Modeling and Querying Vague Spatial Objects Using Shapelets

Daniel Zinn\textsuperscript{1}    Jim Bosch\textsuperscript{2}    Michael Gertz\textsuperscript{1}

\textsuperscript{1}Department of Computer Science
\textsuperscript{2}Department of Physics

University of California at Davis
Vague Spatial Objects

... are localized objects with uncertainties

Examples
- Astronomical objects
- Meteorological phenomena
- Demographic regions
- Eco-regions
- Probability for “X”

Objective: Data and query model for vague spatial objects
Outline

1. Related Work and Contributions
2. High-Level and Low-Level Operations
3. Shapelets
4. Prototype and Evaluation
5. Summary and Ongoing Work
Outline

1. Related Work and Contributions
2. High-Level and Low-Level Operations
3. Shapelets
4. Prototype and Evaluation
5. Summary and Ongoing Work
Related Work

Fuzzy Regions [Schneider et al. ’97–’05]
- Contour representation
- Operations scale with number of vertices
- High number of vertices for smooth objects
- Discrete Values

Pixel Representations
- Field data
- Operations scale with number of pixels
- Many pixels necessary for smooth objects
- Discrete and fixed squares in x/y area

Both representations are built on a discrete “basis”
Contributions

Data and Query Model
- Represent vague spatial objects with shapelets
- Characterized a comprehensive set of low-level operations
- Build high-level operations from low-level operations

Realization and Evaluation
- Implementation in PostgreSQL
- Shapelet as column data type
- 29 low-level operations implemented
- Stored procedures for high-level operations
- Sample queries and performance experiments
- [Indexing technique based on ε-bounding boxes]
Outline

1. Related Work and Contributions
2. High-Level and Low-Level Operations
3. Shapelets
4. Prototype and Evaluation
5. Summary and Ongoing Work
High-level Operations on Vague Spatial Objects

Standard topological Operations
- Window and point operations
- Union, intersection, overlap

Metric Operations
- Area
- Width, height
- Centroid

Geometric Transforms
- Scale
- Translate
- Rotate
High-level Operations on Vague Spatial Objects

Standard topological Operations
- Window and point operations
- Union, intersection, overlap

Metric Operations
- Area
- Width, height
- Centroid

Geometric Transforms
- Scale
- Translate
- Rotate

Low-Level Operations
- Point-wise evaluation, and arithmetic ops.
- min/max ops.
- Integrals and integral moments
- Value-based and integral-based contours
- Scale, translate, rotate
Examples: Overlap and Width

Overlap
Does/How much does rectangle $R$ overlap with vague object $F$?

- $overlap(F, R) := \int_R f(x, y)$, or
- $overlap(F, R) := \max_R \{ f(x, y) \}$

Width
How long is $F$ along the $x$-dimension?

- Value-based contour to measure width of crisp contour
  $width(F) := width(contour(F, t))$, or
- Root-mean-square width, i.e. standard derivation
  $width(F) := \left[ \frac{\int x^2 f(x, y) dx dy}{\int f(x, y) dx dy} \right]^{1/2}$
Examples: Overlap and Width

Overlap
Does/How much does rectangle $R$ overlap with vague object $F$?

- $\text{overlap}(F, R) := \int_{R} f(x, y)$, or
- $\text{overlap}(F, R) := \max_{R}\{f(x, y)\}$

Width
How long is $F$ along the $x$-dimension?

- Value-based contour to measure width of crisp contour
  $\text{width}(F) := \text{width}(\text{contour}(F, t))$, or
- Root-mean-square width, i.e. standard derivation
  $\text{width}(F) := \left[ \frac{\int x^2 f(x, y) dx dy}{\int f(x, y) dx dy} \right]^{1/2}$
Examples: Overlap and Width

Overlap
Does/How much does rectangle $R$ overlap with vague object $F$?
- $\text{overlap}(F, R) := \int_R f(x, y)$, or
- $\text{overlap}(F, R) := \max_R \{f(x, y)\}$

