Inexpensive Reconstruction and Rendering of Realistic Roadside Landscapes

Carlos Andújar[a], Antoni Chica[a], Miguel Angel Vico[a], Sergio Moya[a], Pere Brunet[a]

[a] *Universitat Politècnica de Catalunya, C. Jordi Girona, 1-3, 08034 Barcelona, Spain*

Abstract

In this paper we present an inexpensive approach to create highly-detailed reconstructions of the landscape surrounding a road. Our method is based on a space-efficient semi-procedural representation of the terrain and vegetation supporting high-quality real-time rendering not only for aerial views but also at road level. We can integrate photographs along selected road stretches. We merge the point clouds extracted from these photographs with a low-resolution digital terrain model through a novel algorithm which is robust against noise and missing data. We precompute plausible locations for trees through an algorithm which takes into account perceptual cues. At run-time we render the reconstructed terrain along with plants generated procedurally according to precomputed parameters. Our rendering algorithm ensures visual consistency with aerial imagery and thus it can be integrated seamlessly with current virtual globes.

Keywords: terrain reconstruction, terrain rendering, landscape modeling

1. Introduction

In the last decades we have witnessed significant improvements in digital terrain modeling, mainly through photogrammetric techniques based on satellite and

Email addresses: `andujar@lsi.upc.edu` (Carlos Andújar), `achica@lsi.upc.edu` (Antoni Chica), `mvm9289@gmail.com` (Miguel Angel Vico), `smoya@lsi.upc.edu` (Sergio Moya), `pere@lsi.upc.edu` (Pere Brunet)

aerial photography, as well as terrestrial laser scanning (Kraus, 2007). These techniques allow the creation of Digital Terrain Models (DTM) that can be streamed over the network and explored through virtual globe applications. In rural areas, the environment surrounding roads is of special interest in a number of digital map applications. Publicly available DTMs and associated orthophotos are suitable for rendering aerial views of the environment but fail to provide realistic appearance at ground level for several reasons. First, the typical DTM resolutions nowadays (5, 10 and 30 meters) are not enough to capture sharp slope changes due for example to carved roads in mountains. Higher resolution DTMs do exist, but only for some selected places (often urban areas). Second, DTMs do not capture the geometry of the vegetation on top of the terrain, causing the vegetation to lack 3D appearance when exploring the terrain at grazing angles. Finally, orthophotos do not show the color of the terrain in regions covered by trees. The typical approach of texture mapping the terrain with the orthophoto color becomes unacceptable as the viewpoint gets closer to the ground.

These facts have motivated the use of model acquisition techniques from terrestrial sensors to accurately model terrain and vegetation features. Multi-view stereo techniques (Seitz et al., 2006) are particularly attractive as they can be combined with commodity equipment such as vehicle-mounted cameras to provide an inexpensive digitizing solution. However, the detailed reconstruction of natural environments is particularly challenging. Unlike man-made objects, typical terrain models lack well-defined features and exhibit high levels of self-similarity which hinder the reconstruction process. Moreover, the acquisition of outdoor environments often relies on a constrained set of positions for the sensor device (e.g. along the road), which provides a partial model of the environment. This often results in noisy, sparse, unevenly sampled data providing a limited-coverage model of the environment.

In this paper we propose a new approach to render realistic images of the landscape surrounding a countryside road (Figure 1). Our approach meets the following requirements:

- *Inexpensive data acquisition.* The acquisition does not require special digitizing requirement. Tasks involving the user are kept to a minimum.
- *Reconstruction completeness.* We aim at providing convincing visual complexity of the environment for both aerial and road-level views. This implies that the vegetation on top of the terrain must be reconstructed appropriately.
- *Visual consistency with publicly available data.* We want our approach to integrate seamlessly with current virtual globes. Therefore the reconstructed landscape must be visually consistent with user-provided DMTs and aerial imagery. Rendered landscapes should look like the user-provided DTM and orthophoto for distant aerial views, but have to gain plausible detail and 3D appearance as the camera gets closer to the road.
- *Space efficiency.* The output model is compact enough to support network streaming.
- *Real-time rendering.* Our approach supports real-time realistic rendering with commodity hardware.

Our approach benefits from a collection of images of the landscape surrounding the road (referred to as photographs) captured by the user for example from a car-mounted camera. These photographs, if available, are beneficial to provide an accurate reconstruction of the terrain next to the road, specially in mountainous areas. The collection of photographs is not required to be complete, as this would be unfeasible for a vehicle-mounted camera due to visibility and accessibility problems.

The reconstruction proceeds through the following steps (Figure 2). We use a segmented orthophoto where pixels representing vegetation have been classified into the following three categories: trees, shrubs and herbs. Some possibilities for automatically detecting and classifying vegetation pixels in the orthophoto are outlined in Section 6. These vegetation types are used to create plants at plausible locations on top of the terrain. A critical step is the placement of trees at locations visually consistent with the orthophoto, as this would affect the location of the tree trunks and the crown shape. When photographs are available, we first extract a point cloud from the photographs by detecting small series of overlapping photographs and then using multi-view stereo reconstruction. Then we use a robust algorithm to integrate the user-provided DTM and road model with the high-resolution reconstructed data. This results in a new DTM significantly more faithful to the actual terrain. At runtime, vegetation is created and rendered on-the-fly according to the vegetation type associated to each orthophoto pixel. The user is allowed to adjust a few parameters (such as average height and size of the species) controlling the procedural generation of leaves. We also use a fractal perturbation approach to add visual detail at close-up views by perturbing both color and terrain displacement values. Our rendering algorithm also handles the rendering of the terrain covered by trees and uses an efficient technique to simulate complex shadows cast by the foliage onto the terrain. To the best of our knowledge, the pipeline we present in this paper is the first one encompassing all the elements above to create realistic reconstructions of roadside landscapes.

2. Previous work

Terrain modeling and surface reconstruction. The creation of large-scale DTMs is accomplished nowadays by means of photogrammetric techniques based on satellite and aerial photography, as well as terrestrial laser scanning. We refer the interested reader to (Kraus, 2007) for a complete review. Multi-view stereo schemes (Seitz et al., 2006; Furukawa and Ponce, 2010) provide an inexpensive acquisition alternative by using arbitrary collections of partially overlapping images to create oriented point clouds. However, converting such point clouds into a plausible surface is a challenging task in the presence of partial data (Gross and Pfister, 2007). Methods based on isosurfaces of implicit functions can be either local (Alexa et al., 2001) or global (Kazhdan et al., 2006). Unfortunately, these methods fail when the input data covers a small portion of the real surface. Our proposed reconstruction algorithm combines point samples from multi-view stereo with a low-resolution DTM to overcome these limitations.

Fractal-based modeling of terrains. Many natural scenes and geological structures often exhibit self-similarity over some range of scales (Mandelbrot, 1983). Based on this observation, a number of approaches for fractal-based modeling of artificial terrains have been proposed over the last decades (Ebert et al., 2003; Dachsbacher, 2006). The most common approach is to generate the altitude values of a procedural terrain using either polygon subdivision plus midpoint perturbation (Miller, 1986), or noise synthesis (Perlin, 1985; Saupe, 1989). Current procedural approaches for terrain modeling create artificial landscapes with impressive realism (Ebert et al., 2003; Schneider et al., 2006; Dachsbacher, 2006;

Dachsbacher and Stamminger, 2004). However, despite offering a plethora of parameters for terrain synthesis, generating artificial terrains reproducing a real acquired model at multiple scales is a very time-consuming task. We use fractal techniques to add consistent visual complexity to digital models of real terrains rather than creating artificial terrains from scratch.

Estimation of fractal properties. Estimation of fractal properties is of great interest in nature sciences. A number of methods have been proposed to compute the fractal parameters of self-affine signals, including dispersion analysis, correlation analysis and power spectral analysis (see (Schepers et al., 1992) for a comparison). The roughness-length method (Malinverno, 1990) is based on measuring the standard deviation $S(w)$ of the data considering the signal within windows of varying sizes. We adopted this method to estimate multiple fractal parameters of the terrain.

Terrain appearance. Surface textures acquired from the real-world or computed using procedural models are crucial for photorealistic terrain rendering. We refer to Dachsbacher PhD thesis (Dachsbacher, 2006) for a review on procedural terrain textures. A widely used scheme is texture splatting, i.e. blending a set of tileable textures (e.g. rock and grass) according to some criteria. Dachsbacher et al. (Dachsbacher and Stamminger, 2005) create terrain textures on the fly by compositing procedural surface layers according to a set of constraints based on height and slope ranges. This method has been extended to include rainfall, solar radiation, and temperature data (Dachsbacher et al., 2006). Our procedural model builds on these ideas but perturbs chromacity in addition to bumpiness. Example-based texture synthesis provides a powerful tool to complete images with arbitrarily large holes (Drori et al., 2003; Hays and Efros, 2007).

