Video architectures and networks

Wayne Wolf
Dept. of EE
Princeton University

Outline

- Programmability and design methodologies.
- Lv, Ozer, Wolf: Instruction-level parallelism is not enough.
- Lv, Yang, Wolf: Streaming is a myth.
- Xu, Wolf, Henkel, Chakradhar: Networks-on-chips require top-down and bottom-up analysis.
- Lin, Lv, Ozer, Wolf: Networks of MPSoCs are inevitable.

Why programmable processors?

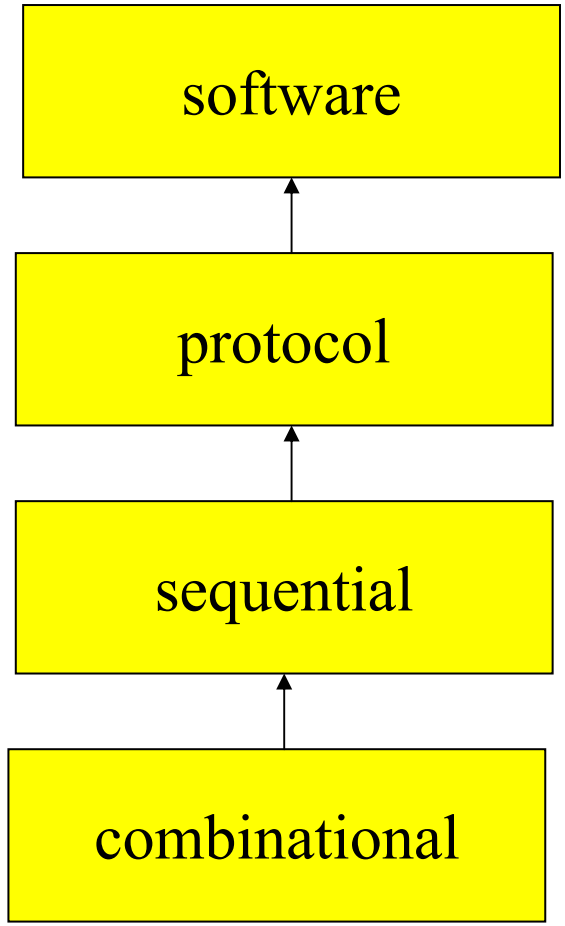
- Too hard to get an all-hardware system to work.
- Algorithms may change:
  - During development.
  - After shipment.
  - Over product generations.

Why heterogeneous architectures?

- Processors:
  - Different microarchitectures for different tasks.
  - More energy efficient.
- Memory:
  - Reduce memory cost.
  - Increase memory efficiency.

Design methodologies for software-intensive systems

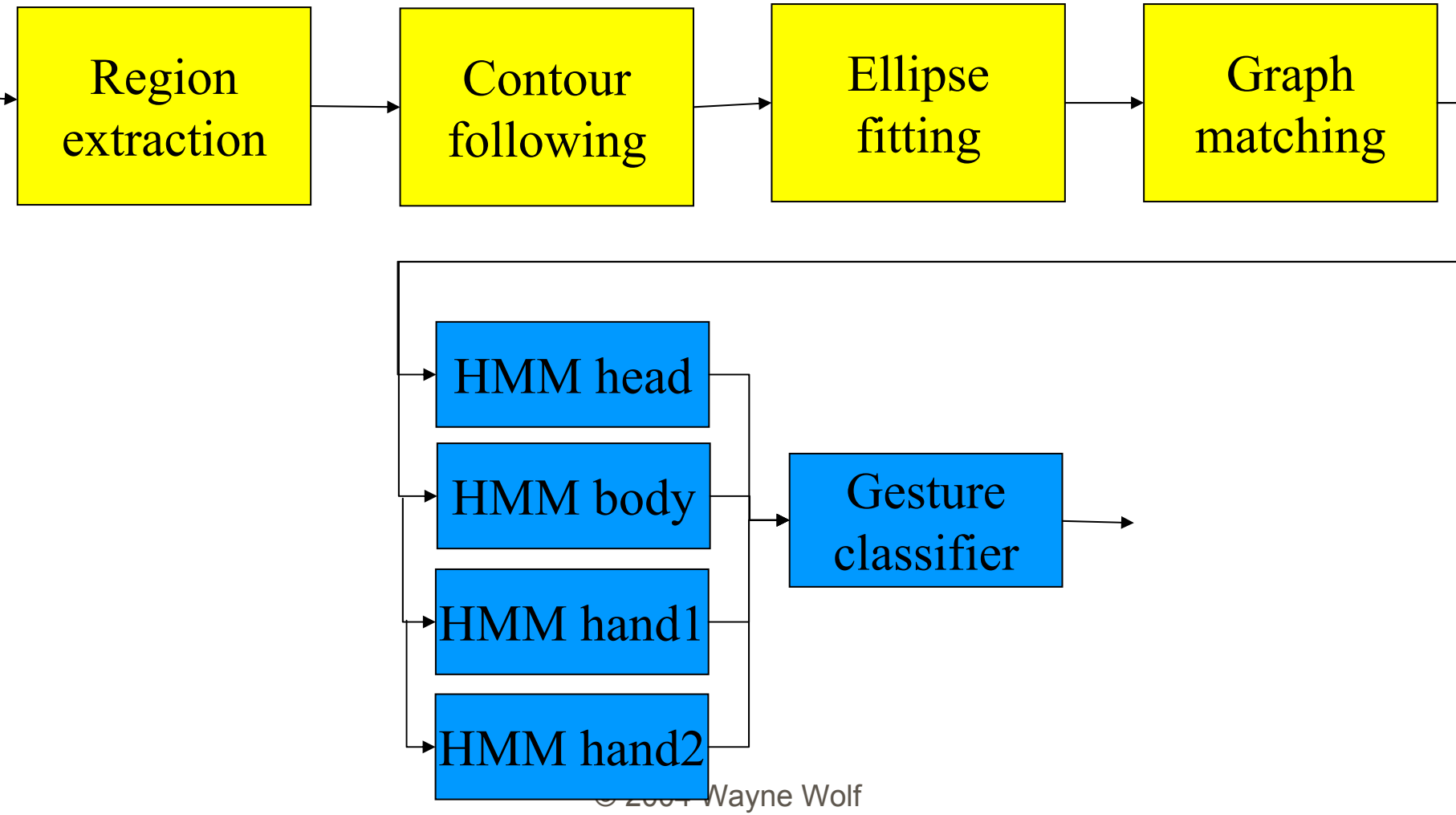
- Traditional hardware design is combinationally rich, sequentially somewhat shallow.
- Components of systems can be verified to satisfy operations.
- Complete applications must verify rich behavior: buffer overflow, etc.



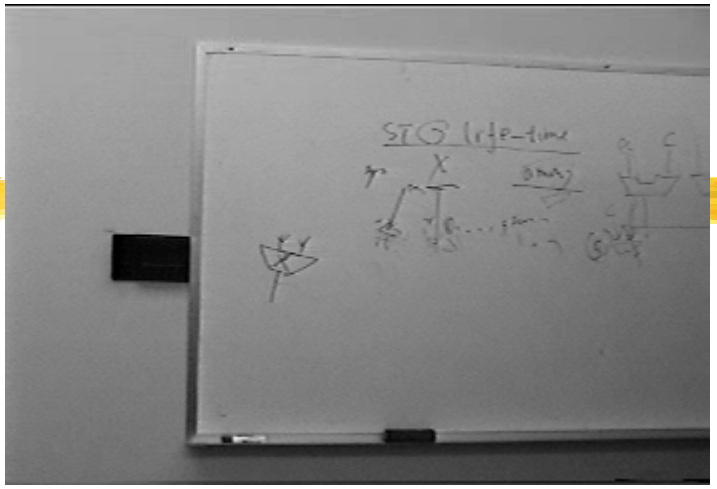
The Princeton Smart Camera Project

- Goal: design SoC networks for real-time distributed vision.
  - The best way to get a good design example is to create our own.
  - Video is a high-performance, low-power, cost-sensitive application.
  - Vision is an important problem.

