A Performance Evaluation Of Multimedia Kernels Using AltiVec Streaming SIMD Extensions.

Julien Sebot (sebot@lri.fr)
Université Paris-Sud,
Laboratoire de Recherche en Informatique, Bâtiment 490,
F-91405 Orsay Cedex, France
Phone: 0 (0 33) 1 69 14 42 22; fax: 0 (0 33) 1 69 15 65 86

Abstract

This paper aims to provide an understanding of performance of multimedia applications that use floating-point computations on recent general-purpose microprocessors using streaming SIMD ISA extensions. We used 8 benchmarks to study the impact of these extensions on general application performance and identify the eventual bottlenecks introduced.

1. Introduction

The introduction of floating point SIMD instructions in general purpose microprocessors is quite recent, and there isn't currently a lot of studies on the subject. On the other hand integer SIMD workloads have already been well studied. Our purpose is to provide a better understanding of performance improvements in multimedia applications due to these ISA extensions. We have used AltiVec, introduced by Apple and Motorola in their newest microprocessor: the PowerPC G4. AltiVec works with 128 bits vectors and allows both integer and single precision vector processing.

The eight micro-kernels we use as benchmarks have been extracted from commonly used multimedia applications. We will compare the original optimised C kernel with an AltiVec optimised one. These eight micro-kernels are FIR/IIR filter, min/max, vertices transformation and projection, normals transformation and normalization, backface culling, Phong lighting, DAXPY and generic matrix-matrix multiplication. Diverse audio filtering and speech compression are based on FIR/IIR filter, the Viterbi algorithm used in speech recognition uses the minmax kernel. Mesa, a 3-D graphics library with an API which is very similar to that of OpenGL uses 3-D geometry transformations and lighting to render 3D polygonal scenes. Vertices transformation, projection, perspective correction, normals transformation and renormalization, backface culling and Phong lighting are the critical parts of geometry transformations. Some functions like DAXPY, the sum of two floating-point arrays and generic matrix multiplication have been studied too because of their intensive use in numerical algorithms. All these kernels have been hand tuned using well-known techniques like loop unrolling and software pipelining.

For our performance evaluation, we used tools provided by Apple and Motorola. The AltiVec kernels are not fully written in assembly. A set of C functions that work on variables instead of registers and that maps on AltiVec instructions allows easiest programming, and fine tuning. We generate execution trace with pitsTT6 tool by linking the kernel with pitsTT6 library and calling the functions to start tracing and to end it. The G4 simulator gives a lot of statistics on the execution of the tt6 trace. All the results provided come from the simulation of a G4 processor executing the previously extracted TT6 execution trace.

The speedup obtained with AltiVec for these benchmarks range from two (FIR/IIR) to ten (64x64 generic matrix multiplication). The final speedup for the 3D geometry pipeline is 4 when using two dynamics light sources i.e. two passes into the Phong lighting step which is the most accelerated part of the 3D kernel. The use of streaming AltiVec prefetch improves the global performances by a factor of 1.1 times to 2 times.

The memory behaviour of our micro-kernels is characterized by streaming data access. Data are quite never reused and the first level cache size has no impact on our micro-kernels performances when over 4Kb. Most of the benchmarks become memory bound when using AltiVec, but the use of prefetch reverts this tendency. We also study the impact of increasing memory bandwidth on the final performance of these benchmarks. The performance improvement using a doubled memory bandwidth at a constant latency is for most of the benchmarks around 20%. For the 3D benchmarks we also evaluated the importance of data organisation on the performance improvement due to the use of vector processing. We found the same results as Intel on its SSE streaming media extensions. On all the benchmarks we explain the reason for the performance improvement, and for over 4x improvement using 4 element wide SIMD processing.

To summarize the floating point benchmarks we have studied are improved by the use of AltiVec streaming SIMD extensions from 2 times to 10 times. The impact of prefetch ranges from 1.1 times to 2 times performance enhancements. Improving main memory bandwidth without lowering latency has a noticeable impact on final performance due to streaming software directed prefetch.