High-resolution exemplars can be used to synthesize images of arbitrary resolution (Hertzmann et al., 2001; Freeman et al., 2002). Our procedural scheme for soil synthesis relies on a few exemplars but these are used only at construction time rather than at rendering time, making our approach more suitable for network streaming. For close views, vegetation and small scale detail, e.g. rocks, is often represented geometrically (Deussen et al., 2002; Deussen and Lintermann, 2005; Colditz et al., 2005).

Forest rendering. Realistic real-time lighting and rendering of forests is a challenging problem due to the large number of trees in aerial forest views. Common representations include billboards, 3D textures (Decaudin and Neyret, 2004; Behrendt et al., 2005), point clouds (Gilet et al., 2005) and texture lobes (Livny et al., 2011). The problem we address though is conceptually different as the forest geometry and appearance is already encoded in the DTM and orthophoto, although at low resolution. Therefore we do not care about rendering distant trees, nor about simulating realistic lighting (Bruneton and Neyret, 2012), but instead we aim to render plausible vegetation consistent with the existing data. Therefore variability and adaptability are key factors of our vegetation rendering approach.

Ground-based and airborne acquisition merging. There is also work on merging aerial laser scans and ground-based acquisition for cities. In particular, (Früh and Zakhor, 2003) present a system capable of acquiring 3D geometry and texture data from urban environments using 2D laser scanners in horizontal and vertical configurations. The data is collected continuously to be later processed offline, where Monte-Carlo Localization is used to reduce cumulative positional error. As a result, both the scan points and texture images are registered to airborne data, which eases the fusion of the captured data. Similarly, (Wang et al., 2006) uses ground-based panoramas combined with aerial images to reconstruct urban models. The user outlines the building footprints in the orthorectified aerial image and a panorama is registered to the aerial image.

3. Terrain reconstruction

The terrain reconstruction algorithm takes as input a DTM, an orthophoto and a 3D textured model of the road. The DTM is assumed to provide only a rough description of the terrain, as typical DTM resolutions for rural areas do not capture well the terrain slope next to the road (Figure 4). We assume that typical road vector data (centerlines, roadsides, crossing nodes) have been converted into a 3D model of the road. Simple automatic algorithms for such a conversion do exist (Section 6). We put no constraint on the user-provided road model other that roadside curves must be available. The terrain reconstruction steps are described in detail below.

3.1. Point cloud generation through multi-view stereo

This step is only performed when photographs are available. Photographs are processed to correct distortions using PtLens. Subsets of n_p consecutive photographs (we used $n_p = 12$) are used for the reconstruction of partial point cloud models, see Figure 5. We create a partial point cloud for every subset of n_p consecutive photographs, and we use an overlap of 2 photographs between neighbor photo subsets to ensure overlaps between consecutive partial point clouds for registration purposes. In other words, the first partial point cloud is reconstructed from photographs $1..n_p$ while the second partial point cloud is generated from photos $(n_p - 1)..(2 * n_p - 2)$. Camera parameters and initial sparse reconstructions are computed with Bundler (Snavely et al., 2006). We then use multi-view stereo (Furukawa and Ponce, 2010) and its Patch-based (PMVS) implementation for point cloud generation. PMVS methods work very well with scenes with a sufficient number of well-defined features (e.g. indoor scenes or outdoor scenes in urban areas), but exhibit lower performance in natural environments including self-similar vegetation, where the correspondence problem is less robust. Alignment of PMVS point clouds with the DTM data is based on roadside curves and on the road centerline. Roadside and certerline curves are detected in the partial point clouds generated by PMVS, based on its white color. They are also available on the orthophoto which we assume DTM-registered. Point cloud registration is then addressed as a 1D problem, the only degree of freedom being the location of each partial point cloud along their guiding roadside curves. This is a simple registration problem which can be efficiently solved by the Iterative Closest Point (ICP) algorithm. The output of this step is a registered collection of point clouds $\mathcal{P}$ (Figures 3 and 5).

3.2. Reconstruction of the terrain mesh

This step merges in an error-bounded way the low-resolution elevation data in the DTM with the 3D road model and the high-resolution 3D geometry in the reconstructed point clouds, in roadside areas with positive elevation with respect to the road. We also assume that, in these areas, the terrain is locally monotonic with slopes facing the road and without local hollows. We consider this as a reasonable assumption for slopes captured from the road, as non-monotonic terrain areas in the road side cannot be acquired due to occlusion. The algorithm starts by initializing a terrain elevation model E given by a uniform quadrilateral decomposition of the horizontal domain of the terrain. E is finer than the initial DTM. Initially, the cells $E[i, j]$ contain two elevation values: $h_t[i, j]$ (computed by linear interpolation of the DTM values), and $h_c[i, j]$ computed as follows. When photographs are available, let e_{ij} be the average elevation of the points from $\mathcal{P}$ within the prism over the $[i, j]$ cell, and let ε be a user-defined error bound. In this case, the elevation $h_c[i, j]$ is initialized as

$$h_c[i, j] = \begin{cases} \min(h_t[i, j] + \varepsilon, e_{ij}), & \text{if } e_{ij} \geq h_t[i, j] \\ \max(h_t[i, j] - \varepsilon, e_{ij}), & \text{if } e_{ij} < h_t[i, j]. \end{cases} \tag{1}$$

Some cells will have an initial nil value for $h_c[i, j]$ as e_{ij} can be undefined due to missing data. When photographs are not available we initialize $h_c[i, j] = h_t[i, j]$. Moreover, the cells $E[i, j]$ also contain distances $d[i, j]$ computed as Chamfer distances (Bailey, 2004) to the road axis. The algorithm assumes that the terrain is locally monotonic: for every $[i, j]$, $h_c[i, j]$ must be greater than the $h_c[k, l]$ value of the cells in its 1-ring such that $d[k, l] < d[i, j]$, and $h_c[i, j]$ must be smaller than the $h_c[k, l]$ value of the cells in its 1-ring such that $d[k, l] > d[i, j]$ (the 1-ring of $[i, j]$ contains 8 cells). The algorithm proceeds by flooding E cells in areas of positive terrain elevation, starting from the road and in increasing distance order (a cell $[i, j]$ belongs to a positive terrain elevation area if $h_t[i, j]$ is greater than the value h_t of the closest point on the road axis). When flooding a cell $[i, j]$, non-nil h_c values from neighboring cells with lower distance value are considered, and $h_c[i, j]$ is updated (or created, if $[i, j]$ had a nil value) to ensure monotonicity while guaranteeing the error bound $|h_t[i, j] - h_c[i, j]| \leq \varepsilon$.

After having estimated h_c values for all cells in regions with positive terrain elevation, terrain height is computed as a linear interpolation between $h_t[i, j]$ and $h_c[i, j]$. We consider interpolation regions along the road, their width being defined by the parameter w. Interpolation regions are only defined with positive terrain elevation, and include all points in these areas with a distance d to the road axis between w_r and $w_r + w$, where w_r is the road width (distance between the centerline and one of the roadside curves). Then, for any cell belonging to an interpolation region, the interpolation parameter α is computed as $\alpha = (d - w_r)/w$. The height $h_t[i, j]$ is only recomputed within interpolation regions, and if $0 < \alpha < 1$. In this case, $h_t[i, j] = \alpha h_t[i, j] + (1 - \alpha)h_c[i, j]$. After terrain height interpolation and as a last step, we smooth $h_t[i, j]$ by applying a number of iterations of a standard Laplacian filter, to smooth h_c values and reduce discontinuities between cells with point data and cells with missing data. After several experiments, we observed that plausible reconstructions can be produced with a number of Laplacian iterations between 50 and 100, the terrain surface beginning to be too flat above 500 iterations (Figure 7).

Figure 6 shows a reconstruction example. The low-resolution DTM (Figs. 6a,e) fails to capture well the slope of the terrain and thus deviates significantly from the high-resolution point-cloud (Figs. 6b,f) which reveals a nearly-vertical wall in the roadside terrain. Our reconstructed mesh with $w = 5$ meters and 100 Laplacian iterations (Figs. 6c,g) inherits the shape reconstructed from the captured photographs in the areas close to the road while reducing the effect of spurious vegetation points in the reconstructed PMVS meshes, and adapts to the DTM in other areas.

3.3. Estimation of fractal parameters

Fractal parameters will be used during rendering to provide plausible detail of the terrain at close-up views. We assume that the local displacement values of the terrain exhibit a fractal behavior characterized by amplitude A_d and roughness H_d values (Saupe, 1989) such that the standard deviation $S_d(w)$ of displacement values within a sampling window of size w follows the power law $S_d(w) = A_d w^{H_d}$, where A_d is a proportionality constant and $0 < H_d < 1$ is the Hurst exponent. Although in practice A_d and H_d vary across the terrain, according to our experiments using the same A_d and H_d values for the whole terrain also yields plausible results. We estimate H_d and A_d as the slope and intercept of the regression line of a plot of $\log S_d(w)$ vs. $\log w$ (Malinverno, 1990) using the 3D point samples within a user-provided 3D window.