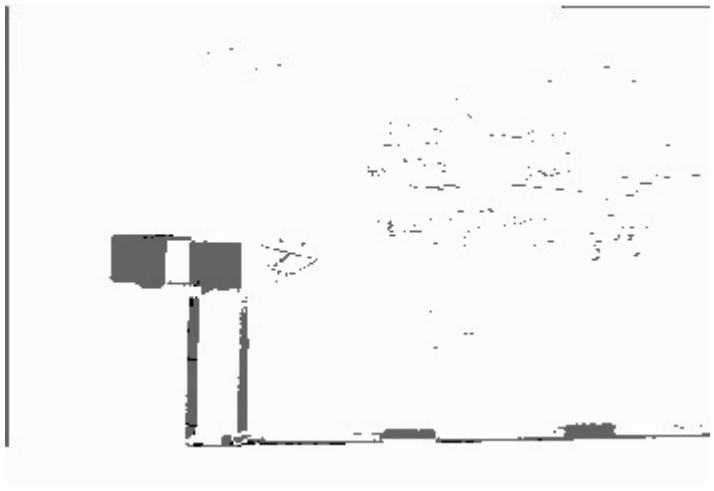
Ozer et al: human activity recognition algorithm



Real-time analysis



Original



Region finding

Ellipse fitting

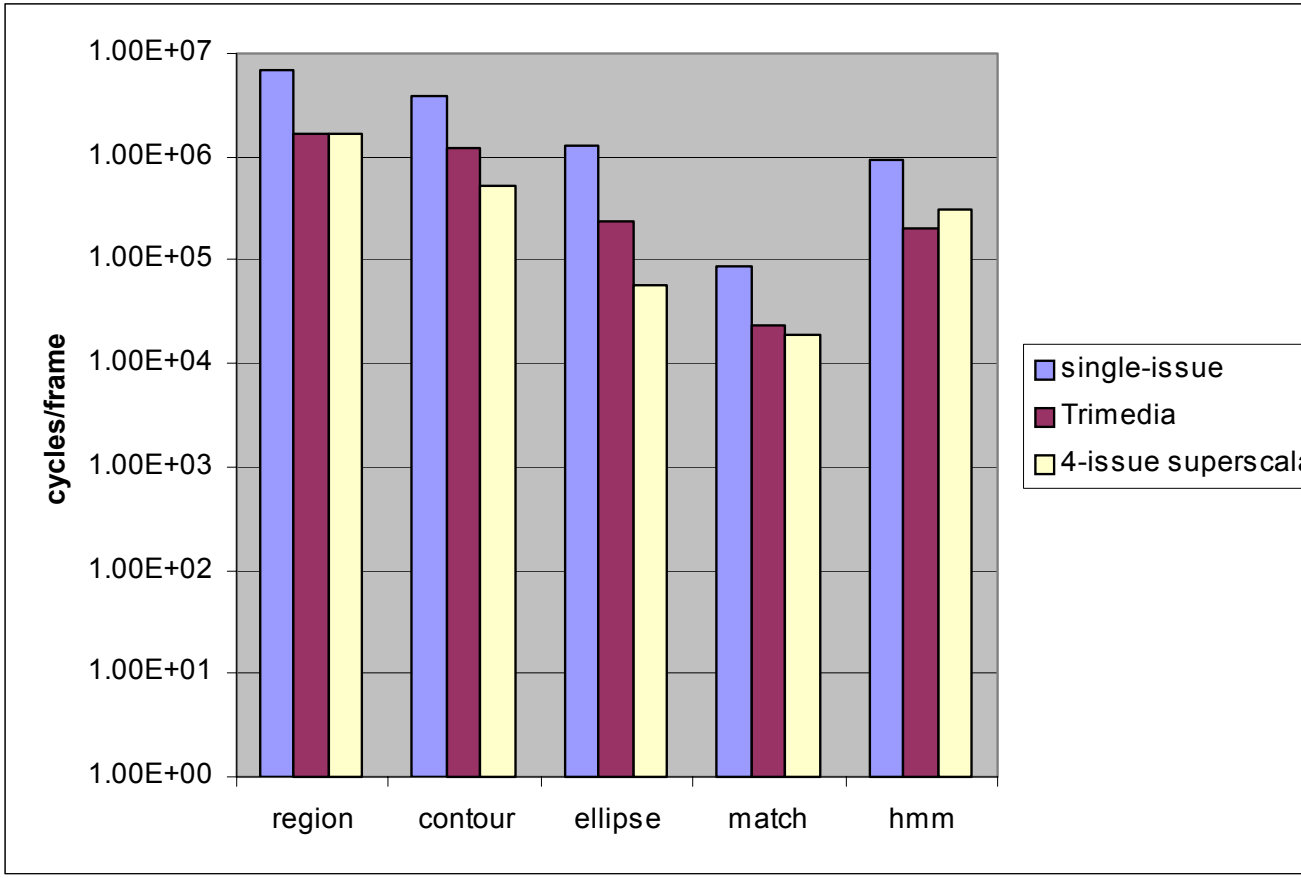
TM-1300 VLIW Processor

- Characteristics
  - Media processing oriented
  - 5 issue VLIW processor
  - Floating point support
  - Sub-word parallelism support
  - If Conversion
  - Additional custom operations

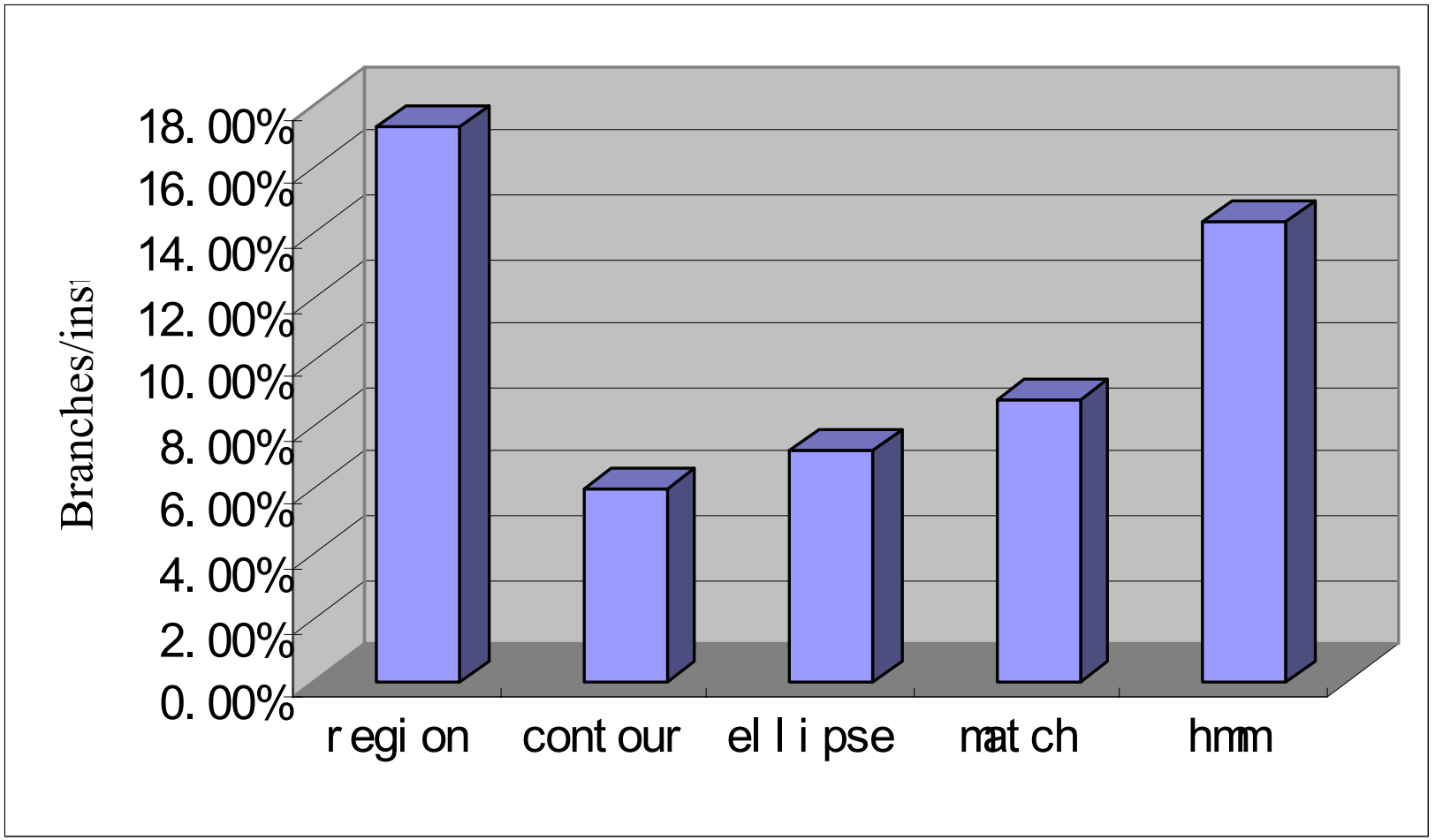
#Functional Unit	Constant	5
	Integer ALU	5
	Load/Store	2
	DSP ALU	2
	DSPMUL	2
	Shifter	2
	Branch	3
	INT/Float MUL	2
	Float ALU	2
	Float Compare	1
	Float sqrt/div	1
#Register		128
Instruction cache		32KB, 8 way
Data cache		16KB, 8 way
#Operation slots/instruction		5

