Eurographics Workshop on Urban Data Modelling and Visualisation (2016)
F. Biljecki and V. Tourre (Editors)

Realistic urban road network modelling from GIS data

H.H. Nguyen [1,2,3], B. Desbenoit [1], and M. Daniel[1]

[1]Aix-Marseille University, CNRS, LSIS UMR 7296, 13009, Marseille, France
[2]Institute of Information Technology, VAST, Hanoi, Vietnam [3]University of Science and Technology of Hanoi, VAST, Hanoi, Vietnam

Abstract

Starting from GIS data, which are sampled and often inaccurate, this paper presents a method to reconstruct urban road surfaces respecting important geometric constraints selected from civil engineering. We propose a mathematical road surface model based upon road axes and properties. In addition, we introduce a process to produce a mesh representing the roads and the terrain so that roads and terrain match. Experiments and compelling results prove the efficiency of our framework.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual reality

1. Introduction

Modeling real-world urban road networks is an active and challenging topic in computer graphics. Roads play an important role of urban models in a variety of applications including serious games, virtual traveling, driving simulation, and especially autonomous cars. There have been several researches devoted for road models but very little attention has been paid on real urban roads and their geometric properties. Consequently, the existing models could look fancy while not being suitable for driving simulation. On the contrary, this work aims at reconstructing urban road networks with respect to the important geometric constraints of real-world roads.

The data for real roads, like other spacial data, are often stored and managed by a Geographic Information System (GIS) which facilitates data interrogation by allowing users to ask queries like: "What are the roads existing in a desired region?", "What are the coordinates of the first point of a specific road?". Nowadays, there exists numerous GIS data sources, both freely available such as OpenStreetMap, Natural Earth and commercial like TIGER/Line data, Here map data. Therefore, GIS data are very suitable for road reconstruction because of their standardization, ease of exploiting, and availability.

In reality, road blueprints consist of road axes, road cross sections, and the details of intersections as described in section 2. The reconstruction of a single road axis from a polyline stored in a GIS database has been well addressed in [WSL12, NDD14, WLS14]. This paper investigates the forthcoming problems to obtain road surfaces from axes and other GIS data types. Beyond the polylines for road axis, we are provided with the following data types:

- **Road properties** such as *name, width, direction* (two-way, same or opposite direction to the polyline direction), *number of lanes, road type* (highway, entry, exit, bridge, tunnel), and the *importance* ranging from 1 (most important) to 5 (least important).
- **Terrain heightmap**, a.k.a DEM, which is a 5mx5m regular grid with a 0.5m-altitude accuracy. From this, a mesh representing terrain could be easily built but the mesh resolution is insufficient to present road surface as illustrated in Figure 7a.
- **Orthography** which is a raster image of the terrain plus its corresponding geographical coordinates. It can be used to verify the position of shape of the resulting road axis and surface.

These data raise at least two issues. The terrain heightmap has an accuracy incompatible with the road network. This network is only given by its axes which are irregularly sampled, generally noisy and not relevant everywhere.

Our method, which has to cope with these difficulties and must overcome them, can be overviewed as follows (Figure 1). We start with a 3D road axis network generated from polylines and road properties by the method that we extend from [NDD14]. Like in real road design, this method decomposes the data in order to design an horizontal curve and a vertical curve, both having specific rules, and merges them to obtain the 3D axis. Based on the axis, as in civil engineering, we define a mathematical road surface model with 3D patches for road intersections and cross-section for trajectories. To produce a geometry representing the model, we first build an initial mesh from the input terrain heightmap which is in a low resolution, and carry out a two-step mesh-displacement process. In the former step, we subdivide this initial mesh adaptively so that the higher curvature regions such as road center and borders have higher resolutions, and the quite flat zone from road center to borders has a lower resolution. Then, mimicking the way that real roads are constructed by digging out and embarking the original terrain, we adjust the altitude of each vertex inside the road zone.

