Structure Recognition using Geometric Hashing

By-

MinChi Hu

Cassian D'Cunha

Outline

- Introduction
- Geometric hashing (Two Dimensional).
- Geometric hashing (Three Dimensional).
- Test Results.
- Sample Output.

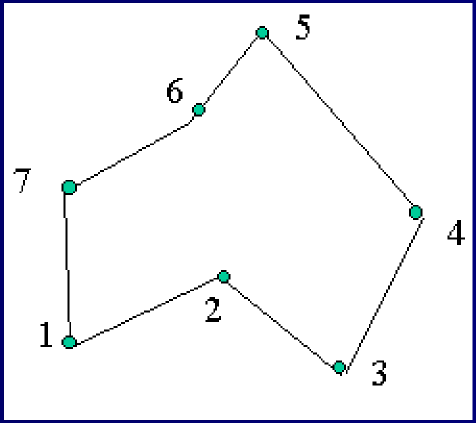
Introduction

- Geometric Hashing was originally developed for object recognition problems in Computer vision.
- Later found applications in other domains such as
  - Ligand - Protein or Protein - Protein docking in structural biology.
  - medical image registration.
  - Detection of defects in boundary of objects in CAD models

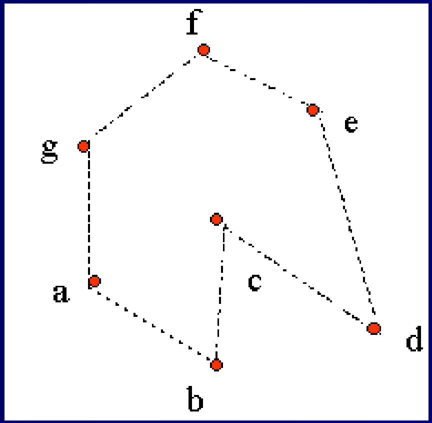
Idea

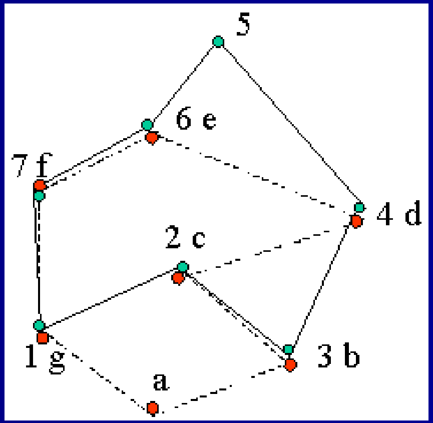
- Is there a rotated and translated subset of some structure which matches a subset of the observed structure, so that both the geometric and labeling constraints are satisfied?

Idea



We could place the two structures on each other and find the number of coincidence points.





2D Geometric Hashing – Example

- Input: given two Structures 'A' and 'B'
  - 'A' is put in, or is already present in database (m points).
  - 'B' is the query (n points).
- Output: Structure 'B' ***is*** similar to 'A'
  or it is ***not***.

Geometric Hashing – Two Phases

- Preprocessing
  - Each structure is processed and added to a database. i.e. geometric information encoded in a hash table.
- Detection
  - Features of structure to be detected is extracted and mapped to multiple entries in the hash table.

Definition of Terms

- Reference Frame: Coordinate system defined for both figures A and B

- Base Pair: Two points (since 2 D) that define the reference frame; one at the origin and the other along the positive x-axis.

