Real-Time Train Speed Prediction using *Ip-Camera* Based On Time Difference

Eko Pramunanto
Computer Engineering Departement
Faculty of Electrical Engineering
Institut Teknologi Sepuluh Nopember
ekopram@te.its.ac.id

Eko Mulyanto Yuniarno
Computer Engineering Departement
Faculty of Electrical Engineering
Institut Teknologi Sepuluh Nopember
ekomulyanto@ee.its.ac.id

Arifandi Putrawardana
Computer Engineering Departement
Faculty of Electrical Engineering
Institut Teknologi Sepuluh Nopember
arifandi14@mhs.te.its.ac.id

Abstract — In 2014, PT. KAI has added Remote locomotive speed detection which is useful for increasing travel safety. The indicator will turn red if the speed of the monitored train exceeds the predetermined speed. This monitoring can only focus on overspeed only, while it needs to be given specifically at places such as train intersections, and stops. In This, a security system will be made for Monitoring Train Speed at certain points, such as train gates, intersections, and stations. In this case, the Real-Time Train Speed Prediction using IP-Camera is an effective and affordable solution to be applied. This system uses IP-camera to capture the input image and an SBC to process the image to produce the output direction and speed data. The Camera is used to detect the train, then the speed is calculated based on the different frames in the image input.

Index Terms—Computer Vision, Speed Detection, OpenCV.

I. INTRODUCTION

In 2014, PT. KAI has added a Remote locomotive speed detection device or locotrack in locomotive operation which is useful for increasing travel safety. The device works using a Global Positioning System (GPS) over a cellular network which can provide speed reports along the track directly to the Central Railway Operations Control Center. The monitor panel indicator will turn red if the speed of the monitored train exceeds the predetermined speed (Over Speed) [1]. The monitoring can only be focused on one Over Speed only, while special Over Speed needs to be given at places such as train gates, intersections and stops (stations).

Image processing technology is one of the fastest growing research fields in recent years. The emergence of pattern recognition methods, object shapes, and image processing-based measurements have been utilized in a variety of life applications. To be able to calculate the speed of a moving object passing through a path, the system must be able to detect objects recorded by the camera.

Therefore, this device will greatly assist monitoring and can support the Railway security system using technology in the field of image processing to detect train presence and calculate the speed. This system will take a picture when the train passes through a certain area to be input in the speed calculation process which will then be an integer result.

II. BASIC THEORY

A. Speed Measurement

The calculation process to get the speed of the vehicle from the captured video with moving objects can be calculated using the following equation [2]:

$$\text{Speed} = \frac{\text{Distance of Roi} \times \text{fps} \times 3600}{\text{Sum of Frame} \times 1000}\text{km/h} \quad (1)$$

B. Adaptive Thresholding

Based on the paper reference by Payel Roy, the conventional thresholding uses a global threshold for all pixels, where *adaptive thresholding* dynamically change the threshold value in the image. The author of this paper compares the various existing adaptive threslholding methods, so that the conclusion is that the Entropy method is the best of all comparisons, namely the Otsu, Rosin, and Kapur methods. In this research, adaptive thresholding is used to detect railroad tracks. This method is very effective and can be used in all conditions of time (morning, afternoon, evening, night). Adaptive threshold-ing is available in the opencv library so that it can be used immediately [4].

III. SYSTEM DESIGN

The system in this work is using image processing to measure train speed which aims to obtain speed data to monitor train speed at a certain point. The work process of the system is shown in the figure 1.

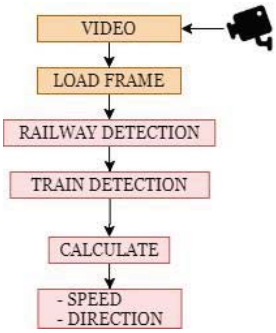


Fig. 1: Diagram flow system Real-Time Train Speed Prediction using Ip-Camera

978-1-7281-8283-4/20/$31.00 ©2020 IEEE

The frame is read after executing the **cap.read** command in a python program. Then the color reference will be determined taken from the colors around the rail line, in this case it requires a color range between the darkest to the lightest of the reference color. After that the program will filter (filtered) the colors contained in the frame and then display only the colors that are around the rail line. So that the results will come out from the threshold to detect the railroad tracks (result) as in the diagram 2.

