Targets and Shapes Tracking
Advanced Seminar
IEEE MDM 2018, Aalborg, Denmark

Goce Trajcevski
Dept. of ECpE
Iowa State University

Peter Scheuermann
Dept. of EECS
Northwestern University

Students:
Oliviu Ghica, Sausan Yazji, Anan Yaagoub, Besim Avci,
Bing Zhang, Mas-ud Hussein, Andi Zang, Xu Teng, Hooman Hashemi
Christoph Kiesl, Klaus Schmidt
Tracking... Plethora of Application Domains

MDM 2018, Aalborg
Process involving:

Devices capable of sensing particular phenomena

Sensors:
Vibration, Acoustic, Temperature, Humidity, Acceleration, Images, Location, Heart rate, Salinity, CO2...

Constraints on sensors:

GPS: imprecision; does not work indoors
Roadside Sensors: cannot distinguish lanes
Cameras: Cannot be place enough-many (not cheap)
RFID: non-flexible (cheap tags, but readers location)

Devices capable of communication

Constraints:
WSN: energy depletion
VANET: proximity; freshness
Participatory Sensing: privacy; coverage
Indoor: scattering...
Processing Environments and Algorithms:

Distributed (in-network)
Centralized

With a purpose of:

Detecting occurrence of event or predicate (composite event)
Constructing notifications (and other actuations)
Maintaining answers to (continuous) queries

and, with Constraints… (syntax, processing,…)

Devices capable of sensing particular phenomena
Devices capable of communication
Tracking – Meta Perspective

- Data/Analytics
- Devices (Protocols; QoS)
- Real-time Constraints
- Location/Mobility
- Security/Privacy
- Control/Processes

Other contexts
Mostly, impact of location (+ time);
Impact(s) of detection(s);
Global Settings:
- Location Based Services
- Data Fusion
- Social Networks
-...
Outline

- History
  - Ancient
  - Less Ancient
- Moving Objects Tracking
  - Road Networks
  - WSN
- Queries/Predicates
  - Shapes Tracking
  - Applications’ domains and properties
- Tracking Boundaries and Topologies
- Tracking and Social Networks
- Influence and Diffusion Tracking
• Historically, the first “organized”/recognized tracking was in the domain of hunting
  ◦ Still popular today (apps…)
• Similarly – warfare…
• Less commonly acknowledged in the past centuries, but much more so in the more recent past:
  ◦ Tracking of celestial bodies
  ◦ Correlation with earthly tasks/processes
    • Seasons; Planning (harvest)
    • Navigation
• Tracking (…not exactly mobile…) data:
  ◦ Census
  ◦ Stock market

• WW II:
  ◦ Radars – issues:
    • Representation
    • Coverage
  ◦ Sonars (relative positioning)

• Tracking
  ◦ Ships/Trains
  ◦ Goods/People


• **Generation of data:**
  - **GPS; Sensors; Inductive Loops; Probe-cars:**
  - **Smart phones**

“Net”-effects:
O(PB) from GPS data (x 400 for cellular-based)
Tens/Hundreds of Sensors, generating ~20GB daily (not counting apps…)

Source: Scotiabank, BI Intelligence Estimates
Moving Object

Changes its spatial whereabouts over time (during its existence)

Any real-world element that may be perceived as “unique” (a car, a person)

**Trajectory:** continuous mapping from Time to (some Geographical) 2D Space

$I(\subseteq) \mathbb{R} \to \mathbb{R}^2$

or, parameterization over time: $\alpha(t) = (\alpha_x(t), \alpha_y(t))$

$\text{Trajectory} = \{(\alpha_x(t), \alpha_y(t), t) | t \in I\}$

Objects may have (spatial) extent – mobile shapes/volumes…

- Description
- Representation
- Manipulations (Querying)
Inductive loop sensor

Cars equipped with internet access (stream music, be alerted of traffic and weather conditions, driving-assistance services…)

GPS location data

\[
\begin{align*}
    t_i & \leq t \leq t_{i+1} \\
    (x - x_i)^2 + (y - y_i)^2 & \leq [(t - t_i)v_{i\text{max}}]^2 \\
    (x - x_{i+1})^2 + (y - y_{i+1})^2 & \leq [(t_{i+1} - t)v_{i\text{max}}]^2
\end{align*}
\]
- So, two sources…
On Lake Shore Drive, Cars back up at the rightmost lane by Belmont Avenue exit.

