Mathematical Problems in Engineering

An Improved Building Reconstruction Algorithm Based on Manhattan-World Assumption and Line-Restricted Hypothetical Plane Fitting

Xiaoguo Zhang,[1] Guo Wang,[2] Ye Gao,[1] Huiqing Wang,[1] and Qing Wang[1]

[1] School of Instrument Science and Engineering, Southeast University, Nanjing 210000, China.
[2] Arashi Vision Company Limited, Shenzhen 518000, China.

Correspondence should be addressed to Xiaoguo Zhang; xgzhang@seu.edu.cn

1. Supplementary Information for Methods

1.1 Algorithm 1

The algorithm of preliminary clustering of the 3D line segments can be descried using Algorithm 1.

Algorithm 1 Preliminary clustering of the 3D line segments

Input: $\{\boldsymbol{L_i}\}$, the initial 3D line segments from Line3D++ algorithm, where i=0…n_1

Output: **{**C_i}, the classified line-segment cluster set, where k=0..n

```
Insert {L_i | i = 0 ... n_1} into U
Sort U by the length of L_i ∈ U in descending order
Insert L_0 into C_0
while (loop ≤ 3 && |U|! = 0) do
    for (each L_i1 ∈ U) do
        for (each C_k ∈ C) do
            for (each L_i2 ∈ C_k) do
                if (angle between L_i1 and L_i2 less than α1) then
                    N1++
                end if
                N2++
            end for
            if (N1 / N2 > R)then
                Insert L_i1 into C_k
```

break
end if
end for
if($L_{i1} \notin (\forall C_k \in C)$) **then**
if(length of $L_{i1} > \bar{l}$) **then**
Add a new cluster into C
Insert L_{i1} into the new cluster
else
Put L_{i1} back into U
end if
end if
end for
loop++
end while

1.2 Algorithm 2

The extraction algorithm of axis-aligned and non-axis-aligned clusters of line segments can be described using Algorithm 2.

Algorithm 2 Extraction algorithm of axis-aligned and non-axis-aligned clusters of line segments

Input: {C_i}, the classified line-segment cluster set, where k=0..n
Output: O_0**, the optimal axis-aligned cluster trio**
N, the non-axis-aligned line-segment cluster set
Compute dominant directions of clusters in C
Sort clusters in C by the sum-length of $C_k \in C$ in descending order
Insert top 20 clusters of C into C''
for (each $C_{k1} \in C'', C_{k2} \in C'', k_1 \neq k_2$) **do**
if (angle between $\overrightarrow{D_{k_1}}$ and $\overrightarrow{D_{k_2}} \in (\alpha 2, 90°)$) **then**
Insert (C_{k_1}, C_{k_2}) into G
end if
end for
for (each (C_{k_1}, C_{k_2}) $\in G$) **do**
for (each $C_{k_3} \in C'', k_3 \neq k_1 \&\& k_3 \neq k_2$) **do**
if(angles between$(\overrightarrow{D_{k_1}}, \overrightarrow{D_{k_3}})and(\overrightarrow{D_{k_2}}, \overrightarrow{D_{k_3}})$)are both $\in (\alpha 2, 90°)$) **then**
Insert $(C_{k_1}, C_{k_2}, C_{k_3})$ into O
end if
end for
end for
for (each $O_j \in O$) **do**
Compute the weight W_j of O_j
end for

Sort $\boldsymbol{O}$ by the weight $\boldsymbol{W_j}$ of $\boldsymbol{O_j} \in \boldsymbol{O}$ in descending order
Choose $\boldsymbol{O_0}$ as the cluster of orthogonal directions
for (each $\boldsymbol{C_k} \in \boldsymbol{C}\&\&\boldsymbol{C_k} \notin \boldsymbol{O_0}$)**do**
if (angles between$\overrightarrow{\boldsymbol{D_k}}$ and every cluster directions in $\boldsymbol{O_0}$ are all$\in$ $(\boldsymbol{\alpha 3}, \boldsymbol{90}°)$) **then**
Insert $\boldsymbol{C_k}$ into $\boldsymbol{N}$
end if
end for

1.3 Algorithm 3

The algorithm for non-axis-aligned plane fitting can be described using Algorithm 3.