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

delivered by EUROGRAPHICS DIGITAL LIBRARY www.eg.org diglib.eg.org

DOI: 10.2312/udmv.20161414

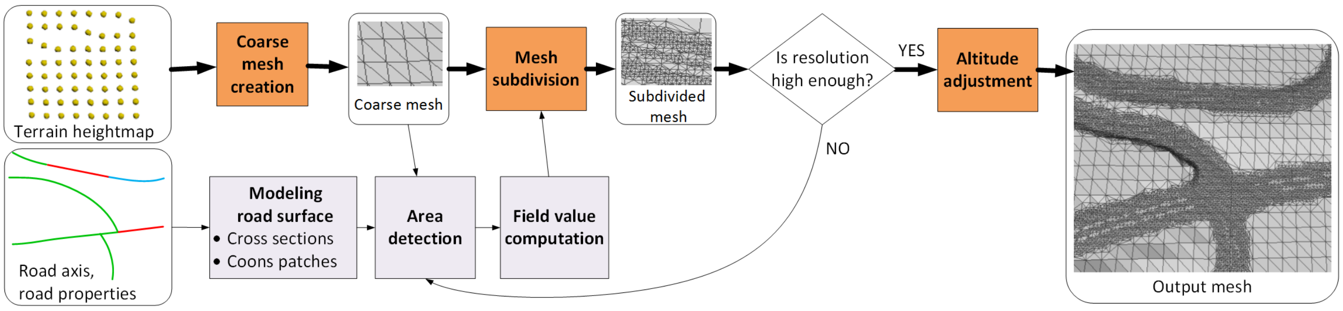

Figure 1: *Overall of our method.*

The next section describes our input data and constraints. We discuss related work in Section 3. To obtain an urban road network axis, we extend the algorithm of [NDD14] in Section 4. A mathematical model for the road surface is proposed in Section 5, then the mesh displacement is detailed in Section 6. Section 7 is devoted to our results and Section 8 concludes the paper.

2. Geometric constraints

In order to model real roads, the following important constraints, according to civil engineering rules [AAS11, SET02], must be respected.

Firstly, the road surface must follow the given **road axis**, i.e. the center of the generated surface must be very close to the axis line given in the GIS by a set of irregular sampling points.

Secondly, the road surface should conform to the **cross section** defined by slicing up the road surface with a plane perpendicular to the axis (the red line in Figure 2). In this work, we only consider the two shoulders and the traveled way because they are the most important components (Figure 2). One may notice that the urban roads only allow low speed of vehicle, so superelevation, the technique tilting the road surface (and cross section) at high-curvature and high-speed trajectories, is not considered in this paper.

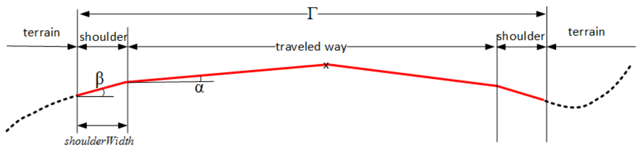

Figure 2: *Road cross section.*

Finally, the surfaces of **intersections** have two important constraints. First, at an intersection, the top-down projection of the traveled way edges connecting two consecutive roads must be close to a circular arc and its connections with road borders are G^1 continuous (Figure 3). Second, engineers often try to smooth the intersection surface, but the smoothness of the connection between two consecutive roads depends on the *importance* level of roads:

- *Seam-line transition* (Figure 3a): If an intersection has 2 major roads, the connection between them is G^1 continuous. Between the major and minor roads is a crease that we call *seam line*.
- *Seamless transition:* Otherwise, the transition between consecutive roads is smooth (see Figure 3b) which is obtained by warping the surface zone between two consecutive roads, whereas the road center still passes through the road axes.

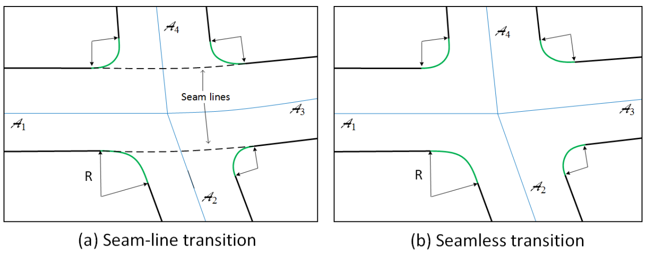

Figure 3: *The top-down projection of two intersection types. The rounded turning edges are painted in green.*

3. Related work

Researchers first addressed the problem of modeling virtual urban roads. The early road networks were based on axes which were generated by either a grammar rule named L-system [PM01] or a pattern-based method [SYBG02]. For more controllable and interactive procedures, in Citygen system [KH07], the user designs the primary roads for a city by placing road intersections, then the trajectories between intersections are automatically generated and adapted to supplied terrain. The secondary-road generation is controlled by a set of parameters. Alternatively, Chen et al. [CEW*08] use a tensor field from which a graph representing the road network is computed. Smelik et al. [STKB11] proposed a declarative procedure to model virtual cities, in which the roads are automatically adapted to the environments thanks to the consistency-maintenance ability of Sketcha-World framework. However, no further constraint of roads is considered as the main goal of authors is just to provide more user control and flexibility.

The lack of reality of these models suggests that real data should be considered. The earliest data sources for road reconstruction were aerial and satellite monochromatic images. The survey paper [FZAW99] summaries the traditional methods on road extraction from images. The important output information of these researches involve the road border and center, the road network topology. Later, the attention in road reconstruction shifted to utilizing LASER and LIDAR data [EV06, BF11] but researchers still put in main effort to consume these kinds of data rather than take into account civil engineering constraints.