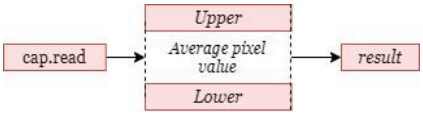


Fig. 2: Diagram flow Railway detection using *Adaptive Thresholding*

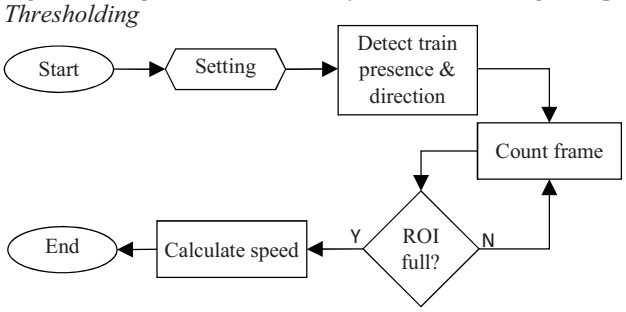


Fig. 3: Speed Calculation Flow Diagram

The process of finding a moving object in a sequence of frames is known as tracing. This tracking can be done using feature extraction and detecting objects or moving objects in the sequence of frames. By using the object position value in each frame we can calculate the position and speed of the moving object. The distance traveled by the object is determined using the center point of the bounding box. The distance is calculated using the euclidean distance formula. The formula for the euclidean distance in two dimensions is the displacement of a symbol from one point to another on the X and Y axes [2].

$$D = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \ldots . (2)$$

Where:
D = distance traveled
X_1 = position of point 1 on the X axis
X_2 = position of point 2 on the X axis
Y_1 = position of point 1 on the Y axis
Y_2 = position of point 2 on the Y axis

The value of D is used to find the number of frames contained in equation 1.

A. Data Acquisition

Video data acquisition in this study used an action cam camera mounted above and right in the middle of the train tracks. Based on the picture 4, camera is installed at a height of 8.45 meters from the rail line so that the camera is positioned right on the JPO "*Jembatan Penyeberangan Orang"* (Pedestrian Bridge) and set the camera's tilt angle of 39.8 degrees.

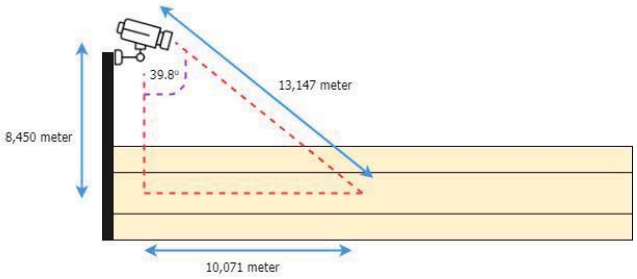


Fig. 4: Design Ilustration System

1) **Location** : The data acquisition location is on the railway next to Jalan Ahmad Yani, Surabaya City. Located next to the road in front of the Sunan Ampel State Islamic University campus in Surabaya, it can be seen in Figure 3.6 with latitude at -7.321775 and longtitude at coordinates 112.732959 on Google Maps [14].

At this location there is a JPO (Pedestrian Bridge) which makes it easy to take pictures because it takes advantage of its height so that the video captures can reach to record the RoI area with a predetermined distance, 10 meters long with a rail width of 1.069 meters and has one rail line so that it supports for video data retrieval of trains passing through this line.

(a) (b)

Fig. 5: (a) GPS Location; (b) JPO Jl. Ahmad Yani, Surabaya

2) **RoI** : (Region of Interest) used as a reference for the starting point and end point in the process of calculating vehicle speed. RoI is placed next to the rail with a distance of 10 meters as shown 6. The RoI distance is used as a comparison variable in testing this system.

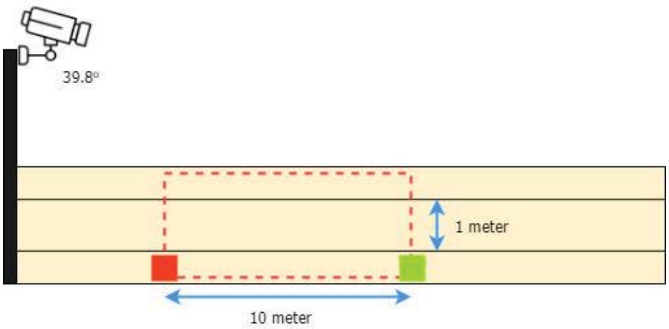


Fig. 6: Illustration of installing RoI

(a) (b)

