Lateral Detection

Qing Zhong[1], Martin Schindler[2], Christoph Stamm[2]

[1]Swiss Federal Institute of Technology (ETH) Zurich
[2]Institute for Mobile and Distributed Systems,
University of Applied Sciences Northwestern Switzerland
Email: qing.zhong@acm.org, {martin.schindler,christoph.stamm}@fhnw.ch

Abstract

This paper presents an approach to detecting laterals which consists of three steps. First, pipe images are restored and enhanced by implementing image processing techniques. Second, gray-scale morphology, anisotropic diffusion filters and histogram thresholding are performed to segment candidate laterals. In the third phase, AdaBoost is used to classify candidate laterals and its performance is compared to Support Vector Machine and K-NN. Experimental results show that AdaBoost with twenty weak classifiers outperform other algorithms. Our approach achieve about 90% test accuracy and has been tested on pipelines of $10,000$ meters in length or about 6000 scanned images of real sewer pipes from various cities all over the world.

1 Introduction

Communal sewer networks should be examined periodically (approx. every 8 years) for smaller and larger damage, because on the one hand leaching of fresh water is a waste of resources and on the other hand a contamination of fresh water through foreign substances could be dangerous. Such regular inspections of sewer network are time consuming and hence expensive tasks. In order to better understand the necessary effort, we regard here for example the city Zurich. The sewerage watershed of Zurich (and a few neighboring municipalities) covers about 60 square kilometers (lakes and forest excluded) and accommodates approximately $400,000$ inhabitants. The overall length of the public and private sewers in this area is 950 and 3050 kilometers respectively. While the ratio of one kilometer of sewers per 100 citizens is quite high, the average in developed countries, e.g. in the United States, is about the half, one kilometer per 200 citizens.

For more than ten years, sewer network condition assessment has been done by using closed circuit television (CCTV) or more recently sewer scanner elevation technology (SSET). Cameras mounted on robots produce either video records or digital images of the pipe's condition. The video records are later evaluated by a human operator, and pipe defects are classified against documented criteria. Such an evaluation is very time consuming and fatiguing. Therefore, semi or completely automated sewer pipe condition assessment systems could reduce time consumption and focus human expertise to interesting parts of the pipe. Because such a condition assessment system is based on digital images (Figure 1) and videos, analog video records have to be sampled and preprocessed by domain specific hard- and software. Our major industrial partner is the Swiss company CDLab AG. CDLab develops WinCan, a world-wide leading software in sewer pipe assessment.

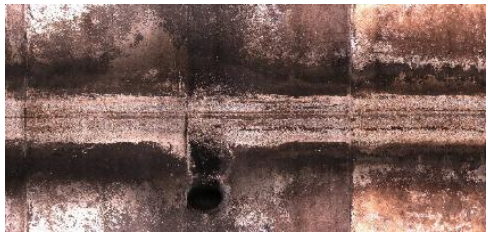


Figure 1: An original pipe image in color

A first step towards an automated sewer pipe condition assessment system is the reliable automated detection of joints and laterals, because leaks, cracks and fissures are often located near joints and laterals. While a semi-automated system leads the operator directly to potential defects and uses the expert's opinion, a completely auto-

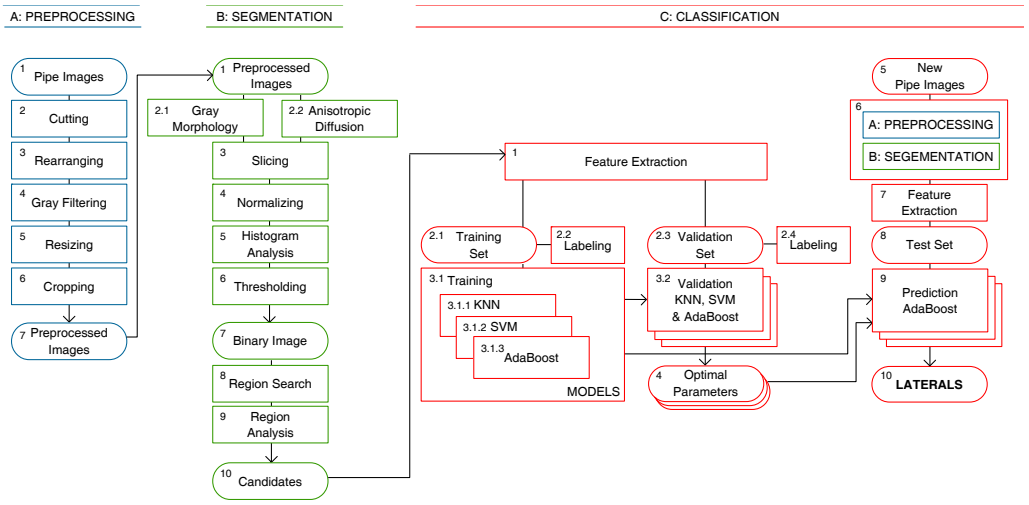


Figure 2: Schematic of our three-step lateral detection

mated system detects defects reliably. In our project (founded by the Swiss Confederation's innovation promotion agency; CTI project numbers: 7785.2 ESPP-ES, 8641.1 ESPP-ES), we focused on image preprocessing, joint and lateral detection, the very basics of both semi and completely automated condition assessment systems.