We also compute color fractal parameters that will be used during rendering to provide plausible detail to the terrain color. We work on YCoCg color space where Y represents luminance and Co, Cg are chromacity values. We will assume that these values also exhibit a fractal behavior characterized by amplitude and roughness values. We compute these values A_y, H_y, A_o, H_o, A_g, H_g using the roughness-length method as described above, taking the standard deviation of YCoCg components within a user-provided image.

4. Vegetation distribution

The vegetation types in the segmented orthophoto are used to compute plausible locations for the trees and shrubs. The output is a collection of 3D points corresponding to the center of the plant projected on the ground that we call *tree buffer*. No additional vegetation geometry is created at this time; vegetation will be created procedurally on-the-fly according to the simple leaf distribution model described below.

4.1. Leaf distribution model

A tree crown can be described using a tuple $\{h, t(r), b(r), d\}$ where h is the height of the crown center and $t(r)$ and $b(r)$ are functions giving the height of the crown top (resp. bottom) at distance r from the tree axis. Functions $t(r)$ and $b(r)$

describe the upper and lower profiles of the crown (see Fig. 8), which we assume to have approximate axial symmetry. For example, a spherical crown would be described by $t(r) = b(r) = \sqrt{1 - r^2}$ if $0 \leq r \leq 1$ and $t(r) = b(r) = 0$ elsewhere. The parameter d expresses the leaf density, which we assume to be constant within the crown. Let P be a vertical prism at a distance r from the tree axis, and let a be the area of the 2D footprint of P. The volume of P is thus $v = a(t(r) + b(r))$, and P has $n = d \cdot v$ leaves with heights in the interval $[h - b(r), h + t(r)]$. In order to introduce some randomness in the crown shape, we perturb $t(r)$ and $b(r)$ using a noise function.

A family of tree crowns can be characterized by average and standard deviation values for h and d along with the $t(r)$ and $b(r)$ functions. We use a small vegetation dictionary where each entry holds μ_h, σ_h, μ_d, σ_d, $t(r)$ and $b(r)$ for a few representative plant species according to the local vegetation in the area to be reconstructed. Each vegetation type also holds a small collection of representative leaf textures. Textures can represent single leaves or groups of leaves; in the latter case the values of μ_d and σ_d must be reduced accordingly.

4.2. Shadow-based adjustment

The segmented orthophoto might include misclassified vegetation pixels. Since we will render the vegetation with a color visually consistent with the user-provided orthophoto, we have observed that the effect of misclassification errors in the output images is not perceptually important. An exception is when the shadows cast by trees provide a clear perceptual cue about the presence of such trees. If these pixels are misclassified as shrub or herb, the resulting image will exhibit inconsistent shadows. We use the shadows present in the orthophoto to improve the vegetation classification. We first label pixels in the orthophoto as shadow if their luminance is below a certain threshold. We then apply a morphological opening operation to the set of shadow pixels to reduce the effect of noise. The result is a binary image I that contains an estimation of the shadows cast by high vegetation. Since the inclination of the Sun introduces a relative shadow displacement, we translate I pixels by the corresponding 2D vector. This 2D vector could potentially be extracted from the orthophoto, but we introduce this value manually for each orthophoto that must be segmented. Finally the corresponding vegetation pixels in the orthophoto are labeled as tree type. This will promote misclassified shrubs and herb pixels to tree type in certain areas (Figure 9). Notice that only pixels already classified as vegetation will be affected by this step.

4.3. Tree placement

A key step is the placement of trees at locations visually consistent with the orthophoto, as this would affect the location of the tree trunks and the crown shape. We use a dart throwing algorithm to place trees within those areas classified as trees, satisfying the condition that no pair of trees are closer to each other than a certain minimum distance (6 m for the trees in our examples). We store the resulting positions in a tree buffer and compute a discrete distance field (with the same resolution as the orthophoto) where each pixel stores its distance r to the nearest tree axis.

5. Rendering

The runtime representation consists of the following data: the reconstructed terrain mesh, the estimated fractal parameters, the tree buffer, the tree distance field, the parameters for each vegetation type, and the initial user-provided data -road and orthophotos- (Figure 10). The vegetation type and the distance field are encoded in a single-component texture with 2 and 6 bits, respectively. The following sections describe the rendering algorithm using OpenGL notation.

5.1. Terrain rendering

Since our reconstructed terrain is a conventional triangle mesh, it can be rendered in a variety of ways including ROAM meshes (Duchaineau et al., 1997), Geometry Clipmaps (Losasso and Hoppe, 2004; Asirvatham and Hoppe, 2005), and Batched Dynamic Meshes (Cignoni et al., 2003; Gobbetti et al., 2006), all of which provide view-dependent level of detail management. Here we focus only on the GPU shaders we use to compute the final terrain color at fragment level.

Two colors are available for each terrain fragment: the one from the input orthophoto c_o, and a synthetic one c_t created through fractal noise from a user-provided exemplar. Recall that c_o often does not represent the terrain color e.g. in those areas covered by vegetation. We compute the base terrain color as the linear interpolation $c_b = (1-t)c_o + tc_t$, where $t = \text{clamp}(\rho, \alpha, 1.0)$, $0 \leq \rho \leq 1$ is a per-fragment interpolated value indicating the presence (1) or absence of vegetation (0), and $0 \leq \alpha \leq 1$ a user-defined parameter expressing the strength of the synthetic color in areas not covered by the vegetation. As α approaches 1, the base color is synthetic even in areas not covered by vegetation. We used $\alpha = 0.7$ for the *Arrabassada* dataset and $\alpha = 0.0$ for the rest of models shown in Section 6.

We enrich the terrain geometry with fractal detail. We use Saupe's fractal evaluation method (Saupe, 1989) to compute a surface displacement value along the normal direction, and then use Perlin's approach (Perlin, 1985) to perturb the per-fragment normal. The fragment is then lighted using the perturbed normal. For consistency with the orthophoto, we use a single directional light.

Standard techniques such as shadow mapping can be used to simulate the shadows cast e.g. by the vegetation onto the terrain. However, shadow mapping assumes opaque geometry and thus it would cause top leaves to cast hard shadows onto everything below. Instead we use a more efficient approach for generating high-resolution shadows. Let $\mathbf{x}$ be the world space coordinates of the terrain fragment being processed, and let c be the fragment color after lighting. The attenuated color is computed as $c_a = \mathrm{mix}(c, c \cdot t(\mathbf{x}), \rho(\mathbf{x}))$, where $\rho(\mathbf{x})$ indicates the presence of vegetation and $t(\mathbf{x})$ is a luminance value sampled from a predefined periodic texture representing the shadow cast by multiple leaves. This results in highly detailed shadows (see accompanying video).

5.2. Vegetation rendering

Vegetation leaves are rendered using OpenGL 4 tessellation and geometry shaders. We first sample the orthophoto using a regular grid. For each grid patch we use the vegetation dictionary to retrieve the parameters associated to its dominant vegetation type and then we sample the vegetation parameters h and d. Next, we create *n leaf seeds* randomly distributed above the patch according to our tree model. The number of leaf seeds is computed as $n = a \cdot d \cdot k \cdot (t(r) + b(r))$ (Section 4.1), where $0 \leq k \leq 1$ is a user-defined value indicating the number of initial seeds above the patch. Leaf seeds are generated in the CPU and stored in a vertex buffer object. We use the ability of tessellation engines for refining geometry to replicate seeds in a view-dependent manner. The tessellation evaluator shader distributes new leaves around the seeds according to our crown model. A geometry shader is used to create a texture-mapped billboard rectangle for each final leaf. Unlike detailed polygonal models of trees (Neubert et al., 2011) our approach is model-independent and thus compatible with the randomness of procedurally-generated instances.

Each leaf is initially assigned the orthophoto color. The fragment shader modulates this color by sampling the luminance from one of the typical leaf textures associated to the vegetation type (Figure 8), while preserving chromacity values. Since the leaf textures are used to perturb the vegetation color sampled from the orthophoto, we adjust their histograms so that they all have the same average luminance. This allows us to add detail to the vegetation without introducing a noticeable color shift with respect to the orthophoto (see Figure 14). Finally, self-shadowing effects are approximated by attenuating the color according to the distance from the top of the crown $t(r) - y$.

Trunks are often not visible in the orthophoto as they are covered by vegetation. Nevertheless, rendering trunks is key for achieving a plausible rendering of trees in non-zenithal views of the scene. Generation of realistic trunks (Runions et al., 2007) is out of the scope of this paper; we just instantiate a few predefined trunk models conveniently scaled to match the tree parameters.

6. Results and Discussion

6.1. Test datasets

We tested our algorithm on five different datasets representing a variety of Mediterranean landscapes. Table 1 summarizes the input data available for each dataset. The 50 cm and 25 cm orthophotos are shown in Figure 11. The *Arrabassada* dataset contains a carved road which is poorly represented in the DTM due to the abrupt slope changes. For this dataset we applied the complete reconstruction pipeline depicted in Figure 2 using road-level photographs (Fig. 5) captured from a car moving along the road. The initial set had roughly one image every 1.6 m.

