Video-Rate Stereo Matching using MRF TRW-S Inference on a Hybrid CPU+FPGA Computing Platform

Jungwook Choi and Rob A. Rutenbar
{jchoi67, rutenbar}@illinois.edu

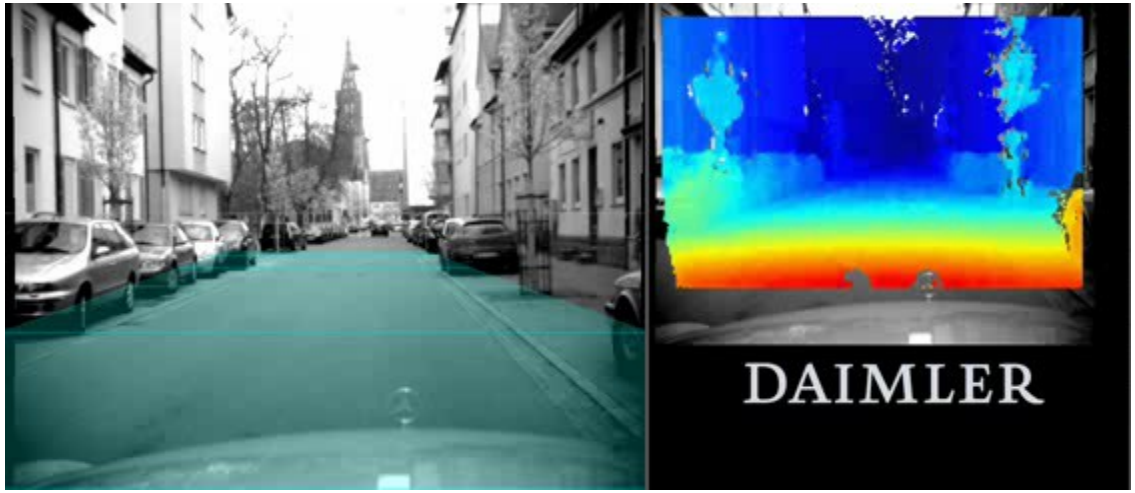
ISFPGA 2013, Feb. 12 2013

Overview

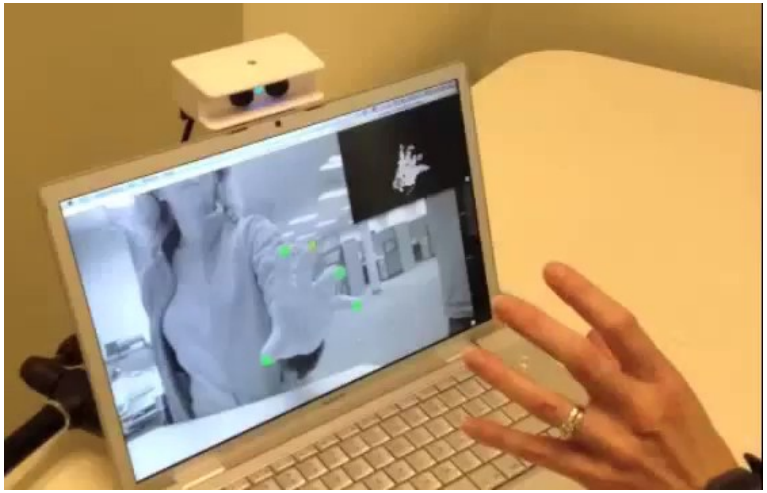
- Overall goal: Explore probabilistic inference methods in custom hardware
- This work: Fast & high quality stereo matching via Markov random field (MRF) inference in FPGA
- Three key contributions:
  - CPU-FPGA partitioning of stereo matching procedure
  - Functional level pipelining across CPU and FPGAs
  - Frame level parallelization to process multi-frames in parallel
- Results: 15~40 frame/sec for QVGA stereo matching

Fast & High Quality Stereo Matching

- Variety of real-world apps demand **Fast & High Quality** stereo matching



Pedestrian detection for autonomous vehicle*
(M. Gavrila, IJCV07, etc.)



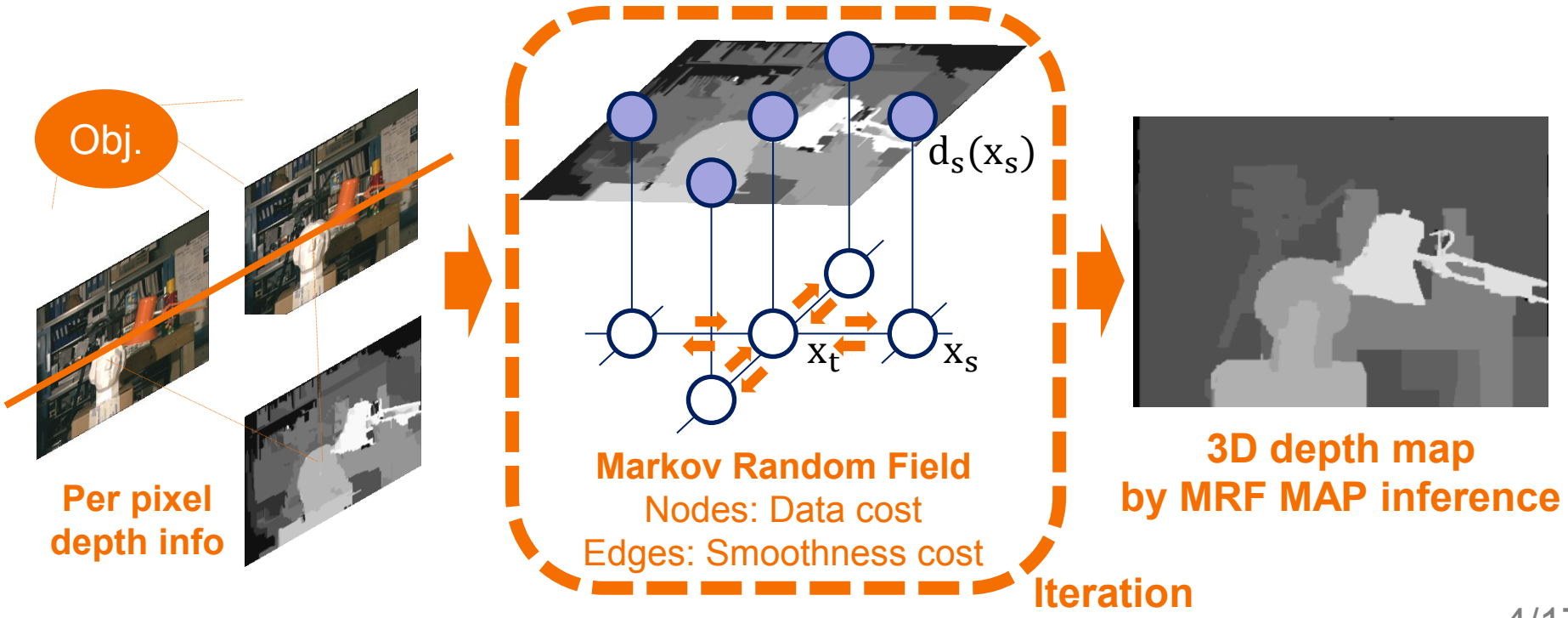
Laptop control using gestures**
(TYZX G3 EVS)