The research described in this paper is focused on automatic Lateral Detection (LD) in real imagery based on analog video records and digital images of very different quality. In Section 1.1, we present related work of LD. In Section 1.2 we briefly describe our approach to LD (Figure 2) and regard it as a twofold challenge in computer vision and machine learning. The first challenge is to distinguish the different objects, i.e., to delineate regions in pipe images corresponding to candidate laterals and segment them from these images. The second is to discriminate real laterals from false ones in the set of candidate laterals. We propose our three-step algorithm including preprocessing, segmentation and classification in Section 2, 3, and 4 respectively. In Section 5 we show the experimental results of classifying candidate laterals with AdaBoost and compare its performance with Support Vector Machine (SVM) and K-NN (where $K = 1$ and $K = 3$). The performance comparison is done in terms of tenfold cross validation which estimates the test errors, whereas an independent test set is used to assess the true test errors of different classifiers.

1.1 Related Work

Unlike joints detection, determination of whereabouts of laterals turn out to be a more difficult problem. Traditional segmentation methods such as Hough transformation [1] and linear thresholding do not fulfill our expectation. Though Hough transformation works well with lines, it fails to discriminate circular structures (e.g. laterals). Besides, linear thresholding makes the assumption that different objects "reside" in distinguish positions in a histogram, i.e., bimodal distributed, which is not often the case for pipe images with laterals. Therefore, Hough transformation and linear thresholding are only appropriate for joint detection [8].

Several papers [5, 9, 10] have proposed procedures that comprise methods like Otsu's thresholding [6], gray-scale morphology [17], linear discriminant functions, K-NN, and neural-fuzzy networks [14, 16, 18]. These methods have delivered sound results on their pipe image sets. Since these image sets are often not available, we have to use our own pipe image set instead. However, our experiment results show that those proposed methods work well with our image set only in the cases where the laterals are ideally shaped and colored (circular and dark), but fail in many others due to the fact that our pipe images span a wider range of characteristics, i.e., image intensity, lateral size, lateral shape, image noise, inhomogeneous background with non-

uniformed illuminations and a large variety of pipe material types. Moreover, gray-scale morphology introduces structuring element which often distorts one of the lateral's intrinsic properties, namely circular shape, whereas Otsu's thresholding assumes a bimodal distributed histogram and a uniformly illuminated image but neglects the fact that the pixel count of a lateral (spatial coherence) in a pipe image is bounded from above.

1.2 Our Approach

Our approach to LD is a three-step algorithm which consists of preprocessing, segmentation and classification. First, the original RGB pipe images I are cut, rearranged, filtered, resized and cropped. We name this first step as preprocessing and denote the resulted images as $\widetilde{I}$

$$\text{Preprocess} : I \rightarrow \widetilde{I}. \tag{1}$$

Second, we filter the images $\widetilde{I}$ once by applying gray-scale morphology and once anisotropic diffusion, and then perform thresholding to binarize the images. We use chain code algorithm [2] and region analysis to find positions of possible laterals, subject to constraints such as pixel count and object shape. We refer this second step as segmentation which outputs n positions

$$\text{Segment} : \widetilde{I} \rightarrow (i_k, j_k), k \in [1, n]. \tag{2}$$

Third, for each position (i_k, j_k), centered at which we create an image patch and name it as a candidate lateral $C_k(i_k, j_k)$. Given all candidate laterals, LD now becomes a binary classification problem, i.e., to classify candidate laterals (image patches) into two classes: presence or absence of laterals in these image patches. We thus extract p features from each candidate lateral and represent it as a feature vector $\mathbf{x}_k \in \mathcal{X}$ in a p-dimensional feature space $\mathcal{X} \subset \mathbb{R}^p$

$$\text{Extract} : C_k \rightarrow \mathbf{x}_k = [x_{k,1}, .., x_{k,p}]^T. \tag{3}$$

We then construct the data set $\mathcal{D}$ containing these n feature vectors. As for supervised learning, the data set $\mathcal{D}$ is manually labeled with $+1$ (with laterals) and -1 (without laterals) respectively and the data is assumed to be independent and identically distributed (i.i.d.)

$$\text{Label} : (\mathbf{x}_k, y_k), y_k \in \mathbb{G} = \{-1, +1\}. \tag{4}$$

Hence, the data set $\mathcal{D}$ can be represented as an $n \times (p+1)$ matrix whose k-th row is $[\mathbf{x}_k^T, y_k]$, where n denotes the number of candidate laterals, p represents the dimensionality of the feature space and the 1 designates the label.