The advanced extraction techniques of road information for GIS update, as overviewed in [Men03], has permitted the development of GIS data sources such as OpenStreetMap, Natural Earth, TIGER/Line, Here map, NAVTEQ. As urban areas draw more attention than others, their GIS data sources are often more detailed and precise. Road reconstruction from GIS data seems to be an emerging trend. The roads in [BN08] are reconstructed from a layer of a GIS database but the axes are based on Bézier curves. Wilkie et al. [WSL12] present an approach for large-scale traffic simulation. Their road axes rely on straight lines and circular arcs, and no constraint of surface like road cross section is taken into account.

The most similar work to ours may be [WLS14] as it also aims at reconstructing realistic roads from GIS data, taking many important civil engineering rules. However, when reconstructing the axis for each road, the authors use Hermite curves instead of clothoids to connect straight lines and circular curves which does not correspond to civil engineering rules. Moreover, they neither guarantee smooth transitions at turning edges nor describe how to triangulate intersections which could be complex.

4. Axis network reconstruction

[NDD14] has proposed an algorithm called LSGA for single road axis reconstruction. Its main idea is to seek for an accurate G^1 piecewise curve that fits well the input polyline by growing a sequence of primitive segments. Like in civil engineering approach, the polyline is decomposed into an horizontal and a vertical polylines and each planar polyline must be fit by a sequence of line segments, circular arcs and clothoids for the horizontal curve and line segments and parabola for the vertical curve, both conforming to pre-defined syntaxes. The final curves must be as simple as possible and are finally combined to obtain a 3D road axis. However, that paper has not addressed the continuity at intersections and roundabout axis reconstruction which we tackle in this section.

We first connect related polylines whose corresponding axes must have G^1 connections to form so-called hyper-polylines with the connecting criteria:

- The two roads have the same name, importance, width, number of directions, type (normal road, bridge,...), and number of lanes.
- They are not less important than the others of this intersection.

The hyper-polylines are then classified into two types: (1) *roundabout* if its two ends are coincident, (2) and *normal road* for the remaining. Each hyper-polyline type is fit by a specific routine:

- *Roundabout:*. The top-down projections is a circle while its *vertical alignment* is a plane. Like in [NDD14], a least square fitting method is launched to determine the different coefficients.
- *Normal road:*. In order to obtain a G^0 connectivity of road axes at intersections, we extend LSGA so that the resulting curve passes through two hyper-polyline ends. To fix the starting point, we assign the starting coordinates of the curve to the first point of the hyper-polyline. Parameter-freezing is not, however, applicable for the last point since there is no parameters corresponding to the ending position of the curves. Inspired by the weighted least square method, when optimizing parameters, we just duplicate the last sample point into k instances.

Our resulting roundabout axis (Figure 4a) is a big circle fitting well a closed hyper-polyline. All axes in Figure 4b are connected at intersections. Moreover, thanks to the fact that we apply LSGA to hyper-polylines which consists of related polylines, our process is able to create long G^1 curves passing through these intersections.

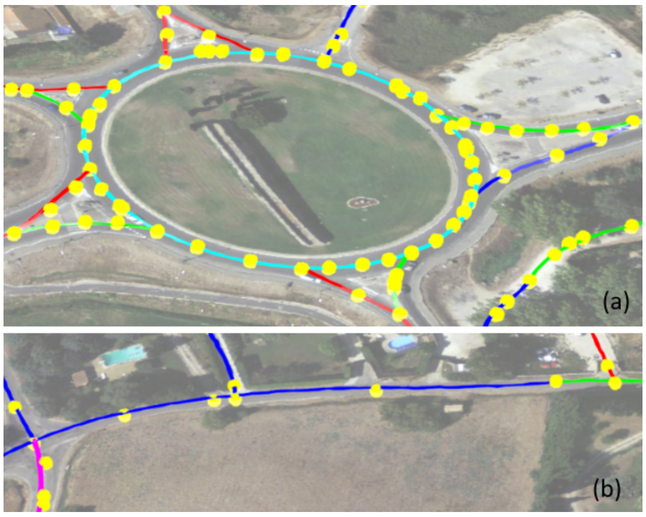

Figure 4: *Complete road-axis network: (a) roundabout, and (b) axes connecting properly at intersections.*

5. Mathematical model

5.1. Zones and notations

This work does not consider trajectories separated from the terrain such as bridges and tunnels. Thus, we can consider the projected domain of our objective road surface as $\Omega \subset \Re^2$, and the remaining region of terrain $\Omega' = \Re^2 \setminus \Omega$ (Figure 5). Since road surfaces are made up of regions with different geometric characteristics, we split Ω into non-overlapping zones:

- **Traveled way** $\hat{\Omega}$ contains two non-overlapping zones: $\hat{\Omega}_T$, the trajectory surface between the intersections or the ends and $\hat{\Omega}_I$, the transition of traveled way between legs of the intersections.
- **Shoulder and blending** $\bar{\Omega}$**.** We call the width of transition from the shoulder to the terrain $w_{shoulder}$. We also divide $\bar{\Omega}$ into $\bar{\Omega}_T$ and $\bar{\Omega}_I$ for trajectories and intersection respectively.

