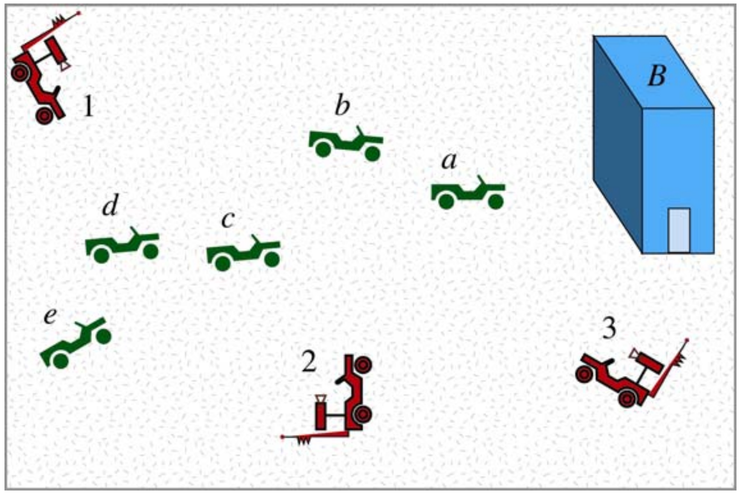
High-Level Tracking and Reasoning Using Distributed Sensors

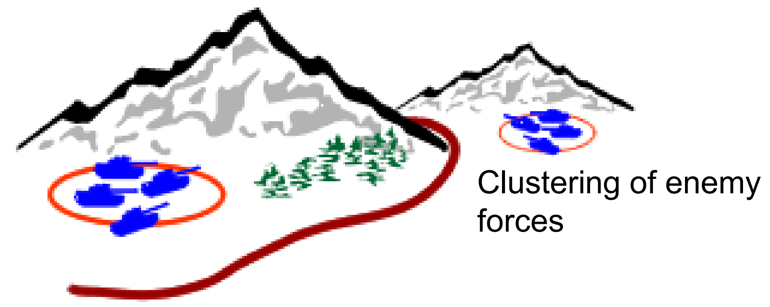
Leonidas Guibas
Stanford University



Sensing for Reasoning and Acting

A system of collaborating sensors should provide data that can:

- Lead to high-level understanding of a situation
- Support efficient decision-making
- Allow for efficient updates as the world-state changes



Optimal route selection

From continuous to discrete

From Pixels to Predicates

To be able to provide such high-level functions, a sensor system must address a number of key technical challenges:

- Data size: sensor data is usually voluminous, yet it must be transmitted and aggregated over a wide area
- Robustness: Noise and uncertainty in the data must be handled
- Sensor control: Sensors must be appropriately tasked to collect and communicate the information relevant to the problem at hand
- Efficient updates: As objects move and the word-state changes, the high-level reasoning performed by the net must be incrementally updated

I. Relational Tracking

- [guibas02] Leonidas Guibas. "Sensing, Tracking and Reasoning with Relations", **IEEE Signal Processing Magazine**, Volume: 19 Issue: 2, Mar 2002.

Focus on Relations between Objects

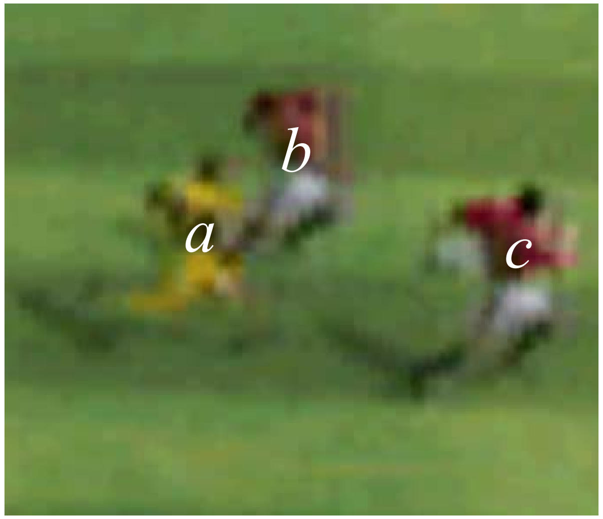
High-level reasoning is often based on relations between objects, or other aggregate information. Focusing on relations has many advantages:

- Data size: relational or aggregated information can be compactly encoded and efficiently transmitted
- Robustness: aggregated information can be statistically more robust than individual data
- Sensor control: Reasoning is naturally relation-based; well-established theories exist on how to go from observations to conclusions

We want to push the discrete/continuous interface as low in the system architecture as possible.