Different lanes on the same road-segment may have significantly different speeds.
Solution:
1. Augment map-matching with lane awareness;
2. Generate Unit Cell (UC) for each lane;
3. Cluster “acceptably close” UC

\[
P_{\text{lane}} = \frac{D_p(x, y, t, r) \cap A}{D_p(x, y, t, r)}
\]
1. Augment map-matching with lane awareness;
2. Generate Unit Cell (UC) for each lane;
3. Cluster “acceptably close” UC

- spatial range $\Delta S_{kl} = d^+_{kl} - d^-_{kl}$
- temporal interval $\Delta T_{kl} = t^+_{kl} - t^-_{kl}$
- set of trajectories $D_{kl} = [\text{Tr}_1, \text{Tr}_2, \ldots, \text{Tr}_n]$ the GPS points of which belong to it
Wireless Sensor Networks (WSN)
- $10^2 - 10^3$ nodes capable of
  - Sensing
  - Computation
  - Self-organizing in a network

Contexts:
- Energy-sensitive
  - 2-3 orders of magnitude more “eaten up” by communication
- Deployment
  - Irregular/Imprecise
  - Hostile environments
- Queries’ answers may be needed in dedicated *sink(s)*
- Trajectory determined by tracking
  - Location ← collaborative trilateration
Tracking:
- Does every location-sample need to be transmitted?
- Who is the tracking principal?
**Epoch:**
Discretely-synchronized time interval during which ranging measurements are aggregated to achieve one sample.

Active tracking **NOT** required during Idle/Sleep periods.
• **Prediction Approach**
  - *Current Location*
  - *Predicted velocity*

• **Goal**
  - Pick farthest node along the predicted trajectory to minimize number of handoffs

• **Mechanism:**
  - Define a sector-area where ideally the best candidate principal node should reside

When target $o_i$ approaches the limits of the sensing area $L_k$ of the current tracking principal $P_k = sn_1$, based on velocity information, node $sn_5$ is selected as next principal $P_{k+1}$, which is located within the defined relay area.
Tracking in WSN – other issues

Using Uncertainty: Which nodes to awake during epoch-based tracking?

Minimum Disks Cover

Tracking \Rightarrow\text{ adjusting of routing structures…}
Detect an occurrence of a shape $S$ in the network such that

- The area of $S > A$
- The smallest value read by sensors in $S$ is at least $\gamma$
• Group relevant points via DBScan with density \( \varepsilon = 1, \varepsilon = \) sensing range.

• Apply \( \chi \)–shape algorithm to approximate concave polygonal boundary.

• Calculate the area, discard if not large enough to satisfy the definition of the predicates.

• If partial shape is on the edge of the zone controlled by a local principal, forward it along the hierarchy.
- Does one really need all the “evolutionary” events and predicate-changes to be reported?
- If not, what is the “…I can leave with…”:
  - Probabilistic Syntactic Constructs/Semantics;
  - Implementation;
Consider a rectangle that is FOV of a drone:

What should be the trajectory of the drone that ensures max. number of “objects” in its FOV?

- Continuous detection of “most interesting” regions for mobile entities
Tracking Clusters Mobile Data: $k$-MaxRS in WSN

Sink
- Serial Port Communication
- Radio Transceiver Module
- Slab-file Merging Module

Parameters over Serial Port
- Slab-files per cluster

Cluster-Head Tree
- MaxRS Processing Module
- Event Monitoring Module

Cluster-Head
- Sensing Module
- Awake, or Idle

Data Flow
- WSN
- Base-station

MDM 2018, Aalborg
“If an object has been continuously moving towards RI for more than 5min., begin tracking it with at least 2 cameras when it gets closer than 100 yards.

Subsequently, if that object is identified as human carrying a bag, maximize the focus of the closest camera, and transmit its stream to the monitor in the closest patrol vehicle.”
Voronoi diagram of the query region;
Disseminate the boundary (edge or vertex) throughout the V-cell

Inside an edge-cell:
object can only move towards or away in-between successive location samples

Inside a vertex-cell:
similarly, except:
if the perpendicular to the line defined by consecutive location samples falls inside the segment, the object “flips”

Maintain a temporal-accumulator and update it with every new location sample
• add of moved towards and still does;
• reset to 0 if “flips” from towards to away

Observation:
consumption of primitive events?
• GPS does not work inside buildings

• Indoor LBS applications abound:
  ◦ Navigation (people/robots)
  ◦ Inventory
  ◦ Patients/Staff
  ◦ Ambient Intelligence
  ◦ Luggage
  ◦ …

• Must resort to other “tricks”:
  ◦ RFID based
  ◦ Wi-Fi bases
  ◦ Cameras

• Coupled with other data
  ◦ Floor-plans
Indoor Tracking – RFID

- Relies on
  - Tags (transponders)
    - Active or Passive
  - Readers (transducers)
  - DB

- Typically, partitions space in grids
- Uses RSSI based “Euclidian” distance
Indoor tracking – a RFID vs. Wi-Fi

**RFID:**
- Tracking of goods/people/animals/books/luggage; Travel documents; Car keys
- **Features:**
  - Contactless; Re-Writeable; Data ranges/capacity; No LOS; Multi Tags