To avoid overfitting and estimate the test (generalization) error, we use K-fold cross validation, where the data set $\mathcal{D}$ is randomly partitioned into K equally sized subsets, from which the training set $\mathcal{T}$ of size $t = |\mathcal{D}|(1 - 1/K)$ and the independent validation set $\mathcal{V}$ of size $|\mathcal{D}|/K$ are formed. In oder to assess the test error of the final chosen model, we use a test set $\mathcal{M}$ computed from a set of new independent pipe images. Consequently, by applying the best model $\hat{c}^*$ (classifier) selected by cross validation to the test set $\mathcal{M}$

$$\text{Classify } (\hat{c}^*) : \mathbf{x}_k \rightarrow y_k, \mathbf{x}_k \in \mathcal{M}, \tag{5}$$

the true test error

$$\text{Err} = \text{E}[\mathbf{1}\{y \neq \hat{c}^*(\mathcal{M})\}] \tag{6}$$

of the final chosen model $\hat{c}^*$ can be assessed, where $\mathbf{1}\{\cdot\}$ denotes the indicator function whose value is 1 if $y \neq \hat{c}^*(\mathcal{M})$, otherwise 0. Equation 6 therefore expresses the expected prediction error over the test set $\mathcal{M}$. We denote this third step as classification which produces the best model $\hat{c}^*$ for future prediction, and use AdaBoost, SVM and K-NN as the underlying classification algorithms.

2 Preprocessing

Different pipe inspection companies use different robot systems, the resolution of raw images therefore varies from city to city. These undesirable properties greatly complicate the segmentation of pipe images. Therefore, preprocessing digital image is required to make input images as homogeneous as possible (Figure 3). First, we cut the pipe image vertically to make the width of each pipe image be about three times of the ideal lateral radius R obtained by averaging the radii of all candidate laterals in the data set $\mathcal{D}$. Then, we rearrange the image so that the areas of water stain are replaced at the top and bottom of the image. For efficiency purposes, images are converted to gray scale. Next, we resize the image according to R ensuring that laterals will have similar sizes in pipe images. Last, we crop the image to remove areas of water stain. Preprocessed pipe images $\tilde{I}$ retain key lateral properties, dark color and circular shape.

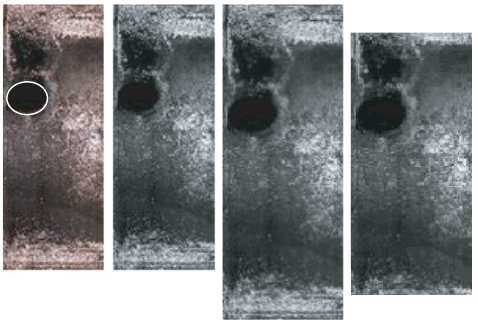


Figure 3: Intermediate steps of preprocessing (the lateral is marked with a white circle)

3 Segmentation

Segmentation is an important and difficult step in LD, which determines the eventual success or failure of our consecutive classification step. The objective of segmentation is to subdivide pipe images into regions where laterals are present and regions where laterals are absent, i.e., we would like to know which pixels make up a lateral in a particular preprocessed pipe image. Dozens of segmentation methods can be found in the literature. They range from histogram based thresholding techniques [15] over edge detection methods [17] and entropy based approach to K-means clustering [14, 16, 19]. The difficulty of segmentation, however, is that no universal technique is known that solve all segmentation problems. Domain specific knowledge is often required to find the best segmentation algorithm for a particular problem. In general, segmentation algorithms are based on one of two basic properties of intensity values: discontinuity and similarity. Therefore, before segmentation, some edge preserving filters should be applied. Such filters globally smooth the image and locally sharpen the edge pixels, so that all pixels that form an object (lateral) are clustered (similarity property) and edges are preserved (discontinuity property).

Sinha et al. mentioned in one of their papers [10] that some of the above methods failed to segment pipe images and proposed the combination of gray scale morphology (an edge preserving filter) and Otsu's thresholding as the underlying segmentation algorithm. But one disadvantage of gray-scale morphology is its introduction of undesirable structure in the filtered image, which often confuses the segmentation. Besides, Otsu's thresholding makes the assumption that histograms are bimodal and images are uniformly illuminated. Since these conditions do not apply to our pipe images, we use anisotropic diffusion in addition to gray-scale morphology to boost the overall segmentation performance. However, pipe images are still so noisy after preprocessing and filtering that simple thresholding fails. We thus introduce the region of interest, where we apply our own histogram thresholding instead of Otsu's to perform segmentation.

3.1 Gray-Scale Morphology

Because preprocessing preserves the key lateral properties, dark color and circular shape, we use the *closing* operator of gray-scale morphology with circular structuring element. Gray-scale morphology is an extension to basic binary morphology. Both input image $\tilde{I} = f(i,j)$ (preprocessed image) and structuring element $B = b(i,j)$ are treated as functions. The closing operator is simply the dilation of f by b which produces an image that is brighter than the original and in which small and dark details have been reduced or eliminated, followed by an erosion of the result by b which darkens the image without reintroducing the details removed by dilation. Therefore closing operator closes gaps, smooths contour sections, fuses narrow breaks and long thin gulfs, as well as eliminates small dark details in pipe images, whereas it simultaneously enhances the geometry and intensity properties of a particular lateral. Figure 5a shows a result of morphological closing.

