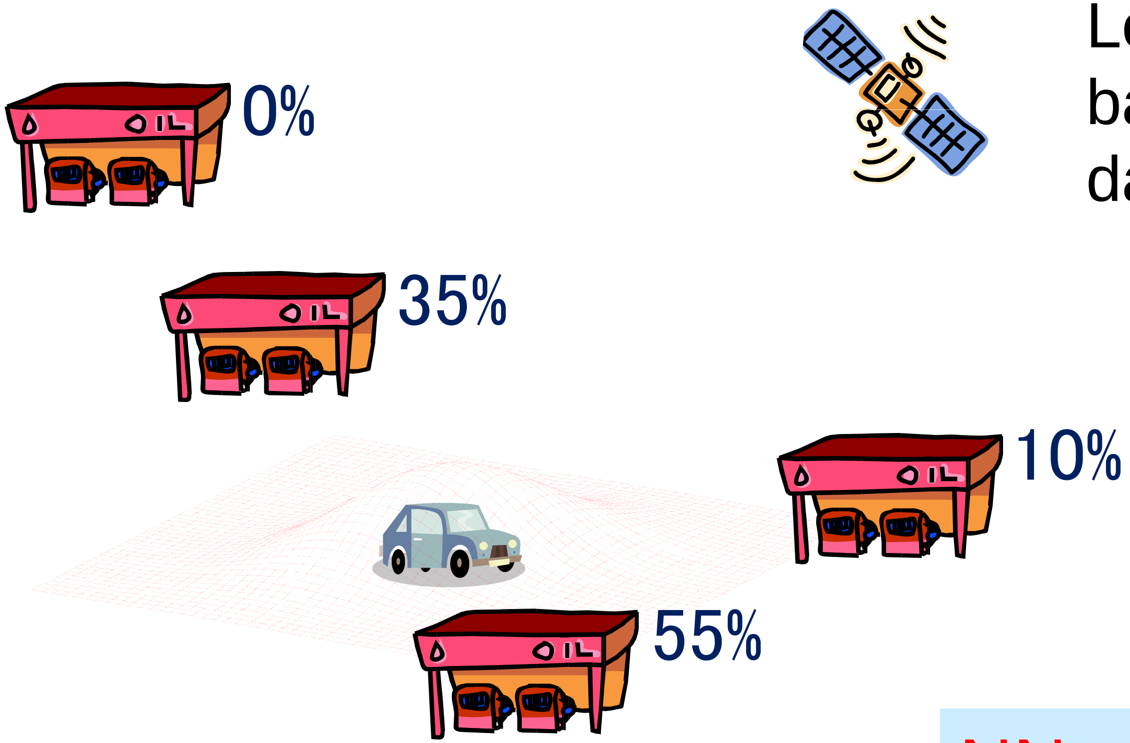
Nearest Neighbor Queries

- Nearest Neighbor Queries
  - Example: Find the closest bus stop
  - Traditional problem in spatial databases
    - Efficient query processing using spatial indices
    - Extensible to multi-dimensional cases (e.g., image retrieval)
- What happen if the location of query object is uncertain?



Example Scenario (1)

- Query: Find the nearest gas station



Location estimation based on noisy GPS data

Nearest gas station depends on the possible car locations

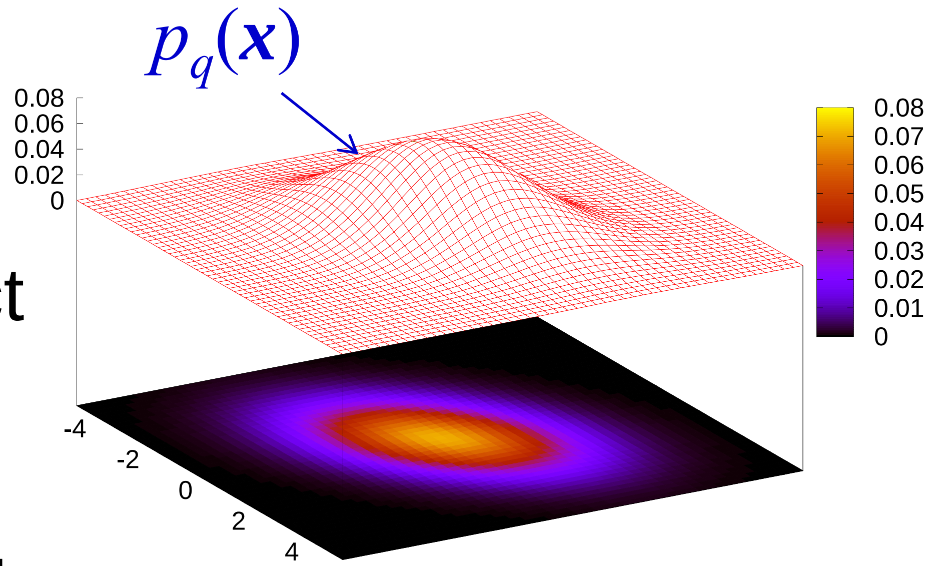
NN objects are defined in a probabilistic way

Example Scenario (2)

- Mobile Robotics
  - Location of the robot is estimated based on movement histories and sensor data
  - Measurements are noisy
  - Localization based on probabilistic modeling
    - Kalman filter, particle filter, etc.
  - Estimated location is given as a probabilistic density function (PDF)
    - PDF changes on each estimation

Probabilistic Nearest Neighbor Query (1)

- PNNQ for short
- Assumptions
  - Location of query object q is specified as a Gaussian distribution
  - Target data: static points



- Gaussian Distribution

$$p_q(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2} |\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2} (\boldsymbol{x} - \boldsymbol{q})^t \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{q}) \right]$$

  - $\boldsymbol{\Sigma}$: Covariance matrix

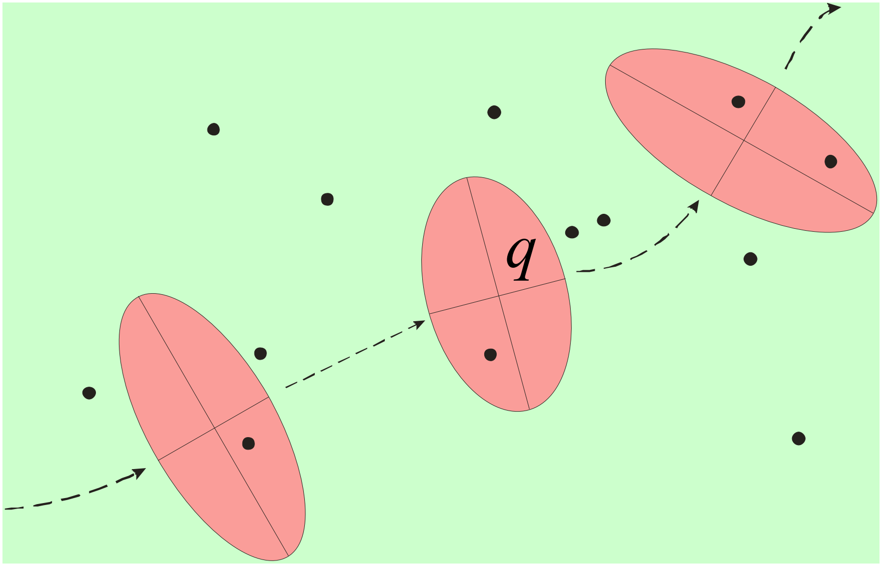
Probabilistic Nearest Neighbor Query (2)