Width
How long is $F$ along the $x$-dimension?
- Value-based contour to measure width of crisp contour
  $\text{width}(F) := \text{width}(\text{contour}(F, t))$, or
- Root-mean-square width, i.e. standard derivation
  $\text{width}(F) := \left[ \frac{\int x^2 f(x,y) dx dy}{\int f(x,y) dx dy} \right]^{1/2}$
Outline

1. Related Work and Contributions
2. High-Level and Low-Level Operations
3. Shapelets
4. Prototype and Evaluation
5. Summary and Ongoing Work
Shapelets – *Intro*

**Shapelets**
- Image compression technique, which has been developed in astronomy
- Distorted 2-dimensional Gaussian functions

**Our Contributions**
- Use shapelets for representing general vague objects
- Refined math for low-level operations (eg. for overlap)
- Developed math for $\varepsilon$-bounding boxes
Shapelets – *Series Expansion*

Series expansion

\[
f(x, y) = \sum_{n=0}^{\infty} a_n \phi_n(x, y)
\]

- Basis functions weighted by coefficients
- Representation of arbitrary vague objects

\[
f(x, y) = a_{00} a_{01} a_{02} a_{03} a_{10} a_{11} a_{12} a_{13} a_{20} a_{21} a_{22} a_{23} a_{30} a_{31} a_{32} a_{33} \]

\[
\phi_n(x, y)
\]

Zinn, Bosch, Gertz
Shapelets – *Localized, Smooth Basis Functions*

1D Shapelet Basis Functions

\[
\phi_n(x) = \left[2^n \pi^{1/2} n!\right]^{-1/2} H_n(x) e^{-\frac{x^2}{2}}
\]

- **Hermite Polynomials**, weighted by a Gaussian
- \(H_{n+1}(x) = xH_n(x) - H'_n(x)\)

### Hermite Polynomials

- \(H_0 = 1\)
- \(H_1 = x\)
- \(H_2 = x^2 - 1\)
- \(H_3 = x^3 - 3x\)
Representing Arbitrary Objects

- Arbitrary smooth objects
- Quality improves with number of coefficients
- Excellent for smooth objects, OK for crisp objects

Avg. Squared Error
Shapelets vs. Pixels vs. Polygons

Each representation is limited to the same amount of memory (36 floating point values).

- Outstanding for Gaussian-like objects
- Outperform pixel representation
- Outperform polygon representation – even for the polygons

<table>
<thead>
<tr>
<th>Orig.</th>
<th>Polyg.</th>
<th>Pixel</th>
<th>Shap.</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
<td><img src="image7.png" alt="Image" /></td>
<td><img src="image8.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image9.png" alt="Image" /></td>
<td><img src="image10.png" alt="Image" /></td>
<td><img src="image11.png" alt="Image" /></td>
<td><img src="image12.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image13.png" alt="Image" /></td>
<td><img src="image14.png" alt="Image" /></td>
<td><img src="image15.png" alt="Image" /></td>
<td><img src="image16.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image17.png" alt="Image" /></td>
<td><img src="image18.png" alt="Image" /></td>
<td><img src="image19.png" alt="Image" /></td>
<td><img src="image20.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image21.png" alt="Image" /></td>
<td><img src="image22.png" alt="Image" /></td>
<td><img src="image23.png" alt="Image" /></td>
<td><img src="image24.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image25.png" alt="Image" /></td>
<td><img src="image26.png" alt="Image" /></td>
<td><img src="image27.png" alt="Image" /></td>
<td><img src="image28.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image29.png" alt="Image" /></td>
<td><img src="image30.png" alt="Image" /></td>
<td><img src="image31.png" alt="Image" /></td>
<td><img src="image32.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image33.png" alt="Image" /></td>
<td><img src="image34.png" alt="Image" /></td>
<td><img src="image35.png" alt="Image" /></td>
<td><img src="image36.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image37.png" alt="Image" /></td>
<td><img src="image38.png" alt="Image" /></td>
<td><img src="image39.png" alt="Image" /></td>
<td><img src="image40.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image41.png" alt="Image" /></td>
<td><img src="image42.png" alt="Image" /></td>
<td><img src="image43.png" alt="Image" /></td>
<td><img src="image44.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image45.png" alt="Image" /></td>
<td><img src="image46.png" alt="Image" /></td>
<td><img src="image47.png" alt="Image" /></td>
<td><img src="image48.png" alt="Image" /></td>
</tr>
</tbody>
</table>