Algorithm 3 The algorithm for non-axis-aligned plane fitting

Input: N, the non-axis-aligned line-segment cluster set
Output: F, the final set of non-axis-aligned hypothetical planes

for (each $\boldsymbol{C_i} \in \boldsymbol{N}$) **do**
Insert $\boldsymbol{C_i}$ into $\boldsymbol{U}$
while ($\boldsymbol{loop} \geq \boldsymbol{10}\&\&|\boldsymbol{U}|! = \boldsymbol{0}$) **do**
Generate point cloud $\boldsymbol{V}$ from $\boldsymbol{U}$
Random select $\boldsymbol{L_p}$ and $\boldsymbol{L_q}$ from $\boldsymbol{U}$
Get the plane $\boldsymbol{S}$ from $\boldsymbol{L_p}$ and $\boldsymbol{L_q}$
for (each $\boldsymbol{L_k} \in \boldsymbol{U}, \boldsymbol{k} \neq \boldsymbol{p}\&\&\boldsymbol{k} \neq \boldsymbol{q}$) **do**
if ($\boldsymbol{L_k}$ is on plane $\boldsymbol{S}$) **then**
$\boldsymbol{ln_1}$ ++
end if
end for
if ($\boldsymbol{ln_1} \geq \boldsymbol{1}$) **then**
Get the normal vector of plane $\boldsymbol{S}$ as $\vec{\boldsymbol{n}}$
Mean shift cluster along $\vec{\boldsymbol{n}}$ on point cloud $\boldsymbol{V}$
Save the cluster result $\{\boldsymbol{P_a}|\boldsymbol{a} = \boldsymbol{0}, \dots, \boldsymbol{c_m} - \boldsymbol{1}\}$ as $\boldsymbol{Q_m}$
for (each $\boldsymbol{P_a} \in \boldsymbol{Q_m}$) **do**
Get the number of segments $\boldsymbol{P_a}$ included in as $\boldsymbol{ln_2}$
if ($\boldsymbol{ln_2} > \boldsymbol{50}\%|\boldsymbol{C_i}|$) **then**
Insert$\boldsymbol{P_a}$ into $\boldsymbol{F}$ and delete $\boldsymbol{P_a}$ from $\boldsymbol{U}$
end if
end for
end if
$\boldsymbol{loop}$++
end while
Select $\boldsymbol{Q'}$ which has max $\boldsymbol{W_3}$ from $\{\boldsymbol{Q_m}|\boldsymbol{m} = \boldsymbol{0}, \dots, \boldsymbol{n_9} - \boldsymbol{1}\}$
Plane filter on $\boldsymbol{Q'}$

Insert Q' into F
end for

1.4 Algorithm 4

The algorithm of subdivision of triangle meshes can be described using Algorithm 4.

Algorithm 4 The algorithm of subdivision of triangle meshes

Input: T_a, **the initial triangle mesh**
Output: T_i, **the subdivided triangle mesh**
for (each $t_m \in T_i$) **do**
Compute the area threshold s_0
if (area of $t_m > s_0$) **then**
Insert t_m into UT
while (UT is not empty) **do**
for (each $t_m \in UT$) **do**
if (area of $t_m > s_0$) **then**
Add the center v_c of t_m into P_i
Update the normal $\vec{n}$ and visible image list V of v_c
Add the new 3 triangles into UT'
else
Add t_m into T_i
end if
end for
Swap UT with UT', then clear UT'
end while
end if
end for

1.5 Algorithm 5

The algorithm of subdivision penalty mechanism can be described using Algorithm 5.

Algorithm 5 Subdivision penalty mechanism

Input: T_a**, the initial triangle-mesh**
Output: T_b**, the update triangle mesh using subdivision penalty mechanism**
Compute the ratio r_i of each triangle $t_i \in T_b$
for (each $v_j \in T_b$) **do**
if (v_j is expanded vertex) **then**

Get the parent triangle of $\boldsymbol{v_j}$ as $\boldsymbol{t_p}$

if (the ratio of $\boldsymbol{t_p r_{t_p}} \leq \boldsymbol{\beta_3}$) **then**

$\boldsymbol{f'_{v_j}} \leftarrow \boldsymbol{fu} + (\boldsymbol{f_{t_p 0}} + \boldsymbol{f_{t_p 1}} + \boldsymbol{f_{t_p 2}}) \,/\, \mathbf{3}$

else

$\boldsymbol{f'_{v_j}} \leftarrow (\boldsymbol{f_{t_p 0}} + \boldsymbol{f_{t_p 1}} + \boldsymbol{f_{t_p 2}}) \,/\, \mathbf{3}$

end if

else

$\boldsymbol{f'_{v_j}} \leftarrow \boldsymbol{f_{v_j}}$

end if

end for