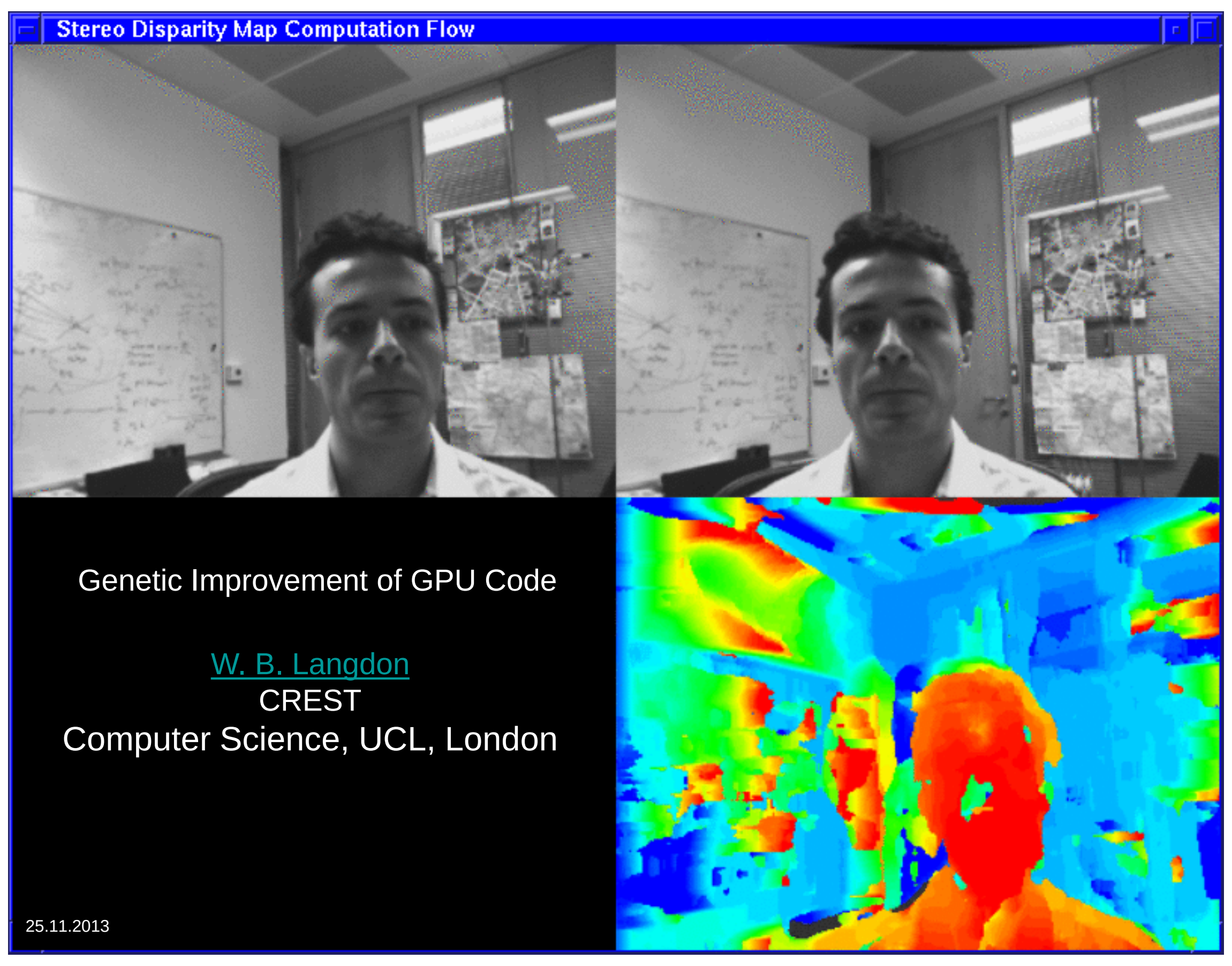
CREST

Genetically Improved CUDA

EPSRC
Engineering and Physical Sciences Research Council

W. B. Langdon
GISMO, Department of Computer Science, University College London, Gower Street, London WC1E 6BT, UK (Email: w.langdon@cs.ucl.ac.uk)