**Avg. Squared Error**

- Gaussian
- Mt. Shasta
- Freehand
- Polygons

-shapelet | poly | img | Avg. Squared Error
Revisiting Low-level Operations

Example: Integral-Operator

Recursion Relations for Hermite Polynomials

\[ H_n(x) = 2xH_{n-1}(x) - 2(n-1)H_{n-2}(x) \]
\[ \frac{dH_n(x)}{dx} = 2nH_{n-1}(x) \]

Recursion Relations for Integration over Shapelets

\[ I_n = \int_a^b \phi_n(x) \]
\[ I_n = -\sqrt{\frac{2}{n}}[\phi_{n-1}(x)]_a^b + I_{n-2}\sqrt{1 - 1/n} \]
\[ I_0 = \sqrt{\frac{\pi^2}{2}}[\text{erf}(x/\sqrt{2})]_a^b \]
\[ I_1 = -\sqrt{2}[\phi_0(x)]_a^b \]

Operation scales linearly with number of coefficients!
Outline

1. Related Work and Contributions
2. High-Level and Low-Level Operations
3. Shapelets
4. Prototype and Evaluation
5. Summary and Ongoing Work
### Architecture

- New column datatype: **Shapelet**
- Stored Procedures for high-level operations
- C++ class Shapelet, which implements 29 low-level operations
- GNU Scientific Library (GSL) for matrix operations
- C struct RawShapelet as data container
Example: Neighbor Overlap

C-Binding for Low-level Operations

```sql
CREATE FUNCTION s_overlap(shapelet, shapelet) RETURNS FLOAT8 AS '_OBJWD_/shapelet', 'RawShapelet_overlap'
LANGUAGE C IMMUTABLE STRICT;
```

High-level to Low-level Mapping: Overlap

```sql
CREATE FUNCTION overlap_Symmetric(shapelet, shapelet) RETURNS double precision AS
'SELECT s_overlap($1,$2)/
( s_integrateAll($1) * s_integrateAll($2) )
AS result;'
LANGUAGE SQL;

CREATE FUNCTION overlap_Asymmetric(shapelet, shapelet) RETURNS double precision AS
'SELECT s_overlap($1,$2)/s_integrateAll($1)^2 AS result;'
LANGUAGE SQL;
```
# Implemented Low-level Operations

<table>
<thead>
<tr>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>importString(t):s</td>
</tr>
<tr>
<td>exportString(s):t</td>
</tr>
<tr>
<td>importPNG(t,p,f,i):s</td>
</tr>
<tr>
<td>exportPNG(s,i,i,b,t)</td>
</tr>
<tr>
<td>makeGaussian(p,f,f):s</td>
</tr>
<tr>
<td>getCenter(s):p</td>
</tr>
<tr>
<td>setCenter(s,p)</td>
</tr>
<tr>
<td>getBeta(s):f</td>
</tr>
<tr>
<td>setBeta(s,f)</td>
</tr>
<tr>
<td>evalAtPoint(s,p):f</td>
</tr>
<tr>
<td>integrateBox(s,b):f</td>
</tr>
<tr>
<td>integrateAll(s):f</td>
</tr>
<tr>
<td>getIntBBox(s,f):b</td>
</tr>
<tr>
<td>getMaxBBox(s,f):b</td>
</tr>
<tr>
<td>getEpsBBox(s,f):b</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>multiplyScalar(s,f):s</td>
</tr>
<tr>
<td>multiply(s,s):s</td>
</tr>
<tr>
<td>add(s,s):s</td>
</tr>
<tr>
<td>subtract(s,s):s</td>
</tr>
<tr>
<td>normalize(s):s</td>
</tr>
<tr>
<td>intersection(s,s):s</td>
</tr>
<tr>
<td>union(s,s):s</td>
</tr>
<tr>
<td>overlap(s,s):f</td>
</tr>
<tr>
<td>scale(s,f,f,bl):s</td>
</tr>
<tr>
<td>rescale(s,f,i):s</td>
</tr>
<tr>
<td>recenter(s,p):s</td>
</tr>
<tr>
<td>translate(s,p):s</td>
</tr>
<tr>
<td>rotate(s,f):s</td>
</tr>
</tbody>
</table>

