
Junhao HUA
11/16/2012
Distributed Image Processing

Background

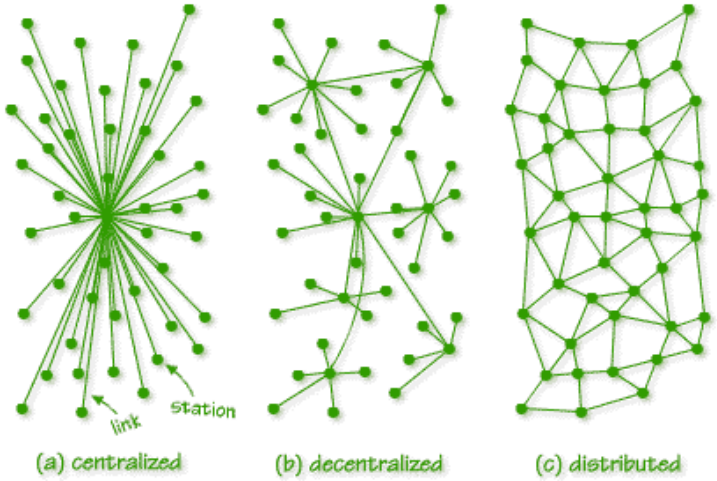
- The emergence of Image Processing System

Challenges

- Bandwidth limitations of networks
- Computational load at nodes
- Communication costs
- System scalability

Solution

- Distributed architecture
  - Advantage: without any global knowledge.
  - Challenge: resource constraints…
  - Objective: to minimize the penalty compared to a centralized solution.



Distributed Camera Networks

[integrated sensing and analysis for wide-area scene understanding]

Background

- Consider a video cameras network
  - Manually analyzed.
  - Fixed Cameras
  - Acquire the desired resolution or viewpoint ineffectively.
  - Difficult in analysis of the video.

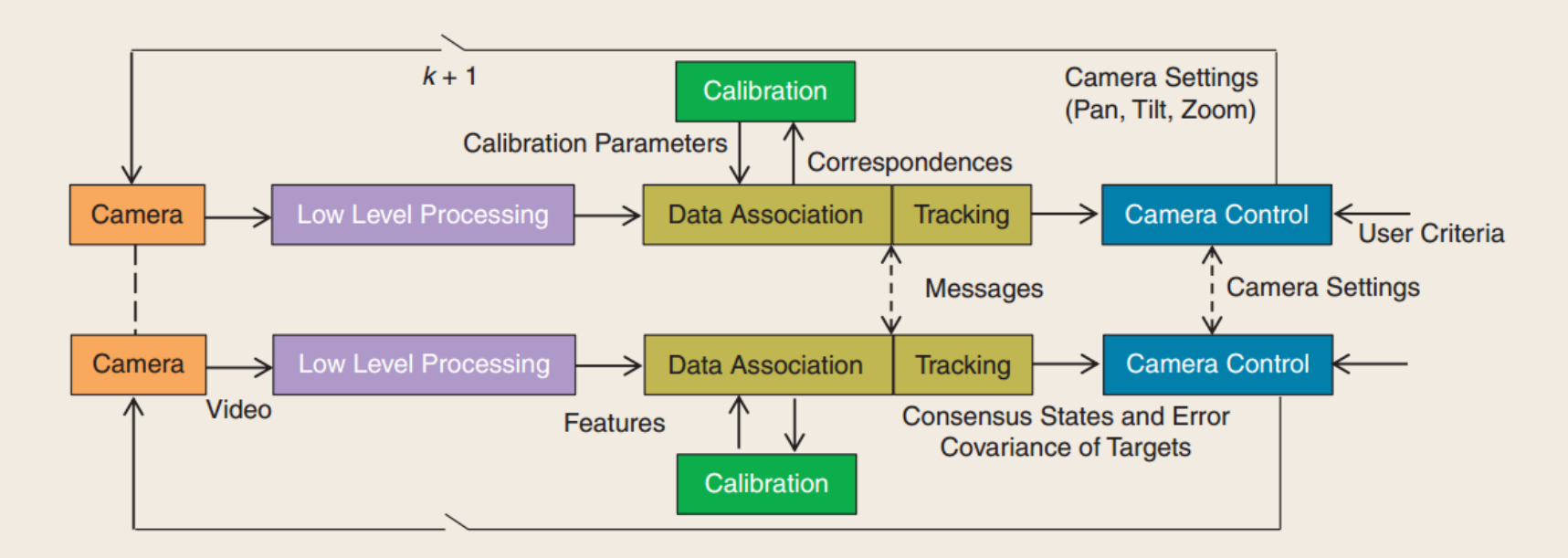
Background

- A possible solution
  - To Integrate the analysis and sensing task more closely by controlling the parameters of a pan-tilt-zoom(PTZ) camera network.
  - [Decentralized fashion] The cameras, acting as autonomous agents, analyze the raw data locally, exchange only distilled information and reach a global analysis.

Solution

- A closed-loop framework
- dynamic scene analysis in a reconfigurable, distributed PTZ camera network
- Integrating a number of component parts that have been studied more or less separately.

The Integrated System Structure



[FIG1] **Overall system diagram depicting a framework for integrated sensing and analysis in a reconfigurable, distributed camera network. The user criteria can define what performance metrics the network will optimize. The user criteria could include covering the entire area at a desired resolution, obtaining facial shots, maximizing image resolution, and so on.**

Data association

- Joint Probabilistic Data-Association Filters (JPDAFs)[1]
  - Update the positions with a probabilistic fusion.
  - Exchange information with neighbors.
  - Associated the closest tracks to each other.
  - Fused together using a Kalman consensus filter.
- A graphical method[2]

[1] N. Sandell and R. Olfati-Saber, “Distributed data association for multi-target tracking in sensor networks,” in Proc. IEEE Conf. Decision and Control, 2008, pp. 1085–1090.
[2] L. Chen, M. Cetin, and A. Willsky, “Distributed data association for multi-target tracking in sensor networks,” in Proc. Int. Conf. Information Fusion, Philadelphia, PA, July 2005, pp. 9–16.

Distributed calibration

- Average consensus-based methods[3] and graphical methods[4] for estimating calibration parameters.
- Data association and calibration are closely interlinked.

[3] E. Elhamifar and R. Vidal, “Distributed calibration of camera sensor net-works,” in Proc. IEEE/ACM Int. Conf. Distributed Smart Cameras, Como, Italy, Aug. 2009, pp. 1–8.
[4] D. Devarajan and R. Radke, “Calibrating distributed camera networks using belief propagation,” EURASIP J. Appl. Signal Process., vol. 2007, no. 1, pp. 1–10, Jan. 2007.

Distributed tracking

- Kalman-consensus tracker
  - Mathematical framework
    - ..
  - Algorithm description
    - ..
- handoff in consensus-tracking algorithm
  - Target moves to overlapping/non-overlapping camera.
  - Sudden failure of camera

Reconfiguration

- Optimal camera placement strategies
- The path-planning strategy
  - Static cameras / PTZ cameras
- Random occluding objects[centralized]
- Distributed approach:
  - Expectation-Maximization(EM)
  - Multiplayer learning in games

Game-theoretic frameworks

- Introduction
  - Multiplayer game: each camera is a player and interested in optimizing its own utility.
  - Local utility functions are aligned with the global utility function.
  - The agreeable setting: Nash equilibrium.

Game-theoretic frameworks

- Model:
  - N_t targets
    - A location vector, a resolution parameter r_l, l=1,.., N_t
  - N_c cameras
    - Camera $C_i \in C$ will select its own set of parameters $a_i \subseteq \mathrm{A}_i$, C_i is the parameter profile that A_i can select from, to optimize its own utility function $U_{C_i}(\mathrm{a}_i)$.
  - $a^* = (a_1^*, ..., a_{N_c}^*)$ is a pure Nash equilibrium if

$$U_{C_i}(a_i^*, a_{-i}^*) = \max_{a_i \in A_i} U_{C_i}(a_i, a_{-i}^*), \ \forall C_i \in C.$$

Game-theoretic frameworks

- Design utility functions
  - Target utility $U_{T_1}(A)$
    - View criterion: $M_{V_1}(A) = 1 - \prod_i (1 - p_{il})$,where

$$p_{il} = \begin{cases} 1 - e^{-\lambda \frac{r_{il}}{r_{\max}}} & if \ \ r_{il} < r_0, \\ 0 & \text{otherwise} \end{cases}$$

    - Tracking criterion: $M_{Tr_l}(A) = \exp\{-Trace(P_i^{l+})\}$, where

$$(P_i^{l+})^{-1} = (P_i^l)^{-1} + \sum_{j \in (C_i \cup C_i^n)} (F_j^l(A))^T R_j^l(A)^{-1} F_j^l(A).$$

      - P is the error-covariance matrices
      - F is the measurement matrix
      - R is the measurement-error covariance

- Global utility
  - the desirability of the settings profile a
  - the global utility function as,

$$U_g(A) = \sum_l V_l \cdot U_{T_l}(A)$$

    - The weight of targets V can be set based on the user's input.
- Camera utility
  - Define as its marginal contribution to the global utility

$$U_{C_i}(A) = U_g(A) - U_g(a_{-i}) = \sum_l V_l (U_{T_l}(A) - U_{T_l}(a_{-i}))$$