Processor selection

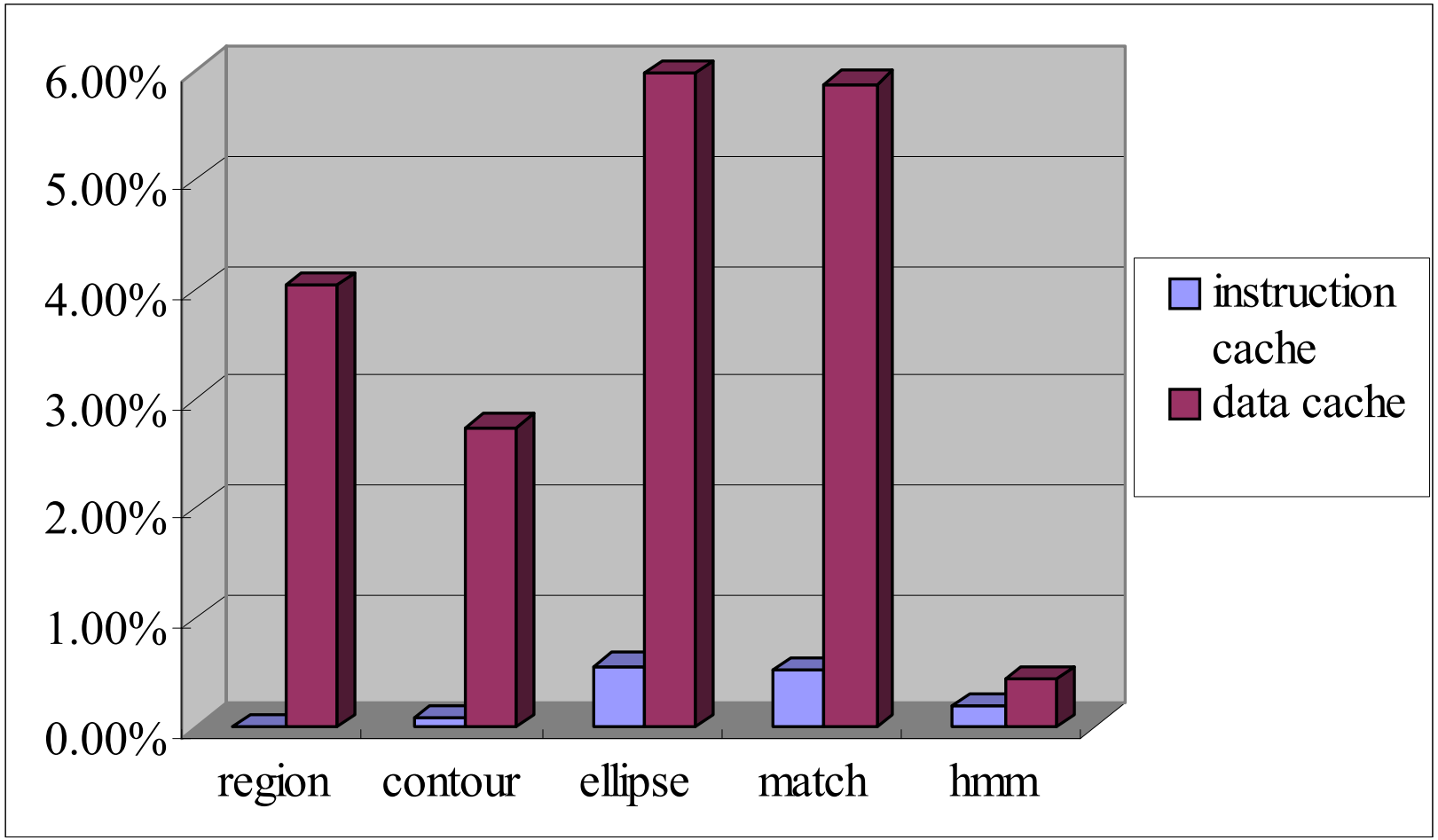
- Cycles per frame for each stage:
  - Single issue.
  - Trimedia.
  - 4-issue superscalar.



