A data fusion approach for track monitoring from multiple in-service trains

George Lederman a,b,* , Siheng Chen b , James H. Garrett a , Jelena Kovačević b,c , Hae Young Noh a , Jacobo Bielak a

aCivil and Environmental Engineering, Carnegie Mellon University, USA
bElectrical and Computer Engineering, Carnegie Mellon University, USA
cBiomedical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA

Article info
Article history:
Received 24 October 2016
Received in revised form 12 March 2017
Accepted 17 March 2017
Available online 6 April 2017

Keywords:
Data fusion
Signal processing
Adaptive Kalman filter
Vehicle-based inspection
Inertial sensing

Abstract
We present a data fusion approach for enabling data-driven rail-infrastructure monitoring from multiple in-service trains. A number of researchers have proposed using vibration data collected from in-service trains as a low-cost method to monitor track geometry. The majority of this work has focused on developing novel features to extract information about the tracks from data produced by individual sensors on individual trains. We extend this work by presenting a technique to combine extracted features from multiple passes over the tracks from multiple sensors aboard multiple vehicles. There are a number of challenges in combining multiple data sources, like different relative position coordinates depending on the location of the sensor within the train. Furthermore, as the number of sensors increases, the likelihood that some will malfunction also increases. We use a two-step approach that first minimizes position offset errors through data alignment, then fuses the data with a novel adaptive Kalman filter that weights data according to its estimated reliability. We show the efficacy of this approach both through simulations and on a data-set collected from two instrumented trains operating over a one-year period. Combining data from numerous in-service trains allows for more continuous and more reliable data-driven monitoring than analyzing data from any one train alone; as the number of instrumented trains increases, the proposed fusion approach could facilitate track monitoring of entire rail-networks.

1. Introduction

Monitoring the health of a track network is important both to ensure the safety of its users, and to reduce maintenance costs by early detection of faults [1,2]. Track operators want inspection techniques that are low-cost and reliable, and that can detect faults soon after they occur. Currently, the most widely used methods of track inspection are visual inspection and inspection with track geometry cars [1]. The former is subjective, so can have low reliability, but tends to be employed frequently to identify major faults. The latter is more reliable, but given the high cost of operating dedicated track geometry cars, tends to be performed infrequently.
A number of researchers have proposed inspecting track networks using sensors aboard in-service trains \[3–17\]. Such a technique could be a low-cost method for providing near continuous inspection, particularly if enough in-service trains were instrumented. Most researchers have studied how to increase inspection reliability by developing new features that could be extracted from the raw data of individual sensors, such as wavelets \[3,4,18,19\], the Short Time Fourier Transform \[7\], signal standard deviation \[12\], and signal energy \[11\]. Others have proposed techniques to estimate the track profile \[5,8,14,16\], where the track profile itself serves as a feature of track condition. Rather than propose a new feature, here we introduce a novel data fusion approach that can take as input the features currently used in the monitoring community. As will be shown, our fusion approach not only enables more continuous monitoring by leveraging data from multiple trains, but also, by combining multiple sensors, increases the overall reliability of inspection from in-service trains.

Before describing our proposed method, we briefly discuss prior work on data fusion in the vehicle-based infrastructure monitoring space. To do so in a structured manner, we introduce the most prevalent data fusion model, the JDL Model (Joint Directors of Laboratories of the US Department of Defense), which has been developed and refined for defense applications since the mid-1980s \[21–25\]. In this model, data fusion is categorized into distinct “levels” within a data-processing pipeline. Level 0 Fusion is “Sub-object Data Association and Estimation,” Level 1 Fusion is “Object Refinement,” and Level 2 Fusion is “Situation Refinement.” In our proposed data processing pipeline, shown in Fig. 1, Level 0 Fusion means combining raw data from different sensors, for example, fusing position data with data about the track condition. Level 1 Fusion means combining object level data, in this case, features extracted from multiple passes over the same track, where the track section is the “object” of interest. Level 2 Fusion means combining situation level information, in this case, the detection outcome from anomaly detection. In our proposed pipeline, all the data are fused at Level 1, so no Level 2 Fusion is required.

One of the earliest studies in train-based track monitoring was conducted by Bocciolone et al. \[3\]. They developed a system to detect track corrugations in the Milan Metro by looking at the wavelet transform of data acquired from accelerometers mounted on the axle of a passing train. In their study, they performed Level 0 Fusion by combining accelerometer and position data; they then analyzed the accelerometer data in the wavelength domain which is position dependent. Their study does not go beyond feature extraction; they do not combine data from different passes over the track, different accelerometers, or different trains.

The same year, Weston et al. \[12\] published a study on monitoring track alignment using sensors placed on a train in the Tyne and Wear Metro. They propose a technique to extract the standard deviation of the track profile, which is a common parameter of track geometry used in traditional track geometry car inspection. In terms of data fusion, Weston et al. go beyond the work by Bocciolone et al. by presenting results from multiple passes over the same track separated by several months. They note regions in which changes in the track appear to have occurred, but they do not present formal Level 1 Fusion approaches or anomaly detection techniques.

