Bridge detection in grid terrains and improved drainage enforcement

Ryan Carlson
Swarthmore College
Swarthmore, PA 19081
rcarlso1@swarthmore.edu

Andrew Danner
Swarthmore College
Swarthmore, PA 19081
adanner@cs.swarthmore.edu

ABSTRACT

Bare Earth gridded digital elevation models (DEMs) are often used to extract hydrologic features such as rivers and watersheds. DEMs must be conditioned by removing spurious *sinks* (or depressions) which impede water flow in the model, but are not true hydrologic barriers. This conditioning process is designed to enforce proper drainage and connect real hydrologic networks (rivers) that would otherwise be disconnected in the unconditioned DEM. Primary means of conditioning DEMs include filling sinks and cutting barriers.

The availability of high resolution DEMs derived from lidar introduces new forms of false hydrologic barriers, primarily bridges. While attempts are made to automatically remove trees, buildings and bridges from bare Earth terrains, in practice many bridges remain in the final "cleaned" DEM. We present a supervised machine learning approach for detecting bridges and other hydrologic barriers in DEMs. Furthermore, we locally apply a simple cutting algorithm to condition DEMs in areas tagged as barriers by the machine learning step. After cutting, we use a filling technique to remove any remaining spurious depressions. Experimental results indicate that our approach accurately identifies a variety of bridge and bridge-like features. Our final conditioned DEM both modifies fewer grid cells and modifies cells to a lesser extent than other traditional conditioning approaches. The result is more realistic hydrologic models on high resolution terrains.

Categories and Subject Descriptors: I.4.9 [Image Processing and Computer Vision]: Applications

General Terms: Algorithms

Keywords: Supervised Learning, Terrain Modeling

1 Introduction

Digital elevation models (DEMs) are a common base layer from which a wide variety of terrain analysis applications are derived. Examples include [26] flow modeling, watershed extraction, viewshed analysis, erosion modeling, flood plain mapping, and landslide risk assessment. Many of these applications begin by modeling the flow of water over the terrain. Flow modeling can be defined for both grid DEMs and triangulated irregular networks (TINs). We present an overview of the flow modeling process for grid DEMs which are the focus of this paper.

Flow modeling [10] consists of two primary phases: *flow routing* and *flow accumulation*. In the flow routing phase, we designate for each grid cell p one or more downslope neighbor cells to receive flow from p. Though many flow routing models exist, e.g., [5, 22, 11], the single flow direction (SFD) model is a common approach which directs all of p's flow toward the single steepest downslope neighbor of p. In the flow accumulation phase, a uniform amount of water is placed on each grid cell and the water from each cell is distributed to the downslope neighbors according to the flow routing mechanism. The flow accumulation (sometimes called the *upslope contributing area*) of a cell p is the total contribution of water from all upslope neighbors connected via a flow routing path to p. Intuitively, cells along river channels have high flow accumulation values while cells along ridges have low flow accumulation values. River networks are frequently extracted from a DEM by selecting lines of high flow accumulation values. Watershed boundaries can also be extracted using flow accumulation information [24, 3]. River networks and watershed boundaries are then used for other GIS applications including erosion modeling and floodplain mapping.

1.1 Drainage Enforcement

Because flow modeling algorithms route flow downhill, there is no outgoing flow for local minima or *sinks* in the DEM. In practice, local minima are classified as *significant* sinks or *spurious/insignificant* sinks. Examples of significant sinks include quarries, open pit mines, sinkholes or large natural closed basins with no surface drainage outlet. Spurious sinks are noise in the sampling or digitization of the input data, artifacts from vegetation or building removal, or small natural features that flood easily. After raising the elevation of a few cells within the terrain, flow can be routed out of spurious sinks and towards the edge of the DEM or towards a significant sink. Because spurious sinks artificially disrupt flow modeling and result in inaccurate disconnected hydrologic networks, DEMs must be *hydrologically conditioned* before flow modeling analysis begins.

Hydrologic conditioning modifies the elevation of a DEM

ACM GIS '10, November 2–5, 2010, San Jose, California, USA.
Copyright 2010 ACM 978-1-4503-0428-3/10/11...$10.00.

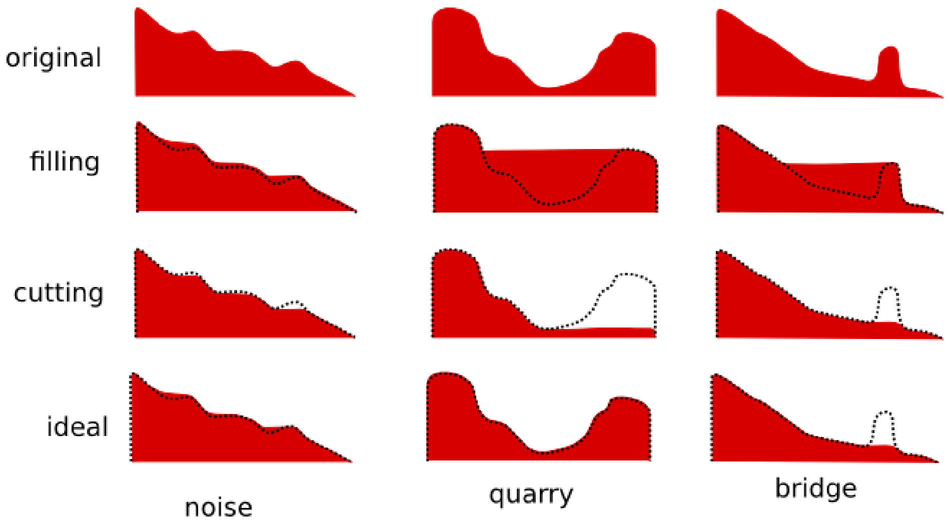


Figure 1: Examples of types of sinks and means of conditioning. For simplicity examples are shown only for one dimensional data.

to remove spurious sinks. The two primary means for conditioning DEMs are *filling* [10, 2] and *cutting* (also called *carving* or *breaching*) [13, 21]. An example of these conditioning methods is shown in Figure 1. Filling modifies the terrain by raising the elevations of grid cells within a spurious sink until there is a path of monotonically non-increasing height from the sink to a global outlet or a significant sink. Cutting modifies the terrain by lowering the elevation of a barrier on the boundary of the sink to create an outlet path. Combinations of filling and cutting can also be used [20]. Ideally, filling should be used in spurious sinks where the error or noise in the DEM artificially lowered elevations, while cutting should be used on artificially high cells. It is often difficult to automatically distinguish between these filling and cutting cases, or to even distinguish between significant and spurious sinks. In practice, GIS tools apply one hydrologic conditioning method (filling or cutting) to all sinks in the DEM regardless of type.