- Definition

$$\Pr_{NN}(q, o) = \Pr(\forall o' \in O, o' \neq o, \|\mathbf{x} - \mathbf{o}\| \leq \|\mathbf{x} - \mathbf{o}'\|)$$

$$PNNQ(q, \theta) = \{o \mid o \in O, \Pr_{NN}(q, o) \geq \theta\}$$

- Find objects which satisfy the condition
  - The probability that the object is the nearest neighbor of q is greater than or equal to θ



Outline

- Background and Problem Formulation
- Related Work
- Query Processing Strategies
- Experimental Results
- Conclusions

Related Work

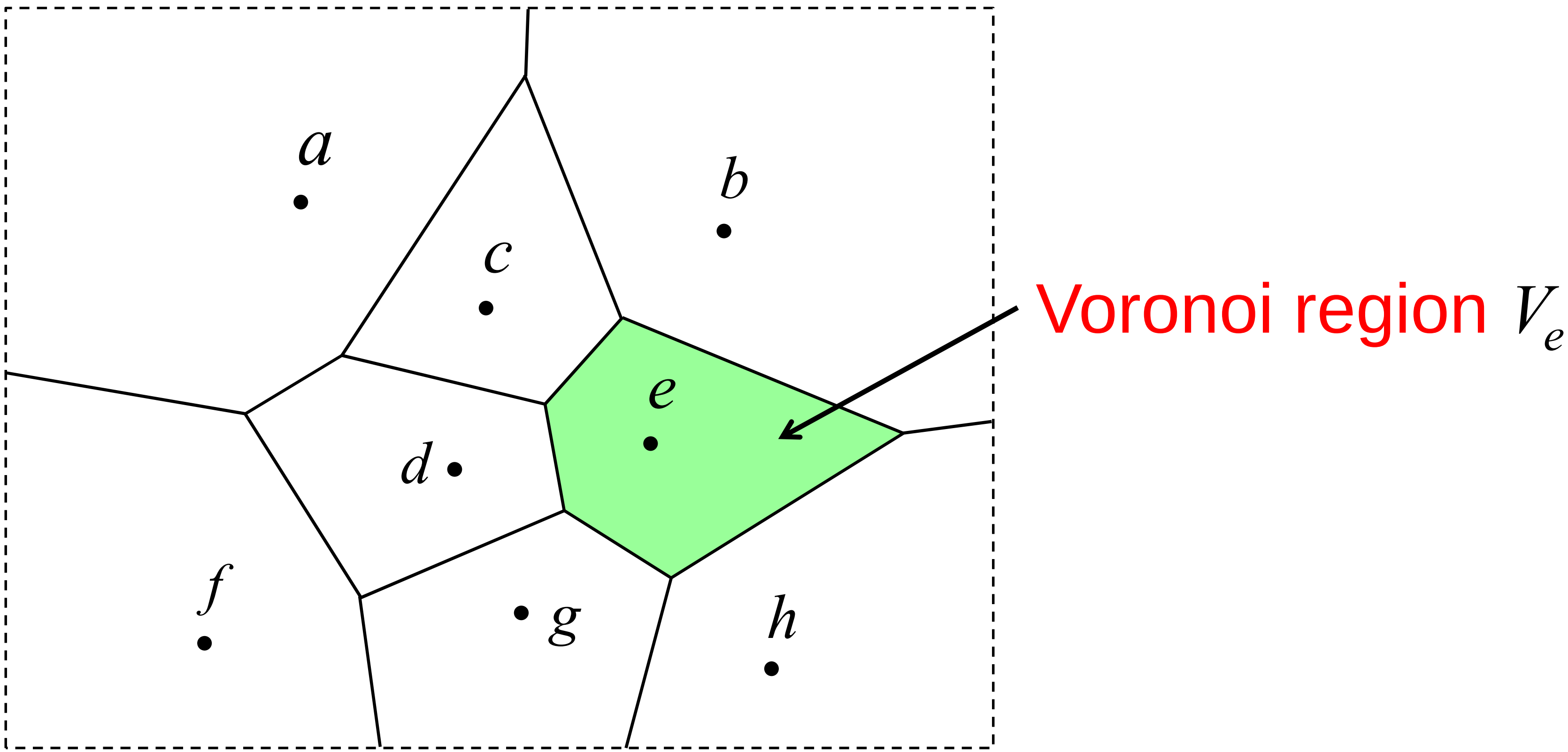
- Query processing methods for uncertain (location) data
  - Cheng, Prabhakar, et al. (SIGMOD'03, VLDB'04, …)
  - Tao et al. (VLDB'05, TODS'07)
  - Consider arbitrary PDFs or uniform PDFs
  - Most of the case, target objects are imprecise
- Research related to Gaussian distribution
  - Gauss-tree [Böhm et al., ICDE'06]
    - Target objects are based on Gaussian distributions
- Our former work
  - Ishikawa, Iijima, Yu (ICDE'09): Probabilistic range queries

Outline

- Background and Problem Formulation
- Related Work
- Query Processing Strategies
- Experimental Results
- Conclusions

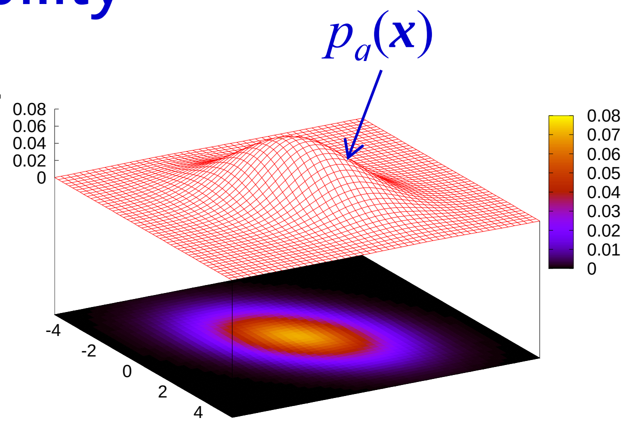
Naïve Approach (1)

- Use of Voronoi Diagram
  - Well-known method for standard (non-imprecise) nearest neighbor queries



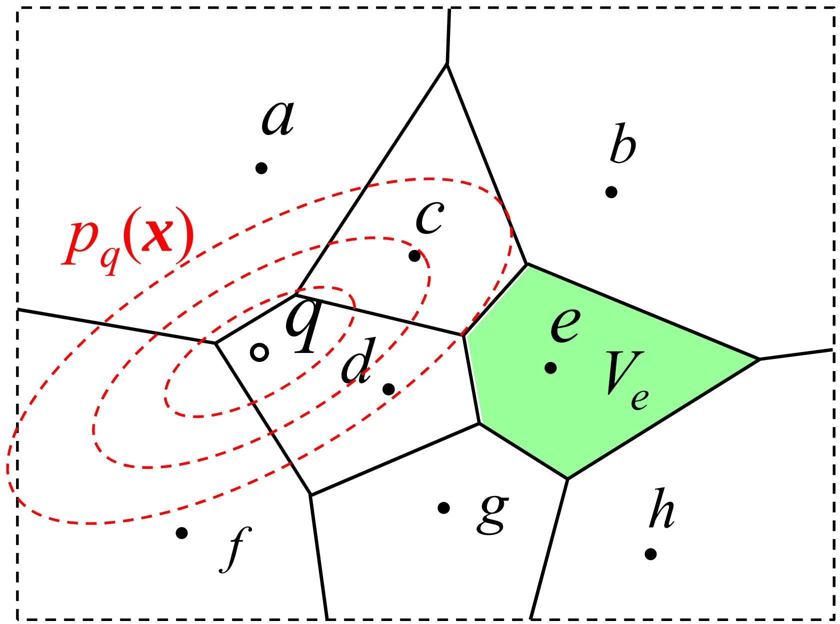
Naïve Approach (2)

- $\Pr_{NN}(q, o)$: Nearest neighbor probability
  - Probability that object o is the nearest neighbor of query object q
  - It can be computed by integrating the probability density function $p_q(\mathbf{x})$ over Voronoi region V_o



$$\Pr_{NN}(q,o) = \int_{V_o} p_q(\mathbf{x}) d\mathbf{x}$$

- Problem
  - Need to consider all target objects
  - Numerical integration (Monte Carlo method) is quite costly!



