Abstract

We presented a novel procedure to extract ground road networks from airborne LiDAR data. First point clouds were separated into ground and non-ground parts, and ground roads were to be extracted from ground planes. Then, buildings and trees were distinguished in an energy minimization framework after incorporation of two new features. The separation provided supportive information for later road extractions. After that, we designed structure templates to search for roads on ground intensity images, and road widths and orientations were determined by a subsequent voting scheme. This local searching process produced road candidates only, in order to prune false positives and infer undetected roads. A scene-dependent Markov network was constructed to help infer a global road network. Combination of local template fitting and global MRF inference made extracted roads more accurate and complete. Finally, we extended developed methods to elevated roads extraction from non-ground points and combined them with the ground road network to formulate a whole network.

1. Introduction

Network extraction has a wide application in remote sensing, medical imaging and computer vision. For road network extraction, different methods have been suggested under different situations, some dealing with roads on satellite and aerial images while others working on LiDAR data. In this article, we focus on ribbon-road network extraction from airborne LiDAR data.

On 2D imageries, fully automatic and high-accuracy road extraction is still challenging due to occlusion by trees (Fig.1(a)), self-overlapping (Fig.1(b)), disturbance of similar reflective objects and lack of context knowledge. There are several solutions like incorporating external geographic information [2], making assumptions on road intensity distributions or taking advantage of prior constraints on the geometry of networks[9]. Compared to 2D imageries, airborne LiDAR data provides additional 3D coordinates. This geometric information may help us to handle those tricky cases in 2D. In this paper, we rely solely on LiDAR data with no external information to extract road networks, aiming to detect roads with different reflective spectra, varied widths and orientations, overlapped mutually, and even occluded by trees partially or globally. Our contribution is manifold: (1) proposed two novel features, flatness and convexity, to discriminate buildings and trees; (2) designed structure templates to fit local road shapes, and presented a voting scheme to determine road widths and orientations; (3) constructed a Markov graph, which explicitly incorporates scene knowledge, to make a global inference on road networks.

2. Related Work

We briefly review works on road extraction from spectral imageries and LiDAR data. In [6], the authors made use of both height and intensity information to hierarchically classify point clouds into roads and non-roads. Hu et al.[7] relied mainly on LiDAR data, with some support from aerial images, to extract urban grid networks. They developed an iterative Hough Transform procedure to detect straight ribbons and used a hypothesis-test scheme to do verification. Zhu and Mordohai[1] addressed road network extraction from LiDAR by first performing segmentation using boundary and interior features on intensity images and then employing minimum cover set algorithm to search the most possible set of segments. Volodymyr et al.[5] presented a neural network learning approach to detect roads on high-resolution aerial images. The proposed neural network contains millions of trainable weights which take a large context into account. In [4], Pechaud et al. extracted tubular structures on 2D images by searching shortest geodesic paths in a 4D space.

For works focusing on road networks, Ting et al. [3] presented phase field higher-order active contour models, and exploited them to segment road network from high resolution images. Tupin et al.[8] first detected individual line segments from SAR images, and then modeled interactions among line segments by MRF and network extraction was transformed into a MRF labeling problem. Negri et al.[13] integrated road junction information into the road network extraction process and made an improvement over the approach using no junction information when experimenting on SAR images. In [12],
authors modeled the line network by an object process. Their model considered both topological and radiometric properties of line networks, and was optimized by RJMCMC algorithm.

In our work, we aim to extract both ground and elevated roads from airborne LiDAR data, and formulate them into a connected road network.

3. Overview of the approach

We start by separating non-ground points from ground planes by hierarchical morphological opening.

Then we classify elevated points into two classes: buildings and trees. With a slight abuse of terminology, in the rest of the paper, we call all elevated man-made objects as buildings. Two features, flatness and convexity, are presented to discriminate them. Their separation helps to support the later ground road extraction. Moreover, elevated roads are to be extracted from buildings.

Afterwards, we search ground road candidates from the ground plane. To get road widths and orientations simultaneously, elongated structure templates with varied widths and directions are designed to fit local intensity distributions and a subsequent field voting scheme is introduced to refine candidate widths and orientations.

