Train Detection and Tracking in Optical Time Domain Reflectometry (OTDR) Signals

Adam Papp[1(✉)], Christoph Wiesmeyr[1], Martin Litzenberger[1], Heinrich Garn[1], and Walter Kropatsch[2]

[1] Digital Safety and Security Department,
Austrian Institute of Technology GmbH, Vienna, Austria
{adam.papp.fl,christoph.wiesmeyr}@ait.ac.at
[2] Pattern Recognition and Image Processing Group,
Vienna University of Technology, Vienna, Austria

Abstract. We propose a novel method for the detection of vibrations caused by trains in an optical fiber buried nearby the railway track. Using optical time-domain reflectometry vibrations in the ground caused by different sources can be detected with high accuracy in time and space. While several algorithms have been proposed in the literature for train tracking using OTDR signals they have not been tested on longer recordings. The presented method learns the characteristic pattern in the Fourier domain using a support vector machine (SVM) and it becomes more robust to any kind of noise and artifacts in the signal. The point-based causal train tracking has two stages to minimize the influence of false classifications of the vibration detection. Our technical contribution is the evaluation of the presented algorithm based on two hour long recording and demonstration of open problems for commercial usage.

1 Introduction

Railway safety is an important task, as millions of people and cargo being transported every day. Conventional tracking is often achieved by bidirectional communication between train and track equipment. Other systems need detectors to be installed next to the railway and they deliver a signal when a train passes through the detector. In this paper we investigate a different technique, by the use of an *optical time-domain reflectometry* (OTDR) instrument. It injects a series of light pulses into a fiber cable and measures the scattered or reflected light (Rayleigh backscatter) at the same end of the cable. The refractive index change due to pressure change on the cable is the major cause for scattering, which can be measured with a positional resolution down to 0.5 m [1]. The interested reader on different type of OTDR devices and their physical principle is referred to the survey of Bao and Chen [1]. OTDR devices can be used to detect break points in fiber cable (e.g. underwater cable reparation), for intrusion detection [3,6,10,13] or even for rock slide detection [2].

Vibrations of train movements nearby the railway track can be monitored using an OTDR device, where in most cases fiber cables are already installed

B. Rosenhahn and B. Andres (Eds.): GCPR 2016, LNCS 9796, pp. 320–331, 2016.
DOI: 10.1007/978-3-319-45886-1_26

for communication purposes. The OTDR is an active measurement device as it injects light pulses into the fiber cable. The optical fiber at the railway track can be considered as passive, since trains do not need to communicate with the OTDR device. Several methods for train tracking using OTDR signals have been proposed in the literature [12,20]. However, these algorithms were not evaluated for longer recordings with many trains crossing. We will review the existing literature and describe the problem in detail in the following section. All the algorithmic details, as well as a discussion of the accuracy of the proposed method will be given in the later sections. We will further provide an outlook for future work and improvements.

2 Problem Description and Related Work

Our datasets were acquired with an OTDR device [8] and stored as a matrix, where each row is the measurements of the optical cable characteristics measured with one laser pulse. The dimension of the matrix is determined by the number of cable segments and the duration of the recording. The fiber cable consist of a fixed number of segments in each recording and a segment has a length of 0.68 m. The monitored section has two parallel railway tracks, includes a number of stops and a shunting yard. Figure 1 visualizes both input datasets. Although in this contribution the data along the complete time axis are available, the detection and tracking have been implemented for online processing using only a given chunk of measurements at a time.

Given the raw OTDR measurements $m(x, t)$ we want to detect and track trains, i.e. for each train that passes through the monitored stretch of railway tracks we want to automatically detect tuples with positions and times denoting the positions of the train at a given time. The quality of the tracking algorithm is then measured based on two main criteria: correct detection of all the trains passing the monitored track and the accuracy of the individual tracks. The rest of this section will be devoted to the discussion of the algorithms for OTDR signal processing and related motion tracking that can be found in the literature.

2.1 Signal Processing in OTDR

OTDR signals have been used in many different fields and many different signal processing techniques have been applied to these signals. In the following we provide an overview of the literature available.

For intrusion detection averaging methods were used [3,6], where the latest measurement is subtracted from the average. Averaging methods are accurate enough for intrusion detection where the position of the optical fiber can be chosen such that an event causes a large signal. For train vibration detection these methods proved not to be accurate enough due to the highly varying signal to noise ratio (SNR).