Example

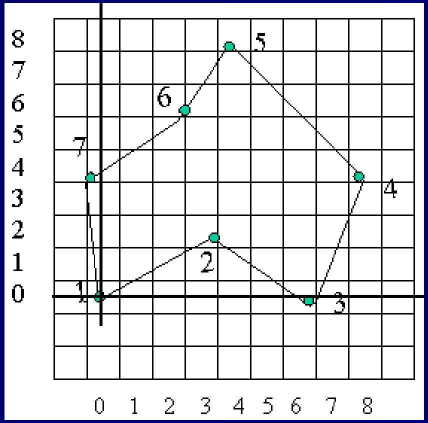


Fig 1

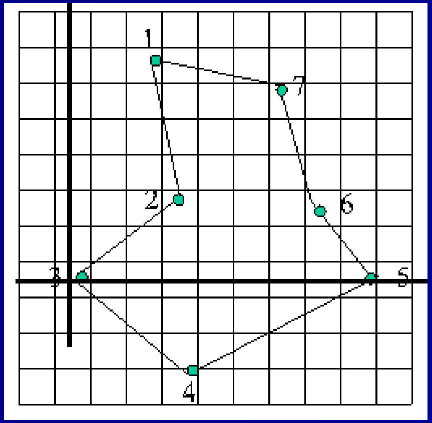


Fig 2

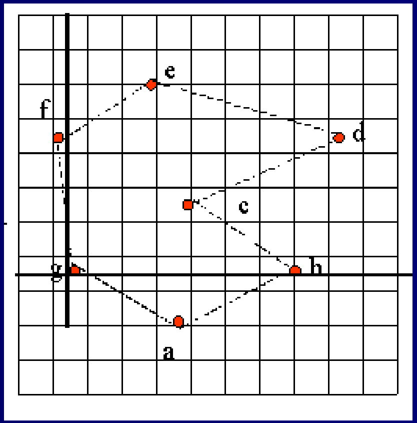


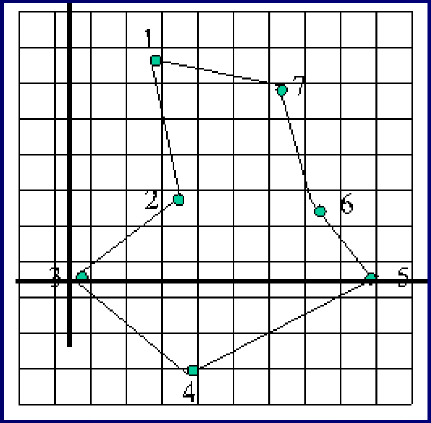
Fig 3

Fig 1 – Structure A; base pair [1, 3]

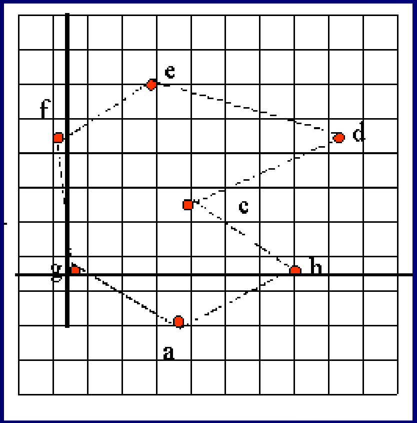
Fig 2 – Structure A; base pair [3, 5]

Fig 3 – Structure B (Query); base pair [g, h]

Example



A [3, 5]



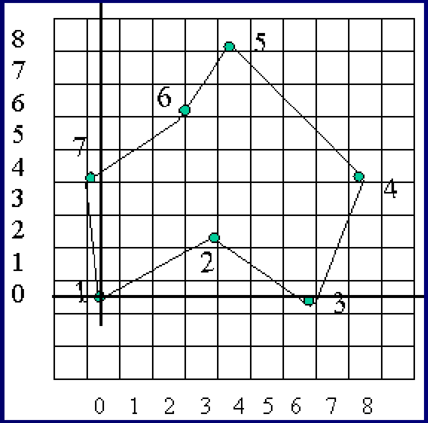
B [g, h]

There are 2 points that coincide from the two figures.

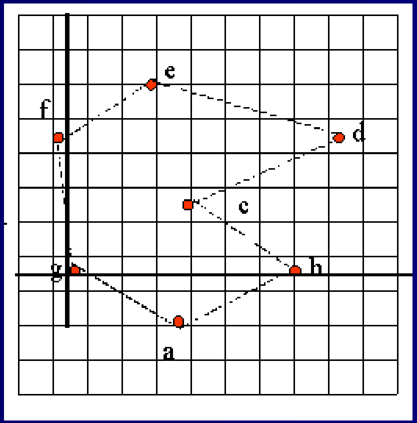
(3, g)

(2, c)

Example



A [1, 3]



B [g, h]

There are 5 points that coincide from the two figures.

(1, g)

(2, c)

(3, b)

(4, d)

(7, f)

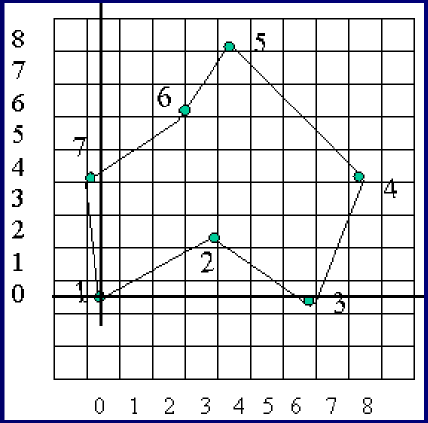
Note: point 6(A) and e (B) seem to coincide but have different x - y values.