Our Approach

- Outline of processing
  1. Filtering
     - Prune non-candidate objects whose $\Pr_{NN}$ are obviously smaller than the threshold θ
     - Low-cost filtering conditions
  2. Numerical integration for the remaining candidate objects
- Two strategies
  – θ-region-based Approach
  – SES-based Approach
    - SES: Smallest Enclosing Sphere

Strategy 1: θ-Region-Based Approach (1)

- θ-region
  - Similar concepts are often used in query processing for uncertain spatial databases
  - Definition: Ellipsoidal region for which the result of the integration becomes $1-2\theta$:

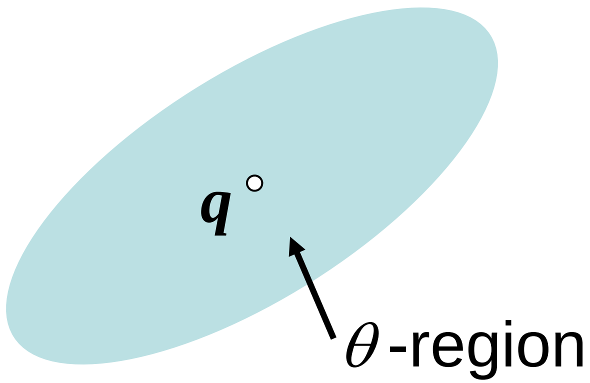
$$\int_{(\boldsymbol{x}-\boldsymbol{q})^t \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{q}) \le r_\theta^2} p_q(\boldsymbol{x}) d\boldsymbol{x} = 1 - 2\theta$$

The ellipsoidal region

$$(\boldsymbol{x}-\boldsymbol{q})^t \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{q}) \le r_\theta^2$$

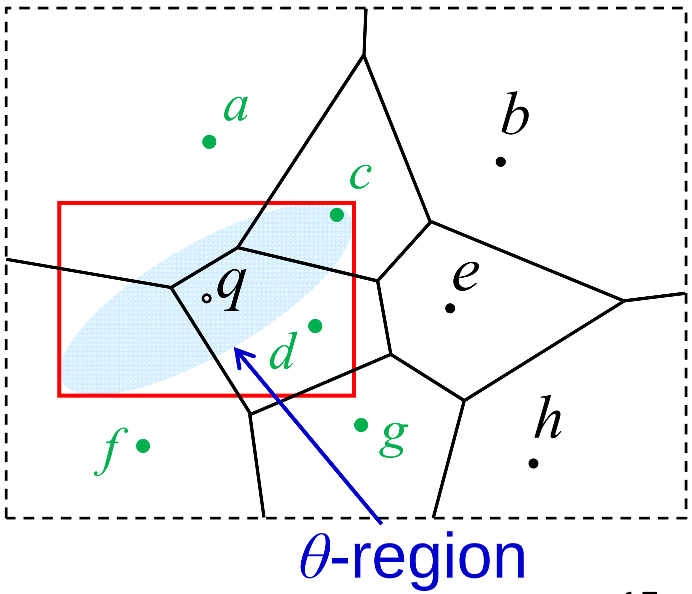
is the θ-region

- Example: θ is specified as 1%, we consider 98% θ-region



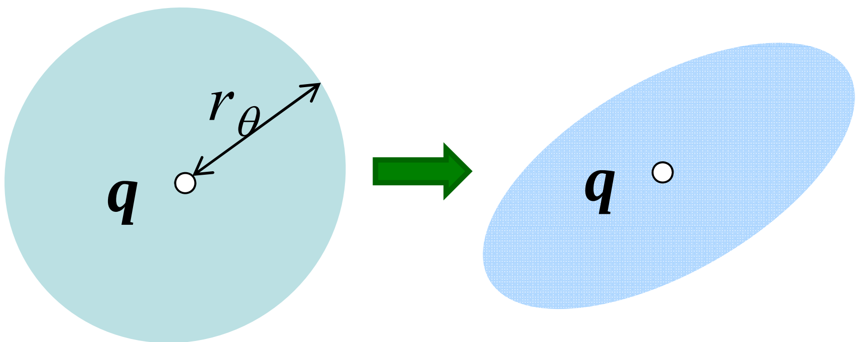
Strategy 1: θ-Region-Based Approach (2)

- Query Processing
  1. θ-region for the query is computed at first
     - θ-region can be derived using r_θ-table:
       - It is constructed for the normal Gaussian ($\Sigma = \mathbf{I}, \boldsymbol{q} = \mathbf{0}$): Given θ, it returns appropriate r_θ
     - Final θ-region can be obtained by transformation
  2. Derive the bounding box of θ-region
  3. Objects whose Voronoi regions overlaps with the box are the candidates
     - $\{a, c, d, f, g\}$, in this example



Strategy 1: θ-Region-Based Approach (3)

- Derivation details
- First, we consider the normal Gaussian
  - Using r_θ-table, we can get the appropriate r_θ for given θ
- Second, a spherical θ-region is derived based on transformation
  - Transformation is performed by analyzing Σ