s:Shapelet. t:Text. f:Float. p:Point. b:Box. i:Int. bl:Boolean
Sample Queries

Inserting galaxies from an astronomical catalog

```sql
CREATE TABLE galaxies AS (
  SELECT rotate(scale(makeGaussian(c,1.0,f),a,b,TRUE),theta)
  FROM catalog );
```

Galaxies, having >50% of their brightness in a certain box

```sql
SELECT * FROM galaxies
WHERE integrateBox(g, BOX '((5000,5000),(5500,5500))') > 0.5*integrateAll(g);
```

PNG export

```sql
SELECT shapelet_exportPNG(allgal, 100,100, 5000,5000, 5500,5500, '/tmp/out.png')
FROM (
  SELECT array_accum(g * 250000 * 200 ) AS allgal
  FROM galaxies
  WHERE bbox_0_005 && '(5000,5000,5500,5500)'
) AS foo;
```
Performance Experiments

Experimental Setup

- Varying number of coefficients (1 \ldots 120)
- 1 million shapelets of each resolution
- Measured query runtime (with PostgreSQL query statistics)

Result

- Operations scale linearly with number of coefficients
- Operations \textit{integrateAll} and \textit{integrateBox} comparable to \textit{count}
Outline

1. Related Work and Contributions
2. High-Level and Low-Level Operations
3. Shapelets
4. Prototype and Evaluation
5. Summary and Ongoing Work
Summary

High-level vs. Low-level Operations
- A specific set of low-level operations is sufficient to provide a basis for implementing important high-level operations useful in several application areas.

Series Expansion on Shapelet Basis
- Arbitrary objects can be represented
- Localized, smooth set of basis functions
- “Nice” mathematical properties

PostgreSQL Implementation
- Ready-to-use implementation for PostgreSQL
- Indexing with $\varepsilon$-bounding boxes
Future Work

- Multi-shapelets
- Contouring shapelets
- Providing a full set of high-level operations for a specific application domain
- GIST for indexing shapelets
Questions?

Acknowledgments

This work is in part supported by the National Science Foundation under Awards IIS-0326517 and ATM-0619139
C implementation for RawShapelet_integrateAll

```c
PG_FUNCTION_INFO_V1(RawShapelet_integrateAll);
Datum RawShapelet_integrateAll(PG_FUNCTION_ARGS) {
    RawShapelet *s = (RawShapelet *) PG_GETARG_POINTER(0);
    double result;
    RawShapeletIntegrateAll(s, &result);
    PG_RETURN_FLOAT8(result);
}
```

RawShapelet

```c
typedef struct RawShapelet {
    int size; double beta, x, y;
    double data; // starting element for data array
} RawShapelet;
// Low-level data access methods
inline void setData(RawShapelet *s, int offs, double v) {
    (&(s->data))[offs] = v;
}
```
Architecture

- PostgreSQL Stored Procedures
  - High-Level Operations / Queries
  - PostgreSQL Kernel: Functions
    - PerlShapelet
    - C-Wrappers
      - Perl
      - SWIG
      - C
      - C++
    - Shapelet
      - Memory Management
        - Operations on Shapelets: String I/O, integrals and moments, geometric operations, etc.
        - User-friendly interface
      - Data Storage
        - Variable length C-struct
        - Raw data access
  - Rapid Prototyping
    - Test of C++ library
  - PNG Output
    - Overlap
      - ...
  - Data Storage
  - RawShapelet
  - C
Intersection/Union with Multiplication and Addition

\[ f(x) \quad g(x) \]

Intersection
\[ \min(f,g) - f \cdot g \]

Union
\[ f + g - f \cdot g \]

\[ \max(f,g) \]

Zinn, Bosch, Gertz