A Vision System for Monitoring Intermodal Freight Trains

Avinash Kumar, Narendra Ahuja, John M Hart
Dept. of Electrical and Computer Engineering
University of Illinois,Urbana-Champaign
Urbana, Illinois 61801
{avinash,ahuja,jmh}@vision.ai.uiuc.edu

U K Visesh, P J Narayanan, C V Jawahar
Center for Visual Information Technology
International Institute of Information Technology
Hyderabad, India
{ukvisesh,pjn,jawahar}@iiit.ac.in

Abstract

We describe the design and implementation of a vision based Intermodal Train Monitoring System(ITMS) for extracting various features like length of gaps in an intermodal(IM) train which can later be used for higher level inferences. An intermodal train is a freight train consisting of two basic types of loads - containers and trailers. Our system first captures the video of an IM train, and applies image processing and machine learning techniques developed in this work to identify the various types of loads as containers and trailers. The whole process relies on a sequence of following tasks - robust background subtraction in each frame of the video, estimation of train velocity, creation of mosaic of the whole train from the video and classification of train loads into containers and trailers. Finally, the length of gaps between the loads of the IM train is estimated and is used to analyze the aerodynamic efficiency of the loading pattern of the train, which is a critical aspect of freight trains. This paper focusses on the machine vision aspect of the whole system.

1. Introduction

Intermodal(IM) freight trains have become the most widespread and fastest growing portion of the North American Freight Railroads. Their traffic has grown from 6.2 million in 1990 to 11 million in 2004, an increase of 77 percent [1]. These trains are generally more than 1 mile long and their operating speeds can be as high as 79 miles per hour(mph). While traveling at such high speeds, IM trains suffer large aerodynamic resistance owing to the big gaps between IM loads, thus resulting in high energy cost. This is a timely issue because of the fuel crisis in the past 5-10 years which has led to indirect effects of increase in transportation cost. Therefore, it is necessary to make IM trains more fuel efficient. In the following paragraphs we briefly introduce terms relating to IM trains, reasons for more fuel consumption and how an intermodal train analysis can help in achieving fuel efficiency.

Each load of an IM train is placed on a long iron platform with wheels called as a *rail car* as shown in Fig. 1(a) and a series of such rail cars of different lengths are attached together to form a train. Loads of different sizes and types, as shown in Fig. 1(b-f), can be placed on each of the rail cars. We define the arrangement of these loads across the length of an IM train as the *loading pattern* for that train. Fig. 2 shows our notion of good and bad loading patterns. Thus, poor loading assignments between loads and railcars lead to large gaps in IM trains. In [4] it was found that such inefficient loading patterns contribute to considerable increase in aerodynamic penalties. A good loading pattern would reduce the air resistance by as much as 27 percent and the fuel consumptions by a gallon per mile per train [5]. Therefore, a vision based system is developed to measure the loading efficiency and provide feedbacks to terminals, i.e., train yards where the IM trains are loaded.

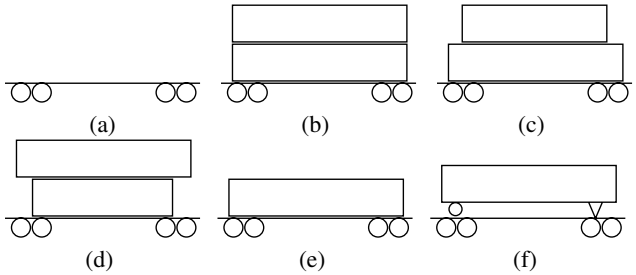


Figure 1. (a) Railcar (b-f) Different kinds of loads (b) Double Stack with upper and lower stack of same length (c)&(d) Double Stack with upper and lower stack of different length (e) Single Stack (f) Trailer.

A loading pattern analysis would involve measuring the gaps between consecutive loads of the train and then use this information to determine the aerodynamic efficiency of the loading assignment as in [5]. One way of doing this

IEEE Workshop on Applications of Computer Vision (WACV'07)
0-7695-2794-9/07 $20.00 © 2007 IEEE

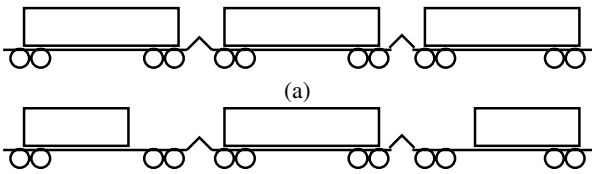


Figure 2. (a) good loading pattern - length of railcars match the length of the loads (b) bad loading pattern - smaller loads are kept on longer railcars leading to more aerodynamic resistance.