While higher-level data fusion approaches have not been studied for train-based infrastructure monitoring, they have been employed in related fields. Eriksson et al. \[20\] propose a method for pothole detection using vibration data collected from cell-phones in taxi-cabs. In their study, potholes are defined as events which exceed a predefined threshold; several detection events from individual vehicles must occur in the same vicinity before a pothole is detected by the overall system. Essentially, data from each individual pass is analyzed independently all the way through anomaly detection; if a pothole is detected, this detection serves as a “vote” that a pothole has occurred at that location. Voting represents the most common type of Level 2 Fusion.

Performing anomaly detection directly on data from a single sensor and a single pass requires that the effect of the damaged track on the signal is known \emph{a priori}. This is the case in some train-based monitoring studies, for example, in the work of Molodova et al. \[9\]. They study rail-squats and define this type of damage as any point in the tracks where a particular feature exceeds a threshold set \emph{a priori}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{pipeline.png}
\caption{Proposed processing pipeline. The proposed data fusion approach in this paper is a Level 1 Fusion method. Level 0 Fusion is the combination of raw data, Level 1, of features, and, Level 2, of decisions. Although Level 2 Fusion is not required for the proposed pipeline, it describes a type of fusion used in related studies \[20\], and is thus included for completeness.}
\end{figure}
However, in many cases it is more desirable not to specify a threshold or the type of damage of interest \textit{a priori} so that a wider array of possible deterioration mechanisms can be detected. In these cases, a data-driven approach can be used to build a baseline estimate of the typical track features, then detect changes when deviations from this baseline occur. This ensures that sections of track with complex geometry do not trigger false alarms; in contrast, the Molodova et al. technique only works on tangent sections of track.

Building a baseline model from multiple passes and multiple vehicles requires a Level 1 Fusion approach; neither the Level 2 voting approach, nor the Level 0 combination of raw data proposed previously, fill this need. There are three challenges in combining data from in-service trains at Level 1. First, the trains do not necessarily pass over the tracks at regular intervals; thus the pass data is asynchronous. Second, position data, typically from GPS, has variable levels of accuracy, and data from different passes are often not well aligned spatially. Third, individual track sensors or entire train monitoring systems can malfunction or exhibit high noise levels. Because the sensors and the systems themselves can be reset, the reliability of the data changes with each pass.

To address these challenges, we propose a novel fusion approach. The method is based on the Kalman filter, which can handle asynchronous data. In addition, we propose an alignment technique as a pre-processing step to mitigate GPS error. Finally, in order to combine data from as many sensors as possible while ensuring malfunctioning sensors do not degrade the overall output, we estimate the sensor reliability, then weight each sensor accordingly.

We validate our approach on both simulated data and data collected from two in-service trains in Pittsburgh, PA. For the simulation study, the ground truth is known so the performance of the data fusion approach can be readily evaluated. For the operational data, the precise state of the tracks is not known at any moment; in this case, we investigate whether known changes in the tracks can be more readily detected after fusing the data versus analyzing the data from individual sensors independently.

The ultimate goal of the fusion approach would be to detect unknown changes (i.e. deterioration). That has not been analyzed in this study because it is not possible to evaluate the algorithm’s detection performance when ground truth for the deterioration is unknown. And determining the ground truth across an entire network is difficult if not impossible.

2. Algorithm

The goal of our data fusion approach is to estimate the state of the tracks using data from multiple trains. We perform Level 1 Fusion \cite{22,23}, combining features extracted from the raw signal of multiple sensors. In this application, the estimated track state is a feature representation of the state of the tracks; if the fusion is successful, the resultant feature representation is more accurate and consistent than if it were extracted from any individual sensor alone. As our proposed technique is based on the Kalman filter, we first provide a brief overview of the filter and its benefits.

The Kalman filter provides an estimate of the current state of a system by combining current and past data collected from that system. It is one of the most popular data fusion techniques because it is computationally efficient, works in on-line applications, and can process data collected asynchronously. For a linear process where the noise has a mean of zero, the estimate is optimal \cite{26,27}.

As an example, let $x_k$ be a vector describing the state of a linear system at time-step $k$, which relates to the previous state according to some constant $a$,

$$ x_k = a x_{k-1}. \quad (1) $$

Our goal is to estimate the state at each time-step from a set of noisy observations acquired from $j$ sensors, $z_{kj}$ (also a vector),

$$ z_{kj} = x_k + e_j, \quad (2) $$

where $e_j$ is some zero-mean noise specific to the sensor.

\textbf{Algorithm 1.} Kalman filter.

This algorithm is repeated for each time-step $k$.

