Adaptive Workload Equalization in Multi-Camera Surveillance Systems

Pradeep K. Atrey
University of Winnipeg, Canada
p.atrey@uwinnipeg.ca
www.acs.uwinnipeg.ca/pkatrey/

Winnipeg


UWINNIPEG
THE HEART OF THE CITY,
the heart of the continent
THE UNIVERSITY OF WINNIPEG


Fort McMurray
Manitoba
Saskatchewan
Grande Prairie
Prince George
Edmonton
Flin Flon
Thompson
Red Deer
Prince Albert
Kamloops
Calgary
Saskatoon
Victoria
Vancouver
Swift Current
Regina
Seattle
Spokane
Brandon
Winnipeg
Ontario
Timmins
Amos
Quebec
Fredericton
Charlott
Halifax
Vancouver
Portland
Williston
Helena
Grand Forks
Minnesota
Duluth
Ottawa
Montreal
Augusta
Butte
Bismarck
Concord
Boston
Oregon
Boise
St. Paul
Minneapolis
Toronto
Rochester
Providence
Twin Falls
Milwaukee
Detroit
Buffalo
Hartford
Reno
Salt Lake City
Cheyenne
Scottsbluff
Omaha
Chicago
Cleveland
New York
Philadelphia
Oakland
Provo
Boulder
Pittsburgh
Baltimore
Indianapolis
Dayton
Washington D. C.
Denver
Kansas City
Cincinnati
Richmond
Norfolk
Las Vegas
St. Louis
Louisville
U. S. A.
Santa Fe
Nashville
Los Angeles
Oklahoma City
Charlotte
San Diego
Phoenix
Albuquerque
Memphis
Columbia
Mexicali
Tucson
Little Rock
Birmingham
Atlanta
Dallas
ATLANTIC
Jackson
Tallahassee
Jacksonville
OCEAN
Hermosillo
Austin
Baton Rouge

Winnipeg Summer

Winnipeg Winter

THE UNIVERSITY OF WINNIPEG

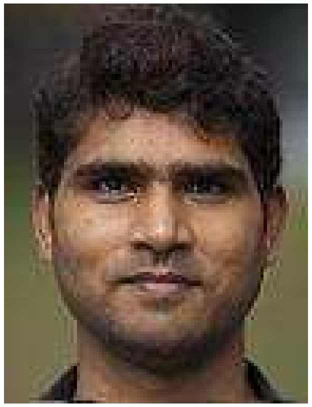
Acknowledgement

- This research is partly supported by

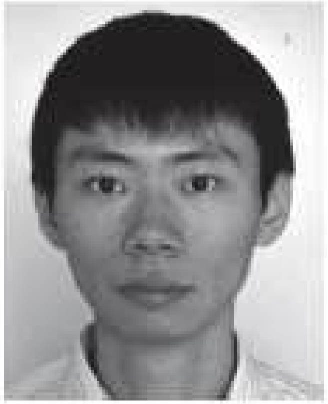
Natural Sciences and Engineering Research Council of Canada

Conseil de recherches en sciences naturelles et en génie du Canada

Other Contributors



Mukesh Saini



Xiangyu Wang

Mohan Kankanhalli

NUS
National University of Singapore

Outline

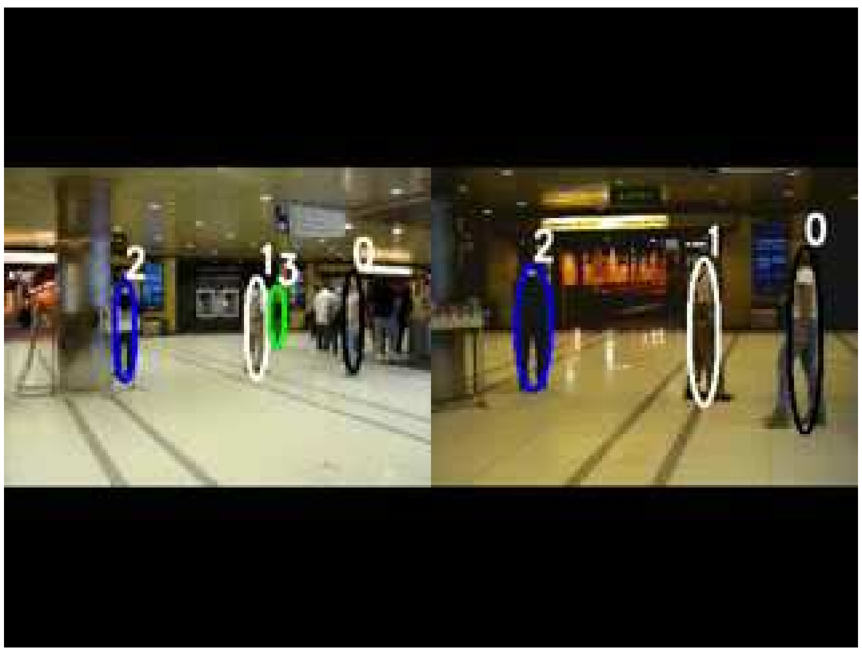
- Introduction and Motivation
- Related Work
- Workload Model
- Dynamic Load Sharing
- Conclusions

Outline

- **Introduction and Motivation**
- Related Work
- Workload Model
- Dynamic Load Sharing
- Conclusions