* http://www.gavrila.net/Research/Pedestrian_Detection/pedestrian_detection.html
** http://www.tyzx.com/technology/gallery.html

MRF Inference for High Quality Stereo Matching

- Stereo matching is mapped to energy minimization (or inference) on Markov random fields (MRF)

$$\text{Energy}(\mathbf{x}) = \left(\underbrace{\sum_{s} d_s(x_s)}_{\text{Data cost}} + \underbrace{\sum_{s,t} V_{st}(x_s, x_t)}_{\text{Smoothness cost}} \right)$$

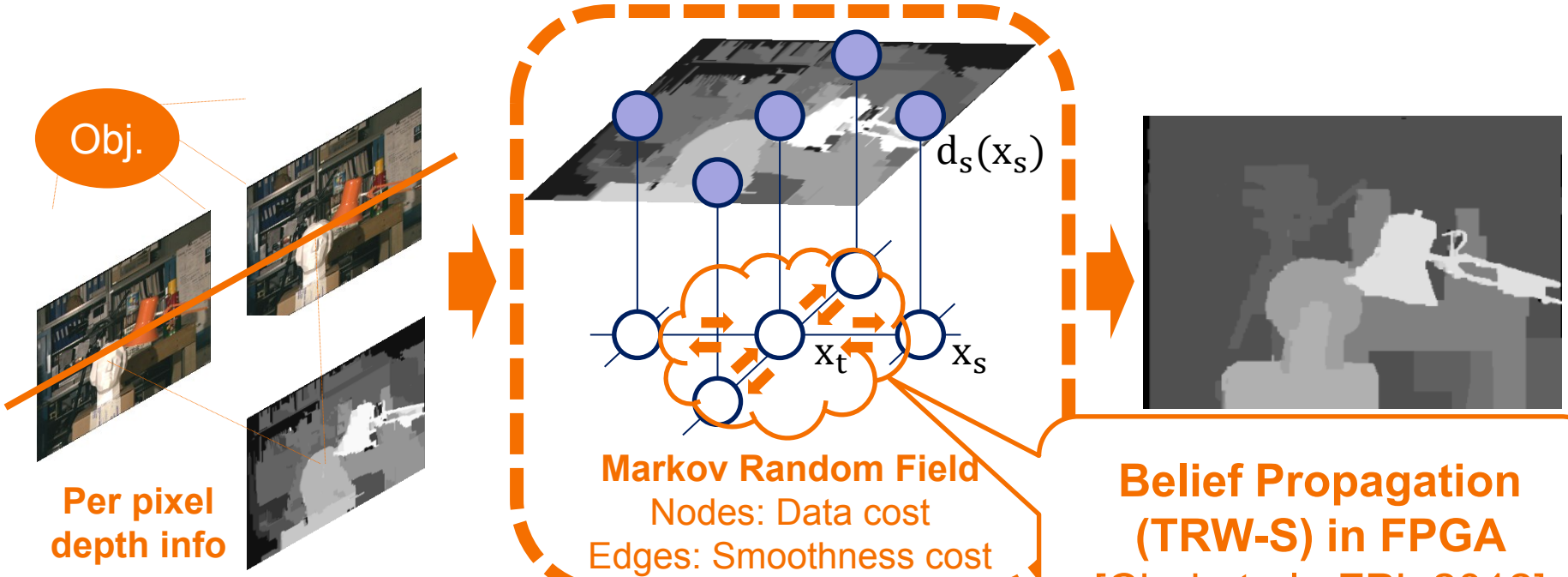


MRF Inference for High Quality Stereo Matching

- Stereo matching is mapped to energy minimization (or inference) on Markov random fields (MRF)

$$\text{Energy}(\mathbf{x}) = \left(\sum_{s} d_s(x_s) + \sum_{s,t} V_{st}(x_s, x_t) \right)$$