    - $a_{-i} = A - a_i$ is the parameter profile of all the cameras except C_i

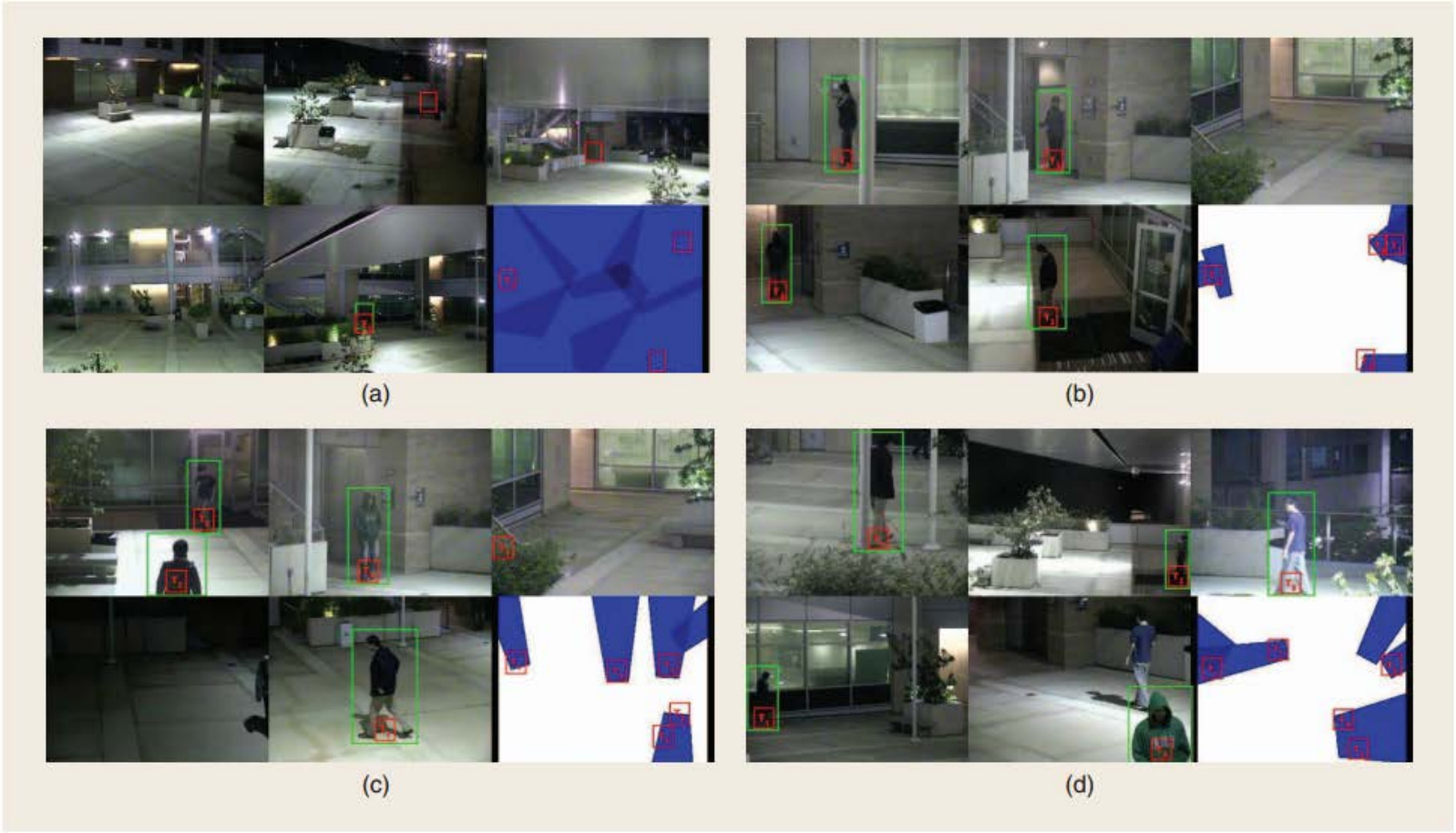
Game-theoretic frameworks

- Optimization strategy(Nash equilibrium)
  - Random choice camera
  - Search for a set of parameters that maximizes its camera utility based on previous negotiation step.
  - Broadcast its choice to its neighboring cameras.
  - Until no cameras increase its own utility.
- Nash equilibrium VS. EM approach

Experiment

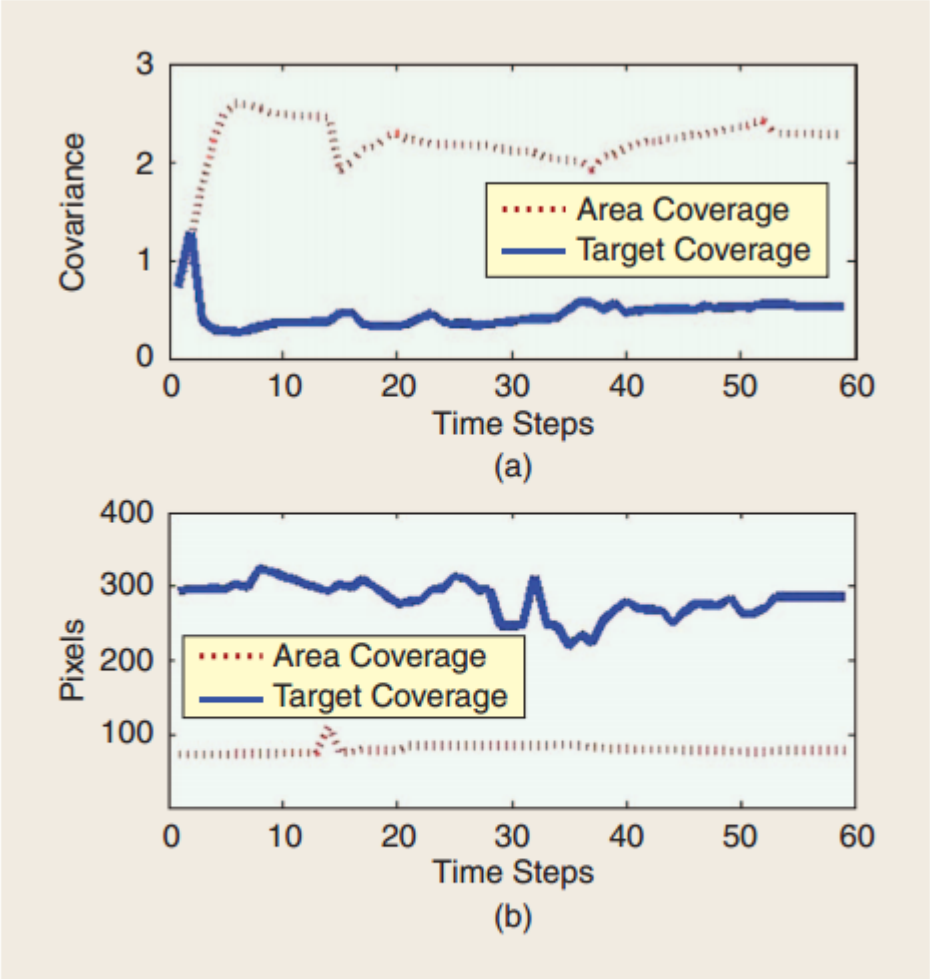
- Setup
  - Cameras connected through a wireless network.
  - Two tasks:
    - To cover the entire area(area coverage)
    - To cover the targets and the entry/exit regions(target coverage)

- Performance analysis



[FIG2] Dynamic camera-control images. Blue regions mark the FOVs. The targets are marked in green with a red label. This figure is best viewed on a computer monitor. (The video as well as the experimental results and example code are available at http://www.ee.ucr.edu/ amitrc/CameraNetworks.php.) Time steps are as follows: (a) $k = 0$, (b) $k = 2$, (c) $k = 19$, and (d) $k = 36$.

- Performance analysis



[FIG3] **Comparison of the average tracker covariance and resolution of all targets being actively tracked by a system for target coverage versus the one for area coverage.**

Discussion

- Scalability
  - The data that need to be exchanged
    - information vector u(4x1)
    - information matrix U(4x4)
    - state estimate x(4x1)
    - PTZ parameters(finite) of cameras.
- Latency
  - Does not have any effect on experimental results.
- Accuracy

Comparative analysis

[TABLE 1] COMPARISON BETWEEN CAMERA NETWORK RECONFIGURATION STRATEGIES.

APPROACH	OBJECTIVE	ARCHITECTURE	OUTCOMES
MITTAL AND DAVIS [33]	STATIC CAMERA PLACEMENT	CENTRALIZED	GLOBAL MAXIMA OF AREA COVERED WHILE CONSIDERING OCCLUSION
SOTO ET AL. [35]	AREA COVERAGE	DISTRIBUTED	LOCAL MAXIMA OF TOTAL AREA COVERED
PICIARELLI ET AL. [34]	WEIGHTED AREA COVERAGE BASED ON PRIOR ACTIVITY MAP	DISTRIBUTED	LOCAL MAXIMA OF WEIGHTED AREA COVERED
QURESHI AND TERZOPOULOS [32]	CAMERA-TO-TARGET ASSIGNMENT	CENTRALIZED	TRACK-BASED ONE-TO-ONE MAPPING (BETWEEN CAMERAS AND TARGETS) AND HANDOFF
INTEGRATED APPROACH	SATISFIES MULTIPLE CRITERIA	DISTRIBUTED	TRACK-BASED MANY-TO-MANY MAPPING (BETWEEN CAMERAS AND TARGETS)

Future work

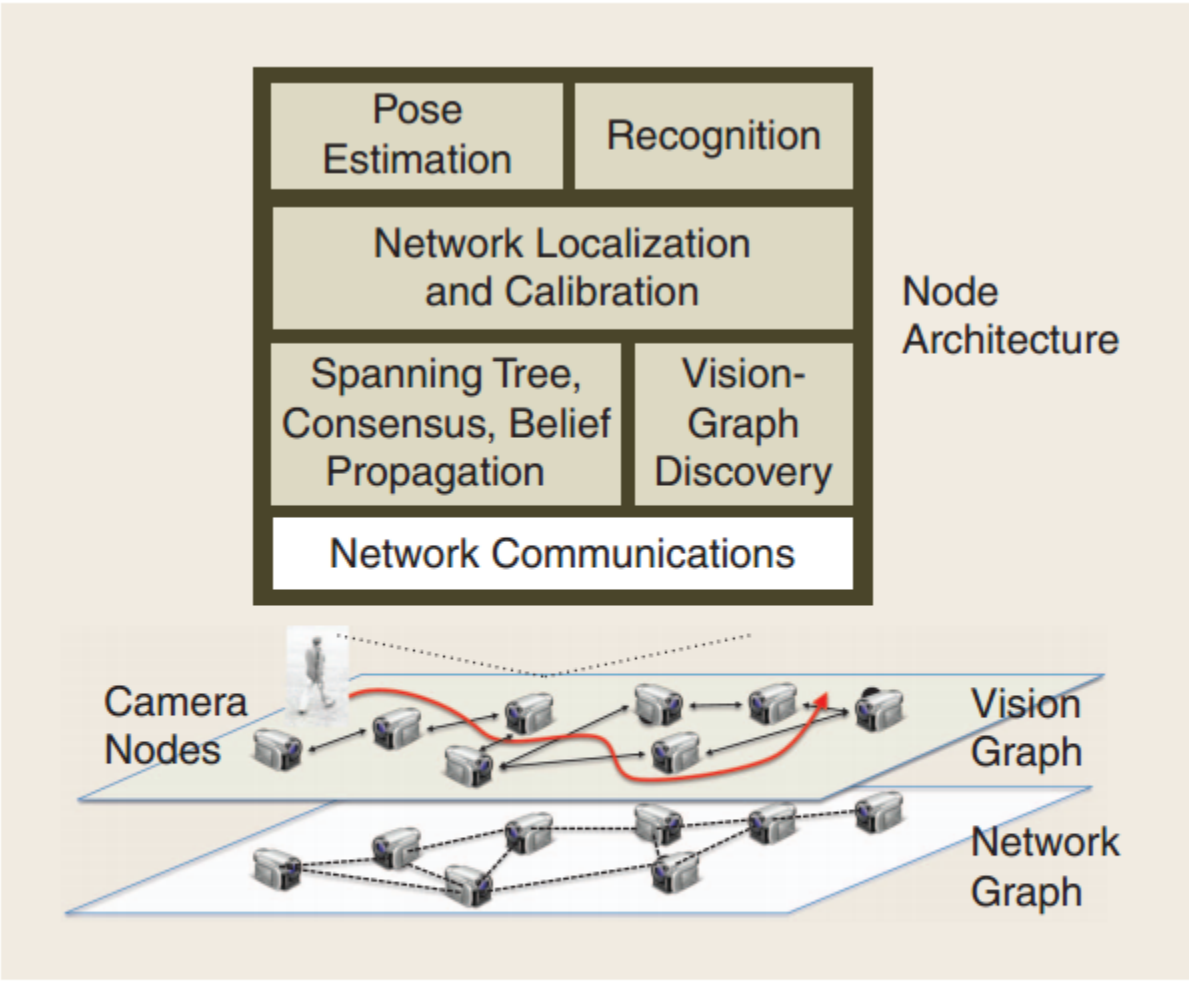
- Robustness of the networks;
- distributed data storage and retrieval;
- Learning semantic models;
- Performance analysis of complex distributed system;
- Visualization tools, etc.