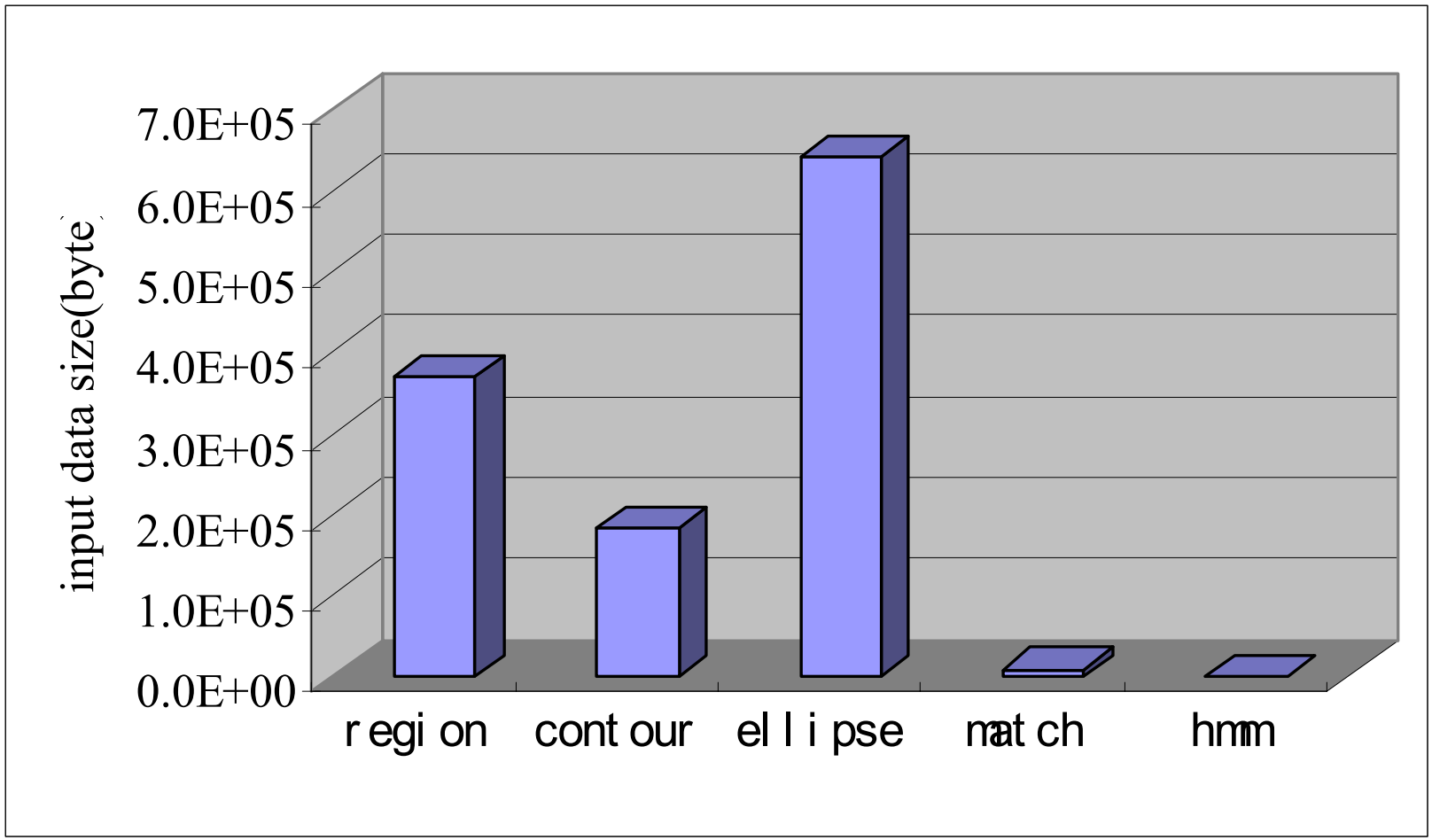
Branches per instruction



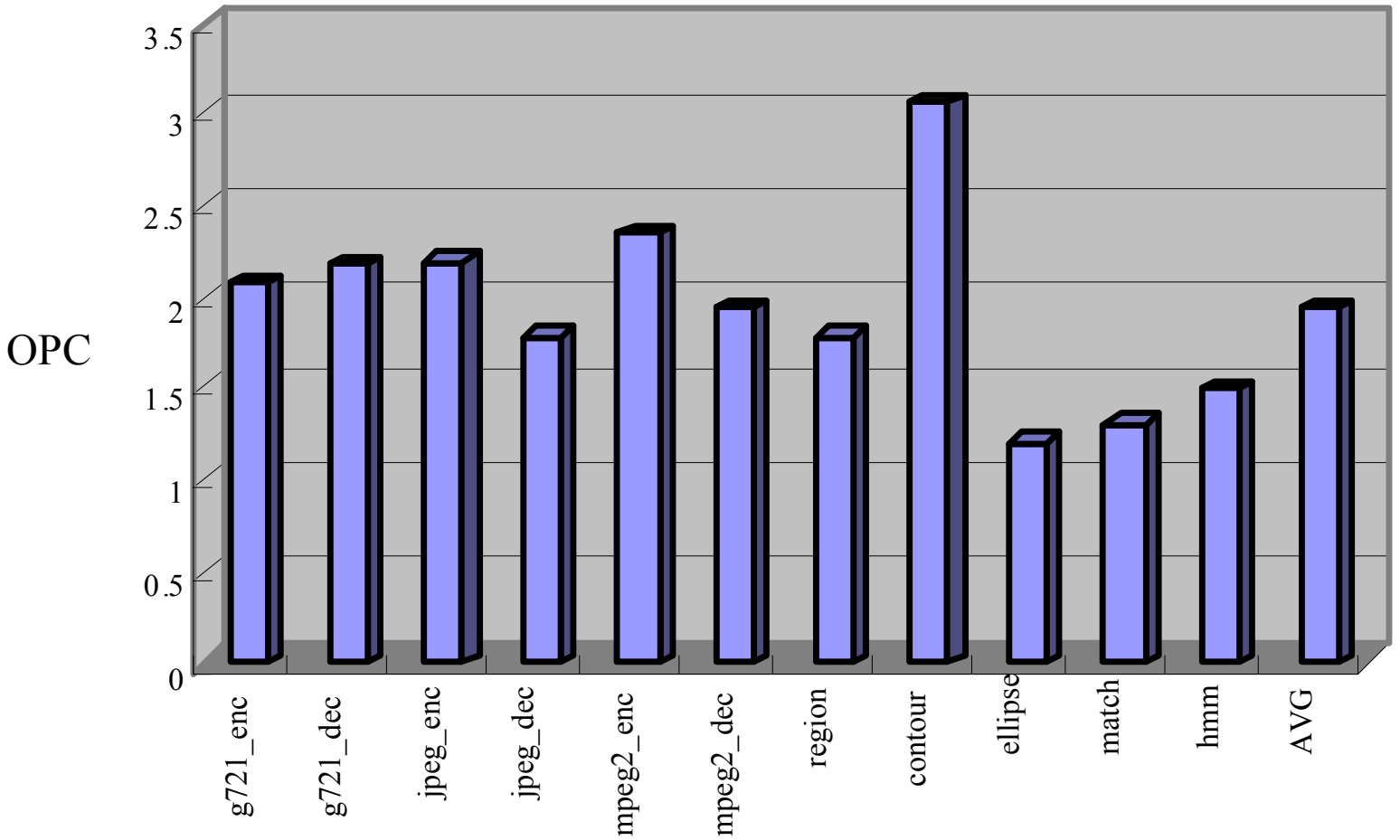
Cache miss rates



Data per stage



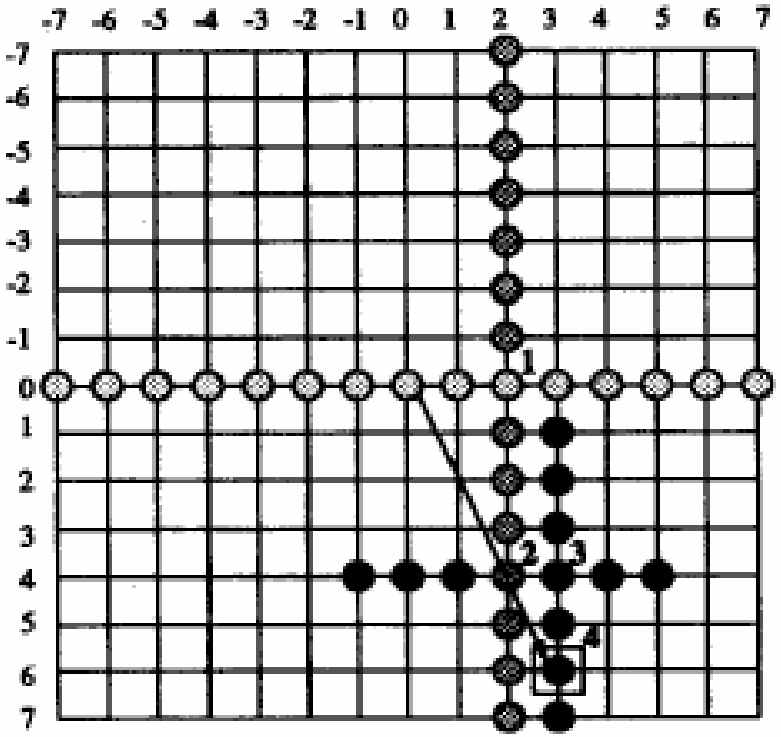
Operation Level Parallelism Results



Multimedia requires control

- High-quality algorithms require control, adaptation.
- Control biases architectures toward programmability, requires careful design for paralellism.

ODFS and PLS algorithms



One-Dimensional Full Search Procedure
Search Range -7 to +7
Motion Vector (3,6) in this case

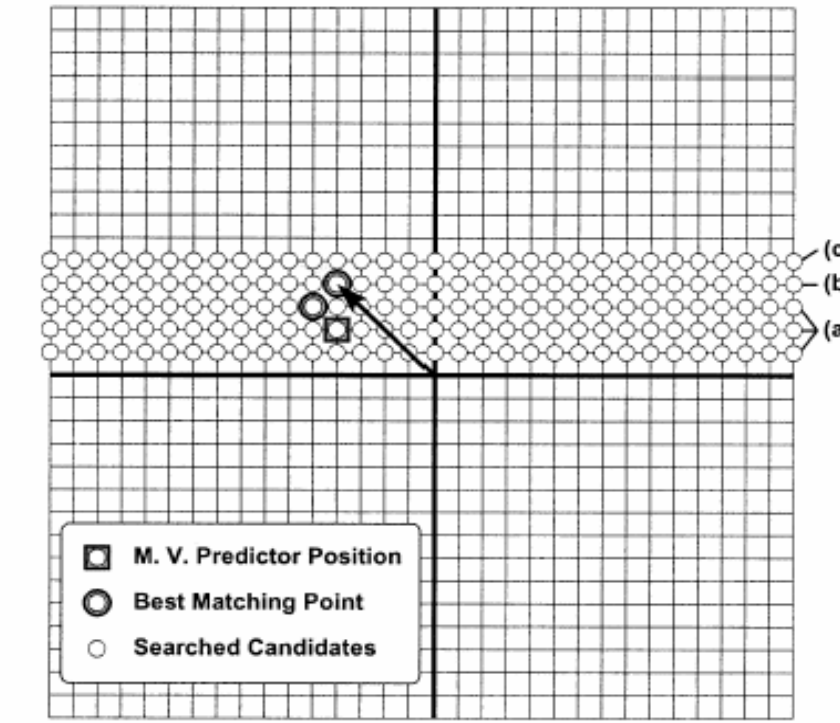
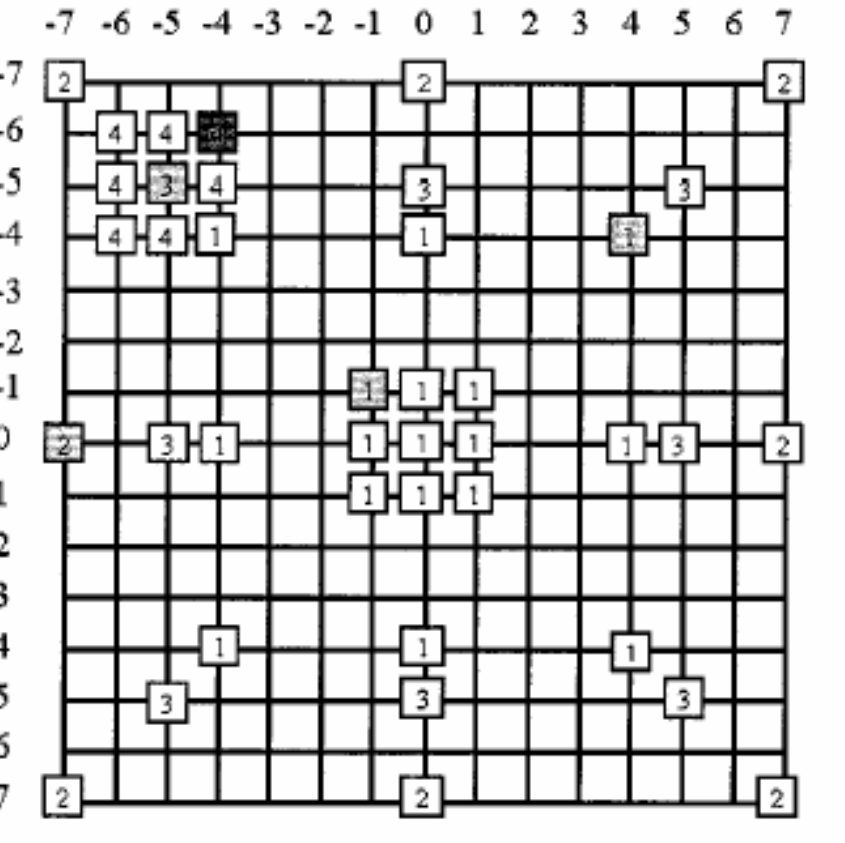
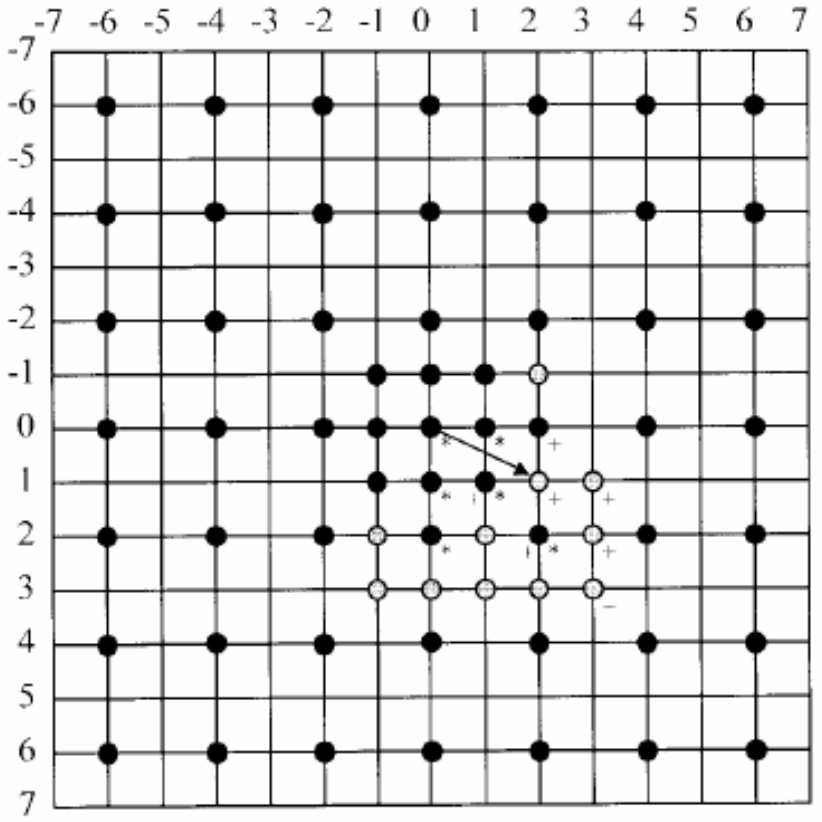


