



Towards Efficient Video Compression Using Scalable Vector Graphics on the Cell Broadband Engine



**Andreea Sandu, Emil Slusanschi, Alin Murarasu, Andreea Serban,
Alexandru Herisanu, Teodor Stoenescu**

**University Politehnica of Bucharest
Computer Science and Engineering Department**

**International Workshop on Multi-core Software Engineering,
Cape Town, South Africa, May 1st 2010**



- Video Codecs & Image Characteristics
- NURBS Curves
- Image Representation
- Image Encoding
- Porting to the Cell/B.E.
- Results
- Related Projects @cs.pub.ro
- Conclusions & Outlook



Video Codecs

3

- A software program or library
- Encodes/Decodes the video component of a movie/clip in a digital format
- Aim: create a decoder using scalar vector graphics (SVG)
- Advantages of SVG:
 - Data Compression – efficient representation
 - Lossless display at any resolution – shape preservation
- Disadvantages of SVG: difficult conversion from raster

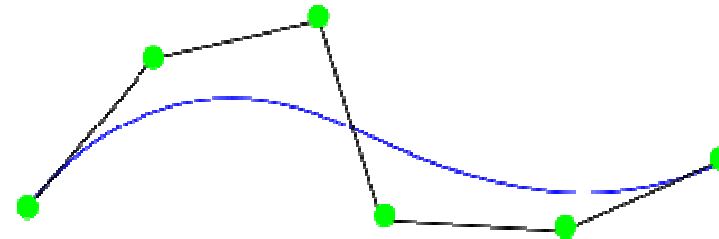




NURBS Curves

4

- NURBS = Non-uniform rational B-splines
- Can be used to represent curves and surfaces
- Used extensively in Computer Aided Design (CAD)
- Parameters
 - Degree (1,2,3,5,...)
 - Control points & weights
 - Knots
- NURBS advantages:
 - Invariant to scalar transformations
 - Computable with stable algorithms (e.g. DeBoor)
 - Can represent complex features with few parameters
 - A curve can be handled easily through its parameters

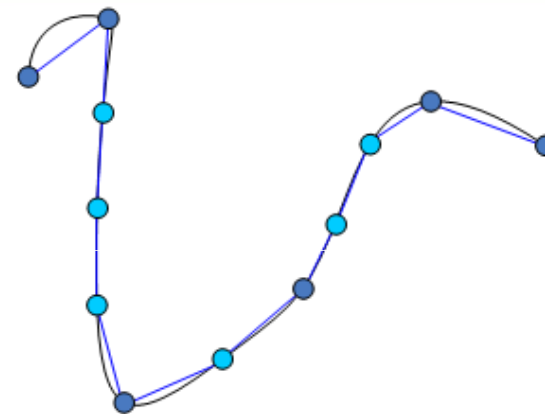




NURBS Conversion

5

- Polygonal approximation:
 - Curve evaluation – deBoor's algorithm
 - Initial approximation to curve knots
 - Iterative process of adding nodes
- Integrated in ffmpeg & ogg & used in the VLC player