Qin et al. [14] introduced wavelet denoising in Φ-OTDR signal processing by thresholding the wavelet coefficients. They point out, that wavelet denoising

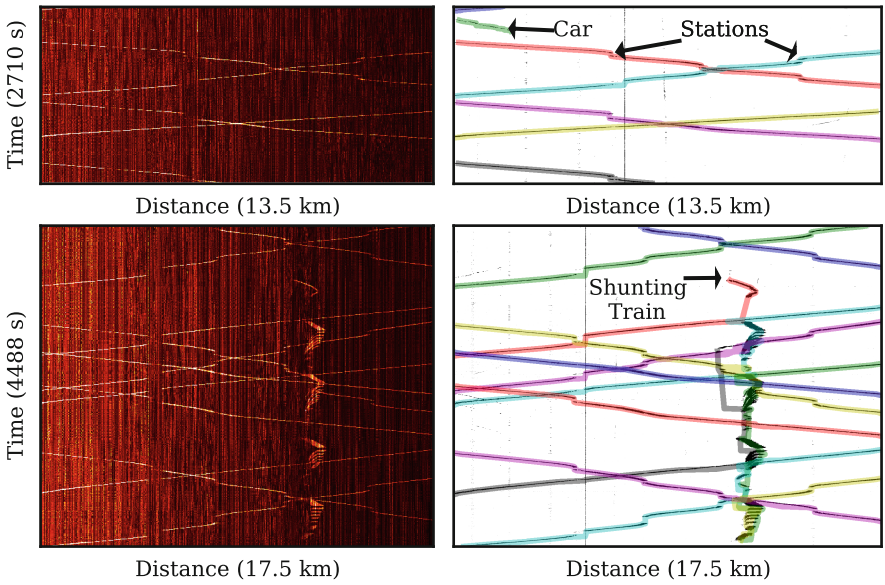

Fig. 1. The left side shows the input datasets, where each second of measurement is aggregated into one pixel. The pixel intensity is the energy of frequency coefficients between 50 and 150 Hz. The left-top data were taken with 4 kHz and the left-bottom were taken with 2 kHz. The right side shows the detection and tracking, where each tracked object is encoded with a different color. For clarification we labeled a tracked car, a shunting train and two of the stations. The two datasets were recorded with one minute offset to change the parameters of the recording device (Color figure online).

allows to remove noise that is present across all the frequencies while other frequently used methods (e.g. moving average) often only remove high frequency content from the signal. Wavelet denoising was applied by Peng et al. for pipeline intrusion detection [13] and for train detection [12]. To obtain locations of rising and falling edge of train signals normalized sliding variance was applied.

Using a spectral decomposition of the signal different authors have tried to distinguish between different events detected in OTDR signals. Kong et al. [9] have presented a method to detect events in OTDR signals based on Short-time Fourier Transformation (STFT) and correlation matching. Wu et al. [22] presented a method based on phase space matrix construction, eigenvalue decomposition and neural network classification to process signals from Φ-OTDR signals. Timofeev et al. [21] applied Gaussian Mixture Model (GMM) with Support Vector Machine (SVM) kernel to distinguish between different events (e.g. a walking man, a truck moving) in signals acquired with C-OTDR device. The (GMM-SVM) method was introduced by You et al. [24] for speaker recognition. The feature values they use are built using Linear-Frequency Spaced Filterbank Cepstrum Coefficients (LFCC) from 200 to 3000 Hz. Timofeev [19] applied

(GMM-SVM) also to detect activities (e.g. soil digging with excavator or working with crowbar) nearby the railway track in signals from C-OTDR devices.

We propose a Fourier transformation based method, as our tests have shown that it is possible to extract features to distinguish between vibration and background signals even with low SNR. Other methods such as wavelet denoising and averaging methods have been considered for our problem, but these need thresholds, which may vary under different circumstances. We found GMM not suitable for our problem since the two clusters for background and vibration signals do not have the shape of a Gaussian distribution. It is important to point out that we normalize the Fourier coefficients before classification, therefore the energy of the signal is not relevant for the classification. We have seen considerable variance of the energy of the signal present in the different cable segments.