2. Methodology

2.1. AltiVec and G4 Architecture

AltiVec is a Single Instruction Multiple Data (SIMD) machine comprised of:

- 32 x 128-bit "vector" registers
- two fully pipelined processing unit: a vector permute
- and an ALU unit.
- 170 instructions
- AltiVec registers can hold either 16 8-bits Integers, 8 16-bits integers, 4 32-bits Integers than 4 IEEE-754 single precision floats. AltiVec provide streaming prefetch instructions.

The PowerPC G4 is the first processor to implement the AltiVec instruction set. It has a four stages pipeline (Fetch, Dispatch, Execution, Completion).

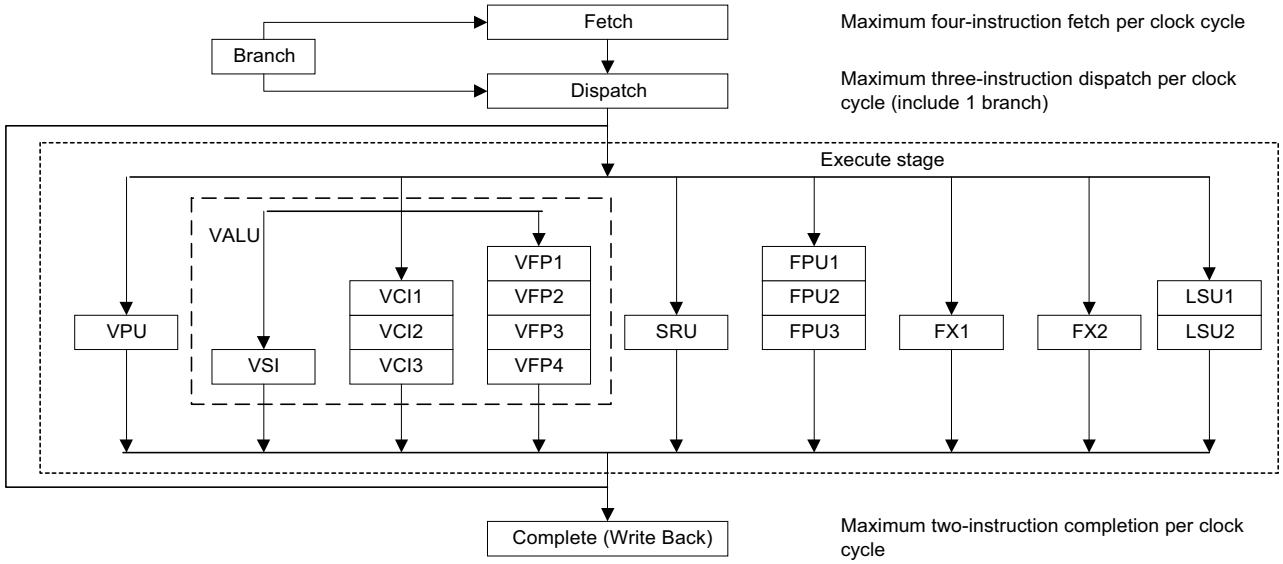


Figure 1: PowerPC G4 pipeline diagram

The simulated PowerPC G4 runs at 400Mhz, has a 32Kb first level data cache, 512KB L2 at half the processor's speed and a 100Mhz system bus with SDRAM. There is more information about the PowerPC G4 in [PPC7400] and about the G4 simulator in [G4SIM].

2.2. Methodology

The AltiVec programming interface

We have used Apple MrC compilator through a Metrowerks Codewarrior pro 4 plug-in. MrC allows easy AltiVec programming through some extensions which enable the use of AltiVec functions in a C program. The C programming model for AltiVec used in MrC is designed to look like function calls, but each of this function call represents and generates an AltiVec instruction. Variables are defined using C types that match AltiVec data types and variables declared using these types are all 16-byte aligned. The sample following program performs the sum of two arrays of N float (N multiple of 4).

```
vector float A[N], B[N], C[N];
int i;
for ( i=0 ; i<N ; i++ )
    A[i] = vec_add( B[i] , C[i] );
```