Fig. 7: (a) Installing ROI ; (b) The RoI

TABLE I: Schedule of train attendance to be monitored

Schedule Presence	South Direction		North Direction	
	Nama KA	Time	Name of Train	Time
Morning	Penataran 449	7.35	Penataran 449	7.11
Noon	Probowangi 338	14.35	Probowangi 338	13.50
Afternoon	Penataran 453	18.15	Penataran 453	17.56
Night	Probowangi 338	21.54	Probowangi 338	21.10

3) **Capturing Video Data***:* Video data collection was carried out in the morning, afternoon, evening, and evening with several parameters according to the need for testing, namely different directions. Shooting video requires a marker that has been installed as RoI. After that, video recording was carried out when the train crossed the lane and then passed the JPO. The proposed resolution size is 1080x1920 pixels with a standard recording speed of 60 fps (frames per second). The vertical position of the camera can capture a longer rail image. The following is the schedule and type of train taken. Table I is the train schedule at the time of data collection, the time shown is the time when crossing the pedestrian bridge on Jalan Ahmad Yani.

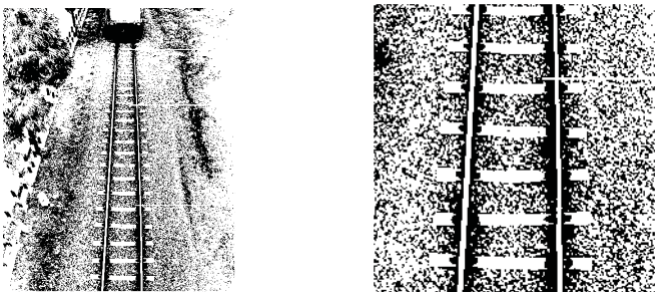
Every time the video shoots get two-way video data, namely south and north, first the northbound train passes the JPO first, a few minutes later, the southbound train will pass the JPO again.

B. Railway Detection

The rail detection process is carried out by means of Adaptive Thresholding [4]. First of all, all frames are averaged, then the program will determine the upper and lower values based on the average value. Then the threshold results will be seen in the form of a 2 bit image. After the rail has looked like in the picture 8 then it can be a reference for calculating the average pixel value like the methodology in the picture 3.

C. Train Detection

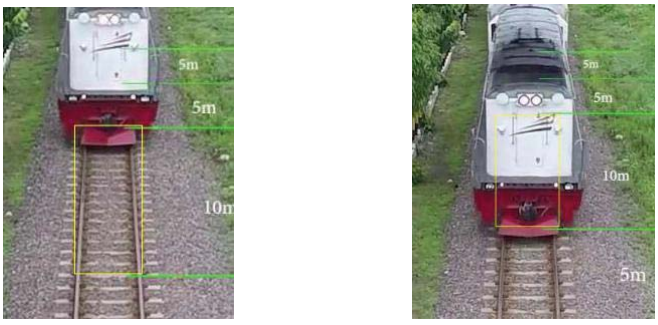
The next step is to create a crop area box according to where Roi is. The crop area in the program can be changed according to the desired RoI in the GUI. When a train enters the crop area from the top (1xB Pixel), the program will determine the direction, i.e. the train is going south. If the train enters the crop area from the bottom (AxB Pixel), the program will determine the direction, i.e. the train is going north. Then the directional results will immediately appear as output. After



(a) (b)

Fig. 8: (a) Railway Detection using *Adaptive Thresholding*; (b) Close up railway

that the program starts the calculation, after leaving the line the calculation process will end.



(a) (b)

Fig. 9: (a) Direction 1 ; (b) Direction 2 (reverse)

D. Train Speed and Direction

From the time the train enters RoI until it leaves RoI, the number of frames will be counted. The number of frames will be used as a comparison based on the number of fps in the video used. Because in the video 30 frames can be taken in 1 second, knowing the number of frames when passing RoI will get how many seconds the train passes the RoI.

Speed is calculated by means of equation 1. The result of the number of frames is obtained by means of equation 2 when the train is on RoI. On picture 10, is the result of running the program in the python shell, using an MSI laptop with an Intel Core i7 CPU and an NVIDIA GTX960 GPU. The speed shown is the result in km / hour, the frame count obtained is the number of frames when the train enters to exits the predetermined RoI.