More Relational Advantages

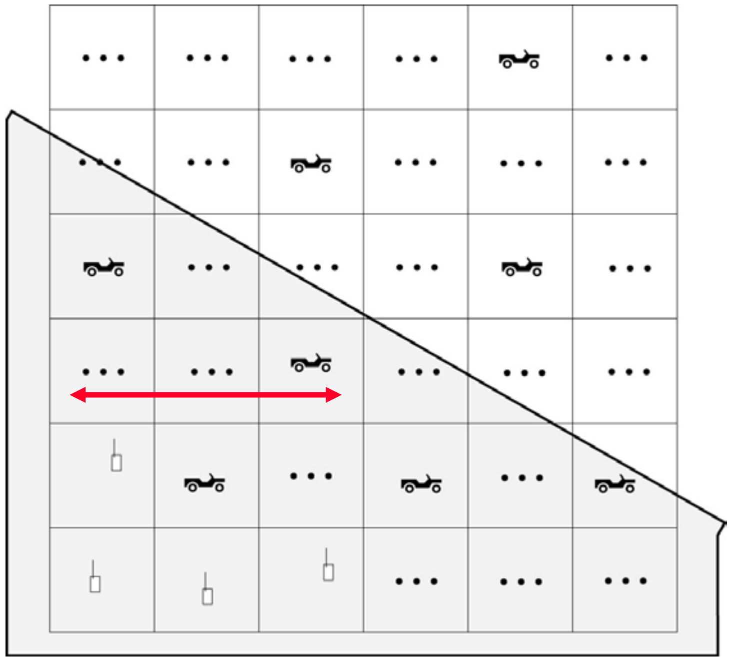
- Often relations can be sensed directly by the sensors, without first estimating object positions or poses
- Aggregated information, if carefully selected, can be stored and re-used
- Relations between objects are more robust and stable than object positions and poses
- When certain relations become unsupportable, alternate relations may do just as well



$\leftarrow$ Efficient updates

An Example: Distributed Range Searching

- Store partial aggregates
- Assemble final answer by combining these pre-stored aggregates



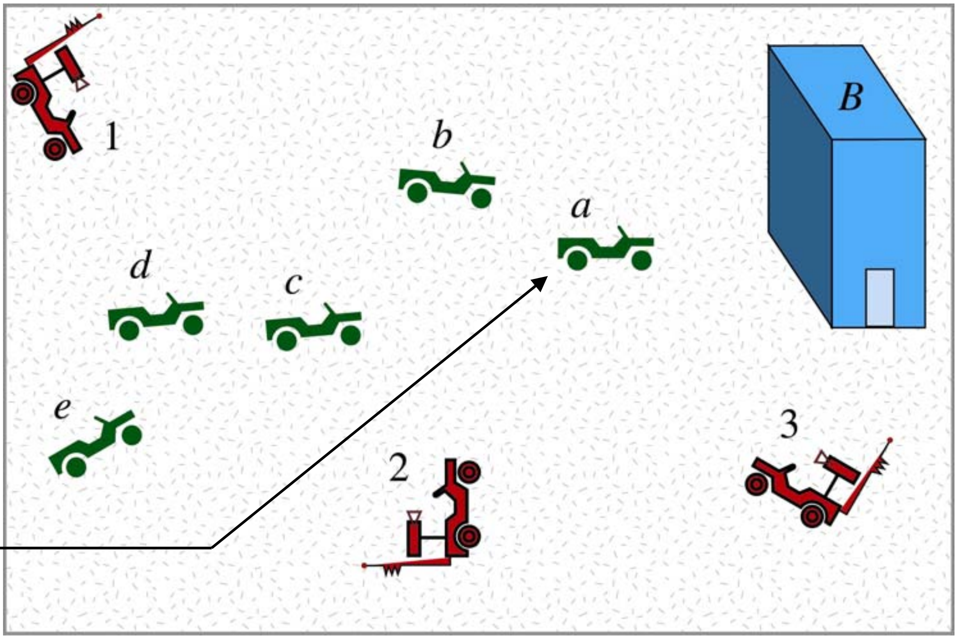
Another Example: First_to_Reach_Building