Trace extraction and Simulation environment

All tools we have used have been written by Apple [AAV99]. We have used pittsTT6 v1.4 trace generator, which works as a shared library. A call to the *startTrace* function launch the trace generation, and a call to *stopTrace* stops it. By example, here is the code for the sum of two single precision float arrays of 4 x N elements, with pittsTT6 tracing enabled:

```
startTrace("av_function.tt6");
av_function();
stopTrace();
```

The AltiVec Emulator extension has been used to allow us to run the AltiVec programs. At runtime, the TT6 execution trace is generated and written to disk. After that this trace is used as input for the G4 simulator (v1.1.2). The G4 simulator has all the characteristics of a PowerPC G4.

2.3. Workloads

We have written the following kernels: sum of two arrays, min/max of an array with corresponding index, FIR/IIR, 3D geometry transformations and lighting, DAXPY and generic matrix multiplication. Each of these kernels use single precision floats and will be used as a benchmark. The data set we have used are small for reasonable trace extraction time and simulation time. The kernels we have used doesn't have much temporal locality, and when they present some, the datasets we use are all fitting into the first level cache. Minmax is used by the Viterbi Algorithm for speech recognition, FIR and IIR are used for audio processing.

3. Performances

We have evaluated the performances of our eight benchmarks using the Apple G4 simulator. The simulator gives statistics about the different pipeline stall reasons, and the hit/miss rate into the two levels of cache. The reasons for pipeline stall are lack of rename register, instruction buffer empty, reorder buffer full and unit busy. The rest of the time is spent into the executions unit (without stalls). The stalls due to unit busy are hardware dependency stalls, and the reorder buffer full one are due to data dependencies and memory hierarchy. The theoretical speedups of all our benchmarks are 4 because they all use single precision floating point values, and because AltiVec processes these data 4 by 4.