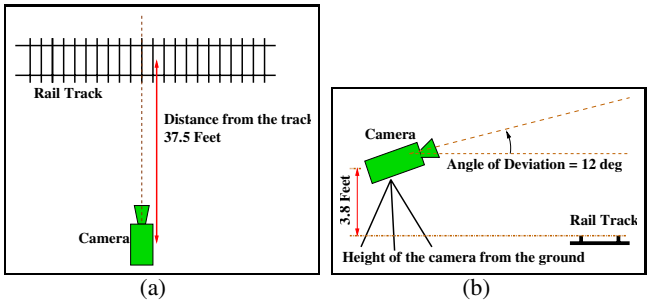


Figure 3. Setup (a) Top view (b) Side view

could be manually checking the length of the gaps of the train, which is a tedious process, especially if the length of the train extends to a mile. Our work intends to automate this whole process with less or no manual intervention. The main purpose of this research is to develop a camera based *automatic train monitoring system*, which will capture a video of a moving train and apply image processing and machine learning techniques to process this video. This task is made challenging by the fact that our system must be real time and handle various imaging conditions e.g., cloudy skies and dim light conditions. The prototype system we developed captures the video of a train, does background subtraction on individual frames, generates the mosaic of the train and then calculates the gaps of the IM train. The gap lengths are then used to calculate the aerodynamic efficiency of the train.

The system developed in [2] uses laser based techniques to analyze the wheels of trains. To our knowledge, there has not been any other such system developed before, which monitors the loads of a freight train, extracts useful features of the train and then does high level processing tasks e.g., calculate aerodynamic efficiency of the train, finding empty rail cars. In Section 2 we describe the camera setup and modules of the whole system namely background subtraction, mosaic generation and gap estimation. Section 3 shows results on detection accuracy of our system and robustness of gap estimation and mosaic generation.

2. System Overview

The whole system consists of two parts.
Camera Setup : A test location with high frequency of IM trains was chosen for capturing videos of the train. The speed of these trains at this location was mostly around 70-75 mph. A calibrated CCTV camera capable of capturing frames at resolution of 640x480 at 30 frames per second was placed facing the track. The camera would get activated and start capturing video as soon as the IM train comes into view. The camera setup at the test site had the parameters as shown in Fig. 3. **Software** : The software we developed is called as Intermodal Train Monitoring System(ITMS). A flowchart describing the complete system is shown in Fig. 4. The algorithm first takes the video obtained at the test site as input. It then extracts individual frames from the video. The images are not corrected for distortion as this could cost us more time in execution of the system. The background is then subtracted from each frame by using edge based and learning techniques (section 2.1). By correlating two consecutive background subtracted frames the velocity of the train is calculated in terms of *pixel shift per frame* (section 2.2). This velocity is used to create the mosaic of the IM train (section 2.3). After mosaic creation, the boundaries of the loads are detected (section 2.3) and the loads are classified (section 2.4) into containers and trailers. The mosaic is used to calculate the gap lengths between the loads (section 2.5).

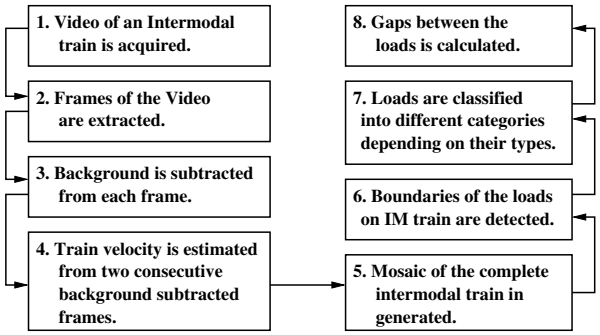


Figure 4. Flowchart of the Machine Vision Algorithm

2.1. Background Removal

The loads of an IM train can be broadly classified into *containers* and *trailers*. The containers are rectangular box shaped structures as shown in Fig 1(b-e). The trailers differ from containers in that they have wheels near their bottom as shown in Fig 1(f). The containers are stacked on rail cars in the following two configurations: *Single Stack* which has only one container and *Double Stack* which has two containers stacked over each other and placed on the rail car. Once the video of IM train is obtained, the next step is to separate the foreground from the background. The background is defined as any part of the image which does not

belong to the IM train e.g. sky, ground behind the train. See Fig. 5 for sample background and foreground frames from the videos we captured. A simple template based background subtraction algorithm does not work properly for our case, since the background changes dynamically e.g. clouds change position over the duration of train movement. Thus, for robustness of background subtraction, we adopted the following three stage algorithm.