\textbf{Predict} 

1. Estimate the state where $a$ is some known scalar constant describing the linear process of interest,

$$ x_k = a x_{k-1}. \quad (3) $$

2. Estimate the error variance of the prediction, $p_k$,

$$ p_k = ap_{k-1}. \quad (4) $$

\textbf{Update} 

(continued on next page)
**Algorithm 1 (continued)**

3. Update the Kalman gain, $g_{k,j}$, for each of the $j$ sensors. Here $r_j$ is the variance for each of the $j$ sensors; in the basic Kalman filter, the sensor variance is assumed to be known a priori,

$$g_{k,j} = \frac{p_k}{p_k + r_j}.$$  

(5)

4. Update the state estimate, $\hat{x}_k$, based on the current measurement data, $z_{k,j}$, from the $j$ sensors,

$$\hat{x}_k = \hat{x}_{k-1} + \sum_j g_{k,j} (z_{k,j} - \hat{x}_{k-1}).$$  

(6)

5. Update the prediction variance, $p_k$,

$$p_k = \left(1 - \sum_j g_{k,j}\right)p_k.$$  

(7)

The Kalman filter can be used to find an optimal estimate of the system from these observations, assuming $a$, a parameter of the linear process, is known. The steps for one iteration of the most basic Kalman filter are shown in **Algorithm 1**; these steps are repeated each time that new information is collected. Throughout this paper, we will use bold letters to denote vectors, and hat symbols ($\hat{}$) to denote values that are only estimates of the true values. First the estimate of the state of the system, $\hat{x}_k$, is predicted from the estimated state at the previous time-step, $\hat{x}_{k-1}$, multiplied by the known constant of the linear process, $a$, as shown in Eq. (3). The prediction error variance, $p_k$, provides an estimate of how much the prediction from the previous step can be trusted; this too is updated in Eq. (4).

While Eqs. (3) and (4) provide an a priori estimate, Eqs. (5)–(7) provide an a posteriori estimate, and thus are categorized under a new heading, “Update” [27]. The first step under this heading is to estimate the Kalman Gain for each of the $j$ sensors, $g_{k,j}$, that controls the trade-off between trusting the estimate from the last time-step versus trusting the newly observed data. This “trust” level depends on both the prediction error variance, $p_k$, and the sensor variance, $r_j$. Eq. (6) then provides the updated estimate, including the newly observed data, $z_{k,j}$. Finally, the a posteriori estimate of the prediction error, $p_k$, is calculated in Eq. (7).

**Algorithm 2.** Proposed Fusion.

This algorithm is repeated with each new $k$th pass over the tracks of interest.

1. Align Data I - find the optimal offset, $m_{k,j}$, between the measured data at the $k$th pass from the $j$th sensor, $z_{k,j}$, and the estimated state at the previous pass, $\hat{x}_{k-1,j}$, by finding the value of $m_{k,j}$ that maximizes the cross correlation,

$$\text{argmax}_{m_{k,j}}\left(\sum_{n=1}^{N} \hat{x}_{k-1,j}[n]z_{k,j}[n + m_{k,j}]\right).$$  

(8)

2. Align Data II - determine the estimate of the correctly aligned data, $\hat{z}_{k,j}$,

$$\hat{z}_{k,j}[n] = z_{k,j}[n + m_{k,j}].$$  

(9)

For values of $n$ where $z_{k,j}[n + m_{k,j}]$ is undefined (i.e. $n + m_{k,j} > N$), set $\hat{z}_{k,j}[n] = 0$.

3. Kalman filter I - estimate the sensor variance, $r_{k,j}$,

$$r_{k,j} = (\hat{z}_{k,j} - \hat{x}_{k-1,j})^T(\hat{z}_{k,j} - \hat{x}_{k-1,j}).$$  

(10)

4. Kalman filter II - calculate the Kalman Gain, $g_{k,j}$. Here $p_{k-1}$ is the prediction error from the previous step,

$$g_{k,j} = \frac{p_{k-1}}{p_{k-1} + r_{k,j}}.$$  

(11)

(continued on next page)
Algorithm 2 (continued)

5. Kalman filter III - estimate the current state,

\[ x_k = x_{k-1} + \sum_j g_{kj} (z_{kj} - x_{k-1}) \]  

(12)

6. Kalman filter IV - update the prediction error,

\[ p_k = \left(1 - \sum_j g_{kj}\right) p_{k-1} \]  

(13)

If \( p_k < p_{\text{min}} \), \( p_k = p_{\text{min}} \).

For the case of track monitoring, we are interested in estimating the state of the tracks with each newly collected pass of the train over them; thus \( k \) is used to refer not to time-steps but to the pass numbers. Although the passes can be thought of as time-steps, we avoid using the word “time” as it could be confused with the time-domain data collected within each pass.

As discussed in the Introduction, there are three basic challenges in fusing data about the state of the tracks: the data is collected asynchronously; the data can be misaligned spatially; and individual sensors can be noisy or malfunctioning.

To address the first challenge, we select a Kalman filter as the basic data fusion technique. The filter can provide a new prediction of the state of the tracks with data collected from each new pass of the train. This is the core of the proposed data fusion approach we present in Algorithm 2.

To address the second challenge, we align observed data from each pass over the tracks so that they fit more closely with the previous estimates of the feature representation for the tracks. The misalignment occurs due to position error from the GPS such as multipath reception, a particular problem when the GPS receiver does not have a clear view of the sky. This error tends to be relatively constant for a given pass over a short section of track, because the multipath error is constant for a particular configuration of satellites relative to the train [28]. To align the observed data with the estimate of the state of the tracks, we determine the offset using cross correlation in Eq. (8), then provide an estimate of the properly aligned observed data, \( \hat{z}_{kj} \), in Eq. (9).