1.2 New Challenges

Modern remote sensing techniques such as lidar present new challenges for hydrologic conditioning. Grid DEMs are frequently derived from millions or billions of lidar points. These DEMs are many gigabytes in size and processing such large terrains frequently strains computational resources. Furthermore, at the high resolution of lidar DEMs (1–10 meters, or 3–32 ft, resolution per cell is common), many small features not captured at coarse resolutions are now visible. Significant sinks such as quarries, excavation sites, and shallow depressions in karst dominated topography are much more prevalent in high resolution terrains. Furthermore, while some attempt is made to automatically remove vegetation and man-made features, including bridges, from raw lidar data to create bare-earth models of the ground surface, some building footprints, low lying vegetation, roadways, and bridges remain in the DEM. Bridges and bridge-like features, such as berms with drainage culverts and check dams, are particularly problematic in lidar DEMs because they create spurious sinks precisely along locations of water flow. Removing the bridges from a DEM is a form of hydrologic conditioning which cuts through the bridge barrier responsible for creating the spurious upstream sink.

Preprocessing a DEM for surface water flow analysis would ideally classify each sink as one of three types: real sinks such as quarries or closed basins, small spurious sinks due to noise, and spurious sinks created by bridge-like barriers crossing a waterway in the DEM. Real sinks should persist in a hydrologically conditioned DEM. Small spurious sinks can be removed using a filling approach with little impact. Bridge-like barriers should be removed using a cutting approach. Since we are interested in processing large, high resolution DEMs, an ideal hydrologic conditioning approach should be scalable to large data sets.

1.3 Our approach

We present a method that moves towards an ideal hydrologic conditioning system by describing a supervised machine learning technique for classifying bridge-like features in grid terrains and implementing a cutting method for removing these features from the DEM. Our approach uses only algorithms scalable to large data sets and considers only small local windows around each grid cell. When compared to other hydrologic correction tools, our method modifies fewer grids cells and modifies cells to a lesser extent, while still qualitatively providing improved hydrological models.

2 Related Work

Numerous techniques for hydrologic conditioning and modeling water flow over grid DEMs have been proposed in GIS, computer science, and hydrology literature. Single flow models and filling techniques for conditioning [10] were proposed first, but additional flow models [22, 11] and cutting/hybrid conditioning models [13, 21, 20] have been well studied. While our primary focus is on grid DEMs, hydrological models for TINs have also been defined [6, 14].

Correcting DEMs using filling and computing flow accumulation values uses a number of basic graph algorithms that work well on small data sets, but are difficult to scale to data sets larger than main memory. TerraFlow [2] and

TerraStream [5] are two software packages that implement scalable algorithms for hydrologic conditioning and flow modeling on very large, multi-gigabyte terrains. TerraStream also has the advantage of working on both grids and TINs.

Most hydrologic conditioning algorithms apply conditioning to all sinks in the terrain, resulting in a conditioned DEM with no sinks, except for cells on the grid boundary. As noted in the introduction, some sinks in the terrain correspond to real features. Based on the concepts of topological persistence and the Morse-Smale complex [8, 7], TerraStream assigns a *persistence score* to each local minima such that smaller sinks which require less filling have low scores, while larger sinks corresponding to quarries and other real sinks have higher scores. The score is closely related to the height difference between the minimum height cell in the sink and the lowest height cell on the boundary of the sink. TerraStream can selectively condition only sinks with a score below a user provided threshold. While this selective conditioning could be very useful, sinks created by tall bridges blocking a river channel are assigned a high score in this topological persistence model. In Figure 1, the quarry and bridge in the original terrains have similar large scores, while the sinks in the original terrain with noise are small and have small scores. Since TerraStream only conditions using a filling approach, the inability to distinguish between bridges and quarries limits the effectiveness of selective conditioning. Removing bridges using cutting and then computing the scores of remaining sinks would allow the quarry to be preserved by filling with a low score threshold.

To effectively and efficiently classify bridges, we leverage the Adaptive Boosting machine learning algorithm proposed by Freund and Schapire [9]. AdaBoost adaptively combines a number of weak classifiers that may perform only slightly better than random guessing to boost the overall accuracy of a final strong classifier. We were motivated to select this machine learning approach due to its speed and its success in applications similar to ours such as real time face detection in digital images [25] and the identification of trees in aerial imagery [27]. We describe AdaBoost in more detail in Section 3.1.

Feature extraction from GIS data including lidar DEMs continues to be a well studied topic. Morphological approaches [19, 18, 4] are commonly used to extract features with well defined edges. Regarding the related problem of road extraction, a survey by Mena [15] summarizes a wide variety of techniques for detecting roads in GIS. However, methods for the detection of bridges and bridge-like features for the purpose of hydrologic conditioning seem lacking. In practice, data providers perform a simple intersection of road and river vector features to identify likely bridge candidates and then "burn" the river lines into the DEM by lowering the elevation of the DEM along these bridge intersections. This approach suffers from potential alignment issues particularly in high resolution terrains. Furthermore, since many small streams are not vectorized and several raised surfaces with drainage pipes are not along roads and classified as bridges, a simple vector intersection does not solve the problem.

The USGS Center for Lidar Information Coordination and Knowledge (CLICK) [23] has lidar DEMs of watershed scale areas in North Carolina, Ohio, and Pennsylvania that have numerous bridges and bridge-like features across small and large stream channels. These DEMs must undergo significant hydrologic correction before they are suitable for hydrologic analysis. The abundance of bridges in these data sets and the lack of tools designed explicitly for their removal motivates our efforts to improve hydrologic conditioning for improved drainage enforcement on high resolution grid DEMs.

3 Methods

We aim to improve the conditioning of DEMs using a two phase approach. First, we identify likely bridge locations given a small training set of hand selected bridge and non-bridge locations. We accomplish this task by using AdaBoost to construct a classifier which assigns a numerical score to each cell in a DEM such that cells with higher scores are more likely to be bridges. Once our classification phase is complete, we hydrologically condition the DEM by applying a morphological operator that cuts through only those cells identified as likely bridges in the previous phase. Below we describe these phases in more detail, starting with a quick overview of AdaBoost.