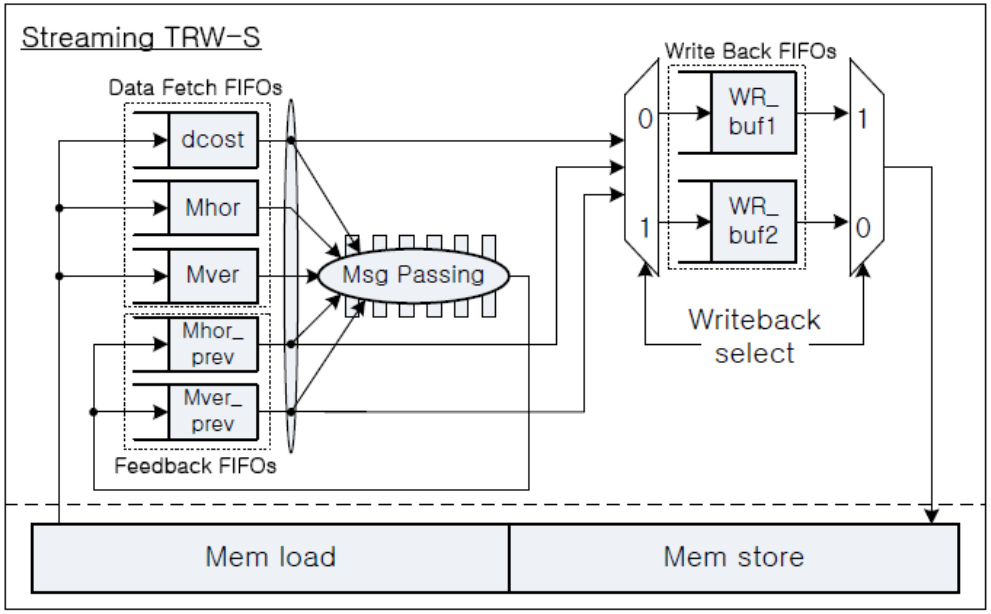
Data cost Smoothness cost

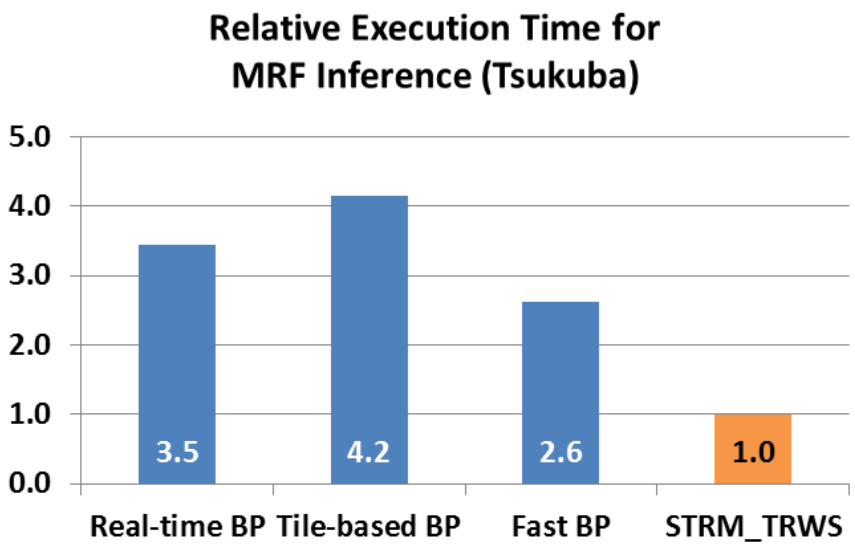


Belief Propagation (TRW-S) in FPGA
[Choi et al., FPL 2012]

Custom HW Accelerator for Fast Stereo Matching

- HW impl. for computation intensive MRF infer.
- Custom HW accel. is 2~4x faster than GPU impl.
  - Thanks to fully pipelined data path and streaming data access

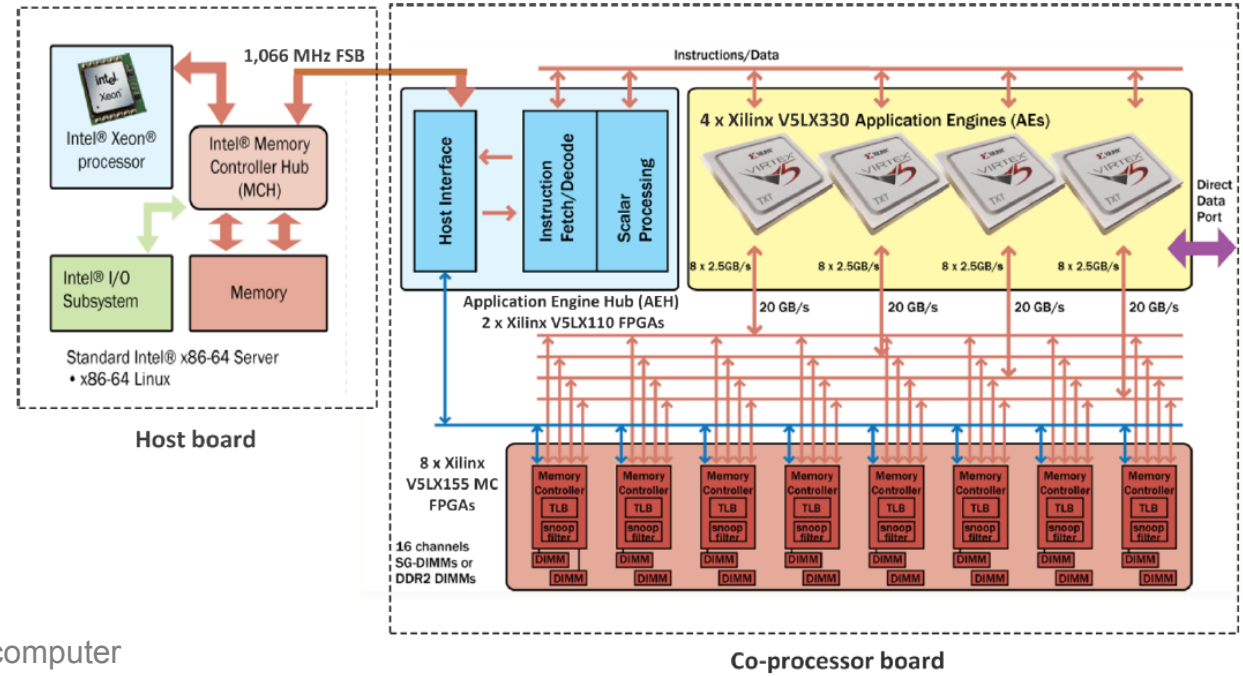




J. Choi , and R.A. Rutenbar, "Hardware implementation of MRF MAP inference on an FPGA platform," FPL 2012

Hybrid CPU+FPGA Platform

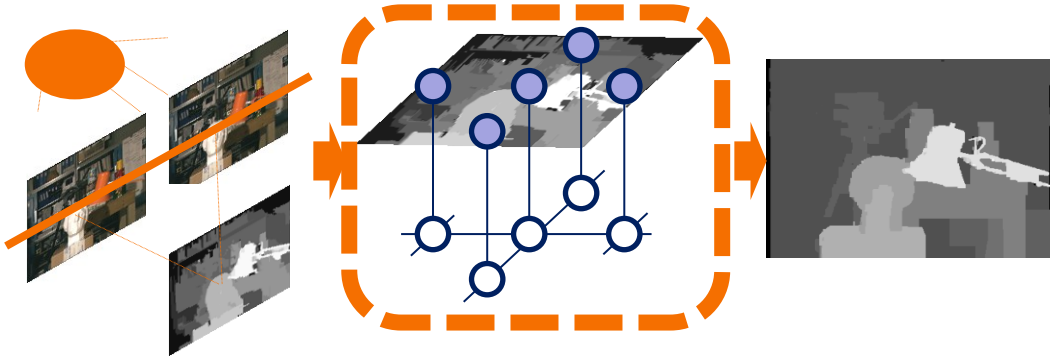
- Our platform: Convey HC-1
  - CPU-FPGA cache-coherent virtual memory system
  - Max memory BW: 1Kbit/cycle(~20GB/sec)/FPGA (runs @150MHz)
  - Non-blocking FPGA function call