Video frame

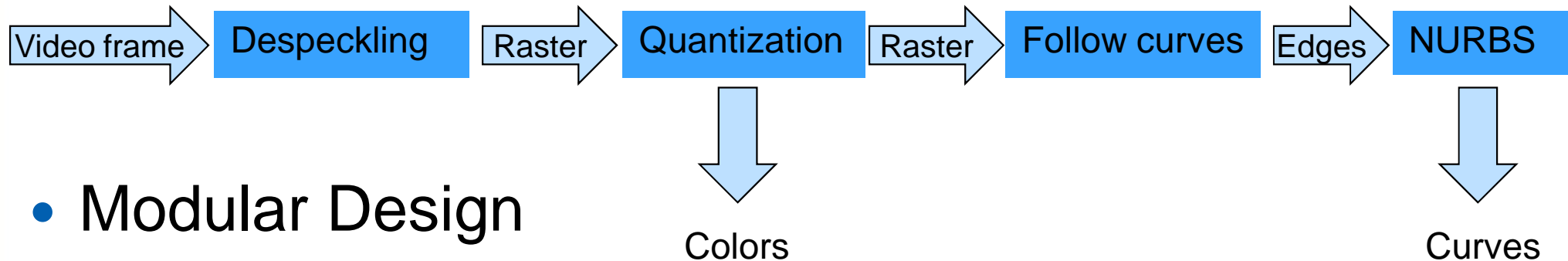


Internal Representation



Image Encoding

6



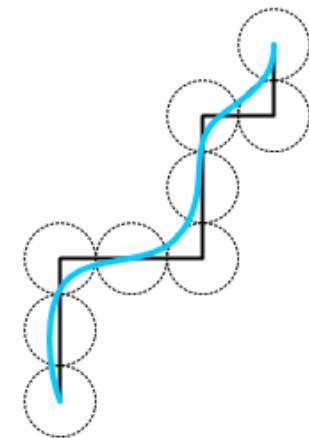
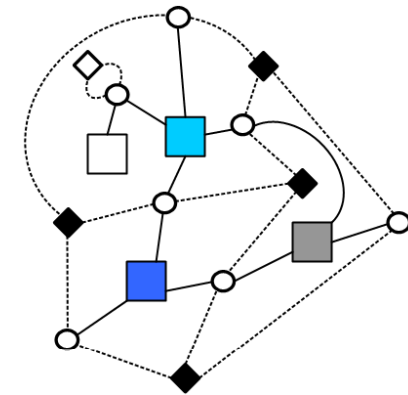
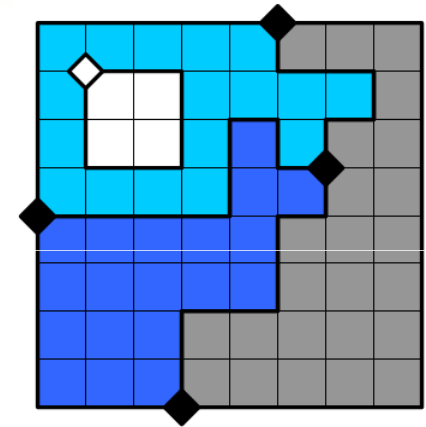
- Modular Design
- Stage algorithms can be treated independently:
 - Despeckling & noise filtering
 - Create big pieces of same color zones – similar to AutoTrace, by smoothing/combining neighboring pixels of similar colors
 - Color quantization
 - Create a new color scale
 - The algorithm is based on octrees
 - Reduce number of colors in order to reduce the image size in the vector representation – loses details/quality vs. original image