Dataset	Size	DTM	Orthophoto	Photographs	Road model
Arrabassada	$128{,}000\text{m}^2$	5m	50cm/px	380	900m
Andorra	$810{,}000\text{m}^2$	5m	25cm/px	0	1,200m
Montserrat	$810{,}000\text{m}^2$	5m	25cm/px	0	-
Roca del Corb	$810{,}000\text{m}^2$	5m	25cm/px	0	-
Garraf	$810{,}000\text{m}^2$	5m	25cm/px	0	180m

Table 1: Main features of the test datasets

We used a subset of 380 images. The other datasets either contain non-asphalt trails (*Montserrat*, *Roca del Corb*) or paved roads reasonably well represented in the DTM (e.g. the road going through the valley in the *Andorra* dataset). For these datasets we used our simplified version of the reconstruction pipeline with no road-level photographs and no multi-view stereo, as road-level photographs were not available in these cases.

The input road vector data for the *Arrabassada*, *Andorra* and *Garraf* datasets was a polyline defining the road centerline. Several control points were edited to improve its fitting to the orthophoto, and the road width was set manually. Clothoids could have been used to improve the accuracy of the road model, but the manual fitting was considered sufficient for the purpose of our tests. The generation of the road geometry was straightforward for the road stretches considered (see e.g. Fig. 4). More complex road networks with crossings, bridges and proper road marking can be handled in an automatic way (Bruneton and Neyret, 2008).

The test hardware was an Intel Core I7 960 PC equipped with an NVidia 570.

Dataset	PMVS	Terrain reconst.	User time	Overall
Arrabassada	16 min	6 s	8 min	24 min
Andorra	N/A	<1 s	2 min	2 min
Montserrat	N/A	<1 s	65 s	65 s
Roca del Corb	N/A	<1 s	40 s	40 s
Garraf	N/A	<1 s	40 s	40 s

Table 2: Construction times for the test datasets

6.2. Preprocessing and reconstruction

The orthophoto was first segmented into vegetation and non-vegetation pixels. For this task we used the Normalized Difference Vegetation Index (NDVI) (Kriegler et al., 1969; Roettger, 2007). Each vegetation pixel was further classified into trees, shrubs and herbs. We selected a few portions of the orthophoto showing each vegetation type and used as signature the mean HSL color of the vegetation together with the inverse covariance matrix of the pixels. Using this signature, each vegetation pixel p in the orthophoto was assigned the vegetation type closest to p in the Mahalanobis distance (Mahalanobis, 1936) sense. We found this first-order descriptor to be reasonably accurate for our test orthophotos, see Figure 11. More complex scenarios can be segmented using classification techniques based on texture features (Ruiz et al., 2004; Balaguer et al., 2010).

Construction times are reported in Table 2, including point-cloud generation (PMVS), terrain reconstruction, and user time. User tasks included the road fitting and the selection of exemplars. Note that most of the time was spent in the

Dataset	Terrain	Road	Orthophoto	Vegetation	Overall
Arrabassada	4.7 MB	97 KB	4 MB (1024×1024)	1.9 MB	10.7 MB
Andorra	6 MB	145 KB	49 MB (4096×4096)	6.4 MB	61.4 MB
Montserrat	6 MB	-	49 MB (4096×4096)	5.0 MB	60.0 MB
Roca del Corb	6 MB	-	49 MB (4096×4096)	5.3 MB	60.3 MB
Garraf	6 MB	200KB	49 MB (4096×4096)	2.7 MB	57.7 MB

Table 3: Size of reconstructed models

Bundler and multi-view stereo step, which can be omitted whenever the DTM representation of the road-side terrain is considered acceptable.

Table 3 reports the size of the reconstructed models. Despite the *Arrabassada* dataset had the smallest extents (Table 1), its terrain mesh is much denser due to the 50 cm voxel size used for its construction.

6.3. Visual results

The reconstructed *Arrabassada* model is shown in Figure 12 and in the accompanying videos. Figure 13 shows a visual comparison with the input DTM. Whereas for zenithal views our approach reproduces faithfully the orthophoto data, the benefits of our approach become evident at grazing angles and extreme zoom levels. The final terrain mesh is more accurate than the DTM in regions close to the road (Figure 4).

Figure 15 shows the results for the *Andorra* dataset. This dataset features a rich variety of plant species which caused a less accurate classification of the vegetation. Our tree placement algorithm assumes a roughly constant separation between tree trunks. In areas with a rich variety of tree species and a large range of tree sizes this might result in some small trees reconstructed as a unique tree and, conversely, large trees reconstructed as multiple smaller trees. These artifacts might be noticeable if rendered images are compared side-to-side with the orthophoto (see accompanying *Andorra* video). Notice that the DTM was not adjusted to the road and that no road-level photographs were used for this model. This resulted in a somewhat unrealistic bank angle which can be noticed in Figure 15, view 2, and in the accompanying videos.

Renderings of the *Montserrat* and *Roca del Corb* models are shown in Figures 16 and 17. The corresponding orthophotos exhibit large shadow areas which result in a reconstructed model with slightly darker appearance due to double-lighting effects. Since dry vegetation is hard to detect with our first-order descriptor, some dry vegetation pixels are misclassified as terrain, causing some inconsistencies which can be detected if compared with the input images (Figure 17, view 3). Despite these limitations, our reconstructed model provides a plausible reconstruction with richer details and realistic silhouettes.

The reconstructed Garraf model is shown in Figure 18. The NDVI descriptor succeeded in classifying quarry pixels as terrain pixels, providing a clear and realistic view of the terrain at multiple levels of scale (see view 1 and the accompanying video). This model also exhibits some regions with sparse vegetation (mountain top in view 1) which illustrates the critical nature of the tree placement algorithm. Again, the benefits of our approach are more evident at close-up views of the terrain (view 3).

Figure 19 shows real and reconstructed images of two different landscapes in order to test the plausibility of our reconstructions. Although the time of the

Dataset	Veget. coverage	Terrain (ms)	Vegetation (ms)	Framerate (fps)
Andorra	72%	2.0/14.1	8.9/10.1	92/41
Montserrat	54%	1.4/10.6	6.9/9.2	120/51
Roca del Corb	58%	1.6/11.2	6.8/9.9	119/47
Garraf	30%	0.8/3.6	8.3/9.2	110/78

Table 4: Rendering times for the test datasets. Performance times refer to a distant view (first value) and a close-up view (second value).

day, the daylight, the season and the camera locations are not identical, our reconstructions are capturing the structure of the landscape by producing plausible visualizations which can be clearly identified as belonging to each of the two specific places.

6.4. Rendering times

Table 4 shows the rendering times on a 1280×720 viewport. For close-up views (we use no occlusion culling) the most expensive step is the rendering of the vegetation leaves, which is affected by the percentage of orthophoto pixels classified as vegetation (second column of Table 4). As expected, the slowest speed is achieved with the *Andorra* model (40 fps on a typical close-up view) with about 72% of vegetation coverage. Conversely, *Garraf* was the fastest to render (78 fps) with 30% of coverage. Vegetation rendering can be accelerated by using leaf textures representing larger leaf aggregations at the expense of tree quality.

6.5. Discussion

Although the vegetation we create is synthetic and thus differs from the real vegetation, it preserves its basic appearance as it inherits the color captured by the orthophoto. This ensures color variety and consistency. Our leaf-based approach offers multiple advantages: the memory footprint is small (1 byte/texel for the vegetation type and distance field) and each resulting tree is unique. This variety on the vegetation is often missing in forest rendering approaches (Roettger, 2007; Bruneton and Neyret, 2012). Leaves can also be animated easily to simulate the effect of wind. We used the same set of vegetation parameters and leaf textures for all the test models shown in this paper. A more accurate reconstruction could be achieved by tuning these parameters according to the features of the local vegetation. In a network streaming scenario, our approach introduces a small overhead. In terms of per-texel data, we only need to transmit the vegetation type (2 bits), which represents a very small overhead.

7. Conclusions and Future Work

In this paper we have proposed a new inexpensive approach to model and render plausible landscape reconstructions. Our algorithm succeeds in generating a detail-rich representation of the terrain and vegetation, by refining the initial DTM to better reproduce the shape and appearance of the terrain surrounding the road, and creating plausible vegetation on top of it. In terms of acquisition cost and storage space, our approach fills the gap between publicly available DTM and orthophoto data, which lack realistic appearance for close views, and more sophisticated methods (e.g. LiDAR-based) that provide very accurate reconstructions but require expensive acquisition equipment and generate huge datasets not suitable for network streaming. An interesting avenue for future work is to explore level-of-detail techniques for the vegetation while preserving the variety and flexibility of our approach. Considering additional vegetation types would require higher order descriptors. We plan to use texture descriptors to detect more vegetation classes and to get better estimates of the vegetation height.