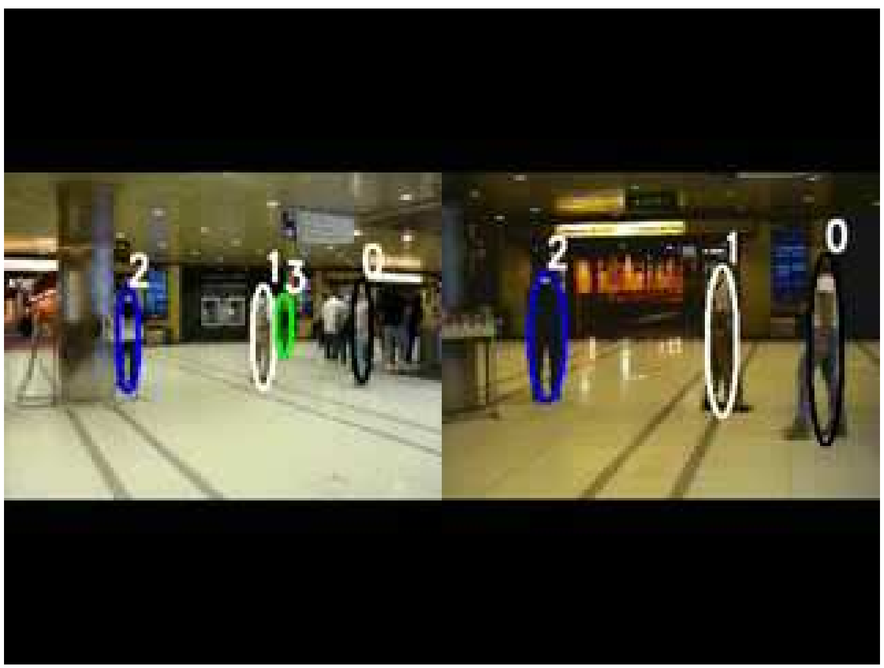
Automated Surveillance

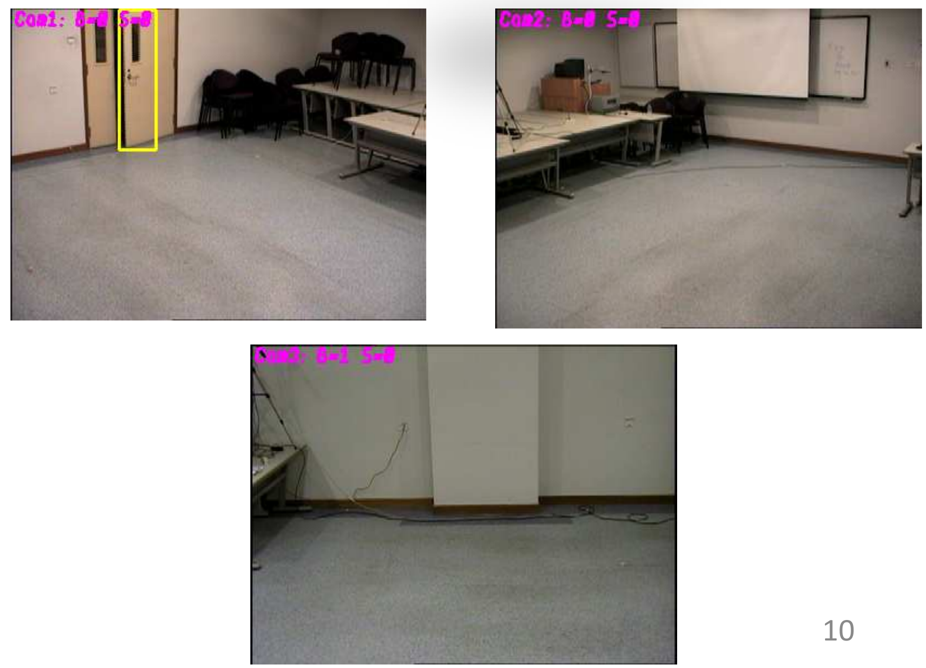
- Large number of cameras
  - Detection/Recognition
  - Tracking
  - Activity Analysis



Automated Surveillance

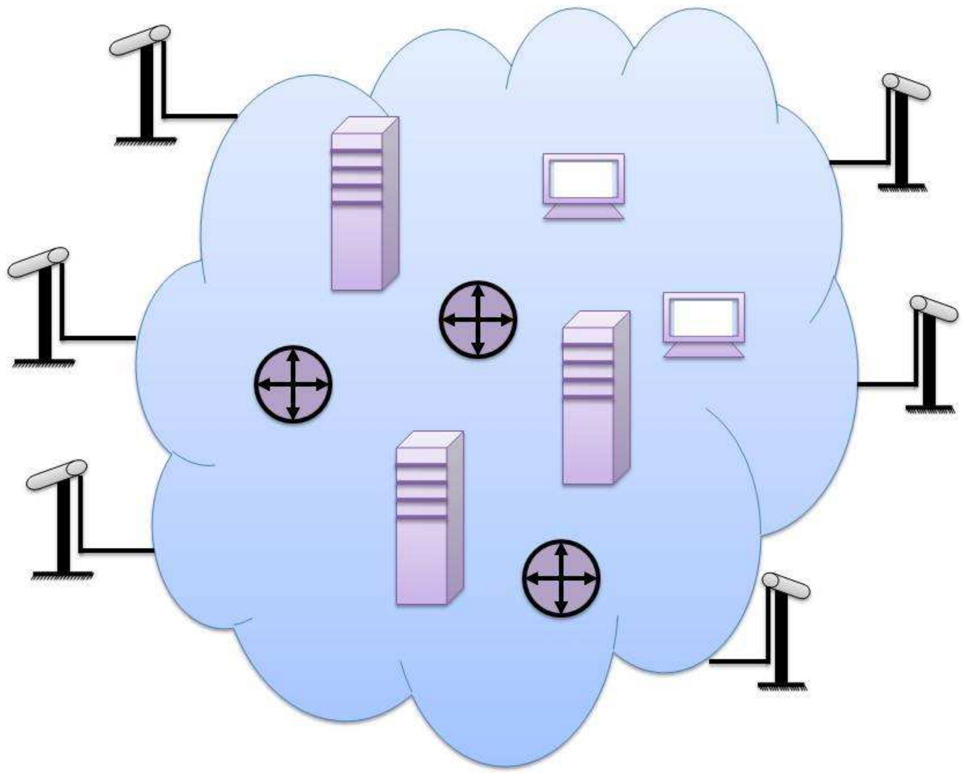
- Large number of cameras
  - Detection/Recognition
  - Tracking
  - Activity Analysis





