

## Re: Locating Nearest Neighbors in space (fast)

---

*Source:* <http://coding.derkeiler.com/Archive/General/comp.programming/2005-09/msg00346.html>

---

- *From:* Jon Harrop <[usenet@xxxxxxxxxxxxxxxx](mailto:usenet@xxxxxxxxxxxxxxxx)>
  - *Date:* Fri, 09 Sep 2005 23:19:04 +0100
- 

KRK wrote:

> Does anyone know a fast algorithm to do this, preferably one that doesn't  
> require fancy knowledge of data structures etc since I'm not a  
> professional programmer.

The brute force algorithm that you described is  $O(n^2)$ , as you hinted. The first step to faster code is improving that asymptotic algorithmic complexity from  $O(n^2)$  to  $O(n \log n)$  or even  $O(n)$ .

Very similar problems arise in astrophysics (simulating stars) and in condensed matter physics (simulating atoms). However, the solutions are very different because these two applications have wildly different distributions of particle separations. In astrophysics, the stars can (and often do) cluster, so the nearest neighbour distribution is very wide. In condensed matter physics, atoms tend to bustle until they are "at arms length", so the nearest neighbour distance is very sharp.

If your distribution of nearest neighbour separations is heterogeneous (like stars) then I'd recommend an octree or k-D tree (adaptively subdividing space). If your distribution of nearest neighbour separations is very sharp (like atoms in a solid) then I'd recommend voxels (uniformly dividing space up into cubes).

You can do voxels easily enough in C/C++ but if you're going to use trees then I'd recommend switching to a more modern language, like OCaml or SML.

--

Dr Jon D Harrop, Flying Frog Consultancy  
<http://www.ffconsultancy.com>

---

- *References:*
  - ◆ ***Locating Nearest Neighbors in space (fast)***
    - ◇ *From:* KRK
- Prev by Date: ***Re: Balanced trees vs. B-trees***
- Next by Date: ***Re: Do all programming languages use files?***

Re: Locating Nearest Neighbors in space (fast)

- Previous by thread: ***Re: Locating Nearest Neighbors in space (fast)***
- Next by thread: ***Re: Locating Nearest Neighbors in space (fast)***
- Index(es):
  - ◆ ***Date***
  - ◆ ***Thread***