We plan to integrate color information from the road-level photographs to further adjust the vegetation type and parameters. Our current tree placement algorithm uses a simple dart throwing approach with a minimum distance constraint. We also plan to extend this approach by identifying and using vegetation samples from the acquired point clouds to place and get shape estimates of trees in regions captured by the road-level photographs. This would enhance the fitting of the reconstructed vegetation models to the real vegetation. Finally, adding ambient occlusion and HDR lighting does not seem to be a hard task.

Acknowledgments

Helpful discussions with Isabel Navazo and Alvar Vinacua are gratefully acknowledged. This work has been partially funded by the Spanish Ministry of Science and Innovation under grant TIN2010-20590-C01-01.

References

Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., Silva, C. T., 2001. Point set surfaces. In: Proceedings of Visualization ’01. IEEE VIS ’01. pp. 21–28.

Andújar, C., Chica, A., Vico, M. A., Moya, S., Brunet, P., 2014. Inexpensive reconstruction and rendering of realistic roadside landscapes. Computer Graphics Forum 33 (6), 101–117.

Asirvatham, A., Hoppe, H., 2005. Terrain rendering using GPU-based geometry clipmaps. Addison Wesley, Ch. 2, pp. 27–46.

Bailey, D. G., 2004. An efficient euclidean distance transform. In: International Workshop on Combinatorial Image Analysis, IWCIA 2004. pp. 394–408.

Balaguer, A., Ruiz, L., Hermosilla, T., Recio, J., 2010. Definition of a comprehensive set of texture semivariogram features and their evaluation for object-oriented image classification. Computers & Geosciences 36 (2), 231 – 240.

Behrendt, S., Colditz, C., Franzke, O., Kopf, J., Deussen, O., 2005. Realistic real-time rendering of landscapes using billboard clouds. In: Proceedings of Eurographics 2005. pp. 507–516.

Bruneton, E., Neyret, F., 2008. Real-time rendering and editing of vector-based terrains. Computer Graphics Forum 27 (2), 311–320.

Bruneton, E., Neyret, F., 2012. Real-time realistic rendering and lighting of forests. Computer Graphics Forum 31, 373–382.

Cignoni, P., Ganovelli, F., Gobbetti, E., Marton, F., Ponchio, F., Scopigno, R., oct. 2003. Planet-sized batched dynamic adaptive meshes (p-bdam). In: Proceedings of Visualization, 2003. pp. 147 –154.

Colditz, C., Coconu, L., Deussen, O., Hege, C., 2005. Realtime rendering of complex photorealistic landscapes using hybrid level-of-detail approaches. In: Trends in Real-Time Landscape Visualization and Participation. pp. 97–106.

Dachsbacher, C., 2006. Interactive terrain rendering - towards realism with procedural models and graphics hardware. Ph.D. thesis, Universität Erlangen-Nürnberg.

Dachsbacher, C., Bolch, T., Stamminger, M., 2006. Procedural reproduction of terrain textures with geographic data. In: Aach, T., Westermann, R. (Eds.), Proceedings of Vision Modeling and Visualization. pp. 105–112.

Dachsbacher, C., Stamminger, M., 2004. Rendering procedural terrain by geometry image warping. In: Proceedings of the Eurographics Symposium on Rendering 2004. pp. 103–110.

Dachsbacher, C., Stamminger, M., 2005. Shader X4 - Cached Procedural Textures for Terrain Rendering. Charles River Media, Ch. 7.3, pp. 457–466.

Decaudin, P., Neyret, F., June 2004. Rendering forest scenes in real-time. In: Rendering Techniques '04 (Eurographics Symposium on Rendering). pp. 93–102.

Deussen, O., Colditz, C., Stamminger, M., Drettakis, G., 2002. Interactive visualization of complex plant ecosystems. In: VIS '02: Proceedings of the conference on Visualization '02. pp. 219–226.

Deussen, O., Lintermann, B., 2005. Digital Design of Nature. Springer.

Drori, I., Cohen-Or, D., Yeshurun, H., 2003. Fragment-based image completion. ACM Transactions on Graphics 22, 303–312.

Duchaineau, M., Wolinsky, M., Sigeti, D., Miller, M., Aldrich, C., Mineev-Weinstein, M., oct. 1997. Roaming terrain: Real-time optimally adapting meshes. In: Proceedings of Visualization '97. pp. 81 –88.

Ebert, D. S., Musgrave, F. K., Peachey, D., Perlin, K., Worley, S., 2003. Texturing and modeling. A procedural approach. The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling. Morgan Kaufmann Publishers.

Freeman, W., Jones, T., Pasztor, E., mar/apr 2002. Example-based super-resolution. IEEE Computer Graphics and Applications 22 (2), 56 –65.

Früh, C., Zakhor, A., Nov. 2003. Constructing 3d city models by merging aerial and ground views. IEEE Computer Graphics and Applications 23 (6), 52–61.

Furukawa, Y., Ponce, J., 2010. Accurate, dense, and robust multiview stereopsis. IEEE Transactions on Pattern Analysis and Machine Intelligence 32, 1362–1376.

Gilet, G., Meyer, A., Neyret, F., 2005. Point-based rendering of trees. In: Poulin, P., Galin, E. (Eds.), Proceedings of Eurographics Workshop on Natural Phenomena. pp. 67–72.

Gobbetti, E., Marton, F., Cignoni, P., Di Benedetto, M., Ganovelli, F., 2006. C-bdam compressed batched dynamic adaptive meshes for terrain rendering. Computer Graphics Forum 25 (3), 333–342.

Gross, M., Pfister, H., 2007. Point-based graphics. Morgan Kaufmann Series in Computer Graphics. Elsevier.

Hays, J., Efros, A. A., 2007. Scene completion using millions of photographs. ACM Transactions on Graphics 26 (3), 4:1–4:7.

Hertzmann, A., Jacobs, C. E., Oliver, N., Curless, B., Salesin, D. H., 2001. Image analogies. In: Proceedings of SIGGRAPH 2001. ACM, pp. 327–340.

Kazhdan, M., Bolitho, M., Hoppe, H., 2006. Poisson surface reconstruction. In: EG Symposium on Geometry processing. SGP '06. Eurographics Association, pp. 61–70.

Kraus, K., 2007. Photogrammetry: Geometry from Images and Laser Scans. Walter de Gruyter.

Kriegler, F. J., Malila, W. A., Nalepka, R. F., Richardson, W., October 1969. Preprocessing transformations and their effects on multispectral recognition. In: Proceedings of the Symposium on Remote Sensing of Environment '69. pp. 97–131.

Livny, Y., Pirk, S., Cheng, Z., Yan, F., Deussen, O., Cohen-Or, D., Chen, B., 2011. Texture-lobes for tree modelling. In: Proceedings of SIGGRAPH 2011. SIGGRAPH '11. ACM, New York, NY, USA, pp. 53:1–53:10.

Losasso, F., Hoppe, H., Aug. 2004. Geometry clipmaps: terrain rendering using nested regular grids. ACM Transactions on Graphics 23 (3), 769–776.

Mahalanobis, P. C., 1936. On the generalised distance in statistics. In: Proceedings of National Institute of Science, India. No. 1 in 2. National Institute of Science, India, pp. 49–55.

Malinverno, A., 1990. A simple method to estimate the fractal dimension of a self-affine series. Geophysical Research Letters 17 (11), 1953–1956.

Mandelbrot, B. B., 1983. The fractal geometry of nature. Revised and enlarged edition. W. H. Freeman and Co.

Miller, G., 1986. The definition and rendering of terrain maps. In: Proceedings of SIGGRAPH ’86. pp. 39–48.

Neubert, B., Pirk, S., Deussen, O., Dachsbacher, C., 2011. Improved model- and view-dependent pruning of large botanical scenes. Computer Graphics Forum 30 (6), 1708–1718.

Perlin, K., 1985. An image synthesizer. In: Proceedings of the 12th annual conference on Computer graphics and interactive techniques. Proceedings of SIGGRAPH ’85. ACM, ACM, New York, NY, USA, pp. 287–296.

Roettger, S., 2007. Ndvi-based vegetation rendering. In: Proceedings of the Ninth IASTED International Conference on Computer Graphics and Imaging. ACTA Press, pp. 41–45.

Ruiz, L., Fdez-Sarría, A., Recio, J., 2004. Texture feature extraction for classification of remote sensing data using wavelet decomposition: a comparative study. In: 20th ISPRS Congress. pp. 1109–1115.

Runions, A., Lane, B., Prusinkiewicz, P., 2007. Modeling trees with a space colonization algorithm. In: Proceedings of Eurographics Workshop on Natural Phenomena 2007. pp. 63–70.

Saupe, D., 1989. Point evaluation of multi-variable random fractals. In: Visualisierung in Mathematik und Naturissenschaft-Bremer Computergraphik Tage. Heidelberg. Springer-Verlag, pp. 114–126.