To address the third challenge, we estimate sensor variance for each sensor and each pass over a given section of track. Typically the error level of a sensor is constant for each pass. For example, if the sensor is malfunctioning, the collected signal is not useful over the entire pass. Conversely, if the signal appears similar to previous readings over the first half of a section of track, the second half of the signal also tends to match historical data. Thus, we calculate the sensor variance, \( r_{kj} \), for each pass according to the variance between the observed signal over the entire pass and the estimate of the track state over the entire length of the track section, as shown in Eq. (10). Because we calculate the variance with each pass, we make the Kalman filter “adaptive.” The majority of adaptive Kalman filters proposed in the literature also present novel methods for estimating sensor variance [29,30], so the present innovation follows this trend.

Eqs. (11)–(13) in our proposed algorithm follow closely from Eqs. (5)–(7) in the standard Kalman filter. Eqs. (3) and (4) from the standard Kalman filter have not been reproduced because in our application, the state of the tracks is predicted to remain the same between individual passes, so \( a = 1 \). Thus, we simply use \( \hat{x}_{k-1} \) in Eqs. (11)–(13) rather than \( \hat{x}_k \) as is used in Eqs. (5)–(7).

One concern in using a Kalman filter is that it is designed to estimate a system in steady-state; this too must be modified for our application. While the state of the tracks is estimated to remain constant, it could change at any time. Because of this, the prediction error should never be zero, as this would mean that the Kalman filter would trust only its prediction and ignore the newly observed data. In the standard Kalman filter, the prediction error \( p_k \) approaches zero asymptotically; to avoid this, we set a minimum level of \( p_k \) which we refer to as \( p_{\text{min}} \) as the last step in Algorithm 2. The value of \( p_{\text{min}} \) is the only parameter which must be set in our proposed data fusion method. As will be discussed in the next section, determining the optimal value for \( p_{\text{min}} \) is an important component of implementing our data-fusion approach. If \( p_{\text{min}} \) is too small, the model updates slowly after a change in the tracks; if \( p_{\text{min}} \) is too large, the filter is sensitive to noise in newly collected data.

3. Data fusion applied to simulated data

To test our proposed data fusion approach, we first apply it to simulated data. As mentioned previously, our data-driven approach could be applied to fuse a wide variety of features; one of the benefits of testing the approach in simulation is that more generic data can be generated and tested to show the generality of the approach. In addition, in simulation, the ground truth is known, so the performance of the approach can be readily quantified. This is important, not only for judging the effectiveness of the approach, but also for parameter selection.
3.1. Data generation

We simulate some feature vector over the length of the track in two different states as shown on the left of Fig. 2; this is a feature representation of the ground truth state, $x_k$. The change between State 1 and State 2 symbolizes some sort of track deterioration or maintenance that causes the tracks themselves to change.

The right side of Fig. 2 shows two examples of the observed features from the train. These are simulated by adding a random offset $m_{kj}$ to the true feature state, $x_k$, along with some noise $e_j$ scaled by a value $c_k$ which is constant for each pass, $z_{kj}[n] = x_k[n + m_{kj}] + c_k e_j[n]$, where
\begin{align}
m_{kj} &\sim \mathcal{U}(0, 50) \\
c_k &\sim \mathcal{U}(0, 2) \\
e_j[n] &\sim \mathcal{N}(0, 1).
\end{align}

Here, $\mathcal{U}(a, b)$ indicates a uniform distribution between $a$ and $b$, while $\mathcal{N}(\mu, \sigma)$ indicates a normal distribution with mean $\mu$ and standard deviation $\sigma$. Note that these random constants represent the challenges mentioned in Section 2: $m_{kj}$ is the offset representing GPS noise, while $c_k$ scales the noise in each pass indicating whether the observed data is accurate or noisy. Due to the low signal-to-noise ratio (less than 0.1 for some passes), determining the state of the tracks from a single pass provides an inaccurate estimate. However, in the case of train-based monitoring, particularly if several trains are instrumented, numerous passes can be collected, and the data from each pass fused to achieve a better estimate.

Fig. 3 shows 100 simulated passes over the track section of interest, assuming that for the first 50 passes, the track is in State 1, and for the second 50 passes, the track is in State 2. In Fig. 3a, the true state, $x_k$, is shown for $k$ passes where $k = [1, 2, \ldots, 50]$, while in Fig. 3b, the observed data for one of the $j$ sensors is shown ($z_{kj}$). In this simulation, we assume we have two sensors ($j = [1, 2]$) on a single train, so we assume $c_k$ is the same in both cases. This makes the fusion more challenging, because if one sensor is noisy, the other sensor is also noisy, so the estimate for that pass must rely more heavily on the estimate from the previous passes. Note that the observed data have been normalized such that the length of the vector is equal to 1: this is beneficial because noisy passes tend to have larger amplitudes (both in this simulation and in the data we have collected from our instrumented trains) so normalization helps by reducing the magnitude of these noisy passes.