- Entire background above the top of the loads (region marked Red in Fig. 5(d)) is removed using *edge detection* methods.
- Gaps between consecutive loads (region marked Green in Fig. 5(d)) is removed using *edge detection* based methods.
- The gap boundaries are not straight for gaps having double stacks with unequal lengths. To handle the background in the small region near the edge of the smaller stack (region marked Blue in Fig. 5(d)), we use an *adaptive* background subtraction method from [6] .

Each of these methods is explained below. The loads have

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

Figure 5. (a) and (b) Background template images with clouds,sky and fields (c) Foreground containing load (d) Regions where different subtraction algorithms are applied.

box shaped structure, which gets projected as a rectangular shape in an image. Thus a load can be characterized by a top edge and two side edges. The enclosed region corresponds to the load i.e. foreground, and the outside region is background. A gradient based edge detector is applied to each frame to obtain a binary image with edges of the loads getting the highest intensity value of 255. Due to overexposure, some of the detected edges may not be continuous, thus we dilate the edge image using a 5x5 mask. Fig 6 shows the edge detection and dilation results.

In this dilated edge image, we need to identify the top edge of the load. As the background usually contains structures like sky, clouds and bushes, which have low frequency components, the edge detection process detects very few edges from the background. Since the loads are almost rectangular in shape their top edge is assumed to be a straight line. Thus the first pixel location where the sum of intensities along x-direction peaks is taken to be the top of the

(a) (b) (c)

Figure 6. (a) Original Frame containing a load (b) Edges of the load detected (c) Dilated Edge image.

container. This is depicted in Fig. 7. The region above this pixel location in the image frame is considered to be background. Now, we remove background from the gaps lying

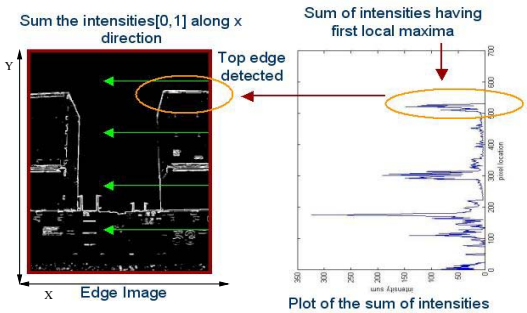
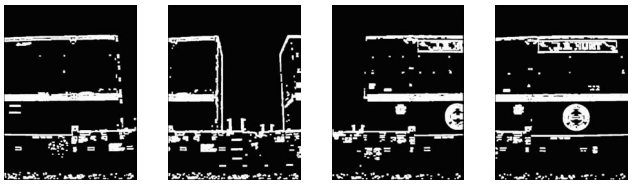


Figure 7. Detection of top edge of the load.

between the vertical boundaries of consecutive loads. Since the containers and trailers are long, only some portion of their length gets imaged in consecutive frames. In fact any load can be imaged in four possible configurations as shown in Fig. 8. Three of these configurations (a-c) contain gaps or part of the gaps. To detect these gaps, we start from the



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

Figure 8. (a) Left part of the gap is visible (b) Complete gap is visible (c) Right part of the gap is visible (d)No gap visible.

leftmost column of the image frame and look at the location of the highest edge pixels along y direction. These locations have higher y coordinate values for loads and lower values for gaps. We decide on a threshold Th, and whenever the difference in measurement in consecutive columns exceeds Th we signal the presence of left side of the gap. Similarly we repeat the process to find the right side of the gap. The threshold Th can be calculated as follows. The height of the rail car and the containers is fixed and can be obtained from freight train manual [3]. Assuming perspective projection, the height of rail car h_{rc} in image pixels is computed

using the parameters of the camera setup as shown in Fig. 3. Similarly we can calculate the height of a single stack(smallest in height among all loads) in image pixels as h_{ss}. Their difference i.e., $h_{ss} - h_{rc}$ is our threshold Th. Fig. 9 depicts the gap detection algorithm. The above algorithm