Feature Extraction with NURBS

7

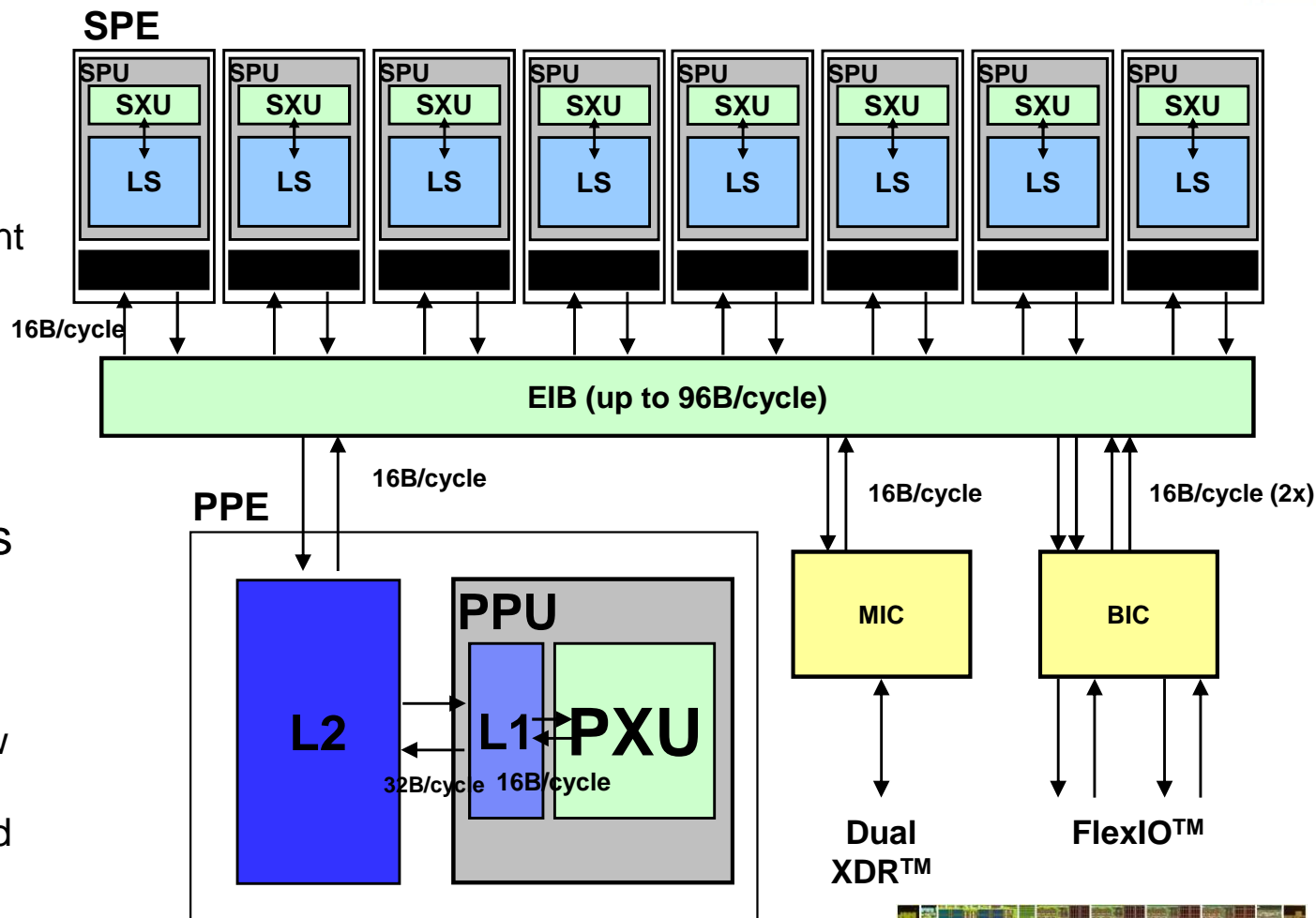
- Determine zones of constant color
- Determine edges between these zones using NURBS curves
- Determine knots of sharing edges
 - The approximation is passing through these knots
 - The approximation uses a least-squares approach



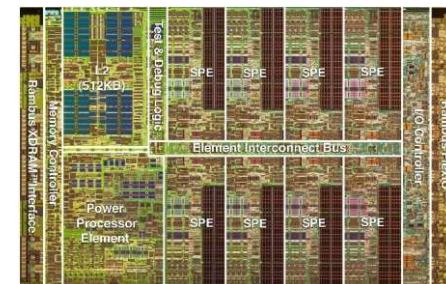


IBM's Cell/B.E. Processor

- Heterogeneous multi-core system architecture
 - Power Processor Element for control tasks
 - Synergistic Processor Elements for data-intensive processing
- Synergistic Processor Element (SPE) consists of
 - Synergistic Processor Unit (SPU)
 - Synergistic Memory Flow Control (SMF)
 - Data movement and synchronization
 - Interface to high-performance Element Interconnect Bus