3.2 Anisotropic Diffusion

An undesirable effect of gray-scale morphology is the strong geometric influence of a circular structuring element, which results in unwanted lateral-like pixel clusters. Anisotropic diffusion [7] avoids this effect.

To investigate the temporal change $\frac{\partial}{\partial t}$ of the gradient f_i in 1D we arrive at

$$\frac{\partial}{\partial t} f_i = \phi''(f_i) f_{ii}^2 + \phi'(f_i) f_{iii} \qquad (7)$$

where $\phi(f_i)$ is the flow function. Assume that at the location of an edge, the second derivative is zero

and the third derivative is negative. The right-hand side of equation 7 can be simplified to $\phi'(f_i)f_{iii}$. Hence we obtain

$$\phi'(f_i) > 0 \Rightarrow \frac{\partial}{\partial t} f_i < 0 \quad : \quad \text{smoothing}, \quad (8)$$

$$\phi'(f_i) < 0 \Rightarrow \frac{\partial}{\partial t} f_i > 0 \quad : \quad \text{sharpening}. \quad (9)$$

For LD, we set the diffusion coefficient of anisotropic diffusion to $\exp(-f_i^2/2\kappa^2)$, and obtain

$$\phi(f_i)' = \exp(-\frac{f_i^2}{2\kappa^2})(1 - \frac{f_i^2}{\kappa^2}). \quad (10)$$

Since $\phi'(f_i)$ is zero at $f_i = \kappa$, $\phi'(f_i)$ is positive for $f_i < \kappa$, and negative otherwise, we have sharpening for $f_i > \kappa$. We set κ according to the minimal edge strength calculated from the data set. It provides a threshold separating regions where smoothing is needed from regions where sharpening is needed. In Figures 5a and 5c a clear difference can be seen between gray-scale morphology with a circular structuring element of radius = 5 pixels and anisotropic diffusion with ten iterations of step size 0.2 and $\kappa = 15$.

3.3 Normalization and Thresholding

The goal of thresholding is to binarize the filtered image and make all lateral pixels as "foreground" object (black pixels) and all other pixels as "background" object (white pixels) based on discontinuity and similarity properties assured by applying gray-scale morphology and anisotropic diffusion filters. However, for filtered images as in Figures 5a and 5c, simple global thresholding fails because the lateral only resides in a small fraction of the image and dark pixels can be distributed anywhere, i.e., no bimodal distribution in the image. We therefore impose a constraint that is dictated by neighborhood size. The larger the size of the neighborhood, the stronger the constraint, and the more sensitive the solution is to the particular choice of constraint. The trade-off between large and small size is resolved by defining an appropriate neighborhood such that the histogram analysis becomes a local estimation which simplifies thresholding. For LD, we denote the region whose width is about twice of the R, and height is about the half of the filtered pipe image, as the region of interest (ROI), and define it as the ideal neighborhood in which we apply thresholding. Figure 4a shows the overlapped ROIs with their corresponding histograms after slicing the filtered pipe image into three.

To further simplify thresholding and make histograms "look" similar, we normalize ROIs. Most ROIs have intensity values between 0 and 255 after gray-scale morphology. We use simple contrast normalization method to ensure that the histogram of each ROI is shifted to zero and its highest intensity value is less or equal to $z = 85$ as formulated in Equation 11, where $\tilde{I}_{ROI}$ denotes a ROI, $q_{\min}$ as well as $q_{\max}$ are the smallest and largest pixel values in the ROI, and $\tilde{I}_n$ is the normalized image.

$$\tilde{I}_n \leftarrow (\tilde{I}_{ROI} - q_{\min}) \cdot \frac{\min(z, q_{\max} - q_{\min})}{q_{\max} - q_{\min}} \quad (11)$$

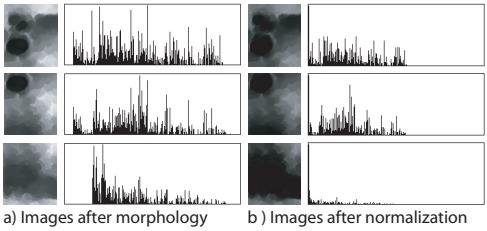


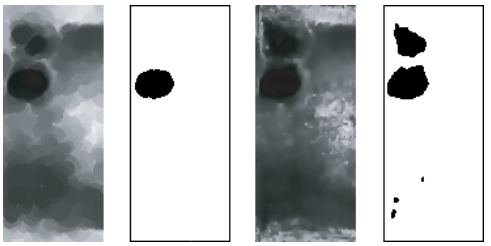
Figure 4: Depicts ROIs with their histograms. The top two ROIs on each side contain laterals.

The histogram of a ROI with a lateral differs from those without laterals after gray-scale morphology and normalization. Since the normalized histogram of a ROI has an exponential decay form in the most left part of the histogram (Figure 4b), our thresholding algorithm only investigates the lowest five intensity vales and chooses the brightest one subject to the empirical `valleyValue`.