Fig. 4. PLS procedure. The search range is (—16, 15), the motion-ve predictor is (—4, —2), and the best-matching point is (—4, —4) in this exam

CBAS and FE2SS



Center-biased adaptive search



Fast and efficient 2 step search

3SS related algorithms

- E3SS differs from N3SS in that:

1. A small diamond patter is used instead of a square in the central area
2. Unrestricted search step for the small diamond rather than a single movement for the small square.
3. Test sequences: *Coastguard, Football, Salesman, Suzie*

4. FS 3SS 4SS N3SS DS E3SS

(1) Large search window: 31*31, E3SS performs better in terms of MSE and search points than any other non-full search algorithms

(2) Small window: 15*15, E3SS is similar like DS and N3SS

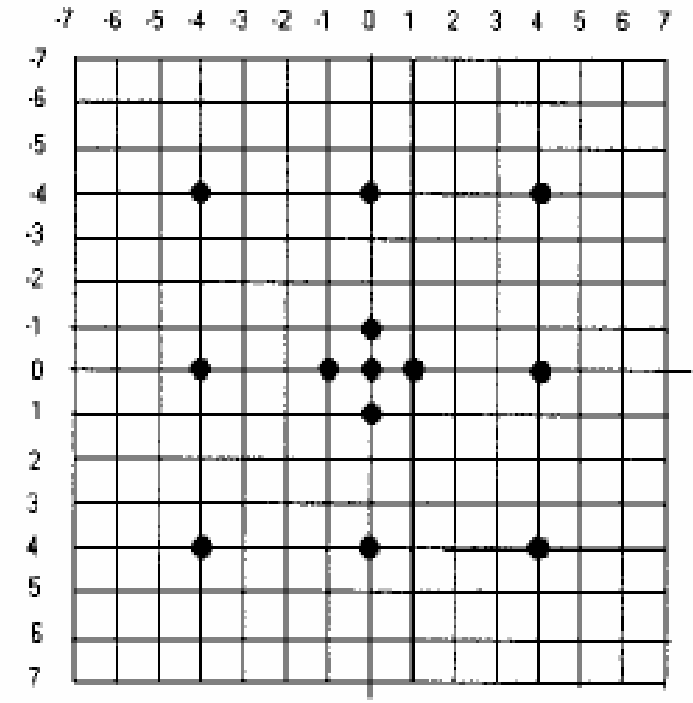
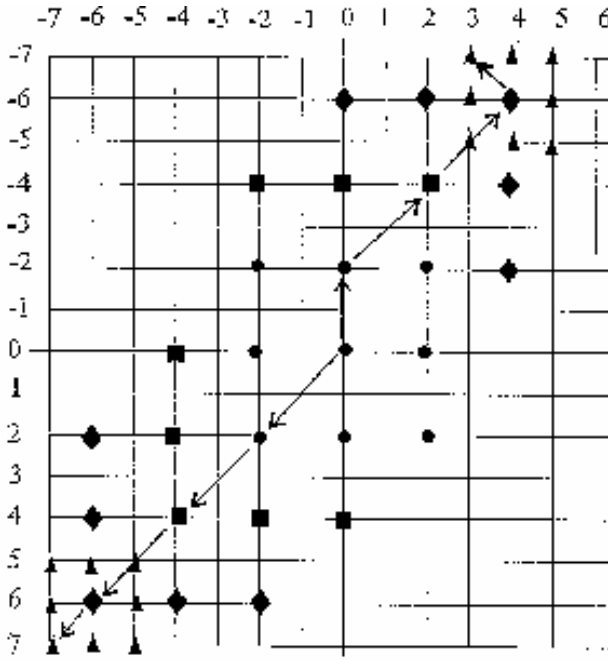


Fig 2. Search pattern used in the first step of E3S

4SS related algorithms

- 4SS

1. Three 5*5 search windows and a final 3*3 window. First step uses 9 points. Second/third step uses three or five points. Final step uses 8 points.
2. Smaller search window 5*5 in the first step of 4SS VS 9*9 in 3SS related algorithms.
3. More regular search pattern than N3SS.
4. 4SS has similar or worse image quality than N3SS but less searching points



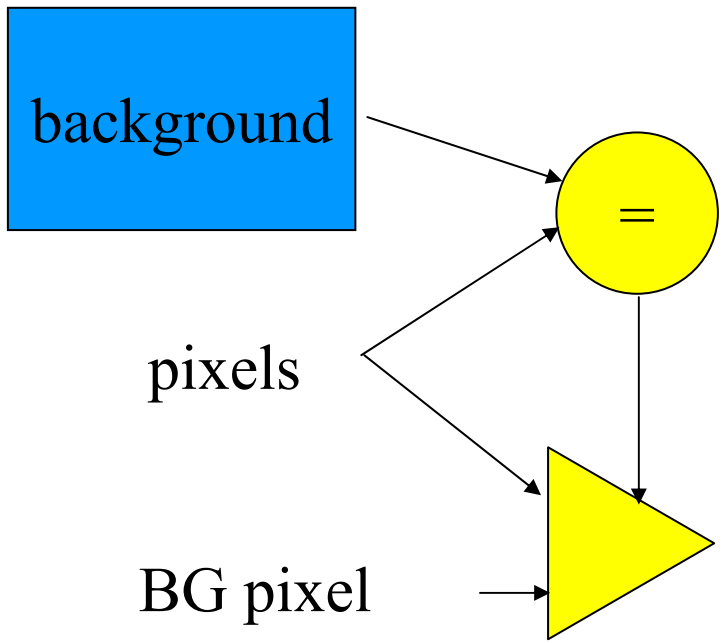
- Other 4SS related algorithms: E4SS

Average Search points: E4SS<4SS<N3SS<3SS

MSE performance is similar like N3SS.

Background elimination

- Simple algorithm subtracts stored background:



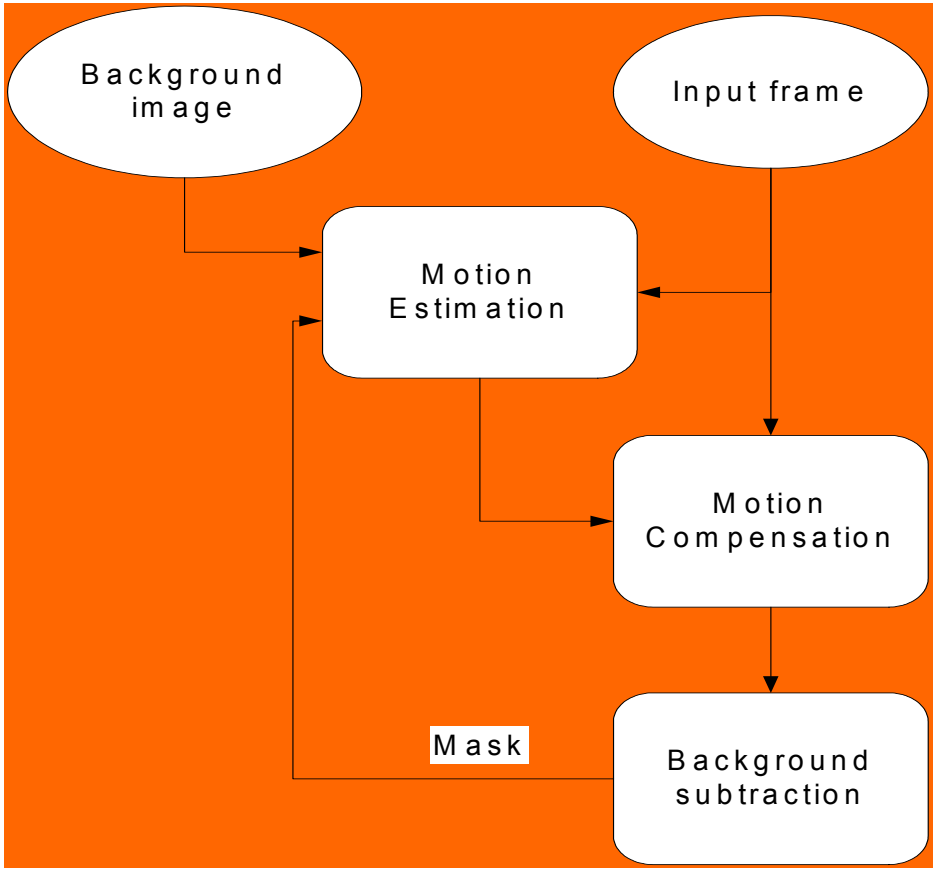
Challenges

- Several types of motion mess up simple background elimination:
  - Large-scale object motion.
  - Small-scale object motion.
  - Camera motion.

Related work in background elimination

- Edge based
  - Yang et al, Comparing the edge information of the current frame with the background edge information to determine the introduced objects.
- Block based
  - Hsu et al, Using statistical likelihood test to determine the blocks with significant changes
- Pixel based
  - Adaptive Gaussian Model (Wren et al)
  - Gaussian Mixture Model Based (Stauffer and Grimson)
  - ...