* Image from Convey computer

CPU-FPGA Partitioning of Stereo Matching Procedure

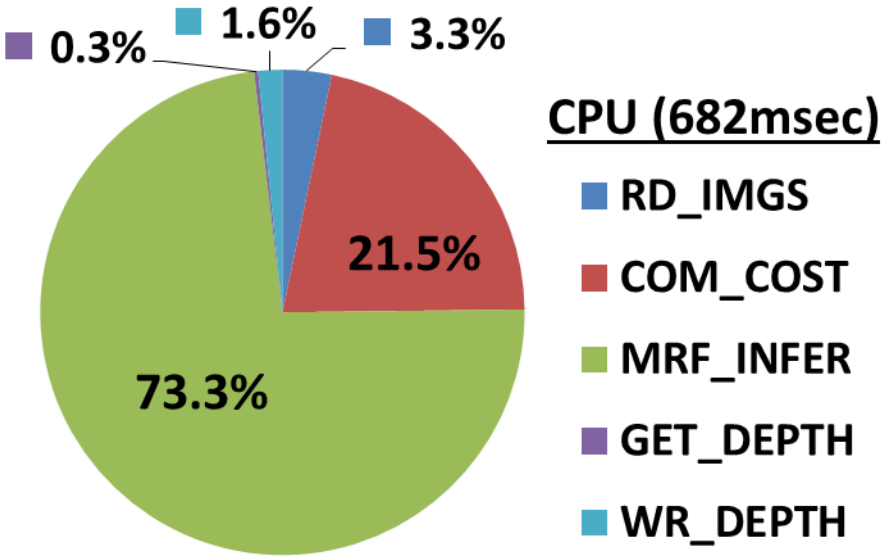
CPU



Initialize
For (idx=0; idx<numFrame; idx++)
Begin

Get frame images (RD_IMGS)
Compute MRF costs (COM_COST)
Run MRF inference (MRF_INFER)
Obtain depth result (GET_DEPTH) Produce depth map (WR_DEPTH)

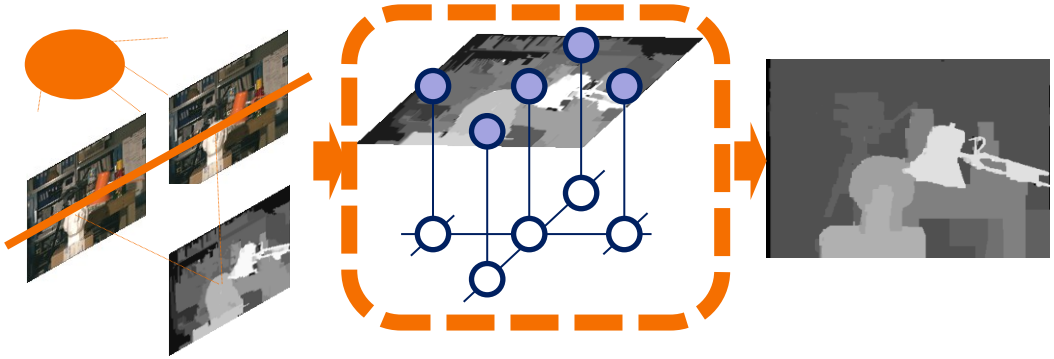
End



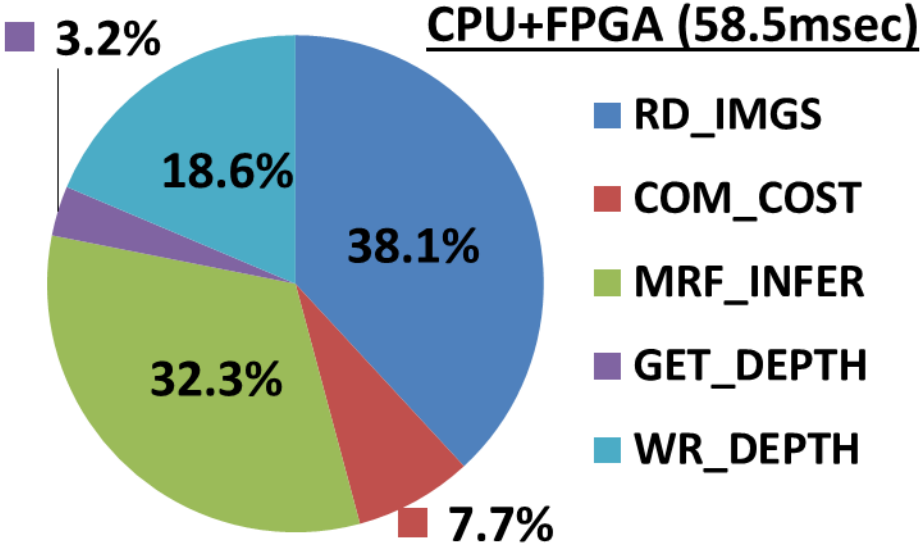
CPU-FPGA Partitioning of Stereo Matching Procedure