For images filtered by anisotropic diffusion, no normalization (Equation 11) is needed. The thresholding algorithm simply counts the pixel values in the histogram from left-hand side to right-hand side and stops at the position where the total pixel counts exceeds a certain threshold determined by the largest lateral (with the largest pixel count) in the data set.

These two versions of our thresholding algorithms work due to the facts that lateral pixels are clustered in disjoint classes and laterals have the darkest pixels with circular edges in a pipe image. Figures 5b and 5d depict the results of the same pipe

image after theses two versions of algorithms. Figure 5d shows five objects, three of which are obvious not laterals because of their sizes. The one on the top is not a lateral either as far as roundness is concerned. Therefore, we use chain code algorithm to search black pixel clusters and perform region analysis to eliminate apparent errors such as huge diameters, distorted aspect ratios, skewed roundness and unrealistic form factors.



a) Image after morphology

b) Image after morphology and binarization

c) Image after anisotropic diffusion

d) Image after anisotropic diffusion and binarization

Figure 5: Binary images after gray-scale morphology (b) and anisotropic diffusion (d)

3.4 Region Analysis

Region analysis is based on some lateral features in binary images and is performed by checking whether these feature values satisfy the conjunction of the following constraints.

Feature Name	Constraints
height	$(0, (R+1) \cdot 6]$
width	$(0, (R+1) \cdot 6]$
aspectRatio	$[\frac{1}{6.25}, 6.25]$
roundness	$[0.5, 2.0]$
formFactor	$[0.5, 2.0]$

Table 1: Decision boundaries

Width and height are calculated directly from the chain code algorithm. Once they are determined, the aspect ratio is defined as the ratio between these two values. Roundness addresses the circular geometry of a lateral, which is computed as $Roundness = 4 \cdot Area/\pi/Length^2$, where $Area$ is estimated by chain code and $Length$ is the average of $width$ and $height$. An ideal lateral is a circle and hence its $Roundness$ value is 1.0. Form factor addresses the disk-form of a lateral and is calculated as $FormFactor = 4\pi \cdot Area/Perimeter^2$. An ideal lateral results in a $FormFactor$ value of 1.0.

By our region analysis we obtain n positions of possible laterals. For each position (i_k, j_k), $k \in [1, n]$, centered at which we create an RGB image patch (Figure 6) whose width and height are about three times of R and name it as a candidate lateral $C_k(i_k, j_k)$. Given all candidate laterals, LD becomes a binary classification problem, i.e., to classify candidate laterals (image patches) into two classes: image patches with laterals and image patches without laterals.

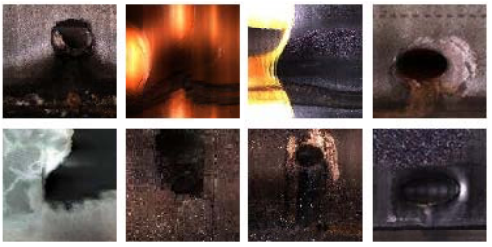


Figure 6: Candidate laterals

4 Classification

In previous sections we have shown the first two phases of our three-step algorithm. If the pipe image set were easy to separate, i.e., the laterals were perfectly shaped and colored, then the second step (segmentation) would suffice to detect laterals. Since our pipe image set spans a wider range of characteristics, segmentation alone is inadequate. Consequently, we use supervised learning techniques in addition to classify candidate laterals.

4.1 Feature Extraction

Feature extraction is to abstractly characterize objects (candidate laterals) by measurements whose values are very similar for objects in the same class, but very different for objects in a different class. The choice of the distinguishing features is a critical design step and depends on the characteristics of the problem domain. Besides, feature extraction is also an art, because once ideal features are found, then the job of the classifier becomes trivial. Sinha et al.

have used prior knowledge of laterals in the binary image domain to extract features such as size, form factor, aspect ratio, and area in their papers [9, 10]. For our LD, we would like to capture more intrinsic properties of laterals not only in the domain of binary images but also in the domain of gray-scale and RGB images. We thus propose nine new features in addition to $Roundness$ and $FormFactor$, and describe the two most important ones in detail.

4.1.1 Monte Carlo Area (MCA) and Random Area (RAE)

We use Monte Carlo method to estimate the possibility of being a lateral. First, we randomly select a pixel at a position (i, j) within the rectangular area (RA) of a candidate lateral and calculate whether $i^2 + j^2 \leq R^2$, and then check whether the pixel at (i, j) is darker than a random pixel outside the RA. If true, we increase a counting variable n. We repeat this random procedure m times, then the estimated area of solution (AOS) is $n/m \cdot RA$. AOS should be close to the value of optimal region (OR) defined by $(\pi r^2)/RA$. The relative error is used as a measure of MCA, i.e., $MCA = |(AOS - OR)/OR|$. RAE is a short version of MCA. It is the ratio between the counting variable n and the number of random procedure m.