3.2. Data fusion

We can now apply the approach presented in Algorithm 2 on the simulated data shown in Fig. 3b. The output, the estimate of the state of the tracks, is shown in Fig. 4. This figure bears remarkable similarity to the true state of the tracks shown in Fig. 3a, meaning the fusion approach is largely successful. Notice that for the first several passes, the estimate is not good, but by around the 20th pass, the filter produces an accurate estimate of the state. Immediately after the change from State 1 to State 2 at pass #50, the filter is slow to adapt to the change. The trade-off between adapting quickly to changes versus making the most accurate estimate of the track if it is in steady-state is controlled by the chosen value of $p_{min}$ in our proposed Kalman filter. In this example we have chosen $p_{min} = 1.5 \times 10^{-3}$; other values can be seen in Fig. 5. When $p_{min}$ is zero, the filter approaches a steady-state where it no longer considers new inputs; when $p_{min}$ is large, it never trusts its estimate
Fig. 3. The true state of the tracks and the observed state of the tracks, in terms of extracted features. (a) The track states over 100 passes. Each horizontal line shows one of the two track states from Fig. 2, with the first 50 passes showing State 1, and the second 50 passes showing State 2. (b) The data observed about the state of the system from a passing train. It is normalized such that the length of each vector is one ($\sum |z_i|^2/n = 1$).

Fig. 4. Estimate of the state of the track using the proposed approach. Qualitatively, this estimate appears to be successful as it is close to the true state shown in Fig. 3a.

Fig. 5. Outputs of the data-fusion pipeline with varying values of $p_{min}$. For (a) $p_{min} = 0$, for (b) $p_{min} = 10^{-5}$ and for (c) $p_{min} = 10^{-4}$. 

$\sum |z_i|^2/n = 1$.
too strongly, so it is susceptible to newly observed data that is potentially noisy. Because the ground truth feature state of the tracks is available for this example, we can determine the optimal value of $p_{\text{min}}$, that is, the value of $p_{\text{min}}$ which minimizes the error ratio between the estimated feature state and the true feature state, where error ratio is defined as,

$$\text{error ratio} = \sum_k \frac{||x_k - \bar{x}_k||_2}{||x_k - z_k||_2}.$$  \hspace{1cm} (15)

Here the numerator is the difference between the true state and the fused data, and the denominator is the difference between the true state and the observed data. Thus an error ratio of 1 would mean the fused data is no better than the observed data, and an error ratio of 0 would mean the fused data perfectly matches the true state. A plot of the error ratio for various $p_{\text{min}}$ values is shown in Fig. 6 revealing that $p_{\text{min}} = 1.5 \times 10^{-5}$ is approximately optimal. In cases where a change occurs in the tracks, the error function tends to be convex. For this simulated data, the data fusion reduces the noise from the observed data by more than 70%.

In addition to studying $p_{\text{min}}$, we can also study two other novel components of our data fusion algorithm. The first is our estimation of the position error (or offset) using the cross correlation of the observed data with the estimated state from Eq. (8). Fig. 7a shows a comparison of how well our estimated offsets match the simulated offsets for one of the two simulated sensors, truncated to show only the first 20 passes for clarity. The true offset and the estimated offset do not line up well for the first several passes because the estimate of the current state of the tracks is not accurate. By the 4th pass, the estimated offset and true offset line up closely.

The second parameter we estimate is the sensor variance, a measure of the noise in the data. When generating the data for this simulation, we multiplied the noise in each pass by a constant, $c_k$, shown in Eq. (14), as a way of representing the fact that the noise level tends to be constant for each pass. In our algorithm, we approximate the noise level using Eq. (10) by calculating the variance between the feature state estimate and the observed feature. In Fig. 7b we shows both the estimated sensor variance and the amount of noise which was originally added to the simulated data. Although the values themselves are quite different, both parameters follow a similar trend. One interesting phenomenon is the interplay between the amount of added noise and the estimated offset. Notice that for pass #13 the value of $c_k$ is high; at the same pass, there

![Figure 6](image_url)

Fig. 6. Effect of varying the value of $p_{\text{min}}$ on the error of the data fusion. Here, an error ratio of 1 is as bad as the observed data itself, while an error ratio of zero means that the fusion has perfectly reconstructed the ground-truth.

![Figure 7](image_url)

Fig. 7. Estimation of the simulated offset values and the sensor noise level. (a) Comparison of the estimated offset with the simulated offset for the first 20 passes. (b) Comparison of the estimated sensor variance with the simulated noise level, $c_k$, for the same 20 passes.
is a difference between the estimated offset and the true offset in Fig. 7a, likely due to the large amount of noise. If the estimated offset is incorrect, then the data are not properly aligned, so we might expect the estimated sensor variance to be higher. However, this variance does not appear to increase dramatically because the features we are fusing together are relatively smooth over the length of the track. If the feature were less smooth, our approach would be more sensitive to misalignment.

4. Validation on operational data

In this section we validate our data fusion approach on data collected from a light-rail network in Pittsburgh, PA. This data-set is convenient for exploring data fusion because it was collected from two trains operating over a one year period. To evaluate the data fusion approach, we analyze data from a section of track where a faulty joint was repaired; the goal is to examine whether the proposed data fusion approach can help detect this repair more accurately and more rapidly than by detecting the change using each sensor individually.