CPU

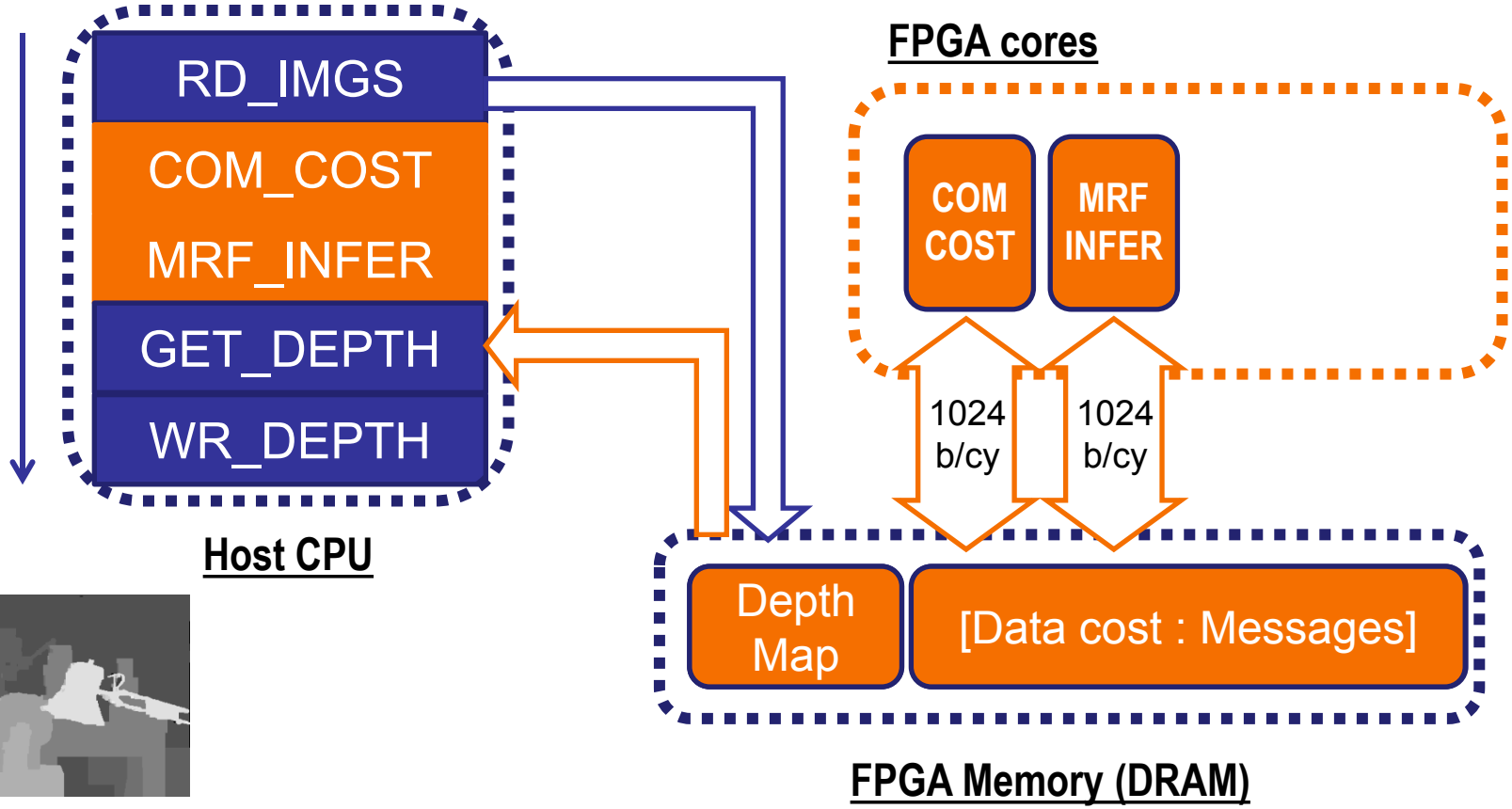
FPGA



Initialize
For (idx=0; idx<numFrame; idx++)
Begin
Get frame images (RD_IMGS)
Compute MRF costs (COM_COST)
Run MRF inference (MRF_INFER)
Obtain depth result (GET_DEPTH)
Produce depth map (WR_DEPTH)
End



Video-Rate Stereo Matching CPU+FPGA System



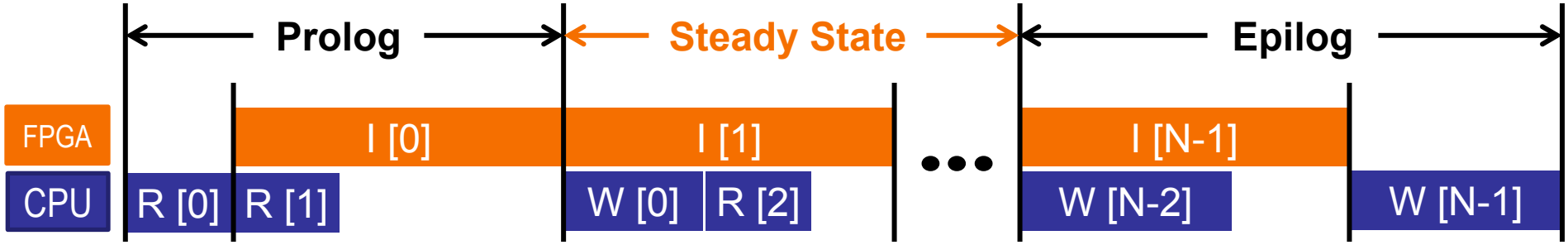
Function Level Pipelining

- Apply SW pipelining technique to overlap execution time of CPU and FPGA functions.
- While FPGA functions work on *current* frame, CPU functions process *previous* and *next* frames in Steady State

Initialize
For (idx=0; idx<NumFrame; idx++)
Begin

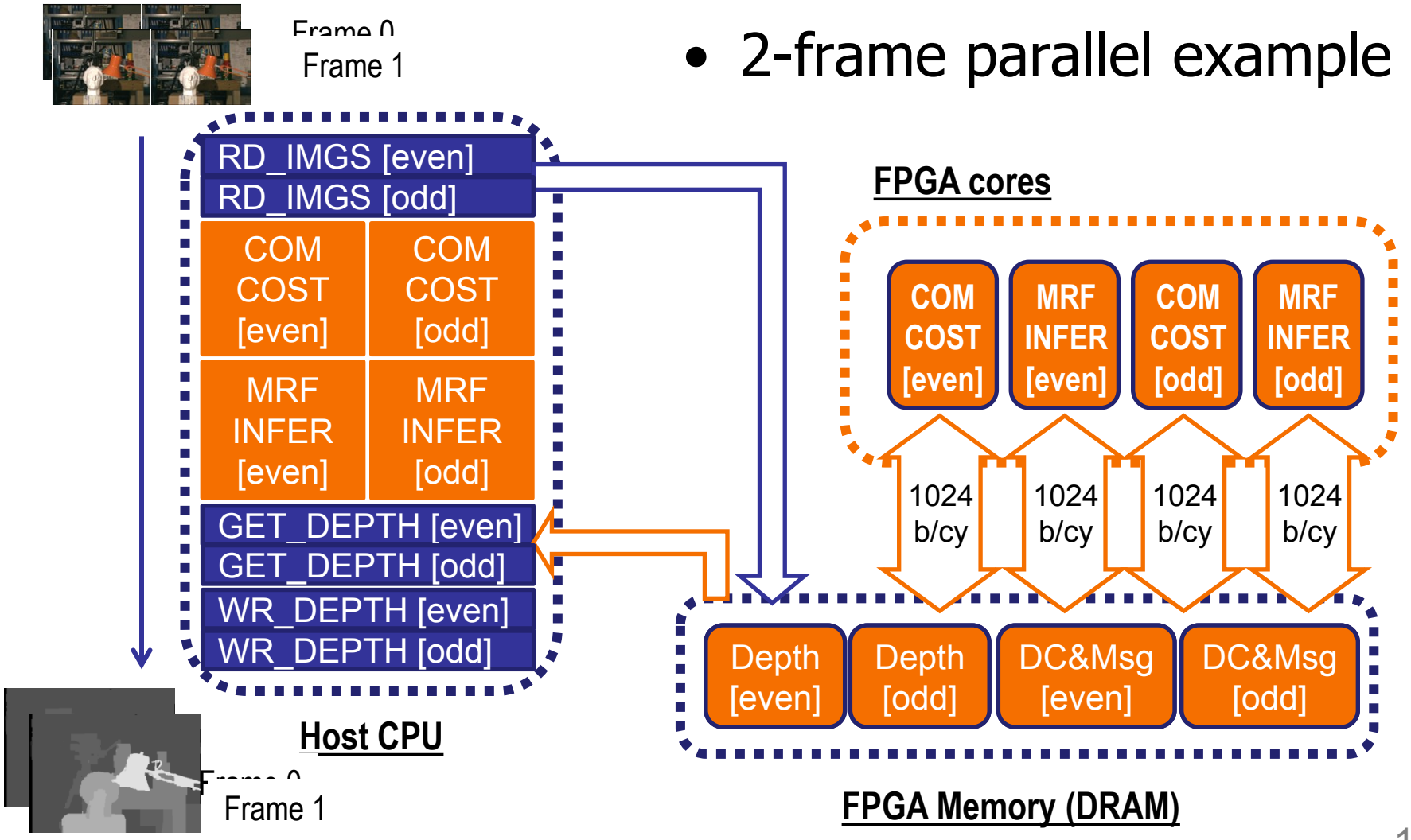
R	RD_IMGS	[idx]
I	COM_COST	[idx]
	MRF_INFER	[idx]
W	GET_DEPTH	[idx]
	WR_DEPTH	[idx]