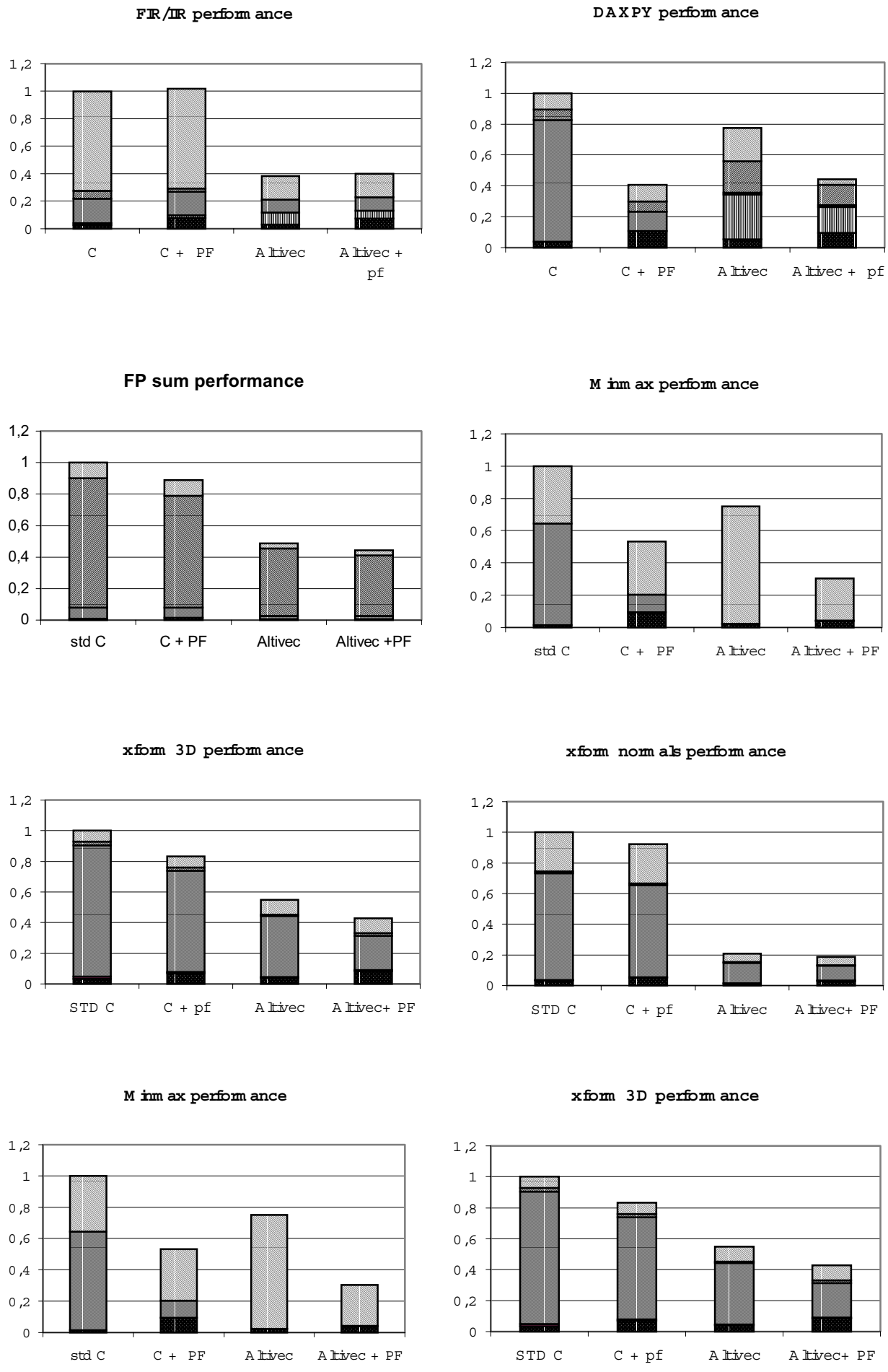


Figure 2: Benchmark results, speedup of all versions compared to the standard C one.

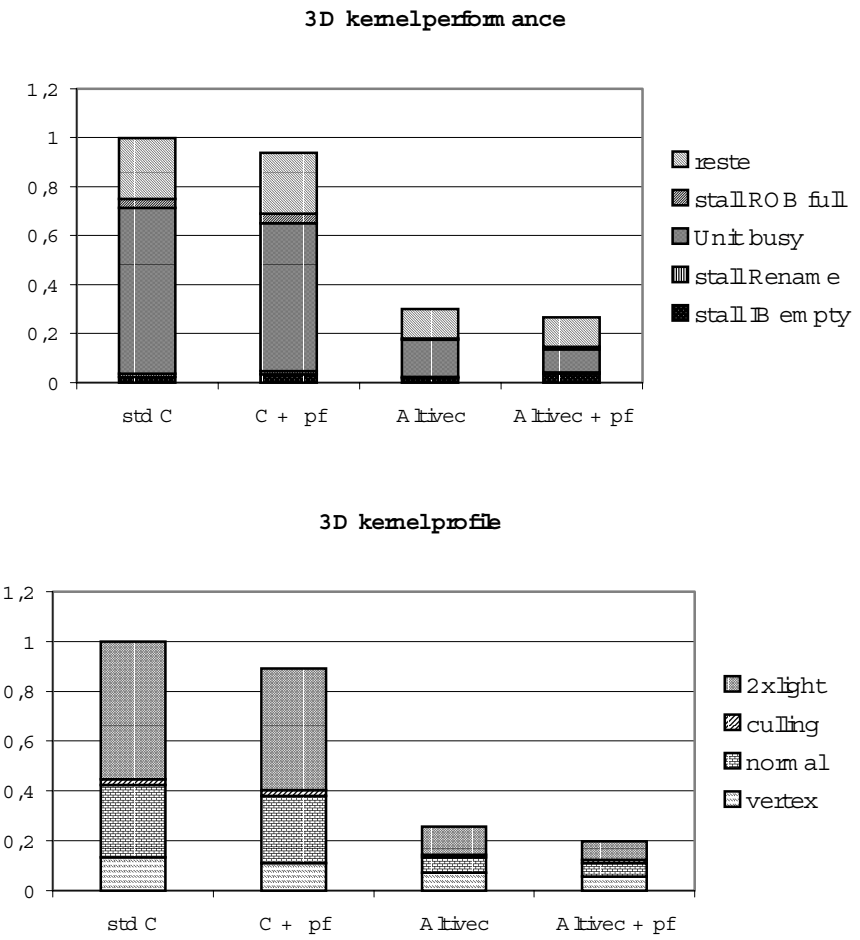
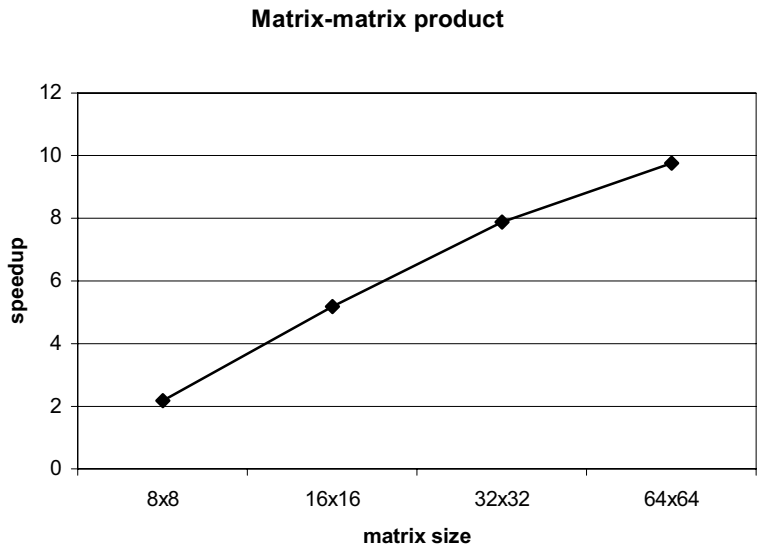