4.1. The data-set

The trains are operated by the Port Authority of Allegheny County; the Port Authority generously helped install and maintain the sensors for this research project. Our goal was to build a system that could be widely deployed; to that end, we used low-cost off-the-shelf components and placed most of the sensors inside the cabin where they were easy to install and were protected from the elements. For this study, we used data from the two uni-axial accelerometers mounted vertically on both trains, all four of which are Vibrametrics model 5102 ceramic shear accelerometers. In addition, we used position information from BU-353 GPS antennas placed just under the roof of the first train (shown in Fig. 8), and inside the interurban light housing on the second (shown in Fig. 9). The vibration data are sampled at 1.6 kHz from the accelerometers on the first train, and at 2 kHz from the accelerometers on the second; on both trains, position data are sampled at 1 Hz from the GPS antennas. Additional information on the instrumentation can be found in [11,31].

One of the challenges in this study is that the GPS units are low-quality, do not have differential capabilities, and do not have an unobstructed view of the sky. Together, these factors lead to low-accuracy position estimates. While this makes the analysis challenging, the GPS units are inexpensive (<$30) so they could be easily deployed, and our techniques are more general, as they would work with both low and high quality GPS data.

In addition to position uncertainty, the train’s speed can vary between passes over a section of track, further complicating comparisons between passes. This is explored in detail in [11]. Over most sections of track, the speed variation is not huge, but it is large enough that comparing the signals in the time domain does not make sense because different parts of the track are traversed at different points in time.

To address these challenges, we use the signal-energy feature [11], which has been shown to be robust to varying train speed. This feature consists of squaring the raw vibration data (which are collected in the time-domain), smoothing the squared signal (here we smooth over a period of 0.3 s), then interpolating the data spatially. Because the data is analyzed...
in the spatial domain, the varying speed does not affect the alignment of the signals, but it does affect the size of the vibrations. Speed variability can be thought of as a source of noise, which is handled by a combination of the feature selection and the proposed Kalman filter.

For the interpolation in this paper, we sampled at a rate of 1.5 points per meter. For our analysis, we consider a 1 km section of track; thus the observed data vector of the extracted energy feature, \( z_{kj} \), for each of the \( k \) passes and \( j \) sensors has a length of 1500.

Energy feature data for a sensor mounted on each train is shown in Fig. 10. Each horizontal line represents one pass where the color indicates the value of the feature. As in the simulation, each pass vector is normalized so it has a length of 1. By plotting each successive pass sequentially in rows, certain patterns of vertical lines emerge; this occurs because certain hardware in the tracks, like the two pieces of switch gear around 0.6 km, consistently cause high vibrations, resulting in large energy feature values. These vertical lines are not perfectly straight because of position uncertainty due to GPS error.

Fig. 10a and b shows 117 passes from the two sensors on Train 1. The dates of the passes are shown in Fig. 11a. A number of passes have very low values; this can happen for a variety of reasons like a malfunction in the acquisition hardware or because a very large vibration event occurred, and due to the normalization, all other values were reduced accordingly. The plots from each sensor are nearly identical; this is both good and bad. It is good because it means that sensors placed within the cabin of the train are relatively insensitive to position, so they can be placed wherever is most convenient in future instrumentations. It is bad because the two sensors are so closely correlated, that when fusing the data, relatively little additional information can be gained. Of particular interest is the vertical line around 0.1 km which was due to high vibrations from a faulty track joint; this joint was replaced on December 15, 2015, or after pass #72; accordingly, the energy at this location of the track is reduced.

Fig. 10c and d shows 89 passes from Train 2; the dates of the passes are shown in Fig. 11b. Here too we see a change in the data after the track joint was repaired, at 0.1 km, in this case after pass #47. Notice that there appears to be less GPS error in the data from the second train (i.e. the vertical lines are "straighter"). This likely has to do with the position of the GPS antenna. For Train 1, the antenna was placed under the metal roof so it had no direct view of the sky; in Train 2, the GPS was placed in the interurban light enclosure shown in Fig. 9 so it had a partial view of the sky.

If we view the data collected by each pass as an observation of the current state of the tracks, the question is whether we can combine data from multiple passes and both trains to build a better estimate of the state of the tracks than could be estimated from any one pass or any one train. In the next section we explore this question by combining the collected data using our proposed data fusion approach.

4.2. Data fusion

In this section we explore fusion both at the train level, i.e. combining extracted features from different sensors, and at the network level, i.e. combining extracted features from different trains. However, in both cases, perhaps the most important
The estimate of the state of the tracks found by fusing together features extracted from Train 1 can be seen in Fig. 12a. Our approach is applied in an on-line manner; the estimate at a given pass relies only on the data from that pass and the data from the previous passes. Note that the large features values from the switch gear at 0.6 km end up controlling the alignment process, so the vertical line around 0.6 km is almost perfectly straight. The energy from the faulty track joint at 0.1 km is less

Component of fusion is combining features between passes, because as we have seen, individual passes can have abnormally low values or high position offsets, both of which are handled by our proposed fusion approach.

