

# Re: That's not the Answer to Your Question

---

*Source:* <http://sci.tech--archive.net/Archive/sci.image.processing/2007-04/msg00156.html>

---

- *From:* Martin Brown <|||newspam|||@nezumi.demon.co.uk>
  - *Date:* 17 Apr 2007 02:11:57 -0700
- 

On Apr 16, 7:43 pm, "aruzinsky" <aruzin...@xxxxxxxxxxxxxxxxx@cathexis.com> wrote:

"The lack of appreciation of the essential factors involved in block processing is entirely yours."

Absolutely, and that lack of appreciation is warranted if block processing is unnecessary to solve the OP's speed problem. And, it is you, not I, who is remiss for not trying to determining the exact cause of the OP's speed problem before pontificating on block processing.

You are blissfully ignorant of the technical issues and apparently unwilling to learn.

The whole problem of the OPs speed with the long stride length implicit in the vertical direction of access to his image array is entirely due to cache misses. The performance difference on my P4 for optimised image code ranges between 20–30x faster than naive code accessing a 1024 square 8bit image using the wrong direction depending on the algorithm, blocksize and level of optimisation. Don't take my word for it do the experiment!!!

Meditate on this:

OP: "as processing pixel in a vertical fashion was *\_very\_* slow (0.31 vs. 0.009 for the horizontal)."

And that is about right for naive code on a large array that forces a read cache miss delay on every access. There might be something else wrong but solving his memory access problems would get things a factor of 10–20 faster in the vertical direction.

Tight sequential access to local blocks of memory in level 1 cache as far as possible and then in level 2 *\*is\** the solution. The OP at least knew enough to ask the right question. You still don't seem to understand the problem.

## Re: That's not the Answer to Your Question

How is it that I am the only one who suspects an outstandingly dumb error in the OP's code?

I have never encountered anywhere near such a large disparity between horizontal and vertical processing speeds on a personal computer, even

Your lack of experience is showing. The outstandingly dumb error is naive obvious coding.

"And also annoyingly by modifying the header you break the threading on some newsreaders that don't handle followups too elegantly."

You should blame the newsreaders instead of using me as a scapegoat for your annoyance. You apparently missed two corrections to my post.

If you change the headers willy nilly you must expect some posts to get lost. I read via free access NNTP but since my ISP no longer carries Usenet I post via Google. Your messed up subject headers fool Google.

Very few optimising compilers are quite so dumb but there are better ways to cater for odd N."

I wouldn't know, because I haven't done a survey of compilers, but I suspect calculation in the loop might be part of the C or C++ language standard because the philosophy is to give the programmer control. I'll ask some programmers and get back to you.

Optimising compilers have been taking unnecessary stuff out of loops since the mid 70's. Most PC compilers caught up with mainframe optimising techniques in the late 80's and after that almost any loop invariant expression would be spotted by the compiler and moved outside the loop. This was sometimes dangerous if for example a function call to a random number generator was erroneously optimised out of a loop (because it wasn't declared volatile).

C provides too many ways for programmers to make life difficult for optimising compilers but that is a separate issue.

Regards,  
Martin Brown

.