For the ease of reference, we denote the intersection zone $\hat{\Omega}_I \cup \bar{\Omega}_I$ by Ω_I, and the trajectory zone $\hat{\Omega}_T \cup \bar{\Omega}_T$ by Ω_T.

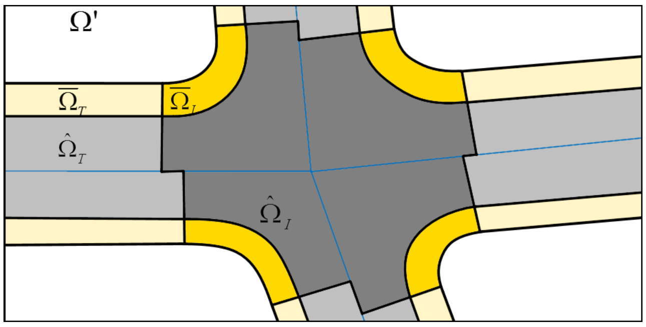

Figure 5: *Terrain (Ω') and different road zones ($\hat{\Omega}_T, \hat{\Omega}_I, \bar{\Omega}_T, \bar{\Omega}_I$).*

The boundaries $A_1B_1, B_1C_1, A_2B_2, B_2C_2$, between zones of two adjacent roads with axes $\mathcal{A}_1$ and $\mathcal{A}_2$ (see Figure 6a) are identified as follows. Let K be the intersection of two traveled-way borders which are the offset of road axes by half width of roads, and φ be the angle between them. On the bisector of φ from K, we identify the turning center J so that: $KJ = R/sin(\varphi/2)$ where R is the turning radius regulated by civil engineering. We project J onto axes to get A_1 and A_2, onto borders to get B_1, B_2, C_1, C_2.

The turning edge B_1B_2 is not a perfect circular arc because the road axes are not absolutely straight meaning $||JB_1|| \approx ||JB_2||$. We approximate the 2D circular arc by an Hermite segment whose two ends are B_1 and B_2, direction at two ends are given by $\mathcal{A}_1$ and $\mathcal{A}_2$. To ensure our Hermite curve approximates well a circular arc, the magnitude of it tangents are computed as in [Sal06].

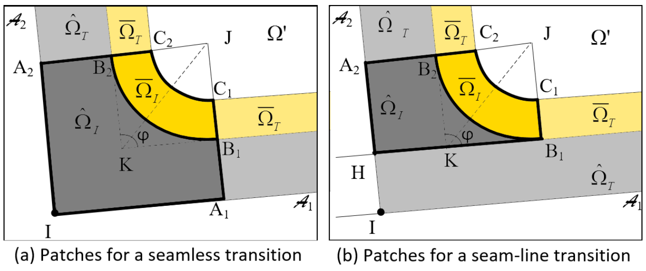

Figure 6: *Zones and patches for two kinds of intersection.*

5.2. Modeling intersection (Ω_I)

While the road surface between two ends (Ω_T) is defined by the road axis and cross section, the intersection zones (Ω_I) are more complex. We employ Coons patches [Coo67, Far88] to define Ω_I between two adjacent roads having axes $\mathcal{A}_1, \mathcal{A}_2$ for three reasons: (1) the boundaries of Ω_I are already specified in subsections 5.1, and (2) Coons patches defined by these boundaries always fit completely to Ω_T, ensuring there is no hole in our surface and (3) the bilinary interpolation of this model is adapted to our problem.

Seamless transition For a smooth transition between roads, we model $\hat{\Omega}_I$ and $\bar{\Omega}_I$ by patches $A_1IA_2B_2B_1$ and $B_1B_2C_2C_1$ illustrated in Figure 6a. Concerning the edges of patches, segments $A_1B_1, B_1C_1, A_2B_2, B_2C_2$ are 3D straight lines and they completely fit to the cross sections which are straight lines as well. We define A_1I, A_2I as parts of road axes $\mathcal{A}_1, \mathcal{A}_2$ respectively. We also define C_1C_2 by an Hermite segment similarly to B_1B_2. Patch $A_1IA_2B_2B_1$ has 5 edges at the glance but we consider line A_1IA_2 as a special edge with a crack at I. This "combined" line A_1IA_2 does not violate any condition of Coons patch.

Seam-line transition Without loss of generality, suppose $\mathcal{A}_1$ be the axis of the major road (see Figure 6b). The reconstructed surface must fit the given main leg axis $\mathcal{A}_1$, and the cross section Γ_1 of the major road is kept throughout the intersection. One can imagine that the part of the intersection surface corresponding to road 1 is created by extruding Γ_1 along $\mathcal{A}_1$. We design two patches $B_1HA_2B_2$ $B_1B_2C_2C_1$ for seam-line transition similarly to the case of seamless transition. The difference is the definition of transition of the traveled way. The patch $B_1HA_2B_2$ is used to model solely the transition from the edge of the major road to the minor road.