2.2 Motion Tracking

This section gives an overview on motion tracking between image frames based on the survey of Yilmaz et al. [23] and we discuss the possibilities of point-tracking, kernel-tracking and silhouette-tracking for our purpose. Point-tracking relies on an external mechanism that detects objects in every frame. Deterministic methods solve the correspondence between consecutive frames by defining a cost between tracked points. Minimization of the cost can be formulated as a combinatorial problem [5,15–17]. Stochastic methods, e.g. Kalman filter [7], solve the correspondence by taking the measurement and model uncertainties into account. A Kalman filter can be only applied for one object, multiple objects need multiple Kalman filters.

Kernel-tracking e.g. mean-shift clustering proposed by Comaniciu and Meer [4] or KLT tracker proposed by Shi and Tomasi [18] are based on maximizing or comparing the similarity of appearance between consecutive frames by defining a feature vector for each tracked object or region. The proposed feature vectors extracted in this contribution are not reliable enough to apply kernel-tracking.

Silhouette-tracking is based on tracking objects e.g. hands by a complex descriptor of the region. Train signals acquired by an OTDR device after classification form in ideal case rectangular signals and do not require complex descriptors. Signal edges can be detected and a point-tracking mechanism can be adapted. It is essential to handle entry and exit of objects as trains are entering and leaving the measurement range. Multiple data association must be handled also, as multiple trains can appear in one time frame.

3 Algorithm

In the following we will describe our algorithm for train tracking, which is carried out in two main steps. In the first step, the beginning and end points of trains for a given second of time in the raw data are detected. In the second step these points are used for extracting attributes from trains i.e. position, speed and length for the given second.

3.1 Point Detection

The first step of the point detection is to classify points in time and space where vibration is present in the cable. To extract feature values for classification, Fourier transformation is used. For each cable segment we compute the Fourier transform for each second to receive a high dimensional feature vector $d_{i,j}(k)$, where i denotes the cable segment, j denotes the position in time (in seconds) and k denotes the frequency channel of the Fourier transform. The applied transformation can be seen as a sparse Short-Time Fourier transformation on the complete signal. It is important to note that, even for cable segments which provide a noisy signal, the Fourier transform has a typical pattern when vibrations of a train are present, see Fig. 2, therefore detection can be also applied on segments with low SNR.

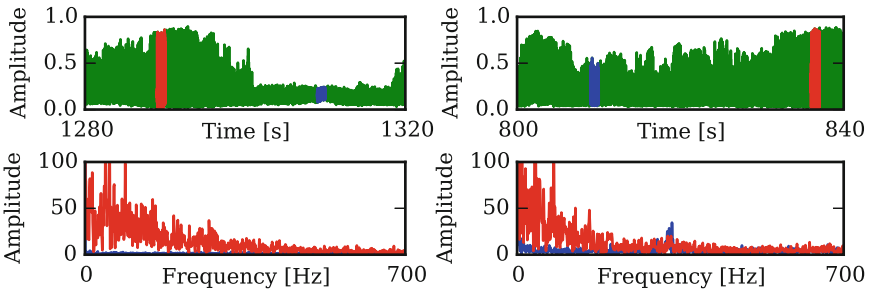

Fig. 2. The top-left and top-right images are part of raw input signals, where the top-left has high SNR and top-right has low SNR. From both signals two one second signals have been selected, one for train signal (encoded with red) and one for background signal (encoded with blue). The corresponding FFTs of the selected signals are visualized with the same color in the bottom images, where it can be seen that vibration can be detected even in signals with low SNR (Color figure online)

Using all Fourier coefficients as feature vectors would be expensive and many coefficients would be redundant or carry no information about the signal, therefore feature vector reduction is performed by scalar quantization. Each Fourier coefficient is assigned to a distinct bin and the reduced feature vector contains the normalized sum of the corresponding coefficients for each bin, where the normalization divider is the sum of all coefficients and the bins are linearly spaced. Principal Component Analysis (PCA) is then used to further reduce the feature vector. The reduced feature vectors are classified using a Support Vector Machine (SVM). The SVM was trained to make binary decisions between background-signal and vibration-signal using a Radial Basis Function (RBF) kernel. By an abuse of mathematical notation we can write the complete event detection process as the composition of the following steps: $r_{i,j} = \text{SVM} \circ \text{PCA} \circ \text{QUANT} \circ \mathcal{F}$. The complexity of the detection is linear in

both time (seconds) and space (number of segments), while due to Fourier transformation it has logarithmic complexity with respect to the sampling rate of the OTDR device.