Distributed Computer vision algorithms

[challenges faced in deployment of camera sensor networks]

Background

- Camera Sensor Networks(CSN)
  - Resource-constraints
  - To continuously observe the scene and carry out automatic analysis.
  - A centralized fashion can not address this issue.
  - Distributed computer vision algorithms promises to significantly advance the state of art in computer vision system.
  - But a number of fundamental challenges exist.



[FIG1] An example of distributed scene analysis with a CSN. Camera nodes discover the network's vision graph and use distributed algorithms to localize the CSN. After this initial phase, the nodes can collaborate to track and recognize a target. (Illustration courtesy of Prof. Andreas Terzis.)

Challenges to traditional computer vision algorithms

- Centralized computer vision algorithms apply to CSN deployments.
  - Communications and observations limited as CSNs are constrained by severe network capacity and energy constraints.
  - Collecting all the raw data at a single location is impractical for a CSN.
  - Many computer vision tasks cannot be performed in real time in a low-power computing platform.
  - Current work on CSNs often assumes that cameras are fully calibrated.

Challenges
to traditional distributed algorithms

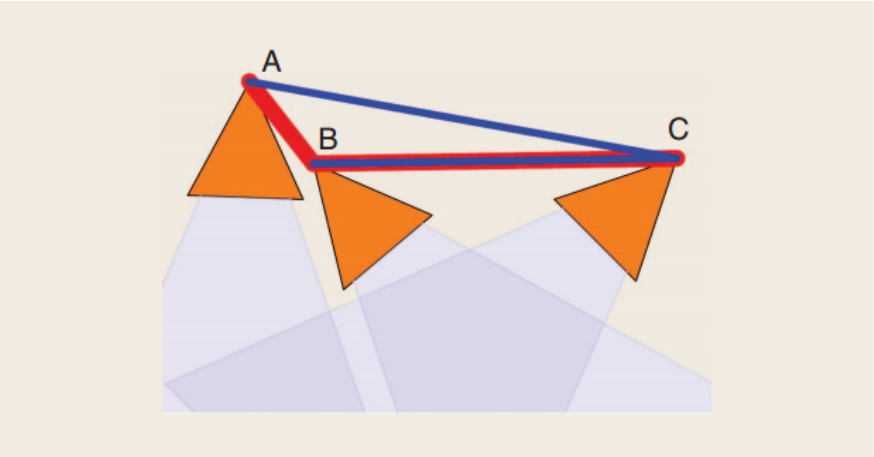
- Distributed algorithms used to be developed in a WSN apply to CSNs.
  - Images and videos are high-dimensional measurements.[scalability]
  - Plagued with noise, outliers, missing data and clutter.[robustness]
  - Information can be time varying.[dynamic]
  - The camera projection model, images and videos are nonlinear functions.
  - There are inherent ambiguities in the estimation of the state, which is not Euclidean.[nonlinear]

Challenges
to traditional sensor network architectures

- The classical distributed computer vision Alg. for WSNs can be extended for addressing following problems:
  - *Vision-Graph Discovery*
  - *Distributed CSN Localization*
  - *Distributed CSN Calibration*
  - *Distributed Object-Pose*
  - *Distributed Action Recognition*
  - *Distributed Tracking(not covered)*

CSN Model

- The communication graph: $G_c = (\upsilon, \varepsilon_c)$
- The vision graph: $G_v = (\upsilon, \varepsilon_v)$
- The nodes of the CSN: $\upsilon = \{1, \ldots, N\}$
- The edge: $\varepsilon_c \subseteq \upsilon \times \upsilon$
- The set of neighbors of node *i*: $\mathrm{N}_i = \{ j \in \upsilon \,|\, (i, j) \in \varepsilon \}$



[FIG2] The communication graph (red lines) and the vision graph (blue lines) are, in general, different. For instance, in the example, cameras A and B can communicate but their fields of view do not intersect.

CSN Model

- The pose of each camera node *i* : $g_i = (R_i, T_i) \in SE(3)$
  - Where SE(3) is the space of rigid-body transformations. Each element of SE(3) is composed of a rotation matrix

$$R_i \in SO(3) = \{R \in R^{3\times 3} : R^T R = I, \det(R) = 1\}$$

  - and a translation vector $T_i \in R^3$
- The standard projective camera model
  - a 3-D point $X \in R^{3\times 3}$ to its image in the *i*th camera $x_i = [x_i, y_i, 1] \in R^3$, by the formula

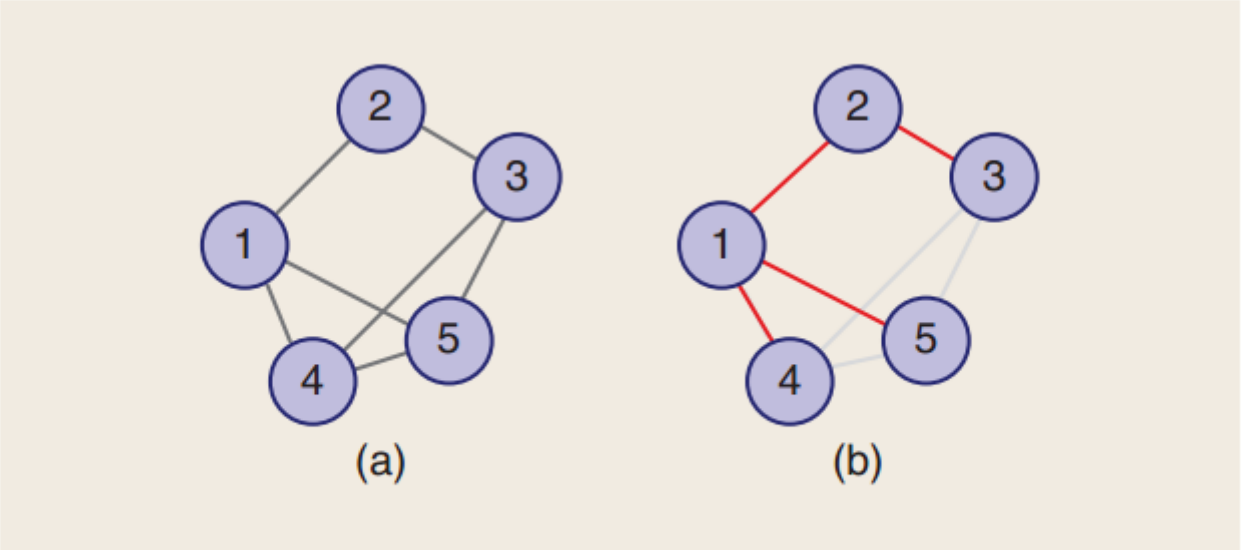
$$\lambda_i x_i = K_i (R_i X + T_i)$$

    - $\lambda_i \in R^+$ is the depth of the point X in camera *i.*
    - $K_i \in R^{3\times 3}$ is an upper triangular matrix called the calibration matrix, which transforms the coordinates of an image point from metric to pixel coordinates.

Review of Distributed Alg.

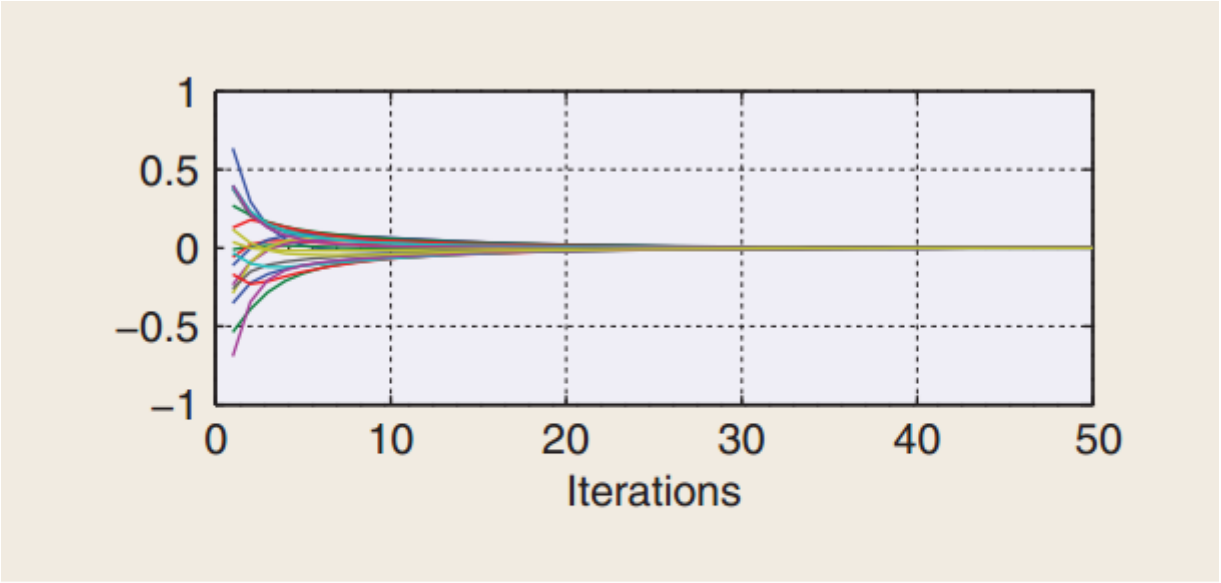
- Spanning-tree algorithms
- Consensus algorithms
- Belief-propagation algorithms

Spanning-tree algorithms



[FIG3] (a) A network with five nodes and (b) a possible spanning tree highlighted in red.