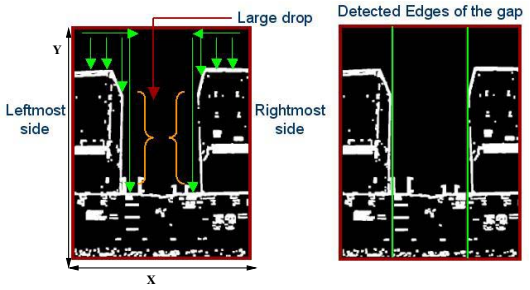


Figure 9. Detection of gaps in between loads.

is sufficient for detecting gaps, which do not have a double stack container with unequal length stacks on either of its sides(Fig. 5(c)). In such cases, the above technique based on edge detection only helps in removing a part of the gaps between longer stacks as shown in green color in Fig. 5(d). In order to remove background near the shorter of the two stacks (blue region in Fig. 5(d)), we apply an adaptive background subtraction method described in [6]. In this method the temporal pixel intensities obtained across frames at one particular location are modeled as a mixture of gaussians. In our work, we input all the intensities in background frames captured before the arrival of IM train (see Fig. 5(a,b)) and the intensities from the background regions detected using edge based method to learn the parameters of the gaussians corresponding to the background. Since we do not have any prior knowledge about the presence of such kind of gaps, we apply this adaptive algorithm near the boundaries of all the gaps detected using our previous edge based method. We thus use edge based and adaptive learning techniques for robust background removal as seen in Fig. 10.

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

Figure 10. (a) and (c) Example frames from a video (b) and (d) Corresponding background subtracted frames.

2.2. Velocity Estimation

In order to generate a mosaic, the next step is to detect the velocity of the IM train. We assume that the motion of the train is horizontal and there is negligible vertical motion. A correlation based technique is applied to get the pixel location where there is a best match between consecutive frames. Since 2D correlation is not very fast and our application should be real time, we approximate it with a 1D correlation. This is done by summing up the intensities in two consecutive images column wise and then correlating these summed up 1D arrays. The summing operation takes care of the slight motions in vertical direction. The array index of maximum correlation denotes the optimal pixel shifts between consecutive frames and is thus the velocity of the train in *pixel shift per frame*. Thus the estimated optimal velocity $v_{opt}(I_1, I_2)$ can be written as

$$v_{opt}(I_1, I_2) = \underset{v}{argmax} \sum_x \left(\sum_y I_1(x,y) \cdot \sum_y I_2(x+v, y) \right)$$

where, I_1 and I_2 are two neighboring image frames.

2.3. Mosaic Generation and Detection of Boundaries of the Load

Mosaic generation is important because it results in one big panorama of the train in which the loads are visible as a single complete block. The detection and classification of loads becomes easier on the whole mosaic, since it depends on the global properties of the complete load like length of the load, which is not visible in a single frame. To generate the mosaic, we extract a patch of pixels of certain width from the center of the frames and then paste these patches on one large image. The width of each patch is equal to the velocity estimate (as calculated in Section 2.2) of the train in the frame, from which the patch was taken. The reason being that the velocity estimate describes the amount by which the pixels have been shifted. Thus by selecting patches of length equal to the velocity we make sure that there is least overlapping region between consecutive patches when we create the mosaic. Since distortion is least in the center of the image, we choose the patch located at the center of the image. Fig. 11 shows our results on mosaic generation for one IM train.

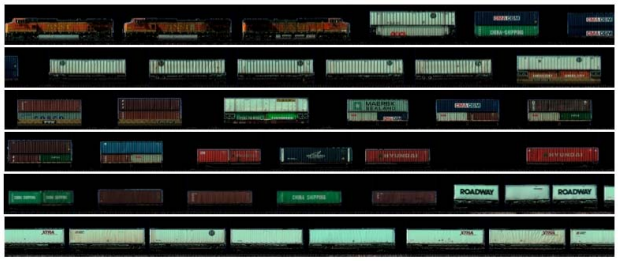


Figure 11. Mosaic of an intermodal train consisting of background subtracted loads.