The classification for each second of the original measurements results in a vector of binary classification $r_{i,j}$, where i denotes the cable segment and j denotes the position in time (in seconds). For a given second in time, the classifications of the last 10 s are summed along the time axis resulting in a vector describing the number of positive classifications in one cable segment within the time window. On this vector Gaussian filtering is performed along the space axis. Thresholding this vector leads to a filtered binary classification along space for a given second. Finally, the beginning (rising) and end (falling) points of trains are detected by taking the first derivative of the filtered results. Figure 3 visualizes the filtering steps.

3.2 Motion Tracking

Motion tracking is solved between two consecutive seconds as an optimization problem. The tracking is modeled in two stages. First, the beginning and end points of trains are tracked. These points can disappear when a train stops or when the trajectory of two trains are overlapping, therefore a second stage is defined for train tracking. Point tracking and train tracking are both solved as an optimization problem, where the following vertex sets are defined:

- V_{np} represents the new points detected after filtering of the classification results. This vertex set is redefined every second.
- V_{tp} represents the tracked points. Insertion and deletion in this set can be done in each second after solving the optimization problem.
- V_{tt} represents the tracked trains. Insertion and deletion in this set can be done in each second after solving the optimization problem.

For tracking in the two stages the following complete weighted undirected bipartite graphs are defined:

- $\mathcal{G}_p(V_p, E_p, w_p)$, where $V_p = V_{np} \cup V_{tp}$, $E_p = \{\{v_{np}, v_{tp}\} : v_{np} \in V_{np}, v_{tp} \in V_{tp}\}$
- $\mathcal{G}_t(V_t, E_t, w_t)$, where $V_t = V_{tp} \cup V_{tt}$, $E_t = \{\{v_{tp}, v_{tt}\} : v_{tp} \in V_{tp}, v_{tt} \in V_{tt}\}$.

To set up the optimization problems we define the following functions for vertices in the defined vertex sets:

- $u(v) \in \{-1, +1\}$ for $v \in V_{np} \cup V_{tp}$, determines if a point is rising (beginning of train) or falling (end of train).
- $p(v) \in \{0, ..., K\}$ for $v \in V_{np} \cup V_{tp} \cup V_{tt}$, denotes the position (segment), where K is the number of segments.
- $c_{tp}(v) \in [0, 1]$ for $v \in V_{tp}$, denotes the confidence of a tracked point, which is computed from the weighted sum of valid observations in time, the weights are forming a normalized exponential probability distribution function.
- $c_{tt}(v) \in [0, 1]$ for $v \in V_{tt}$, denotes the confidence of a tracked train, which is computed from weighted average of tracked point confidence observations. It also takes into account the number of rising/falling points.

- $s_{tp}(v) \in \mathbb{R}^+$ for $v \in V_{tp}$, determines velocity of tracked point, which is computed from the slope of the position of the last 10 s.
- $s_{tt}(v) \in \mathbb{R}^+$ for $v \in V_{tt}$, determines velocity of tracked train, which is computed from weighted average of tracked point velocity observations.
- $\Delta_T(v) \in \mathbb{N}^+$ for $v \in V_{tt}$, denotes the number of seconds since last update of tracked train.
- $\Delta(v_1, v_2) = p(v_1) - p(v_2)$ $v_1, v_2 \in V_{np} \cup V_{tp} \cup V_{tt}$, is a function that computes the distance between two vertices in the same graph.
- $\Delta_E(v) = p(v) + s(v)$ $v \in V_{tp}$, is a function that computes the expected distance between the current and the next position of the tracked point.
- $\sigma_{\mathcal{G}_p}(v)$ gives the set of all incident edges for a vertex $v \in V_p$ in graph $\mathcal{G}_p$.
- $\sigma_{\mathcal{G}_t}(v)$ gives the set of all incident edges for a vertex $v \in V_t$ in graph $\mathcal{G}_t$.