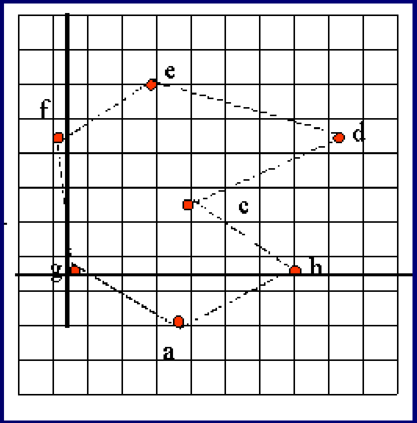
Remarks

- The number of coincident points depend on the *resolution* of the coordinate system and the *base pairs* used.

- Generally, all possible *base pairs* should be used. [i.e. $\frac{m(m-1)}{2} \cdot \frac{n(n-1)}{2}$]

Remarks



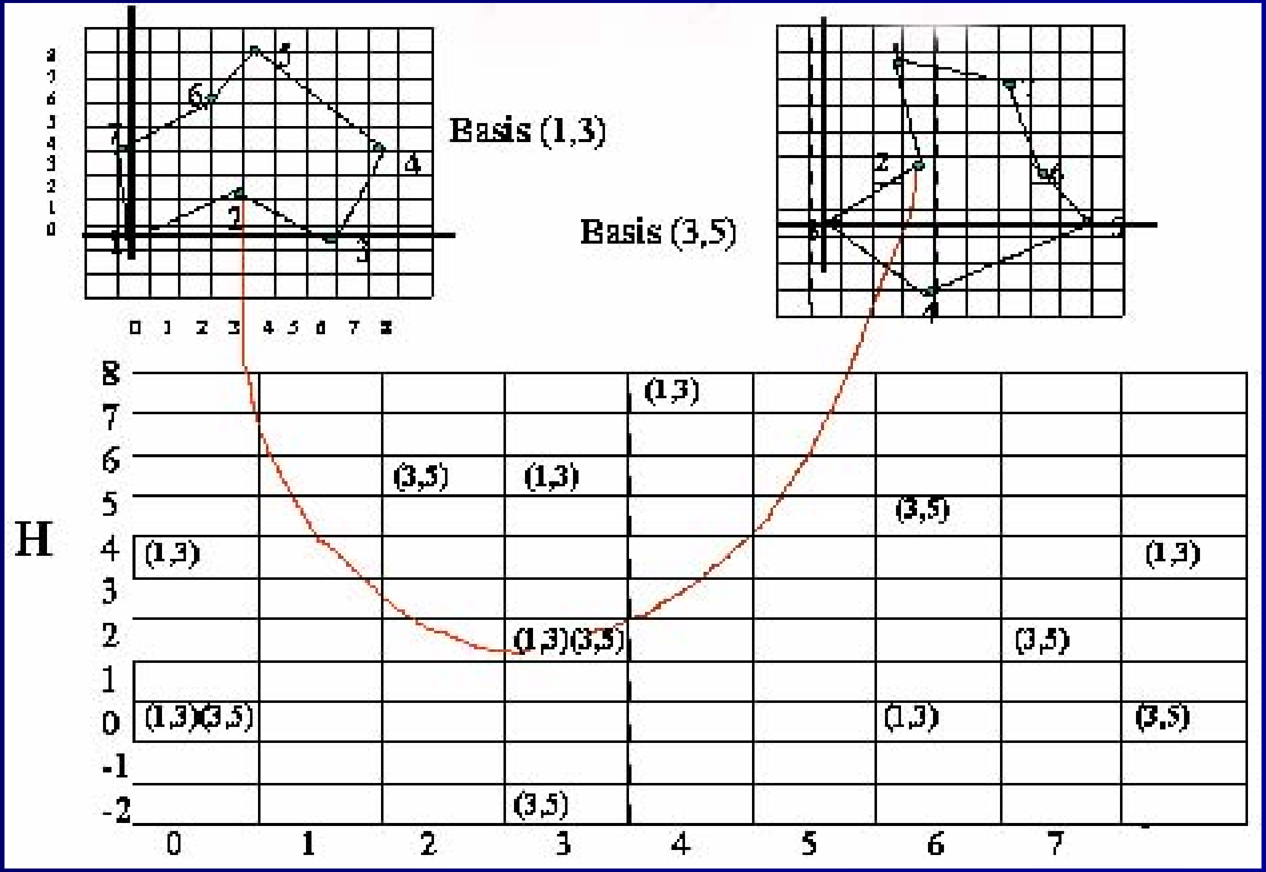


- Using all combinations will introduce redundancies.
  - e.g. if for base pairs $[A_k, A_l]$ and $[B_m, B_n]$ coincidence points are (A_s, B_p) and (A_q, B_r), then is likely that the base pairs $[A_s, A_q]$ and $[B_p, B_r]$ will result in the same coincidence points.