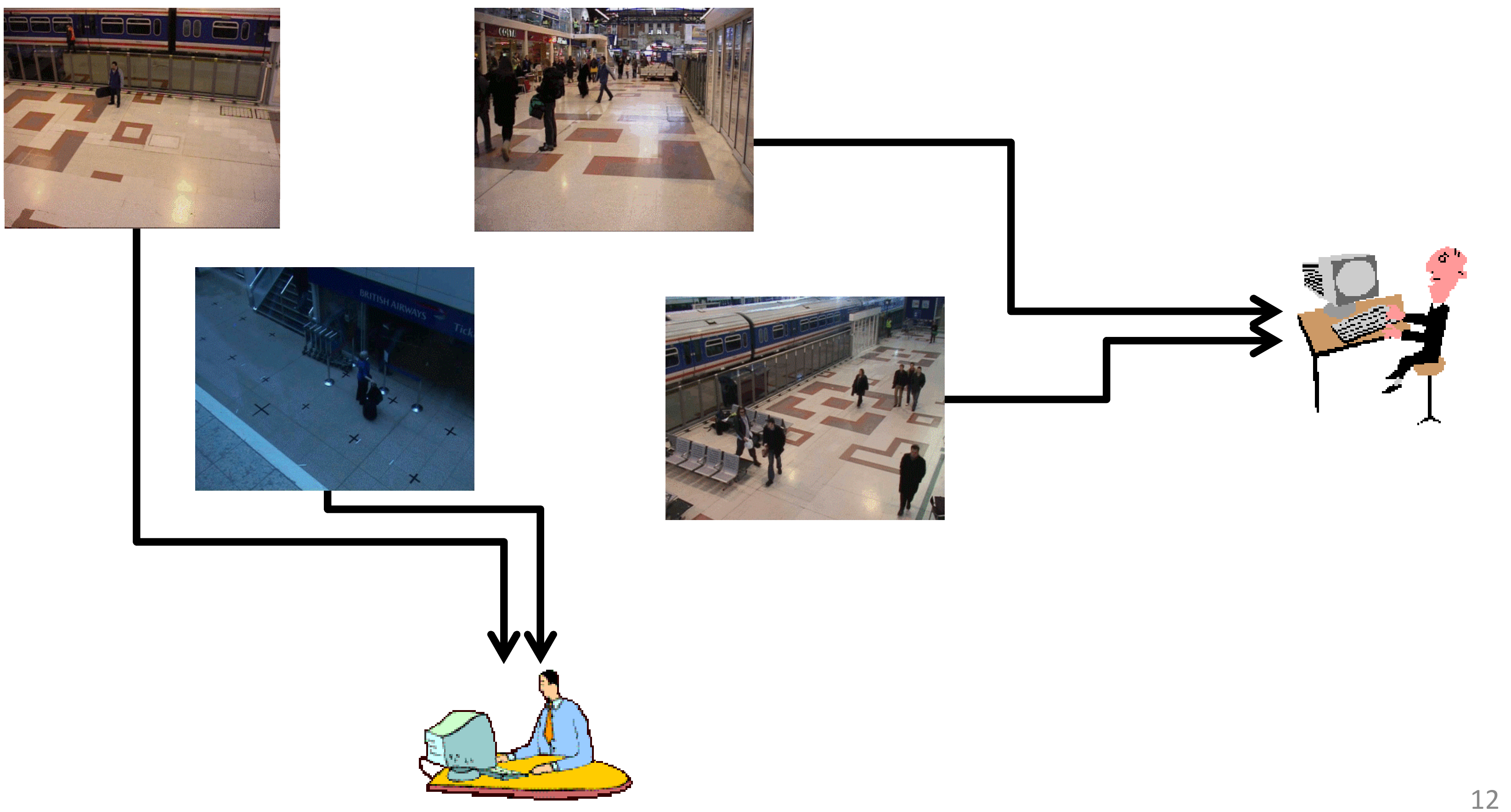
Current Trend: Distributed Processing

- IP Cameras and distributed processors



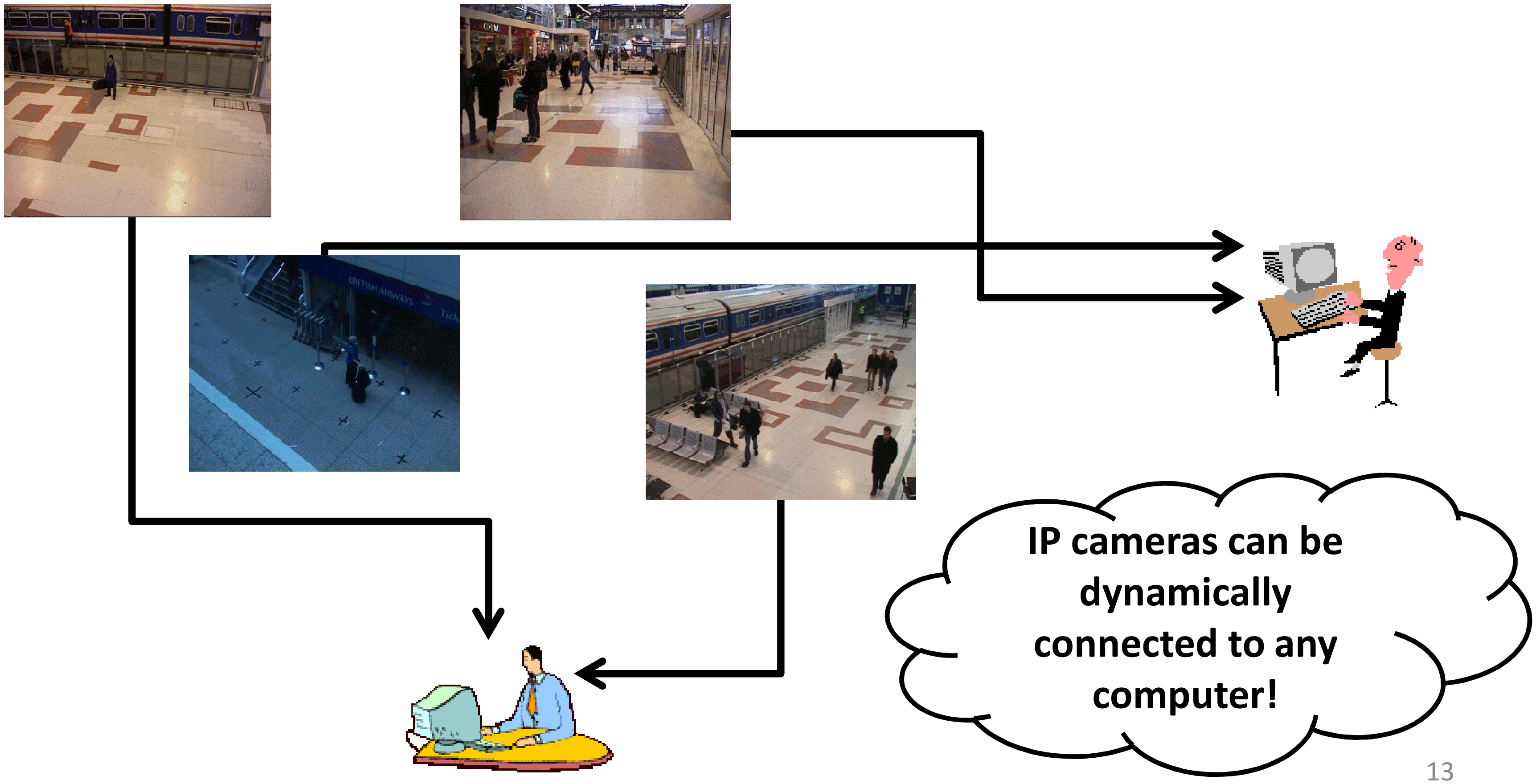
What is the Problem?

- Host (machine or human operator) to camera ratio is generally fixed $\Rightarrow$ workload imbalance



Solution?

- Dynamic assignment of cameras-to-hosts based on workload



Key Research Questions

Smoking is not good for health

- Workload
  - How does workload vary with time?
  - What does it depend on?
- Dynamic Scheduling
  - How to schedule the cameras to hosts in a dynamic manner?

Outline

- Introduction and Motivation
- **Related Work**
- Workload Model
- Dynamic Load Sharing
- Conclusions

Previous Workload Models

- GMM
- Multi-class
- Cluster based

Do not capture dynamics

- Markov Model

Need to answer what does it depend on!

Novelty over existing works

Comparison With Previous Works

Work	Workload modeled?	Semantic considered?	Dynamic assignment	CCTV Application?
Soldatini et al. [6]	No	No	No	No
Marcenaro et al. [7]	Static	No	No	No
Detmold et al. [8]	No	No	No	No
Collins et al. [9]	No	No	No	No
Marchesotti et al. [10]	No	No	No	No
Trivedi et al. [11]	No	No	No	No
Dias et al. [12]	No	No	No	No
Calderara et al. [13]	No	No	No	No
Saini et al. [2]	Static	Yes	No	No
Chang et al. [14]	Static	No	No	No
Proposed Method	**Adaptive**	**Yes**	**Yes**	**Yes**

Key Contributions

- The Markov chain based workload model which exploits the environment semantics to capture the variability of the workload.
- Dynamic load sharing methods which equalize the workload of hosts (or processors) to improve the surveillance performance.

Outline

- Introduction and Motivation
- Related Work
- **Workload Model**
- Dynamic Load Sharing Method
- Conclusions

Workload Modeling as Markov Chain

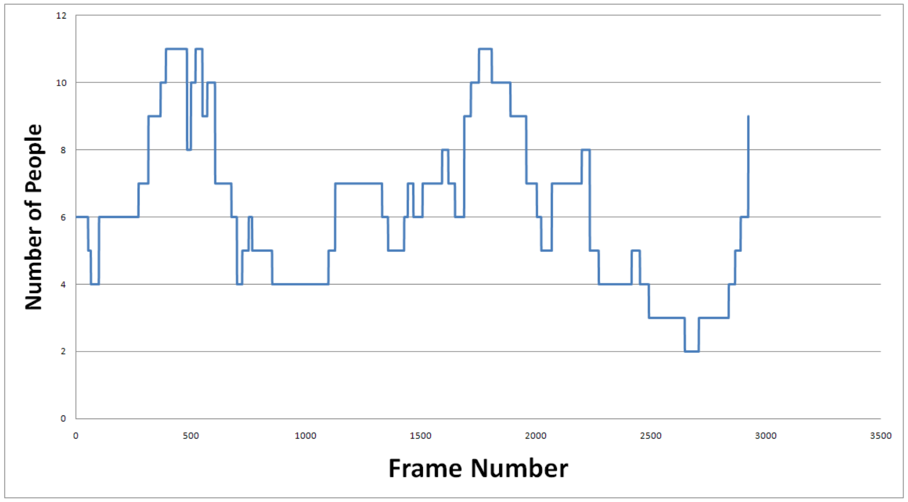
- Semantic characteristics of the workload are captured in a Markov chain, with states representing the number of targets in the environment.