Genetic Programming Speeds up Existing CUDA kernel


Stereo Disparity Map Computation Flow

Genetic Improvement of GPU Code

W. B. Langdon
CREST
Computer Science, UCL, London

25.11.2013

1 Helping the GPU Programmer

GPUs have a deserved reputation for being hard to program. In addition to the usual programming tasks of algorithm design, an efficient design must exploit the GPUs hardware. A programmer must answer questions like: what is the best way of allocating data to textures, global, shared or local memory? This stereo vision example shows that genetic programming (GP) can take this load from the programer's shoulders and automatically configure and modify CUDA kernels to best exploit the capabilities of each GPU.

2 6 types of GPU

Name	year		MP	Cores	Clock
Quadro NVS 290	2007	1.1	2 × 8	16	0.92 GHz
GeForce GTX 295	2009	1.3	30 × 8	240	1.24 GHz
Tesla T10	2009	1.3	30 × 8	240	1.30 GHz
Tesla C2050	2010	2.0	14 × 32	448	1.15 GHz
GeForce GTX 580	2010	2.0	16 × 32	512	1.54 GHz
Tesla K20c	2012	3.5	13 × 192	2496	0.71 GHz

3 Tradeoff 2 objectives Pareto front

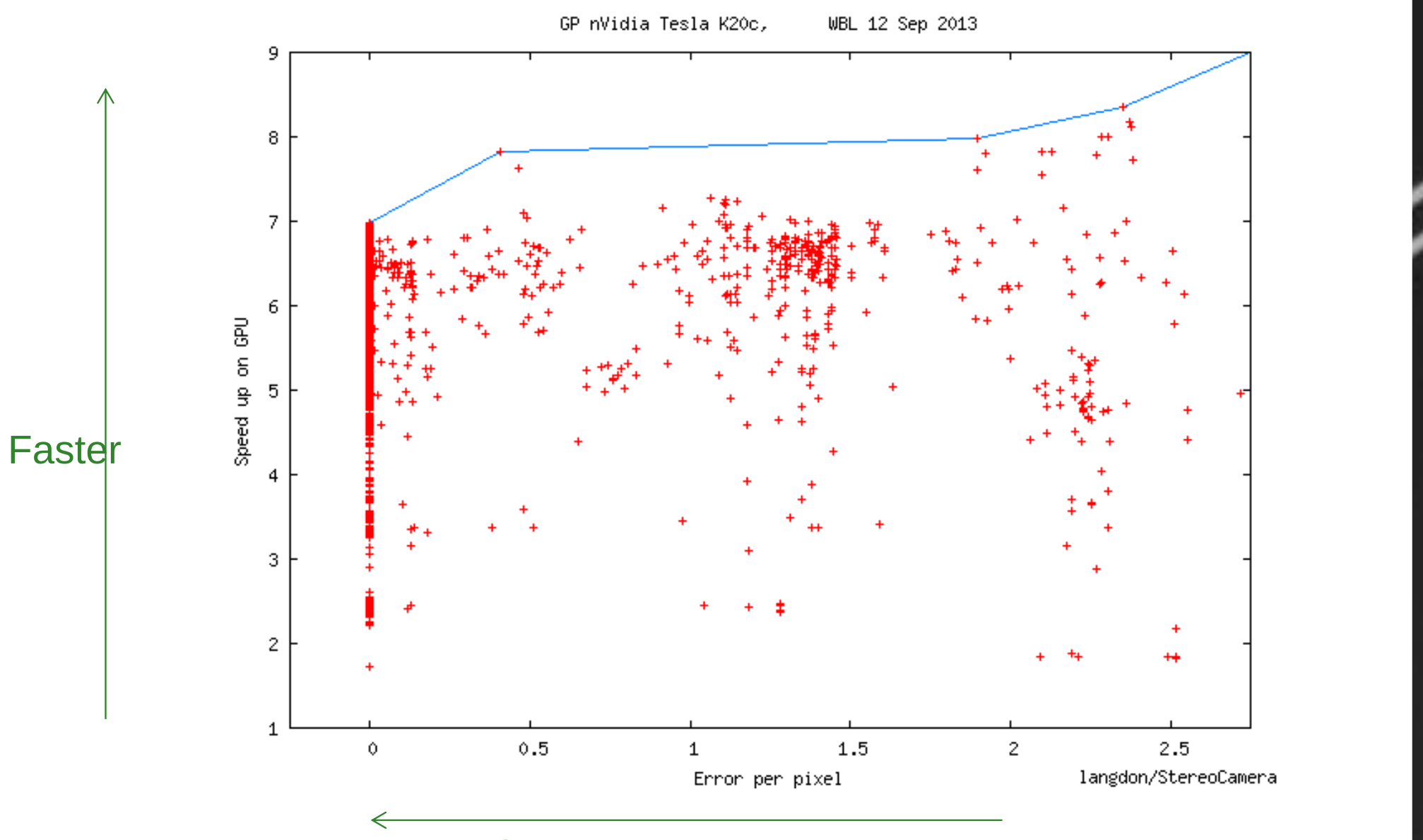


Figure 2

4 Calculating stereo discrepancy

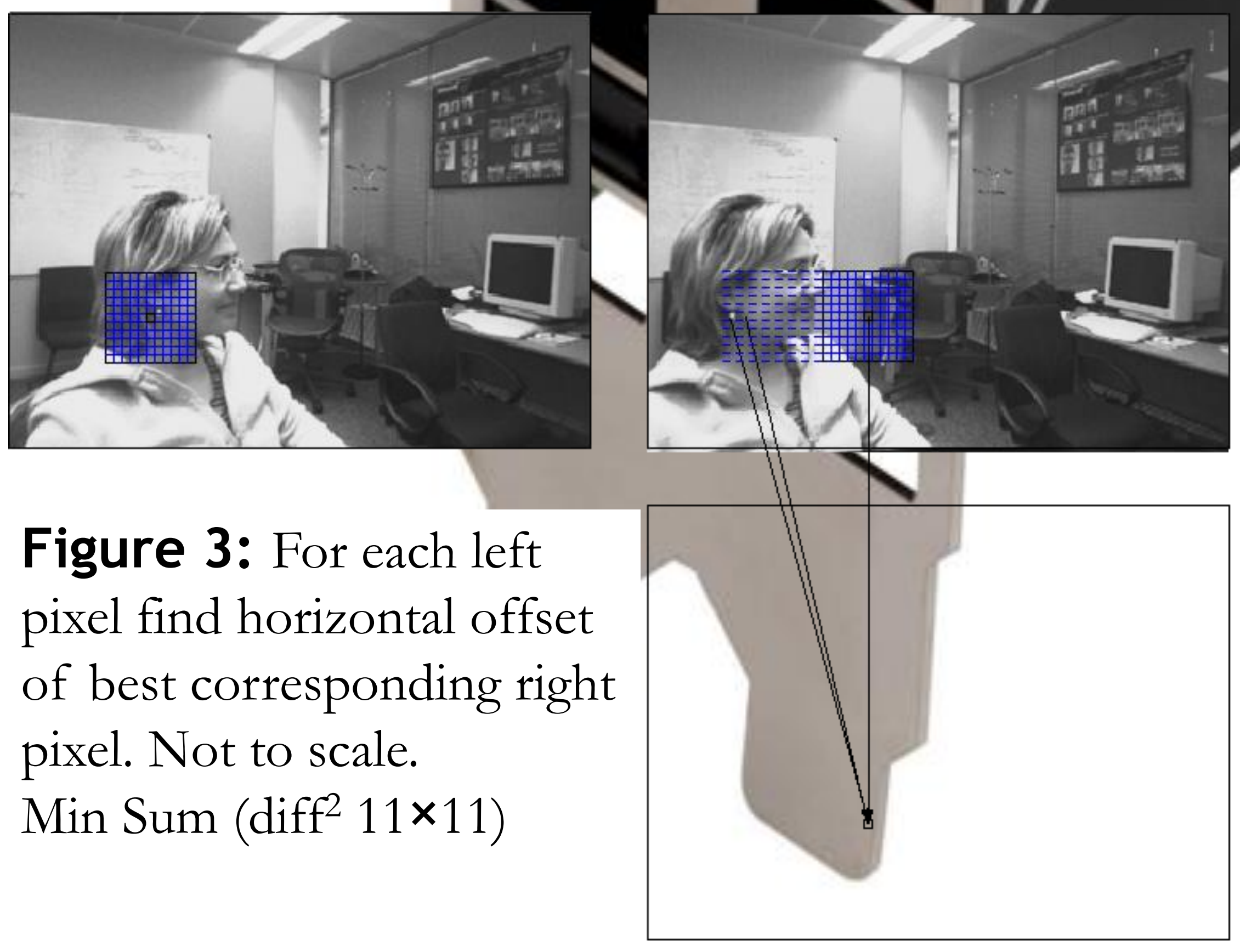


Figure 3: For each left pixel find horizontal offset of best corresponding right pixel. Not to scale.
Min Sum (diff^2 11×11)