Now, the foreground pixels in the mosaic are given a mask value of 1 and the background pixels of 0 to create a *foreground mask* image. To detect the boundaries of the

load in the mosaic, we count the number of foreground pixels in a column. Thus, for a mosaic of dimensions 640x1000 we get a 1D array of size 1x1000 containing the number of foreground pixels taken column wise. Since, the gaps have some foreground in the form of parts of the iron connectors between consecutive rail cars which are very less compared to that in the loads, we can apply k-means algorithm with k=2 (foreground in loads and foreground in the gaps) over this 1D array. Based on the two clusters we decide on a threshold value for obtaining the boundaries of the loads. For double stacks with same length containers, trailers and single stacks these boundaries correspond to correct gap edges. But for a double stack with unequal length containers, this method detects only the outer most boundary. The innermost boundary corresponding to shorter stack is calculated as a byproduct of the classification of the load into double stack as in section 2.4.

2.4. Classification of Loads

Once the edges of the loads are detected, the next step is to classify the loads into one of the following three categories - Single Stack, Double Stack and Trailer. The accuracy of this classification is important because based on a load being classified as a double stack, we look for the edge of the smaller stack in an unequal sized double stack configuration. The algorithm for load classification is described in the following subsections.

Single Stack Detection The single stacks differ from the other types of loads in that their height is small, roughly around 3-4 ft. From the load specifications on the height of a single stack [3] and using camera position and the height of the rail car, we can calculate the maximum possible height h_{ss} of a single stack in pixel values in an image. As explained in Section 2.1, we also have the height of the top of a load h_l. Thus if $h_l \leq h_{ss}$, we classify that load as a single stack. Since double stacks and trailers could be of same size, we cannot use similar techniques for identifying them.

Trailers The trailers are characterized by container shaped body but having wheels and an axle at the bottom. Due to the existence of a gap at the bottom of a trailer, the camera is able to view the base of the trailer. The base is characterized by low intensity values in the range of 0-10, as there is no direct natural light falling on it. To detect the trailer we look for a region of pixels near the base of the trailer, which falls in this low intensity range. If we are able to find such a region of pixels, we classify that load as a trailer. See Fig 12.

Double Stack Detection All the loads, which are not single stack or trailer, are assumed to be double stacks. The double stacks are characterized by two stacks of equal or unequal lengths kept on top of one another such that there is always a thin gap between the two stacks. The position of the camera is such that this gap is detected as a thin strip (2 or 3 pixels wide) of black line of intensity ~ 0. To detect the presence of this gap, we take a window of some size around the center of the double stack configuration as shown in Fig. 13. The intensity values in this window are projected horizontally along the x-direction by summing them up to give rise to a 1D array. The location of the minimum intensity value in this 1D array corresponds to the location of the *midline*, which is defined as the boundary line between the upper and the lower stack. See Fig. 13 for detection process for midline. To detect if the lower and upper stacks are of same size or different size, we choose two windows near the left boundary of the double stack, one of these is above the middle line and the other is below the middle line. We look for the presence of the edge of the smaller stack in these windows, by projecting the foreground mask profile(described in the second paragraph of section 2.3) along y direction(see Fig. 13) in that region and finding the location of steep change in projected profile which will correspond to the edge of the load. We repeat this process for the right boundary of the double stack. Thus we detect double stack containers along with the widths of the upper and the lower stacks.

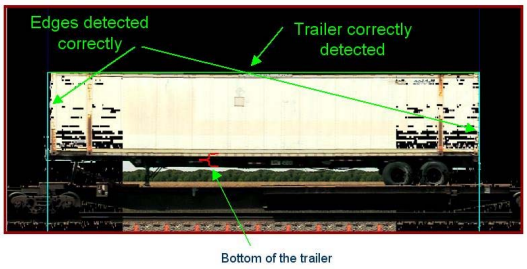


Figure 12. Detection of trailers. The small region near the bottom of the trailer that has pixels with low intensity is utilized for their detection.

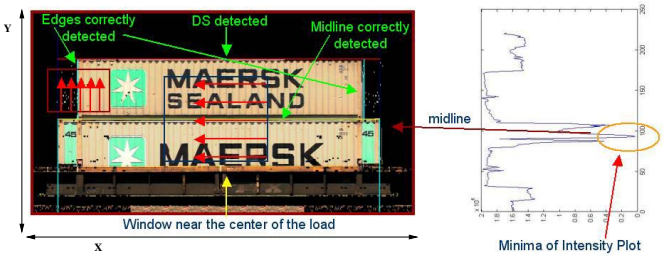
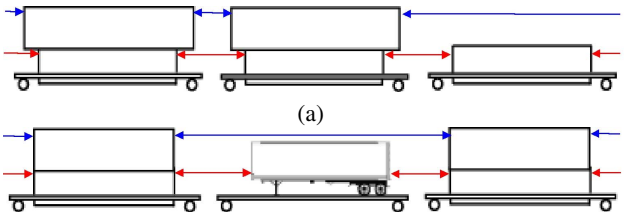


Figure 13. Detection of Double Stack.