64-bit Power Architecture with VMX

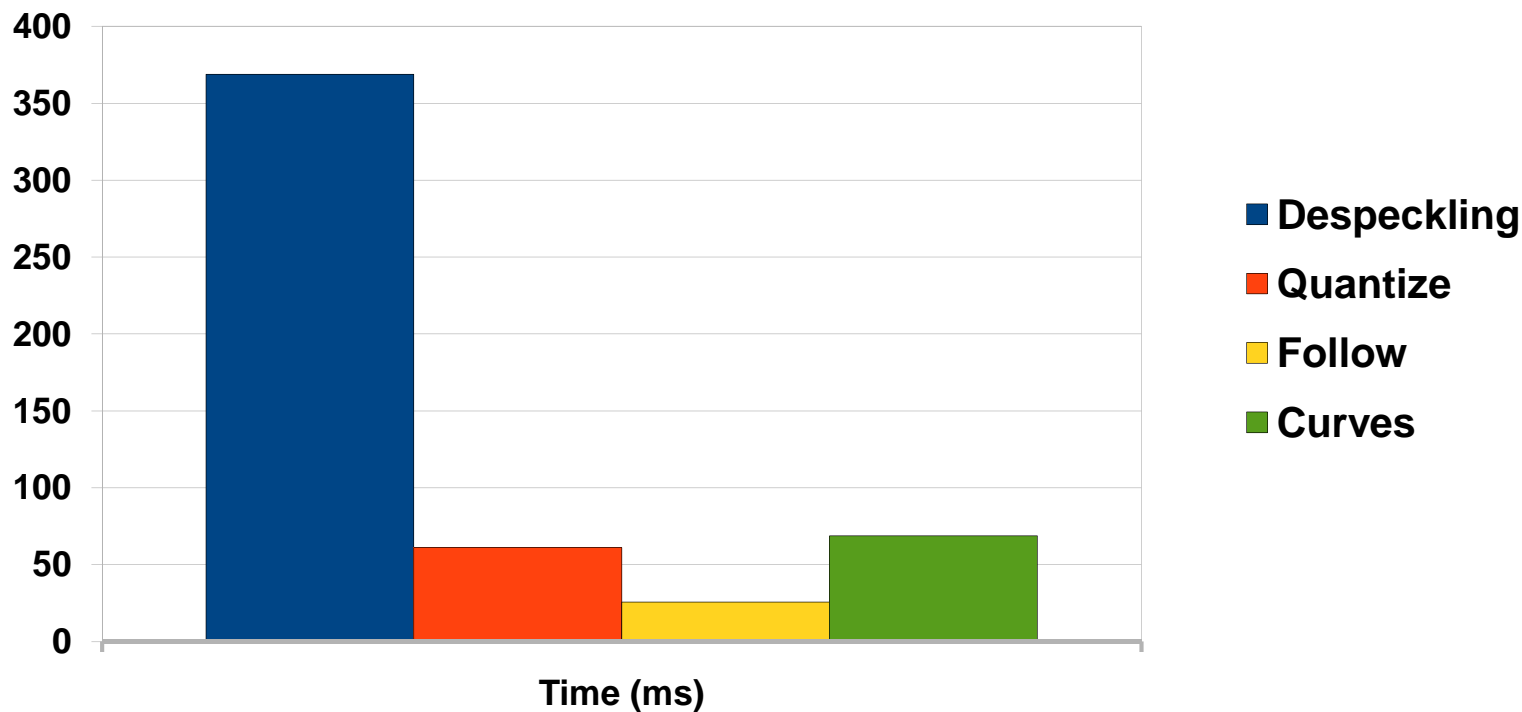




Encoding – Serial Profiling

9

- Profiling is done on slices of 308 x 400 pixels



Phase	Despeckling	Quantize	Follow	Curves
Time (ms)	368.8	61.051	25.822	68.884
Percentage	70.31%	11.64%	4.92%	13.13%



Porting to the Cell/B.E. Architecture

10

- IBM's Cell/B.E. is heterogeneous: PPE/SPE
- Usual image processing algorithms methodology
 - Divide the problem – process data – reconstruct results
- Image Despeckling
 - Whole algorithm is run on the SPEs
- Image Quantization
 - The PPE divides the frame
 - The SPEs generate the octrees
 - The PPE fuses the octrees together
- NURBS curves
 - Entire algorithm is run on SPEs



Despeckling Design Tradeoffs

11

- Split image in slices and distribute them to SPUs
- Smoothing is done independently by each SPU
- The PPU rebuilds the image from the processed fragments
- Tradeoffs:
 - The slices are too small – the smoothing will be exaggerated
 - The slices are too big – they will not fit on the SPU local storage memory



