# Toy Train Safety Control System

Team 54

Members: Zhonghao Wang
Jin Yan
Jianghuai Liu
TA: Yuchen He

**Contents**

## 1. Introduction

### 1.1 Objective

The toy train is a lot of fun for children and it decorates our Christmas tree at home. However, as many users complain, toy trains may be derailed and damaged by running too fast at the bend or hitting an obstacle on track, which creates inconvenience and even money loss for toy train users [1]. Therefore, we want to design a safety control system able to detect the bend and the obstacle on track, and controls the train to slow down to a safe speed when it goes through the bend and stop before it is about to hit an obstacle on track.

We will use the computer vision as our main method to solve this problem. Specifically, we will install a camera at the locomotive of the toy train, and transmit the down sampled images shot by the camera to the computing unit for image processing. The computing unit should recognize patterns such as the curvature of the track and the obstacle amidst the track. To detect the curvature of the track, we can know where the bend on the track is and thereafter slow down the train speed for two reasons: 1) if the speed is too high, the train may be derailed at the bend; 2) Because the bend gives the computer little vision of the track, the train needs more time to stop when it goes through the bend and an obstacle is on the track. Based on the recognition, the computing unit should determine the action of the train (run normally, run slowly, or stop), and send out a command signal to the motor control system. The motor control system is a circuit that receives the command sent from the computer, and changes the duty cycle of the power supplied to the train according to the command to control the train speed.

### 1.2 Background

As we researched on how the derailment is caused and how to avoid the derailment, we found three suggestions given by National Model Railroad Association: 1) the scenery or remodeling is done at the area of derailment; for example, an obstacle is stuck at the track, and the solution to this issue is to check the track again and remove that obstacle; 2) the toy train itself has problems; the solution to this issue is to check the inner elements of the toy train, the engine, the wheel and etc; 3) the switch of the track is not smooth. The solution is to clean the frog (the crossing point of two rails) and keep rails free of debris [2]. However, all these troubleshootings are based on the fact that the train has already derailed, people have to check the system and environment conditions of the train. After people fixed the problem, the train and the environment cannot be changed, or the train might derail again. Our concern is that the train cannot automatically respond to the environment and prevent itself from derailing. For example, supposing that the train runs normally on the track but suddenly a gift box hanging in the Christmas tree falling onto the track, if the train cannot automatically respond to the environment changes, it may directly hit the gift box on track and cause damages to itself.

Therefore, we plan to build a system which can detect the potential hazards in the environment as mentioned above. Our terminal goal is that the industry can add our safety control

system to the model train production, so that the quality of the model train can be efficiently increased and consumers will be more satisfied with their consumption.

### 1.3 High-level Requirements

- The image processing algorithm can recognize the obstacle on track and the bend of the track.
- The motor control system can control the train speed according to commands sent from the computer. It is able to stop the train before the train is about to hit an obstacle amidst the track, and decelerate the train to half of its normal speed at most (≤ 5 cm/s) before the train goes through the bend of the track. The train must stop within 5 seconds if the train is about to hit the obstacle in 5 seconds with its normal speed (10 ± 3 cm/s).
- The wifi and transmitting module can transmit the command data to the motor control system.

## 2. Design

### 2.1 Block Diagram

The toy train safety control system consists of four Modules: power module, motor control module, computing module, and camera and transmitting module.

The Power Module supplies power to both the power control module and the wifi and transmitting module. A 120V AC power will be converted to 12V DC power to supply power to the motor driving circuit by an AC-DC converter. The 12V DC power converted will be further regulated to a 5V DC power to power the duty cycle controller by a DC voltage regulator. We will charge a Li-Ion battery with a Li-Ion charger, and the battery will be installed on the train to power the wifi transmitter, the microprocessor and the camera. The reason we use a battery will be explained in the functional review part.

The motor control module receives the command from the computing unit; according to the commands received, it controls the duty cycle of the voltage supplied to the DC motor. The wifi receiver will receive a command sent from the computer. There are three kinds of commands representing three states the train runs with: 1) the train runs in normal speed (10 ± 3 cm/s); 2) the train slows down to a speed below half of its normal speed (≤ 5 cm/s); 3) the train stops. The duty cycle controller will translate this command into the corresponding duty cycle which should be imposed on the motor; this is realized by a finite state machine. The motor driving will power the DC motor with a voltage having the duty cycle indicated by the motor control circuit.