Schepers, H., van Beek, J., Bassingthwaighte, J., jun 1992. Four methods to estimate the fractal dimension from self-affine signals. Engineering in Medicine and Biology Magazine 11 (2), 57–64, 71.

Figure 1: Given commonly available data (low-resolution DTM, orthophoto, road vector data) and a set of photographs taken from a road, we model the surrounding landscape using a data-driven semi-procedural representation that provides plausible terrain and vegetation details.

Schneider, J., Boldte, T., Westermann, R., 2006. Real-time editing, synthesis, and rendering of infinite landscapes on GPUs. In: Vision, Modeling and Visualization 2006. pp. 145–152.

Seitz, S., Curless, B., Diebel, J., Scharstein, D., Szeliski, R., june 2006. A comparison and evaluation of multi-view stereo reconstruction algorithms. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 519 – 528.

Snavely, N., Seitz, S. M., Szeliski, R., Jul. 2006. Photo tourism: exploring photo collections in 3d. ACM Transactions on Graphics 25 (3), 835–846.

Wang, L., You, S., Neumann, U., 2006. Large-scale urban modeling by combining ground level panoramic and aerial imagery. In: Proceedings of 3DPVT 2006. IEEE Computer Society, pp. 806–813.

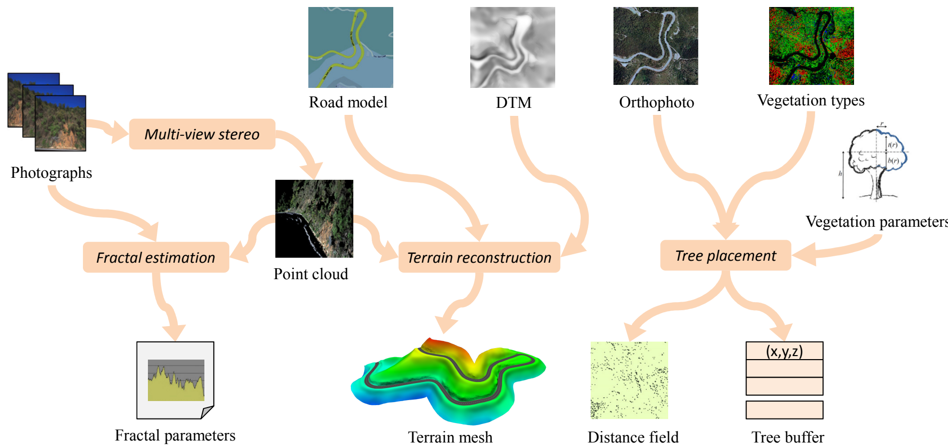


Figure 2: Overview of our reconstruction algorithm.

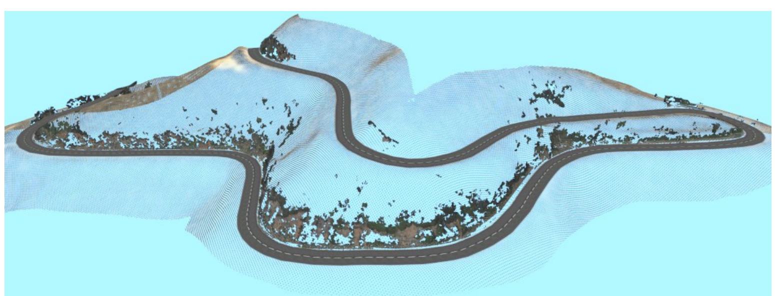


Figure 3: Point cloud (shown as point splats) reconstructed from the road-level photographs of the *Arrabassada* dataset. The visibility from the road is limited and thus the photographs only capture a part of the environment (shown as small dots).

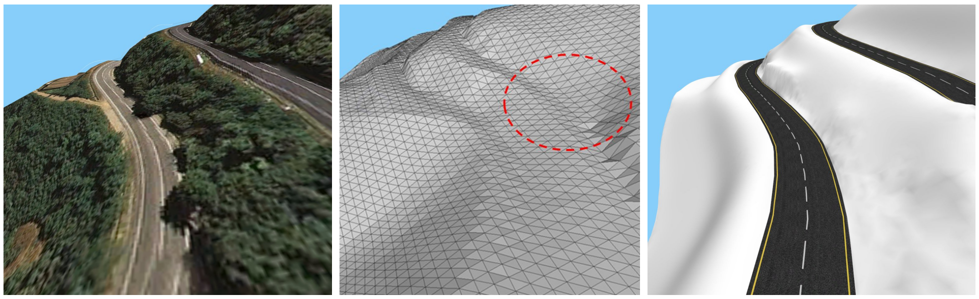


Figure 4: Traditional DTMs (left and middle) do not capture well the terrain slope next to the road. Roads are not flat and their slope is affected by that of the neighboring terrain. We address this problem (right) by integrating the road model and the acquired point clouds into the user-provided DTM.

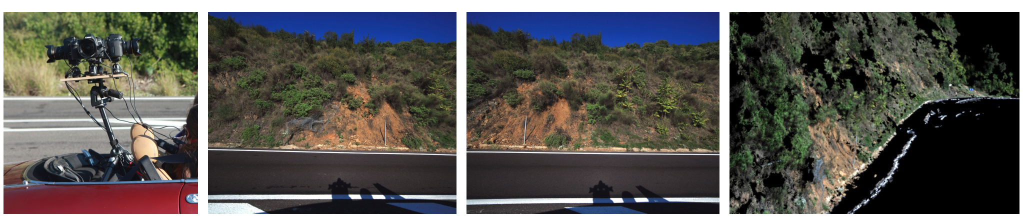
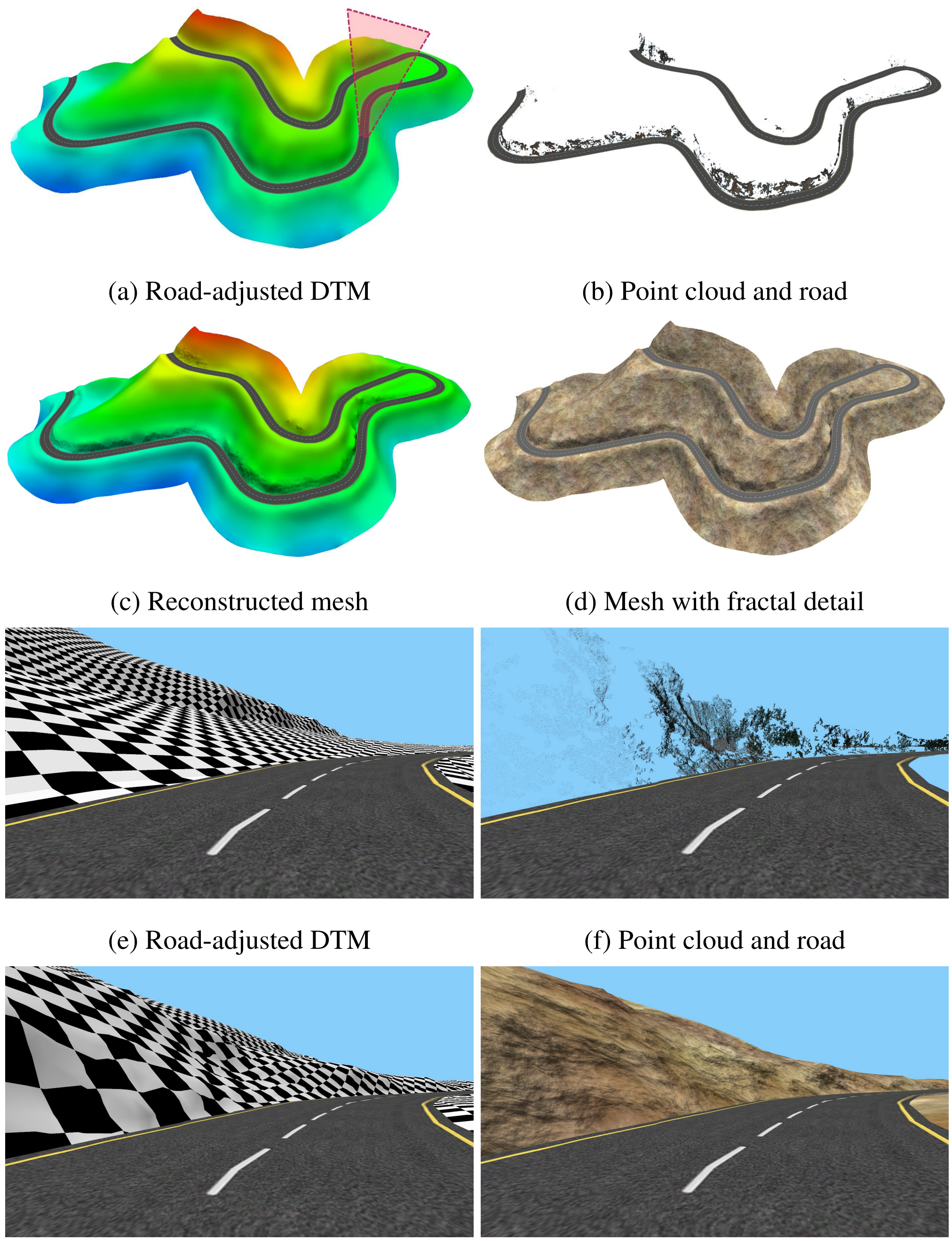


Figure 5: Vehicle-mounted cameras used to acquire the photographs from the road, two different photographs of the same area, and the point cloud as reconstructed from 12 successive photographs.



