Aggregation on the Basis of Structure Recognition

Jagdish Lal Raheja,
Department of Cartography
Technical University of Munich
Structure recognition: identify specific cartographic objects or aggregates as well as spatial relations, and measures of importance.

A city model consists mainly of buildings of different shape, style and roads.
Topics to be discussed

- Identification of objects (buildings).
- Identification of spatial relations.
- Aggregations rules based upon structure recognition.
Simple Buildings:

- Plane
- Pyramid
- Gable
- Hip
- L-Type
- U-type
- T-type
- Gambrel
- A-Frame
- Mansard
- Lean to wall
- Dutch hip
Complex buildings

a. buildings which has a roof as a combination of same or different roof styles.

- Cross gable
- Gable - Lean
- Arrow gable

b. building as a group of small buildings joined together forming a special shape

- Closed square
- Bridge
- Cross

- Arc
- Trapezium
Building recognition using NN
NN-bsiac structure
Test Data
### Neural Network (NN) approach

| BuildingId | BuildingType | x0  | y0  | z0  | x1  | y1  | z1  | x2  | y2  | z2  | x3  | y3  | z3  | x4  | y4  | z4  | x5  | y5  | z5  | x6  | y6  | z6  | x7  | y7  | z7  | x8  | y8  | z8  | x9  | y9  | z9  | x10 | y10 | z10 | x11 | y11 | z11 | x12 | y12 | z12 | x13 | y13 | z13 | x14 | y14 | z14 | x15 | y15 | z15 | x16 | y16 | z16 | x17 | y17 | z17 | x18 | y18 | z18 | x19 | y19 | z19 | x20 | y20 | z20 | x21 | y21 | z21 | x22 | y22 | z22 | x23 | y23 | z23 | x24 | y24 | z24 | x25 | y25 | z25 | x26 | y26 | z26 | x27 | y27 | z27 | x28 | y28 | z28 | x29 | y29 | z29 | x30 | y30 | z30 | x31 | y31 | z31 | x32 | y32 | z32 | x33 | y33 | z33 | x34 | y34 | z34 | x35 | y35 | z35 | x36 | y36 | z36 | x37 | y37 | z37 | x38 | y38 | z38 | x39 | y39 | z39 | x40 | y40 | z40 | x41 | y41 | z41 | x42 | y42 | z42 | x43 | y43 | z43 | x44 | y44 | z44 | x45 | y45 | z45 | x46 | y46 | z46 | x47 | y47 | z47 | x48 | y48 | z48 | x49 | y49 | z49 | x50 | y50 | z50 | x51 | y51 | z51 | x52 | y52 | z52 | x53 | y53 | z53 | x54 | y54 | z54 | x55 | y55 | z55 | x56 | y56 | z56 | x57 | y57 | z57 | x58 | y58 | z58 | x59 | y59 | z59 | x60 | y60 | z60 | x61 | y61 | z61 | x62 | y62 | z62 | x63 | y63 | z63 | x64 | y64 | z64 | x65 | y65 | z65 | x66 | y66 | z66 | x67 | y67 | z67 | x68 | y68 | z68 | x69 | y69 | z69 | x70 | y70 | z70 | x71 | y71 | z71 | x72 | y72 | z72 | x73 | y73 | z73 | x74 | y74 | z74 | x75 | y75 | z75 | x76 | y76 | z76 | x77 | y77 | z77 | x78 | y78 | z78 | x79 | y79 | z79 | x80 | y80 | z80 | x81 | y81 | z81 | x82 | y82 | z82 | x83 | y83 | z83 | x84 | y84 | z84 | x85 | y85 | z85 | x86 | y86 | z86 | x87 | y87 | z87 | x88 | y88 | z88 | x89 | y89 | z89 | x90 | y90 | z90 | x91 | y91 | z91 | x92 | y92 | z92 | x93 | y93 | z93 | x94 | y94 | z94 | x95 | y95 | z95 | x96 | y96 | z96 | x97 | y97 | z97 | x98 | y98 | z98 | x99 | y99 | z99 | x100| y100| z100| x101| y101| z101| x102| y102| z102| x103| y103| z103|
|----------------|-------------|-----|-----|----|-----|-----|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---
Input Data
<table>
<thead>
<tr>
<th>BuildingType</th>
<th>Predicted</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>664</td>
<td>445</td>
</tr>
<tr>
<td>2</td>
<td>642</td>
<td>559</td>
</tr>
<tr>
<td>3</td>
<td>640</td>
<td>536</td>
</tr>
<tr>
<td>4</td>
<td>622</td>
<td>519</td>
</tr>
<tr>
<td>5</td>
<td>603</td>
<td>498</td>
</tr>
<tr>
<td>6</td>
<td>583</td>
<td>479</td>
</tr>
<tr>
<td>7</td>
<td>559</td>
<td>455</td>
</tr>
<tr>
<td>8</td>
<td>526</td>
<td>422</td>
</tr>
<tr>
<td>9</td>
<td>500</td>
<td>396</td>
</tr>
<tr>
<td>10</td>
<td>476</td>
<td>372</td>
</tr>
<tr>
<td>11</td>
<td>459</td>
<td>355</td>
</tr>
<tr>
<td>12</td>
<td>442</td>
<td>338</td>
</tr>
<tr>
<td>13</td>
<td>425</td>
<td>321</td>
</tr>
<tr>
<td>14</td>
<td>408</td>
<td>304</td>
</tr>
<tr>
<td>15</td>
<td>391</td>
<td>287</td>
</tr>
<tr>
<td>16</td>
<td>374</td>
<td>279</td>
</tr>
<tr>
<td>17</td>
<td>356</td>
<td>255</td>
</tr>
<tr>
<td>18</td>
<td>339</td>
<td>236</td>
</tr>
<tr>
<td>19</td>
<td>322</td>
<td>222</td>
</tr>
<tr>
<td>20</td>
<td>305</td>
<td>203</td>
</tr>
</tbody>
</table>
Spatial relationship