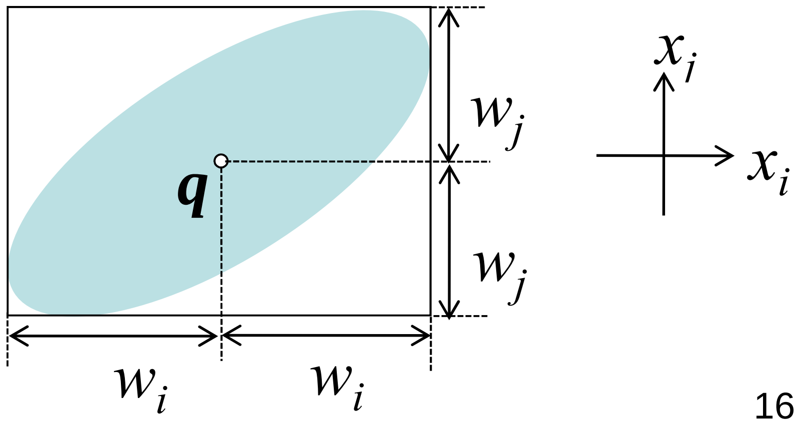


- Third, the bonding box is calculated

$$w_i = \sigma_i r_\theta$$

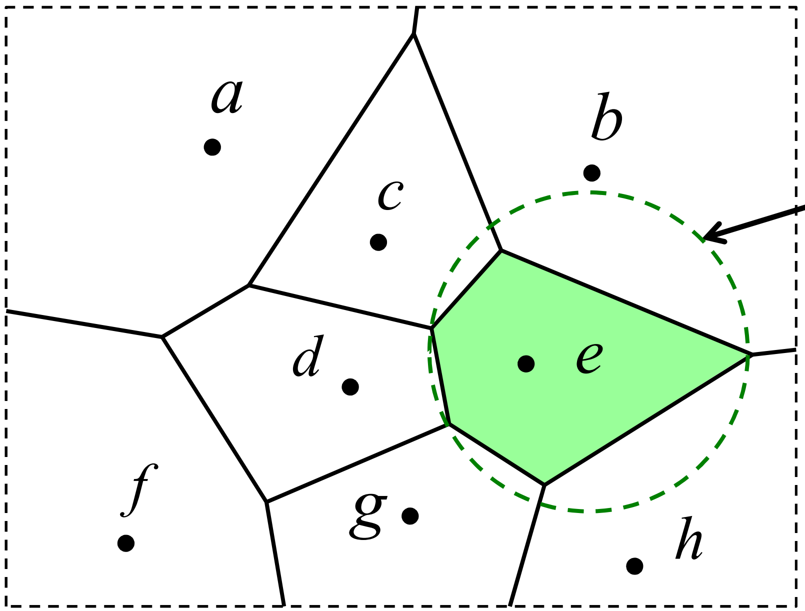
$$\sigma_i = \sqrt{(\boldsymbol{\Sigma})_{ii}}$$

where $(\boldsymbol{\Sigma})_{ii}$ is the (i, i) entry of $\boldsymbol{\Sigma}$



Strategy 2: SES-Based Approach (1)

- SES: Smallest Enclosing Sphere
  - For each Voronoi region V_o, we compute its SES SES_o beforehand



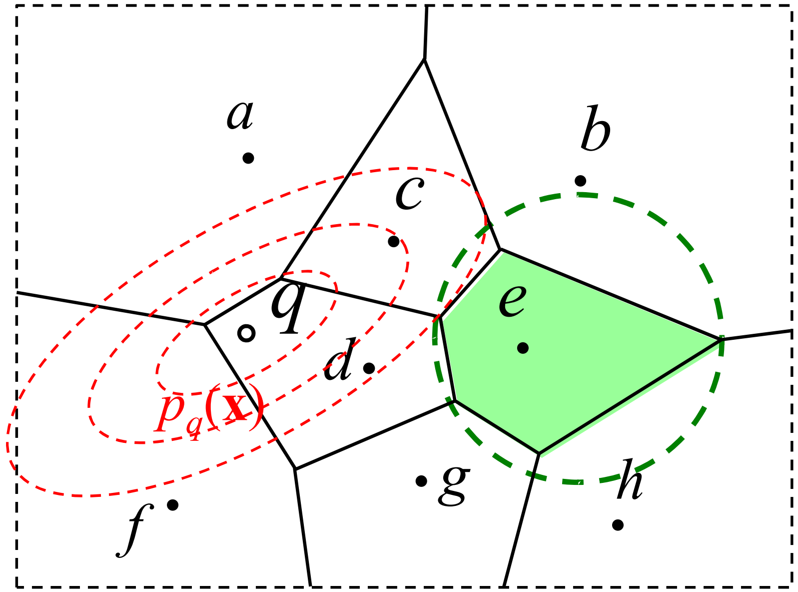
SES_e: SES for Voronoi region of e

Strategy 2: SES-Based Approach (2)

- Integration over SES_o gives the upper-bound for $\Pr_{NN}(q, o)$

$$\Pr_{NN}(q,o) = \int_{V_o} p_q(\mathbf{x})d\mathbf{x} < \int_{SES_o} p_q(\mathbf{x})d\mathbf{x}$$

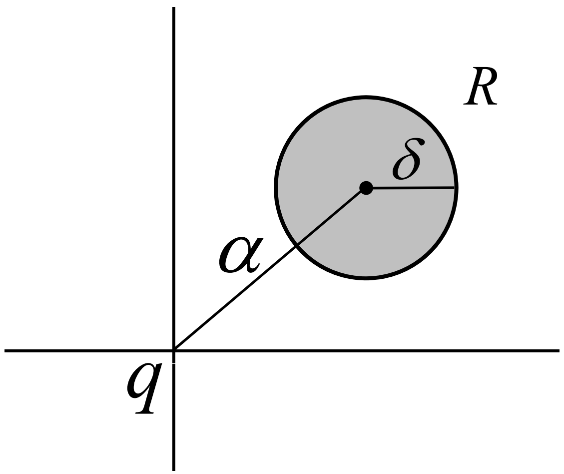
  – Integration over a sphere region is more easier to compute
    - We use a table called U-catalog constructed beforehand



Strategy 2: SES-Based Approach (3)

- What is U-catalog?
  - Given two parameters α and δ, it returns corresponding integral
  - U-catalog is made for different (α, δ) pairs by computing the integral of normal Gaussian over sphere region R

$$\pi(\alpha, \delta) = \int_{\boldsymbol{x} \in R} p_{\mathrm{norm}}(\boldsymbol{x}) d\boldsymbol{x}$$

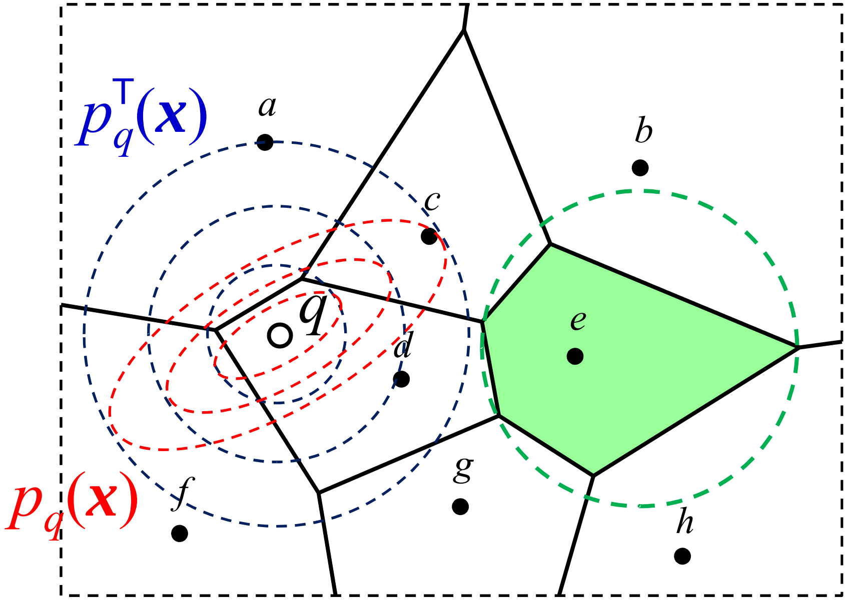


α	δ	$\pi(\alpha, \delta)$
0.0	0.1	…
0.0	0.2	…
…	…	…
1.0	0.1	
…	…	…

Strategy 2: SES-Based Approach (4)

- To use U-catalog, another approximation is required since it is only useful for normal Gaussian
  - Use of upper-bounding function $p_q^{\top}(\mathbf{x})$
  - $p_q^{\top}(\mathbf{x})$ tightly bounds $p_q(\mathbf{x})$ and has spherical isosurfaces
  - For $p_q^{\top}(\mathbf{x})$, we can easily derive its integral over SES_o using U-catalog
- In summary, we use two approximations:

$$\begin{aligned} \Pr_{NN}(q,o) &= \int_{V_o} p_q(\mathbf{x})d\mathbf{x} \\ &< \int_{SES_o} p_q(\mathbf{x})d\mathbf{x} \\ &\leq \int_{SES_o} p_q^{\top}(\mathbf{x})d\mathbf{x} \end{aligned}$$