(a) Road-adjusted DTM
(b) Point cloud and road
(c) Reconstructed mesh
(d) Mesh with fractal detail
(e) Road-adjusted DTM
(f) Point cloud and road
(g) Reconstructed mesh
(h) Mesh with fractal detail

Figure 6: Terrain reconstruction example (*Arrabassada* model).

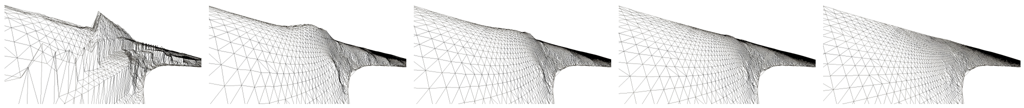
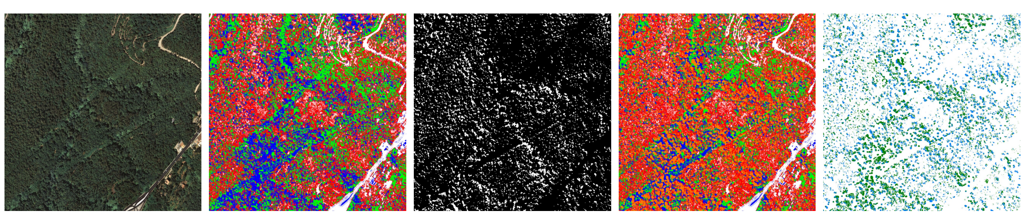


Figure 7: Smoothing of a terrain mesh extracted from a 10 cm voxelization with 0, 10, 50, 100 and 500 Laplacian iterations.

Figure 8: Leaf parameters, a sample tree created with our model, and two leaf textures.



(a) (b) (c) (d) (e)

Figure 9: Shadow-based adjustment: (a) Orthophoto, (b) Classification, (c) Detected shadows, (d) Adjusted classification, (e) Pixels affected by the adjustment.

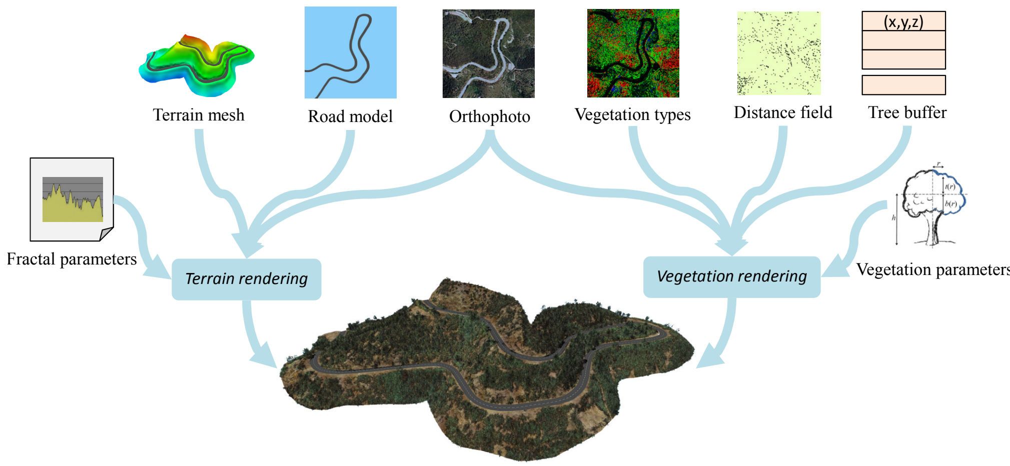
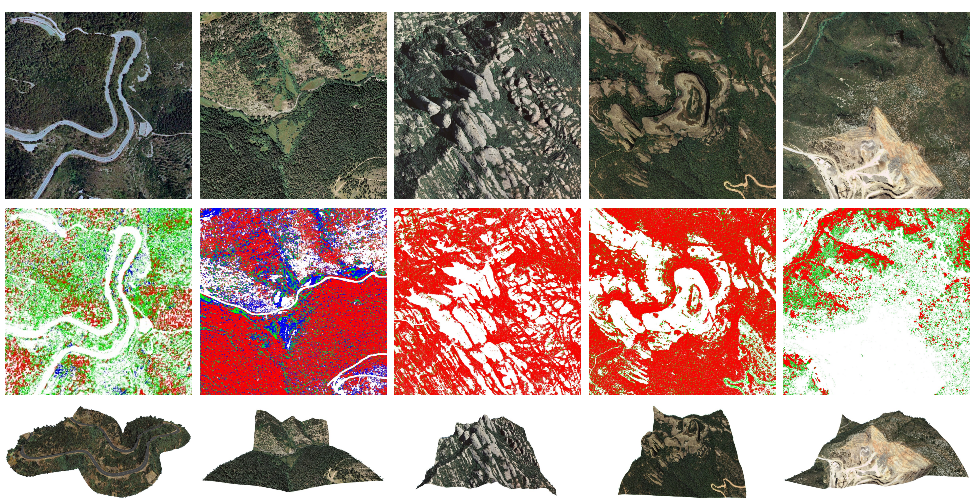


Figure 10: Overview of our rendering algorithm.



(a) Arrabassada (b) Andorra (c) Montserrat (d) Roca del Corb (e) Garraf

Figure 11: Test datasets: Input orthophotos (top row), classifications (middle row) and reconstructed models (bottom row). The classification types are trees (in red), shrubs (in green), grass (in blue) and terrain/road (in white).

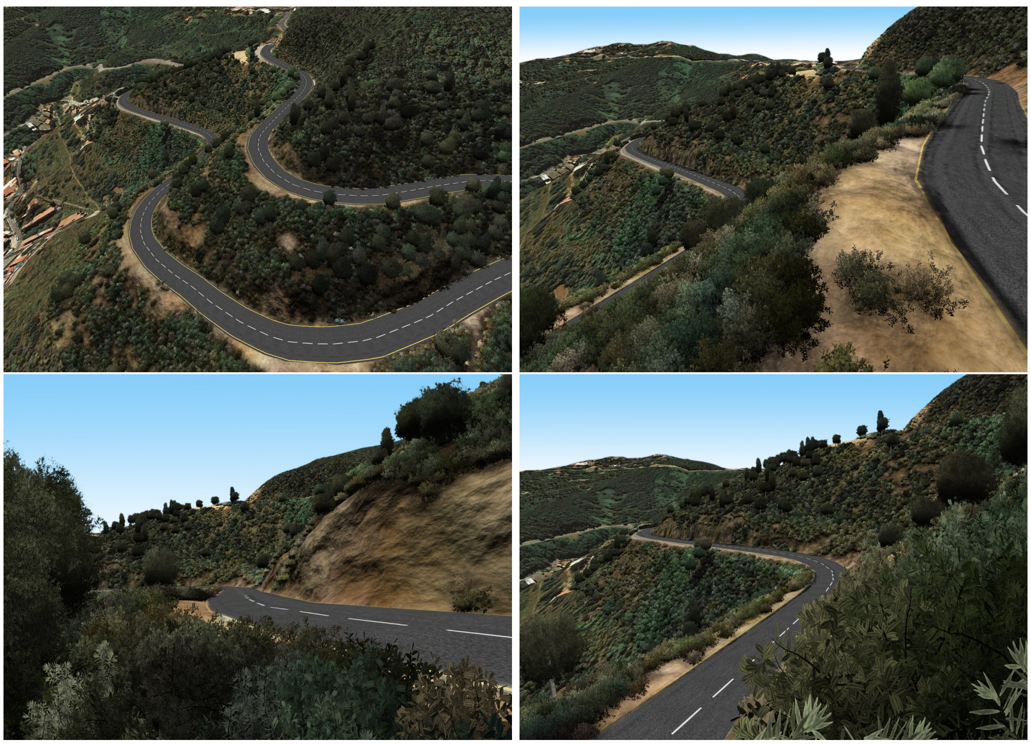


Figure 12: Renderings of the *Arrabassada* model.