IV. Testing and Result

This chapter presents the test results and analysis of the system design and implementation. The data used in program testing is obtained by recording a video of the traffic flow on the JPO which is taken using the action Cam from above the JPO right above the object. The video data that has been

(a)

Fig. 10: Result of Speed and Direction

```
Speed : 77.14 Count frame 14 [INFO]
elapsed time: 35.77 [INFO] approx.
FPS: 15.79
>>>>
```

Fig. 11: Display in python console

recorded is then processed by the program and tested with several different parameters. The tests carried out are divided into several parts, namely as follows:

1) Frame rate difference testing
2) RoI difference testing
3) Time (schedule) difference testing
4) Resolution difference testing
5) Real-time program testing

After calculating the number of vehicles manually and the system being compared, then the next step is to calculate the error value. The error value is obtained from the difference in the manual count and the number of systems then compared with the actual number multiplied by 100%. The calculation of the error value for each test can be found using the equation 3

$$G = \frac{\text{ManualResult} - \text{SystemResult}}{\text{M anualResult}} * 100\% \quad (3)$$

A. Frame rate difference testing

Frame rate difference testing aims to determine the effect of the amount of *frame rate* on the results of direction detection and train speed calculation. The amount of *frame rate* set in the video editing application Premiere Pro, with the video resolution used in this test is 1920x1080 pixels. The video used is when data retrieval with a RoI of 10 meters. There are 4 parameters for the number of textit frame rate used, namely 60 fps, 30 fps, 25 fps, and 15 fps.

In Table VI is a comparison between the calculation data for the amount of speed manually using sensory vision in the form of human eyes with the calculation of the speed in the system using the program. There are 4 types of trains with calculated time differences, P = morning, Si = Noon, So = Afternoon, M = Night and J = total.

TABLE II: Testing Table *Train Speed* with *Frame rate* difference

FPS	Manualy (km/h)					System (km/h)				
	P	Si	So	M	J	P	Si	So	M	J
60	65	71	68	90	294	68	72	70	0	210
30	65	71	68	90	294	72	67	73	0	212
25	65	71	68	90	294	75	69	77	0	221
15	65	71	68	90	294	77	67	83	0	227

TABLE III: Error Testing Table *Train Speed* with *Frame rate* difference

FPS	Galat (%)				
	P	Si	So	M	J
60	4,6	1.4	2.9	100	109
30	10.8	5.6	7.4	100	123,8
25	15.4	2.8	13.2	100	131,4
15	18.5	5.6	22.1	100	146,2

B. RoI difference testing

RoI difference testing aims to determine the effect of RoI distance on the results of detection of the direction and calculation of train speed. RoI is installed and placed in a pair according to a predetermined distance on the edge of the rail before taking data with the resolution used in this test, namely 1920x1080 pixels and a camera speed of 60fps. The video used is during daytime data collection and with a frame rate of 60 fps. There are 4 parameters of the RoI distance used, namely 10m, 15m, 20m, and 25m.

TABLE IV: Testing Table *Train Speed* with RoI difference

Meter	Manualy (km/h)			System (km/h)		
	dir 1	dir 2	Total	dir 1	dir 2	Total
10	71.00	69.67	140.67	71.88	69.67	141.55
15	77.14	68.94	146.08	75.13	68.74	143.87
20	75.65	67.50	143.15	75.65	67.37	143.02
25	74.90	68.35	143.25	74.90	67.85	142.75

TABLE V: Error Testing Table *Train Speed* with RoI differ-ence

Meter	Error (%)		
	dir 1	dir 2	Total
10	1.24	0	1.24
15	0.60	0.29	0.89
20	0	1.07	1.07
25	0	0.73	0.73

C. Time (schedule) difference testing

Time (schedule) difference testing aims to determine the effect of time on the results of detection of the direction and calculation of train speed. The number of frame rates is set

in the Premiere Pro video editing application, with the video resolution used in this test, namely 1920x1080 pixels. The video used is when data retrieval with a RoI of 10 meters. There are 4 parameters of the number textit frame rate used, namely P = morning, Si = Afternoon, So = Afternoon, M = Night.

TABLE VI: Testing Table *Train Speed* with Time (schedule) difference

Time	Manualy (km/h)					System (km/h)				
	60	30	25	15	J	60	30	25	15	J
P	65	65	65	65	260	68	72	75	77	292
Si	71	71	71	71	284	72	67	69	67	275
So	68	68	68	68	272	70	73	77	83	303
M	90	90	90	90	360	0	0	0	0	0

TABLE VII: Testing Table Error with Time (schedule) difference

Waktu	Error (%)				
	60	30	25	15	J
P	4,6	10.8	15.4	18.5	49.2
Si	1.4	5.6	2.8	5.6	15.5
So	2.9	7.4	13.2	22.1	45.6
M	100	100	100	100	400

D. Resolution difference testing

Testing with different resolutions aims to determine the effect of resolution on the results of detection of the direction and calculation of train speed. The video resolution is set in the Premiere Pro video editing application, with the video frame rate used in this test is 60 fps. The video used is when data retrieval with a RoI of 10 meters. There are 3 resolution parameters used, namely 1080p, 720p, and 480p.