Strategy 2: SES-Based Approach (5)

- Bounding Function
  - Original Gaussian PDF

$$p_q(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{q})^t \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{q})\right]$$

  - Upper-Bounding Function

$$p_q^{\top}(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{\lambda^{\top}}{2}\|\boldsymbol{x}-\boldsymbol{q}\|^2\right]$$

where

$$\boldsymbol{\Sigma}^{-1} = \sum_{i=1}^{d} \lambda_i \mathbf{v}_i \mathbf{v}_i^t$$

$$\lambda^{\top} = \min\{\lambda_i\}$$

$$p_q(\boldsymbol{x}) \le p_q^{\top}(\boldsymbol{x})$$

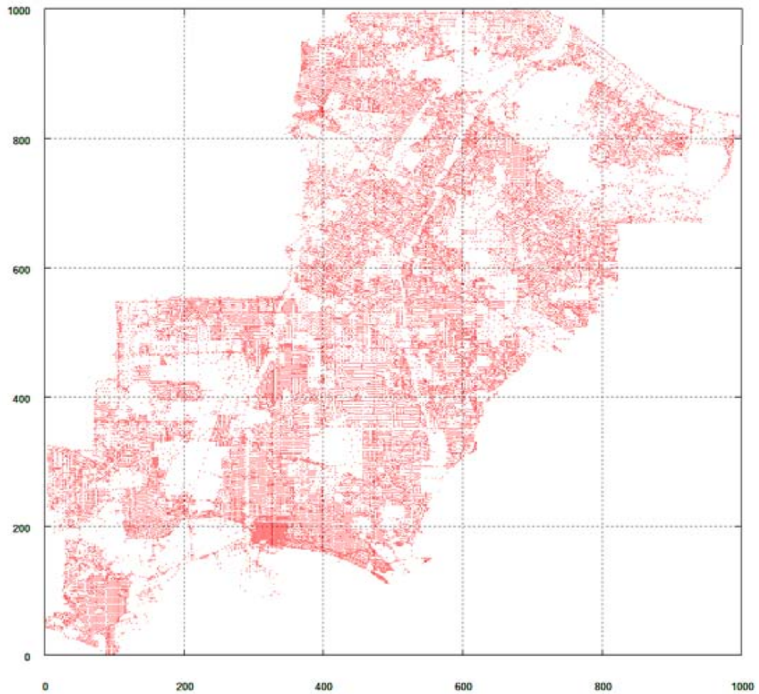
is satisfied

Outline

- Background and Problem Formulation
- Related Work
- Query Processing Strategies
- Experimental Results
- and Conclusions

Setup of Experiments (1)

- Target Data
  - 2D point data (50K entries)
  - Based on road line segments in Long Beach

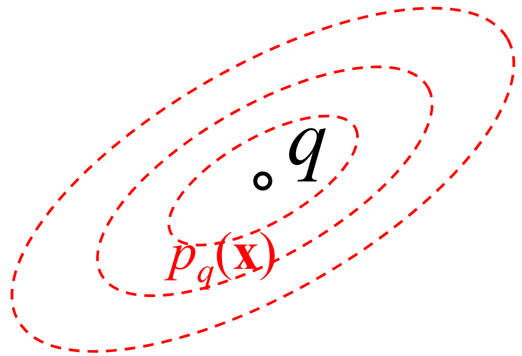


- Computation of Voronoi regions and SESs
  - LEDA package was used
- Comparison
  - Strategies 1, 2, and their hybrid approach
  - Evaluation metric: Response time

Setup of Experiments (2)

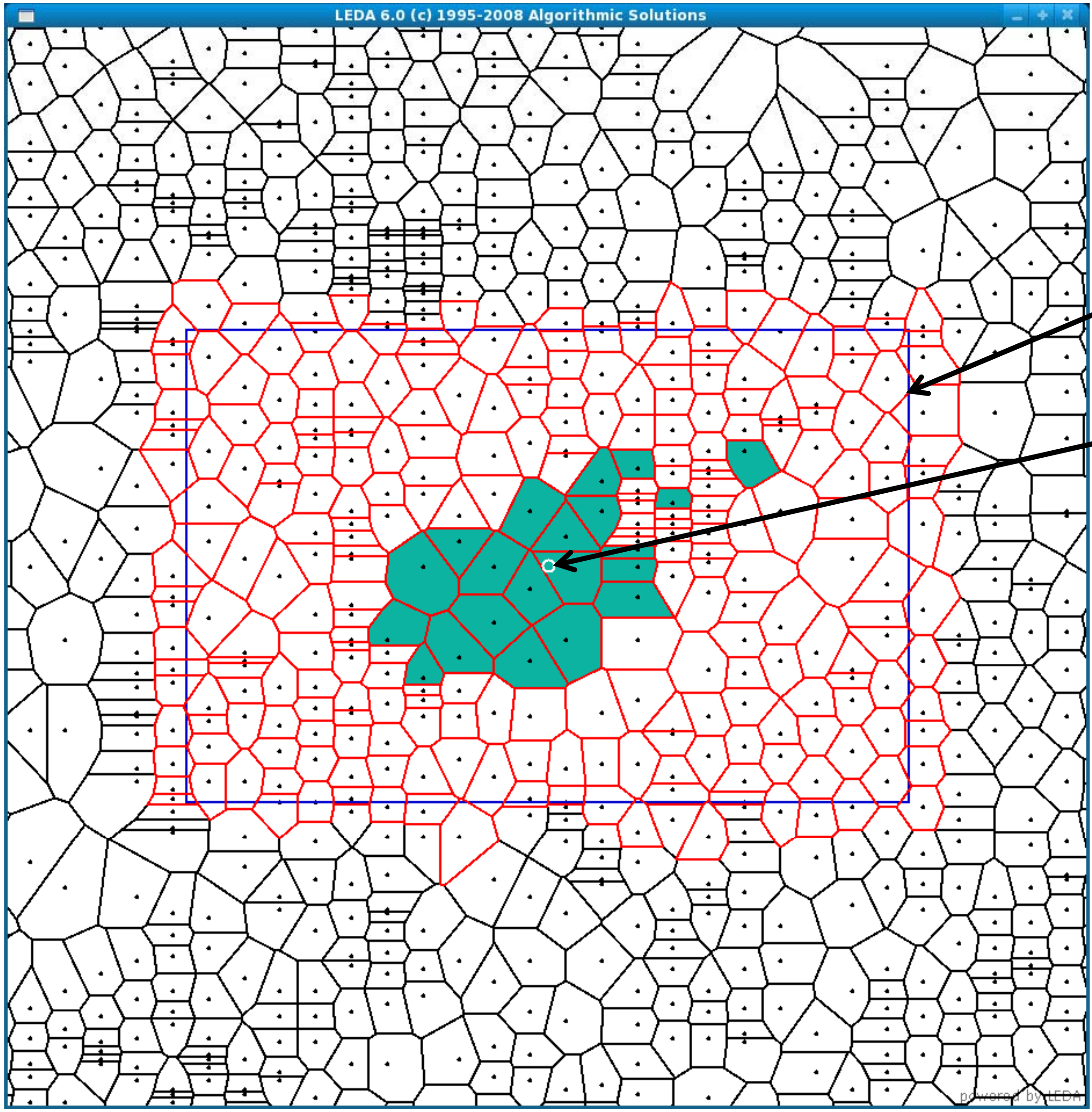
- Default Parameters
  - Covariance matrix

$$\mathbf{\Sigma} = \gamma \begin{bmatrix} 7 & 2\sqrt{3} \\ 2\sqrt{3} & 7 \end{bmatrix}$$