3.1 Bridge Classification Using Adaptive Boosting

We wish to train a classifier $h(x)$ which takes as input a set of *features* describing a single cell x in a terrain and outputs a score which indicates the likelihood of x having a particular property. We want to design a classifier which assigns high scores to likely bridges and low scores to non-bridge cells. AdaBoost [9, 25] is a machine learning algorithm that systematically combines sets of weak classifiers, each which may individually have a high error in classification, to form a strong classifier with low overall error.

The first step in training such a classifier is to hand-tag a small number of positive and negative training samples, $x_i, 1 \leq i \leq k$, each with a classification $y_i \in \{+1, -1\}$. For AdaBoost to be effective, these samples must be selected to adequately cover the search domain. Each training sample has an associated weight, w_i, and a feature vector describing the local characteristics around the selected point (see Section 3.2). The weights of all samples are initially equal.

AdaBoost proceeds in a series of T rounds, for some user specified value T. In each round t, the algorithm selects a weak classifier, $h_t(x)$ which best distinguishes the weighted positive and negative training samples. In each round, we use as our weak classifier a linear threshold θ_t on the feature f_t that best minimizes the classification error. More formally, we have

$$h_t(x) = \begin{cases} 1 & \text{if } p_t f_t(x) < p_t \theta_t \\ -1 & \text{otherwise} \end{cases} \tag{1}$$

where $p_t \in \{+, -\}$ is used to change the direction of inequality, if needed.

We compute the error ϵ_t of a classifier by summing the weights of the incorrectly classified training samples. Based on the error of the best weak classifier, we calculate a classifier weight α_t of $h_t(x)$ in round t:

$$\alpha_t = \frac{1}{2} \ln \frac{1 - \epsilon_t}{\epsilon_t} \tag{2}$$

We then use classifier weight to recompute the sample weights w_i of each training sample x_i, which reflects how well the algorithm has categorized the training sample so far. The weight of the sample in round $t+1$, $w_i(t+1)$ is given by:

$$w_i(t+1) = \frac{w_i(t)e^{-\alpha_t y_i h_t(x_i)}}{Z_t}, \tag{3}$$

where Z_t is a normalization constant such that the sum of all weights in each round is unity.

After selecting a weak classifier and reweighting training samples for T rounds, the weak classifiers are combined to produce a strong classifier, $H(x)$. After training a strong classifier, we use it to evaluate new terrain data. For each cell x that we wish to evaluate, we extract only the features used in the strong classifier. The strong classifier then evaluates the cell at x, assigning it a confidence score:

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \tag{4}$$

Finally, we normalize this score into the range (0,1) using the following sigmoid function:

$$H_{norm}(x) = \frac{1}{1+e^{-H(x)}} \tag{5}$$

We can view the normalized score as a probability, with higher confidence that we have found a bridge as the score approaches one.

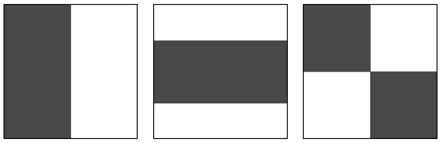
3.2 Feature Extraction

In addition to choosing representative training samples, AdaBoost relies on the features we extract from the elevation data to achieve robust categorization. Better context provided by the features translates to better categorization. All features described below require a small window inside of which local characteristics of a given map are taken into account.

We have empirically determined that the following five maps enable useful feature extraction. The first is the original DEM itself, followed by four derived maps. Our second map results from applying a 3×3 Sobel operator, typically used in edge detection, to each cell in the elevation data. This produces a map that differentiates areas of high and low gradient.

We calculate discrete Tangential and Profile curvatures from the DEM using a 3×3 window for our third and fourth maps. Tangential curvature is measured along the direction of contour lines for a given cell. Profile curvature is measured along the direction of the gradient. Tangential curvature helps identify the typical smoothness of bridge edges while profile curvature identifies the steep slopes in the transition from the top of the bridge to the ground below.

For our final map, we use TerraStream [5] to efficiently compute a Fill map. To create this map, we begin by hydrologically conditioning the entire DEM using filling. We take this corrected map and subtract from each cell the original elevation value. The result is a Fill map that reflects only areas of extensive filling. Since uncut bridges “block” large volumes of water, we see very large Fill values locally upstream of bridges (See Figure 4a.)



(a) Vertical feature. (b) Horizontal line feature. (c) Checkered feature.

Figure 2: Some Haar-like features. To calculate the scalar result, we sum values of a given map corresponding to each zone and then subtract white from gray areas.

During the training process, the algorithm visits each training sample in turn and extracts a large set of features based on local variations in each map. Extracting the features uses a window around the point of interest to provide local context. We calculate the minimum, maximum, average, and variance within these windows. Additionally, we make extensive use of Haar-like features [12], shown in Figure 2. These features are simple and efficient to compute as well as scalable to a variety of window sizes. They focus on patterns that arise locally in the derived maps. For each of the Haar-like features, we also use a top hat filter [17] over the feature, which serves to smooth some of the variance in the DEM and derived maps. With these maps and features, we can effectively distinguish bridges from other terrain features.

3.3 Cleaning Classified Map

Applying the strong classifier to a testing region results in a map in which each cell is annotated with a bridge confidence score. This resulting map is somewhat noisy (an example is shown in Figure 5a in Section 5), so we use a morphological open operator to clean the map and reduce noise. The open operator consists of an erosion operator followed by a dilation operator. The erosion and dilation operators set the center point in a small window of cells to the minimum or maximum value in the window, respectively. When we apply the erosion operator the small noisy classified regions are eliminated while larger, more robust classified regions are reduced. The following dilation enhances the robust regions back to their original size while reducing or eliminating noise. The noise reduction serves to remove spurious spikes of high classification scores while keeping the true bridge classifications intact. For high resolution DEMs, bridges extend over several cells. We enhance high confidence areas using further dilations to ensure high confidence scores extend over the entire width of the bridge instead of highlighting only an edge or center of a bridge.