Consensus algorithms



[FIG4] An example of consensus for a network of $N = 20$ nodes in a ring topology and where the mean of the local measurements is $\bar{u} = 0$. We plot the state of each node after each iteration. All the states converge to the average $\bar{u}$.

Belief-propagation algorithms

- The marginal distribution $p(z_i) = \int_{\{z_j\}_{j \neq i}} p(z_1, \ldots, z_N) dz_j.$
- Markov can be factorized as $p(z_1, \ldots, z_N) \propto \prod_{i \in \mathcal{V}} \phi_i(z_i) \prod_{j \in \mathcal{N}_i} \psi_{ij}(z_i, z_j),$
- Using an iterative message-passing algorithm

$$m_{ij}^0(z_j) = 1,$$

$$m_{ij}^t(z_j) = \int_{z_i} \psi_{ij}(z_i, z_j) \phi_i(z_i) \prod_{k \in \mathcal{N}_i, k \neq j} m_{ki}^{t-1}(z_i) dz_i,$$

$$b_i^t(z_i) = \phi_i(z_i) \prod_{j \in \mathcal{N}_i} m_{ji}^t(z_i).$$

  - b is called the belief at node i and an approximation of the marginal density p(zi) [known as sum-product belief propagation]
  - If the state of node z and observation u is statistically independent, then the joint distribution as

$$p(z_1, \ldots, z_N | u_1, \ldots, u_N) \propto p(z_1, \ldots, z_N, u_1, \ldots, u_N)$$
$$= \prod_{i \in \mathcal{V}} p(u_i | z_i) \prod_{j \in \mathcal{N}_i} \psi_{ij}(z_i, z_j),$$

Vision-Graph Discovery

- Feature extraction
- Feature matching and vision-graph discovery

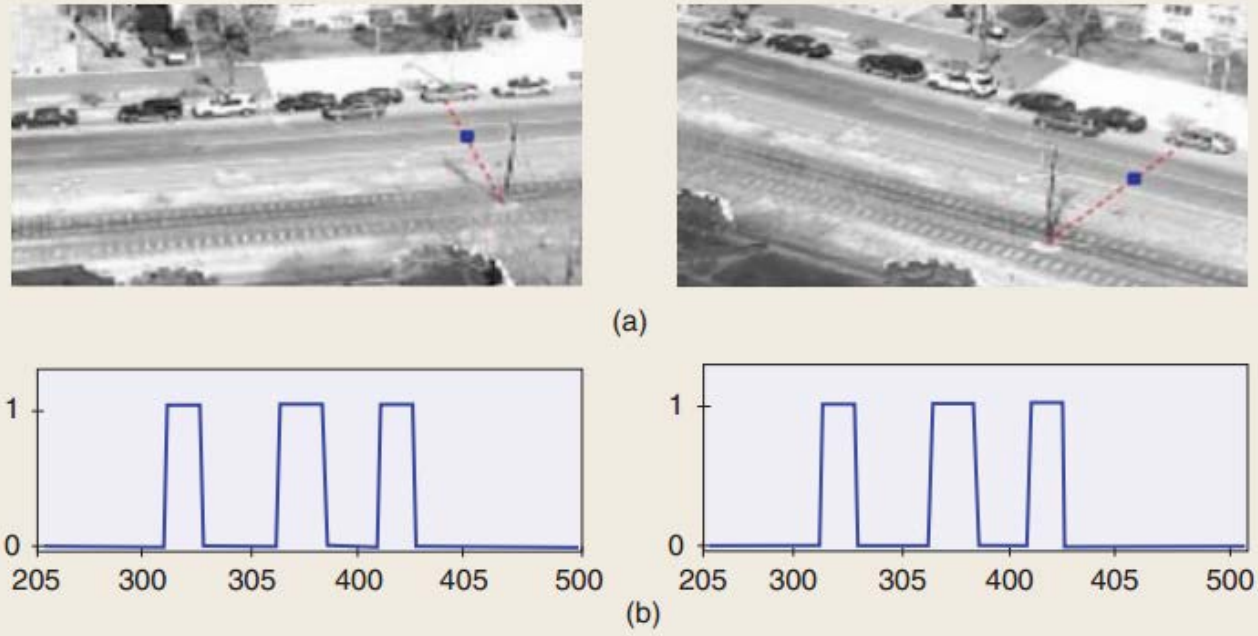
Feature extraction

- Photometric features
  - Traditional choice
    - Scale-invariant feature transform (SIFT)
    - Speeded up robust features (SURF)
    - Histogram of oriented gradient (HOG)
  - Drawback
    - Require the variation of the image intensities to be rich enough.

Feature extraction

- Activity features
  - Standard motion-detection algorithm
    - Background subtraction
    - Hypothesis testing and robust fitting
  - Advantage
    - Providing correspondences for regions with low texture
  - Drawback
    - The scene must be nonstatic

An example of activity features



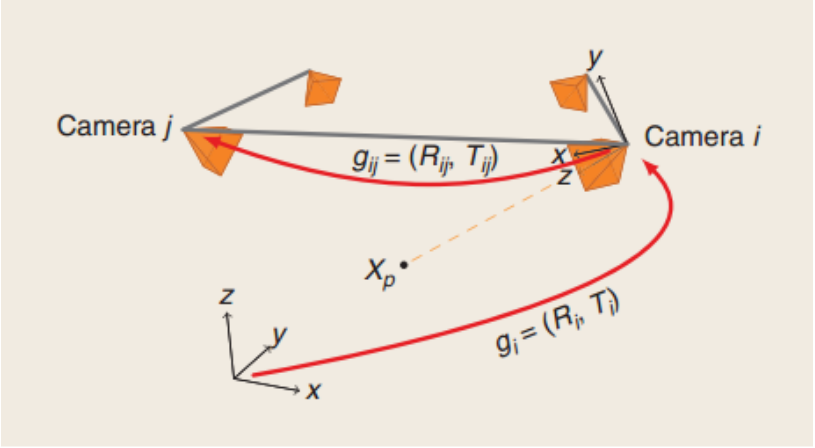
[FIG5] An example of (b) activity features extracted from (a) two different images corresponding to the same point on the road (marked in blue). In spite of the different camera angles, the two feature descriptors are extremely similar. (Photographs and plots courtesy of Prof. V. Saligrama [26].)

Feature matching and vision-graph discovery

- Reduce the burden of match high-dimensional feature
  - Select and transmits the most discriminant SIFT features.
  - Compress feature
    - Compress into a feature digest using PCA etc.
    - Transform the features into binary vectors using CHoG etc.
  - Use the feature digest from other camera to reconstruct its feature descriptors.
- Reduce the number of image pairs
  - The transitivity of the correspondences
  - The theory of random graphs

Distributed structure from motion(SfM)

- Model
  - Consider N cameras with unknown poses {g=(R,T)} observing P 3-D points at P unknown locations.
    - The goal of camera localization is to estimate the camera poses from the images taken by N cameras.
    - The goal of 3-D reconstruction is to find the locations of the 3-D points observed by N cameras.
    - The combination of two is SfM.



[FIG6] A schematic view of a CSN where each node can measure the relative pose of other cameras.

- Solve the SfM in a centralized fashion
  - Assumption
    - All the cameras have an intersecting field of view.
    - The 3-D scene is static
  - Objective function
    - Minimizing the reprojection error

$$RepErr = \sum_{i=1}^{N} \sum_{p=1}^{P} \| x_{ip} - \pi(K_i(R_i X_p + T_i)) \|^2$$

      - where $\pi : R^3 \rightarrow R^2$ represents the perspective projection model
    - Nonlinear optimization problem
- Solve the SfM subtasks(CSN localization and calibration) in distributed fashion

Distributed CSN localization

- Assumption
  - The vision graph is known
  - The feature matches are known and contain no errors.
  - The feature points are corrupted small-to-moderate amounts of noise but not by outliers.
  - the calibration parameters K are known.
- Possible approach
  - Spanning-tree-based approach
  - Consensus-based approach

Spanning-tree-based algorithm

- Model
  - Each camera is equipped with a blinking LED and accelerometer.
  - Some of the cameras are in the direct line of sight.
- algorithm
  - Pairs of cameras that are visible to each other can uniquely determine their relative rotation.
  - Then, using a spanning-tree algorithm to setup a linear system of equations and recover also the relative translation.
- Improve
  - Bundle adjustment
- Evaluate
  - Not require extracting or matching any feature from the image.
  - Require more hardware; more restrictive assumption.

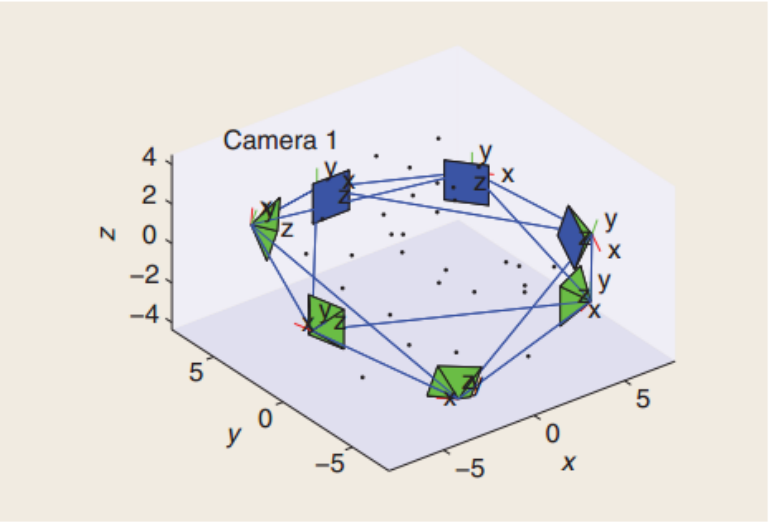
Consensus-based algorithm

- Model
  - Based on two-view geometry to recover noisy estimates $\tilde{g}_{ij} = (\tilde{R}_{ij}, \tilde{T}_{ij})$ of their relative poses.
  - A gradient descent in SE(3) for minimizing the cost function

$$\phi_{loc}(\{g_{ij}\}_{(i,j)\in\varepsilon}) = \frac{1}{2}\sum_{(i,j)\in\varepsilon} d^2_{SE(3)}(g_{ij}, \tilde{g}_{ij})$$

    - The consistency constraints are not distributed.
  - A cycle-distributed solution
    - Sharing the information between each node and all the cycles it belongs to.
  - Reparameterizing
    - Using absolute poses(Ri,Ti), rather than the relative poses(Rij,Tij).
    - Results in a local update, the gradient of the cost function is distributed.