Cameras observe vehicles moving towards building B:

Camera 2 observes "*a* ahead-of *b*", "*b* ahead-of *c*"

Camera 1 observes "*c* ahead-of *d*", "*d* ahead-of *e*"

System conclusion: *a* is the leader -- the one most likely to reach building B first



a ahead-of *b*
b ahead-of *c*
c ahead-of *d*
d ahead-of *e*

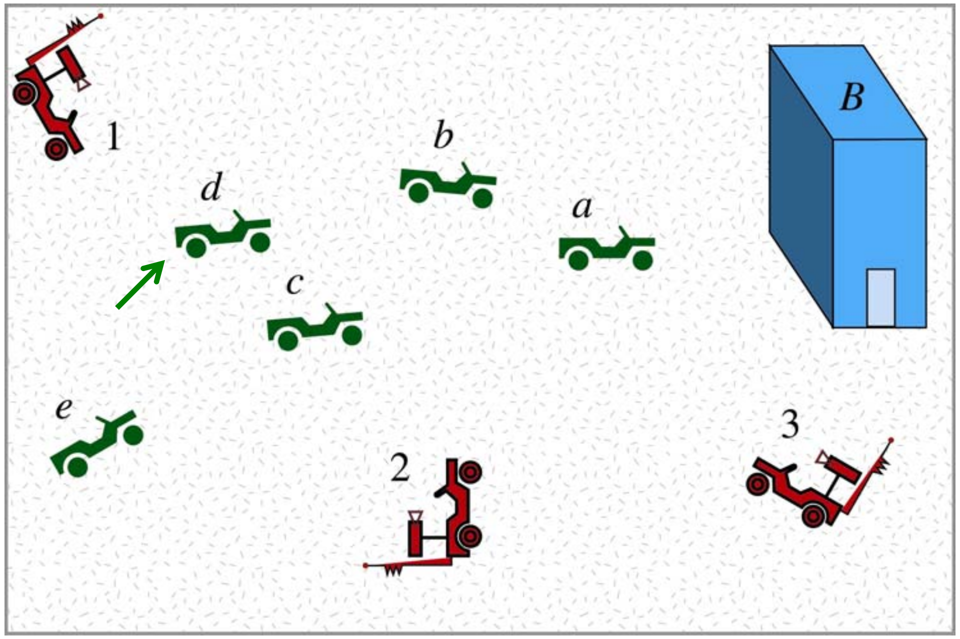
Tracking the Leader under Motion

Suppose *d* moves ahead to overtake *c*

Camera 1 can no longer support the relation "*c* ahead-of *d*"

Camera 1, however, can support the relation "*b* ahead-of *d*"

It follows that *a* is still the leader



a ahead-of *b*
b ahead-of *c*
b ahead-of *d*
d ahead-of *e*

Choosing which Relations to Track

We need to choose sets of elementary relations to track that:

- vary smoothly and stably as the objects move
- always make the computation of the attribute of interest fast