3.4 Hydrologic Conditioning

Once we have successfully identified bridges in a region, we adjust the local terrain to improve water flow analysis accuracy. Over likely bridge locations (cells with high classified scores), we apply an erosion operator to the original elevation map. This operator is applied in a small local window so that the elevation around bridge regions is lowered to the lowest elevation in a local window, cutting through the barriers that had been impeding water flow. Thus, after our algorithm has been trained to recognize bridges and we

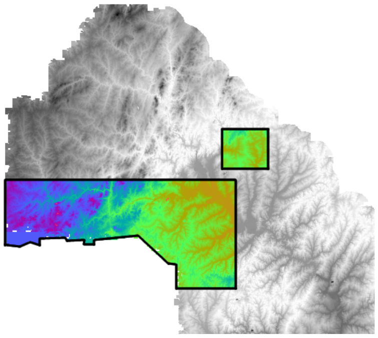


Figure 3: A section of the Neuse river basin. The training area (west) and testing area (east) are outlined.

have classified a region, we can act on the information to correct the elevation such that flow analysis results in correct drainage enforcement.

4 Experiments

To validate our approach, we implemented our algorithms in C++ and tested them on a large DEM covering the Neuse River watershed in North Carolina, USA. The parameters used for tagging training samples, extracting features, and conditioning terrains are described below. All experiments were run on a 2.8 GHz Quad-Core AMD Opteron processor with 8 GB of memory.

4.1 Training

Our initial tests used a grid DEM sampled at 40 foot resolution[1]. The training area contains 5.8 million cells. A portion of the terrain is show in Figure 3 along with outlined training and testing regions. Within this area, we hand-tagged 200 positive and 400 negative training examples. To accurately represent our search space, we needed negative samples from rivers, valleys, plateaus, significant hydrologic sinks, and other non-bridge locations. Since there are numerous types of non bridge features, we tagged more negative examples than positive.

At each sampled location, we extracted features from Elevation, Sobel, Profile, Tangential, and Fill maps. We use the minimum, maximum, average, and variance features along with 15 Haar-like features and 15 top-hat features. Each of the minimum, maximum, average, and variance was extracted at three window sizes ranging from 3×3 to 9×9. The other 30 features vary more widely, but cover sizes from 1×3 to 29×25. Since each of these 34 features is extracted over three different window sizes we have 102 features per map. In total, 510 features describe each training sample, 102 from each of five maps.

We also experimented with the training and testing area sampled at 20 foot resolution. Here the training area is four times denser, containing 23.4 million grid cells. Since our feature extraction windows operate on cells, we increased the window sizes of all features fourfold to cover the same total area as the trials run at 40 foot resolution. Unless otherwise specified, experiment details are reported on the 40 foot resolution. Results for 20ft data were very similar.

Once we extract the feature vector from each training sample, we use it as input to the AdaBoost algorithm. AdaBoost runs for a specified number of rounds, selecting the one weak classifier in each round that best classifies the weighted positive and negative samples (see Section 3.1). Using cross-validation (see Section 5.3) we empirically determined that our strong classifier was most effective using 20 weak classifiers (20 rounds) at 40 foot resolution and 40 classifiers at 20 foot resolution. The feature extraction of training data and strong classifier construction takes less than five seconds.

4.2 Improved Drainage Enforcement

We tested the strong classifier on a 440,000 cell region northeast of the training region. For each cell, only the features required by the strong classifier were extracted from each map as described above. After extraction, each cell is assigned a confidence score by the strong classifier (see Section 3.1). The extraction process takes less than five minutes, while the classification assignment takes about one minute including reading the large input files from disk.

After identifying the bridges – cells with a classification score above 0.5 – in the testing region, we use the erosion operator to remove them as described in Section 3.4. We empirically determined that, at 40 foot resolution, five consecutive erosion operators flattens enough terrain to effectively remove hydrologic barriers and allow for accurate flow modeling without modifying more than a small local area around the bridge. Each erosion operator is applied using a 9×9 window centered on the cell being eroded. These parameters are adjustable based on terrain resolution. Modification of bridge cells in the testing region takes less than two seconds. After modifying the terrain using erosion operators, some spurious sinks may still remain. We remove these sinks by running TerraStream on the eroded terrain. TerraStream uses a filling approach to remove remaining sinks.

One way to judge the effectiveness of our modifications is to calculate the flow accumulation and extract the river networks using TerraStream. By comparing flow routing using our approach against the traditional TerraStream approach of filling all sinks with no bridge detection, the river network extraction process provides a qualitative measure of our ability to correctly alter the terrain. In Figure 8 we show the river networks with and without cutting.

5 Results

In this section we describe both quantitative and qualitative results of our experiments. We report on cross-validation results over the training area to quantitatively evaluate how well our system adapts to unseen data. We also detail the accuracy of our automatic classification of cells using a careful but slow hand classification of a small testing area. Regarding classification, we examine which types of DEM derived maps were selected most often for distinguishing bridges from non-bridges and describe the relative weighting of maps used for feature extraction. We finish with a summary of the improved hydrologic models we can construct with our approach and describe some minor limitations of our solution.

Unless otherwise stated, the results in this section are derived from 40 foot resolution data. Classification maps and

[1] This grid DEM was constructed from original lidar points using a regularized spline with tension [16, 1].

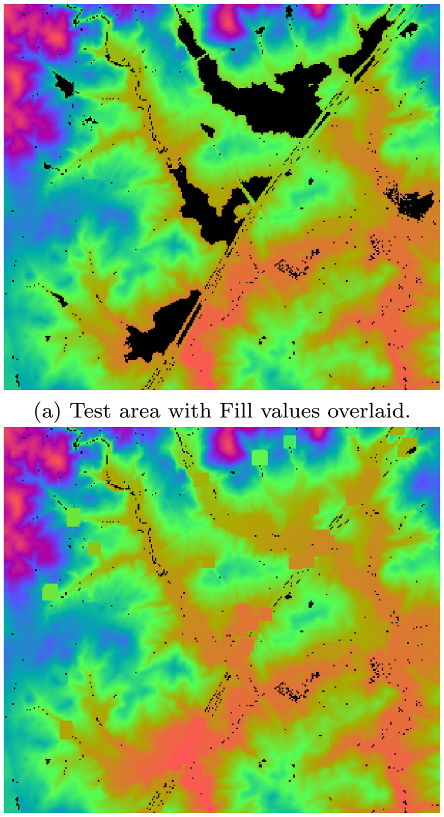
the resulting river networks look nearly identical on higher resolution data. We use the 20 foot resolution dataset to show how our algorithm adapts to changes in scale.