The weights for the edges $e_p \in E_p$ and $e_t \in E_t$ are computed as:

- $w_p(e_p) = w_p(\{v_{np}, v_{tp}\}) = c_e(v_{tp}) * \frac{1}{1+|\Delta(v_{np},v_{tp})-\Delta_E(v_{tp})|} \in [0, 1]$, which denotes the weight of an edge in $\mathcal{G}_p$, where $e_p = \{v_{np}, v_{tp}\} \in E_p$.
- $w_t(e_t) = w_t(\{v_{tp}, v_{tt}\}) = c_p(v_{tt}) * c_e(v_{tp}) * (a + (\frac{1}{\Delta_T(v_{tt})*|\Delta(v_{te},v_{tt})|})) \in [0, 2]$, which denotes the weight of an edge in $\mathcal{G}_t$, where $e_t = \{v_{tp}, v_{tt}\} \in E_t$ and $a = 1$ if both tracked point and tracked train moves in the same direction.

For the two graphs two separate Binary Integer Programs (BIP) are defined, where the objective functions are maximized. For each edge e in one of the two defined graphs we associate a binary variable x_e. For $e \in E_p$, the variable x_e is set to 1 if the corresponding new point is associated with the tracked point. Similarly, for $e \in E_t$, the variable x_e is set to 1 if the corresponding tracked point is associated with the tracked train.

Point tracking

$$\max \sum_{e \in E_p} w_p(e) x_e \quad s.t. \tag{1}$$

$$\sum_{e \in \sigma_{\mathcal{G}_p}(v)} x_e \leq 1 \quad \forall v \in V_p \tag{2}$$

$$x_e u(v_{np}) = x_e u(v_{tp}) \quad \forall e = \{v_{np}, v_{tp}\} \in E_p \tag{3}$$

$$x_e \operatorname{sgn}(s_{tp}(v_{tp})) = x_e \operatorname{sgn}(\Delta(v_{np}, v_{tp})) \quad \forall e = \{v_{np}, v_{tp}\} \in E_p \tag{4}$$

$$x_e \in \{0, 1\} \quad \forall e \in E_p \tag{5}$$

Train tracking

$$\max \sum_{e \in E_t} w_t(e) x_e \quad s.t. \tag{6}$$

$$\sum_{e \in \sigma_{\mathcal{G}_t}(v_{tp})} x_e \leq 1 \quad \forall v_{tp} \in V_{tp} \tag{7}$$

$$x_e c_{te}(v_{tp}) \geq 0.5 x_e \quad \forall e = \{v_{tp}, v_{tt}\} \in E_t \tag{8}$$

$$x_e \Delta(v_{tp}, v_{tt}) \leq 1500 \quad \forall e = \{v_{tp}, v_{tt}\} \in E_t \tag{9}$$

$$x_e \in \{0, 1\} \quad \forall e \in E_t. \tag{10}$$

In the following we will discuss the roles of the constraints in the linear programs. Constraint 2 ensures that each tracked point and each new-point can have only one association. Constraint 3 ensures that rising and falling points

Table 1. Cross-validation of train tracking with ground truth.

Train (ID)	23468	29515	2330	73	23488	Avg	Min	Max
LOOCV (meter)	38.24	4.89	37.76	64.92	48.11	38.78	4.89	64.92

cannot be associated. Constraint 4 ensures that the sign of $\Delta(e)$ and the tracked point velocity must be the same. Constraint 7 ensures that each tracked point can only be association with one tracked train, but multiple tracked points can be assigned to a tracked train. This serves as a noise filter, since points can appear from classification noise and no new trains should be constructed. False-positive edges often have a confidence lower than 0.5, therefore constraint 8 ensures that tracked points only with confidence higher than 0.5 are considered. Constraint 9 ensures that tracked points with large distance cannot be associated.

The BIPs are solved with greedy heuristics and the result is a list of tracked trains, each with the following attributes: position, confidence, velocity and length. The complexity of the greedy solution is derived from the complexity of the sorting algorithm in terms of the number of edge weights within the graph. The visualization in Fig. 3 shows an example of the tracking algorithm. The length of a train is computed by the standard deviation of tracked point position observations. Entry and exit of objects are handled as:

- new points $v_{np} \in V_{np}$ without association are promoted to vertex set V_{tp},
- if a tracked point $v_{tp} \in V_{tp}$ has no association, an empty-edge is assigned to decrease confidence,
- if a tracked point $v_{tp} \in V_{tp}$ reaches the confidence zero, it is removed from vertex set V_{tp},
- tracked points $v_{tp} \in V_{tp}$ without association are promoted to vertex set V_{tt},
- if a tracked train $v_{tt} \in V_{tp}$ has no association and the position is close either to the beginning or the end of the measurement range, it is removed from vertex set $V_{tt} \in V_{tt}$.