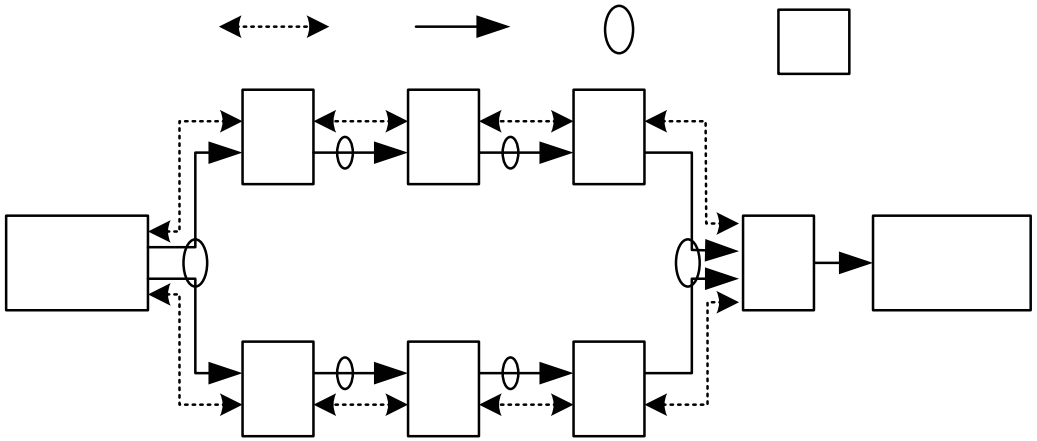
Lv background elimination method



Network-on-chip design

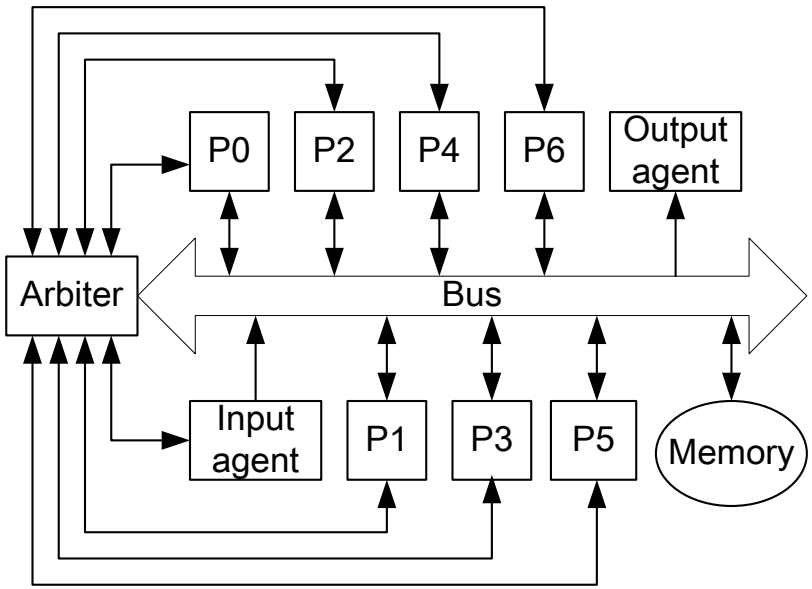
- Physical design: networks simplify delay, clock distribution.
- Architecture/software: packets provide structure for communication.

Test architecture



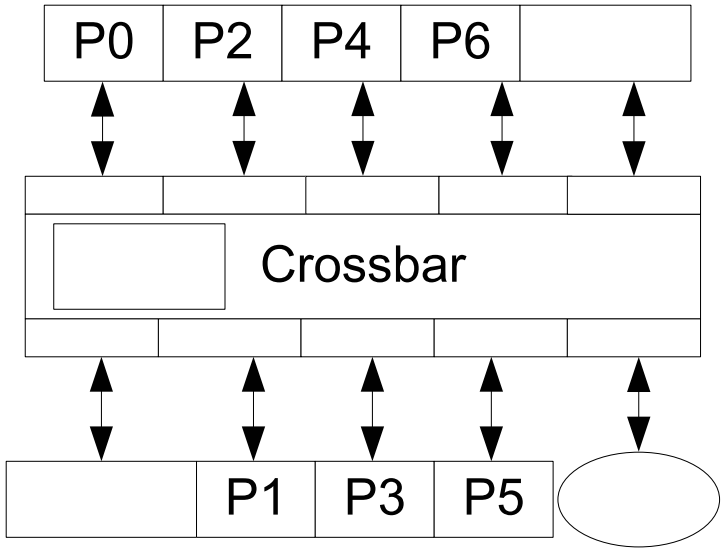
- A good case for NoC
  - Multiple processor IP's --- 7 processors
  - High performance --- 150 frame/sec

Bus-based NoC Model



- Processor-controlled model
  - Processors exchange frame-processing-status through bus
  - Arbiter grants bus based on priorities
- Arbiter-controlled model
  - Frame-processing-status of each processor is sent to arbiter
  - Arbiter grants bus based on more information
- Bus: 32-bit data, 21-bit address, 2-bit control

Switch-based NoC Model (I)



- Same computation nodes
- Different communication architecture
  - Crossbar switch
  - Switch control unit use the same priority as the bus arbiter

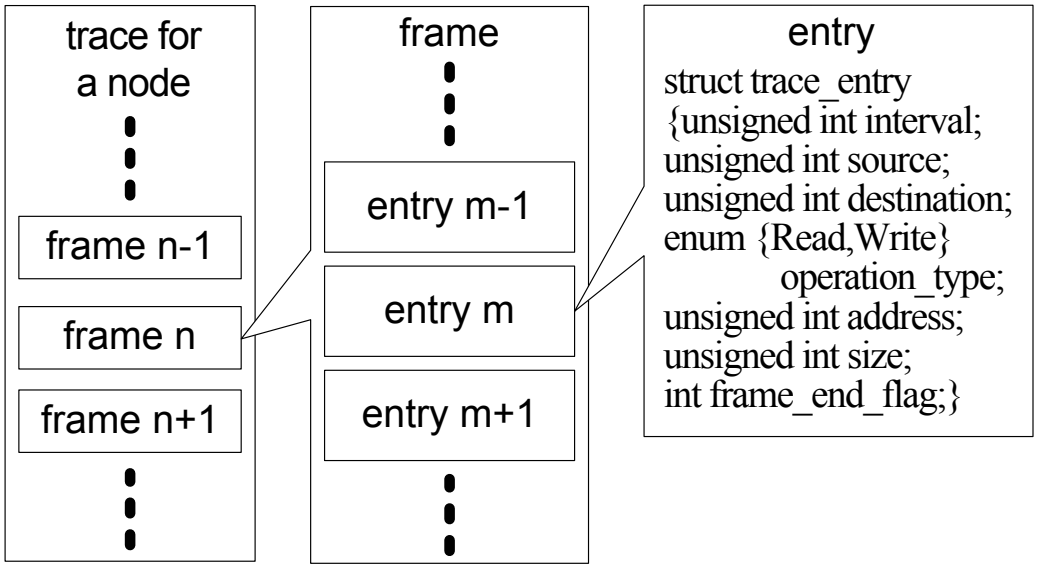
Switch-based NoC Model (II)

- Input buffered NxN crossbar
- N is 10 for single memory, 11 for dual memories
- Port width: 5-bit, 8-bit, 16-bit, 32-bit, and 55-bit
- 4 types of packets: write, read, switch response, and read response

Simulation Environment and Method (I)

- Telecommunication simulator is used
  - OPNET
  - Adaptations on time scale, delay, and synchronization
- The simulation models are cycle-accurate
- Trace-driven simulation
  - Recorded trace from simulation of computational architecture

Simulation Environment and Method (II)



- Trace are recorded in an ideal NoC
- Each node has its own trace

Simulation Results and Analysis (I)

- $3x10^8$ clock cycles are simulated
- Assume the processor and NoC working at the same frequency
  - Optimistic for bus, reasonable for switch
- Required system frequency