- experiment
  - On a network of N = 7 cameras looking at a scene with 30 3-D points



[FIG7] **The synthetic network and 3-D points used to test the consensus-based localization-CSN algorithm of [40]. The network of $N = 7$ cameras has four-regular graph topology.**

[TABLE 1] LOCALIZATION ERRORS OBTAINED BY THE CSN LOCALIZATION ALGORITHM OF [40] ON THE NETWORK CONFIGURATION OF FIGURE 7.

NOISE	0 PX	1 PX	3 PX
ROTATION	0.00 (0.00)	0.13 (0.00)	0.39 (0.03)
TRANSLATION	0.00 (0.00)	0.09 (0.00)	0.29 (0.03)

THE POINT CORRESPONDENCES ARE CORRUPTED BY GAUSSIAN NOISE WITH VARIANCE 0, 1, AND 3 PX. WE REPORT THE MEAN AND VARIANCE (THE LATTER IN PARENTHESIS) OF THE ROTATION AND TRANSLATION ERRORS (IN DEGREES) OF MORE THAN 100 REALIZATIONS OF THE NOISE.

- This approach is more geometric in nature
- The main limitation of this method is that the optimization problem is nonconvex, hence, a good initialization is critical.

Distributed CSN Calibration

- Challenge
  - Calibrating a single camera do not scale well for CSNs.
  - Several autocalibration methods calibrate the cameras by solving nonlinear equations such as Kruppa's equation.[numerically ill]
  - The camera have different intrinsic parameters.
- Possible Solution
  - Integrating visual information across the network
    - Spanning-tree-based approach
    - Belief-propagation approach

Spanning-tree-based approach

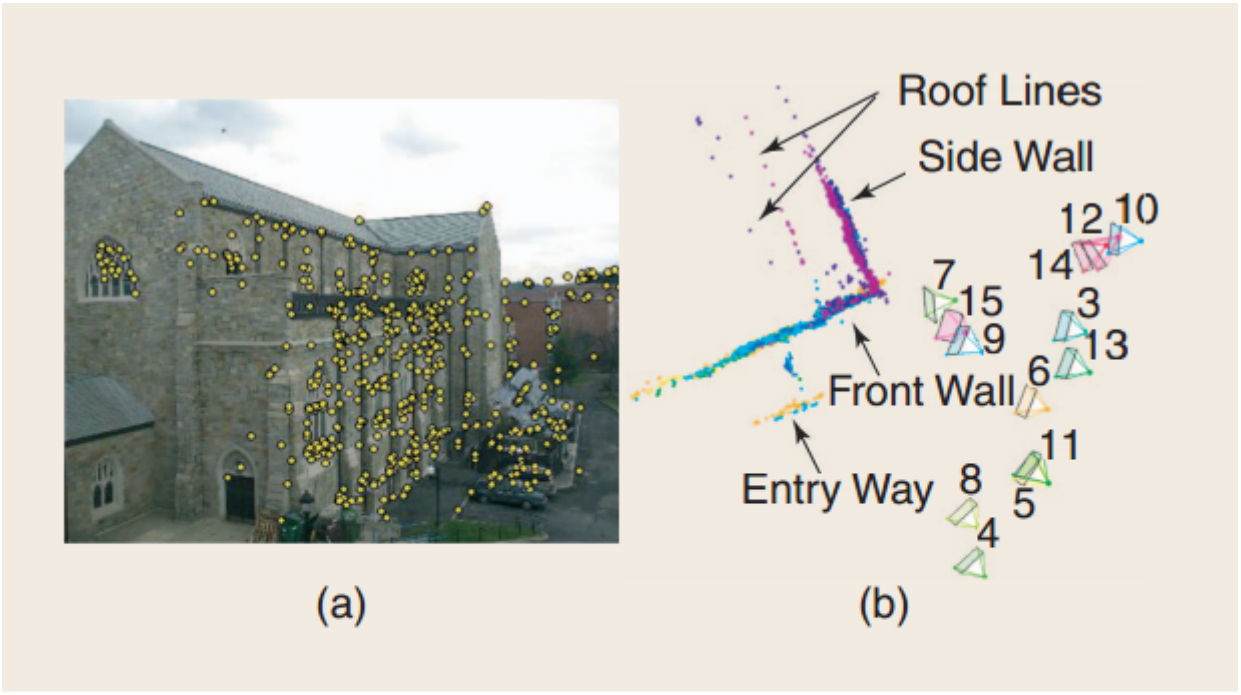
- Find some of the unknown calibration parameters in addition to the relative camera pose.
- Start from the calibrated camera and use its correspondences to localize and calibrate all its neighbors
- Information is propagated along a spanning tree of the network.

[TABLE 2] CALIBRATION ERRORS OBTAINED BY THE CSN CALIBRATION ALGORITHM OF [50] ON A NETWORK CONFIGURATION SIMILAR TO THAT IN FIGURE 7.

NOISE	0 PX	1 PX	2 PX	3 PX
ERRORS	0.00 (0.00)	0.81 (0.03)	2.13 (0.08)	4.38 (0.21)

THE POINT CORRESPONDENCES ARE CORRUPTED WITH GAUSSIAN NOISE WITH VARIANCE 0, 1, 2, AND 3 PX. WE REPORT THE MEAN AND VARIANCE (THE LATTER IN PARENTHESIS) OF THE CALIBRATION ERRORS (IN %) OF MORE THAN 100 REALIZATIONS OF NOISE.

Belief-propagation approach

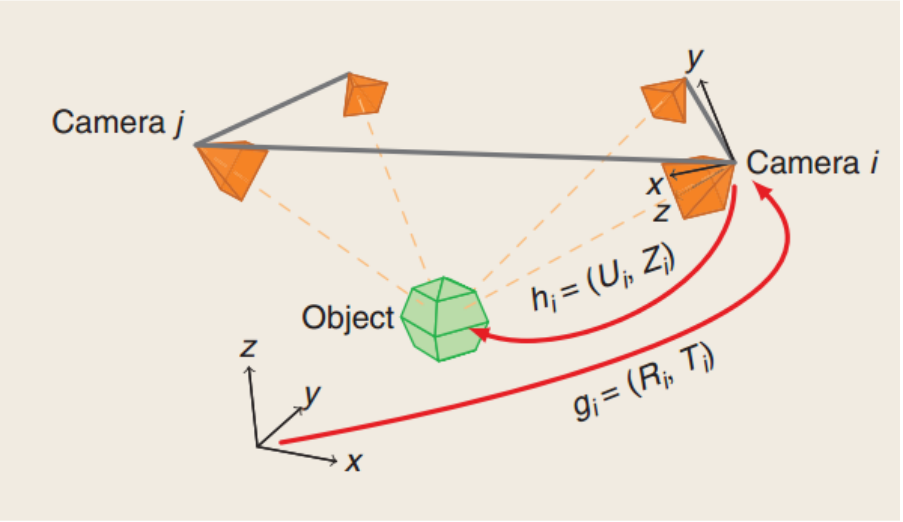


[FIG9] **Network localization and calibration results for the belief-propagation based approach of [51]. (a) Reconstructed 3-D points overimposed on one of the input images. (b) Reconstructed camera poses and 3-D points. (Photographs and images courtesy of Prof. R. Radke [51].)**

Distributed Object-pose Estimation

- Model

- Consensus on SE(3)
- Consensus on 3-D points



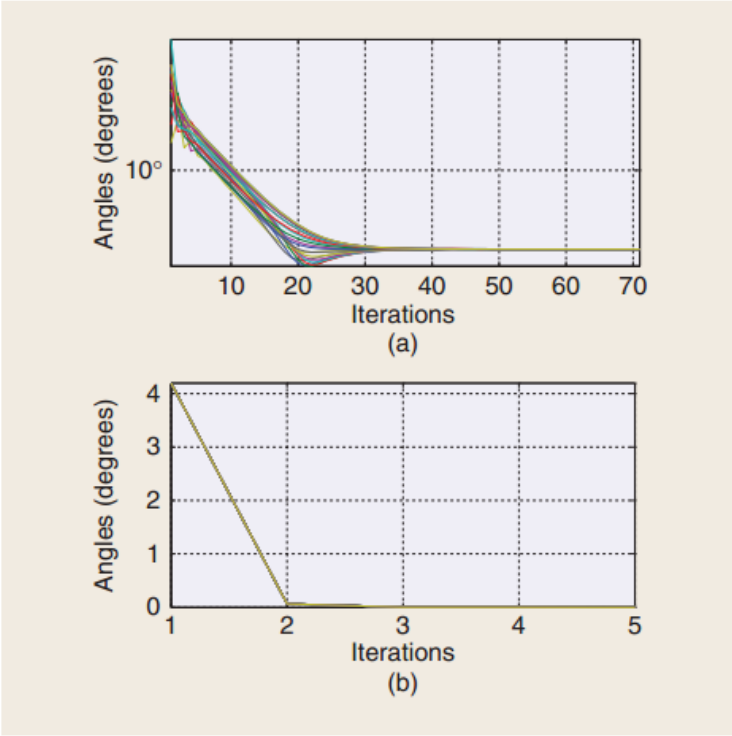
[FIG10] A schematic view of a CSN where each node can measure the pose of an object with known geometry.