4 Evaluation

In the first step, parameters of the classification were investigated. We have received two datasets for the research with a total of around 169 million samples. For training and testing input data 10000 background and 10000 train samples were annotated by hand within the first dataset. To ensure non-overlapping samples, training data was taken from the first half of the dataset and testing data were taken from the second half of the dataset. Accuracy of the SVM classifier has been computed as (TP+TN)/(TP+TN+FP+FN). Frequency intervals from 1, 25, 50, 75, 100 to 200, 300, ..., 900, 1000 with bin count 5, 10, ..., 35, 40 were considered. Table 2 shows that as the beginning of the frequency interval increases, the number of TP decreases. Low frequency vibrations are present in a wider vicinity of a moving train than its high frequency vibrations. Leaving out

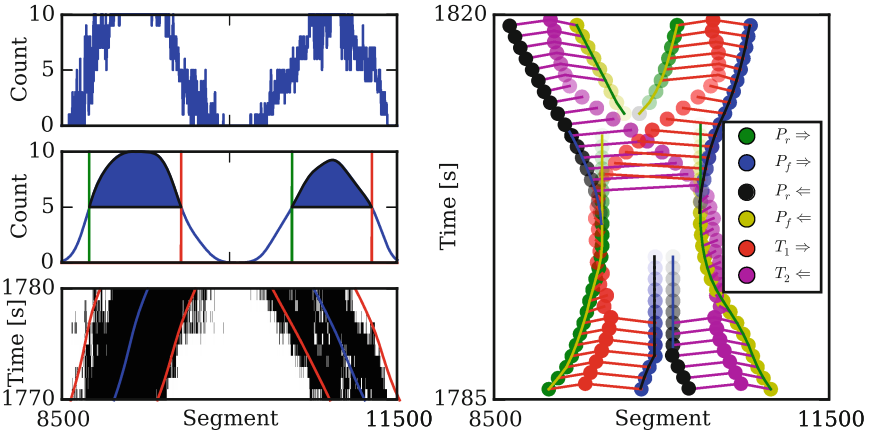

Fig. 3. The detection of begin/end points is visualized on the left side. The left-bottom image shows a 10 s window of the classification result. The left-top image shows the result of summation along the time axis as a one dimensional signal representing the count of positive classifications. The left-middle image shows the Gaussian filtering along the space axis. On the same image the filled section shows where threshold is exceeded, as well as the corresponding point detection, where green is the rising edge and red is the falling edge. On the left-bottom image the blue lines are the middle of trains and red lines are the head and tail of trains. The image on the right shows the train tracking graphs when the trajectories of two trains are crossing each other. Points (V_{tp}) are noted as P, the subscript represents rising/falling. Trains (V_{tt}) are noted as T, the subscript has no effect on the algorithm. Arrows represent the direction of points and trains. After time 1789 the points in the middle are not detectable any more and their confidence is starting to decrease, which is encoded as the alpha channel in the visualization. When their confidence is lower than 0.5, they are not associated to the trains and the confidence of the trains are decreasing also. In the middle of the image, the points change direction and new points are inserted to the set of tracked points. When the confidence of these points reaches 0.5, they are associated to the trains. At the top of the image, the points in the middle are tracked again and the confidence of the trains reaches 100 % again. (Color figure online)

low frequency bins from classification allows to locate trains with higher positional accuracy. The decline in accuracy without quantization shows that the implicit averaging of quantization is important. Reduction using PCA slightly decreases accuracy, but Table 2 shows a positive impact on the processing speed and using two eigenvectors the data can be reconstructed 87 %. Table 2 also shows that applying PCA directly to the Fourier transform is slower than using quantization first. The speed of the SVM classification is not only depending on the number of feature values, but also on the number of support vectors. The speed tests of the Python implementation were executed on a single core of an Intel(R) E5-1620 v3 CPU and the processing speed for one second data from both datasets were processed in around 2 s. We concluded from visual evaluation that false positive classifications have more negative impact on the point detection than false negatives.