The continuities between the patches along the different lines are studied in [Ngu16] and satisfy the requirements.

6. Terrain refinement

The initial coarse mesh created from the input terrain heightmap is not accurate and detailed enough for a road surface (Figure 7a). Therefore, we have to subdivide it (Figure 7b) and then adjust the altitude of it vertices (Figure 7c) so that the final mesh matches.

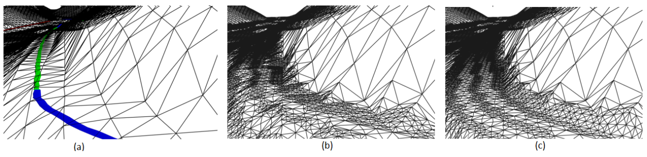

Figure 7: *Mesh evolution at different steps: (a) initial mesh created from the input heightmap, (b) subdivided mesh, and (c) final mesh after adjusting vertex altitudes.*

6.1. Zone detection

In order to modify the mesh, we perform two algorithms to detect vertices inside Coons patches and trajectory zones.

Transition-zone (Ω_I) vertex detection: Each vertex P has a pointer to a Coons patch initialized $patchPointer = NULL$. We process each patch $\mathcal{P}(u,v)$ by the following propagation algorithm:

- *Preprocessing:* we build a quadtree $\mathcal{Q}$ storing the pointers referring to vertices of the current processed mesh.
- *Starting:* we utilize the "*query range*" method of $\mathcal{Q}$ to get all vertices inside a *square* whose center is the central point $\mathcal{P}(0.5, 0.5)$ of the patch $\mathcal{P}$. To ensure that the existing vertices around $\mathcal{P}(0.5, 0.5)$ are found, the edge length of *square* is set to the maximum distance between two consecutive vertices. We then test each vertex of the square until one vertex called *startingV*

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

locating inside $\mathcal{P}$ is found. The inside-patch test is a reverse task: given the 2D vertex coordinate, we use the Newton-Gauss algorithm to numerically compute the corresponding parameters u, v of $\mathcal{P}$. A vertex is inside $\mathcal{P}$ if and only if $0 \leq u, v \leq 1$.

- *Iterating:* once the *startingV* is identified, we push it into an empty *queue*. In each iteration, while the *queue* is not empty, the algorithm removes a vertex from the *queue*, examines the unvisited neighbors of this vertex. If a neighbor sits inside the patch and its $patchPointer = NULL$, it is enqueued and its *patchPointer* is set to $\mathcal{P}$.

Processing all patches will assign the *patchPointer* of P to a relevant patch if any.

Trajectory-zone (Ω_T) vertex detection: Vertices between two ends of each road i are then similarly detected except the center of *square* is the middle point of the road axis.

6.2. *Field* value

For an adaptive subdivision, we define a so-called *field* value at any point $P \in \Omega$ to control the expected mesh resolution. Inspired by the implicit surface technique, we consider that the traveled way edge is an iso-line of the *field* function whose the skeleton is the road axis, and *field* indicates the 2D distance from P to road axis. Thus *field* is negative when p sits inside the traveled way zone ($\hat{\Omega}$), zero on the traveled way edge, and positive outside the traveled way ($\bar{\Omega} \cup \Omega'$). From this specification, we design a *field* function:

$$field(P \in \Re^2) = \begin{cases} d(P, \mathcal{A}_i) - 0.5width(road_i) & \mathrm{P} \in \Omega_T \\ d(P, edge) & \mathrm{P} \in \overset{o}{\bar{\Omega}_I} \\ -\max\limits_{1 \leq k \leq n}(0.5width(road_k)) & \mathrm{P} \in \overset{o}{\hat{\Omega}_I} \\ \infty & \mathrm{P} \in \Omega' \end{cases} \quad (1)$$

where n is the number of roads, $road_i$ is the corresponding road, and *edge* is the traveled way edge of the Coons patch containing P.

6.3. Subdivision

Our objective mesh must have a high resolution at road center, traveled way edges, and transitions between the shoulders and the terrain. Therefore, an edge e is split at each subdivision step if it satisfies one of the following conditions:

- Two vertices of e belong to different zones of road $\hat{\Omega}$ and $\bar{\Omega}$. This condition allows us to split all edges cutting traveled way border.
- Two vertices of e sit at different sides of a road axis. This is detected by using a simple cross product comparison.
- The length of e is bigger than any of two length thresholds corresponding to the *field* values of its two vertices. The length threshold a vertex is determined by its *field* value (Figure 8).

Once selecting edges, we insert a vertex at the median of each edge, and connect it to the existing neighboring vertices.