5.1 Overview

Before describing our results in detail, we start with a quick overview that highlights the capabilities of our method. The most immediate result of removing hydrologic barriers is the reduced need to remove sinks using filling. Recall that our initial Fill map is computed by removing all sinks using the filling approach implemented in TerraStream and then subtracting the original terrain. This Fill map is only used for classification purposes and is not used for our final hydrologic conditioning. To compute our final conditioned map, we first remove or reduce barriers classified as bridges using a morphological erosion operator. We then remove any remaining sinks using TerraStream's fill algorithm. We can compute a new difference map which is the subtraction of the original DEM from our final conditioned DEM. Figure 4a shows a section of testing area in which we compare fill values generated using only TerraStream. Only significant Fill values (above one foot or 30cm) are shown. Uncut bridges can be qualitatively identified by large Fill values upstream. Figure 4b shows that same area when we cut bridges before filling. Markedly less fill is necessary since we have correctly identified bridges and modified the terrain in a more appropriate way. The underlying terrain no longer needs to be artificially raised in as many places to allow water to flow over barriers. In this figure it is possible to see small blocks around the bridge locations that have had their original elevations lowered by the erosion operator to correctly route flow.

To correctly classify bridges in the DEM, we discussed the use of morphological operators in Section 3.3. If we look at the initial classification results in Figure 5a we see too many highlighted regions classified as bridges as a result of noise in the classification. To reduce the set of false positives we apply the 9×9 open operator to each cell. After a single open operation, we see a dramatic reduction in classified areas (Figure 5b). Since the erosion operator eliminates small spikes of noise, the subsequent dilation enhances only points of high confidence. With only one dilation, modifications to the underlying terrain to adjust for water flow have some effect, but additional dilations improve the classification map. In Figure 5c we see that most bridges are completely covered by high classification scores after three more dilations. Note that these images were taken from terrain sampled at 40 foot resolution, but classification of the higher resolution dataset looks identical. The highlighted regions cover enough of the area near bridges so that erosions of the underlying terrain allow for correct flow routing.

Over the larger testing area of 440,000 cells, of which the previous maps are a subset, we can quantitatively look at the classified maps. We consider a cell to be a bridge when the classification score is above 0.5 out of a possible 1. In the initial, noisy classification, 1.39% of cells in the map are classified as bridges. Applying an open operator reduced that number of bridge candidates fourfold to 0.35%. Subsequent dilations caused the classification cover to increase to 1.87% of the testing area. Since we only cut the terrain in areas classified as bridges, reducing noise and false positives reduces unnecessary modifications to the terrain.



(a) Test area with Fill values overlaid.

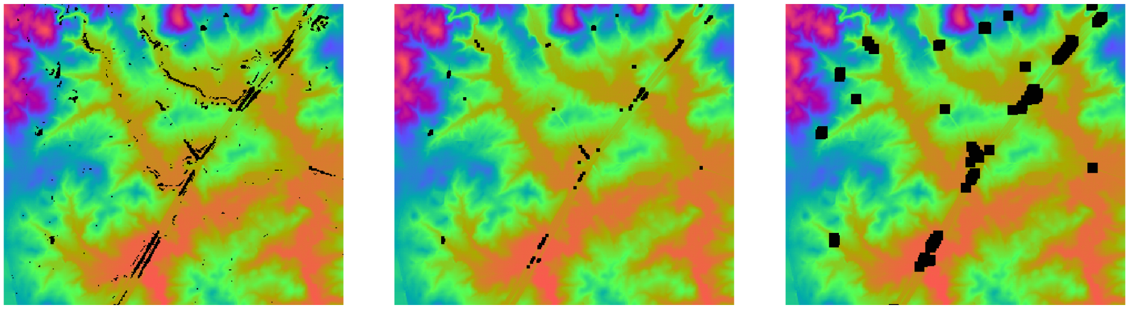
(b) Result after hydrologic conditioning and subsequent flooding. Fill values overlaid.

Figure 4: Test area before and after hydrologic conditioning.

Despite the growth in coverage, the distribution of classified data tightened noticeably. The standard deviation of classification scores for cells labeled as bridges was initially 0.159, but was reduced to 0.129 after the open operator. The value increased slightly to 0.132 after three more dilations. Furthermore, the average classification score of all cells in the testing area fell from 0.031 in the original map to 0.014 after the open operator and rose slightly to 0.018 after further dilations. Classification scores close to zero indicate a cell is likely not a bridge. The low average score for the entire terrain indicates once again that only a small percentage of cells are marked as potential bridges. The morphological operators allow us to correctly narrow down the search for hydrologic barriers. They enable us to focus on the most important bridge locations, isolating them for later modification.

5.2 Validating Testing Area

To quantify how effectively we detected bridges, we manually inspected the entire testing area to check the classification against the ground truth. We let any region with a confidence score above 0.5 qualify as a positive classification. In all, we identified 40 bridges by hand in the testing region. Our algorithm correctly detected 31 of these bridges, missing the other 9. There were no false positives. Thus the recall was 0.78 and the precision was 1. It is encouraging that all the correct classifications were major bridges in the area while all false negatives were much more minor barriers.



(a) A noisy initial classification.

(b) After a single morphological erosion and dilation.

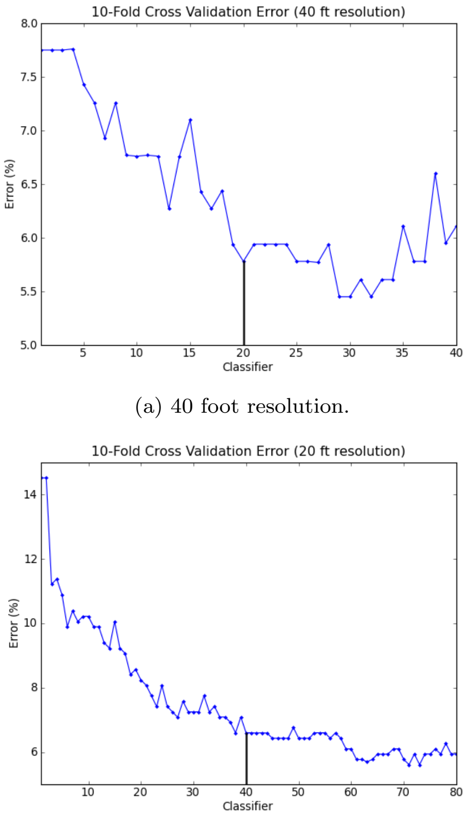
(c) After a single morphological erosion and four dilations.

