Traditional wireless sensor network deployments use a wide variety of low fidelity sensors that capture environmental parameters such as temperature, humidity, light, motion etc. to detect and respond to potential threats. Our work focused on deploying high fidelity multimedia sensors, working without significant in-situ filtering, processing and aggregating sensor data to analyze the stored streams to detect emergent threats; easily missed by simpler real time algorithms and un-anticipated by the sensor network developers. We proposed a distributed approach, storage bricks are freely deployed alongside multimedia sensors to spatially localize the streams and allow for incrementally scalable storage. Our research focuses on self-managing and self-organizing mechanisms for scalable storage of the sensed multimedia. Specifically, we develop overlay mechanisms that allow sensors to locate and store the captured streams in storage bricks, lifetime abstractions to provide self-managing and scalable storage and resource management mechanisms that allow the intermediate sensors to manage the transit traffic.

In this presentation, I will also outline our overlay mechanisms that spatially localize the streams while maintaining connectivity. Our system is designed to be robust to network partitions and can be leveraged to optimize for fragment storage or retrieval. For a 10000 node system, we show that our system can locate objects within four hops for replication ratios as low as 0.01%.
Routing Algorithm classification

**Global or decentralized information?**
- Global:
  - all routers have complete topology, link cost info
  - "link state" algorithms
- Decentralized:
  - router knows physically-connected neighbors, link costs to neighbors
  - iterative process of computation, exchange of info with neighbors
  - "distance vector" algorithms

**Static or dynamic?**
- Static:
  - routes change slowly over time
- Dynamic:
  - routes change more quickly
  - periodic update
  - in response to link cost changes

A Link-State Routing Algorithm

Dijkstra’s algorithm
- net topology, link costs known to all nodes
- accomplished via "link state broadcast"
- all nodes have same info
- computes least cost paths from one node ("source") to all other nodes
- gives routing table for that node
- iterative: after k iterations, know least cost path to k dest.'s

Notation:
- \( c(i,j) \): link cost from node i to j. cost infinite if not direct neighbors
- \( D(v) \): current value of cost of path from source to dest. V
- \( p(v) \): predecessor node along path from source to v, that is next v
- \( N \): set of nodes whose least cost path definitively known

Dijkstra’s Algorithm

1 Initialization:
2 \( N = \{A\} \)
3 for all nodes v
4 if v adjacent to A
5 then \( D(v) = c(A,v) \)
6 else \( D(v) = \infty \)
7
8 Loop
9 find w not in N such that \( D(w) \) is a minimum
10 add w to N
11 update \( D(v) \) for all v adjacent to w and not in N:
12 \( D(v) = \min( D(v), D(w) + c(w,v) ) \)
13 if new cost to v is either old cost to v or known
14 shortest path cost to w plus cost from w to v
15 until all nodes in N
### Dijkstra's algorithm: example

<table>
<thead>
<tr>
<th>Step</th>
<th>start N</th>
<th>D(B)p(B)</th>
<th>D(C)p(C)</th>
<th>D(D)p(D)</th>
<th>D(E)p(E)</th>
<th>D(F)p(F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AD</td>
<td>2.A</td>
<td>4.0</td>
<td>2.0</td>
<td>infinity</td>
<td>infinity</td>
</tr>
<tr>
<td>3</td>
<td>ADEB</td>
<td>3.E</td>
<td>4.E</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ADEBC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ADEBCF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Diagram of network](image)

### Dijkstra's algorithm, discussion

**Algorithm complexity:** n nodes
- Each iteration: need to check all nodes, w, not in N
- n*(n+1)/2 comparisons: O(n^2)
- More efficient implementations possible: O(nlogn)

**Oscillations possible:**
- E.g., link cost = amount of carried traffic

Initially...recompute routing...recompute...recompute

### Link State Routing: Dijkstra's algorithm, discussion

- Each router broadcasts the identities and costs of its attached links to all other routers
- All nodes have an identical and complete view of the network
- Each node runs the LS algorithm and computes the shortest paths