Consensus on SE(3)

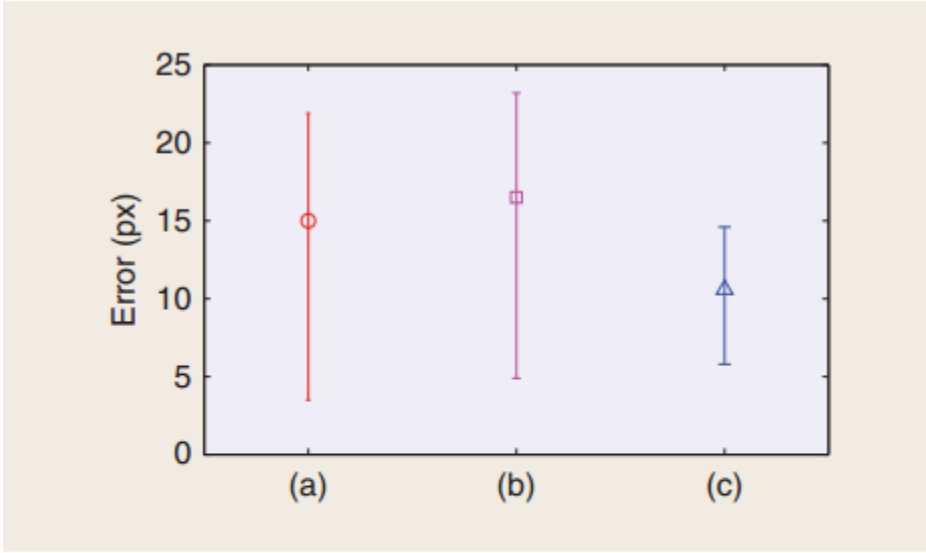
- Model
  - Consider the geodesic distance in the space of rotations SO(3), We denote such a distance as $d_{SO(3)}(R_i, R_j), R_i, R_j \in SO(3)$. We can then define the Fréchet mean $\bar{R} \in SO(3)$ of the N measurement U_i as the point in SO(3) that globally minimizes the sum of squared geodesic distances,

$$\bar{R} = \arg \min_{R \in SO(3)} \sum_{i \in \upsilon} d^2_{SO(3)}(R, R_i, U_i)$$

Consensus on 3-D points



[FIG11] An example of running consensus-based algorithms for distributed pose estimation in a CSN with $N = 20$ nodes in a ring topology. The plots show the rotation errors (in degrees) between the local estimate of the object orientation and the global Fréchet mean for (a) a direct extension of consensus to $SE(3)$ and (b) a distributed implementation of the centralized Fréchet mean algorithm, as described in [19].

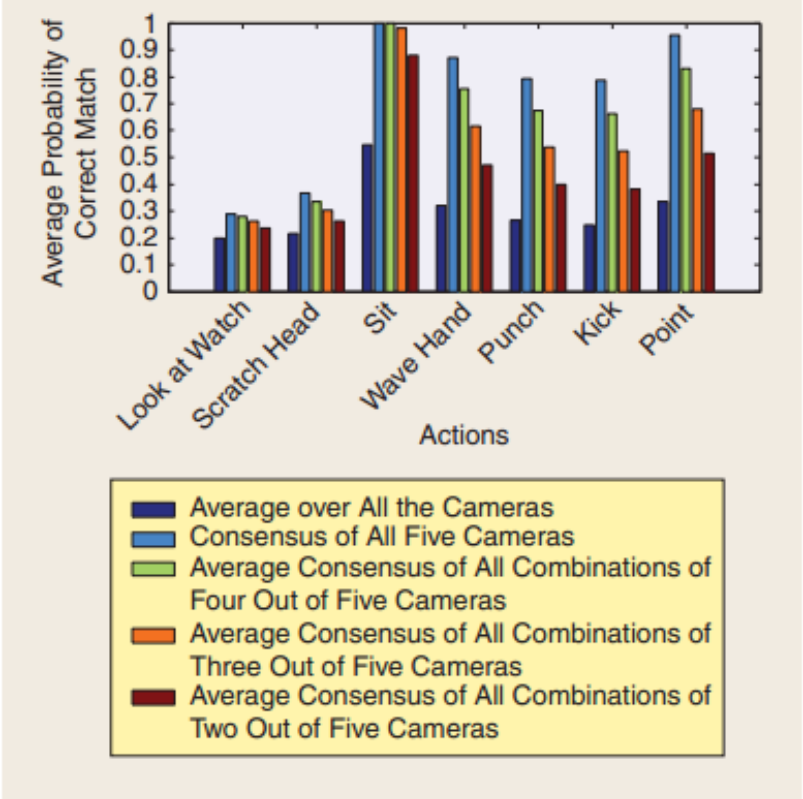


[FIG12] Comparison of consensus-based distributed object-pose estimation algorithms. The average reprojection errors (in pixels) are calculated on (a) the SoftPOSIT output before any consensus method is performed (red, [53]), (b) consensus by penalized world coordinates (magenta, [57]), and (c) consensus by the Frechét mean (blue, [19]). The error bars denote the first quartile, median, and third quartile. (Illustration courtesy of Dr. P. Burlina [57].)

Action Recognition

- Consensus-based method
- Feature histograms-based method

Consensus-based method



[FIG13] **Experimental results on distributed action recognition via consensus [58]. The plot shows a comparison of average probability of correct match for individual cameras and their consensus for all the activities. There are seven sets of bars for seven different actions. In each set, there are five bars where the leftmost one (blue) is the average probability of the correct match for individual cameras, and the next four bars are the average probability of the correct match of the consensus over all the combinations of cameras taking, respectively, five, four, three, and two out of five cameras. (Illustration courtesy of Prof. A.K. Roy-Chowdhury [58].)**

Feature histograms-based method

Conclusions and future directions

Distributed and Decentralized Multicamera Tracking

[Accurate and energy-efficient algorithms]

Background

- The decreasing cost of cameras and advances in miniaturization have favored the deployment of large-scale camera networks.
- The growing number of cameras enables new signal-processing applications that cooperatively use multiple sensors over wide areas.
- Object tracking is an important step in many applications related to security, traffic monitoring , and event recognition.

Introduction

- Discuss decentralized and distributed multicamera tracking approach
  - Cover common algorithmic steps
    - Calibration and Synchronization
    - The selection of fusion centers
  - Compare specific tracking approach.
    - Decentralized trackers
    - Distributed trackers
    - Quantifying communication and computation cost
    - Comparative example

Model

- Consider a network $C = \{C_1, \ldots, C_c, \ldots, C_N\}$ of N cameras monitoring T targets.
- The state of target *i* at time *k* be defined as $x_k^{v,i}$,where $v \in \{c, \pi\}$ represents either the *c*th camera view or an hypothetical top view π (plane).
- State: position, shape, velocity, size, contour, etc.
- Target state estimation on *v* aims to associate noisy measurement $Z_k^{v,i} = \{z_1^{v,i}, \ldots, z_k^{v,i}\}$ to obtain the trajectory $X_k^{v,i} = \{x_1^{v,i}, \ldots, x_k^{v,i}\}$ for each object *i.*

Calibration and synchronization

- Multicamera fusion can be performed through correspondence between measurement $z_k^{c,i}$ or trajectories.
  - Measurement correspondence:
    - Map the features $Z_k^c = \{z_k^{c,i} \mid i = 1, \ldots, M\}$ from each camera view to a common view V using a project matrix H,

$$z_l^{c,v,i} = \mathrm{H}^{c,v} z_k^{c,i}$$

    - The correspondence between feature points is performed using a similarity measure(Euclidean distance, color histogram similarity).
    - The occupancy mask of a target can be projected to obtain an aggregated occupancy on a common view.

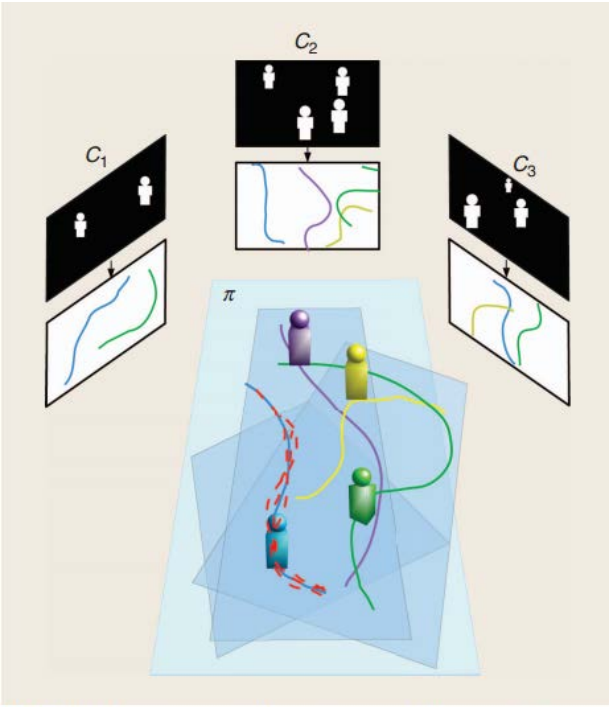
$$Z_k^{v,i} = \{z_k^{c,v,i} \mid c \in C_v^i\}$$

– Trajectory correspondence:
  - The states of the object are projected from each view to a common view using H,

$$x_l^{c,v,i} = \mathrm{H}^{c,v} x_k^{c,i}$$

  - After projection, the tracks $X_k^{c,v,i} = \{x_1^{c,v,i}, \ldots, x_k^{c,v,i}\}_{c=1,\ldots,N;i=1,\ldots,M}$ from different camera are put in correspondence.

– The projection matrix H:
  - Computed by selecting control point[24]
  - Using the scale-invariant feature transform(SIFT)
  - Using three-dimensional feature points
  - Using the relative position and orientation of the sensors in nonoverlapping cameras.



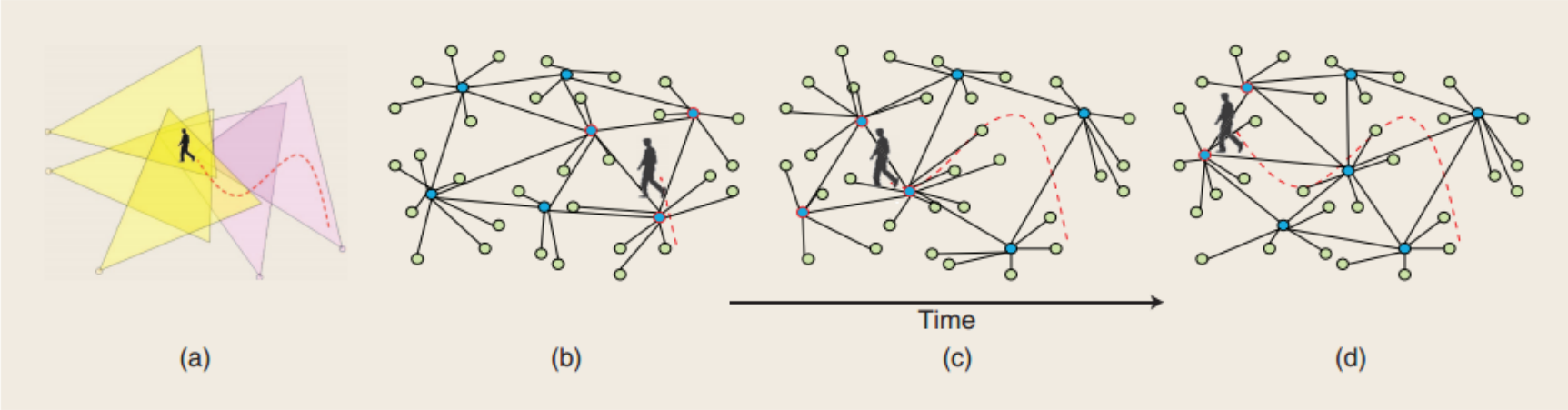
[FIG1] Trajectory correspondence among multicamera views.