2.5. Gap Detection

Once the stacks have been classified and their boundaries detected, we can then calculate the gaps between the loads. The gaps are divided into two categories - Upper Level gaps and Lower Level gaps. The upper level gaps

correspond to gaps between two neighboring upper stacks of a double stack configuration. All the other gaps are classified as lower level gaps. In Fig. 14 we show examples with upper level gaps as *blue* lines and lower level gaps in *red* lines.



(a)

(b)

Figure 14. Different types of gaps : (a) double stack-single stack pair (b) double stack-trailer pair.

3. Results and Conclusion

Once the gap lengths are detected they are sent to another system [5] which can calculate the aerodynamic efficiency of the IM train. The accuracy of this system critically depends on how accurately we detect the gap lengths. As the videos which we have were captured from various locations, we do not have ground truth data of length of the gaps in the IM trains. Thus we derived a measure of how accurately the length of the different loads were detected using our ITMS software. The railway manual [3] contains a description of the possible lengths of the double stack, single stack and trailer in feets. Let this set of lengths be defined as $L = \{l_1, l_2, \ldots, l_n\}$. From our camera setup we know that 40 feets = 1000 pixels. Using this conversion rate we convert the length of the containers into feet. Let this length be l. Thus we define the relative error per unit length of load,

$$Err(l) = \frac{|l_{i^*} - l|}{l_{i^*}} \quad \text{where} \quad i^* = \underset{i}{argmin}\, |l_{i^*} - l|$$

The above error is accumulated over all loads in an IM train to obtain *Panorama Generation Error* as shown in Table 1. The average error rate is less than 4%. Since errors in background removal directly effect velocity computation between two frames and the velocity estimate is used in constructing mosaics, we can infer that low panorama generation errors imply robustness of other modules. The software takes 5-7 minutes (depending on the train length) to output gap lengths on one intermodal train, which is quite comparable to real time systems. The algorithm was tested on 12 sets of videos captured at the test site containing 570 different kinds of loads - 245 double stacks, 84 single stacks and 241 trailers. These videos had various levels of difficulty e.g. clouds in the sky, videos with less exposure, waving bushes in the background. The classification recall rate for double stacks and single stacks was 100% and for trailers was 99%. The results of background subtraction and mosaic generation are shown in Fig. 10 and Fig. 11 respectively. Thus we have developed a vision based system which monitors an IM train, extracts important information and uses them for higher level inferences which in our case is calculation of aerodynamic efficiency [5] of the loading pattern.

Table 1. Panorama Generation Error

S.no	Date	Train index	Error
1	06/08	1	3.47%
2	06/08	2	1.71%
3	06/08	4	1.65%
4	08/07	1	2.67%
5	08/29	1	8.79%
6	09/10	3	3.32%
7	09/10	9	2.86%
8	09/11	1	6.82%
9	09/11	2	4.68%
10	09/11	3	2.84%
11	09/17	3	2.31%
12	09/17	5	4.97%

4. Acknowledgemets

The support of BNSF is gratefully acknowledged.

References

[1] Freight rail transportation: A review of the 2004 experience. Congressional Budget Office, 2004.

[2] BeenaVision. http://www.beenavision.com/aboutus.htm.

[3] R. Corporation. *UMLER Data Specification Manual*. Association of American Railroads(AAR), 2005.

[4] Y. C. Lai and C. P. L. Barkan. Options for improving the energy efficiency of intermodal freight trains. In *Transportation Research Record 1916*, pages 47–55. Transportation Research Board, 2005.

[5] Y. C. Lai, C. P. L. Barkan, J. Drapa, N. Ahuja, J. M. Hart, P. J. Narayanan, C. V. Jawahar, A. Kumar, and L. Milhon. Machine vision analysis of the energy efficiency of intermodal freight trains. *Journal of Rail and Rapid Transit*, 2006.

[6] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of International Conference of Pattern Recognition*, 2004.