4.1.2 Spatial Variation of Color (VAR) and Random Color Variation (RCV)

Inside a lateral we assume a homogeneous color distribution. Therefore, VAR (or RCV) is an appropriate measure to estimate the presence of a lateral. To quantify VAR (or RCV), we calculate the average of the sum of euclidean distances in RGB color space between the center pixel (i_k, j_k) of a candidate lateral C_k (image patch) and each pixel in the 5×5 neighborhood (or a random selected neighborhood for RCV). An image patch with a true lateral has a lower value of VAR (or RCV).

4.1.3 Other Features

Neighbor Color Ratio (RTO) is computed by taking the ratio between given points and their four neighbors (East, West, South, North). The average of these ratios is RTO. Random Ratio (RRO) is the ratio between n random points and their neighbors, the average of which is RRO. Monte Carlo Neighbor Color Ratio (MNR) differs from RRO in that the neighbor points are not within the same area, but outside. Clock (CLK) denotes the clock position of a candidate lateral in the pipe image.

4.1.4 Feature Representation

Each candidate lateral is represented as a feature vector in an eleven dimensional feature space. All these feature vectors are depicted in a scatterplot matrix in Figure 7, where LTL denotes the laterals (output of the classification). Feature abbreviations used in the scatterplot are summarized in table 2 with their corresponding feature names.

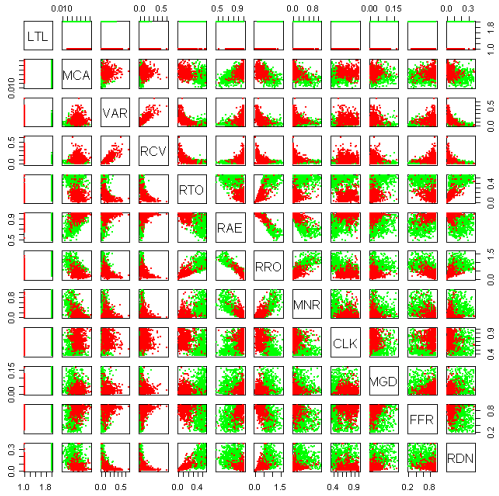


Figure 7: Scatterplot matrix of feature vectors (candidate laterals)

4.2 Classification with AdaBoost

Many classification algorithms are available in the literature. Algorithms such as SVM [14] and AdaBoost [14, 16] are prevailing in the pattern recognition community. However, these algorithms found their applications mostly in areas of face detection [11], real-time tracking and signal classification. For LD, Sinha et al. have used pairwise discriminant function in [10], K-NN and neuro-fuzzy network in [9]. We prefer AdaBoost for their better performance, adaptability and solid statistical learning background.

Feature Name	Abbreviation
Monte Carlo Area	MCA
Spatial Variation of Color	VAR
Random Color Variation	RCV
Neighbor Ratio	RTO
Random Area	RAE
Random Ratio	RRO
Monte Carlo Neighbor Ratio	MNR
Clock	CLK
Maximum Gradient	MGD
FormFactor	FFR
Roundness	RDN

Table 2: Feature Abbreviations

Boosting is one of the most powerful learning algorithms. It is a procedure that combines "weak" classifiers to produce a powerful "ensemble". The most popular boosting algorithm is called AdaBoost.M1 and was proposed by Freund and Schapire [3]. A weak classifier means that it performs only slightly better than random guessing in terms of error rate. The purpose of boosting is to apply weak classification algorithms sequentially to modified versions of data, producing a sequence of weak classifiers $c_b(\mathbf{x}), b \in [1, B]$, so that the weighted sum of these weak classifiers produce the final classifier

$$\hat{c}_B(\mathbf{x}) = \text{sign}\left(\sum_{b=1}^{B} \alpha_b c_b(\mathbf{x})\right) \tag{12}$$

where α_b are computed by the AdaBoost algorithm. The weights α_b give high influence to the more accurate classifiers in the sequence.

A decision stump is a two-terminal node classification tree $c_b(\mathbf{x})$

$$c_b(x_{k,p}, \theta_p) = \begin{cases} +1 & x_{k,p} > \theta_p \\ -1 & \text{otherwise} \end{cases} \tag{13}$$

and can be used as a weak classifier. Since the decision stump ignores all entries of $\mathbf{x}_k$ except $x_{k,p}$, it is equivalent to a linear classifier operating in the pth dimension of the lateral's feature space. The separating hyperplane is orthogonal to the pth axis, with which it intersects at $x_{k,p} = \theta_p$. To train the stumps, cumulative sum is used to determine the extremum θ_p^*. First the data is sorted in ascending order along the pth dimension. Then, $w_{cum} = \sum_{\{k|x_{k,p} \leq \theta_p\}} y_k$ is computed, i.e., the sum of weights on the left side of the threshold, while progressively shifting the threshold to the larger elements. Last, θ_p^* is the extremum of w_{cum} and whose sign determines the orientation of the separating hyperplane.

