Accurate Distance Algorithms for n-Trees

Dagstuhl, 19.1.-24.1.03

Wolfram Luther
University of Duisburg-Essen

Overview

- Motivation
- Recent work
- An algorithm
- Convex hulls
- Outlook

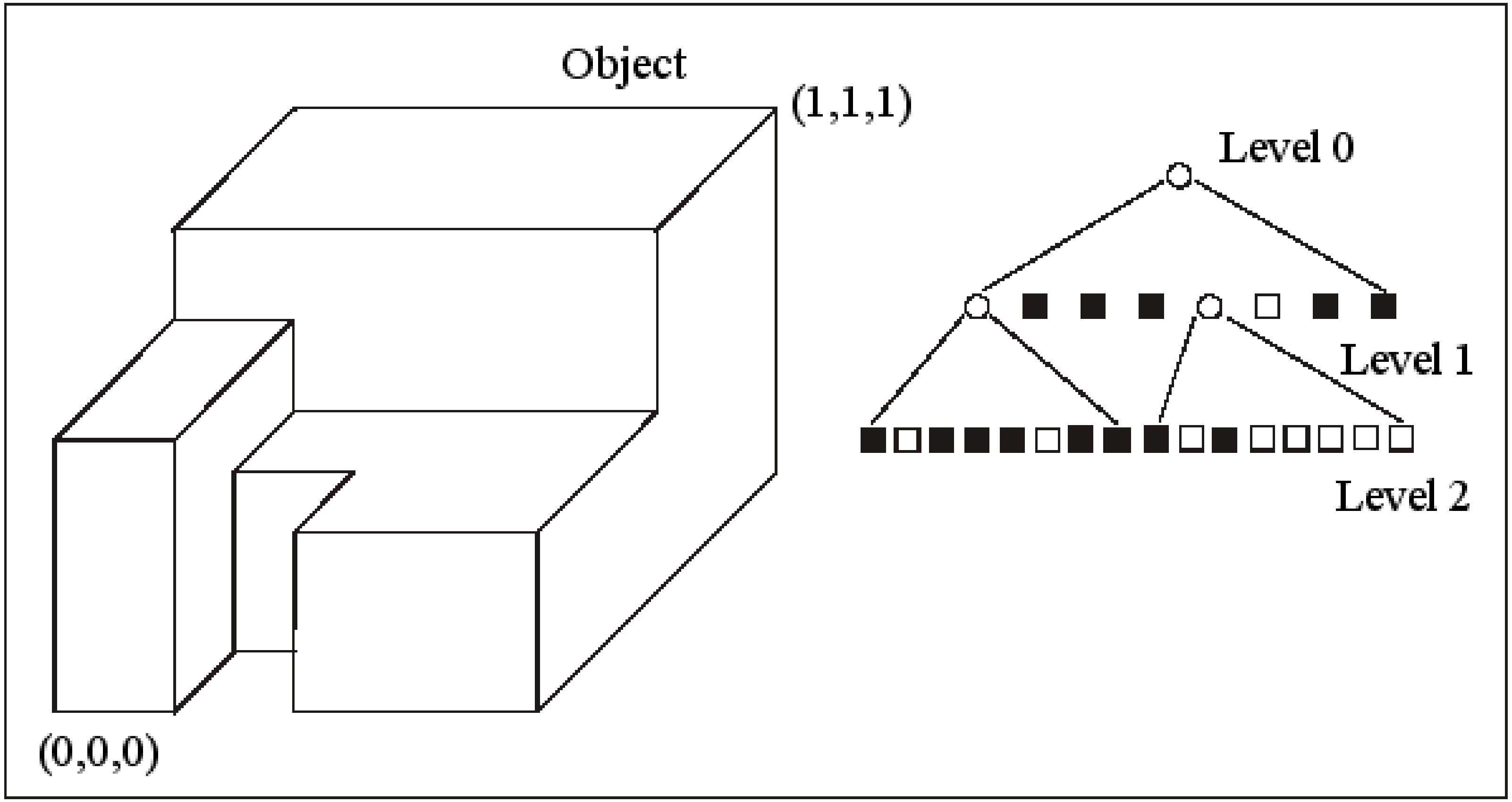
Motivation

- Modeling worlds with hierarchical data structures
- Use quadtrees, octrees and n-trees to represent objects
- Find distance and shortest paths between different objects
- Include extensions for moving objects