Table 2. Accuracy and speed results of different parameters, where SVM was used for classification. FFT was executed on 5000 samples/segment. We show for each frequency interval the highest accuracy results. Note that quantization includes normalization.

Accuracy	TN	FP	FN	TP	FFT	Quant	PCA	SVM	Frequency	Bins	Reduction
%	Count				ns/segment				Hz	Count	Type
99.61	9970	30	47	9953	110.5	2.3	0.0	20.5	25-995	10	None
99.60	9970	30	50	9950	111.1	2.5	0.1	15.6	25-995	10	PCA
99.58	9969	31	53	9947	108.7	2.5	0.0	51.0	1-701	25	None
99.50	9967	33	66	9934	108.2	2.3	0.1	27.8	1-701	25	PCA
99.42	9972	28	88	9912	126.1	2.5	0.0	47.9	50-500	10	None
97.82	9981	19	417	9583	123.7	2.2	0.0	55.6	75-500	10	None
97.67	9974	26	440	9560	109.4	7.8	0.0	4647	1-701	700	None
97.28	9951	49	494	9506	111.0	8.1	3.6	139.1	1-701	700	PCA

The tracking algorithm was evaluated using cross-validation with ground truth data, see also [11]. Leave-one-out cross-validation (LOOCV) was used where a time offset was computed from $n-1$ trains and using this offset an absolute average of distances was computed as the validation value. From Table 1 the conclusion was made that the absolute average difference of the tracking is 38.78 m. Figure 1 shows the detection and tracking results. We find important to point out that all trains were found and tracked correctly. The detection algorithm was not trained to distinguish between cars and trains, therefore a car was tracked by the motion tracking algorithm. The data shown in Fig. 1 include the rail section of a shunting-yard at around kilometer 11 and tracking of trains are not satisfactory at that location. The average processing speed of the tracking algorithm for each second of data was around 10 ns.

5 Conclusion

In this paper a novel method was proposed to extract feature vectors from OTDR signals to distinguish between background-signal and vibration-signal. The algorithm is not dependent on the signal energy but classifies only based on the shape of the spectral distribution. From the given datasets train-signals were classified with an accuracy larger than 98 %. The algorithm is designed to work in real-time and processes raw data of one second in each step. Due to filtering of the classification the algorithm has a delay of 5 s in the current implementation.

Based on the classification a tracking algorithm has been presented that is able to automatically detect and track trains with an average of 38.8 m. To our knowledge this is the first algorithm that is reliably able to track trains automatically with crossings over a long period of time. In our data trains are tracked reliably up to a section with a shunting yard. The overlapping trains in that section lead to errors in the tracking algorithm therefore these cases have to be investigated in more detail. We plan to improve robustness of the tracking

method by including spatio-temporal features and eliminating parameters for train beginning/end detection, furthermore we plan to investigate if the tracking model can be simplified by tracking the center of trains in a single-stage manner.

Acknowledgements. The data used for the work presented in this paper have been collected in the research project *Sensorsystem für Bahnstrecken* funded by the Austrian Research Agency FFG under contract number 840448. We especially acknowledge Wolfgang Zottl (ÖBB Infrastruktur GmbH) for approving the measurements and Günther Neunteufel (Fiber Cable Technologies GmbH) for providing the data.