The estimate of the state of the tracks found by fusing together features extracted from Train 1 can be seen in Fig. 12a. Our approach is applied in an on-line manner; the estimate at a given pass relies only on the data from that pass and the data from the previous passes. Note that the large features values from the switch gear at 0.6 km end up controlling the alignment process, so the vertical line around 0.6 km is almost perfectly straight. The energy from the faulty track joint at 0.1 km is less
Just as in the simulation, we can calculate how the error ratio changes with different values of $p_{\text{min}}$ which must be considered when fusing the data from both trains.

The entirety of the high values from the first switch (a vertical line in the figure) occurs prior to 0.6 km; in essence, the data from the first switch occurs at 0.6 km, with some high values occurring just after 0.6 km. In the case of the second train, the data from the second train is shifted slightly to the left when compared to the data from the first train. This alignment is something we must be aware of when fusing the data from both trains.

One benefit of the proposed data fusion approach is that there is only one parameter to set, $p_{\text{min}}$; however, a discussion of how that parameter was chosen is necessary. In the simulation (considered in the previous section), the optimal $p_{\text{min}}$ value could be found because the ground truth was known. In this case, the ground truth is unknown. One method might be to choose $p_{\text{min}}$ empirically such that the fusion approach appears to smooth the data but not smooth it so much that some track changes are obscured. Doing so indicates that the value around $p_{\text{min}} = 1 \times 10^{-5}$ is reasonable. However, a more quantitative approach is possible if a known change in the tracks occurs. In this case, considering the data from Train 1, we know that the track was repaired after pass #72. We can assume that the tracks are in one state (State 1) from pass 1–72, and another state (State 2) from pass 73–117. Using this assumption, we can calculate how the error ratio changes with different values of $p_{\text{min}}$ as shown in Fig. 13. Again we see a convex function and, in this case, we see that $p_{\text{min}} = 1 \times 10^{-5}$ is indeed optimal. In general, this technique only works for finding the optimal value of $p_{\text{min}}$ in cases where a known change occurs; without a known change the error function is not necessarily convex. However, this pseudo-simulation procedure could be used on a known change to estimate the optimal $p_{\text{min}}$ value for a section track.

Using the same $p_{\text{min}}$ value and the procedures outlined above, we can fuse data from both trains. The first step is to interlace the passes according to the date and time when they were recorded, as seen in Fig. 14b. If this were applied in an on-line setting, it would be equivalent to processing the newest collected data over a section of track as it was acquired. The estimate of the state of tracks from combining both trains is shown in Fig. 14a. In this case, we consider only two sensors on each train; however, different number of sensors on each train could be handled similarly. Because of the alignment step, the difference in GPS error between the trains is handled, and the combined estimate appears to be as consistent as the data from either of the trains alone.

Thus far, in applying our data fusion approach on operational data, we have only been able to offer qualitative evidence that combining multiple sources of data improves the estimate versus analyzing the features independently. Part of this stems from the fact that the ground truth of the state of the tracks in terms of the energy feature is not known. In the next section, we introduce an anomaly detection method to evaluate the proposed data-fusion as part of the complete data processing pipeline shown in Fig. 1.
4.3. Evaluation of data fusion

In order to evaluate the fusion of the operational data, it is first necessary to consider the objectives of rail operators when conducting an inspection. As mentioned in the Introduction, operators want inspection techniques that are low-cost and reliable, and detect faults soon after they occur. We will evaluate our data fusion approach in terms of the second two objectives, reliability and time of detection, because, in practice, the “ground truth” feature state of the tracks for the operational data is not known. Since data fusion is but one step in an overall data processing pipeline, we must introduce an anomaly or change detection approach (change detection and anomaly detection are used here interchangeably) to assist in evaluating the effect of our proposed Level 1 Fusion on the overall detection performance.

Previously, Lederman et al. [11] evaluated three change detection approaches and found a Haar filter to be the most effective for train-based monitoring when using energy features. Following [11] we will use the Haar filter to assess how accurately and how rapidly changes can be detected both in analyzing the extracted features independently and in analyzing the fused features. Fig. 15 shows an example of how change detection with a Haar filter works. A Haar template is shown in Fig. 15a; this template is convolved with the input data shown in Fig. 15b. The result, shown in Fig. 15c, highlights locations in the input data where a vertical change occurs. We are only interested in “vertical changes” given the plotting convention in this study where each pass is shown as a row; such changes indicate that the state of the tracks has changed from one pass to another. Note that the Haar template has two blocks, one where values are equal to 1 and another where the values are −1. The height of each block, what we will refer to as the support length, controls how many passes must be collected before a change is detected. In this example, the support length is 20, meaning 20 passes would have to be collected after a change before the peak change detection value will occur. We will explore a variety of different support lengths in Fig. 17.