**Wi-Fi:**
- Similar tracking-domains and (high-level) architecture/components
- **However:**
  - Bulkier devices; More expensive hardware
  - Often, fingerprinting required; (k)NN algorithms used for distance
  - Suffer more from interference (Access points); Smaller range
  - **Much higher transfer rate/bandwidth**
  - **Better Security (RFID can be easily cloned)**
Tracking in the IoT world…

- Heterogeneity of devices;
- Semantics;
- Coupling(s) between:
  - Data;
  - (Predictive) Analytics;
  - Feedbacks/Actuations

- Changing the operating regime of the very devices that gather the data…

- Typically, ~O(10s) per “device”…
• **Boundary estimation and tracking**—predict as many points on the boundary in order to estimate it.

• **Boundary covering**—**locate** the points on the boundary and physically surround it => can only be done by mobile sensors.

• **Contour definition**: set of points with a given field value.

  - **Problem definition**: estimate the contour with mobile sensors with maximum precision and minimum latency in tracing it.
**Step 1: Converge Phase**

1. **Direct Descent DD**
   Choose direction that minimizes the distance function

   $d_f = (1 - \frac{f}{\tau})^2$ if $f(x,y) \leq \tau$

   $d_f = (1 - \frac{\tau}{f})^2$ else $f(x,y) > \tau$

   - $f(x,y)$ is field value at location $(x,y)$; $\tau$ is the contour field value

   Latency high when sensors are collocated and contour is big. **Need to spread!!**

---

Step 2: Spread Always (SA)

- Choose direction that minimizes spread function $s_f$ that measures how far sensor is from target angle $\theta' = \frac{2\pi}{N}$

$$s_f = \left(\frac{\theta_d}{2\pi}\right)^2$$

$\theta$: Current Angle

$\theta'$: Target Angle $= \frac{2\pi}{4}$

$\theta_d$: $|\theta' - \theta|$ (4)

G: Centroid

Sensors spread around centroid moving towards target angle.
Problem with SA

Latency is high when sensors are deployed far and contour is small—some sensors fail to converge. **Adaptive algorithm** considers extent of sensors spread, size of contour and distance of sensor to contour.
Step 2: Coverage Phase

Use Wall Moving Algorithm to trace

Sensor turning right

Actual Boundary

Sensor turning left

Path traced by sensor

Front

Right
Detecting the Topology of Areal Objects

• **Main Idea:** Three levels of boundary structures
  - Boundary Nodes
  - Boundary Cycles
  - Boundary Orientation

• Decentralized Algorithm for construction can be used to answer topological queries, e.g. containment and adjacency of regions

Boundary Node Detection

Sense whether you are in or out of the region (boundary node has at least one-hop neighbor with a different sensed value)

- If you are outside, ask your immediate neighbors (Left, right, in front, behind) if anyone is inside.
- No routing is necessary

Boundary cycles constructed by adaptation of face routing—faces induced by region components
Computation for Boundary orientation—leader of boundary cycle initiates computation and sum is passed around the boundary cycle

\[
\text{area}(P) = \frac{1}{2} \sum_{i=1}^{n-1} X_i Y_{i+1} - X_{i+1} Y_i
\]
How to generate the containment tree for a complex areal object?
Decentralized IN-TORQUE protocol

- Uses variant of *semiline* containment algorithm
- Algorithm routes a message from each *boundary cycle leader* to the exterior updating the message at each boundary node encountered with information about the number boundary cycles passed.
- *Boundary orientation* is used in order to distinguish an outer boundary (of a region) from the inner boundary (of a hole)
Tracking and Social Networks

- Social Networking
- Content Sharing
- Blogs
- Wiki Forum
- Social Media
- delicious
- social bookmarking
- facebook
- LinkedIn
- LiveJournal
- delicious
- MySpace
- Flickr
- Digg
- YouTube
- Twitter
- Wikipedia
- Epinions
- Blogger

MDM 2018, Aalborg
Community structure in social networks

- Groups of people in the network that have a high density of connections within them and a lower density of connections between them.

Friendship network of children in a US school.
Communities Example (overlapping)

Star Wars

Ninjago
Community structure

- No formally defined concept(s) yet
- Densely connected inside each community
- Less edges/links crossing communities

An adaptive algorithm for Community Detection (AFOCS)


AFOCS: A 2-phase and limited input dependent framework
Phase 1: Basic communities detection

- **Basic communities**
  - Dense parts of the networks
  - Can possibly overlap
  - Bases for adaptive CS update

- **Duties**
  - Locate basic communities
  - Merge them if they are highly overlapped
Phase 1: Basic communities detection

- Locating basic communities: objective function aims to maximize internal density
  - Given \((u, v)\) compute \(\psi\) the number of internal edges in its induced subgraph. If \(\psi\) exceeds a threshold \(\tau\) graph is a local community

\[
\Psi(C) = 0.9 \geq \tau(C) = 0.725
\]