Target Flow Graph

- An operating scenario is represented by a Target Flow graph (TFG), which consists of a set of tuples

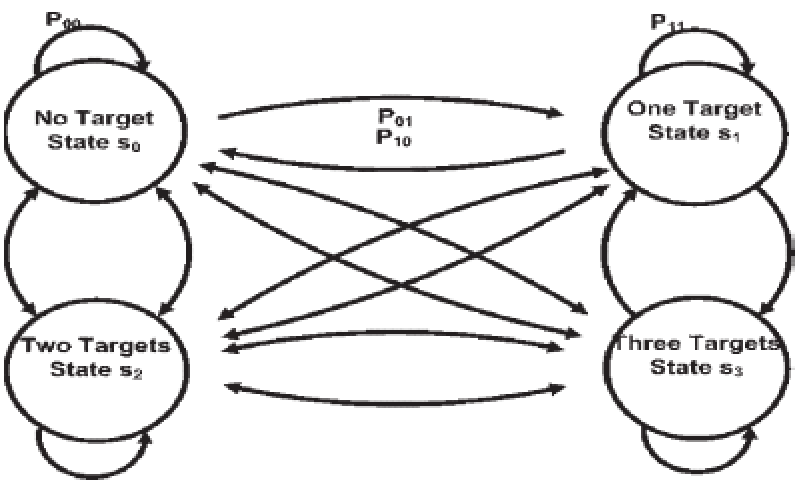
$$TFG = \{(\tau_k, g_k) \mid k \in [1, l]\}$$

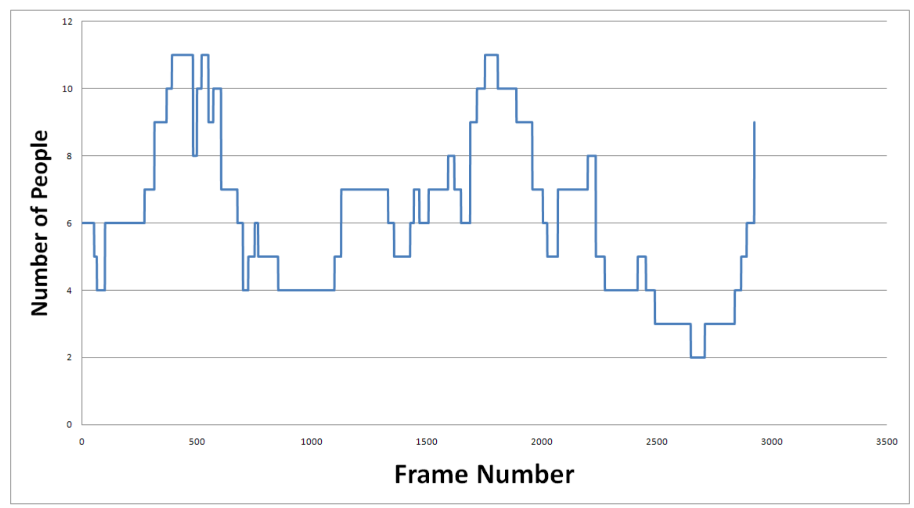
where g_k is the number of targets at time τ_k, and l *is* the total number of observations.



Markov Chain Construction

- The processing time and memory requirement for each frame mainly depend on the number of targets in the surveilled area.
- A Markov chain preserves the temporal behavior of the workload in its states and thus can capture the variability of the workload.





Markov Chain Construction 2

- The number of states in a Markov chain is *m*+1, where *m* is determined by the following equation:

  $$m = \max\{g_k \mid (\tau_k, g_k) \in TFG, k \in [1, l]\}.$$

  In other words, *m* is the maximum number of targets expected in the surveilled area.
- The set of states of this Markov chain can be defined as follows:

  $$S = \{s_0, s_1, \ldots s_m \mid \forall i, j \in [0, m], s_i = i, s_i \neq s_j\}.$$

Markov Chain Construction 3

- The transition probabilities are represented in the form of a matrix .

$$\chi = \left\{ p_{ij} \mid p_{ij} = \frac{n_{ij}}{n_i},\ i, j \in [0, m] \right\}.$$

 where n_i is the number of times the camera is in state s_i and n_{ij} is the number of times the camera transiting from state s_i to state s_j

- Let $\Pi = (\pi_1, \pi_2, \ldots \pi_n)$ be the steady state probabilities of the states.

$$\Pi = \{\pi_i \mid \pi_i = p'_{ij}, p'_{ij} \in \chi^{Inf},\ i = C, j \in [0, m]\}$$

Model Validation

- Experimental setup

Specifications of the System

Operating System	Microsoft Windows XP
Platform	Visual C++ 2008
Additional Libraries	OpenCV
Computer	Intel(R) T2300 @ 2.33GHz, 0.99GB Ram
Image Resolution	320 × 240 captured by AXIS IP camera

- We first record the target flow pattern and construct the TFG. The TFG is then used to calculate the transition and steady state probabilities.

Model Validation 2

- We calculate the mean and variance of the processing time.

STATE-WISE VALUES OF MEAN AND VARIANCE OF PROCESSING TIMES

State/Targets	μ(milliseconds)	σ(milliseconds)
0	496	24
1	518	30
2	557	30
3	662	38
4	733	78

- Processing time α Number of targets.

Outline

- Introduction and Motivation
- Related Work
- Workload Model
- **Dynamic Load Sharing**
- Experiments
- Conclusions

Dynamic Load Sharing: Goal and Need

- Goal: to have similar numbers of targets to be processed by each host
- The workload model is used for dynamically assigning the video streams to hosts to equalize the workload.
- The amount of resources required depends on the state of the environment being observed by the camera.
- These states can be dynamically calculated for each camera.

Equalization Function

- Now, if there are N_{pc} processors and $C_{pc}(j)$ is the set of cameras assigned to the jth processor, our objective is to find an assignment scheme which maximizes the equalization function:

$$E_{pc} = \frac{1}{(L_{av} + \Delta)^{N_{pc}}} \prod_{j=1}^{N_{pc}} \left(\sum_{\forall k; c_k \in C_{pc}(j)} s(c_k) + \Delta \right)$$

$$\text{with} \quad L_{av} = \frac{1}{N_{pc}} \sum_{j=1}^{N_{pc}} L(j) \qquad\qquad L(j) = \sum_{\forall k; c_k \in C_{pc}(j)} s(c_k)$$

- where c_k is the kth camera, is $s(c_k)$ the state of that camera, and Δ is a small number that accounts for the analysis workload when there are no targets.

Dropped Targets

- Apart from equalization, we analyze the number of targets dropped for a given processor limit of *L*th targets at a time:

$$\mathcal{N}_d = \sum_{j=1 \text{ to } N_{pc} \text{ and } L(j)>L_{th}} |L_{th} - L(j)|.$$

Two Main Questions

- Two main questions:
  - How do we do the reassignment?
    - We present three methods (DAS, IDAS and Spiral) for camera assignment
  - When do we recalculate the transition probabilities and do the reassignment?
    - The transition probabilities of the states play an important role in workload equalization
    - We propose an Adaptive Reassignment Strategy based on workload.

Camera Assignment Methods

- Camera Assignment Methods
  - A naive approach - perform the reassignment at each time instant, based on the current number of targets.
    - ensure an equalized, number of targets, but it will cause a large number of camera switching.
  - We describe three methods for camera assignment in which the reassignment is done based on the current workload as well as the future expected workload
    - Divide and Swap (DAS) method
    - Improved Divide and Swap (IDAS) method
    - Spiral method

Camera Assignment Methods 2

- DAS method
  - In this method, the processors are divided into overloaded and workload deficient groups, and then state transition probabilities are compared against thresholds for swapping the cameras between two types of processors.
- IDAS method
  - Remove the probability thresholds which are difficult to compute. Hence, it is a more generic method.
  - Perform the reassignment in decreasing order of workload in the overloaded processors and increasing order in the load deficient processors until one of these groups is empty.