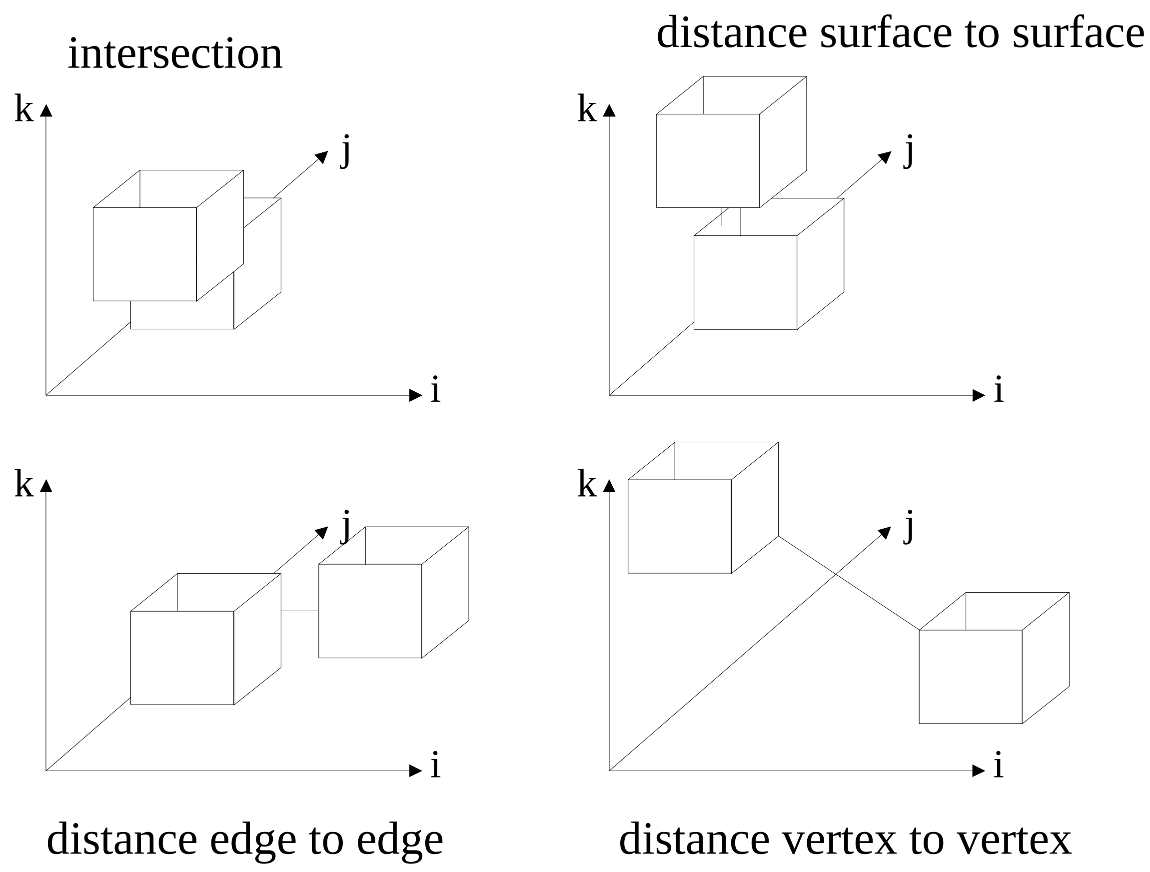
Recent work

- Project NVMM
- Accurate distance algorithms
  - Point to convex object
  - Convex to convex object
  - Point to non-convex object
  - Non-convex to non-convex object
  - Point to B-Spline surface
- Dynamic algorithms