Figure 3: Full graphic pipeline performances, speedup over the standard C version.

Figure 4: Matrix-matrix products speedup (for matrix size that fit into the L1 Cache)



The global performance of our 3D pipeline is 1,7M vertice and normals per second for the C version and 6,2M for the AltiVec Version with prefetch on a 400 Mhz G4 running with PC100 SDRAM. Before, there was just only dedicated hardware that could achieve this kind of performance.

3.1. Impact of AltiVec introduction

- Reduction of instructions in functional units
- Reduction of branch instructions
- Reduction of memory instructions
- Relative reorder buffer size increase
- Data reorganisation done parallely with computations

3.2. The impact of streaming prefetch

On figure 2 and 3 we see that AltiVec streaming SIMD prefetch improves performances from 3%(FIR) to 120% (Min/Max). The benchmark that benefit much from the AltiVec streaming prefetch are those which don't perform a lot of computations. These benchmarks are essentially limited by the main memory latency, and by the main memory bandwidth when the streaming prefetch is used. We present into Table 1 the performance increase due to AltiVec streaming prefetch instructions. The use of prefetch generally increase more the performances of AltiVec programs than standards programs because the AltiVec programs spend relatively more time into memory operations than the others, the reason for that is that the AltiVec computations takes less time than standard non-SIMD computations. DAXPY C with prefetch is as fast as DAXPY AltiVec with prefetch because this benchmark is limited by memory bandwidth when using streaming prefetch. Finally that fact traduces into a higher performance increase for DAXPY C than DAXPY AltiVec.

Prefetch version speedup over:	Sum of 2 fp Arrays	Min/Max	DAXPY	FIR/IIR	3D kernel
C version	1.10	1.77	2.47	1.00	1.05
AltiVec Version	1.10	2.22	1.22	1.02	1.22

Table 1: Impact of AltiVec Streaming prefetch on AltiVec Kernel versus C kernels (speedups).

Finally the streaming prefetch tends to make the micro-benchmarks memory bandwidth bound, we will verify it by simulating higher memory bandwidth.