References

1. Bao, X., Chen, L.: Recent progress in distributed fiber optic sensors. Sensors **12**(7), 8601–8639 (2012)
2. Chen, C., Chen, R., Wei, F., Wu, D.H.: Experimental and application of spiral distributed optical fiber sensors based on OTDR. In: 2011 International Conference on Electric Information and Control Engineering (ICEICE), pp. 5905–5909. IEEE (2011)
3. Choi, K.N., Juarez, J.C., Taylor, H.F.: Distributed fiber optic pressure/seismic sensor for low-cost monitoring of long perimeters. In: AeroSense 2003. International Society for Optics and Photonics, pp. 134–141 (2003)
4. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. IEEE Trans. Pattern Anal. Mach. Intell. **25**(5), 564–577 (2003)
5. Jiang, H., Fels, S., Little, J.J.: A linear programming approach for multiple object tracking. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007, pp. 1–8. IEEE (2007)
6. Juarez, J.C., Maier, E.W., Choi, K.N., Taylor, H.F.: Distributed fiber-optic intrusion sensor system. J. Lightwave Technol. **23**(6), 2081 (2005)
7. Kalman, R.E.: A new approach to linear filtering and prediction problems. J. Basic Eng. **82**(1), 35–45 (1960)
8. Kanellopoulos, S., Shatalin, S.: Detecting a disturbance in the phase of light propagating in an optical waveguide, 11 September 2012, US Patent 8,264,676. https://www.google.com/patents/US8264676
9. Kong, H., Zhou, Q., Xie, W., Dong, Y., Ma, C., Hu, W.: Events detection in OTDR data based on a method combining correlation matching with STFT. In: Asia Communications and Photonics Conference, pp. ATh3A–148. Optical Society of America (2014)
10. Kumagai, T., Sato, S., Nakamura, T.: Fiber-optic vibration sensor for physical security system. In: 2012 International Conference on Condition Monitoring and Diagnosis (CMD), pp. 1171–1174. IEEE (2012)
11. Papp, A., Wiesmeyr, C., Litzenberger, M., Garn, H., Kropatsch, W.: A real-time algorithm for train position monitoring using optical time-domain reflectometry. In: IEEE International Conference on Intelligent Rail Transportation (accepted) (2016)
12. Peng, F., Duan, N., Rao, Y.J., Li, J.: Real-time position and speed monitoring of trains using phase-sensitive OTDR. IEEE Photonics Technol. Lett. **26**(20), 2055–2057 (2014)
13. Peng, F., Wu, H., Jia, X.H., Rao, Y.J., Wang, Z.N., Peng, Z.P.: Ultra-long high-sensitivity ϕ-OTDR for high spatial resolution intrusion detection of pipelines. Opt. Express **22**(11), 13804–13810 (2014)

14. Qin, Z., Chen, L., Bao, X.: Wavelet denoising method for improving detection performance of distributed vibration sensor. IEEE Photonics Technol. Lett. **24**(7), 542–544 (2012)
15. Rangarajan, K., Shah, M.: Establishing motion correspondence. In: 1991 Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 1991, pp. 103–108. IEEE (1991)
16. Sethi, I.K., Jain, R.: Finding trajectories of feature points in a monocular image sequence. IEEE Trans. Pattern Anal. Mach. Intell. **1**, 56–73 (1987)
17. Shafique, K., Shah, M.: A noniterative greedy algorithm for multiframe point correspondence. IEEE Trans. Pattern Anal. Mach. Intell. **27**(1), 51–65 (2005)
18. Shi, J., Tomasi, C.: Good features to track. In: 1994 Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 1994, pp. 593–600. IEEE (1994)
19. Timofeev, A.V.: Monitoring the railways by means of C-OTDR technology. Int. J. Mech. Aerosp. Ind. Mechatron. Eng. **9**(5), 701–704 (2015)
20. Timofeev, A.V., Egorov, D.V., Denisov, V.M.: The rail traffic management with usage of C-OTDR monitoring systems. World Acad. Sci. Eng. Technol. Int. J. Comput. Electr. Autom. Control Inf. Eng. **9**(7), 1492–1495 (2015)
21. Timofeev, A., Egorov, D.: Multichannel classification of target signals by means of an SVM ensemble in C-OTDR systems for remote monitoring of extended objects. In: MVML-2014 Conference Proceedings, vol. 1 (2014)
22. Wu, H., Li, X., Peng, Z., Rao, Y.: A novel intrusion signal processing method for phase-sensitive optical time-domain reflectometry (ϕ-OTDR). In: OFS2014 23rd International Conference on Optical Fiber Sensors. p. 91575O. International Society for Optics and Photonics (2014)
23. Yilmaz, A., Javed, O., Shah, M.: Object tracking: a survey. ACM Comput. Surv. (CSUR) **38**(4), 13 (2006)
24. You, C.H., Lee, K.A., Li, H.: GMM-SVM kernel with a Bhattacharyya-based distance for speaker recognition. IEEE Trans. Audio Speech Lang. Process. **18**(6), 1300–1312 (2010)