- Synchronization
  - Through a centralized server
    - distributes timestamp information or through a event.
  - Automatic synchronization method
    - Introduce a temporal shift
    - To rectify the temporal shift between measurement.
    - Remaining temporal shifts are handled as uncertainty during target-state estimation.

Fusion centers

- Fixed fusion center
  - Higher processing power and energy supply
  - Generate lower-quality observations.
- Dynamically clustering
  - Based on trackablility measures
  - Best-view selection
  - Not necessarily use the best cameras for tracking
  - Online camera clustering
  - Improve scalability and robustness against node failures

Dynamic clustering example



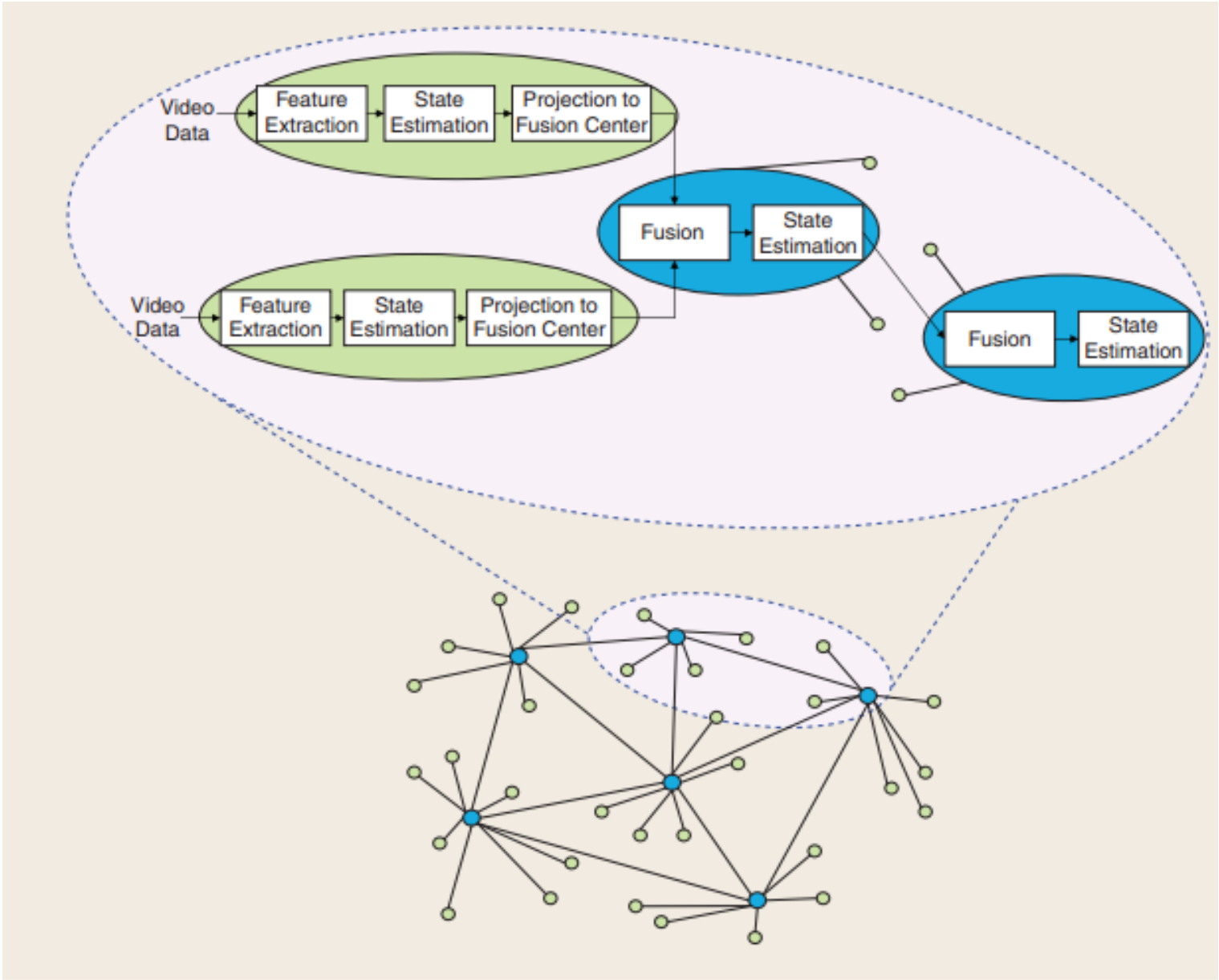
[FIG2] Multiple dynamic clusters observing a target. (a) Example with two camera clusters identified with yellow and violet triangles that represent the fields of view of the cameras. Note that despite the target being visible in three cameras of the yellow cluster and in only one camera of the violet cluster, the chosen active cluster could be the violet one as the target was first seen in this cluster. (b)–(d) Evolution of the camera cluster structure over time as a function of the target position (cameras: green dots; fusion centers: blue dots; active fusion center: red circles).

Decentralized Tracking

- Graph Matching(GM)
- Hidden Markov Model(HMM)
- Particle Filter(PF)
- Gaussian Mixture Particle Filter
- Tracking-Before-Detect Particle Filter(TBDPF)
- Kalman Filter(KF)

[TABLE 1] MAJOR STEPS OF TARGET STATE-ESTIMATION ALGORITHMS.

ALGORITHM	TYPE	MAIN STEPS	
GM	DETERMINISTIC	BIPARTITE GRAPH	MAXIMUM PATH COVER
HMM	PROBABILISTIC	EMISSION PROBABILITIES	VITERBI DECODING
		$p(z_k^{\nu,i} \mid \mathbf{x}_k^{\nu,i} = j)$	
PARTICLE FILTER	PROBABILISTIC	PREDICTION	UPDATE
		$\mathbf{x}_{k\mid k-1}^{\nu,i} = \mathbf{F}_k \mathbf{x}_{k-1\mid k-1}^{\nu,i} + \mathbf{v}_k$	$\omega_t^{\nu,i,r} \propto \omega_{k-1}^{i} \frac{p(z_k \mid \mathbf{x}_k^{\nu,i,r-1}) p(\mathbf{x}_k^{\nu,i,r} \mid \mathbf{x}_{k-1}^{\nu,i,r})}{q(\mathbf{x}_k^{\nu,i,r} \mid \mathbf{x}_{k-1}^{\nu,i,r}, z_k)}$
KF	PROBABILISTIC	PREDICTION	UPDATE
		$\mathbf{x}_{k\mid k-1}^{\nu,i} = \mathbf{F}_k \mathbf{x}_{k-1\mid k-1}^{\nu,i} + \mathbf{v}_k$	$\mathbf{x}_{k\mid k}^{\nu,i} = \mathbf{x}_{k\mid k-1}^{\nu,i} + \mathbf{K}_k (z_k^{\nu,i} - \mathbf{A}_k \mathbf{x}_{k\mid k-1}^{\nu,i})$
		$\Omega_{k\mid k-1}^{\nu,i} = \mathbf{F}_k \Omega_{k-1\mid k-1}^{\nu,i} \mathbf{F}_k^T + \mathbf{Q}_k$	$\Omega_{k\mid k}^{\nu,i} = (\mathbf{I} - \mathbf{K}_k) \mathbf{A}_k \Omega_{k\mid k-1}^{\nu,i}$



[FIG3] **Data fusion and processing steps in decentralized multicamera tracking.**

Graph Matching

- Model
  - Vertices: measurements at the entry and exit point.
  - The edge between two vertices z_a and z_b, is weighted by their similarity $\zeta(z_a, z_b)$
- Algorithm
  - find the maximum-weight path cover over a certain number of consecutive time steps
  - The complexity: $O(n^{2.5})$
- Improve
  - Deal with Overlapping field of view: projected onto a top-view plane π,then GM is applied on generated occupancy map.
  - Improve accuracy: using a multilevel; the collaboration between the cameras and their fusion center.

Hidden Markov Model(HMM)

- Model
  - The search area(occupancy mask) is divided into a regular grid(C,V).
  - Estimate the probability of the measurement and the state ending at location *a* at time *k.* The posterior is

$$p(x_k^{\pi,i} \mid \Psi_k^{\pi}) = p(\Psi_k^{\pi} \mid x_k^{\pi,i} = a) \max_k p(x_k^{\pi,i} = a \mid x_{k-1}^{\pi,i} = b) \times p(x_{k-1}^{\pi,i} \mid \Psi_{k-1}^{\pi})$$

    - $\Psi_k^{\pi} = \{\mathrm{I}_k^{c,\pi} \mid c \in C_v\}$ is the set of projected occupancy masks from the set C_v of cameras observing the target.
    - $p(\Psi_k^{\pi} \mid x_k^{\pi,i} = a)$ is computed based on the color similarity
    - Solved by the Viterbi algorithm
  - Then project to each view for validation.
  - Drawback: Delay

Particle Filter(PF)

- Sampling importance resampling(SIR)
  - Estimate the posterior probability density function,

$$p(x_k^{v,i} \mid Z_k^{v,i}) \approx \sum_{r=1}^{R} w_k^{v,i,r} \delta(x_k^{v,i} - x_k^{v,i,r})$$

  - The particle weights are updated based on the likelihood as

$$w_k^{v,i,r} \propto w_{k-1}^{v,i,r} \frac{p(z_k^{v,i} \mid x_{k-1}^{v,i,r}) p(x_k^{v,i,r} \mid x_{k-1}^{v,i,r})}{q(x_k^{v,i,r} \mid x_{k-1}^{v,i,r}, z_k)}$$

    - Where $q(\cdot)$ is the importance density function.
    - $p(z_k^{v,i} \mid x_k^{v,i})$ is the likelihood function.
    - $p(x_k^{v,i} \mid x_{k-1}^{v,i})$ is the transition density defined by the target motion.

