AUTOMATIC TRAIN DOOR PORTAL SYSTEM USING
REGULATED STRAIGHT MOTION FORMULAS

Sri Winar 1*

1 Universitas Almuslim

* Corresponding author E-mail: sriwinarafifa@gmail.com

1. Introduction

The train station is an important facility. However, the services provided to train passengers are still not optimal. There are still several weaknesses, including the arrival and departure processes of trains that still rely on telephone communication with other station staff. Notification of train arrival will be sent by telephone by the officer at the previous station to the destination station. On receiving notification of arrival, station staff will close the train portal [1]. This process can have a weakness. Namely, if the officer is negligent, it can cause an accident. Besides, for train passengers who will board the train, there is no definite notification regarding the train's arrival [2].

This is due to the absence of an automatic operating system [3] on the railroad crossing portals because the manual system must use human labour or operators to operate and open the railroad crossing portals, so that operator error or failure of manual operation of the portal can increase the likelihood of a train accident occurring at the crossing portal, the above problems can be overcome by implementing an automatic portal closure system. With an automatic portal system, the portal will automatically close if a train passes through the crossing.

2. Method

An object is said to be making regular straight motion if its velocity is always constant. Constant speed means that the velocity, aka velocity's speed and direction, is always constant [6]. Because the velocity, aka velocity, and direction of velocity are always constant, it can be said that the object is moving in a straight path with constant velocity.
Suppose a car is moving straight east at a constant speed of 10 m/s. This means that the car is moving straight east for 10 meters every second. Because the speed is constant, after 2 seconds, the car moves straight east for 20 meters; after 3 seconds, the car moves straight east for 30 meters and so on. Compare with the picture on the side. Pay attention to the size and direction of the arrow. The length of the arrow represents the velocity, aka speed, during the direction. The arrow represents the direction of velocity. Car speed direction = car displacement direction = car motion direction.

Note that when it says velocity, it means instantaneous velocity. Likewise, on the contrary, when it is said to be instantaneous velocity, what is meant is velocity.

When an object is making a straight, regular motion, its velocity is equal to its average velocity. In regular straight motion [7], the object's velocity is always constant. Constant speed means the amount of velocity (velocity = speed), and the direction of velocity is always constant. The velocity or velocity of the object is constant or always the same at any time. Therefore the velocity or velocity must be equal to the average velocity, likewise with objects moving at a constant speed. The object's speed is always constant or always the same, so the average speed is the same. If the object moves at a constant speed of 10 m/s, its average velocity is also 10 m/s. For Regular Straight Motion, the Source formula applies:

- \( s = v \cdot t \)
  - \( s \) = distance reached (change in trajectory)
  - \( v \) = speed
  - \( t \) = time

3. Results and Discussion

3.1. The proposed system

To solve the problem of the current portal closing process, the authors propose a new system [8], namely the portal can close automatically by calculating the train's speed so that the train arrival time can be predicted. Besides, a light sensor is placed to detect a passing train. These light sensors are placed on the left and right of the railroad crossing. If the front train passes the light sensor, the portal will automatically close. And if the rear train passes the light sensor at the rear of the train, the portal will open again.

3.2. System Design

The purpose of the system design is to meet user needs regarding a clear picture of the system design created and implemented. The automatic portal closing simulation system design is divided into two, namely, system design when the train is approaching the crossing and system design. The following is a simulation system design for automatic portal closure when the front train touches the light sensor:
The distance of the sending light sensor from the highway is + 5.5 km, and this distance is taken based on the calculation of the fastest train speed in Indonesia, which is +110 km/hour [9]. The fastest train’s speed is taken as the standard measurement of the distance of the light sensor to the road so that the train does not cross the portal in less than 3 minutes, calculated from the time when the front of the train hits the light sensor.

With the consideration that the railroad portal bars must close at least 3 minutes before the train passes. So by using the GLB formula, the results are obtained:

\[ v = 110 \text{ km/hour} \]
\[ t = 3 \text{ minutes} \]
\[ = 3/60 \text{ hours} \]
\[ = 0.05 \text{ hours} \]
\[ s = \ldots \]

Based on the GLB formula, that \( s = v \cdot t \) is obtained
\[ s = 110 \times 0.05 \]
\[ = 5.5 \text{ km} \]

3.3. Implementation

The method used to predict the arrival time of the train is to use the regular straight motion formula and light sensor simulation, the light sensor used is only an animated simulation in the form of a flash [10]. The main view is the first display that appears when the simulation application for closing the railroad door portal automatically using the formula for regular straight motion is run, the following is the image of the main view of the simulation application of automatic railroad door portal closing:

On the main display page, there are several scripts, including:
- Script button hint

This script serves to redirect the main page to the instructions page. The hint button script is as follows:
Speed (speed) script function

The speed fox script function is to change the speed of the train. The speed change function script is as follows:

```javascript
function rubahSpeed()
{
    vKa1 = parseInt(txtSpeedKa1.text);
    if (vKa1 > batasKecepatanKa1)
    {
        vKa1 = batasKecepatanKa1;
        txtSpeedKa1.text = String(batasKecepatanKa1);
    }
    if (vCar1 == 24)
    {
        vCar1 = 23;
    }
    if (vCar2 == 24)
    {
        vCar2 = 23;
    }
    if (vKa1 == 24)
    {
        vKa1 = 23;
    }
    stopCar1();
    speedCar1 = vCar1;
    startCar1();
    stopCar2();
    speedCar2 = vCar2;
    startCar2();
    stopKa1();
    speedKa1 = vKa1;
    startKa1();
}
```

The latch closing display is a display when the first sensor touches the face of the train so that the sensor automatically closes the latch; here is the display of the latch closing:

![Latch closure display (KA touch sensor 1)](image-url)
The script to detect whether a train has touched the first sensor or not is as follows:

```java
if (kal.hitTestObject(sensor1))
{
    btnRandom.visible = false;
    datang = true;
}
if (datang==true)
{
    txtJarakKal.text = String(jarak.toFixed(2));
    vKal = parseInt(txtSpeedKal.text);
    txtWaktuKal.text = String(t.toFixed(2));
}
else
{
    txtJarakKal.text = ''; 
    txtWaktuKal.text = ''; 
}
if (t<=5.8 && datang==true)
{
    LampuKuning.visible = true;
    LampuMerah.visible = false;
    if ((car1.car.y >= car1Y1 && car1.car.y <= car1Y2 ))
    {
        car1.stop();
        stopCar1();
    }
    if ((car2.car.y >= car2Y2 && car2.car.y <= car2Y1 ))
    {
        car2.stop();
        stopCar2();
    }
}
```

The latch opening view is a display when the second sensor touches the back of the train so that the sensor automatically opens the latch, here is the latch opening view.

![Latch Opening View](image)

**Fig. 6: Door latch opening display (KA touch sensor 2)**

The script to detect whether the train had touched sensor 2 is as follows:
if (kal.ka.sensorKa2.hitTestObject(sensor2))
{
    btnRandom.visible = true;
    LampuKuning.visible = false;
    LampuMerah.visible = false;
    note = true;
    if (((car1.car.y >= car1Y1 && car1.car.y <= car1Y2 ))
    {
        car1.play();
        stopCar1();
        speedCar1 = 0;
        startCar1();
    }
    if ((car2.car.y >= car2Y2 && car2.car.y <= car2Y1 ))
    {
        car2.play();
        stopCar2();
        speedCar2 = 0;
        startCar2();
    }
    tiangKa.visible = false;
    tiangCar.visible = true;
    palangMenutup1.visible = false;
    palangMenutup1.gotoAndStop(1);
    palangMembuka1.visible = true;
    palangMembuka1.play();
    palangMenutup2.visible = false;
    palangMenutup2.gotoAndStop(1);
    palangMembuka2.visible = true;
    palangMembuka2.play();
}
}

The manual page contains a page explaining the use of the simulation application for the automatic closing of a railroad door, along with a description for this application, the following is a display of the instructions page:

Fig. 7: Instructions page

On this manual page, the instructions used to explain the latch closing simulation application are in the form of a video file. The script used to call the video files on this page of instructions is as follows:

```
var myLoader:Loader = new Loader();
var url:URLRequest = new URLRequest("petunjuk_simulasi2.swf");
myLoader.load(url);
addChild(myLoader);
myLoader.x = 50;
myLoader.y = 15;
```
4. Conclusion

From the whole discussion that has been discussed in the previous chapters, the writer can conclude the thesis entitled “Simulation Application of Automatic Train Door Closure Using Regular Straight Motion Formulas” can be concluded as follows:

▪ This simulation is useful for making it easier for Railway Technicians to design an automatic train portal control system.

▪ This simulation reduces the risk of accidents due to not closing the railroad portal when the train crosses the railroad crossing.

In making this application, there are still shortcomings that can then be considered in making further applications. The suggestions that can be conveyed about this application are:

▪ We recommend that this application be combined with GPS technology so that the position of the train can be detected in realtime.

▪ This application should be implemented as a system and can be used and useful for related agencies.

Acknowledgement

This research was held in collaboration with the Research and Community Service Institute at Almuslim University through internal research in 2021. All funding for research implementation to publication in journals comes from Research Institute. Researchers would like to thank the Muslim University Research Institute leaders, the Informatics Engineering Study Program, students involved in the research, and all those who have helped. Hopefully, this research can contribute to technology-based research work.

References

[1] T. S. Yoon, S. Lee, J. H. Hwang, and D. H. Lee, “Prediction and validation on the sonic boom by a high-speed train entering a tunnel,” J. Sound Vib., 2001, doi: 10.1006/jsvi.2000.3482.
[2] D. Dudzik, “Public transportation,” Architectural Digest, 2011, doi: 10.4324/9781351134033-14.
[3] C. Giuffrida, A. Kuijsten, and A. S. Tanenbaum, “Safe and automatic live update for operating systems,” 2013, doi: 10.1145/2451116.2451147.
[4] U. Straumann, C. Roggli, and H. J. Ziegler, “Lotschberg Base Tunnel - Passing under the BLS mountain railway line at Frutigen,” Felsbau, vol. 22, no. 4, 2004.
[5] S. S. Jeon, Y. K. Park, and K. Y. Eum, “Stability assessment of roadbed affected by ground subsidence adjacent to urban railways,” Nat. Hazards Earth Syst. Sci., vol. 18, no. 8, 2018, doi: 10.5194/nhess-18-2261-2018.
[6] N.-S. Cheng, “Simplified Settling Velocity Formula for Sediment Particle,” J. Hydraul. Eng., 1997, doi: 10.1061/(asce)0733-9429(1997)123:2(149).
[7] J. Stigler, “Analytical Velocity Profile,” Eng. Mech., 2014.
[8] R. Hardi, “The Use Of Tsp For The Application Of Pos Indonesia Delivery Service,” VFAST Trans. Softw. Eng., 2015, doi: 10.21015/vtse.v7i1.339.
[9] E. B. Setyawan and N. Novitasari, “Indonesian High-Speed Railway Optimization Planning for Better Decentralized Supply Chain Implementation to Support e-Logistic Last Miles Distribution,” 2019, doi: 10.1088/1742-6596/1381/1/012020.
[10] T. Walter, “Adobe Flash,” 2008.