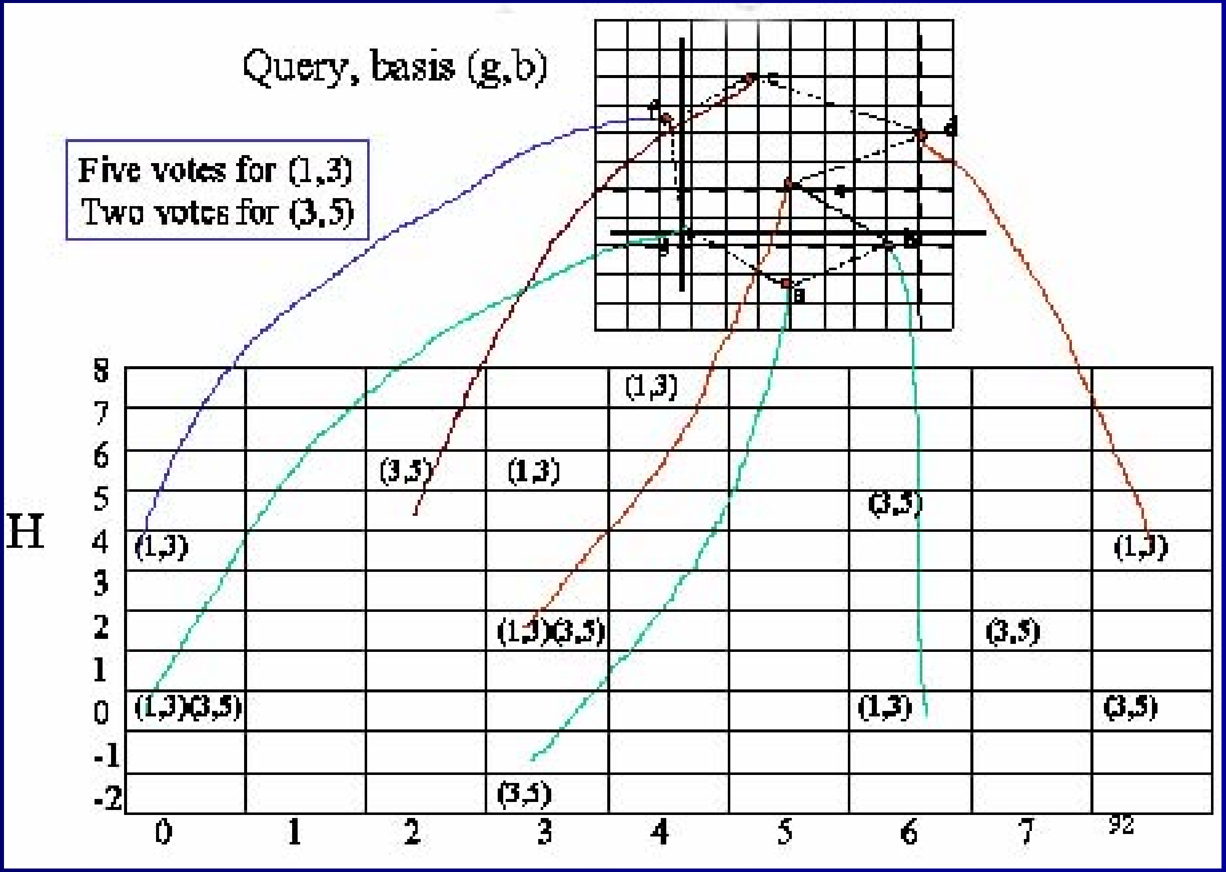
Preprocessing

- A 2-D hash table is used. It has a bin for each cell in the frame systems.
- Coordinates are recomputed for all points for every reference frame of a structure and for every structure.
- Table Entries: if for base pair [A $_k$, A $_l$] a point from the structure A, after appropriate rotation and translation, lies at position (x1, y1), then [A $_k$. A$_l$] is placed at position (x1, y1) in the 2D hash table.