Figure 5: Test area as we refine the classification using morphological operators shown at 40ft resolution. 20ft resolution results look identical.

5.3 Cross-Validation

Given our set of 600 hand tagged training samples of bridges and non-bridges, we applied cross-validation to evaluate the performance of our AdaBoost implementation. We used 10-fold cross-validation in which we partitioned our training samples into ten disjoint sets and repeatedly trained AdaBoost on nine of these sets and tested the resulting classifier on the excluded set of training data. By varying the number of weak classifiers used to build our final strong classifier and measuring the cross-validation error, we empirically determined that, at 40 foot resolution, 20 weak classifiers provides the best classifying power without overfitting. As we see in Figure 6a, error generally decreases until about 20 classifiers before leveling off. At the twentieth classifier, the error is 5.78 percent. After this threshold (marked by the vertical line) the slope indicates that the weak classifiers are beginning to overfit the search space. Precision and recall stay generally constant over the cross-validation process, hovering around 90 percent.

We also cross-validated the training region on 20 foot resolution data. In Figure 6b we show the cross-validation error, which decreases steadily until about 40 classifiers. We see the slope diminish and plateau as the classifiers overfit. Because of the higher resolution dataset, the variance of data values within a given window is typically higher. This increased variance makes the distribution of terrain features slightly noisier and makes classification a bit harder. Furthermore, our cross-validation evaluates classification scores only at an individual cell and compares them against our hand-tagged samples. Because it is more difficult to represent a bridge by a single grid cell at high resolution, false positives and false negatives are artificially higher than a comparison with the ground truth would suggest (see Section 5.1). Still, we are able to reduce error to 6.60 percent after 40 classifiers. Precision and recall vary slightly more at this higher resolution, finishing at 92 and 88 percent, respectively.



(a) 40 foot resolution.

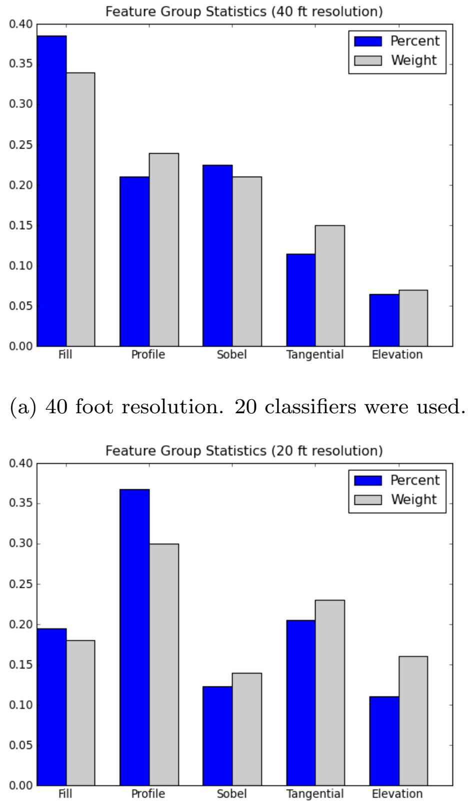
(b) 20 foot resolution.

Figure 6: 10-fold cross-validation error as weak classifiers are added to strong classifier. The vertical line highlights the cutoff classifier.

5.4 Feature Selection

To train the detection algorithm, each training sample was associated with 510 possible features extracted from the DEM or maps derived from the original DEM. Each weak classifier selected by AdaBoost in a particular round represents the best, most discriminating of those features in that round. It is interesting to look at which features were chosen and with what regularity. In Figure 7 we show the normalized weights and the percent of time each type of the five maps was chosen. Figures 7a and 7b show the distribution at 40 foot and 20 foot, respectively.

We see that the distributions of weights and percents change as we test on 20 foot and 40 foot resolution. On the 40 foot sample, the Fill map dominates the other feature groups, being chosen nearly forty percent of the time with an average weight of 0.34, clearly making it the most



(a) 40 foot resolution. 20 classifiers were used.

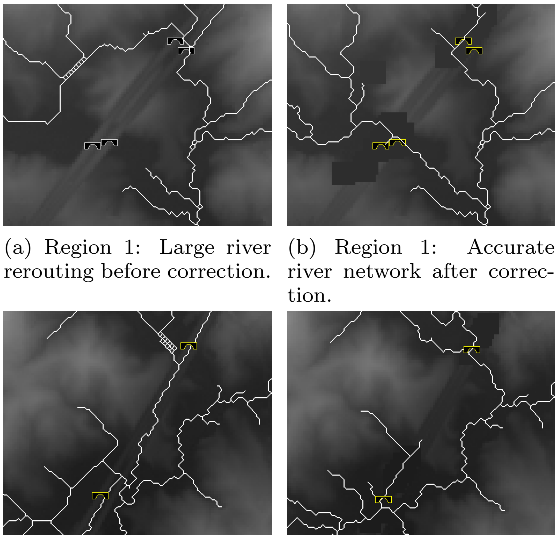
(b) 20 foot resolution. 40 classifiers were used.

Figure 7: Weights and percentages of feature groups used by AdaBoost. Weights and percentages are normalized to the range [0,1].

useful map at that scale. On the 20 foot sample, the Fill map is chosen almost twenty percent of the time with an average weight of 0.18. In higher resolution DEMs, bridges span more cells, and high Fill values are likely to be further from the center of the bridge. This property and the higher variance of features in high-resolution data somewhat de-emphasizes the ability of Fill data to distinguish positive from negative samples. However, the higher resolution data allows us to better capture sharp changes in the terrain and better model parameters such as gradient and profile/tangential curvature. The ability to adaptively change the weights as the scale changes is one benefit of the machine learning approach.

If we eliminate features derived from the Fill map for training and testing at 40 foot resolution, the cross-validation error rises to 7.42 percent, or about two percent higher than the error computed using the map. Since Fill is difficult to calculate, the ability of TerraStream to efficiently compute Fill values for large maps is essential.

In both the 40 foot and 20 foot resolution maps, some popular specific features emerged. The maximum profile curvature was prominent, being selected in 5.50 percent of the weak classifiers during our cross-validation experiments on 40 foot resolution and 3.75 percent on the 20 foot resolution. Since the Profile map has high values along bridges, it is encouraging to see this feature being selected often.



(a) Region 1: Large river rerouting before correction.

(b) Region 1: Accurate river network after correction.

(c) Region 2: Poor routing around bridges before correction.