3.2. The impact of increasing memory bandwidth

We have evaluated the impact of main memory bandwidth on the performance of our micro-benchmarks. This kind of study has been done on multimedia benchmarks with a more accurate simulation of the RAM properties into [RAMPERF]. In our study the model used for the memory is the standard SDRAM because it is the only model that has been implemented into Motorola G4 simulator. The results are presented on Figure 5. We have measured the slowdown for an architecture using a memory bandwidth of half the bandwidth of standard PC100 SDRAM (*0.5), and the speedups for doubled and quadrupled bandwidth. Currently a doubled bandwidth could be simply achieved by using DDRSDRAM [DDRMIC][DDRSAM] and in the next months both DRDRAM [RBUS] and DDRSDRAM will be able to provide a quadrupled bandwidth compared to PC100 SDRAM, without increasing the latency for the DDRSDRAM.

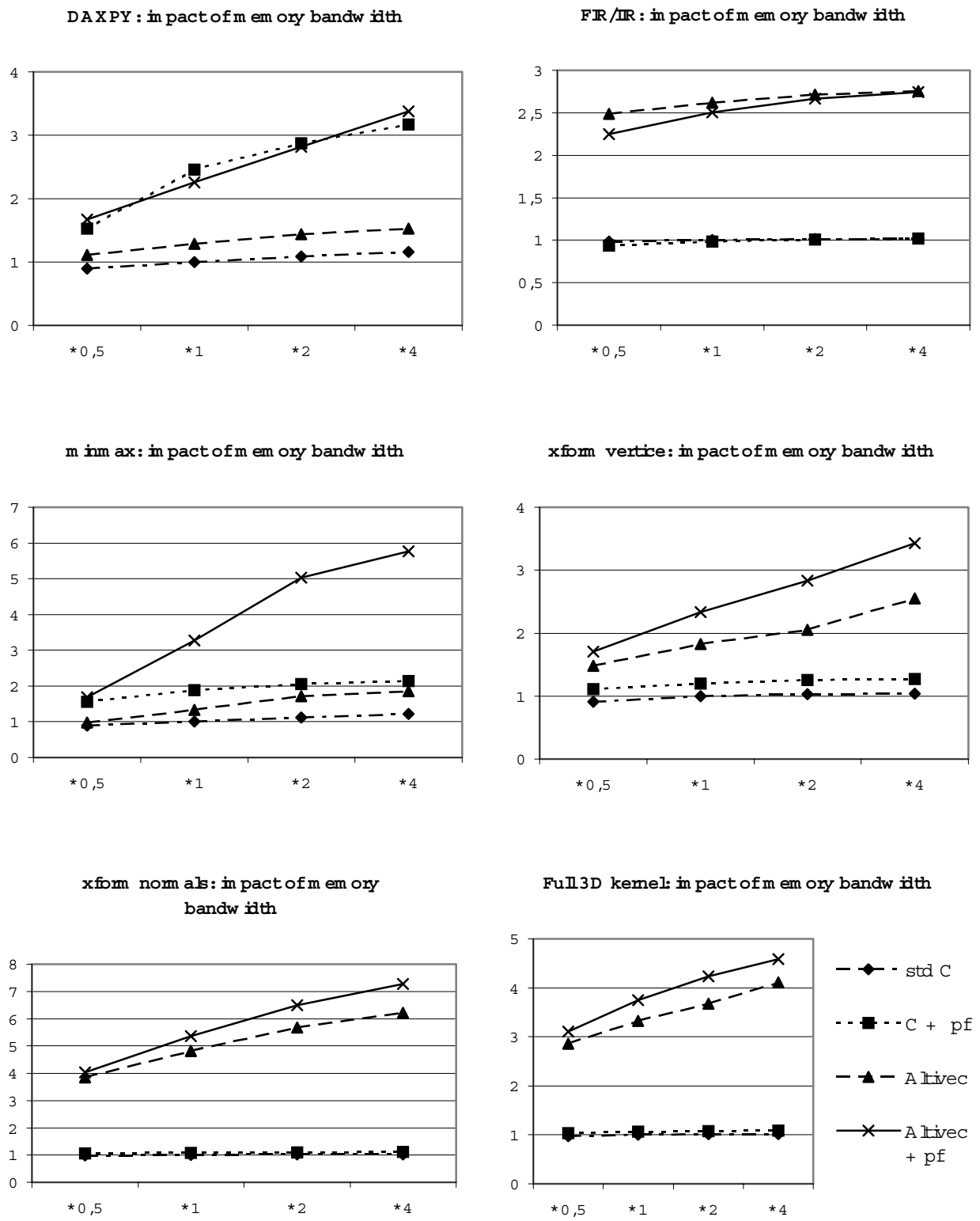


Figure 5: Impact of the increase of main memory bandwidth on some benchmarks in term of speedups. The reference is a speed of 1 for the C version using standard 100Mhz SDRAM.