    - For this matrix the shape of the isosurface of $p_q(\mathbf{x})$ is an ellipse titled at 30 degrees and the major-to-minor axis ratio is 3:1
    - γ : Parameter for controlling vagueness (default: $\gamma = 10$)
  - Probability threshold value: $\theta = 0.01$
  - No. of samples for Monte Carlo method: 1,000,000

Candidate Objects in Strategy 1



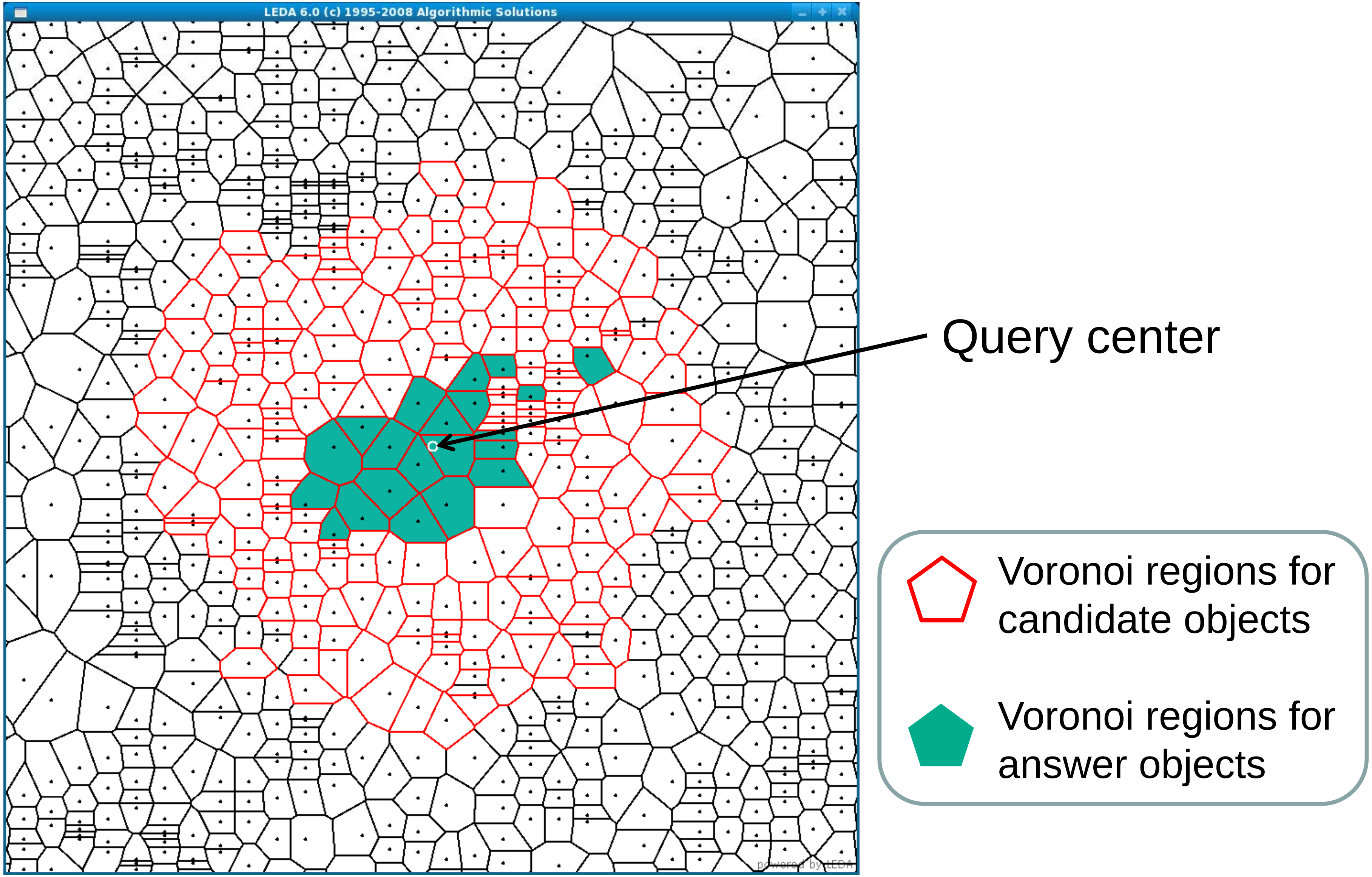
Bounding box of the θ-region

Query center

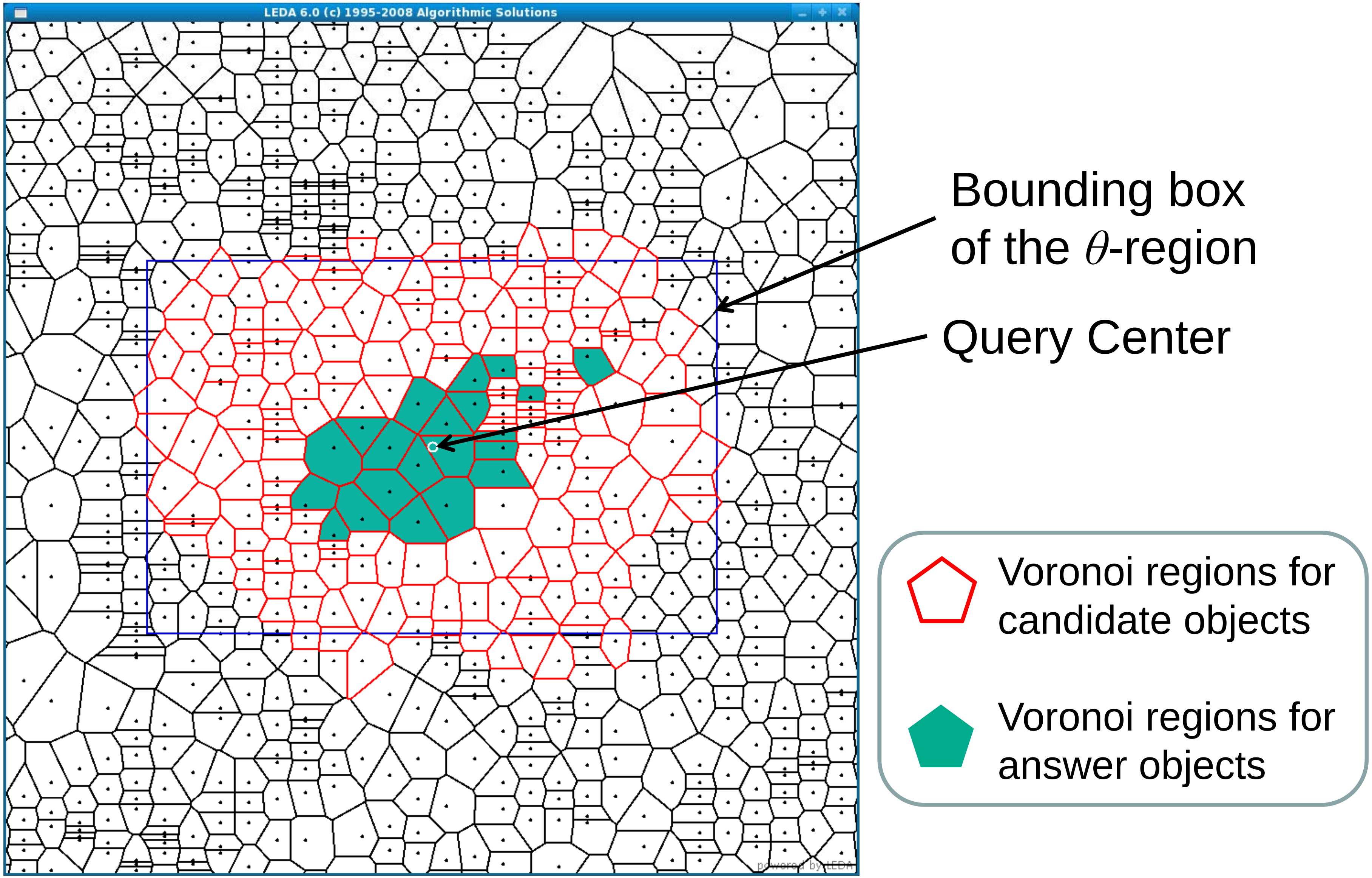
Voronoi regions for candidate objects

Voronoi regions for answer objects