We use discrete AdaBoost.M1 [4] with decision stumps to classify candidate laterals and apply the learning rule

$$(p^*, \theta^*) = \arg\min_{p,\theta_p} \frac{\sum_{k=1}^{t} w_k \mathbf{1}\left\{y_k \neq c_b(x_{k,p}, \theta_p)\right\}}{\sum_{k=1}^{t} w_k} \tag{14}$$

where $\theta^* = \{\theta_p^* \mid p = 1..11\}$, t is the size of the training set and w_k is the data weight. In the training routine, the optimal parameter θ_p^* is found for each dimension p, and then the p^* is selected for which Equation 14 is minimized.

4.3 Cross Validation (CV)

We use tenfold cross validation to estimate the test (generalization) error and avoid overfitting. The data set $\mathcal{D}$ is randomly partitioned into 10 almost equally sized subsets d_i such that $\cup_{i=1}^{10} d_i = \mathcal{D}$, and $d_i \cap d_j = \emptyset$ for $i \neq j$, where $|d_i| = n/10$ forms the validation set $\mathcal{V}$. The data of size $n - |d_i|$ with indices not in d_i forms the training set $\mathcal{T}$ which is used to fit the classifier $(\hat{c}_B)_{n-|d_i|}^{-(d_i)}$. In other words, for the ith part (validation set), the model is fitted to other $10 - 1$ parts (training set) of the data, and the validation error of the fitted model is calculated when predicting the ith part of the data. This procedure is repeated for $i \in [1, 10]$ and the 10 errors are averaged. Therefore the tenfold cross validation error $\mathbf{CV}$ is

$$10^{-1} \sum_{i=1}^{10} |d_i|^{-1} \sum_{k \in d_i} \mathbf{1}\{y_k \neq (\hat{c}_B)_{n-|d_i|}^{-(d_i)}(\mathbf{x}_k)\}. \tag{15}$$

Having applied tenfold cross validation, the optimal ensemble model is found by calculating

$$\hat{c}_B^* = \arg\min_{\hat{c}_B \in C} \mathbf{CV} \tag{16}$$

where C is the class of models and $\hat{c}_B^*$ is the AdaBoost final ensemble model. Usually a local minimum (parsimonious model) instead of the global one is chosen for $\hat{c}_B^*$. The model $\hat{c}_B^*$ that minimizes Equation 16 gives the optimal model complexity in the sense of bias and variance decomposition [19].

Moreover, **CV** estimates the test error whose true value is assessed by applying the chosen model $\hat{c}_B^*$ to the test set $\mathcal{M}$ as shown in Equation 6.

5 Experimental Results

We have applied our three-step algorithm to pipelines of 10,000 meters in length or about 6,000 scanned images of real sewer pipes, where all laterals are manually detected and labeled for training purpose. These images are first preprocessed, and second about nine hundred candidate laterals are segmented from them by applying gray-scale morphology and anisotropic diffusion filters. Third, each candidate lateral is represented as an eleven dimensional feature vector and AdaBoost is used to classify these nine hundred feature vectors into two classes.

In the phase of classification, we use a training set to fit the model and observe that the more complex the model (sufficiently many weak classifiers), the lower the training error is. Figure 8 shows an apparent decreasing trend of error rates when performing AdaBoost on the training set. With 430 weak classifiers, the training error is zero (the green curve as a function of the number of weak classifiers). However, for zero training error we expect performance degradation on test set because of overfitting. To avoid it, we apply tenfold cross validation. One standard error bars are included for tenfold cross validation which is shown as the blue curve in Figure 8. To evaluate the true test error, we use an independent test set $\mathcal{M}$ obtained from new pipe images (about 250 new candidate laterals) and plot it as the red curve in Figure 8. We observe that the test error curve tracks the tenfold cross validation quite closely, which implies that the tenfold cross validation is indeed a good estimation of the true test error for our pipe images.

AdaBoost is robust in the sense that the tenfold cross validation error curve stabilizes after $b > 20$, where b is the number of weak classifiers. Since "one standard error" rule is often used with tenfold cross validation, we choose the most parsimonious model whose error is no more than one standard error above the error of the best model. Therefore, the optimal parsimonious model can be found around $b = 20$. We choose the model at $b^* = 20$ with 8.7% test error ($\pm 2.9\%$ standard error) as the final model for future prediction and observe that true

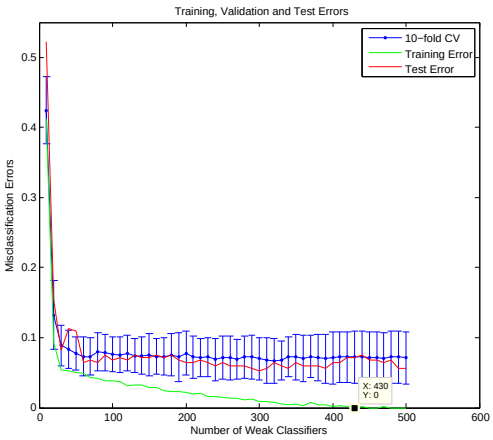


Figure 8: Training, validation and test errors

test error of the final chosen model is 9%. The enlarged version (Figure 9) of Figure 8 depicts this fact. An alternative view of AdaBoost's robustness is that only eight features (without RTO, RRO and RDN) are effectively selected for the chosen model with 20 weak classifiers.