TABLE VIII: Testing Table *Train Speed* with Resolution difference

FPS	Manualy (km/h)			System (km/h)		
	U	S	J	U	S	J
1080p	68	71	139	63	76	139
720p	68	71	139	65	82	147
480p	68	71	139	67	82	149

TABLE IX: Testing Table Error *Train Speed* with Resolution difference

FPS	Error (%)		
	U	S	J
1080p	7.0	7.4	14.4
720p	15.5	4.4	19.9
480p	15.5	1.5	17.0

E. Real-time program testing

Real time system testing aims to determine how capable the system is in real-time directional detection and calculation of train speed. At the time of data collection, the resolution used in this test is 480x852 pixels and a camera speed of 30fps. The video will be processed on SBC raspberry pie 3B + and will then provide direct results and speed. There are 2 parameters for the RoI distance used, namely 10m and 15m.

TABLE X: Testing Table *Train Speed* with real time program

Meter	Manualy (km/h)	Real Time (km/h)
10	71.00	90.00
15	68.74	90.00

TABLE XI: Testing Table Error with real time program

Meter	Error (%)
10	26.8
15	30.9

V. CONCLUSION

From the results of the tests that have been carried out, the following conclusions can be drawn as follow:

1) Real-time testing on the SBC takes time to process data by only getting video with a frame rate of 5 - 10 fps (uncertain) so that it affects the speed calculation with a high error, reaching 30.9%.
2) The system does not get any problems when it detects direction. There were no errors on all tests so the program could detect the direction accurately.
3) Testing the number of FPS on the camera, the smaller the FPS is inversely proportional to the bigger error, the smallest error is on the daytime train with 60 fps recording, which is 1.4%.
4) The longer the RoI distance, the bigger different the speed with manual calculations, but the speed calculation error will be lower.
5) Based on testing with time differences, the best speed calculation results are during the day, which results in the lowest error of 1.4%.
6) The light intensity at night affects the speed calculation with the largest error reaching 100%. This is due to the limitations of the camera in getting clear images at night.

REFERENCES

[1] Tingkatkan keselamatan, KAI pasang alat pendeteksi kecepatan ": https://ekonomi.bisnis.com/read/20140110/98/196539/tingkatkan-keselamatan-kai-pasang-alat-

[2] N. H. Tsani, "The implementattion of vehicle speed detection using webcam with frame difference method", bachelor thesis - Telkom University, August, 2017.

[3] Selvy Andy Wijaya, "Pengukuran Kecepatan Kendaraan berbasis Deep Learning", Tugas Akhir Departemen Teknik Komputer Institut Teknologi Sepuluh Nopember. 2019.

[4] Payel Roy, "Adaptive Thresholding: A comparative study", 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT).

[5] A. B. Tucker, Computer Science Handbook, Massachusetts: CRC Press, 2004.

[6] L. Peddireddi, "Object Tracking and Velocity Determination using TMS320C6416T DSK," 2008

[7] Hakan Koyuncu, "Baki Koyuncu, Vehicle Speed detection by using Camera and image processing software", The International Journal of Engineering and Science (IJES) —— Volume —— 7 —— Issue —— 9 Ver.III —— Pages —— PP 64-72—— 2018 —— ISSN (e): 2319 – 1813 ISSN (p): 23-19 – 1805.

[8] Scott Krig, "Computer Vision Metrics: Survey, Taxonomy, and Analysis", 2014.

[9] Richard Szeliski, "Computer Vision: Algorithms and Applications (Texts in Computer Science)", 2010.

[10] Ester Martinez-Martin, Angel P.del Pobil, ”Robust Motion Detection in Real-Life Scenarios (SpringerBriefs in Computer Science)”, Springer; 2012 edition.

[11] Thierry Bouwmans, Fatih Porikli, Benjamin Hoferlin,¨ Antoine Vacavant, ”Background Modeling and Foreground Detection for Video Surveil-lance”, 2015.

[12] Fabrizio Romano, ”Learning Python: Learn to code like a professional with Python - an open source, versatile, and powerful programming language”, 2015.

[13] L. Peddireddi, ”Object Tracking and Velocity Determination using TMS320C6416T DSK”, 2008.

[14] Lokasi Pengambilan Data, https://goo.gl/maps/se6bCcFBbQzrTNvr8