Camera Assignment Methods 3

- Spiral method
  - The IDAS method tries to respect the old camera-to-processor assignment while doing equalization; therefore, it is locally applied at the processor level.
  - In the spiral method, we give more importance to the equalization and take the problem at a global level.
  - Let $\mathcal{R} = \{c_1, c_2, \ldots c_{N_{cm}}\}$ be the set of cameras arranged in decreasing order of the workload, the current assignment for the *j*th processor is calculated as

$$C_{pc}(j) = \left\{ c_k, \forall k; (k-j)\%\rho = 0,\ \rho = \frac{N_{cm}}{N_{pc}} \right\}$$

  - Drawback: Does not respect existing camera assignment, hence causes more switching.

Camera Assignment Methods 4

- Time complexities of the three methods
  - DAS: $O(N_{pc} \times N_{cm})$
  - IDAS: $O(N_{pc} \times (m+N_{cm}))$
  - Spiral: $O(m \times N_{cm})$
- Since m is usually significantly less than N_{pc} and N_{cm}, Spiral method would be faster than DAS and IDAS.
- However, IDAS would need more time to calculate the expected workload (term m), and would therefore be slower than DAS.

Adaptive Reassignment Strategy

- Monitor equalization error over time.
- Whenever this error becomes more than some threshold, we recalculate the transition matrix and perform the reassignment.
- We show through experiments how to obtain the optimal window length to calculate the transition probabilities threshold for comparing equalization error.

Experiments and Results

- Objective:
  - To demonstrate the utility of the three proposed methods
- Experimental Setup:
  - We simulate a distributed surveillance system with 100 cameras and 20 processors (N_{pc} = 20).
  - All processors are assumed to be of equal power (m = 15).
  - We keep the number of cameras connected to the processors fixed to five and vary their assignment to processors in reassignment phase.
  - Δ is assumed to be 1.

Experiments and Results 2

- Performance Measures:
  - We use four performance measures to evaluate our methods:
  - Equalization error (E_{pc}),
  - Number of targets dropped $(\mathcal{N}_d)$,
  - Number of cameras switched $(\mathcal{N}_s)$, and
  - Number of reassignment instances $(\mathcal{N}_i)$

Image source: http://www.nwlink.com/~donclark/hrd/isd/analysis_iStock.jpg

Experiments and Results 3

- Data Set:
  - Five different videos from PETS, each of which consists of 2000 frames taken at 2000 time instants; and five real surveillance video clips consisting of 5000 frames each.
  - We extracted the blob information from these videos and simulated a distributed system in Matlab to evaluate the performance of the proposed methods.
  - For the PETS dataset, 20 cameras were simulated using each video. The data for 20 cameras is obtained using the same video but shifting the time axis and copying.

PETS: www.cvg.cs.rdg.ac.uk/slides/pets.html

Experiments and Results 4

- TFG

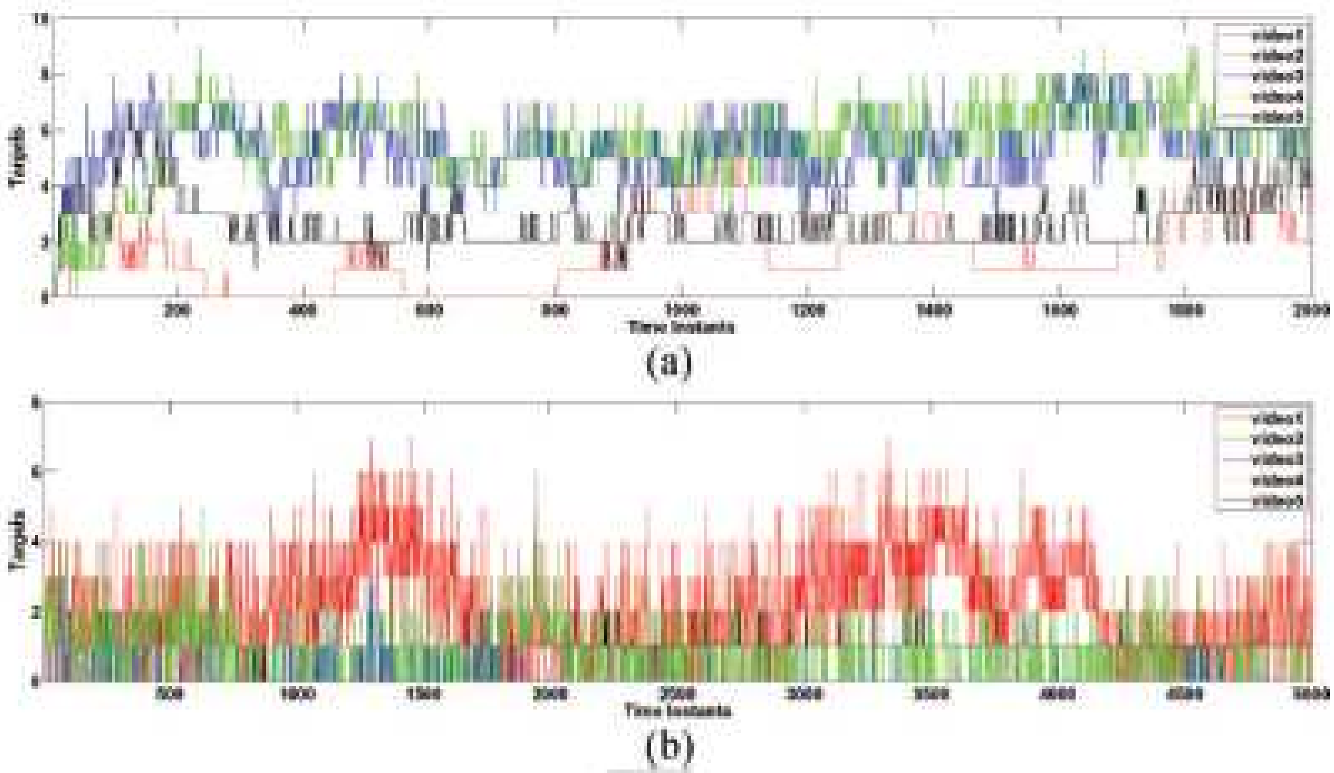


Fig. 3. Target flow graph of the five PETS and five real surveillance and video clips. (a) PETS data. (b) Real data.

Experiments and Results 5

- E_{pc} (Effect of static camera assignment)