Octrees



An algorithm



Distance between objects represented by octrees

First step:

- Establish a procedure $dist^2(Q_1,Q_2)$, Q_1,Q_2 rectilinear boxes, axis-aligned
- $Q_1=(X_1,X_2,X_3,h_1,h_2,h_3)=I_1\times I_2\times I_3$
- $Q_2=(Y_1,Y_2,Y_3,k_1,k_2,k_3)=J_1\times J_2\times J_3$

The following cases appear

- Intersection:
  - $I_1\cap J_1\neq\emptyset \wedge I_2\cap J_2\neq \emptyset \wedge I_3\cap J_3\neq \emptyset \Rightarrow dist=0$

Distance II

Surface to surface:

- $I_l \cap J_l \neq \emptyset \wedge I_m \cap J_m \neq \emptyset \wedge I_n \cap J_n = \emptyset \Rightarrow$

  $dist=(Y_n-X_n-h_n)^2$ resp. $dist=(X_n-Y_n-k_n)^2$

Edge to edge:

- $I_l \cap J_l \neq \emptyset \wedge I_m \cap J_m = \emptyset \wedge I_n \cap J_n = \emptyset \Rightarrow$

  $dist = (Y_m-X_m-h_m)^2 + (X_n-Y_n-k_n)^2$ etc.

Vertex to vertex:

- $I_l \cap J_l = \emptyset \wedge I_m \cap J_m = \emptyset \wedge I_n \cap J_n = \emptyset \Rightarrow$

  $dist = (Y_l-X_l-h_l)^2 + (Y_m-X_m-h_m)^2 + (X_n-Y_n-k_n)^2$ etc. depending on the positions of the boxes.

All results are accurate

Second step

Initialize the lists LS,LW,LG and the distance D=3,W=[0,0,0,0,0,0],S=[1,1,1,0,0,0]

/* LS contains actual black boxes, LW contains actual white boxes, LG contains gray boxes of the i-th level */

For all levels i=0,1,...,N

/* N depth of the octree */

{For all children Q of boxes pred(Q) $\in$ LG of size 2^{-i} in level i {

/* Update the list LG containing all gray boxes for the next level */

Second step cont'd

```
If Q=white then
    { Q→LW; For all T∈LS do
        if dist²(Q,T)<D then
                    {D:=dist²(Q,T);W:=Q;S:=T}}
else if Q=black then
    { Q→LS; For all T ∈ LW do
        if dist²(Q,T)<D then
                    { D:=dist²(Q,T);W:=T;S:=Q}}
  /* two or more different kinds of subboxes */
else
```

Second step cont'd

if Q=gray then Q$\rightarrow$LG}

For all T $\in$ LS,T$\neq$S do

  {for all Q $\in$ LG with wr or bwr calculate $dist^2(Q,T)$;

    if $dist^2 > D$ (for all Q with wr) and $dist^2 > 3 \cdot 2^{-2i-2}$

    (for all Q with bwr) then drop T in LS}

For all T $\in$ LW,T$\neq$W do

  {for all Q$\in$LG with sr or bwr calculate $dist^2(Q,T)$;