Increasing the memory bandwidth has quite no effects on standard C versions of our micro-benchmarks. On the C+ prefetch version it has an impact only for the DAXPY bench which is almost limited by memory bandwidth when prefetching is activated. On the other benchmarks we can see great improvements on the AltiVec and AltiVec+ prefetch versions. It has more impact on the AltiVec + prefetch version because the use of prefetch limits the impact of memory latency on the benchmarks, and by this way these programs are essentially memory bandwidth bound. The performance increase due to a quadrupled bandwidth ranges from 10% (FIRIIR) to 75%(minmax) for Altivec with prefetch versions, and the final performance increase over the C version with Altivec and prefetch enabled ranges from a speedup of 2.7(FIR/IIR) to 4.8(minmax).

4. Conclusion

In this paper we have presented the performance improvements on eight benchmarks, achived with the help of AltiVec, PowerPC G4 new streaming SIMD extensions. We have also studied the impact of the use of AltiVec streaming prefetch instructions. Finally the speedups achieved with AltiVec and prefetching range from 2 to 10. We have analyzed the reasons for these performance improvements, including over 4 speedups. Finaly we have verified that the use of streaming prefetch and AltiVec SIMD extensions tends to make the benchmarks only memory bandwidth bound instead of computation bound as it was initially.

Bibliography

[PPC7400] Motorola, MPC7400 RISC Microprocessor Hardware Specification, September 1999.

[HMM86] L. R. Rabiner and B. H. Juang, An Introduction to Hidden Markov Models, In *IEEE ASSP Magazine*, page 5, January 1986.

[SPR95] R. A. Quinnel, Speech Recognition: No Longer a Dream but Still a Challenge, In *EDN*, pages 41 to 46, January 1995.

[FIR99] Intel, FIR and IIR Filtering using Streaming SIMD Extensions, 1999.

[MESA] Brian Paul, The Mesa 3D-Graphic Library, *http://www.mesa3d.org*, January 1999.

[AAV99] Apple, Apple's AltiVec Home Page, *http://developer.apple.com/hardware/AltiVec/* , May 1999.

[AVMRC] AltiVec Support in MrC[pp],June 998. *http://developper.apple.com/dev/tools/compilers/docs/AltiVec_support.pdf*.

[AVMM] H. Nguyen and L. K. John, Exploiting SIMD Parallelism in DSP and Multimedia Algorithm Using the AltiVec Technology, *ICS99*.

[AART] Alan Watt and Mark Watt, Advanced Animation and Rendering Techniques, *Addison Wesley*, ACM press 1992.

[INTECHJ99] Intel, Streaming SIMD Extensions – 3D Transformations, in *AP597*, January 1999.

[VISPERF] P. Ranganathan, S. Adve , N. Jouppi , Performance of Image Processing with General-Purpose Processors and Media ISA Extensions, *ISCA 26,* may 1999.

[RBUS] R. Crisp. "Direct Rambus technology: The new main memory standard." *IEEE Micr*o, vol. 17, no. 6, pp. 18–28, November 1997.

[RAMPERF] Vinodh Cuppu, Bruce Jacob Brian Davis, Trevor Mudge, A Performance Comparison of Contemporary DRAM Architectures, *ISCA 26* may 1999.

[DDRSAM] Samsung semiconductor ,Double Data rate synchronous DRAM *http://www.intl.samsungsemi.com/Memory/DRAM/Next_Generation/DDR_SDRAM/DDR.htm,* 1999

[DDRMIC] Micron, DDR SDRAM features and options, *http://www.micron.com/mti/msp/html/ddrsdramfeat.html*, 1999.