Quantization Design Tradeoffs

12

- The PPU decides if/when to reduce the number of colors
- The SPUs generate partial color trees with a maximal number of levels by counting pixels of each color
- The PPU combines the SPU generated trees in a global tree
- Tradeoffs:
 - The slices are too big – generate too many partial trees and too many DMA transfers & significant overhead in the global tree reconstruction
 - The slices are too small – processing on the PPU may be more efficient



Quantization SIMD/Vectorization

13

- Groups of 3 bits from the three basic color (RGB) components are forming paths in the partial trees built by SPUs:
 - $\text{bit_R} \ll 2$
 - $\text{bit_G} \ll 1$
 - $\text{bit_B} \ll 0$
- Computing the paths serially is done with successive shifts in 8 iterations
- The vector/SIMD version allows the computing of entire vector paths in the partial tree at once



Ongoing developments

14

- Edge detection component in the quantization phase moved from PPU to SPUs
 - Color trees are aligned to ease transfer and processing
 - Each SPU makes a local copy & converts pixels to codes
 - After conversion release memory to allow edge detection passes to continue
- Tradeoffs:
 - Edge detection algorithm generates useless edges around the current slice to avoid lots of coordinate testing
 - Big slices are good because of code serialization – no more branching code
 - Small slices – generate lots of useless edges thus increasing storage requirements