6.4. Altitude adjustment

We finally adjust the altitude of every vertex $V \in \hat{\Omega}$ to the value defined in Section 5. To obtain a gentle contact between road shoulder and the terrain, we blend the extension region of the shoulder with the original terrain. The blending region involves vertices

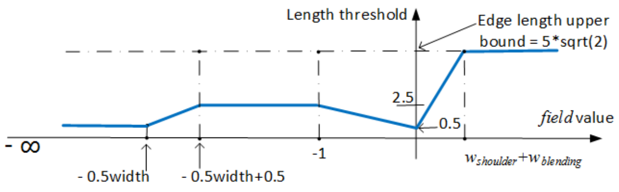

Figure 8: *Edge length threshold function.*

whose distance to the traveled way border ranges from $w_{shoulder}$ to $w_{shoulder} + w_{blending}$. In practice, the value $w_{blending} = 1m$ is sufficient. Let b be the blending coefficient, $z_{theoretical}$ be the altitude of the *theoretical surface*, $z_{subdivided}$ be the altitude of the subdivided mesh, then the final altitude is:

$$z_{final} = z_{theoretical} * b + z_{subdivided} * (1 - b) \quad (2)$$

b is defined based on the *field* value (see Figure 9). If $V \in \Omega'$, the original altitude is kept so b must be 0.

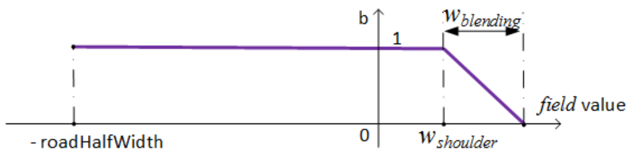

Figure 9: *Blending-coefficient function*

7. Experiments and results

We implemented our method in C++ with CGAL library (http://www.cgal.org) which provides interesting tools for mesh manipulations. Our resulting meshes are textured by the orthographies. We performed experiments on a PC with Intel Core i7 2.8Ghz and 4GB of RAM for an urban area (Figure 10) consisting of 108 roads with a total length of 9634.6m. The meshes along roads have higher resolution than the outside zones. In addition, our road surfaces follow the road axes as the axes do not fly over or hide under the roads. As intersections are the most complicated areas, we analyze two of them in Figures 11 and 12. Since road networks are mainly composed of ruled surfaces, mean or Gaussian curvatures are not relevant. Minimum curvature maps emphasize the results:

- We can see in Figure 11b that the minimum curvature of the four-road seamless intersection at area (1) in Figure 10 is nearly zero, meaning the transitions between roads are smooth. This corresponds to the image from Google (Figure 11a).
- Figure 12 illustrates the three-road seamline intersection at area (2) in Figure 10. In constrast to the seamless one, the minimum curvature between the main roads and the minor one is negative (see Figure 12b) because of the existence of a crease corresponding to a slight valley for water evacuation. The Google Earth image (Figure 12a) and an analysis on site confirm the existance and the geometry of this seamline.

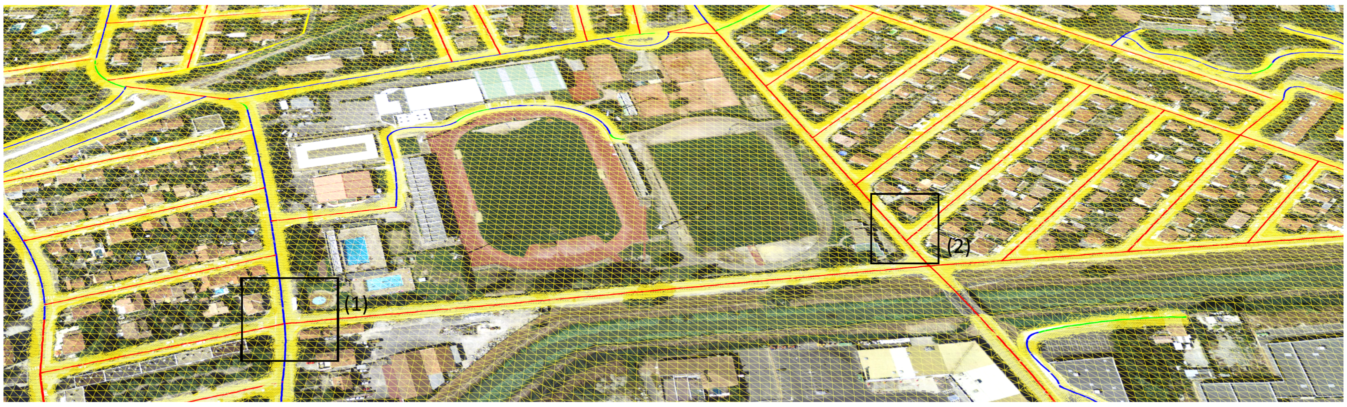

Figure 10: *A resulting urban road network. Areas (1) and (2) are analyzed in Figure 11 and 12 respectively.*

