Fundamentals of Ubiquitous Tracking

Introduction:

In a world of *"perfect tracking"* the state of the environment would be known and context could be easily inferred.

Given that perfect tracking does not exist, so-called *Smart Environments* are equipped with diverse sensors for environmental state estimation.

Sensors differ widely, depending on the application domain they were designed for.

Typical **UbiComp sensor** characteristics:	Typical **Augmented Reality** sensor characteristics:
• low update rate	• high update rate
• low accuracy	• high accuracy
• large working volume	• small working volume
• designed for mobile use	• mostly designed for stationary use
• cheap & numerous	• expensive & scarce

Hitherto no work has focussed on combining complementary aspects of both types of sensor in an ad-hoc way.

A **Ubiquitous Tracking (Ubitrack)** approach comes close to offering *"perfect tracking"* and enables a new generation of UbiComp applications to:

- automatically incorporate user's wearable sensors with those of other users as well as those embedded in the environment
- gracefully degrade due to infrastructure failure (important in emergency scenarios)

Application scenarios in which a responsive and adaptive environment can result in greater efficiency, effectiveness, safety and security, will benefit from *UbiTrack*:

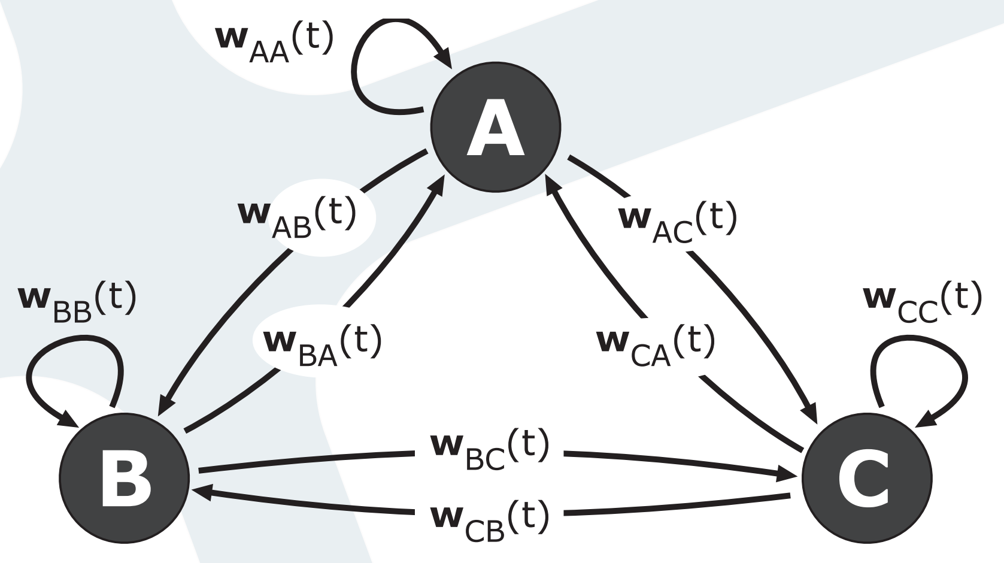
- Hospitals - tracking patients, doctors, drugs and other assets
- Airports - detecting abandoned luggage
- Training scenarios
- Building safety - evacuation management, rescue services

The Formal Framework: From the real-world to inferred knowledge:

Graph-based approach: Objects are nodes, spatial relations are represented by edges.