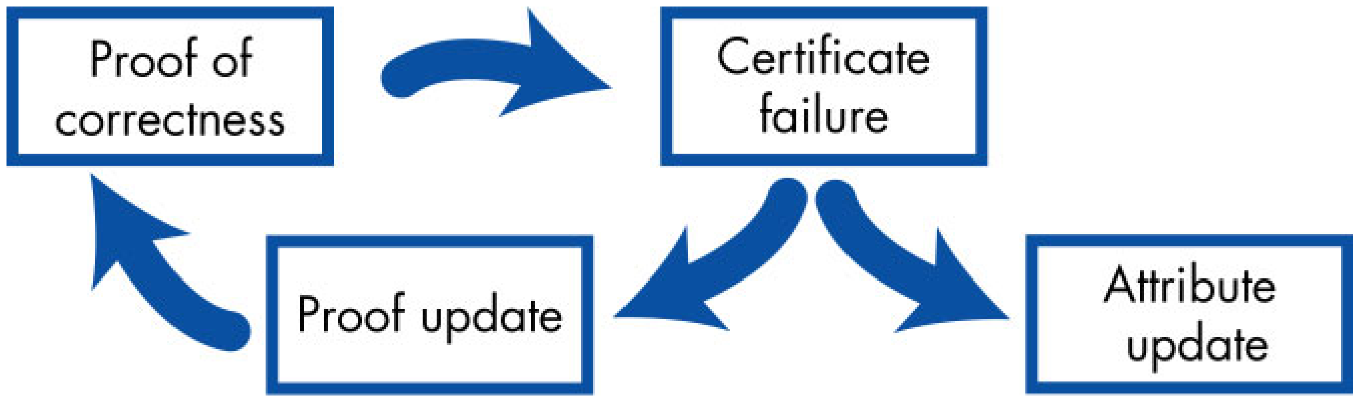
Thus at all times we maintain an assertion cache about the state of the world. This cache is continuously updated as assertions fail, or become unsupportable by the sensors.

Kinetic Data Structures (KDS)

- A KDS for an attribute of interest is an easily repairable set of elementary relations (the certificates) that allows an easy computation of the attribute of interest
- At each certificate failure, the KDS procedure repairs the assertion set
- At all times, the certificates mathematically prove the validity of the attribute computation

A KDS is a proof animated through time.

The Eternal KDS Loop



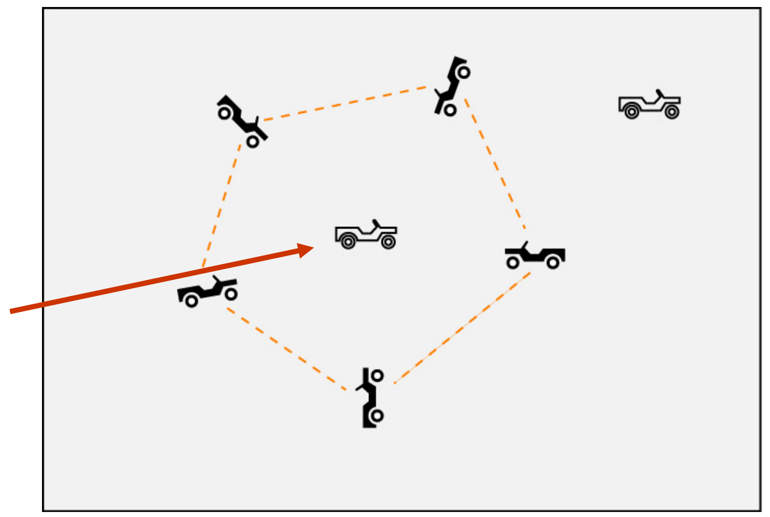
Another Example: “Am_I_Surrounded?”

We have a relatively flat terrain with friendly (white) and enemy (black) vehicles, stationary or moving.

The field contains PIR and acoustic amplitude sensors.

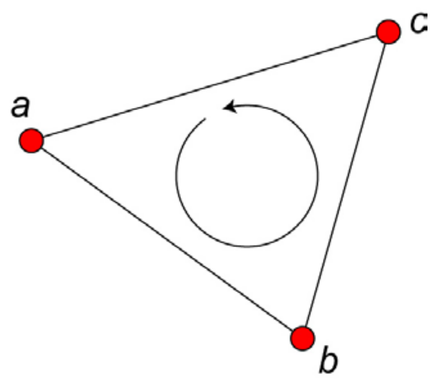
We want to know when a friendly vehicle is surrounded by enemy vehicles.