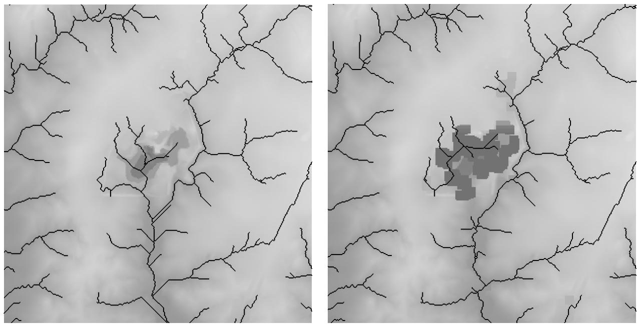
(d) Region 2: Improved network after correction.

Figure 8: River networks extracted from sections of the testing area before and after hydrologic correction. Bridge icons are overlaid.

Another popular feature was the horizontal line Haar-like feature from the Fill map, shown in Figure 2b. If a bridge is centered along the gray area, large fill values will appear upstream, at the edge of the feature, but will be close to zero near the center. This feature was chosen 5 percent of the time on 40 foot resolution and 1.42 percent on the 20 foot resolution.

5.5 Extracting River Networks

After conditioning the DEM by first removing bridges using cutting, we fill any remaining sinks using TerraStream's fill algorithm. We then use TerraStream to calculate the flow accumulation and extract the river networks on the conditioned DEM. Figure 8 shows the river networks of two regions without and with bridge cutting. With no bridge modifications, the routing of flow is inaccurate due to hydrological barriers. In Figure 8a the flow path should cross under the bridge. But, when only filling is used, the flow lines instead reroute in a large arc around the bridges. Figure 8c shows the river network routed parallel to a bridge until a suitable low point of the bridge is found. In both cases, the networks derived from terrain in which no bridge removal is applied are inaccurate. With cutting of bridges, we see much more accurate river networks. Figure 8b shows the water flowing though the bridge instead of around it. A large and incorrect network line has been eliminated and replaced with a more direct and accurate one. In Figure 8d, the network no longer runs along the road. It is again less artificial and more accurate. After identifying bridges and altering the terrain appropriately, the river networks much more accurately reflect the ground truth. We can derive better models using the new modified terrain.



(a) Uncorrected quarry. (b) Corrected quarry.

Figure 9: We incorrectly detect a quarry as a hydrologic barrier and thus condition inappropriately. Even so, river network lines still better reflect ground truth after conditioning.

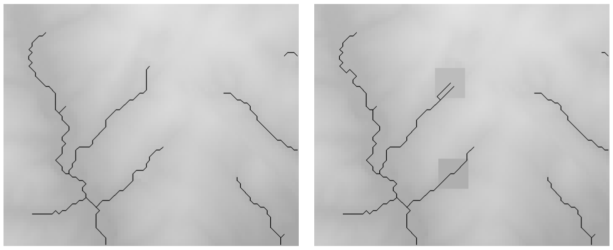
5.6 Impact of Incorrect Classifications

When modifying the underlying terrain based on the classification map, we need to be concerned about incorrectly classifying an area. False positives pose a potential problem because our program alters terrain that should be left in its original state. As we have seen, the initial classification map is cleaned using morphological operators to enhance strong classifications and reduce weaker noise. Still, some non-bridge features remain in the final classification map. The changes that result from these false positives are generally small and are corrected when we condition the terrain using TerraStream to fill remaining sinks.

False negatives may also be problematic, as we miss real bridges that block water flow. The cleaning process occasionally erodes small but correct classified regions in the original map, removing them entirely from the final classification. In these cases, we have empirically determined that the barriers we miss are small and can be filled at a low threshold without significant impact to the river networks.

In Figure 9 we show the river networks around a quarry outside of the testing region. With cutting of bridges (Figure 9a), the river networks incorrectly flow through the quarry. The significance score assigned by TerraStream to the quarry is high, but the significance scores of sinks formed by tall downstream bridges are also high. To remove the sinks created by bridges when using only filling to modify the terrain, we must set the threshold for insignificant sinks to infinity, since there is no way to effectively distinguish between hydrologic barriers and quarries using just the threshold. With cutting applied before filling remaining insignificant sinks (Figure 9b), we can use a significance threshold of twenty feet before recomputing the river networks. Thus the quarry is not completely flooded and, despite our incorrect adjustments, river networks more correctly reflect ground truth. Note that the more reasonable threshold is made possible by our correction algorithm. Since most bridge-like barriers are removed, we are able to lower the threshold for insignificant sinks.

Figure 10 details the misclassification and subsequent conditioning of a stream without any bridges or hydrologic barriers. Figure 10a shows the original terrain with the river network overlaid. Two regions have high enough scores in the final classified map to qualify for hydrologic conditioning. The modified terrain is shown in Figure 10b. Comparing these two maps, we see that no barrier exists where the terrain was altered. It is clear, however, that the river network is largely unaffected by these changes. When we incorrectly classify terrain, the local changes that result are small enough so as not to disturb the river networks. As we have seen, though, correct identification of hydrologic barriers leads to markedly improved networks. Thus, the benefits of cutting through true bridges using morphological erosion operators far outweigh the minor artifacts that result from modifying false positives.



(a) No cutting applied. (b) Cutting applied.

Figure 10: An example of two non-bridge areas incorrectly classified. The river network lines are affected only slightly. In both the uncut and cut images, filling is applied as a final step to remove any spurious sinks.

6 Discussion

Overall, our trained strong classifier computed using AdaBoost performs well in identifying a wide variety of bridge-like features in grid DEMs given only a small training set. The areas of the terrain classified as bridges allow us to distinguish between sinks created as the result of barriers and sinks created as the result of sampling noise.

Our simple morphological operator cuts through bridge-like features and dramatically reduces the extent of DEM modification when compared to a conditioning approach that uses filling for all sinks. Conditioning using filling can be computationally intensive, particularly in the case of nested sinks, and requires sophisticated techniques [2, 5] to scale to large data sets. Our morphological approach is scalable because it only examines a small local window around a barrier to effectively modify the DEM and remove bridge-like features. Because the computation within each window is independent of other classified regions, classification of DEM cells and modification of the DEM are excellent candidates for parallelization to further improve performance.