Then, we model interactions among ground road candidates through a Markov graph, formulating ground road network extraction as a labeling problem. Finally, we extended the developed method to extract elevated roads. The combination of extracted ground and elevated roads are final outputs. The pipeline of road network extraction system is shown in Fig. 2.

4. Ground and non-ground separation

Elevated roads will be self-overlapping, and every individual one will have a different elevation. However, roads on ground have the same height locally and won't be overlapped. Due to different characteristics, they will be extracted in separate procedures. So we separate point clouds into ground and non-ground first: the former for ground roads extraction and the latter for elevated roads extraction. Point clouds are first converted into 2D depth images, and separation is implemented on them.

Morphological filtering is a major kind of methods for digital terrain model (DTM) generation [16] from depth images. Opening is a successive operation of erosion and dilation, after which elevated objects smaller than the structure element (SE) are removed while terrain is retained. However, the larger the SE is, the less accurate the reconstructed terrain is. To address this problem, several scholars [6,16] proposed hierarchical strategy: first using large SE to remove large objects, and decreasing its size gradually to remove small objects like vehicles and individual trees. We adapt the hierarchy strategy in [6] to separate grounds from elevated objects. The output of the hierarchy strategy is a DTM, and by subtracting DTM from depth images, pixels where absolute difference values are greater than certain threshold are elevated, otherwise they are on the ground.

5. Buildings and trees separation

Buildings and trees have distinct spatial relationships with roads, concretely, projection of buildings onto XY plane won’t overlap with roads, while this is not true for trees, e.g. in case where roads are occluded by trees. In later road extraction steps, we will use this scene knowledge to construct road networks.

As a preparation step, elevated points are partitioned into buildings and trees. Two features, flatness and convexity, are devised to distinguish them. After initial binary labeling
(0 for trees and 1 for buildings) of elevated pixels based on two feature values, we solve their optimal labels in an energy minimization framework.

5.1. Flatness

**Definition 1 (Flatness):** For a pixel \( p \) on 2D depth image, its neighbors \( N_p \) are pixels covered by a \( n \times n \) window centering on \( p \). Flatness at pixel \( p \) is defined by votes gotten by \( p \) after a voting process: for a neighboring pixel \( l (l \in N_p, l \neq p) \), if the included angle between its normal \( n_l \) and \( n_p \) (normal at \( p \)) is smaller than threshold \( t_0 \), then it will vote \( p \). After all \( n^2 - 1 \) neighbors finish voting, votes accumulated by \( p \) is defined to be flatness at \( p \). And this voting process is called cumulative voting. The larger the number of votes one point gets, the flatter it is locally at that point. Ideally, if all normals have the same direction, flatness at \( p \) is \( n^2 - 1 \); to another extreme, assuming normal distributes randomly, and the included angle between any pair of normals is greater than \( t_0 \), then flatness at \( p \) is 0.

Generally, normal distribution over a tree crown is random, while its distribution on a local roof, especially a planar roof, has one dominant direction. Flatness is illustrated in Fig.3.

![Figure 3. Flatness. Left: on a locally planar surface, normals have similar directions, and cumulative voting produces a high value; Right: on an irregular surface, normal directions are randomly distributed, and cumulative voting produces a low value.](image)

5.2. Convexity

Cumulative voting is especially effective for identifying planar roofs. However, for sphere, cone, cylinder and other shaped roofs, it’s less distinctive, because normal still varies within a local region. In order to avoid misclassifying these kinds of roofs as trees, we propose convexity. The intuition behind convexity is that most roof surfaces are locally convex, which is usually not true for tree crown surfaces.

**Definition 2 (convexity at a point):** on 2D depth images, we quantize local convexity of surface \( S \) at pixel \( v_0 \) using a numerical value. As in Fig. 4(b), \( \{v_1, v_2, \ldots, v_9\} \) are 8-neighboring pixels of \( v_0 \). Any two consecutive neighbors \( \{v_i, v_{i+1}\} \), together with \( v_0 \), form a triangular facet \( f_k \).

