Improving Fault-Tolerance in Intelligent Video Surveillance by Monitoring, Diagnosis and Dynamic Reconfiguration

Andreas Doblander[1,2], Arnold Maier[1,2], Bernhard Rinner[1], and Helmut Schwabach[2]

ITI TUG

[1] Institute for Technical Informatics Graz University of Technology

AUSTRIA

{doblander, maier, rinner}@iti.tugraz.at

seibersdorf research
An enterprise of the Austrian Research Centers.

[2] Video & Safety Technology

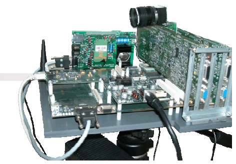
ARC Seibersdorf Research

AUSTRIA

helmut.schwabach@arcs.ac.at

May 20, 2oo5 Hamburg, Germany

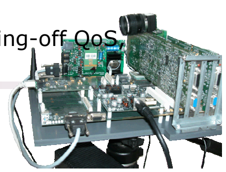
Overview



Motivation

- ❑ Intelligent Video Surveillance (IVS)
  - → Distributed system of collaborating intelligent cameras
  - → Intelligent cameras are implemented as autonomous embedded multi-processor systems
- ❑ Optimizations in IVS Systems
  - → Quality of service (QoS)
  - → Energy consumption

  } *WISES 2oo4: QoS-Triggered Dynamic Power Management in Embedded IVS*
  - → Service availability (by improving fault-tolerance)
  - → Three conflicting design objectives
- ❑ Approach
  - → System monitoring and fault diagnosis
  - → Dynamic system reconfiguration
    - No redundant hardware available
    - Graceful degradation
  - → Computing of a new (degraded) system configuration by trading-off QoS, energy and service availability

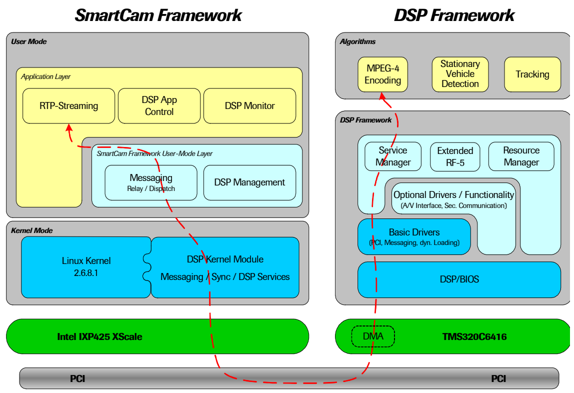


Related Optimization Approaches

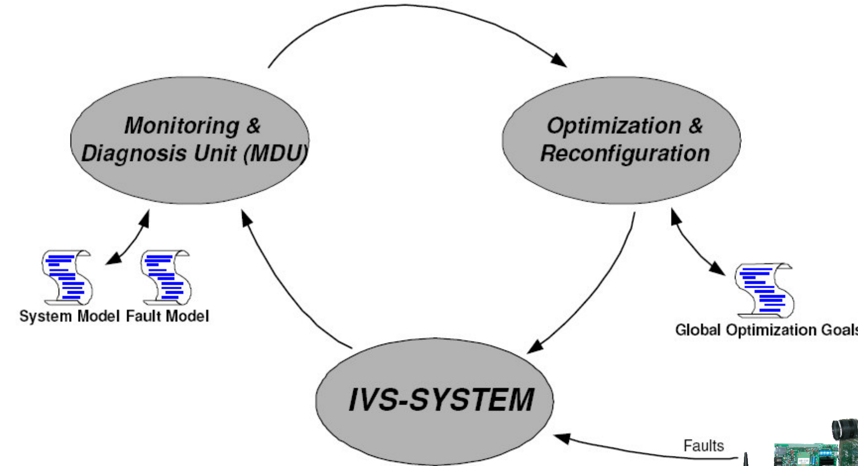
- ❑ Trading Energy and QoS in Video Surveillance
  - → Chiasserini et al.
  - → Focus
    - Image quality in wireless video surveillance networks
    - Sophisticated image compression techniques
- ❑ Fault-Tolerant Energy Aware Embedded Real-Time Systems
  - → Chakrabarty, Krishna et al.
  - → Focus
    - Adaptive Checkpointing Algorithms for Energy Minimization
    - Checkpointing schedule for effective dynamic voltage scaling in System- and application-level fault-tolerance techniques
- ❑ Multi-Criterion Optimization
  - → Thiele, Fonseca, Zitzler et al.
  - → Focus
    - Formulation of optimization problem with conflicting design objectives
    - Various applications