Say “surrounded”   inside the convex hull



“Am_I_Surrounded” via CCW Relations

The notion of being surrounded can always be captured by a a number of CCW predicates on the vehicle positions.

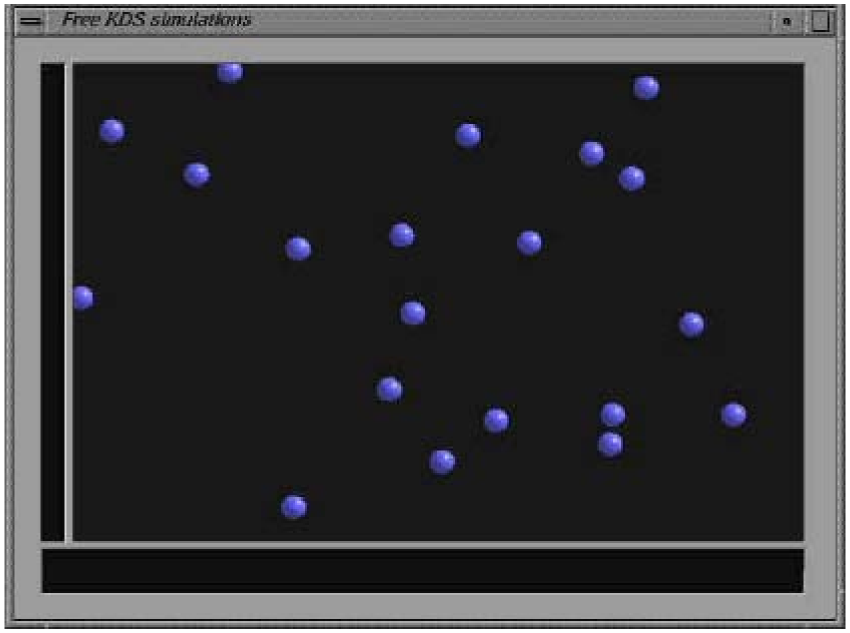


CCW(*a*,*b*,*c*) asserts that the triangle defined by *a*, *b*, and *c* is counterclockwise oriented.

CCW predicates can capture the notion of the convex hull of a set of points, and the notion of inclusion in a convex polygon or triangle.

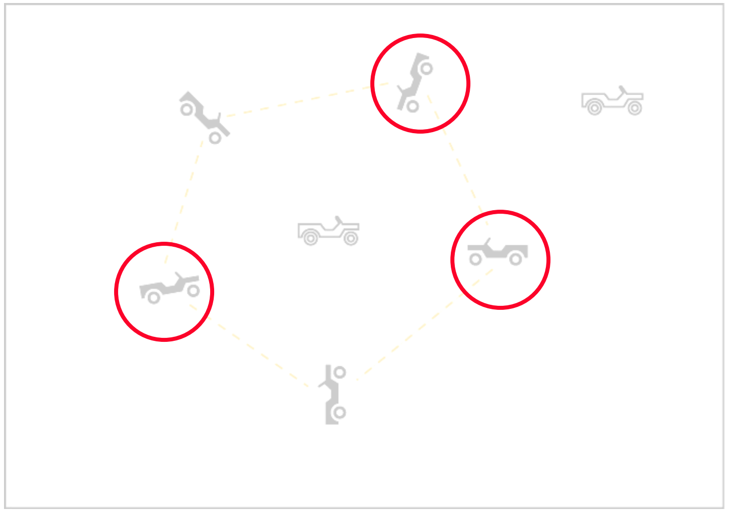
Kinetic Convex Hull

If we had perfect knowledge of the object positions, we could then maintain CH by a standard KDS, on the basis of CCW predicates