Spatial relationships among individuals and neighborhood objects is based upon followings:

i. Proximity
   Closest

ii. Containment
    Contained

iii. Connectivity
     Connect

And can be divided into three levels:
Micro Level

It applies to individual 3D which building can be characterized by:

- A set of measurable parameters is required to characterize a building. The number of minimum parameters required varies from building to building. Following are the details of parameters for a few types of buildings.
Positional parameters
Other parameters

- Symmetry
- Regularity
- Orthogonal walls (in most cases)
- Roof types
- General Shapes
- L & U junctions
- parallel lines
- parallelism of faces
Meso level:

- It applies to an object in relation to its neighbors. Out of the three relations described above, proximity is very important relations among 3D buildings in neighborhood.
- Proximity can be obtained by measuring their mutual orientation, angle, distance between them and their height difference.
Proximity

Height difference

Angle

Alignment

Size contrast

Different roof style

Different building types
Macro level:

Applies to clusters of objects having similar properties such as settlement blocks and is based on visual grouping behaviors. These visual grouping behaviors are an important aspect in understanding images and maps and therefore has to be maintained. Macro level structure recognition helps to maintain this. Main emphasis is on

• Shape and size regularity.
• Height regularity.
• Regularity of roof structure and surface texture.
• Adjacency to other structures.
• Unique, deterministic features.
• Relative distributional density
Aggregation rules

Aggregation rules are based upon structure recognition and can be divided into following groups as follow:
**Linkage rules** define the spatial relations between buildings that must exist for their aggregation.

- **Proximity**: Two buildings must be disjoint but within a certain distance of each other for their aggregation;
- **Alignment**: Two buildings should be aligned or their alignment should not differ much and the difference must be below the permissible limits.
- **Angle**: Two buildings should have minimum angle between them if they are to be aggregated;
- **Height**: Two buildings should have minimum height difference if they are to be aggregated. **Roof Type**: Two buildings should have similar type of roof, i.e., planar or gable if they are to be aggregated;
- **Adjacency**: Two buildings must have their adjacent faces close if they are to be aggregated
Semantic rules: define the semantic relationships that must exist for aggregation. Semantic rules include relationships such as

- Class: Two buildings must be of the same class
- Structural: A group of objects form a common geographic or perceptual structure.
Orientational rules define the historical or local importance of the buildings. Important buildings such as monument, theatre, tower etc. reflect the city identity.

- Buildings which are of great importance should have their identities preserved at maximum of its extent. Therefore two buildings having historical importance should not be aggregated.
- If only one of the buildings is important, even then aggregation should not be done. Instead the important building should be exaggerated. The same rule applies to other local important buildings like TV towers, theatres, schools etc.
Based upon the above structural recognition, an algorithm for aggregation was developed. It is basically a nested set of If..Else statements. For example:

If $d(o_i,o_j) < \Delta d_{\text{min}}$ then AND
If $h(o_i,o_j) < \Delta h_{\text{min}}$ then AND
If $\text{roof}\_\text{type}_1 = \text{roof}\_\text{type}_2$ then AND
If $\Delta A(o_i,o_j) < A_{\text{min}}$ then aggregate

...
```cpp
double Enplui::GetClosestEntityDist(BODY *ent1, BODY *ent2, unit_vector &clUnitVector, unit_vector &clUnitVector2, position &position1, position &position2)
{
    // Get the closest faces of the body and direction between them
    int faceIndex1 = 0; faceIndex2 = 1;

    // This is the third logic (two are given below in commented blocks)
    // It looks to be simple and efficient but has to be thoroughly tested
    // Idea is just find the relative orientation of the building
    // mean how they are placed in the x-y plane as shown below
    //
    // entity1 --- entity2
    //     ______
    //     /     \
    //    //     //
    //    //     //
    //     \
    //     
    // entity2 --- entity1
    //     ______
    //     /     \
    //    //     //
    //    //     //
    //     \
    //     

    // This code does the calculation
    double lx = p0.x * lx;
    double ly = p0.y * ly;
    double lz = p0.z * lz;
    double px = p1.x;
    double py = p1.y;
    double pz = p1.z;
```
**api_combine_body**

**List of Functions**

**Subjects** Model Topology

**Contents** Euler Operations

**Action** Combines two given bodies into one body that contains all the lamps and wires from the original bodies.

**Prototype**

```c
outcome api_combine_body (BODY* from_body,
                         BODY* to_body)
```

**Includes**

- `#include "kernel/ ecs.hx"
- `#include "euler/ kermap/api/euler api.hx"
- `#include "kernel/ kermap/api/api.hx"
- `#include "kernel/ kerdata/top/body.hx"

**Description** The lamps and wires of the *from_body* are combined with those of the *to_body*, then the *from_body* is deleted.

The *to_body* becomes the combined body, the *from_body* is deleted. This is equivalent to the unite function on disjoint objects with no check for disjointness. It is much faster than the unite function and can handle wire.

**Errors**

- The pointer to *from_body* or *to_body* is NULL or does not point to a BODY.

**Limitations** Not applicable.

**Library** euler

**Filename** euler/ kermap/api/euler_api.hx

**Effect** Read-only
Future work:

Macro level spatial relation based upon visual grouping will be studied and consequently quantified to judge the quality of generalization. These relations are based upon:

- Parallelism
- Continuity
- Closure
- Proximity
- Orientation
Thanks!