End

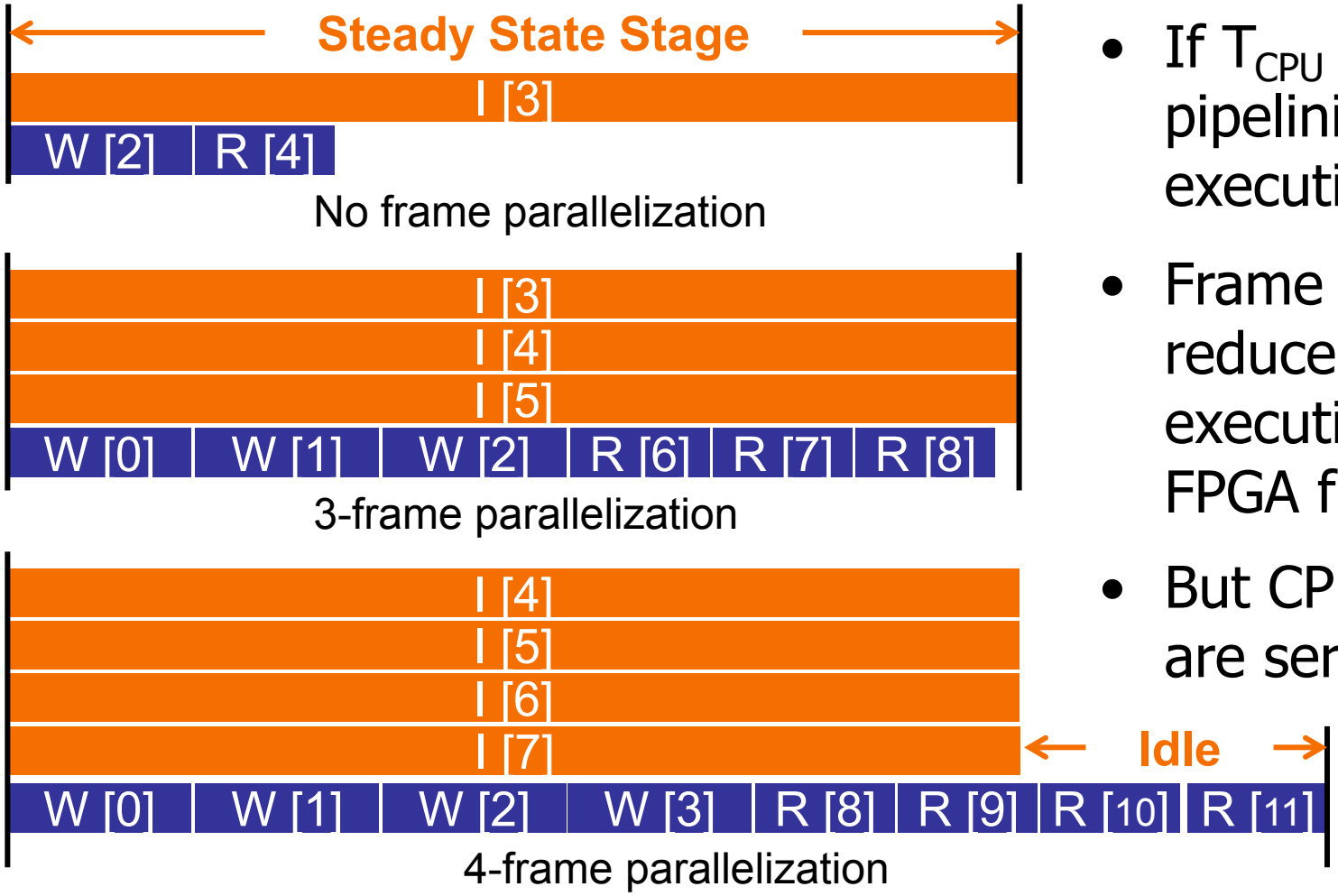


Frame Level Parallelization

- 2-frame parallel example



Function Level Pipelining + Frame Level Parallelization



- If $T_{CPU} < T_{FPGA}$ pipelining hides CPU execution time
- Frame level par. reduces per-frame execution time of FPGA functions
- But CPU functions are serialized

➔No benefit!

Performance Result: Single Frame

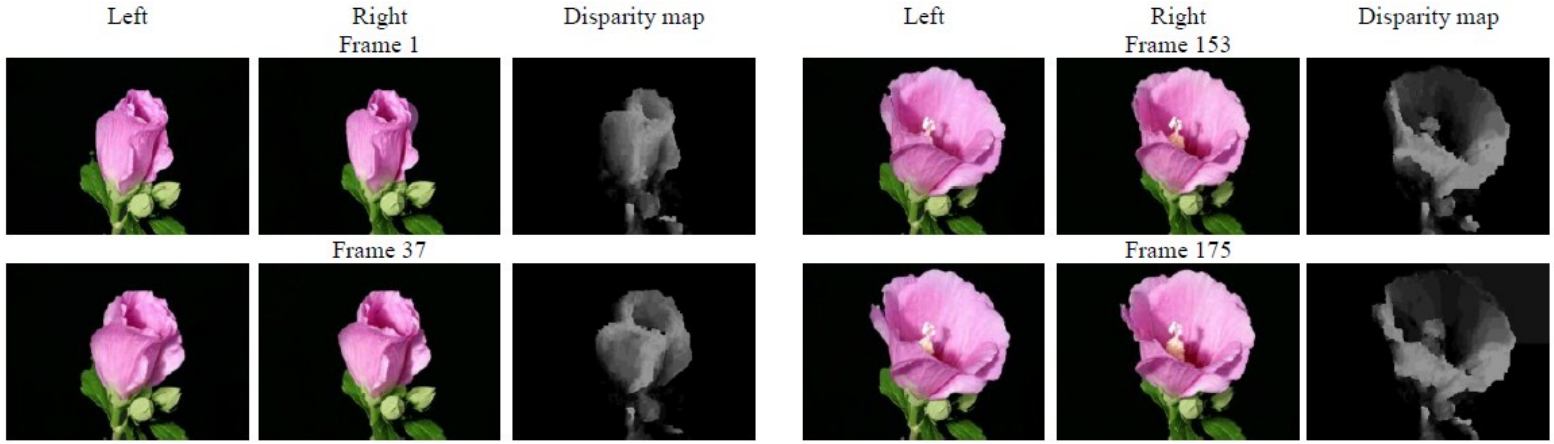
Tsukuba (384x288,16)	Real-time BP* [Yang 2006]	Tile-based BP** [Liang 2011]	Fast BP*** [Xiang 2012]	This work
GPU	NVIDIA GeForce 7900 GTX	NVIDIA GeForce 8800 GTS	NVIDIA GeForce GTX 260	N/A
# Iteration	(4 scales) = (5,5,10,2)	(B, T_I, T_O) = (12, 20, 5)	(3 scales) = (9,6,2)	T_O = 5
Time (msec)	80.8	97.3	61.4	**26.10**
				