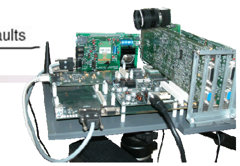
A Generic Software Framework for IVS

- M.Bramberger et al.
- Two-Fold Software Framework
  - → DSP-framework
  - → SmartCam-framework on network processor
- Provided Services
  - → Data management
  - → Performance monitoring
  - → Synchronization
  - → Message-based communication DSP⇔DSP /\ DSP⇔XScale
- Dynamic Loading of Modules (Binaries) on DSPs
  - → Allows dynamic reconfiguration



Basic System Model



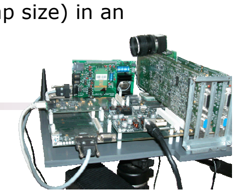


Monitoring and Diagnosis

- Detection of Abnormal Behavior
  - → Provide information about which resources are not available after a fault has occurred
- System- and Node-View
  - → System-view considers the overall distributed application
    - All nodes (hardware and system software)
    - Includes all algorithms running on these nodes
  - → Node-view considers only a single node
- Monitoring and Diagnosis Process Comprises System- and Node-Level Part
  - → M&D-Unit at each node
  - → "Alive-Messages" (CPUs, CMOS Sensor)
  - → "Fusion" of diagnosis of neighbored nodes

Fault Model

- System-Level Faults
  - → Blackout of a node
    - No Alive-Messages
  - → Value-faults of instances of
    - Stationary vehicle detection algorithms or traffic statistics algorithm
    - Detection of erroneous algorithm behavior by employment of majority decisions (comparing results of its two nearest neighboring nodes)
- Node-Level Faults
  - → Crash-faults of
    - DSPs: Software watchdog
    - Algorithms: no (re-)active algorithm communication
  - → Memory leaks
    - Substantial increase in dynamic memory consumption (heap size) in an unchanged system state



Optimization of System Configuration I

- Three Different Optimization Criteria
  - → Maximum QoS, minimum energy consumption, and maximum availability
- Determination of New System-Configuration
  - → Mapping of intended functionality onto remaining system resources
  - → Optimizer determines a set of feasible node configurations
    - Comparing the algorithms resource requirements and the available resources on the node

$C_{Node} = \{C_{a_i}, \ldots, C_{a_N}\}$ *where* $C_{a_i} = [q_j, d_k]$ *with*

$q_j \in \{Q_1, \ldots, Q_3\}$ *is the QoS*

$D = \{d_1, \ldots, d_M\}$ *are the DSPs within the node and*

$A = \{a_1, \ldots, a_N\}$ *are the available algorithms in the system*

Algorithms assigned to a DSP d_k *:* A_{d_k} *where* $A_{d_k} \subseteq A$

Optimization of System Configuration II

- Objective Functions
  - → Energy consumed by a single node:

$$E_{Node} = \sum_{a_i \in A_{d_k}} E_{a_i}$$ *where* E_{a_i} *is the energy consumed by an algorithm*

  - → QoS-Level of a single node:

$$Q_{Node} = \sum_{a_i \in A_{d_k}} V_{a_i}, where\ V_{a_i} = \begin{cases} 3, & q_j = Q_1 \\ 2, & q_j = Q_2 \\ 1, & q_j = Q_3 \end{cases}$$