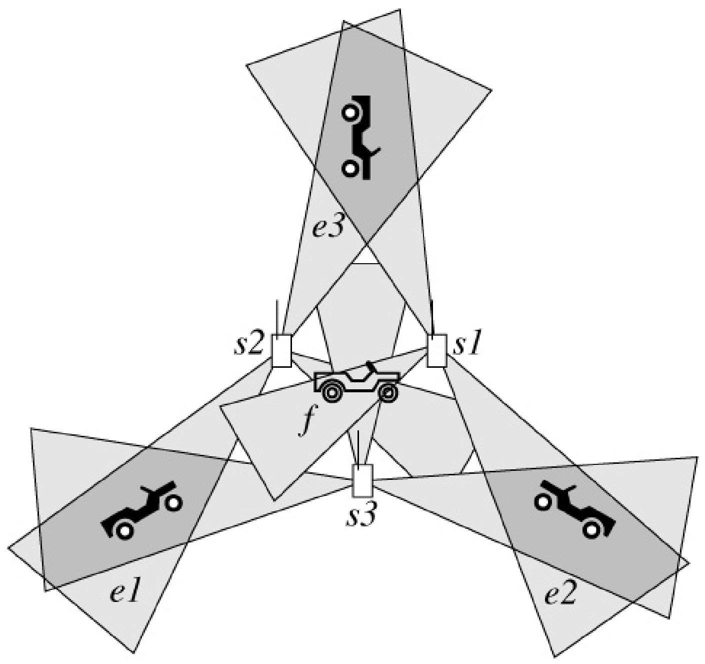


Knowing the Relevant Part of the World State

- The locations of only a small number of vehicles may be sufficient to prove the “surrounded” relation
- Need algorithms by which to search for and find such “certifying” vehicles



Direct Sensing of CCW Relations

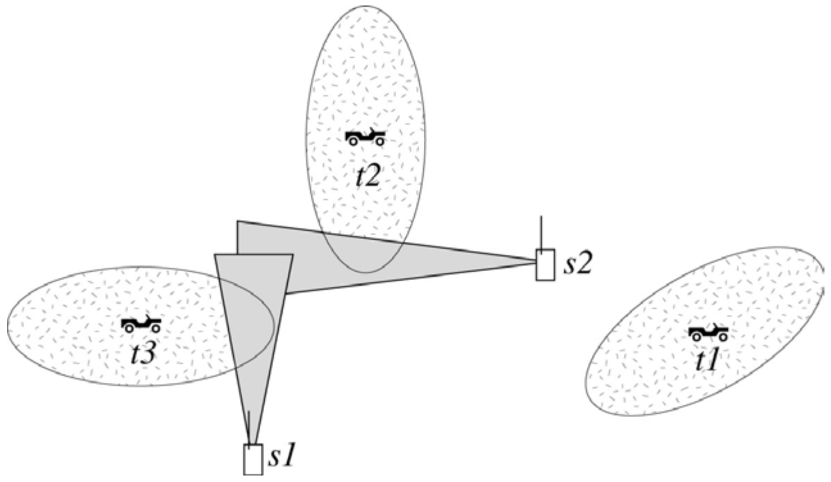


- *s1*, *s2*, and *s3* are PIR sensors
- They directly sense the relations *CCW*(*f*,*e1*,*e2*), *CCW*(*f*,*e2*,*e3*), and *CCW*(*f*,*e3*,*e1*).
- From these relations, the containment of *f* in the triangle defined by *e1*, *e2*, and *e3* follows.

No need to track all vehicles. No need to know exact locations.

Target Localization for CCW Relations

In general, direct sensing is not possible. However, by sufficiently localizing the targets, CCW relations (or their negations) can be estimated with high confidence.