There are totally 8 facets \( \{f_1, f_2, \ldots, f_8\} \), which are the facets going through \( v_0 \) on surface \( S \). The convexity of surface \( S \) at pixel \( v_0 \) is calculated as follows:

1. For each facet \( f_k \) with vertices \( \{v_0, v_i, v_{i+1}\} \), the other 6 points are partitioned into two groups according to their distances to \( f_k \): one group having positive distances and the other having negative ones. Supposing the maximum absolute distance in each group is \( d_{pos} \) and \( d_{neg} \) respectively, we assign value \( \min(d_{pos}, d_{neg}) \) to facet \( f_k \), i.e. let \( d_{f_k} = \min(d_{pos}, d_{neg}) \).

2. Each facet gets its assigned value \( d_{f_k} \) and convexity at point \( v_0 \) is defined as \( \max(d_{f_1}, d_{f_2}, \ldots, d_{f_8}) \).

For points lying on locally convex surfaces (sphere, cone and cylinder shaped roofs), their convexity values are 0, while for tree canopy surfaces, they are no more locally convex, so points on them have larger convexity values.

![Figure 4. Convexity. (a) Left: the surface going through the central point is locally convex, so convexity at the central point is 0. Right: a non-convex surface through the central point, convexity at the central point is a positive value; (b) Convexity computation at pixel \( v_0 \) on depth images.](image)

5.3. Segmentation by energy minimization

After computation of flatness and convexity for elevated pixels, we use MRF to classify them into building and trees. Only first and second order potentials are considered and 8-neighborhood is used as the neighboring norm. The pairwise interaction is defined by the standard Potts model [10] and the data term is a combination of two features defined above:

\[
E_{d_i} = \begin{cases} 
E_{f_i} + aE_{c_i} & \text{if } i \in \text{buildings} \\
\frac{e^{-f_i}}{e^{-f_i} + e^{-c_i}} & \text{if } i \in \text{trees} \\
0 & \text{otherwise}
\end{cases}
\]

\[
E_{f_i} = \begin{cases} 
t_0/c_i & \text{if } c_i \leq t_0 \cap i \in \text{trees} \\
1 - t_0/c_i & \text{if } c_i > t_0 \cap i \in \text{trees} \\
1 - t_0/c_i & \text{if } c_i > t_0 \cap i \in \text{buildings} \\
c_i/t_0 & \text{if } c_i \leq t_0 \cap i \in \text{buildings}
\end{cases}
\]

Where \( f_i \) and \( c_i \) are flatness and convexity at pixel \( i \) and \( n \) is the size of the cumulative voting window. A graph-cut algorithm [10] is used to solve this optimization problem. It’s easy to check that our model fits the requirements of this algorithm. There are three parameters, namely \( t_0 \) and \( \alpha, \beta \). Parameters \( \alpha \) balance energy within data term, while \( \beta \) adjusts ratio between data and interaction terms. In the experiment, we set \( t_0 = 0.3, \alpha = 0.5, \beta = 0.5 \). Although
rigorously one should learn them through labeled samples, we notices that this parameter setting performs well on a variety of LiDAR datasets.

6. Ground road candidates extraction

Ground roads are contained on the ground plane. Since neighboring points on ground have similar elevations, elevation is not a good cue for identifying roads from other ground objects such as parking lots and grass land. Therefore, we disregard depth information and search roads on 2D intensity maps of ground points. This section presents template fitting and field voting methods to extract road candidates.

6.1. Template fitting to search roads

Roads on aerial LiDAR data are ribbon-shaped elongated regions instead of one-pixel width curves. In order to extract roads of various widths and orientations, we devised elongated structure templates with varied sizes and directions. Search of road parameters is equivalent to template fitting.