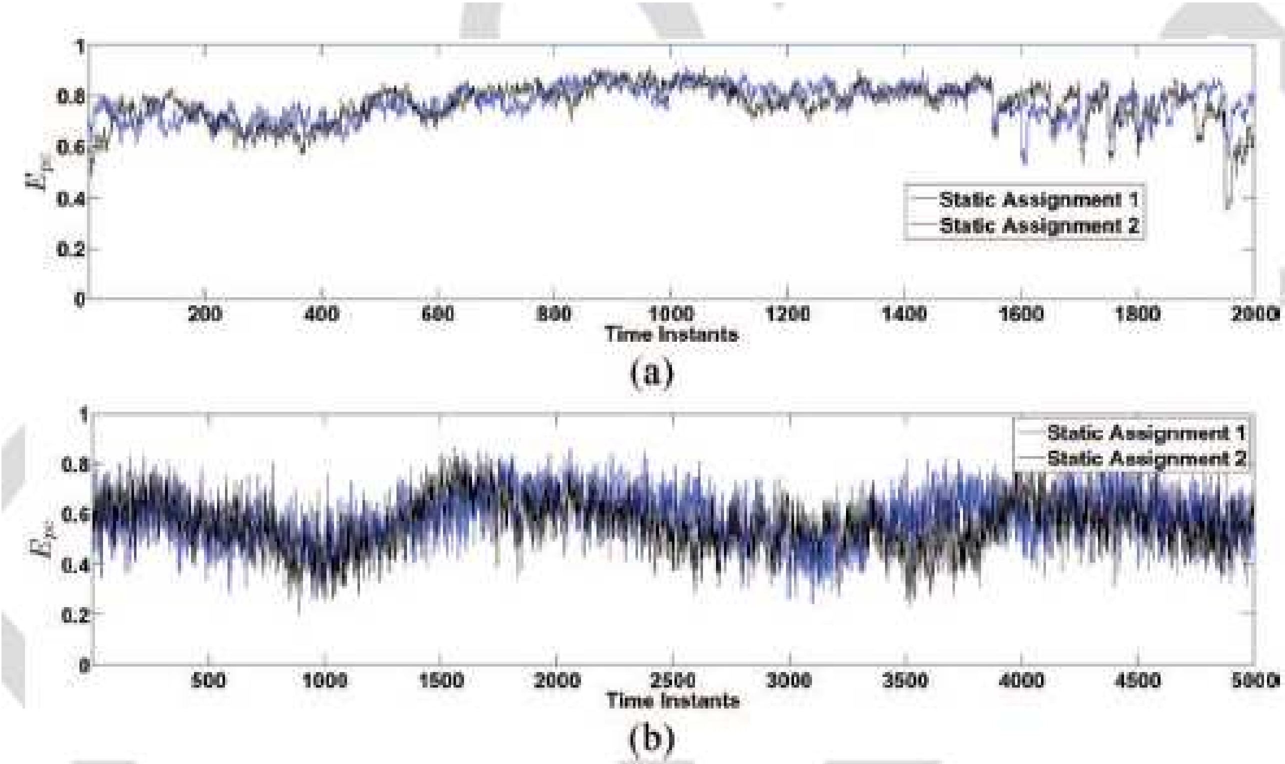


Fig. 4. Equalization E_{pc} for two random static camera assignments. (a) PETS data. (b) Real data.

Both assignments work well in patches and it is hard to find the better one

Experiments and Results 6

- Evaluation of DAS, IDAS and Spiral methods

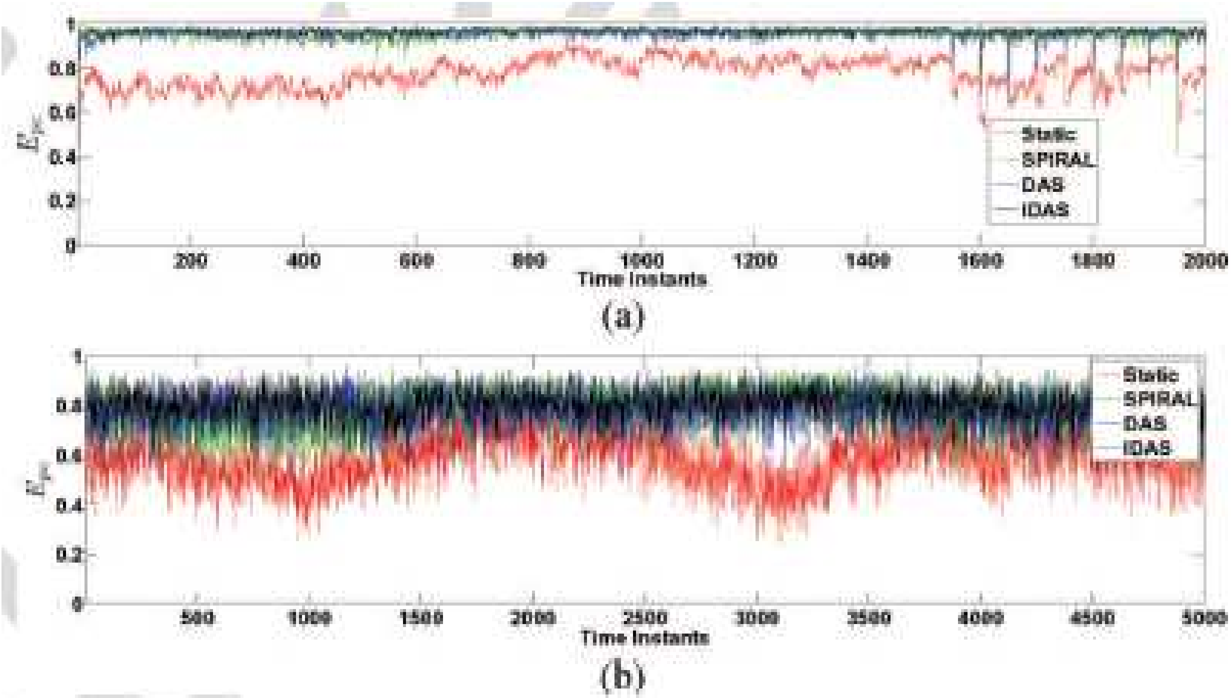


Fig. 5. Comparison of static and dynamic workload assignment methods when reassignment is performed for every frame. (a) PETS data. (b) Real data.

On average, the equalization is improved by 35% for the real data and 24% for the PETS data, but at the cost of

camera switching

Experiments and Results 7

- Evaluation of DAS, IDAS and Spiral methods

TABLE IV
EFFECT OF DYNAMIC CAMERA ASSIGNMENT ON REAL DATA

Method	E_{pc}	Targets Dropped	Camera switchings	Reassignment instants
Static	0.5842	3190	0	0
Spiral	0.7873	139	465069	5000
DAS	0.7835	506	45318	5000
IDAS	0.7967	407	56170	5000

TABLE V
EFFECT OF DYNAMIC CAMERA ASSIGNMENT ON PETS DATA

Method	E_{pc}	Targets Dropped	Camera reassignments	Reassignment instants
Static	0.7736	7436	0	0
Spiral	0.9558	411	183426	2000
DAS	0.9547	740	10180	2000
IDAS	0.9624	408	8682	2000

Experiments and Results 8

- Evaluation of DAS, IDAS and Spiral methods

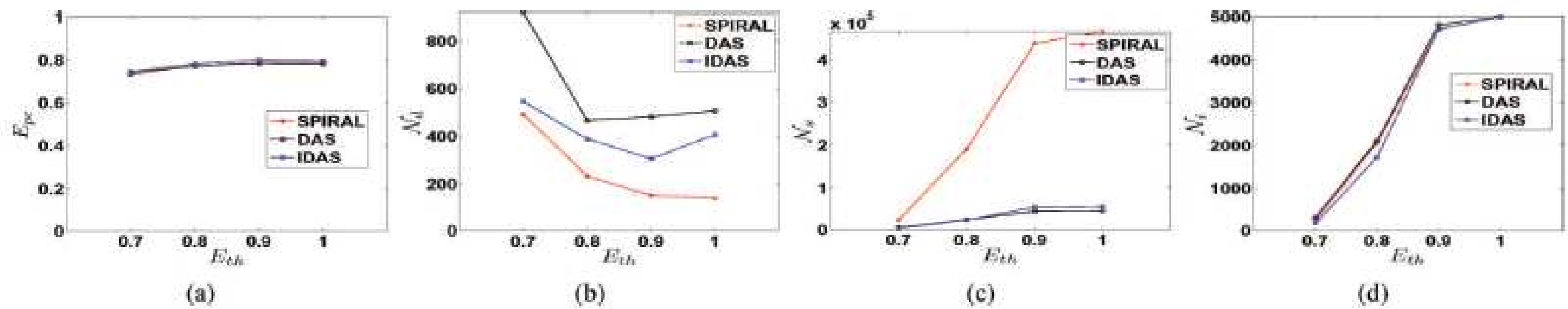


Fig. 6. Performance of dynamic camera assignment methods with E_{pc} as feedback for real data. (a) Equalization. (b) Targets dropped. (c) Cameras switched. (d) Reassignment instants.