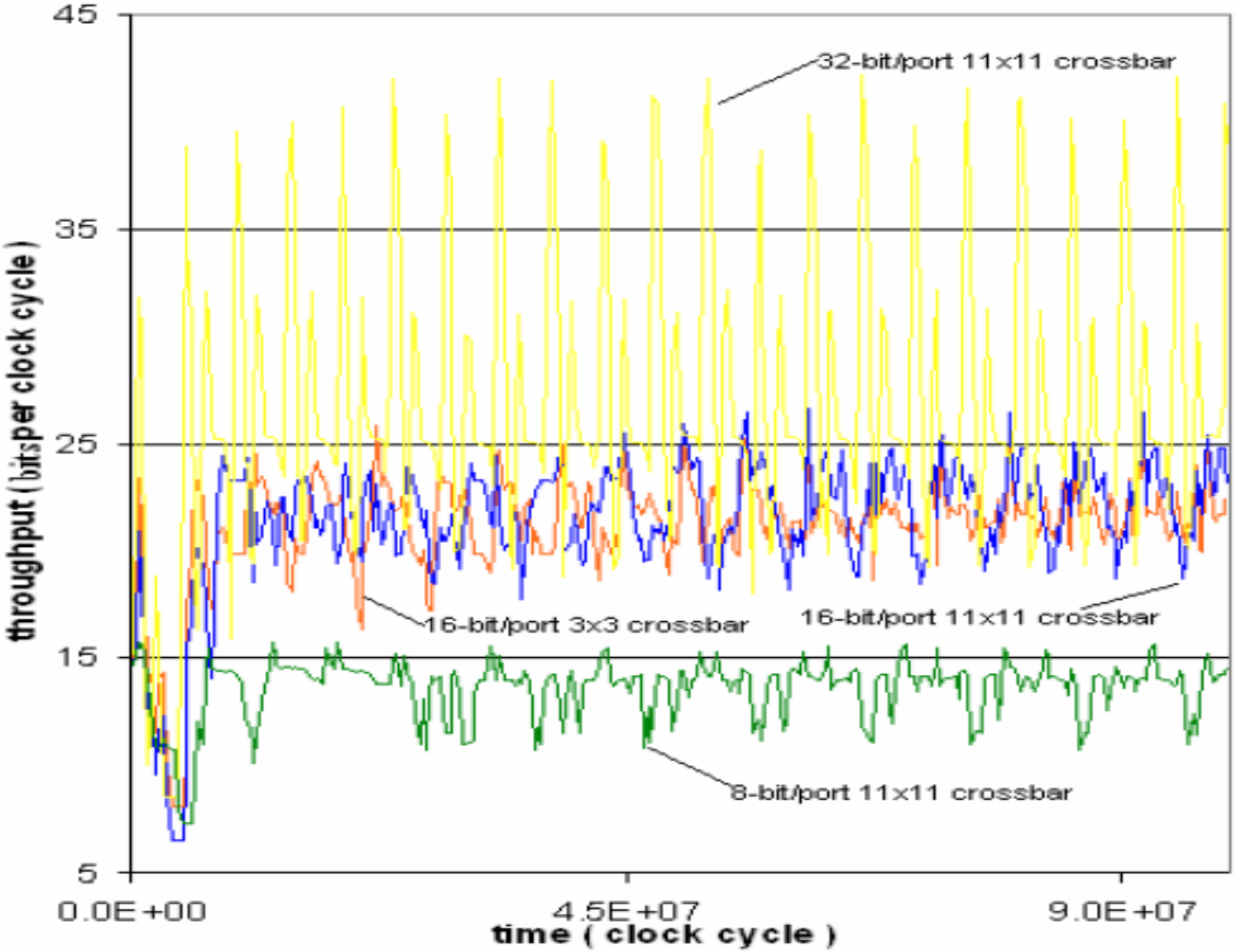
$$System\quad Frequency = \frac{3\times 10^{8}\times 150}{\Pr ocessed\quad Frames}$$

Simulation Results and Analysis (II)

Model Name	Performance (processed frames in $3x10^8$ clock cycles)	Performance improvement (refer to the arbiter-controlled model)	Required frequency (MHz) to reach 150 frame per second
32-bit/port 11x11 crossbar with 2 memories	130	132.1%	346
16-bit/port 11x11 crossbar with 2 memories	106	89.3%	425
16-bit/port 3x3 crossbar with 2 memories	104	85.7%	433
55-bit/port 10x10 crossbar	102	82.1%	441
32-bit/port 10x10 crossbar	91	62.5%	495
8-bit/port 11x11 crossbar with 2 memories	67	19.6%	672
16-bit/port 10x10 crossbar	57	1.8%	789
16-bit/port 3x3 crossbar	56	0	804
Arbiter-controlled	56	reference	804
Processor-controlled	49	-12.5%	918
8-bit/port 10x10 crossbar	36	-35.7%	1250
5-bit/port 10x10 crossbar	23	-58.9%	1957

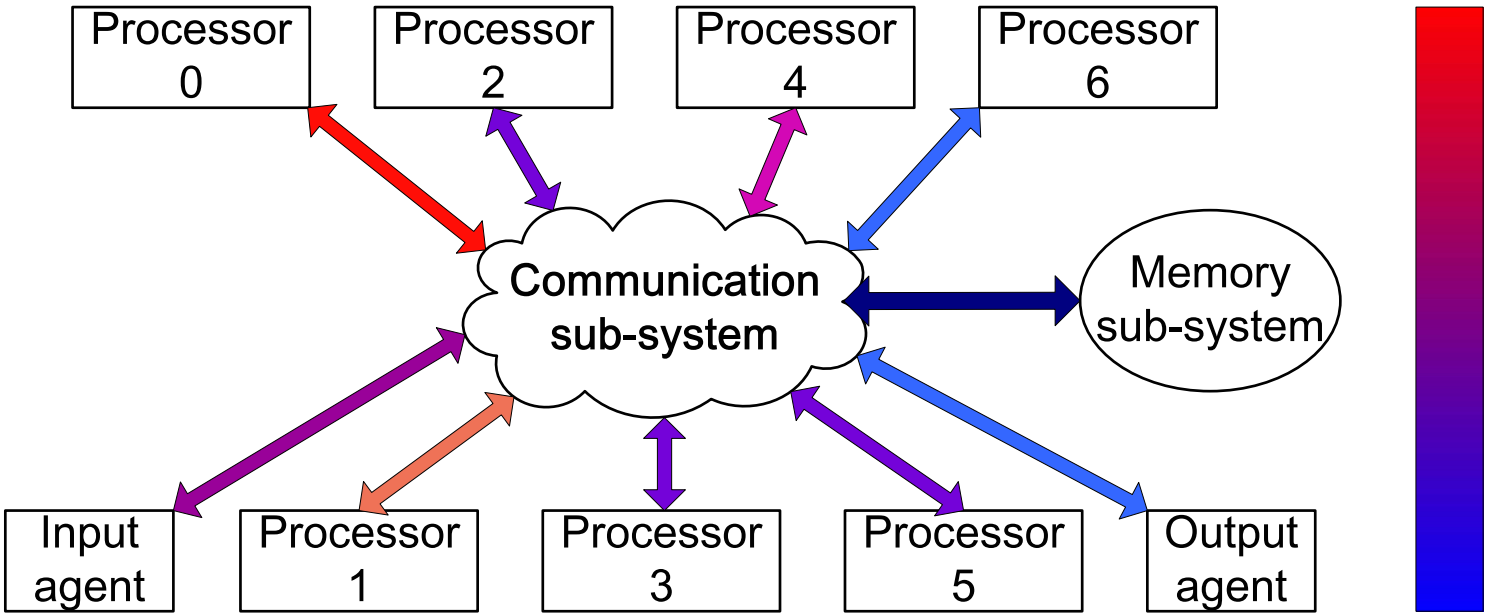
Simulation Results and Analysis (III)

Model Name	Maximum throughput (bit per clock cycle	Average throughput (bit per clock cycle)	Network utilization
32-bit/port 11x11 crossbar with 2 memories	352	27.0	7.7%
16-bit/port 11x11 crossbar with 2 memories	176	22.0	12.5%
16-bit/port 3x3 crossbar with 2 memories	48	21.2	44.2%
55-bit/port 10x10 crossbar	550	21.0	3.8%
32-bit/port 10x10 crossbar	320	18.9	5.9%
8-bit/port 11x11 crossbar with 2 memories	88	13.6	15.5%
16-bit/port 10x10 crossbar	160	11.5	7.2%
16-bit/port 3x3 crossbar	48	11.4	23.8%
Arbiter-controlled	54	10.1	18.7%
Processor-controlled	54	10.2	18.9%
8-bit/port 10x10 crossbar	80	7.3	9.1%
5-bit/port 10x10 crossbar	50	4.7	9.4%



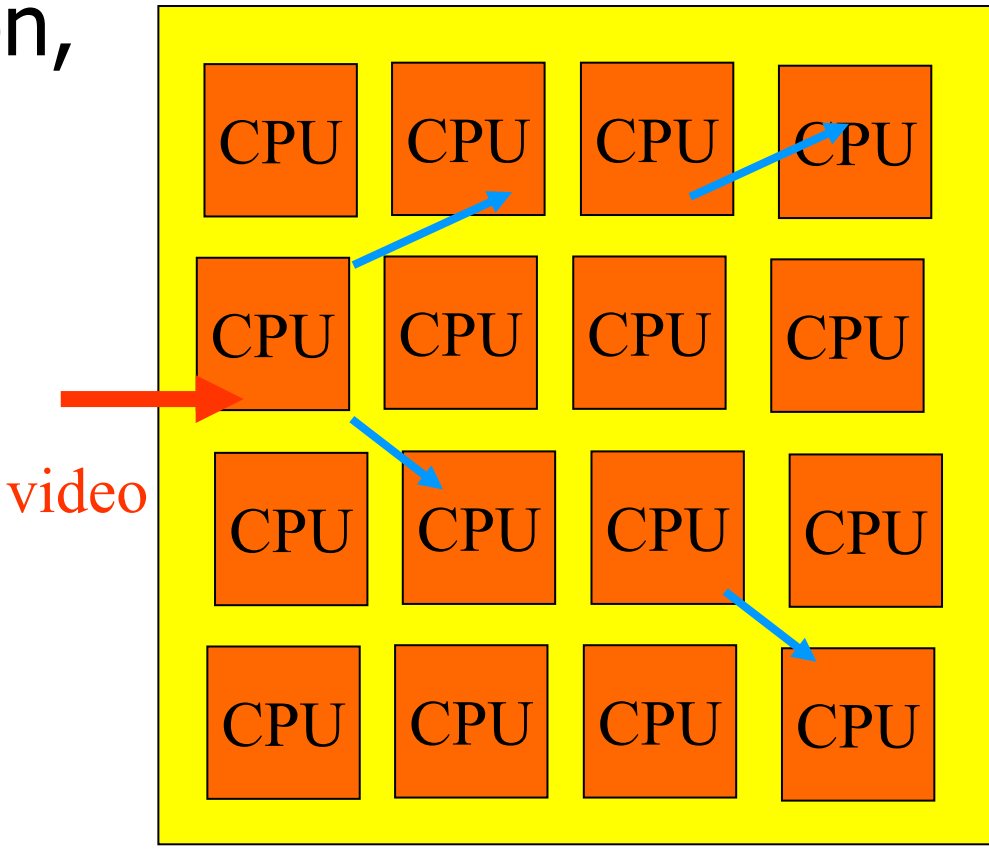
Network throughput using two shared memories

Hot spots



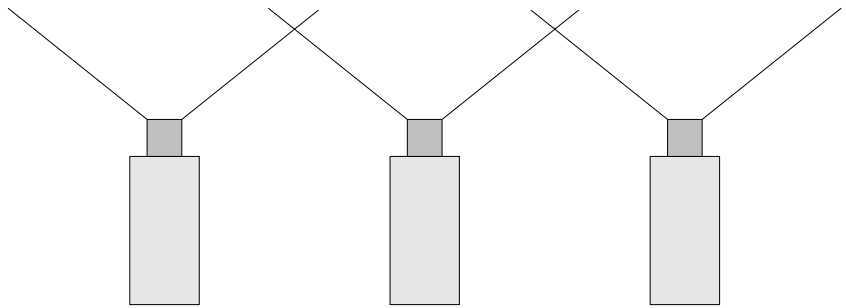
Grand unified application and SoCs

- Gesture recognition, face recognition, facial expression analysis, speech recognition, non-speech sound recognition, Etc.
- Algorithms + architecture.