Occasionally, our erosion operator can make DEM modifications beyond the extent of the actual barrier, or eroding a false positive as shown in Figure 10. This has the side effect of creating small but visible artifacts in the conditioned DEM. However, these artifacts do not significantly alter the hydrologic networks. In contrast, filling large sinks blocked by tall bridges often creates large "lake" artifacts in the conditioned DEM. The Fill map overlay in Figure 4a shows the extent of these lake artifacts. The "lakes" are large flat areas of constant elevation which erase the original terrain and require additional algorithms to route flow on flat surfaces. Most of these algorithms are based on methods that are geometric and computationally efficient. The resulting hydrologic networks in these flat areas look very unnatural and do not reflect the original DEM topography. Computed rivers in flat areas tend to turn at forty-five or ninety degree angles and have multiple parallel flow lines the travel in a constant direction for extended distances (see the left side of Figure 8a and Figure 8c for examples of these unnatural river lines).

While our morphological erosion operator better conditions sinks blocked by barriers when compared to filling, there is still potential for improving our conditioning technique. One possible fix is to identify the upstream and downstream sides of a barrier tagged by our classifier and cut only a thin path through the barrier that connects these two sides. Since the number of regions identified as bridge-like features is small compared to the entire DEM, and since each feature has a limited extent, we can develop more sophisticated and computationally intensive techniques to apply to these small areas. Still, we find that our current approach represents a significant improvement over current hydrologic conditioning techniques for modern high-resolution DEMs and are confident the conditioning can be further improved by making some slight modifications.

References

[1] P. K. Agarwal, L. Arge, and A. Danner. From point cloud to grid DEM: A scalable approach. In A. Riedl, W. Kainz, and G. Elmes, editors, *Progress in Spatial Data Handling. 12th International Symposium on Spatial Data Handling*, pages 771–788. Springer-Verlag, 2006.

[2] L. Arge, J. Chase, P. Halpin, L. Toma, D. Urban, J. S. Vitter, and R. Wickremesinghe. Flow computation on massive grid terrains. *GeoInformatica*, 7(4):283–313, 2003.

[3] L. Arge, A. Danner, H. Haverkort, and N. Zeh. I/O-efficient hierarchical watershed decomposition of grid terrain models. In A. Riedl, W. Kainz, and G. Elmes, editors, *Progress in Spatial Data Handling. 12th International Symposium on Spatial Data Handling*, pages 825–844. Springer-Verlag, 2006.

[4] S. P. Clode, F. Rottensteiner, and P. Kootsookos. Improving city model determination by using road detection from lidar data. In U. Stilla, F. Rottensteiner, and S. Hinz, editors, *CMRT05. IAPRS*, volume XXXVI, Part3/W24. 2005.

[5] A. Danner, T. Mølhave, K. Yi, P. K. Agarwal, L. Arge, and H. Mitasova. Terrastream: From elevation data to watershed hierarchies. In *Proc. ACM Symposium on Advances in Geographic Information Systems*, pages 212–219, 2007.

[6] M. de Berg, P. Bose, K. Dobrindt, M. van Kreveld, M. Overmars, M. de Groot, T. Roos, J. Snoeyink, and S. Yu. The complexity of rivers in triangulated terrains. In *Proceedings 8th Canadian Conference on Computational Geometry*, 1996.

[7] H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical Morse complexes for piecewise linear 2-manifolds. In *Proc. ACM Symposium on Computational Geometry*, pages 70–79, 2001.

[8] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. In *Proc. IEEE Symposium on Foundations of Computer Science*, pages 454–463, 2000.

[9] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, September 1995.

[10] S. Jenson and J. Domingue. Extracting topographic structure from digital elevation data for geographic information system analysis. *Photogrammetric Engineering and Remote Sensing*, 54(11):1593–1600, 1988.

[11] N. L. Lea. An aspect driven kinematic routing algorithm. In A. J. Parsons and A. D. Abrahams, editors, *Overland Flow: Hydraulics and Erosion Mechanics*. Chapman & Hall, New York, 1992.

[12] R. Lienhart and J. Maydt. An extended set of haar-like features for rapid object detection. *IEEE ICIP*, 1:900–903, September 2002.

[13] L. Martz and J. Garbrecht. An outlet breaching algorithm for the treatment of closed depressions in a raster DEM. *Computers and Geosciences*, 25(7):835–844, 1999.

[14] M. McAllister and J. Snoeyink. Extracting consistent watersheds from digital river and elevation data. In *In Proc. ASPRS/ACSM Annu. Conf*, 1999.

[15] J. B. Mena. State of the art on automatic road extraction for GIS update: a novel classification. *Pattern Recognition Letters*, 24(16):3037–3058, December 2003.

[16] H. Mitasova, L. Mitas, and R. S. Harmon. Simultaneous spline interpolation and topographic analysis for lidar elevation data: methods for open source gis. *IEEE Geoscience and Remote Sensing Letters*, 2(4):375–379, 2005.

[17] J. C. Russ. *The Image Processing Handbook*. CRC Press, fourth edition, 2002.

[18] G. Sithole and G. Vosselmen. Bridge detection in airborne laser scanner data. *ISPS journal of photogrammetry and remote sensing*, 61(1):33–46, 2006.

[19] P. Soille. *Morphological Image Analysis*. Springer-Verlag, Berlin, 2004.

[20] P. Soille. Optimal removal of spurious pits in grid digital elevation models. *Water Resources Research*, 40(12), 2004.

[21] P. Soille, J. Vogt, and R. Colombo. Carving and adaptive drainage enforcement of grid digital elevation models. *Water Resources Research*, 39(12):1366–1375, 2003.

[22] D. Tarboton. A new method for the determination of flow directions and contributing areas in grid digital elevation models. *Water Resources Research*, 33:309–319, 1997.

[23] USGS. Center for LIDAR Information Coordination and Knowledge, 22 June 2010. `http://lidar.cr.usgs.gov/`.

[24] K. L. Verdin and J. P. Verdin. A topological system for delineation and codification of the Earth's river basins. *Journal of Hydrology*, 218:1–12, 1999.

[25] P. Viola and M. Jones. Robust real-time object detection detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[26] J. P. Wilson and J. C. Gallant. *Terrain Analysis : Principles and Applications*. John Wiley & Sons, New York, NY, 2000.

[27] L. Yang, X. Wu, E. Praun, and X. Ma. Tree detection from aerial imagery. In *Proc. ACM Symposium on Advances in Geographic Information Systems*, pages 131–137, 2009.