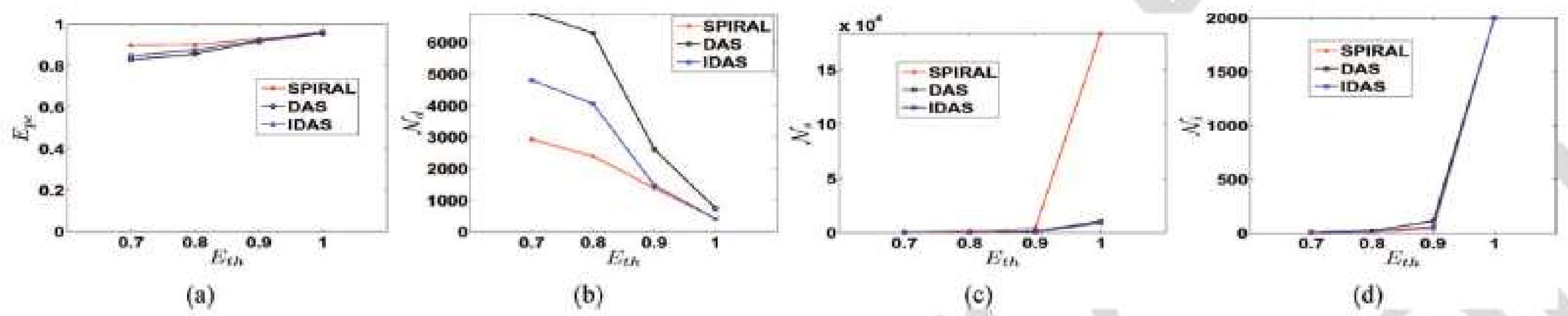


Fig. 7. Performance of dynamic camera assignment methods with E_{pc} as feedback for PETS data. (a) Equalization. (b) Targets dropped. (c) Cameras switched. (d) Reassignment instants.

Experiments and Results 9

- Adaptive Calculation of Transition Probabilities

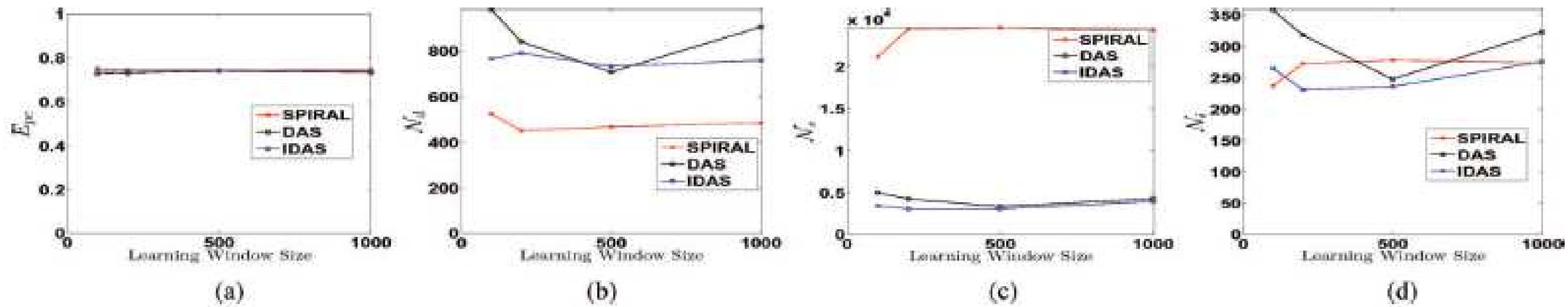


Fig. 8. Performance of adaptive methods for different learning window sizes, $E_{th} = 0.7$ for real data. (a) Equalization. (b) Targets dropped. (c) Cameras switched. (d) Reassignment instants.

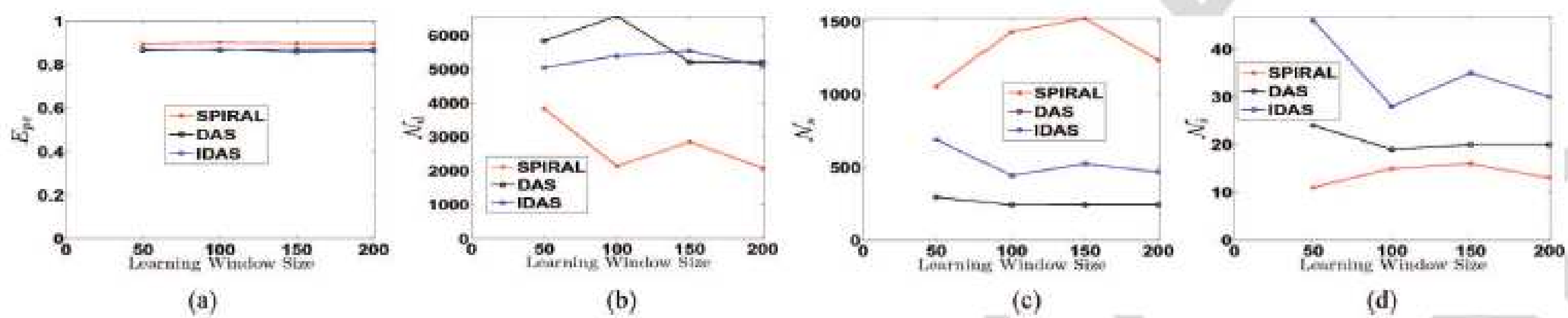


Fig. 9. Performance of adaptive methods for different learning window sizes, $E_{th} = 0.8$ for PETS data. (a) Equalization. (b) Targets dropped. (c) Cameras switched. (d) Reassignment instants.

Experiments and Results 10

- Comparison with random assignment

TABLE VI
EFFECT OF TRANSITION PROBABILITIES FOR REAL DATA

Method	E_{pc}	Targets Dropped	Camera switch-ings	Reassignment in-stants
Random	0.5547	5469	474939	5000
Static	0.5842	3190	0	0
Dynamic	0.7530	525	21166	237

TABLE VII
EFFECT OF TRANSITION PROBABILITIES FOR PETS DATA

Method	E_{pc}	Targets Dropped	Camera switch-ings	Reassignment in-stants
Random	0.7949	13203	189983	2000
Static	0.7736	7436	0	0
Dynamic	0.8644	6569	240	19