Results – Image Quality

15

Original
Image



4SPUs



8SPUs



16SPUs

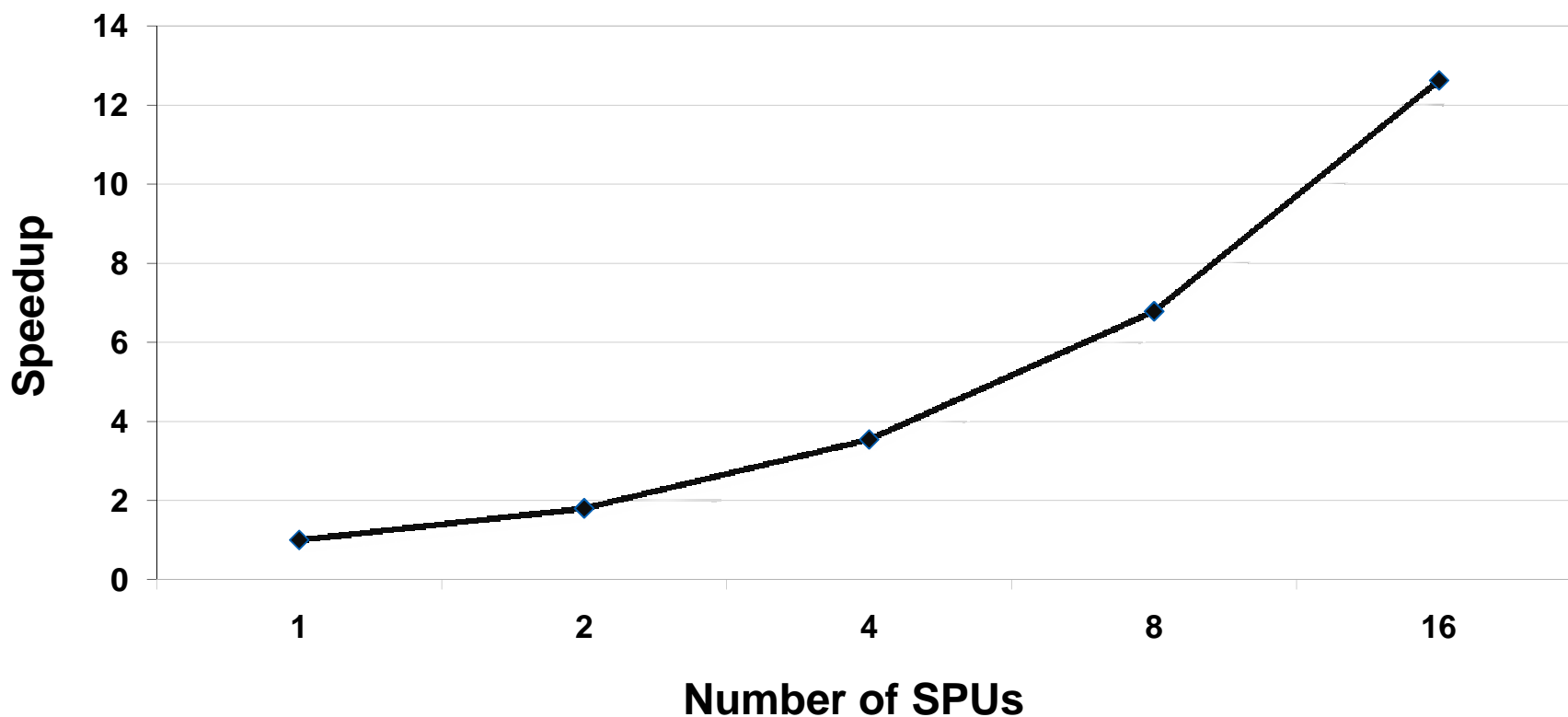


- x86 codec speed: 4-6 fps
- Compression ratio to date: 0.982 – **1.754**



Results – Despeckling@SPUs

SPUs	1	2	4	8	16
Time (ms)	368.80	204.96	104.10	54.35	29.21
Speedup	1.00	1.80	3.54	6.79	12.63