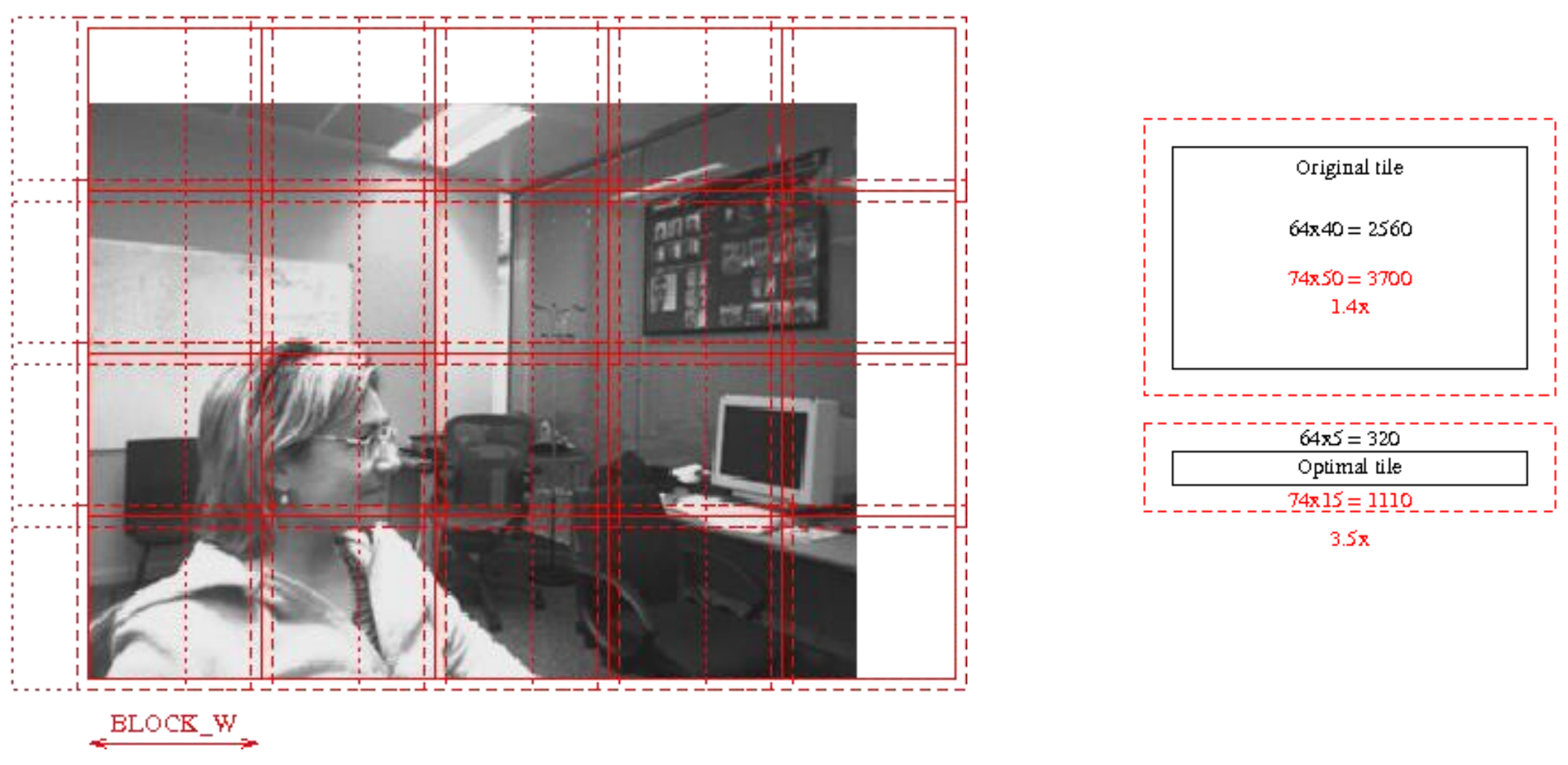


Figure 4: Images split into tiles. Each run in parallel Smaller tiles gives more parallelism but halos mean fastest size processes 2.4x more pixels.