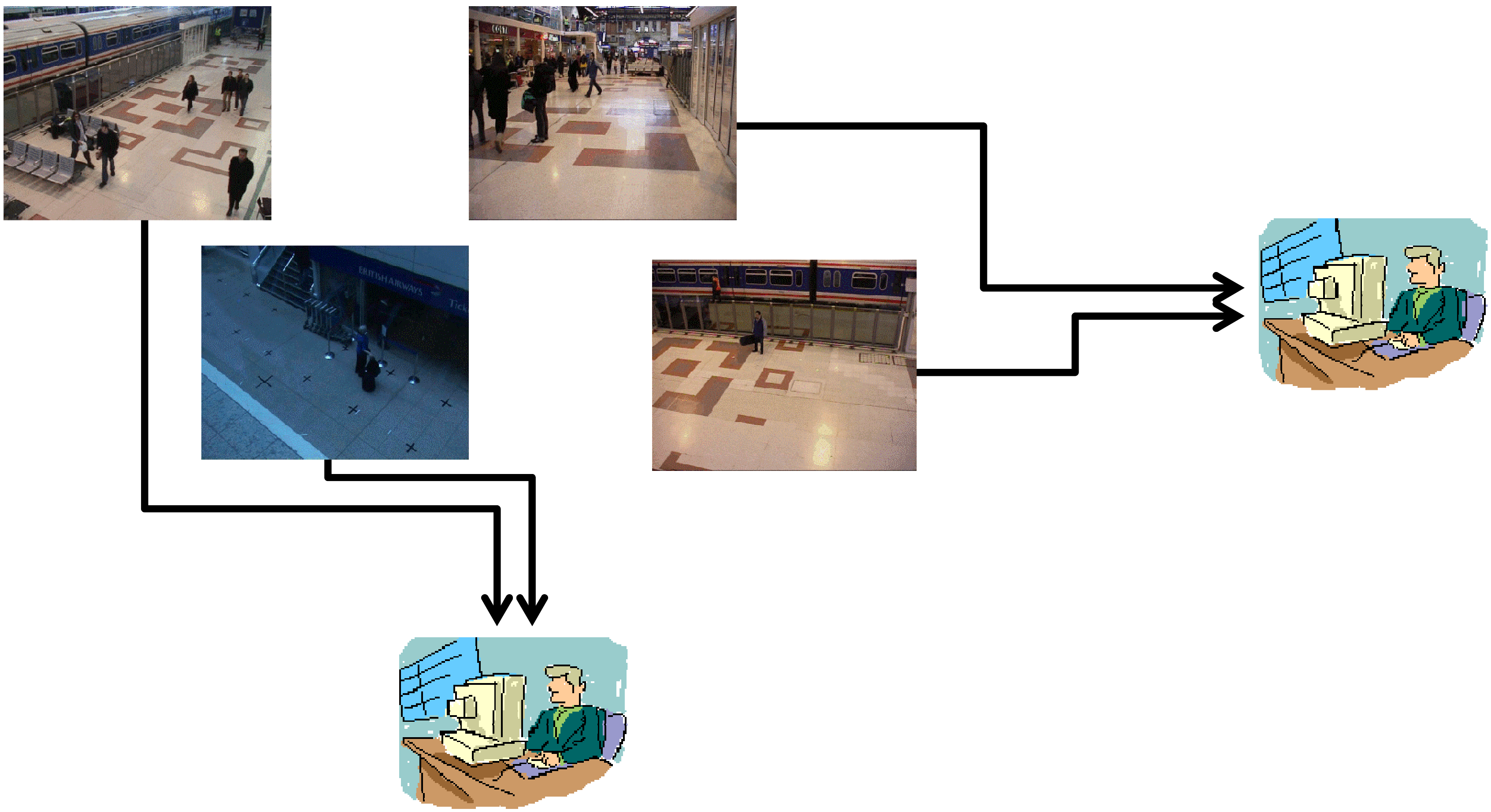
Outline

- Introduction and Motivation
- Related Work
- Workload Model
- Dynamic Load Sharing
- Conclusions

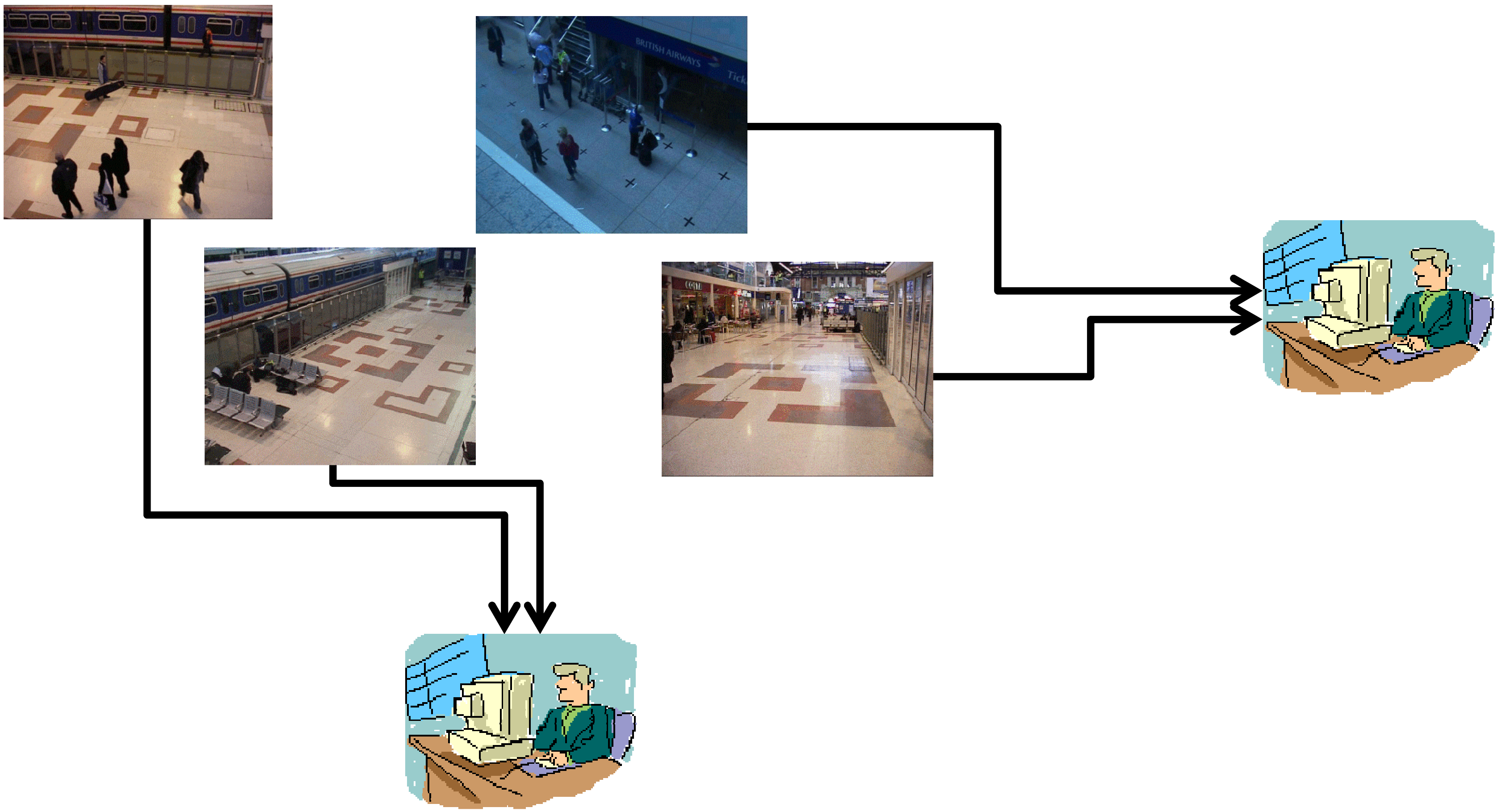
Conclusions

- All three dynamic load sharing methods provide better equalization than static methods.
- Random reassignment method performs better than static method, but poorly in comparison to dynamic load sharing methods.
- Employing a feedback mechanism to perform reassignment reduces the overhead drastically, with marginal compromise in equalization.
- Adaptive calculation of transition probabilities further reduces overhead.
- Spiral method is faster than DAS and IDAS in terms of computational complexity.

Happy Endings



Happy Endings



Publications

- M. Saini, X. Wang, P. K. Atrey and M. S. Kankanhalli. Dynamic workload assignment in video surveillance systems. *IEEE International Conference on Multimedia and Expo (ICME'2011)*, July 2011, Barcelona, Spain.

- M. Saini, X. Wang, P. K. Atrey, and M S. Kankanhalli. Adaptive workload equalization in multi-camera surveillance systems. *IEEE Transactions on Multimedia*, 14(3):555-562 (2012).

What Next?

- This is not the end of the world.

- Evaluate the effectiveness of the dynamic load sharing methods with real surveillance implementations and explore non-preemptive scheduling methods.