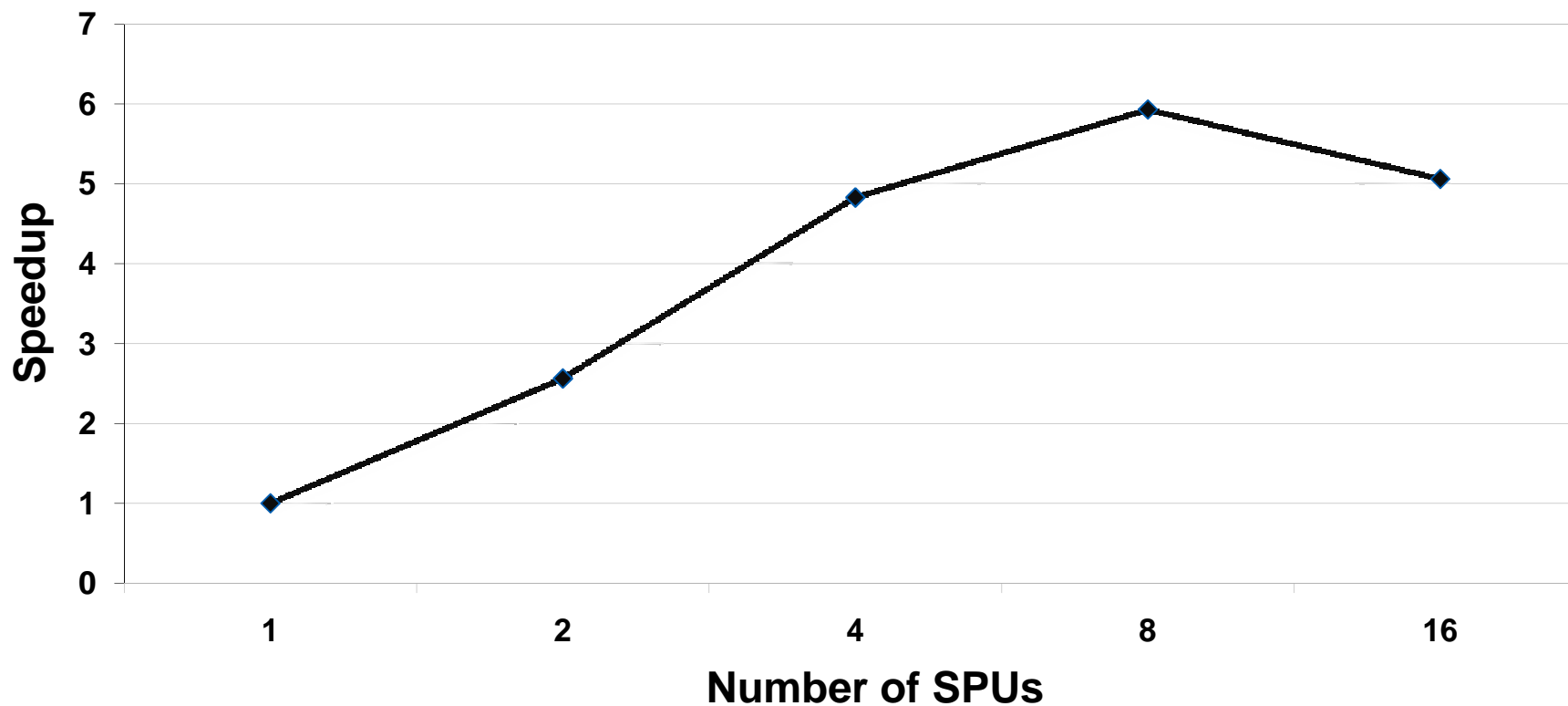




Results – Quantization@SPUs

17

SPUs	1	2	4	8	16
Time (ms)	61.05	23.83	12.64	10.29	12.06
Speedup	1.00	2.56	4.83	5.93	5.06





Feature Extraction from Satellite Images on Hybrid x86/CellBE Systems



Original



Grayscale

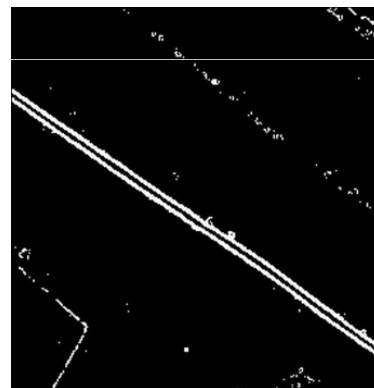
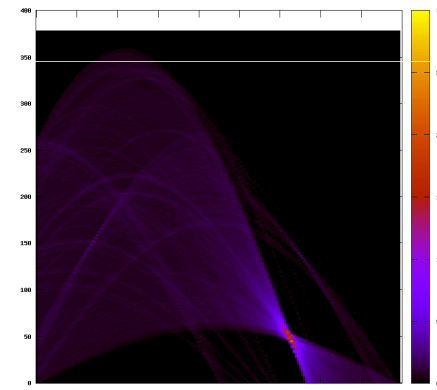
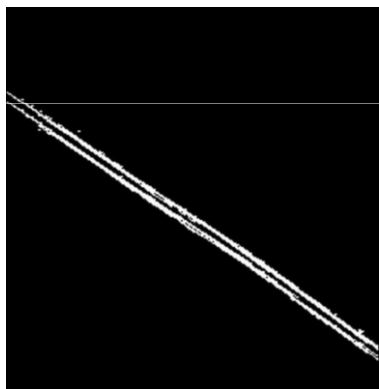


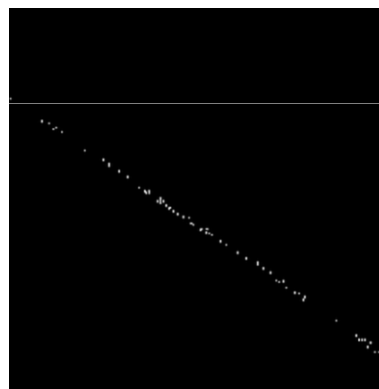
Image Detection
(Sobel)