5 GP Evolving Patches to CUDA

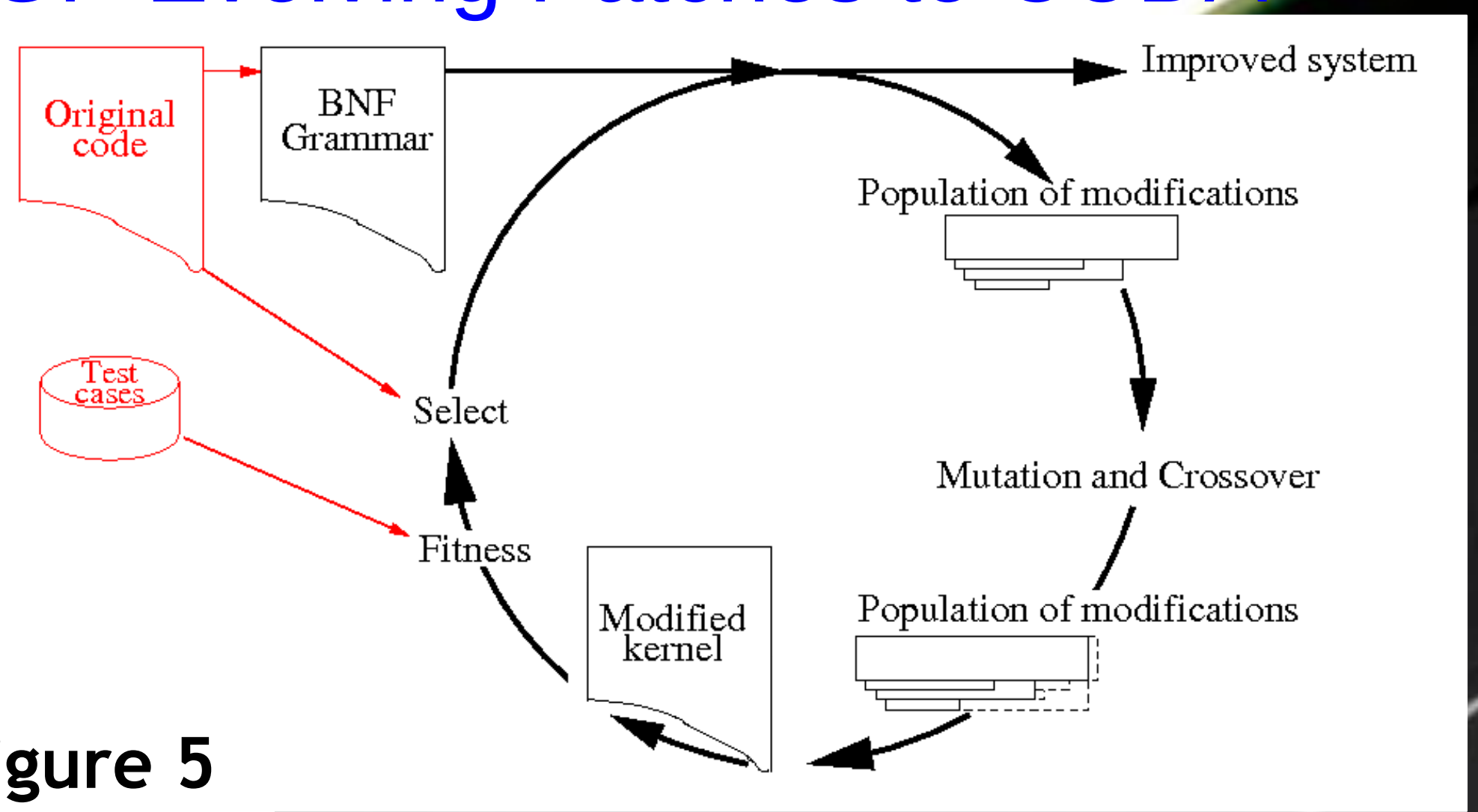


Figure 5

Figure 6: 12 Evolvable configuration macros

Name	Default	Options
Cache preference	None	None, Shared, L1, Equal
-Xptxas –dlcm		ca, cg, cs, cv
OUT_TYPE	float	float, int, short int, unsigned char
STORE_disparityPixel	GLOBAL	GLOBAL, SHARED, LOCAL
STORE_disparityMinSSD	GLOBAL	GLOBAL, SHARED, LOCAL
DPER	Disabled	0,1
XHALO	Disabled	0,1
__mul24	__mul24	__mul24 *
GPtexturereadmode	NormalizedFloat	NormalizedFloat, ElementType, no Textures
texturefilterMode	Linear	Linear, Point
textureaddressMode		Clamp, Mirror, Wrap
texturenormalized		0,1

Fixed mutation	Tesla T10		Tesla C2050		GTX 580		Tesla K20c	
Cache	None	62	L1	52	L1	66	None	48
-Xptxas -dlcm	ca	84	not used	50	cg	42	not used	32
OUT_TYPE	`float`	100	`float`	74	`float`	76	`float`	48
STORE_disparityPixel	LOCAL	100	LOCAL	100	LOCAL	76	GLOBAL	70
STORE_disparityMinSSD	SHARED	100	SHARED	100	SHARED	56	SHARED	76
DPER	disabled	100	disabled	100	used	100	used	100
XHALO	disabled	100	used	100	used	100	used	100
__mul24(a,b)	__mul24	100	*	100	*	70	__mul24	98
GPtexturereadmode	Normalized	100	Normalized	100	Normalized	100	Normalized	100
texturefilterMode	Linear	100	Linear	100	Linear	100	Linear	100
texturenormalized	default	82	default	80	default	72	default	72
textureaddressMode	Wrap	40	Clamp	66	Mirror	42	Mirror	48