6.1.1 Local structure templates

A typical elongated structure template with width w and height h is shown in Fig. 5(a), and it captures the local geometry of a vertical road with width w. This template has three contiguous parts \( R_1, R_2, R_3 \), in which \( R_1 \) indicates roads and \( R_2, R_3 \) indicates non-roads. \( R_2, R_3 \) are both half of the width of \( R_1 \). A standard template \( T_0 \) of size \( w_0 \times h_0 \) is defined as (1: road regions, 0: non-road regions):

\[
T_0(x,y) = \begin{cases} 
0 & 0 < x < 0.5 \cdot w_0, 0 < y < h_0 \\
0.5 \cdot w_0 < x < 1.5 \cdot w_0, 0 < y < h_0 \\
1.5 \cdot w_0 < x < 2 \cdot w_0, 0 < y < h_0 
\end{cases} \tag{2}
\]

Two parameters \((r, \theta)\) are introduced to generate templates with varied widths and orientations from the standard template \( T_0 \):

\[
T(r, \theta) = R_{0} \cdot \frac{r}{w_0} \cdot T_0 \tag{3}
\]

Where \( R_{0} \) is the planar rotation matrix of angle \( \theta \), and \( r \) is the designated template width. Each template \( T(r, \theta) \) represents a road oriented along direction \( \theta \) with width \( r \). Direction \( \theta \) lies within \([0, \pi)\), and \( r \in [w_{\text{min}}, w_{\text{max}}] \), where \( w_{\text{min}}, w_{\text{max}} \) are respectively widths of narrowest and widest roads in the scene. Scaled and rotated templates are shown in Fig. 5(c).

The choice of ratio \( \gamma = h/w \) between height and width of templates is a trade-off between robustness and accuracy of local road detectors. The larger the ratio is, the more robust the detector is, while the smaller the ratio is, the more accurately the detector performs. In the experiment, we let \( \gamma = 2.5 \).

6.1.2 Road saliency within template area

Given a template \( T_i \), how to measure the probability of existence of a road with the same orientation and width within the template area? When observed locally from an image, roads are perceptible especially when their intensity deviates from that of the background. Here we use fisher discriminant to capture the local dis-similarity of intensity distribution between road and background, which takes the form

\[
S = \frac{(m_r - m_{nr})^2}{s_{ir}^2 + s_{itr}^2} \tag{4}
\]

Where \( m_r, m_{nr} \) are respectively the mean intensity values within \( R_1 \) and \( R_2 \cup R_3 \), and \( s_r, s_{nr} \) are the corresponding standard variances. The value of fisher discriminant \( s \) is called road saliency here.

6.1.3 Template fitting

We allow template center shiftable within \( R_1 \), Fig.5(b) shows a center-shiftable template. By not centering the template on the point under consideration, we can find a shift that best fits it with local roads. Given a point \( p \) on roads, to compute its road saliency \( s_i \) under template \( T_i \), we compute fisher discriminants for all shifts and keep the maximum saliency as the saliency under template \( T_i \). Compared to center-fixed templates, there are obvious merits: the saliency produced for a point near road sides are at the same level as that produced for a point at the road center. Suppose \( T_0 \) is the template producing the maximum saliency at \( p \), and \( T_p \) is called fitted template and its width and orientation is said to be the candidate width and orientation of the road going through \( p \).

6.2. Field voting to determine road widths and orientations

Candidate width and orientation at each point are not necessary the true width and orientation of the road going through that point. To get road width and orientation as close as possible to the true ones, we designed a voting scheme, during which width and orientation information are propagated into neighborhood.

After determination of fitting template \( T_p \) at \( p \), a voting

![Figure 5. Fitting template: (a) template; (b) shifting center within region \( R_1 \); (c) designed templates of different widths and orientations](image-url)
field $V_{F_p}$ is created as follows: (1) $V_{F_p}$ is a rectangle, having the same width and orientation as $T_p$; (2) $V_{F_p}$ is fitted to lie between $R_2$ and $R_3$ and overlaps with $R_1$ of $T_p$; (3) $p$ lies on the center line of voting field $V_{F_p}$. The extension of $V_{F_p}$ along the road candidate direction is $h(V_{F_p})$, i.e. points as far as $h(V_{F_p})/2$ apart from $p$ along that direction will still be covered by $V_{F_p}$. The ratio $\gamma_e = h(VF)/w(VF)$ is set by making a balance between accuracy of road detection and gap filling capability of voting field. Fig. 6 shows a fitting template and the constructed voting field.