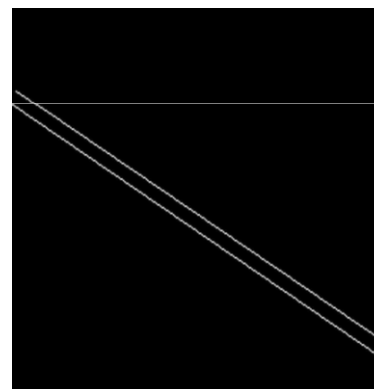
Hough Accumulator



Hough Peaks
over image edges



Mark road segment
edges



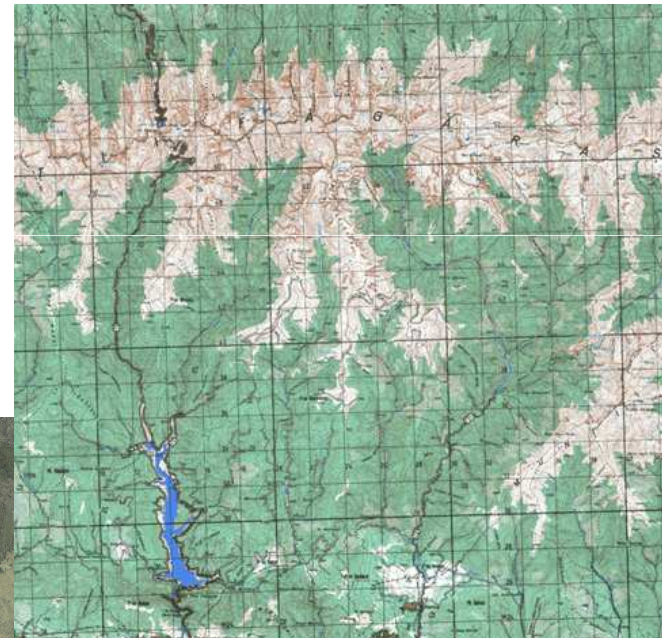
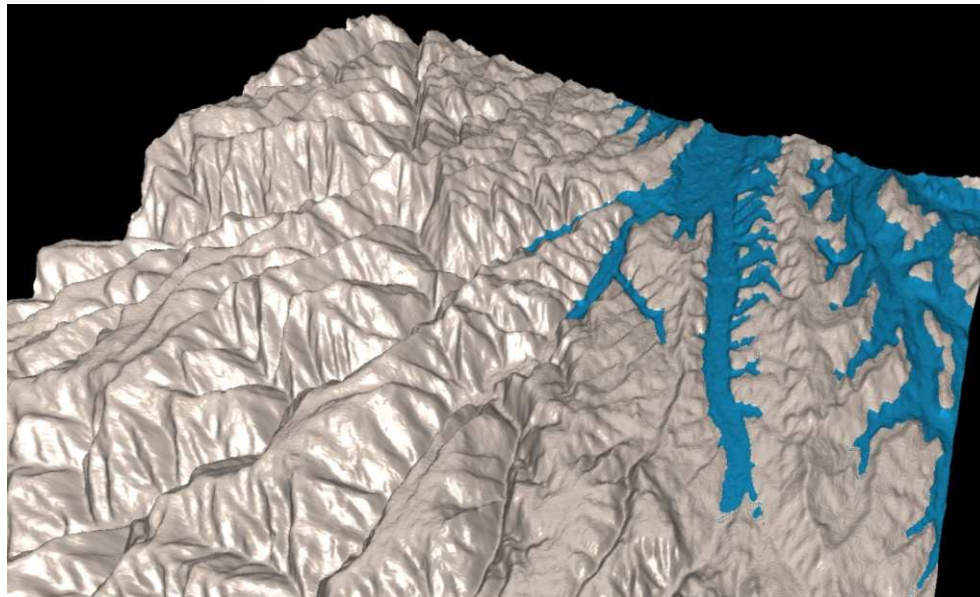
Final identified
feature (road)
Saved as SVG





Related Projects @cs.pub.ro

19



**Interactive
3D Map of
Romania**



**SVG for Map
Representation**



Conclusions & Outlook

20

- Conclusions
 - The performance of the SVG codec benefits from its deployment on the Cell/B.E. architecture
 - The quality and performance of the codec are strongly dependent on design choices in the processing steps
 - The codec compression still requires further improvement
- Outlook
 - Currently only Intra-coded-frames (I-frame) are encoded leading to big SVG file sizes
 - Add support for Predicted (previous) & Bi-coded (previous & next) frames thus improving SVG storage requirements
 - Use motion estimation techniques between reference I-frame blocks & blocks in subsequent frames (translation/rotation/etc)
 - The offsets/differences are stored in motion vectors



Computer Science
& Engineering
Department



Thank you for your attention

Q & A

cs.pub.ro

emil.slusanschi@cs.pub.ro