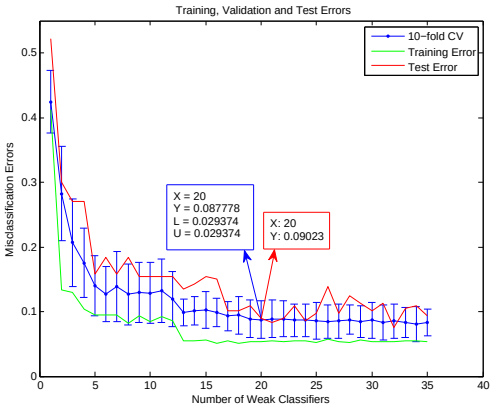


Figure 9: Errors, enlarged version

5.1 Comparison

We compare the performance of AdaBoost to nonlinear SVM (Gaussian `RBF` kernel) and K-NN. Obviously, for K-NN, where $K = 1$, we have zero training error. SVM can also make the training error as small as zero. Like AdaBoost, we are interested

in the prediction power on unseen data, therefore we also apply tenfold cross validation to find the optimal regularization parameter for SVM. Table 3 (Cross Validation error with one standard error: CV, and Test Error: TE) summarizes the performance of AdaBoost, SVM and K-NN (where $K = 1$ and $K = 3$). AdaBoost with 20 weak classifiers outperforms other methods which achieves $100 - 9 = 91$ percent accuracy on the test set $\mathcal{M}$.

Method	CV	TE
AdaBoost	8.7 ± 2.9	9
SVM	13.2 ± 2.1	15.2
1-NN	15.6 ± 2.3	18.1
3-NN	13.3 ± 2.3	16.3

Table 3: Comparison: Errors (%)

Preprocessing and segmentation (the first two steps) cause about one percent misclassification error, because not all laterals are successfully segmented from pipe images (extremely distorted laterals have neither circular shape nor dark color), the overall accuracy of our three-step algorithm is thus about 90%. It is worth mentoning that LD is primarily concerned wih the classification error, although false positive and false negative can be applied to denote the classification error separately.

6 Conclusion and Future Work

We have demonstrated our three-step approach to detecting laterals. Although the three steps in our algorithm are dependent, it can be regarded as a plug-in algorithm. In other words, classification accuracy may be improved by modifying each single step of the algorithm, i.e., use specific preprocessing methods like Fischer discriminant analysis [10], implement other edge-preserving low pass filters such as bilateral filter [12], or apply sophisticated segmentation algorithms like saliency detection [13] or K-means. Moreover, new features might be found that better depict the intrinsic properties of laterals and thus achieve lower classification error. Last, a variant of AdaBoost or other classification algorithms may perform better on our pipe images.

References

[1] R. O. Duda and P. E. Hart. Use of the Hough transformation to detect lines and curves in pictures. *Commun. ACM,* 15(1):11–15, 1972.

[2] H. Freeman. Computer processing of line drawing images. *Computing Surveys,* 6(1):57–97, 1974.

[3] Y. Freund and R. Schapire. A decision-theoretic generalization of online learning and an application to boosting. *Journal of Computer and System Sciences,* 55:119–139, 1997.

[4] J. Friedman, T. Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion). *Annals of Statistics,* 28:337–407, 2000.

[5] R. McKim and S. Sinha. Condition assessment of underground sewer pipes using a modified digital image processing paradigm. *Trenchless Technol. res.,* 14(2):29–37, 1999.

[6] N. Otsu. A threshold selection method from gray level histograms. *IEEE Trans. Systems, Man and Cybernetics,* 9:62–66, 1979.

[7] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.,* 12(7):629–639, 1990.

[8] M. Schindler and C. Stamm. Digitale Bildverarbeitung in der Roehreninspektion. *IMVS Fokusreport,* 1:34–40, November 2007.

[9] S. Sinha and P. Fieguth. Neuro-fuzzy network for the classification of buried pipe defects. *Automation in Construction,* 15:73–83, 2006.

[10] S. Sinha and P. Fieguth. Segmentation of buried concrete pipe images. *Automation in Construction,* 15:47–57, 2006.

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

[12] C.Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Sixth International Conference on Computer Vision,* 839–846, 1998.

[13] X. Hou and L. Zhang. Saliency detection: A spectral residual approach. In *Computer Vision and Pattern Recognition,* 1–8, 2007.

[14] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

[15] W. Burger and M.J.Burge. *Digital Image Processing, An Algorithmic Introduction using Java.* Springer, 2008.

[16] R. Duda, P. Hart, and D. Stork. *Pattern Classification.* John Wiley & SON, INC., 2001.

[17] R. Gonzalez and R. Woods. *Digital Image Processing.* Prentice Hall, 2002.

[18] L. Devroye, L. Gyrfi and G. Lugosi. *A Probabilistic Theory of Pattern Recognition.* Springer, 1996.

[19] T. Hastie, R. Tibshirani and J. Friedman. *The Elements of Statistical Learning.* Springer, 2001.