- Merging: distance metric \(\text{OS}(C_i, C_j) \geq \beta\) — overlap function considers both no. of nodes and edges in common

\(C_i\) and \(C_j\) have 3 nodes + 3 edges in common
Phase 2: Adaptive CS update

- Update network communities when changes are introduced

Need to handle
  - Adding a node/edge
  - Removing a node/edge

+ Locally locate new local communities
+ Merge them if they highly overlap with current ones
Phase 2: Adding a new node

Which community should \( u \) join to maximize the gained internal density?
Phase 2: Adding a new edge

If a new edge \((u,v)\) is introduced solely inside a community \(C\), it should not split \(C\) into smaller substructures.

When a new edge \((u,v)\) is introduced between two disjoint communities \(C_u\) and \(C_v\) neither \(u\) nor \(v\) should be moved to \(C_u\) or \(C_v\).
Phase 2: Removing a node

- Case above keep community structure intact
- If internal density is not sufficient identify the left-over structure(s) on $C\backslash\{u\}$
- Merge overlapping substructure(s)
Available methods (cellular networks)

- Choosing people/devices from different disjoint communities and send patches to them

Community Based method

- Choosing the people/devices in the boundary of the overlap to send patches & have them redistribute the patches
- Initially some nodes are active
- Active nodes spread their influence on the other nodes, and so on...
Scenario 1: Water network

- Given a real city water distribution network
- And data on how contaminants spread in the network
- Problem posed by US Environmental Protection Agency

On which nodes should we place sensors to efficiently detect all the possible contaminations?
Scenario 2: Cascades in blogs

Which blogs should one read to detect cascades as effectively as possible?

Diagram showing the flow of information between blogs and posts, indicating the time-ordered hyperlinks that contribute to the information cascade.
General problem

- Given a dynamic process spreading over the network
- We want to select a set of nodes to detect the process effectively

- Many other applications:
  - Epidemics
  - Influence propagation
  - Network security
**Problem setting**

- Given a graph $G(V,E)$
- and a budget $B$ for sensors
- and data on how contaminations spread over the network:
  - for each contamination $i$ we know the time $T(i, u)$ when it contaminated node $u$
- Select a subset of nodes $A$ that maximize the expected reward

$$\max_{A \subseteq V} R(A) \equiv \sum_{i} P(i) R_i(T(i, A))$$

subject to $cost(A) < B$

Reward for detecting contamination $i$
Two parts to the problem

- **Reward**, e.g.:
  - 1) Minimize time to detection
  - 2) Maximize number of detected propagations
  - 3) Minimize number of infected people

- **Cost** *(location dependent)*:
  - Reading big blogs is more time consuming
  - Placing a sensor in a remote location is expensive
Solving the problem exactly is NP-hard

Main observation:
- objective functions are submodular, i.e. diminishing returns

Placement $A=\{S_1, S_2\}$

Adding $S'$ helps a lot

New sensor: $S'$

Placement $A=\{S_1, S_2, S_3, S_4\}$

Adding $S'$ helps very little
Lazy forward optimization

- Exploiting submodularity, significantly reduce # influence spread evaluations
  - $S_{t-1}$ — seed set selected after round $t - 1$
  - $v_t$ — selected as seed in round $t$: $S_t = S_{t-1} \cup \{v_t\}$
  - $u$ is not a seed yet, $u$’s marginal gain $MG(u|S_t) = \sigma(S_t \cup \{u\}) - \sigma(S_t)$
  - by submodularity, $MG(u|S_t) \leq MG(u|S_{t-1})$
  - This implies, if $MG(u|S_{t-1}) \leq MG(v|S_t)$ for some node $v$, then no need to evaluate $MG(u|S_t)$ in round $t + 1$.
  - Can be implemented efficiently using max-heap
    - take the top of the heap, if it has $MG$ for the current round, then it is the new seed;
    - else compute its $MG$, and re-heapify

- Often, top element after round $k - 1$ remains top in round $k$.
- Up to 700 X speedup

Battle of the Water Sensor Networks

- Largest water distribution considered for sensor placement
  - Consisted of 21,000 nodes and 25,000 pipes (edges)
  - Diffusion based on the Independent Cascade Model
  - Objective functions to be optimized:
    - DT (detection time): time passed from beginning of contamination until the detection by one of the selected nodes.
    - PA (population detection): injection rate varies with the water demand at a particular node and a person who ingest contaminated amount of water becomes infected with a certain probability.

- Outcome: CELF – lazy greedy forward selection alg. of [Leskovec2007] outperformed all other heuristic approaches
References

References


References


• C. C. Aggarwal and J. Han, “A survey of RFID data processing,” in Managing and Mining Sensor Data, 2013, pp. 349–382.


References


Questions?