Min. Energy	N/A	396,953	N/A	**393,434**

* Q. Yang, et al., "Real-time global stereo matching using hierarchical belief propagation," *BMVC,* 2006.
** Liang, et al., "Hardware-Efficient Belief Propagation," *IEEE Trans. Circ. Syst. Video Tech*, May 2011.
*** X. Xiang, et al., "Real-time stereo matching based on fast belief propagation," *MACH VISION APPL*, 2012

Performance Result: Video Sample

- Tasks: Flower*(360x262,16), Ume*(320x240,16)
  - Require 80+ iterations to remove defect at the boundary region

Flower (360x262x16)

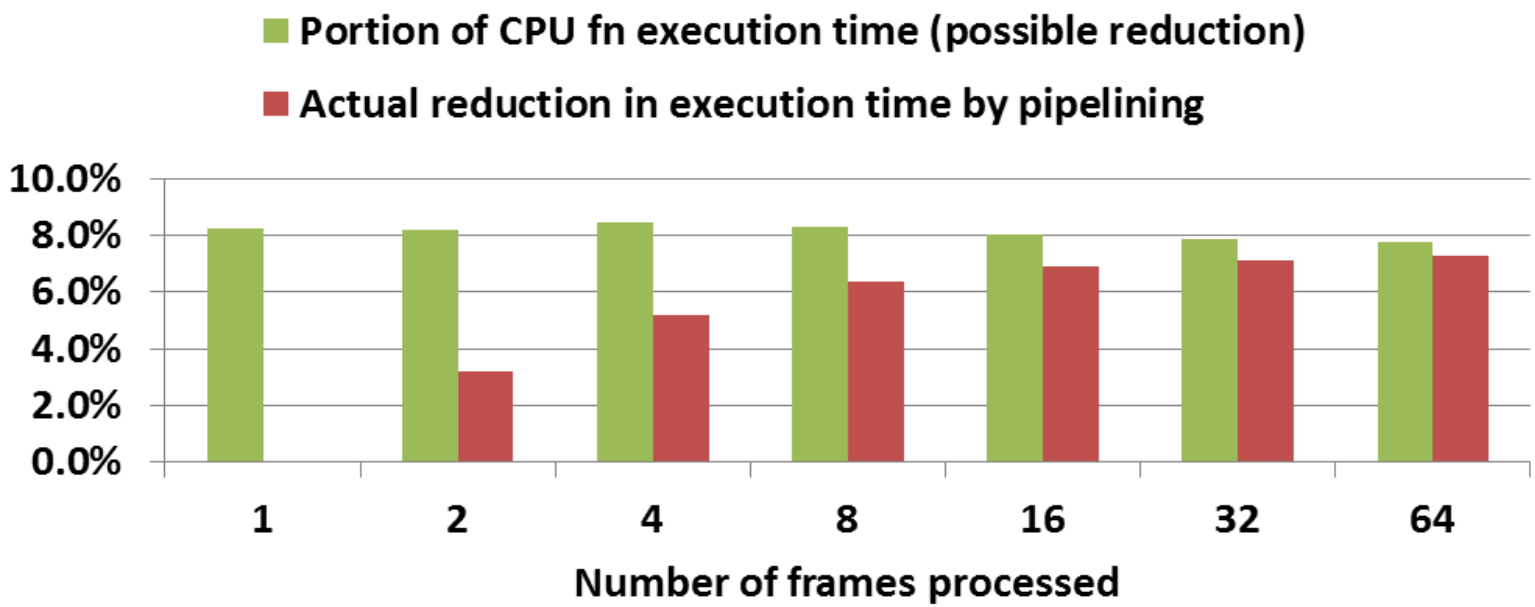


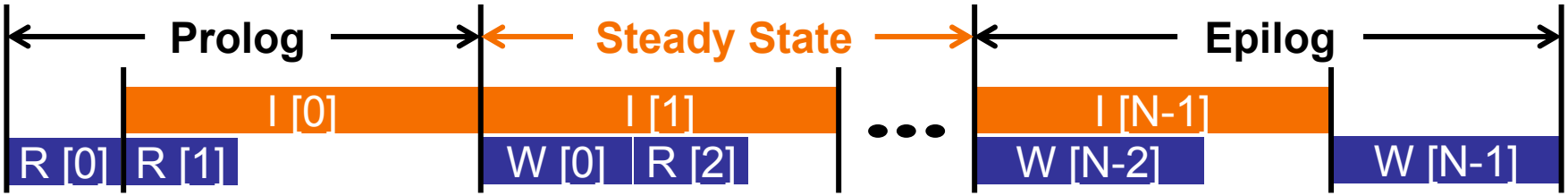
Ume (320x240x16)

* Stereo movie sample. http://www.stereomaker.net/sample/index.html.