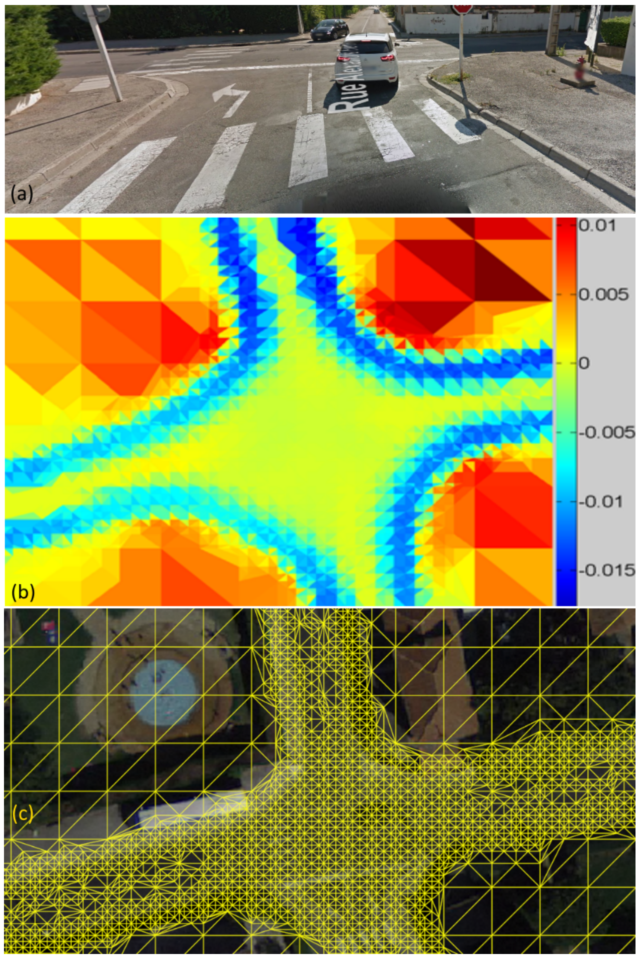

Figure 11: *A seamless intersection: (a) image from Google Earth, (b) top-down view of the minimum curvature diagram, (c) resulting mesh.*

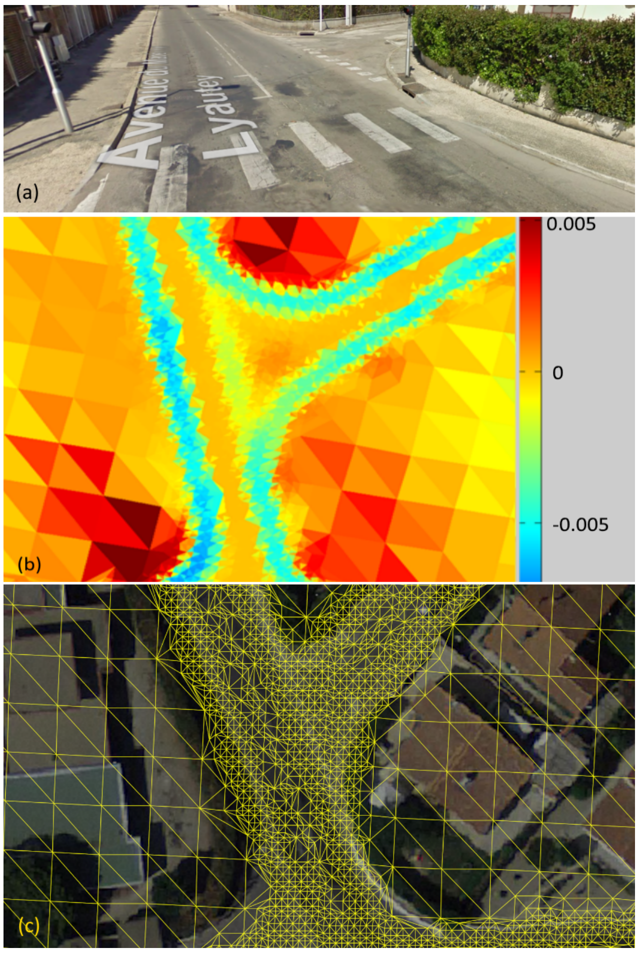

Figure 12: *A seam-line intersection: (a) image from Google Earth, (b) top-down view of the minimum curvature diagram, (c) resulting mesh.*

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

Both Figures 11c and 12c prove that the mesh at road region was subdivided adaptively. The meshes at intersections are homogeneous and dense because their field value is imposed to be $-0.5width$ (see equation 1) so that the corresponding threshold reaches the minimum value according to Figure 8. For the other road parts, the meshes along the road axes and the edges are also dense and rather homogeneous because these roads are narrow, hiding partially the variation of density. Outside the roads and their transitions, the mesh keeps its original resolution (5mx5m).

Concerning the performance of our framework, the statistics of our experiments are reported in Table 1. Our careful investigation reveals that most of the process-time is spent on the field computation task which relies mainly on the computation of 2D distances from vertices to the road axes. Indeed, as we currently use a numerical method to get distance from a vertex to a non-straight segments, our program needs to run several iterations. Although further optimization could improve the performance, we are now able to process large areas.

