Distributed Tracking with Multiple Sensors for Augmented Reality

1. Workshop "Virtuelle und Erweiterte Realität" der GI-Fachgruppe AR/VR

Martin Wagner
Lehrstuhl für Angewandte Softwaretechnik
Fachgebiet Augmented Reality
Institut für Informatik
Technische Universität München
martin.wagner@in.tum.de
Tracking Everywhere

- **Ubiquitous Computing (UbiComp)** blends computing devices into background
- **Augmented Reality (AR)** enriches user’s experience of real world with virtual information
- Vision: make ubiquitous computing power accessible with an AR-based user interface
- Tracking requirements for AR (real time, high accuracy) and UbiComp (distribution, large scale) differ
- Goal of my work: define system that concurrently fulfills all requirements
Overview

• Motivation
• Formal Model: *Ubiquitous Tracking*
• Distributed Implementation Concept
  – Why peer to peer?
  – Distribution strategy
  – Searching optimal tracking inferences
• Current Implementation
• Conclusion & Future Work
Formal Model: Spatial Relationship Graphs

- Use directed graph to model spatial relationships.
- Nodes are objects, edges are spatial relationships.
- In reality, all relationships exist.
- In practice, only some are known, but just estimates are possible.
- Quality of estimates can be described using attributes, examples: latency, tracker error.
Formal Model: Inferences

- Spatial relationships are inherently **transitive**
- New relationships can be **inferred**
- **Attribute propagation** necessary for correct description of inferred relationships’ quality
- Another inference: symmetric relationships, i.e. inversion of edge direction in SR graph
Formal Model: Evaluation Function

- Idea: find optimal inference by detecting all paths in SR graphs between two nodes representing objects.
- Apply application-provided *evaluation function* to this path.
- By convention, path with minimum evaluation function value represents optimal inference.
- *Edgewise* eval function’s value is the sum of the involved edges’ eval function’s values.
- Enables standard shortest path algorithms to detect optimal inference.
Overview

• Motivation
• Formal Model: Ubiquitous Tracking
• Distributed Implementation Concept
  – Why peer to peer?
  – Distribution strategy
  – Searching optimal tracking inferences
• Current Implementation
• Conclusion & Future Work
Implementation: Challenges

• Map formal model on distributed runtime components
  – Store complete representation of SR graph
  – Compute optimal path according to given evaluation function between any two nodes (i.e. compute a shortest path in case of edgewise eval function)
  – Add/remove edges/nodes of SR graph at runtime, triggering recomputation of shortest paths

• Efficiency of runtime communication to fulfil real-time requirements of AR

• Scalability to allow large scale UbiComp applications
Why Peer to Peer?

• Allow ad-hoc connections of mobile setups
  – Use stationary equipment for mobile user’s applications
  – Use mobile users’ equipment (e.g. cameras, accelerometers) for stationary infrastructure, thus enhancing tracking accuracy for other applications
  – Allow two mobile setups to connect without external help

• Make mobile setup self-contained
  – Mobile users should have some tracking results without stationary infrastructure

• No single point of failure
  – Important for security critical applications (e.g. fire or earthquakes)
Distribution Strategy

- Every network node holds only information about locally available SR subgraph
- Keep connections to all other network nodes with adjacent edges in SR graph
- Find adjacent network nodes with standard service discovery algorithms (e.g. SLP, Jini)
Searching Optimal Inferences

• Prerequisites:
  – Edgewise eval function
  – Nodes in SR graph have unique ID, bootstrapping necessary for \textit{anonymous nodes}
  – Detect network node hosting given SR graph node

• Distributed algorithm
  – Asynchronous variant of Bellman-Ford shortest path algorithm
  – Based on message passing between network nodes
  – Result: optimal path between two nodes

  – Runtime infrastructure then can set up an \textit{inference component} aggregating data along this path

• Complexity analysis:
  – Worst case: exponential in number of network nodes
  – Synchronized variant: polynomial complexity
  – But: in practice, number of nodes clearly limited, due to environmental constraints (e.g. consider only all trackers in single room)
Overview

• Motivation
• Formal Model: *Ubiquitous Tracking*
• Distributed Implementation Concept
  – Why peer to peer?
  – Distribution strategy
  – Searching optimal tracking inferences
• **Current Implementation**
• Conclusion & Future Work
Current Implementation

- Based on DWARF middleware
  [MacWilliams, Decentralized Coordination of Distributed Interdependent Services]
  - DWARF service location used for SR graph/network node localization
  - Tracking and inference components modelled as DWARF services

- Small setup
  - 50 SR graph nodes (i.e. objects)
  - 5 network nodes
  - Setup time in range of seconds

- Handle changes in graph topology by recomputation of shortest paths at fixed intervals
Overview

• Motivation
• Formal Model: *Ubiquitous Tracking*
• Distributed Implementation Concept
  – Why peer to peer?
  – Distribution strategy
  – Searching optimal tracking inferences
• Current Implementation
• Conclusion & Future Work
Conclusions

• Formal model can be used to treat all multi-tracker setups uniformly
• Implementation shows that approach is feasible and can be implemented
• Limitations
  – Depends on highly efficient service location (open research problem)
  – Assumes that graph topology and edge attributes change much less frequently than spatial relationships, otherwise suboptimal inferences occur
  – Works on small to medium scale setups only, due to exponential complexity of shortest path search
Future Work

• Implement larger setup
• Integrate locale concept into graph search, i.e. partition graph according to spatial entities
• Identify anonymous nodes in SR graph, e.g. features identified by a natural feature tracker
Thank you!

• Any questions?