Using the same parameters (support length of 20 passes over a track width of 40 m) we can apply the Haar filter to some operational data. The ultimate goal is to determine whether the proposed fusion algorithm makes change detection easier. Thus we apply the Haar filter on the features prior to fusion and after fusion. The results are shown in Fig. 16.
Fig. 16. Change detection results for different levels of data fusion. The raw data are shown on the left panels; the right panels show the result of applying the Haar filter. (a) The raw energy feature data from Train 1 and Sensor 1 for select passes of interest. (b) The resultant change detection output for the raw feature data. Notice the magnitude of erroneous changes (at 0.6 km) are higher than the true change at 0.1 km. (c) The data fused from both trains for select passes of interest. (d) The resultant change detection output for the fused data from both trains. This has a better result, with the magnitude of true change 1.95 larger than the magnitude of any other change.
examples, the change in the tracks at 0.1 km is detected to some extent. What differs is how well that change is detected relative to other erroneous changes. In Fig. 16b, the change detection results from analyzing raw features directly; the erroneous changes detected in the switch gear (around 0.6 km) are larger than the changes detected from the actual track change at 0.1 km. In this case, the “detection ratio,” the ratio of the magnitude of the true changes to the magnitude of the erroneous changes, is below 1. A similar change detection analysis on the fused data, Fig. 16d, produces a much improved detection ratio of 1.9.

In addition to the change from the repair of the faulty joint, a second change can be seen in the change detection results from the fused data. As discussed previously, a portion of the track (around 0.75 km) was tamped after pass #98. Although this change is nearly impossible to detect in the raw features, the change is evident in the fused data shown in Fig. 16c and the second highest change detection value occurs at 0.75 km and pass #98 in Fig. 16d. Although this tamping change is not the focus of this study, it is noteworthy as evidence that the proposed fusion technique might be helpful in detecting more subtle track changes.

Thus far we have assumed that the support length of the Haar template is always 20 passes; this is an important parameter to explore further as it controls how soon the changes could be detected. We can vary the support length while reporting the resultant detection ratio, as shown in Fig. 17. The results of analyzing the features independently are shown in blue; the detection ratio hovers around 1, meaning the change in the tracks is roughly equivalent to erroneous changes. In other words, true changes cannot be detected reliably. We only show support lengths of up to 30 passes for the raw data due to the limited number of passes.

Fusing data from a single train tends to produce higher detection ratios, shown in magenta in Fig. 17. The data from Train 2 offers particularly high detection ratios, perhaps because the GPS alignment between passes is less noisy. The detection ratio produced when fusing together all the data is shown in red. This complete fusion appears slightly better than Train 1 and slightly worse than Train 2 in terms of detection accuracy for a given number of passes. However, the benefit of fusing all the data together is evident in how soon a change can be detected (in days) rather than simply in terms of the number of passes.

To illustrate the strength of fusing the data from both trains, let us consider a hypothetical detection scenario using the historical data we have collected. In this case, the last pass recorded before the track was repaired was on Dec 8th, 2015 (repair work started the following day). Let us assume the minimum acceptable detection ratio is 2 (i.e. a detection event is triggered if one change is twice as high as all other changes). The question is how soon a detection event will be triggered assuming that the data are analyzed after each successive pass, and that in each analysis, Haar templates with different support lengths are tried. If only data from Train 1 were to be used, a detection ratio of 2 would be achieved on Jan 26th 2016, using a Haar template with a support length of 33 (as is shown in Table 1). This would mean detection would require 49 days. If only data from Train 2 were to be used, detection would occur on Feb 1 2016, with a support length of 17. Note that fewer passes are required from Train 2, but that the data was collected less often, so the overall detection time is longer. If data from both trains is considered (i.e. all the data is fused together), detection would occur on Jan 14th, 2016, with a

---

1 For interpretation of color in Fig. 17, the reader is referred to the web version of this article.
support length of 24. Thus by fusing the data from both trains, the change could be detected sooner than if either train was analyzed on its own.

5. Concluding remarks

In this paper, we have presented a novel data fusion approach for data-driven track-monitoring from in-service trains. The goal is to build a formal method to combine extracted features from multiple passes over the tracks from multiple sensors on multiple trains. We overcome some of the challenges inherent in combining data from different sources with a novel adaptive Kalman filter. The approach is computationally efficient and easy to implement as it requires choosing only a single parameter, prediction error variance, which can be approximated empirically or through simulation. We use a data-driven approach for detecting track irregularities, by comparing current data to a historical baseline. Combining features from multiple trains allows for a more reliable baseline and a more rapid detection once a change has occurred.

The benefits of our proposed data fusion approach are shown both with a simulated example, and an example from an operational train system. In the case of the latter, a faulty track joint was repaired, and data from two operational trains passing over the joint both before and after the repair was used to detect when the repair occurred. The performance of the proposed method indicates that fusing data helps to detect track changes and could be an important component in a data processing pipeline for network-level track monitoring from in-service trains.

Future work should explore fusing other types of features and data from heterogeneous sensors, particularly, the type of low-cost sensors found in smartphones. Allowing smartphone data to be included in track models would enable crowdsourcing for track monitoring. This would allow the proposed method to scale rapidly to much larger track networks. And while the initial data suggests this would work for sections of track and networks of any size, this should be explored in future work.

Acknowledgements

This material is based on work supported by the National Science Foundation through a Graduate Research Fellowship for Lederman under Grant No. 0946825, by National Science Foundation awards 1130616 and 1017278, and a University Transportation Center grant (DTRT12-G-UTC11) from the U.S. Department of Transportation. The authors also gratefully acknowledge the Port Authority of Allegheny County for their partnership, particularly David Kramer, and the many helpful workers of Amalgamated Transit Union Local 85.

References