Each point on the ground mask maintains an accumulator array, with one dimension referring to quantized widths $w \in [w_{min}, w_{max}]$ and the other to quantized orientations $\theta \in [0, \pi)$. When it falls within a voting field $V_{F_i}$, it gets one vote and its accumulator array at position $w(\theta)\in V_{F_i}$ increases by one. After it collects votes from all neighboring fields, the position of the peak value in its accumulator array is the width and orientation of roads going through it.

In the experiment, a point $p$ with the peaking votes less than $\lambda^p w^p (\lambda < 1, w_p$ is the road width at $p$) will be removed. All left points consist of road candidates.

**7. Ground road network construction**

Field voting propagates road widths and orientations into neighborhoods, therefore effects of local tree occlusions are compensated (Fig. 7). However, for roads largely occluded by trees, they can’t be detected. In this section, we model road networks as a Markov graph, in order to infer undetected roads and prune false road candidates.

**7.1. Centerline tracking**

For the convenience of management, centerlines of road candidates are tracked and used to substitute for road ribbons. We introduce a multi-step marching algorithm. This algorithm makes a balance between simplicity and accuracy: for a straight road, marching steps are large and it’s finally represented by few long line segments, while for a curvy road, marching steps are adjusted adaptively to shorter ones, making the representation more accurate. This algorithm works on votes maps, and finds the point with the maximum votes as the start marching point. Marching begins along the road direction at current point, and a larger marching step is used first, if the difference between road direction at the next potential point and current marching direction is less than certain value, that point is accepted and marching continues from that point. Otherwise, a smaller step is used until the next valid point is found or road ends have been reached. This process iterates until no road candidates are left.

**7.2. Missing roads hypothesis**

First we make hypotheses on potential missing roads based on network geometry and scene knowledge.

**7.2.1 Road network geometry**

Wholly based on geometry of tracked centerlines, missing roads happen in two cases most often: (1) gaps between segments (Fig. 9); (2) T-intersection connection (connection between one end point and a potential T-intersection). We use criteria in [14] to measure the possibility of every gap connection, and keep top $m$ possible gap connections. For a potential T-intersection connection, the possibility of its existence is measured by:

$$P_T = \sin(\theta) \cdot \left[1 + \exp\left(\frac{d_{gap}}{l_{tar}}\right)^{-1}\right]$$

(5)

Where, $\theta$ is the included angle between reference and target segment, and right angle is preferred. $l_{gap}$ and $l_{tar}$ are lengths of gap and target segment respectively (Fig.9). Top $m$ possible T-intersection connections are kept.
7.2.2 Context knowledge to filter hypothesized missing roads

For \( m + n \) proposed missing roads, if any one going across a building, then it is rejected and removed from initial hypotheses. Left ones are kept as missing road candidates.

7.3. Road network formulation

Based on the observation that roads are generally connected, interactions among segments are modeled by a graph. A node corresponds to either a road candidate or a missing road candidate, and an edge exists between two nodes if they are gap-connected, T-intersection connected or intersecting each other. Road network extraction is equivalent to the labeling of segments in the graph. This labeling procedure is performed under the MRF framework with unary and pairwise potentials. The quality of the labeling procedure is measured by the energy:

\[
E(L) = \sum_{i \in S} D_i(l_i) + \beta \sum_{(i,j) \in E} I_{ij}(l_i, l_j, d_{ij})
\]  

(6)

Where \( i \) corresponds to a tracked centerline or a hypothesized one, \( l_i \) is the label of \( i \) (1 for true roads and 0 otherwise), \( D_i \) and \( I_{ij} \) are data and interaction terms, balanced by a parameter \( \beta \). In the model, interaction term \( I_{ij} \) is data dependent on segment lengths \( d_{ij} \). Different data terms are used for tracked roads and hypothesized ones. For tracked roads, penalty is defined in terms of length:

\[
D_{li} = \begin{cases} 
1 - 0.5d_i/d_0 & \text{if } d_i \leq d_0 \land l_i = 1 \\
\frac{c_1}{d_i} & \text{if } d_i > d_0 \land l_i = 1 \\
\frac{c_2}{d_i} & \text{if } d_i > d_0 \land l_i = 0 \\
0.5d_i/d_0 & \text{if } d_i \leq d_0 \land l_i = 0 
\end{cases}
\]  