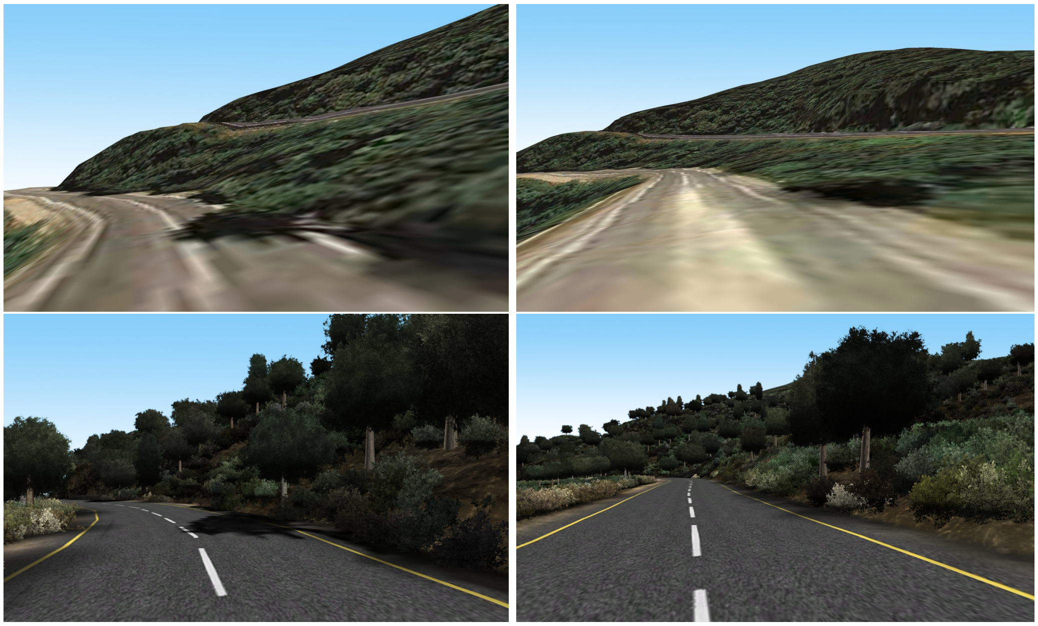
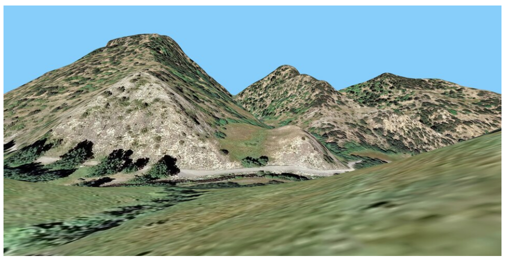


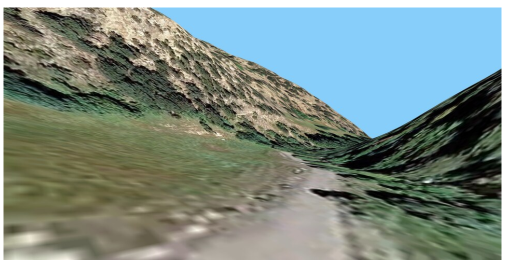
Figure 13: Initial DTM (top) and our reconstruction (bottom) for the *Arrabassada* dataset.

Figure 14: Our reconstructed model integrates seamlessly with surrounding DTM data.



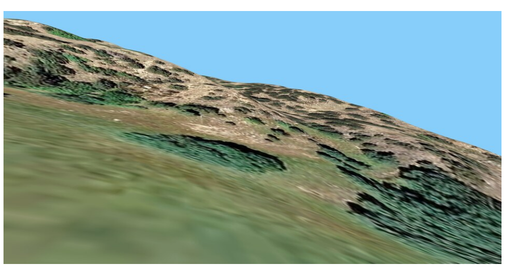
(a) DTM, view 1

(b) Reconstruction, view 1



(c) DTM, view 2

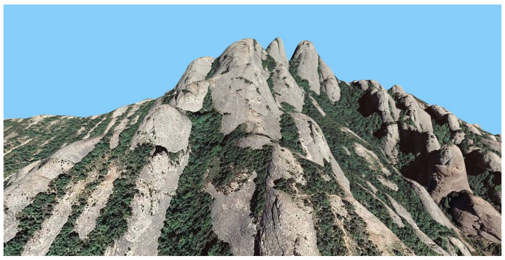
(d) Reconstruction, view 2



(e) DTM, view 3

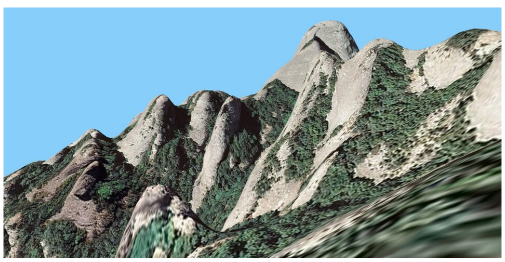
(f) Reconstruction, view 3

Figure 15: Results with the *Andorra* dataset.

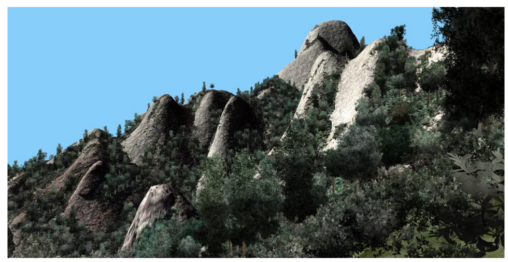


(a) DTM, view 1

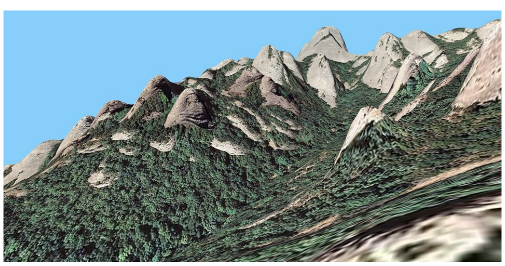
(b) Reconstruction, view 1



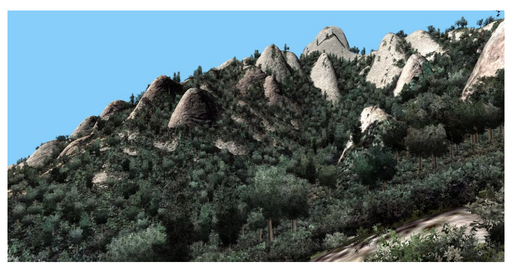
(c) DTM, view 2



(d) Reconstruction, view 2



(e) DTM, view 3

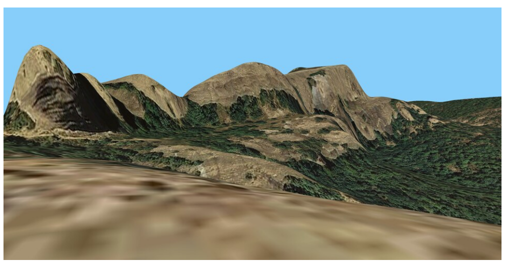


(f) Reconstruction, view 3

Figure 16: Results with the *Montserrat* dataset.

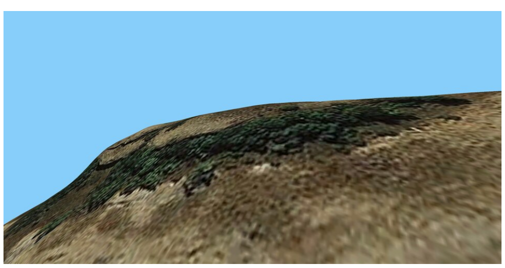
(a) DTM, view 1

(b) Reconstruction, view 1



(c) DTM, view 2

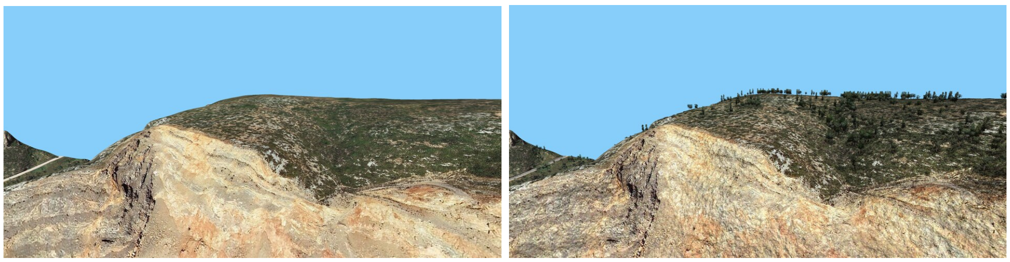
(d) Reconstruction, view 2



(e) DTM, view 3

(f) Reconstruction, view 3

Figure 17: Results with the *Roca del Corb* dataset.



(a) DTM, view 1 (b) Reconstruction, view 1

(c) DTM, view 2 (d) Reconstruction, view 2

(e) DTM, view 3 (f) Reconstruction, view 3

Figure 18: Results with the *Garraf* dataset.

Figure 19: Plausibility analysis. Two different lanscapes are shown: Arrabassada in the top row and Montserrat in the bottom row. The left columns show, in both cases, real photos of the landscapes, while images of our reconstructions are presented in the right columns.