  - → Resource utilization of a single node

$$R_{Node} = \sum_{a_i \in A_{d_k}} R_{a_i}$$

→ Optimizer has to solve the problem:

$$\begin{aligned} &\min(E_{Node}) \\ &\max(Q_{Node}) \\ &\min(R_{Node}) \\ &subject\ to\ C_{a_i} \in S \end{aligned}$$

where S *is the set of all possible configurations whose resource requirements can be satisfied on the node*

Example

Energy requirements:

	d_1			d_2		
	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3
a_1	3	2	1	6	4	2
a_2	6	4	2	9	6	3

Resource requirements[1]:

	d_1			d_2		
	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3
a_1	30	20	10	60	40	20
a_2	60	40	20	90	60	30

[1] *max. 100 per DS*

- Setup
  - → Two DSPs (d_1, d_2) with a total of four algorithms $(a_1, \ldots, a_4)$
  - → Observed node: Only a_1 and a_2 are executed
  - → System policy forces a_1 to run in Q_1, a_2 may either run in Q_1 or Q_2 .
- Feasible Set of Configurations

$$C = \{C_1, \ldots, C_8\} = \{\{c_{a_1,1}, c_{a_2,1}\}, \{c_{a_1,1}, c_{a_2,2}\}, \{c_{a_1,1}, c_{a_2,3}\}, \{c_{a_1,1}, c_{a_2,4}\}, \{c_{a_1,2}, c_{a_2,1}\}, \{c_{a_1,2}, c_{a_2,2}\}, \{c_{a_1,2}, c_{a_2,3}\}, \{c_{a_1,2}, c_{a_2,4}\}\}.$$

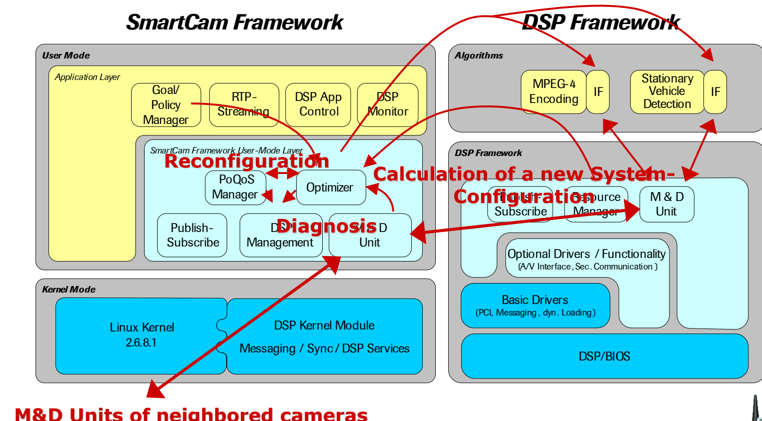
- Values of the Objective Functions

$$E_C = \{E_{C_1}, \ldots, E_{C_8}\} = \{9, 12, 7, 9, 12, 15, 10, 12\}$$
$$R_C = \{R_{C_1}, \ldots, R_{C_8}\} = \{90, 120, 70, 90, 120, 150, 100, 120\}$$
$$Q_C = \{Q_{C_1}, \ldots, Q_{C_8}\} = \{6, 6, 5, 5, 6, 6, 5, 5\}$$

→ C_3 is 'optimal' configuration

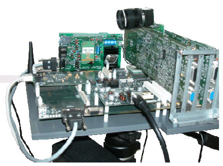
Implementation

- Additional Units in the *Smart**Cam*** Software Framework
  - → Allow dynamic reconfiguration



Conclusion

- Summary
  - → Investigation of a method for improving fault-tolerance and service availability in IVS systems
  - → Addition of monitoring, diagnosis and dynamic reconfiguration
- Future Work
  - → Practical evaluation
    - Experimental determination of resource usage
  - → Involvement of neighboring nodes
  - → Performance tests



Thank you for your attention!

http://www.iti.tugraz.at/smartcam