Preprocessing



Detection


Query, basis (g,b)
Five votes for (1,3)
Two votes for (3,5)
H
8
7
6
5
4
3
2
1
0
-1
-2
0
1
2
3
4
5
6
7
92
(1,3)
(1,3)(3,5)
(3,5)
(1,3)
(1,3)(3,5)
(3,5)
(1,3)
(3,5)
(1,3)
(3,5)
(1,3)
(3,5)
c
d
b
g
a

Geometric Hashing(3Dimension)

- Extending the same idea to 3D as in 2D
  - *Reference Frame* is identified by three non-collinear points.
    - First point at origin (0,0,0).
    - Second point along the positive X – axis (x,0,0) .
    - Third point on the X – Y plane (x,y,0).
  - 3 D hash table is used.

Atom

- Basic unit of molecule structure is Atom
- Atom B:47 O LEU 7 1.15332 2.326867 -0.459129

*	47	: Atom number
*	O	: Atom type
*	LEU	: Residue type
*	7	: Residue number
*	1.15332	: x coordinate
*	2.326867	: y coordinate
*	-0.459129	: z coordinate

Definition of Terms

- In a protein structure, each atom is considered as a point.
- Atom coordinate are used as an index for a hash table entry as in 2D, which will contain a pointer to its other information

Definition of Terms

- Structure: a list of Atoms from a protein PDB file

  {44,47,48,49,55}

  – A number represents an Atom in PDB file.

- The number of Reference frames is (n-1)*(n-2)
  – Since we consider the first point of the list to be at the origin always.

Definition of Terms

- Label: information about Atom attached to hash table entry

- Information contained:

  * protein name, structure (list of atoms), reference frame

  * atom number, atom type, residue type, residue number

  * x coordinates, y coordinates, z coordinates

Hash bin

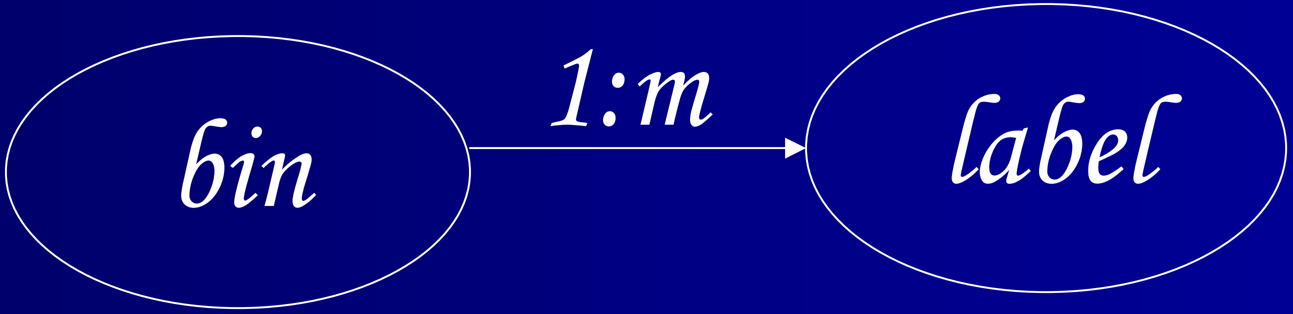
- A bin in 3D hash table is really a cube
- Bin size is scalable.
  - E.g. Atom with coordinates (1.153 2.326 -0.459)

resolution of the hash table	bin
1x1x1	[2,3,-1]
2x2x2	[1,2,-1]

Algorithm for Preprocessing

- Preprocessing (Update Database)
  - Input: a list of atoms from a protein structure
  - Output: update database

Two tables in database:



Algorithm for Preprocessing

```
For a molecule structure do :
        for each reference frame do
                for each atom in structure do
                        compute bin
                        update bin table and label table
                end
        end
```

Algorithm for Detection

Detection

- Input: { (a list of atoms from a protein structure), (percentage of similarity) }

- Output: return a list of similar structures if exists or return null;