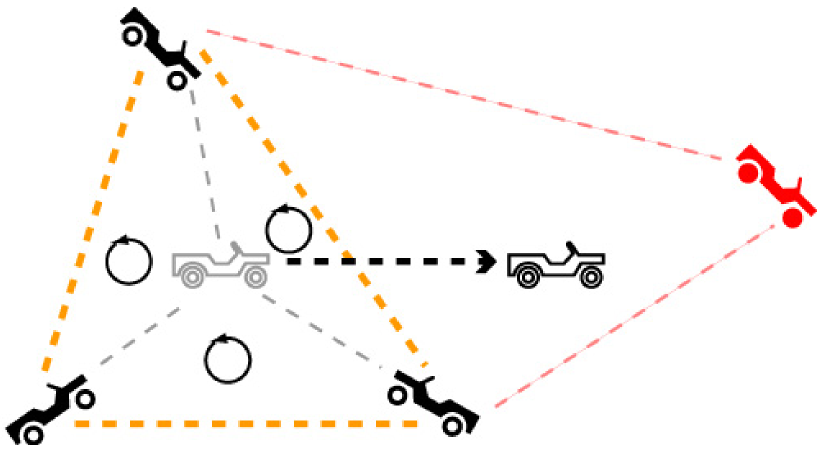
CCW relations can be estimated using the current belief state of the system about the location of each target.

Improved estimation requires a non-trivial sensor selection strategy.

For *CCW*(*t1*,*t2*,*t3*), *s2* is more valuable than *s1*.

Tracking the “Am_I_Surrounded” Relation

As vehicles move, we maintain three CCW relations establishing that the friendly vehicle is indeed in a triangle formed by three enemy vehicles.



When the friendly vehicle escapes the triangle, the sensor net searches for a new enemy vehicle that can still establish containment.

This is akin to a pivoting step in Linear Programming. The triplet basis is thus maintained under motion.

II. Multi-Camera Tracking and Reasoning

- Cameras as distributed sensors generate many novel issues:
  - they are very high data rate devices, so compression and efficient communication are essential
  - visibility is a highly non-local and discontinuous phenomenon, so selecting the most appropriate cameras to use is a challenging task
  - object appearance can vary greatly with view, requiring sophisticated vision techniques to match data from different views

Understanding Environments with People

- Clearly an important practical application
  - activity recognition, people-aware environments
  - security and surveillance
- But generates many challenges
  - Many people imply many occlusions – no single camera can see every individual
  - People are highly deformable shapes, implying little constancy across views

The papers

- [cai96] Q. Cai and J. K. Aggarwal. "Tracking human motion using multiple cameras." **Proc. Intl. Conf. on Pattern Recognition**, pages 68-72, Vienna, Austria, August 1996.
- [mittal01] Anurag Mittal and Larry Davis. "Unified Multi-Camera Detection and Tracking Using Region-Matching." **IEEE Workshop on Multi-Object Tracking**, Vancouver, Canada, July 2001, in conjunction with Int. Cnf. Comp. Vision.
- [yang03] D.B. Yang, H. González-Baños, and L. Guibas. "Counting People in Crowds with a Reat-Time Network of Image Sensors." **ICCV** submitted, 2003**.**

Common Themes

- Centralized processing
- Calibrated cameras
- Background subtraction and segmentation

People tracking and counting

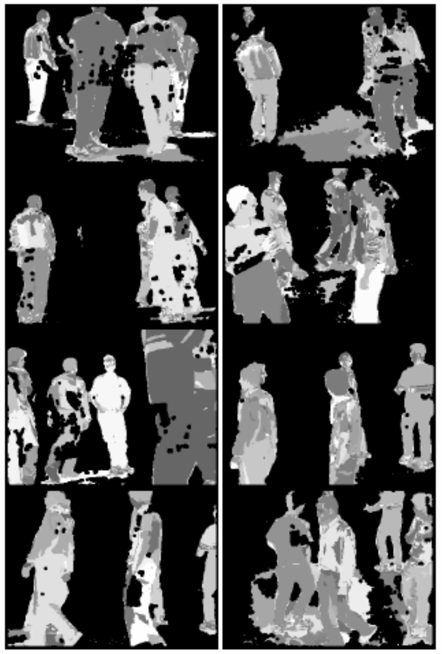


Figure 2: The images from Figure 1, background-subtracted and segmented. The segments are colored randomly and the background is black.