6 Six Tailored Kernels

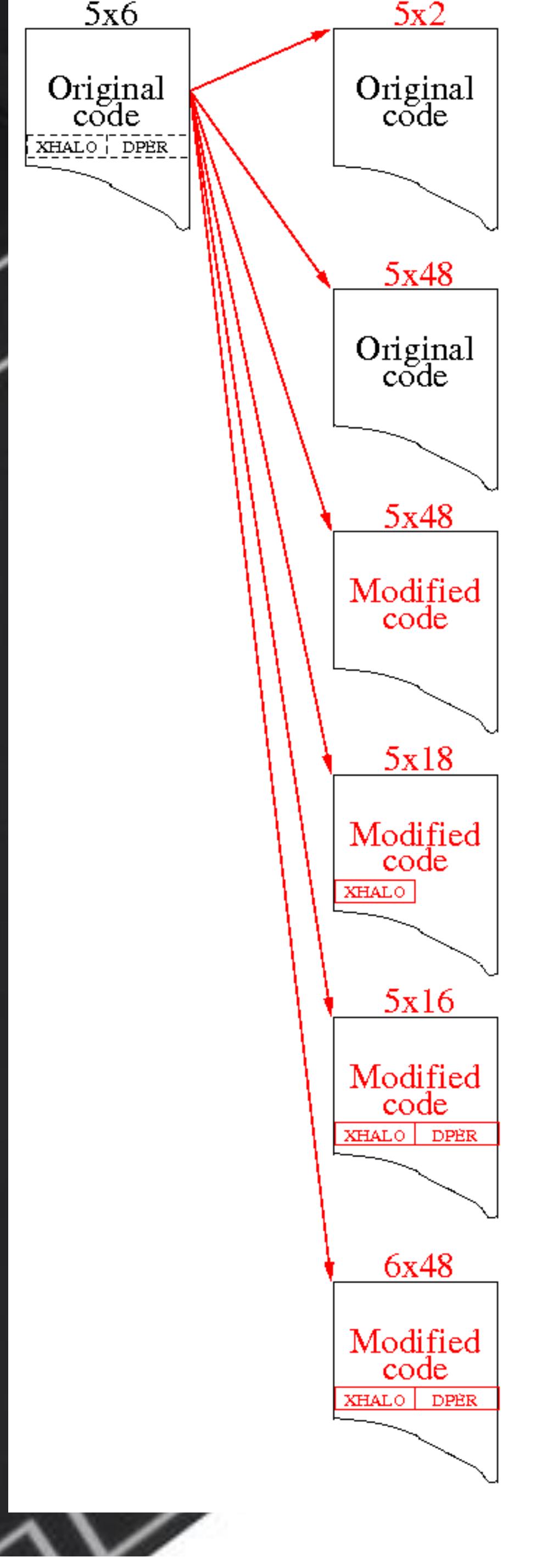


Figure 7: Tuned kernels for each of six GPUs. For K20c 320x240 image pairs split into 6x48=288 tiles processed in parallel. Speedup in red.

7 GP Patches to K20c Kernel

Non-default configuration	value
ROWSperTHREAD	5
BLOCK_W	64
DPER	enabled
dperblock	2
XHALO	enabled
STORE_disparityMinSSD	SHARED
STORE_disparityPixel	SHARED
Remove CUDA code	**New CUDA code**
`int * __restrict__ disparityMinSSD,`	
`volatile extern __attribute__ ((shared)) int col_ssd[];`	`extern __attribute__ ((shared)) int col_ssd[];`
`volatile int* const reduce_ssd = &col_ssd[(64 )*2 -64];`	`int* const reduce_ssd = &col_ssd[(64 )*2 -64];`
	`#pragma unroll 11`
`if(X < width && Y < height)`	`if(dblockIdx==0)`
`__syncthreads();`	
	`#pragma unroll 3`

More examples: Evolving a CUDA Kernel from an nVidia Template, W.B. Langdon and M. Harman. In WCCI 2010, pages 2376-2383, 18-23 July, Barcelona

8 Software speedup up to 6.8 times

- Considerable software speedups possible in addition to hardware speedup by both tuning parameters and adjusting CUDA kernel code when moving to new GPU. (Best 43x.)
- Software speedup up to 6.8x (median 4.0).
- In future optimise multiple properties, e.g. MPI bandwidth, memory, battery life (3D VR on phones).

Acknowledgments: Joe Stam nVidia Stereocamera (SourceForge). Microsoft I2I images. C2050s and K20c given by nVidia.