    if $dist^2 > D$ (for all Q with br) and $dist^2 > 3 \cdot 2^{-2i-2}$

    (for all Q with bwr) then drop T in LW}

  } /* We drop all irrelevant boxes */

}

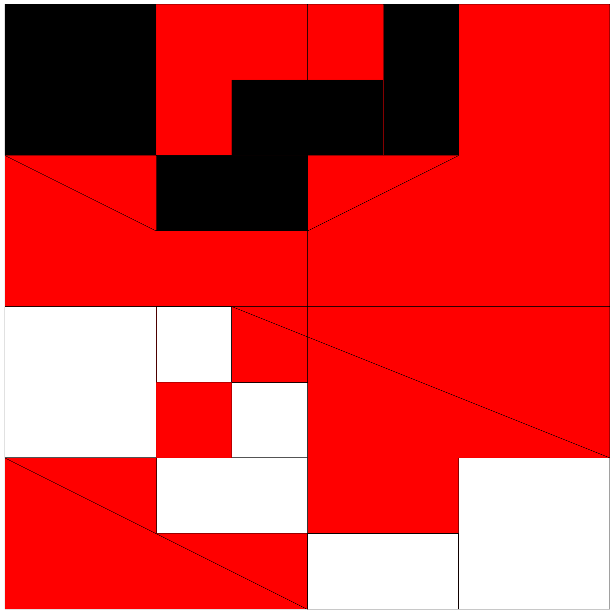
Remarks

- The algorithm can be modified to find all solutions
- The algorithm provides good upper and lower bounds
  - The temporary distance D is an upper bound, (use $3 \cdot 2^{-2i-2}$ if there is a bwr-box on level i)
  - Determine greatest level i with bwr-boxes. Replace on a level $j \geq i$ br-boxes by black boxes and wr-boxes by white boxes. Calculate D as a lower bound
- The algorithm works in higher dimensions

Remarks cont'd

- Use the underlying data structure
- On level i: $2^{6(i+1)}/3$ box comparisons
- Overall complexity $\leq 3 \cdot 2^{6(N+1)}$
- On the highest level tighter (convex) enclosures of the objects inside the boxes can be used to obtain better bounds for D

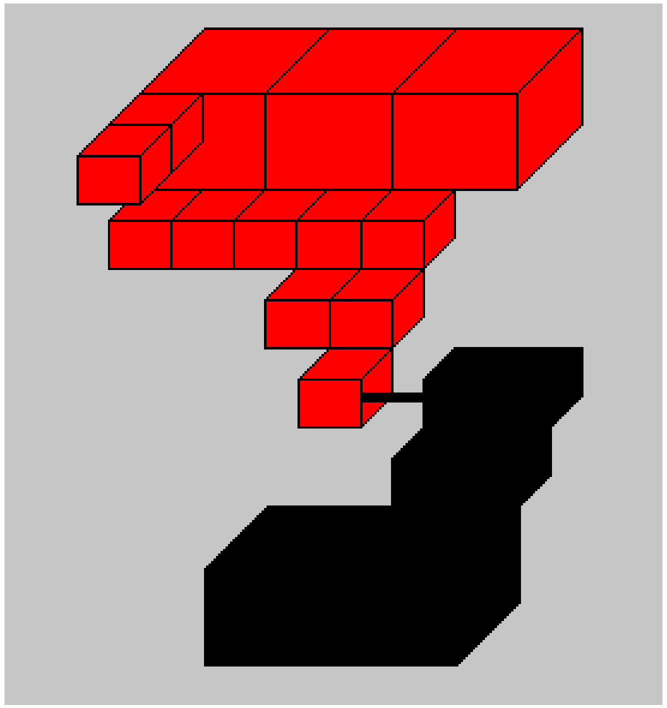
Examples



Level 3 quadtree
with two objects

The white box on the right-hand side is dropped.
The convex hull of the extreme vertices is shown.
The distance remains unchanged.

Examples cont‘d



Octree with two objects - level 3

The algorithm eliminates the boxes near the boundary x=0,1;y=0,1;z=0,1.

Convex hulls

- Split objects into convex parts
- Digitize the objects
- Consider extreme vertices
- Use digital convexity (d.c.)
  - Different rasterization models
  - The object is supported by planes through each box belonging to the boundary
- Construct convex hulls of d.c.-objects
- Algorithms for moving convex objects

Outlook

- Find near convex or locally convex analytic inclusions for objects modeled by octrees or n-trees
- Adapt existing accurate distance algorithms
- Consider moving objects