(7)

Where \( d_i \) is the length of \( i \)th segment, and \( c_1, c_2 \in [0,1] \) with \( c_1 < c_2 \). For hypothesized ones, penalty is defined in terms of a ratio between areas of trees and ground within the hypothesized ribbon:

\[
D_{li} = \begin{cases} 
\frac{1}{1+e^r} & l_i = 1 \\
\frac{c}{l_i} & l_i = 0 
\end{cases}
\]  

(8)

Where \( c \in [0,1] \) and penalizes non-roads. A sigmoid function is used to penalize \( l_i = 1(\text{road}) \) due to the fact that missing roads are generally occluded by trees, while roads in open areas are much less possible to be undetected due to the completeness for road candidate extraction. When there is a large portion of trees in the hypothesized missing ribbons, possibility of being a road is pushed up rapidly.

For interaction terms, two different connections are modeled separately. Gap intersection is modeled by a standard Potts model, which favors smoothness. For T-intersection connections or intersecting relations, the penalty is given by:

\[
I_{ij} = \begin{cases} 
\frac{\min(l_i, l_j)}{\max(l_i, l_j)} & l_i = l_j \\
\frac{\min(l_i, l_j)}{\max(l_i, l_j)} & l_i \neq l_j 
\end{cases}
\]  

(9)

Where \( \theta \) is the T-connection or intersecting angle. Find the labeling configurations, which minimizing the penalty \( E(L) \) is a non-convex optimization problem. However, in a 1km\(^2\) scene, there are generally 200–300 segments, optimization using standard iterated conditional mode approach, graph cut[10] and Gibbs sampling method can be very fast to get an approximately global optimal solution. In the experiment, graph-cut algorithm takes 2s for 200 segments. Ground road network extraction results from a heavily occluded area are shown in Fig. 10.

8. Elevated road extraction

Until now, designed methods are oriented to extract roads on the ground, precluding all elevated roads. However, template fitting, field voting and marching algorithms are easily extended for elevated roads extraction from buildings. There are two major differences:

We use template correlation, instead of Fisher discriminant, to compute local road saliency. The normalized cross-correlation value between reference template \( T_r \) and target template \( T_g \) is used as road saliency. \( T_r \) has value 1 in region \( R_1 \) and 0 in \( R_2 \cup R_3 \), while values in \( T_g \) are determined as follows: for an elevated pixel \( p \) under consideration, let \( S_r, S_{mr} \) be its neighboring pixels in
road region $R_1$ and non-road regions $R_2 \cup R_3$, and let $\pi$ be the plane going through $p$. When the distance from a point $p_i$ in $S_1$ to $\pi$ is smaller than certain value $t_0$, target template value at $p_i$ is set to 1, otherwise to 0; conversely, when the distance from a point $p_{tr}$ in $S_{tr}$ to $\pi$ is larger than $t_0$, target template value at $p_{tr}$ is set to 0, otherwise to 1. $\pi$ can be fitted by points around $p$ using least square.

Second, in centerline marching, an additional constraint is added: elevation differences between two consecutive valid marching points should be inferior to certain value.

9. Experiments

Our road extraction pipeline has been tested on LiDAR data from Denver, Atlanta and Oakland. These datasets mainly cover downtown and surrounding residential areas. As a preparation, 3D point clouds are converted into 2D depth and intensity images, with resampling resolution of 1m at both X and Y directions.

9.1. Distinctiveness of flatness and convexity for separating buildings and trees

We manually labeled all buildings and about 90% trees on three depth images (from 3 regions, each covering 1km$^2$). Labeled buildings include tall downtown buildings with complex roofs and low residential houses. Then, flatness and convexity are computed and their statistics for buildings and trees are shown in Fig.11.