Performance Result: Function Level Pipelining

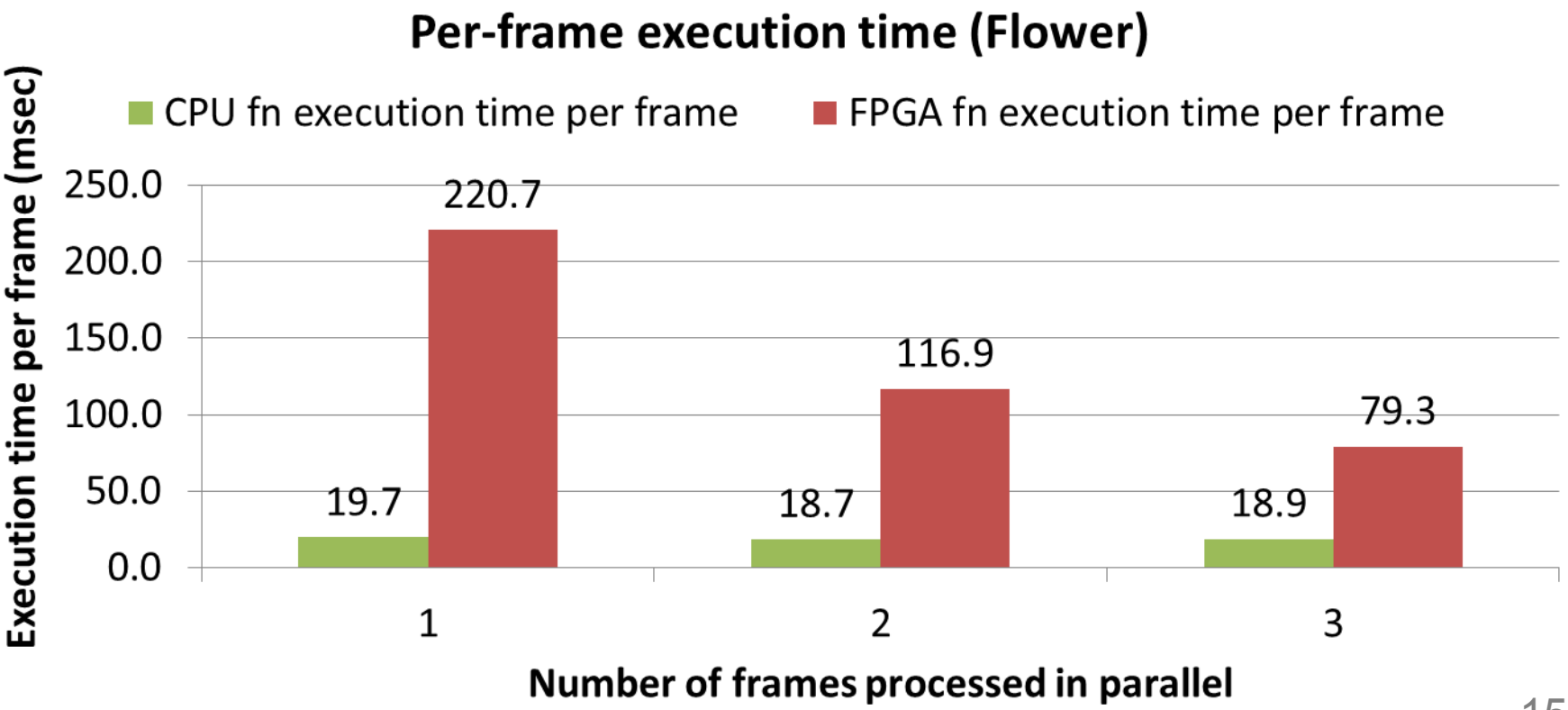
- Function level pipelining hides CPU overhead
  - As frames processed, more CPU fn are overlapped by FPGA fn





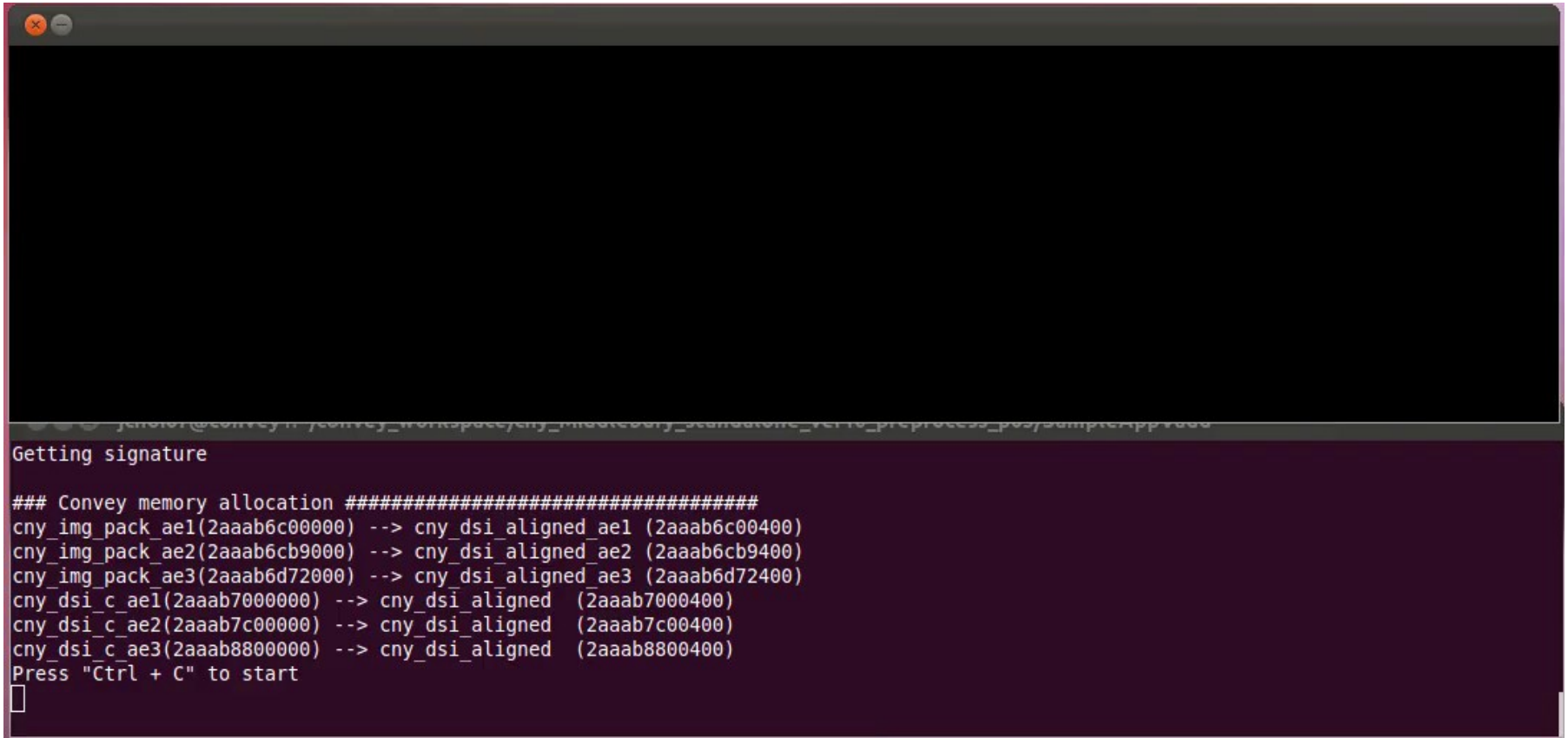
Performance Result:
Frame Level Parallelization

- Frame level parallelization speed-ups FPGA functions
  - CPU fn exec time is unchanged, but minor → hidden by pipelin.



Performance Result: Video Demo

- Function level pipelining + frame level parallelization
  - 3-frame parallel stereo matching, 80 iterations for each inference
  - Speed: Flower 12.3 frame/sec, Ume 15.0 frame/sec



Conclusion

- Video-rate of high quality stereo matching is done using MRF inference on a hybrid platform
- CPU-FPGA partitioned functions of stereo matching procedure are pipelined and frame-level parallelized
- Performance results:
  - Simple QVGA single frame: 40 frame/sec
  - Challenging QVGA video sample: 15 frame/sec
- Fastest ever implementation of TRW-S MRF inference

Thanks !

Any Questions?