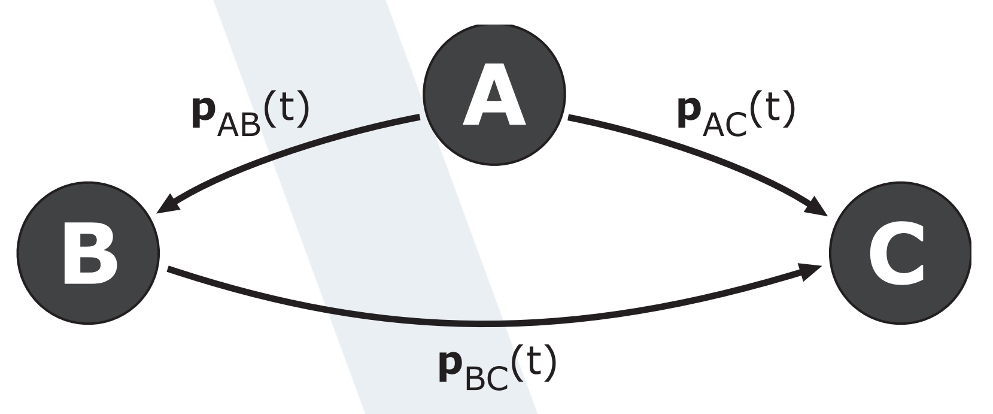
Real World (perfect tracking):

- Precise spatial relations exist between all objects at every point in time
- Graph is complete



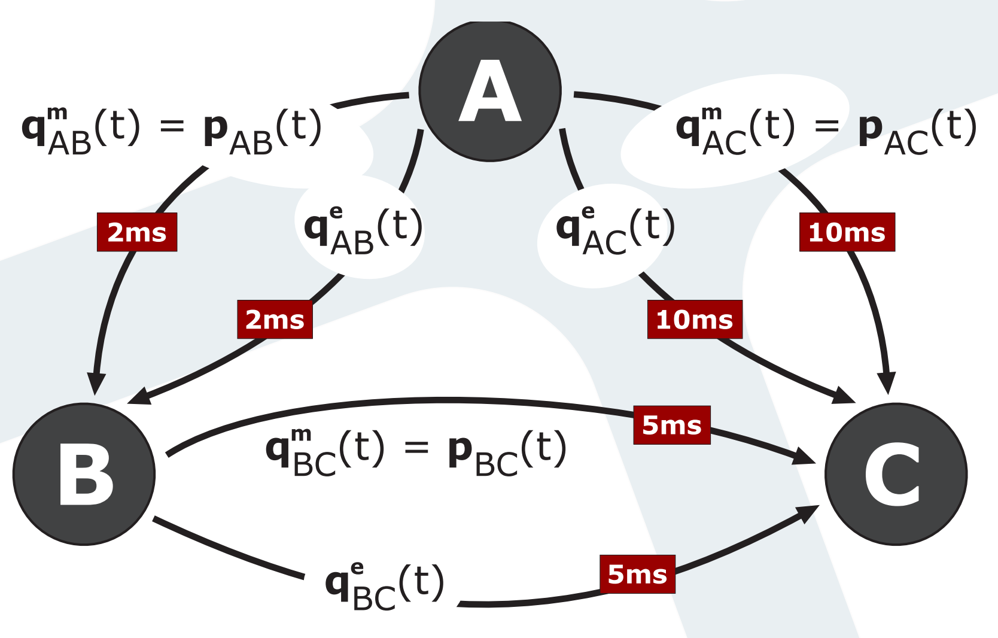
Measured World:

- Only few relations are known
- Measurements are made at sparse, discrete points in time
- Measurements have errors



Inferred Knowledge about the World:

- Using external knowledge, inferences can be made
- Examples: linear motion model used as an estimator over continuous time interval
- Edges along paths (e.g. A→B→C) between two nodes are combined to obtain an inference of a new spatial relation
- Inferred spatial relations also have errors



A
$\mathbf{q}^{m}_{AB}(t) = \mathbf{p}_{AB}(t)$
$\mathbf{q}^{m}_{AC}(t) = \mathbf{p}_{AC}(t)$
2ms
$\mathbf{q}^{e}_{AB}(t)$
$\mathbf{q}^{e}_{AC}(t)$
10ms
2ms
10ms
5ms
B
$\mathbf{q}^{m}_{BC}(t) = \mathbf{p}_{BC}(t)$
C
$\mathbf{q}^{e}_{BC}(t)$
5ms

The Goal:

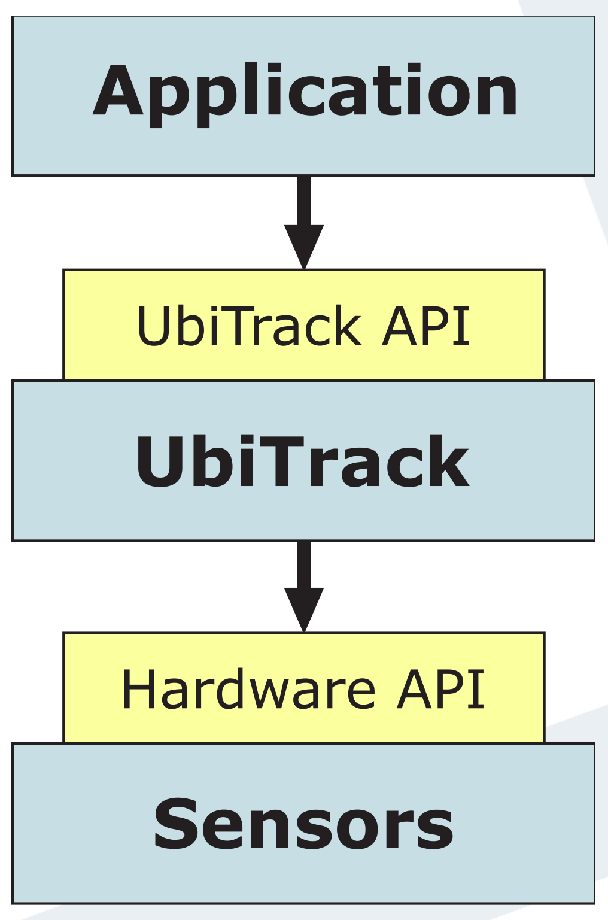
To provide, at any point in time, an optimal estimate of the spatial relationship between two arbitrary objects.

The definition of optimality will depend on the application.

Attributes describe properties of measurements and inferences:

- Choice is application dependent (e.g. a trade-off between monetary cost and precision)
- Examples: update rate, error covariance, confidence, lag
- Multiple attributes can be combined with a user-supplied evaluation function

Implementation Concepts:



- **Layered Architecture**
- **Key assumption:** graph topology and attributes change infrequently compared to measured and inferred spatial relations - two phase approach:
  1. Perform graph search to obtain optimal path
  2. Build data flow graph for efficient propagation and combination of data

Sensor API: common interface to all sensors, based on CORBA IDL, data structure contents:

- Source and Target Object ID
- Position, Orientation, Pose Error
- Time, Time error
- Confidence value

Query API: common interface for applications to access Ubitrack layer, query parameters:

- Source and Target Object ID
- Time offset to query time, update rate
- Evaluation function for choice of path

Open Research Questions:

Centralized versus decentralized architecture:

- TU München: Extension of DWARF framework to allow completely decentralized implementation; drawback: no guarantee for optimal results, communication overhead
- TU Vienna: Extension of Opentracker system to create flexible centralized architecture; drawback: single point of failure, scalability suffers
- Long-term goal: combine both approaches

Choice of attribute set and evaluation function:

- Current set based on experience with existing multi-sensor systems
- Open question: can a general attribute set for all application domains be found?
- Evaluation functions based on edges or paths (efficiency versus generality)?
- Which evaluation functions are general yet efficient?

Optimization issues:

- Graphs become too large for an exhaustive search
- Idea: Create supernodes to reduce graph search complexity
- Problems: How to define supernodes? How do we incorporate mobile users with their own sensor networks?

Martin Wagner, Asa MacWilliams, Martin Bauer, Gudrun Klinker
Institut für Informatik, Technische Universität München
http://www.augmentedreality.de
E-Mail: martin.wagner@in.tum.de

Joseph Newman, Thomas Pintaric, Dieter Schmalstieg
Vienna University of Technology
http://www.studierstube.org
E-Mail: jfn@ims.tuwien.ac.at

This work was partially supported by the Deutsche Forschungsgemeinschaft, the High-Tech-Offensive Bayern, the Austrian Science Foundation under contract no. Y193, Vienna University of Technology, Forschungsinfrastrukturvorhaben TUWP16/2002 and EU contract IST-28610-2000.