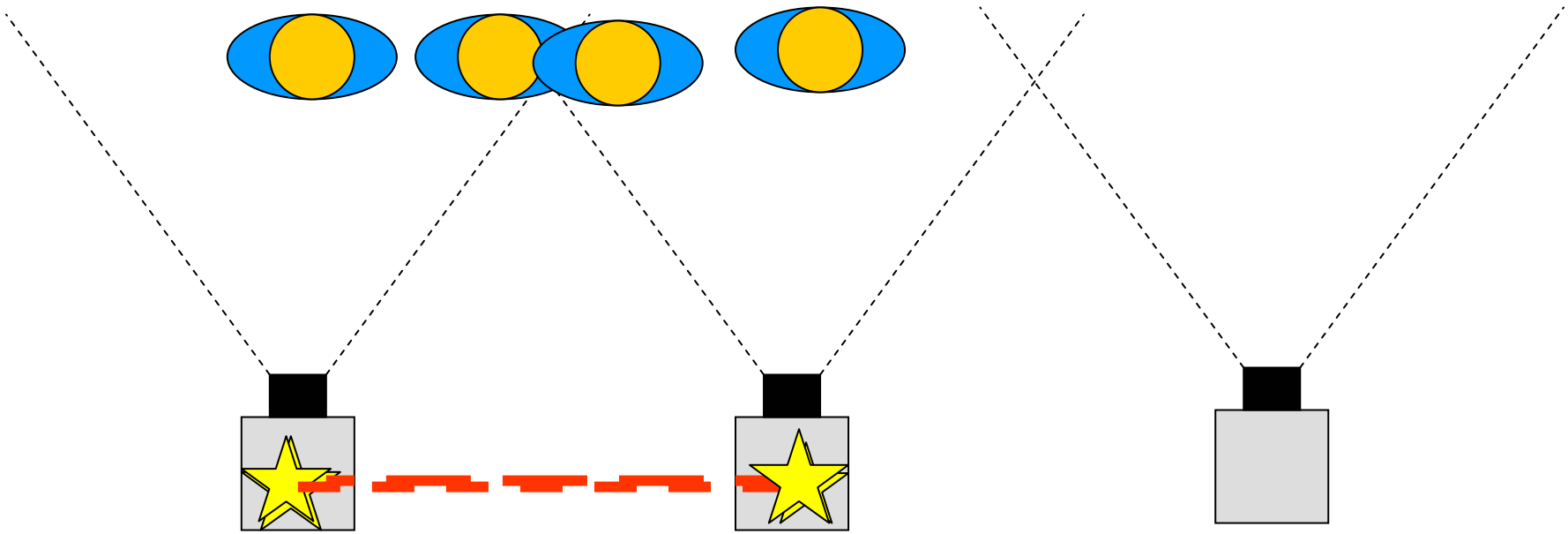


Peer-to-Peer Camera Algorithms

- Distributed computing
- Data exchange
- Migration Methodology
  - Directly inheritable
  - Trade-off between performance and communication cost
  - Example:

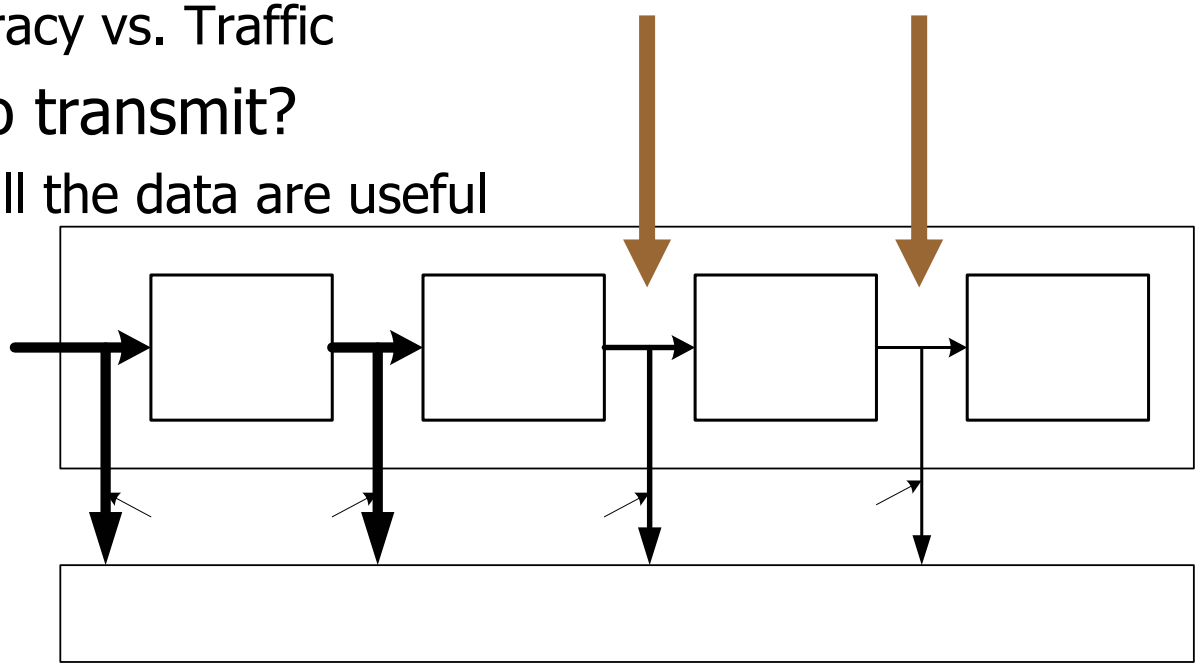


Peer-to-peer video analysis



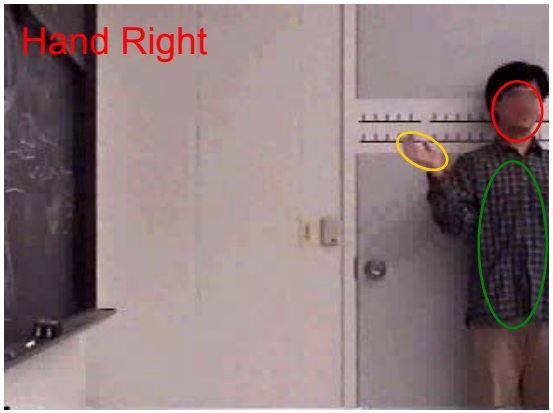
Peer-to-Peer Camera Algorithms (cont.)

- Algorithm partitioning
  - When to transmit?
    - Accuracy vs. Traffic
  - What to transmit?
    - Not all the data are useful

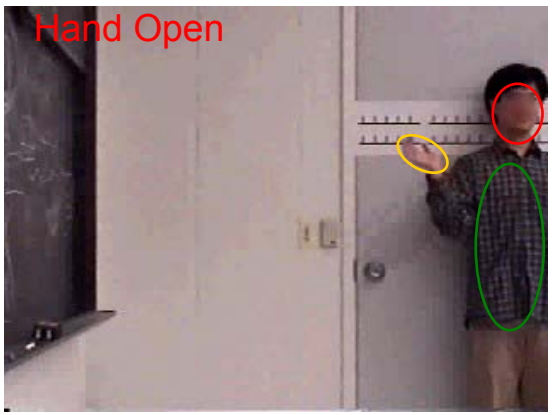


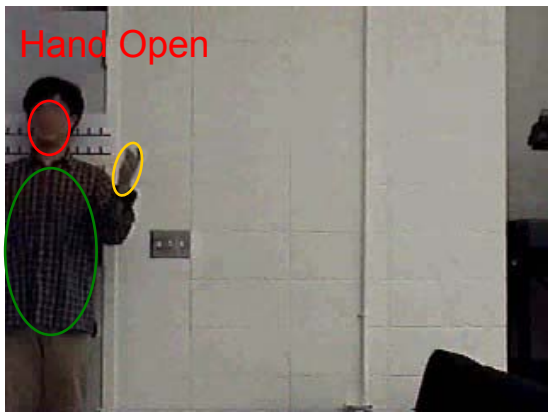
Results

- Single camera



- Multiple cameras





Distributed smart camera node

- Internal network talks to IP (or specialized protocol):