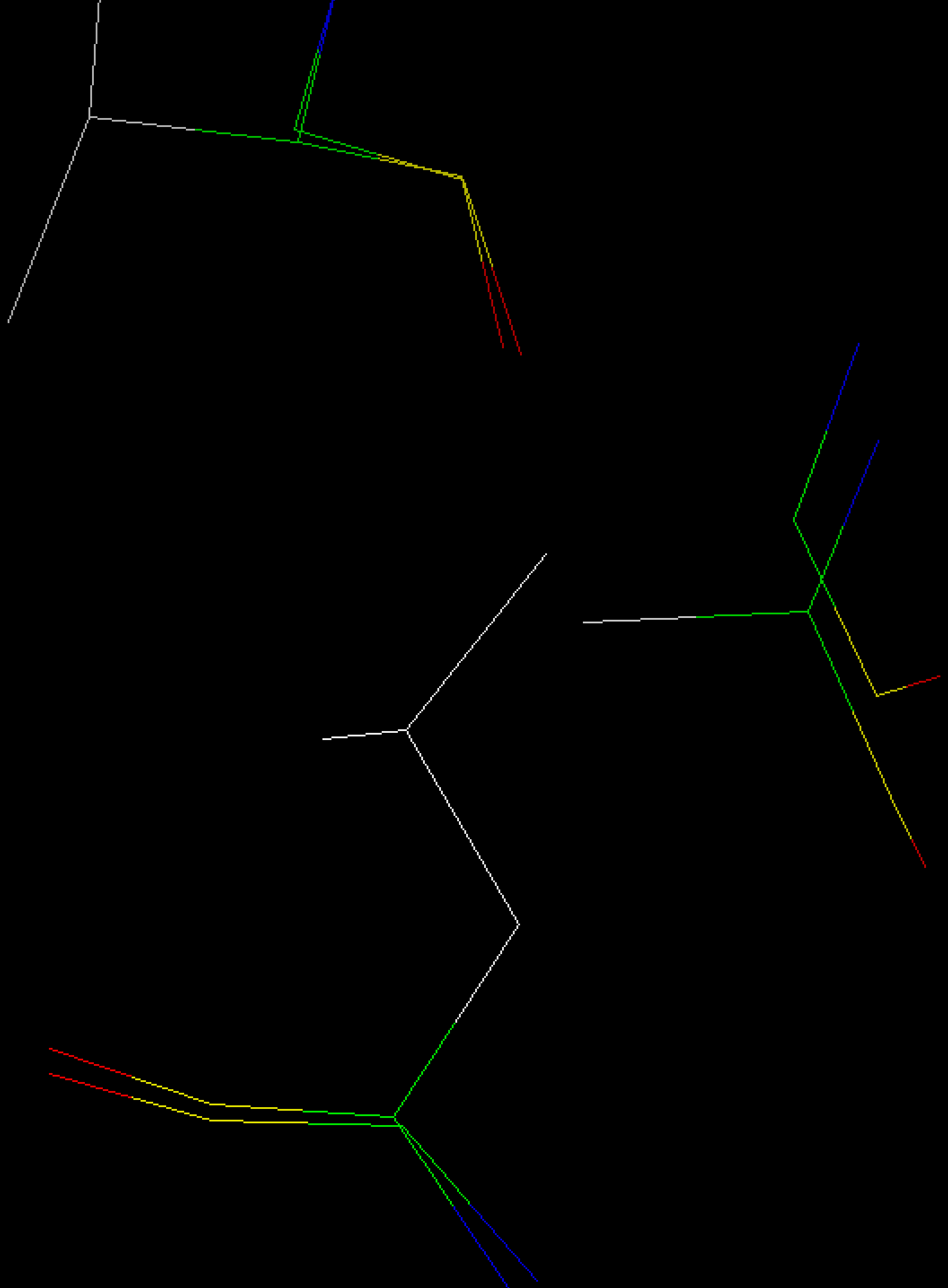
Algorithm for Detection

satisfactory coincidence sets:
The max. of votes>=the number of atoms * percentage of similarity

```
repeat
    initialize the vote table $V$ to 0
    choose three atoms as base
    for each atom do
        compute bin $M$
        for each entry $L$ in bin $M$
            $V(L):=V(L)+1$
        end
    end
Until (satisfactory coincidence sets are found or
    all reference frames are used)
```

Experiment and result

No. of structs in DB.	No. of Atoms in DB	No. of bins in DB	No. of entries in DB	No. of Atoms in query	Time	Match %
1	20	411	6839	12	2.5 min	>90
1	20	411	6839	6	17 sec	>90
1	15	163	3360	6	15 sec	>80
2	32	159	6810	12	3.5 min	>90
3	40	158	7146	12	4 min	>90
3	40	158	7146	10	1 min	>90



Blue: N

Green: CA

Yellow: C

Red: O

Future Works

- Rehashing.
  - In order to keep up the speed of recognition, it is important to limit the number of entries in each hash bin.
  - Rehashing is done if the number of entries in hash bin reaches a limit (preset).
- Check Neighboring Hash bins.
- Assign weighted votes.