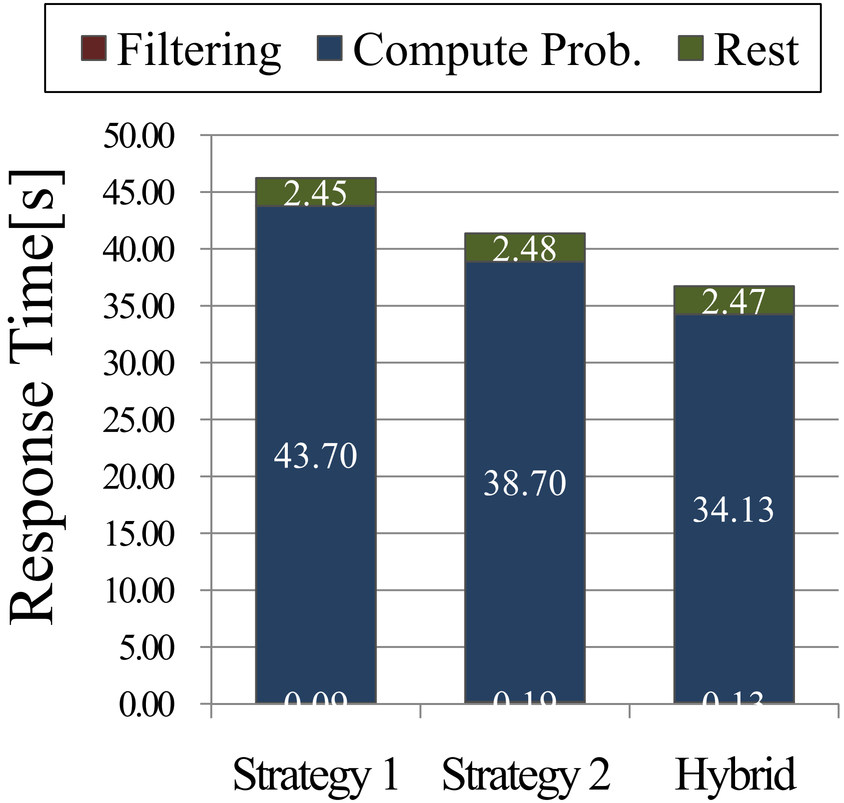
Candidate Objects in Strategy 2



Candidate Objects in Hybrid Strategy



Experimental Results - Default Parameters



	Strategy 1	Strategy 2	Hybrid
Number of candidates	179	150	129
Number of answers	26		

- Most of the processing time is spent in numerical integration

⇩

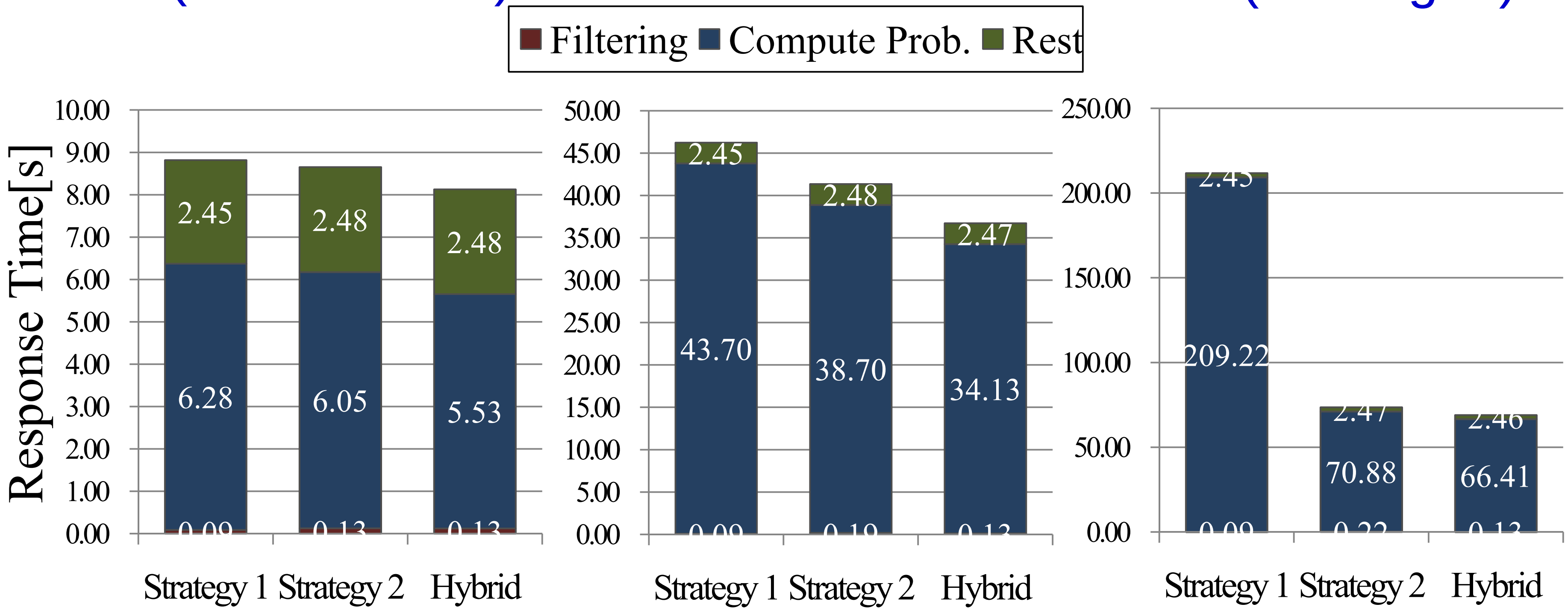
- A strategy that can prune more objects has better performance

Experimental Results - Different γ Values

$\gamma = 1$
(almost exact)

$\gamma = 10$

$\gamma = 50$
(too vague)