Measurements	Numbers
Vertices of the original mesh	32767
Vertices of the final mesh	773714
Number of roads	108
Total length or roads (m)	9634.6
Number of Coons patches	310
Processing time (s)	108
Field computation (s)	106
Mesh subdivision (s)	1.8
Altitude adjustment(s)	0.2

Table 1: *Statistics of our experiment.*

8. Conclusion

We have introduced a framework for reconstructing road surface from real GIS data. Our contributions are twofold: (1) a *theoretical surface* in which the road intersection zones are modeled by Coons patches while the trajectories between intersection are given by an equation of the road axis and cross section; and (2) a mesh-displacement process to obtain the corresponding geometry of the *theoretical surface* on the terrain.

The main advantage of our framework is to be able to create a realistic road network in a reasonable time. The center lines of the resulting roads follow the road axes. The principal geometric constraints according to civil engineering rules are respected. This framework allows the development of applications where realistic roads are mandatory.

Although our modeled cross section respects the description of civil engineering documents, the final cross section could be slightly rounded at the road center to get rid of a sharp crest as steamrollers do in the real word. Moreover, as the zone detection step works with x, y coordinates, only single-layer surfaces can be handled. We will address multi-layer surface roads like overpasses, bridges, and tunnels.

References

[AAS11] *A Policy on Geometric Design of Highways and Streets 2011*. American Association of State Highway and Transportation Officials, 2011. 2

[BF11] Boyko A., Funkhouser T.: Extracting roads from dense point clouds in large scale urban environment. *ISPRS Journal of Photogrammetry and Remote Sensing 66*, 6 (2011), S2–S12. 3

[BN08] Bruneton E., Neyret F.: Real-time rendering and editing of vector-based terrains. In *Eurographics* (2008). 3

[CEW*08] Chen G., Esch G., Wonka P., Müller P., Zhang E.: Interactive procedural street modeling. In *ACM SIGGRAPH 2008 Papers* (2008), SIGGRAPH '08, pp. 103:1–103:10. 2

[Coo67] Coons S. A.: *Surfaces for computer-aided design of space forms*. Tech. rep., Cambridge, MA, USA, 1967. 4

[EV06] Elberink S. O., Vosselman G.: Adding the third dimension to a topographic database using airborne laser scanner data. vol. 36, Citeseer, pp. 92–97. 3

[Far88] Farin G.: *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide*. Academic Press Professional, Inc., San Diego, CA, USA, 1988. 4

[FZAW99] Fortier A., Ziou D., Armenakis C., Wang S.: Survey of work on road extraction in aerial and satellite images. *Center for Topographic Information Geomatics, Ontario, Canada. Technical Report 241* (1999). 3

[KH07] Kelly G., Hugh M. C.: Citygen: An interactive system for procedural city generation. *In Proceeding of GDTW 2007 GDTW '07: Fifth International Conference on Game Design, Liverpool* (2007), 8–16. 2

[Men03] Mena J. B.: State of the art on automatic road extraction for gis update: A novel classification. *Pattern Recogn. Lett. 24*, 16 (Dec. 2003), 3037–3058. 3

[NDD14] Nguyen H., Desbenoit B., Daniel M.: Realistic road path reconstruction from gis data. *Computer Graphics Forum 33*, 7 (2014), 259–268. 1, 2, 3

[Ngu16] Nguyen H. H.: *Automatic reconstruction of realistic road networks from GIS data*. PhD thesis, Aix-Marseille University, 2016. 4

[PM01] Parish Y. I. H., Muller P.: Procedural modeling of cities. In *Proceeding of the 28th annual conference on Computer graphics and interactive techniques* (2001), SIGGRAPH'01, pp. 301–308. 2

[Sal06] Salomon D.: *Curves and Surfaces for Computer Graphics*. Springer-Verlag New York, 2006. 4

[SET02] *The design of interurban intersections on major roads*. Tech. rep., Service d'Etudes Techniques des Routes et Autoroutes Centre de la Sécurité et des Techniques Routière, 2002. 2

[STKB11] Smelik R. M., Tutenel T., Kraker K. J., Bidarra R.: A declarative approach to procedural generation of virtual worlds. *Computers & Graphics 35*, 2 (2011), 352–363. 2

[SYBG02] Sun J., Yu X., Baciu G., Green M.: Template-based generation of road networks for virtual city modeling. *Proceeding of the ACM symposium on Virtual reality software and technology* (2002), 33–40. 2

[WLS14] Wang J., Lawson G., Shen Y.: Automatic high-fidelity 3D road network modeling based on 2D GIS data. *Advances in Engineering Software 76*, 0 (2014), 86 – 98. 1, 3

[WSL12] Wilkie D., Sewall J., Lin M. C.: Transforming gis data into functional road models for large-scale traffic simulation. *IEEE Transactions on Visualization and Computer Graphics 18*, 6 (2012), 890–901. 1, 3