Distribution curves show a clear inverse correlation between buildings and trees. Building separation results using these two features show in Fig.12. On flatness map, the brighter a pixel is, the higher its flatness is, while on convexity map, the bright it is, the smaller its convexity is. These two features show a great capability to suppress trees while strengthen buildings. As to building separation, although we didn’t quantitatively compute precisions, by visually checking, results look satisfactory.

9.2. Strength of template fitting and field voting

Structure template fitting and field voting can extract roads with diverse reflective characteristics, varied widths and orientations and under local tree occlusions. Template fitting only assumes that local road intensities are homogeneous, thus it successfully detected roads of different surface materials, which may show different reflective intensities, i.e. concrete and asphalt roads are both extracted instead of their different intensity value scopes. Fitting templates have changeable widths and orientations, making curvy and width-varying roads be extracted in a uniform process. For local tree occlusions, when road points under occlusion are within effective voting areas, occluded parts can be recovered. Fig. 13 shows the capability of road extraction under different cases.

Figure 11. Flatness(CV value) and convexity distribution of labeled buildings and trees

Figure 12. Building separation from trees using flatness and convexity. Top Left: point clouds; Top Right: flatness map; Bottom Left: convexity map; Bottom Right: detected buildings

Figure 13. Local template fitting and field voting to extract roads under varying conditions. Left: LiDAR point cloud; Right: extracted road networks superimposed on intensity images; the 1st row: various intensity scopes; the 2nd row: curvy roads; the 3rd row: several widths; the 4th row: local tree occlusion
9.3. Road network extraction evaluation

We have experimented on two datasets: Denver and Atlanta areas. Denver dataset covers 20 km² downtown areas, most of objects are buildings and there are only few trees, while Atlanta data covers 35 km² downtown and residential regions, where many roads are occluded by trees. The extracted roads were compared with ground truth roads, which were manually labeled on intensity images, using aerial images for verification. Only centerlines were labeled. Then three indexes [15] (1) completeness (2) correctness and (3) quality are used to evaluate the extraction results. Although our algorithm extracted both road centerlines and boundaries, we only compared centerlines with labeled roads. If the labeled roads are falling within the extracted road ribbons, then the corresponding road centerlines are true positives. If no labeled roads fall within an extracted ribbon, then the corresponding centerline is a false positive. When a labeled road isn’t covered by any extracted ribbons, then it is an undetected road. Statistics for two datasets show in Table 1. In Denver areas, there is much less occlusion, and most roads are extracted during template fitting and field voting, however, in more occluded Atlanta areas, although road candidates are incomplete, many undetected roads are inferred by later scene-context embedded Markov network.

10. Conclusions

In this paper, we proposed an original procedure for road extraction from aerial LiDAR data. The main strength of the procedure is that it combines a robust local detector with a global context-incorporating graph to reach both high correctness and completeness. However, for occluded curvy roads, there is less probability to be detected by our graph model. Moreover, elevated and ground roads are extracted in separate processes, so their interactions are not yet considered. These problems are our further work.

References

[2] Aleksey Boyko, Thomas Funkhouser. Extracting roads from dense point clouds in large scale urban environment. ISPRS
[6] Simon Clode, Peter Kootsookos, Franz Rottensteiner. The automatic extraction of roads from LiDAR data. ISPRS.

<table>
<thead>
<tr>
<th></th>
<th>True length(km)</th>
<th>True positives(km)</th>
<th>False positives(km)</th>
<th>completeness (%)</th>
<th>correctness (%)</th>
<th>quality (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Denver</td>
<td>269.4</td>
<td>247.4</td>
<td>30</td>
<td>91.8</td>
<td>89.2</td>
<td>82.6</td>
</tr>
<tr>
<td>Atlanta</td>
<td>421</td>
<td>367.1</td>
<td>48.2</td>
<td>87.2</td>
<td>88.4</td>
<td>78.3</td>
</tr>
</tbody>
</table>