Number of candidates	24	31	23
Number of answers	8		

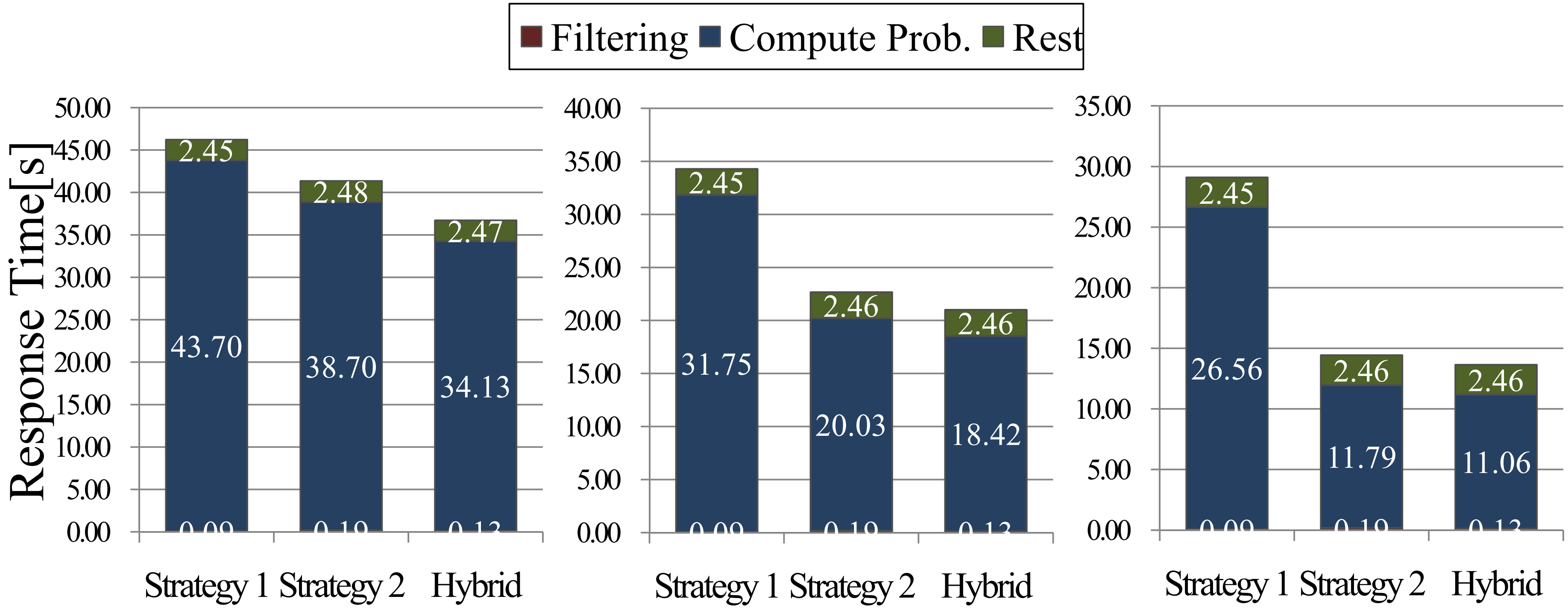
179	150	129
26		

847	276	260
15		

Query is costly when impreciseness is high

Experimental Results - Different θ Values

$\theta = 0.01$ $\theta = 0.03$ $\theta = 0.05$



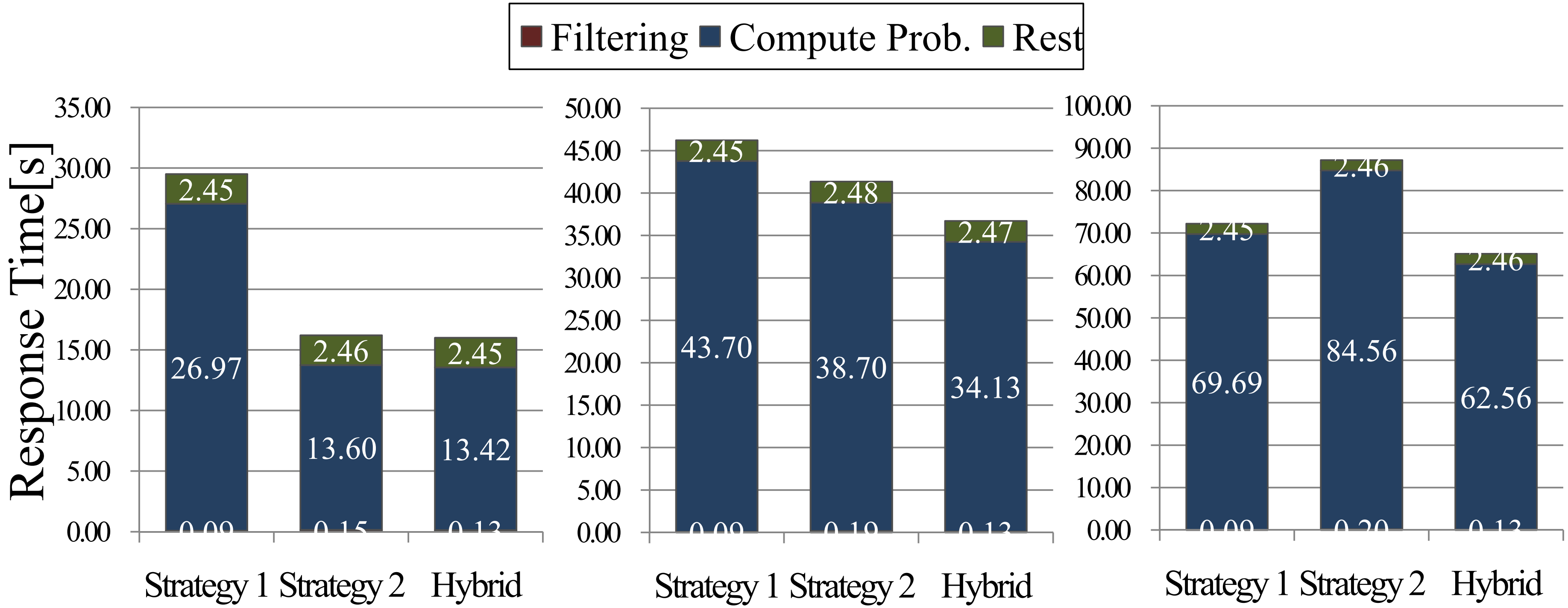
Number of candidates	179	150	129
Number of answers	26		

128	76	68
7		

107	44	40
3		

Experimental Results - Different Shapes

Circle / Default Ellipse / Narrow Ellipse



Number of candidates	115	50	49
Number of answers	26		

179	150	129
26		

276	366	250
24		

Narrow ellipse (correlated distribution) needs to consider additional candidates

Conclusions of Experimental Results

- Most of the processing time is spent in numerical integration
  - ⇒A strategy that can prune more objects has better performance
- Superiority or inferiority of two strategies depends on the given query and the specified parameters
  - No apparent winner
- The hybrid strategy inherits the benefits of two strategies and shows the best performance

Outline

- Background and Problem Formulation
- Related Work
- Query Processing Strategies
- Experimental Results
- Conclusions

Conclusions

- Nearest neighbor query processing methods for imprecise query objects
  - Location of query object is represented by Gaussian distribution
  - Two strategies and their combination
  - Reduction of numerical integration is important
  - Proposal of two pruning strategies and their evaluation
- Future work
  - Evaluation for multi-dimensional cases ($d \geq 3$)

Finding Probabilistic Nearest Neighbors for Query Objects with Imprecise Locations

Yuichi Iijima and Yoshiharu Ishikawa
Nagoya University, Japan