- Adopted strategy
  - Running one PF on the fusion center
    - Project the object height onto the top view, then backproject to the view.
    - The final likelihood for the weight update is computed as

$$p(z_k^{\pi,i} \mid x_{k-1}^{\pi,i}) = \prod_{c \in C_v} e^{-d(H^{c,i}, H^{*c,i})^2}$$

      - Using an distance based on the Bhattacharyya coefficient.
  - Using multiple centers fusion
    - The product of likelihoods $\prod_r p(z_k \mid x_k^r)$, can be approximated with a parametric model $\varsigma^c(x_k^c; \phi_k^c)$
      - Where ϕ_k^c are the learned parameters for camera c.
  - Quantization, vectorization and parameterzation.

Gaussian Mixture Particle Filter

- PF is computationally expensive because of the use of multiple particles.
- Individual PFs can run in parallel in each node.
- The local sufficient statistics(belief) is approximated by a low-dimensional Gaussian mixture model as

$$p(x_k^{v,i} \mid Z_k^{v,i}) = \sum_{\theta=1}^{\Theta} w_k^{v,i,\theta} N(\mu_k^{v,i,\theta}, \sigma_k^{v,i,\theta})$$

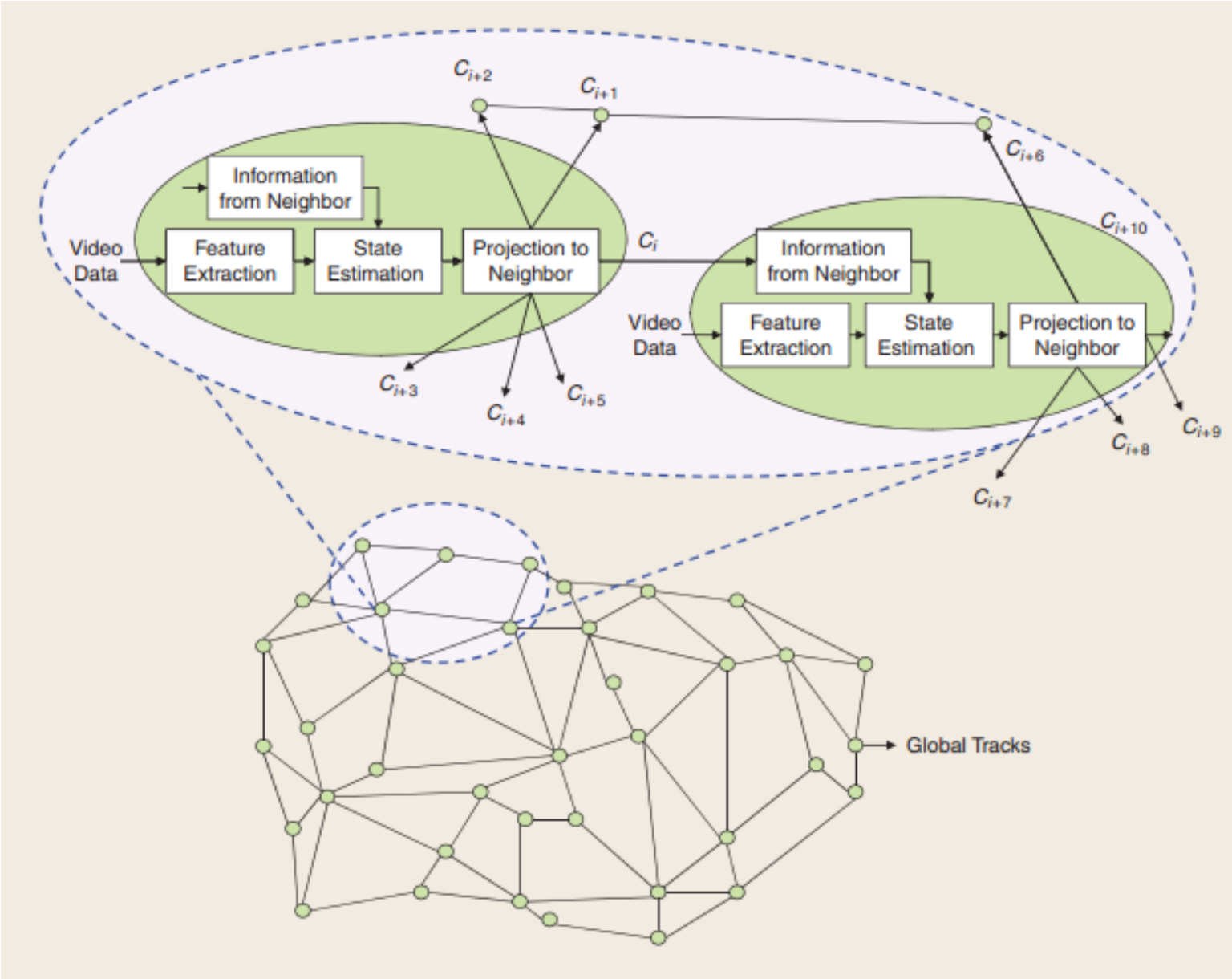
  - Where θ is the number of Gaussians
  - $(\mu_k^{v,i,\theta}, \sigma_k^{v,i,\theta})$ are the mean and standard deviation of the Gaussian.
  - Converge almost surely to the posterior distribution estimated with a centralized Bayesian formulation.

Tracking-Before-Detect Particle Filter

Kalman Filter

Distributed Tracking

- Particle Filter
- Kalman Consensus Filter



[FIG4] **Data fusion and processing steps in distributed multicamera tracking.**

Particle Filter

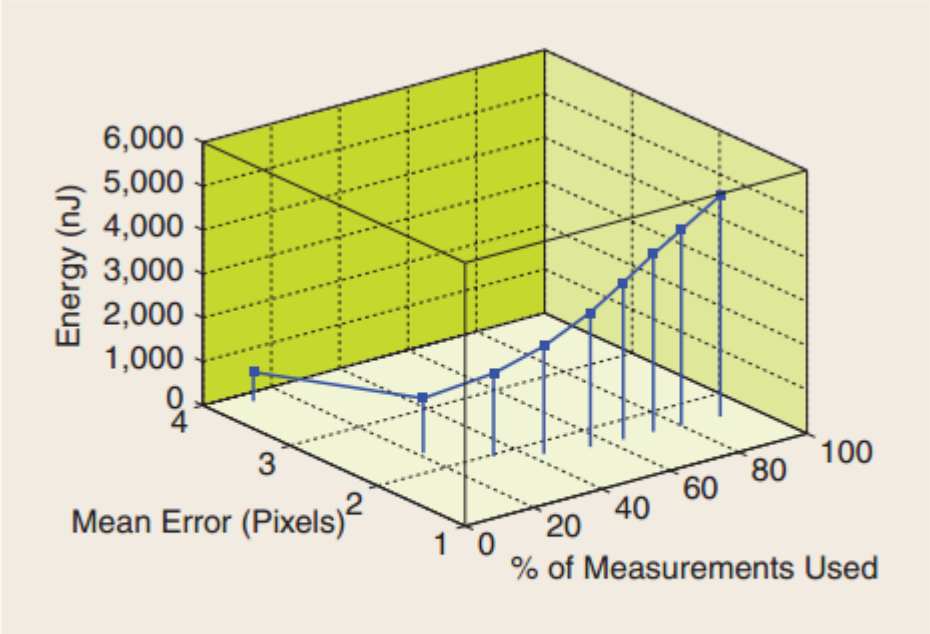
Kalman Consensus Filter

Quantifying communication and Computation Costs

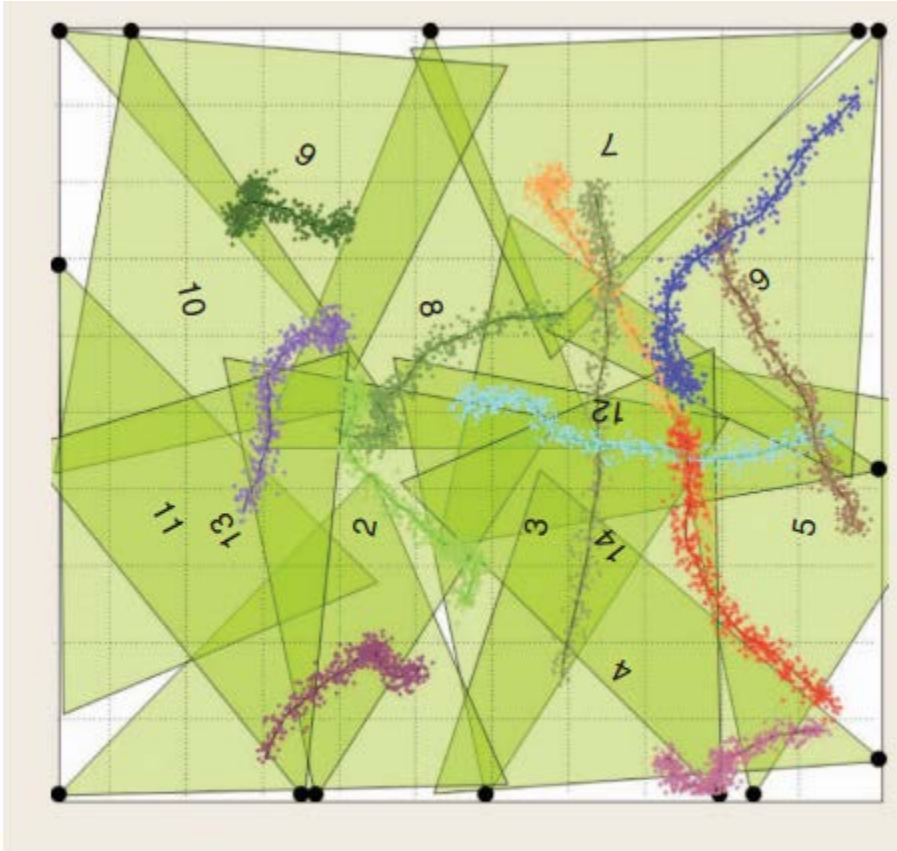
- Modeling Costs
- Comparative example
- Discussion

Modeling Costs

Comparative example



[FIG5] Communication cost versus accuracy while increasing the number of measurements for KCF. Note that the rate of accuracy improvement with respect to the communication cost is much reduced after adding more than 50% of the total measurements.



[FIG6] Multicamera network with $N = 14$ and $T = 12$ targets. Tracks are shown with positions corrupted by various noise levels.

Discussion