The computing unit will process the image sent by the camera and wireless transmitting module. It will determine whether the train is about to hit the obstacle on track, go through the bend, or the train runs normally. Based on the judgement, the computer will send a signal corresponding to the running condition of the train to the motor control module.

The camera and transmitting module has a Li-Ion battery as its power supply. Because this module is attached to the train, it will run together with the train on the track. The camera shoots the picture in the front of the train, and transmits the image data to the microprocessor. The microprocessor down samples the image, and transmits the image data to the computing module through the wifi transmitter.

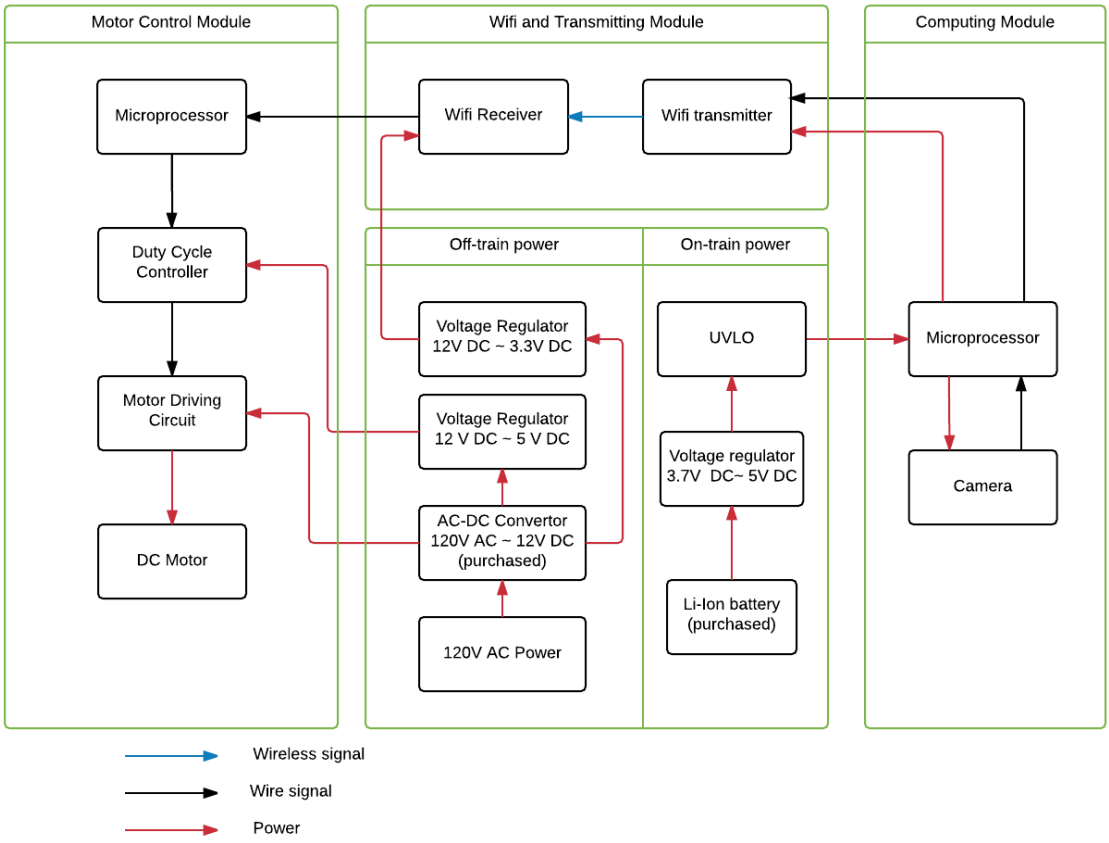


Figure 1. Block Diagram of Toy Train Safety Control System

## 2.2 Functional Overview

### 2.2.1 Power Module

We will buy an AC-DC converter to convert the 120VAC power to 12 VDC power. If we try to build an AC-DC converter to convert 120 VAC power, we may build an unqualified converter and thus cause harmful results like overheating and even burning the converter. The output of this converter will supply 12 VDC power to the duty cycle controller from the motor control module, and will be regulated to 5 VDC to supply power for the motor control circuit. For the same reason that we will not build an AC-DC converter by ourselves, we will buy an adapted Li-Ion charger to charge the Li-Ion battery. After the Li-Ion battery is charged, we will install that battery on the train to supply power to the processor on the train. Because the microprocessor's rated voltage is 5 volts, we will design a voltage regulating circuit and put it on the train to convert the voltage from

3.7 V to 5V. It's worth noticing that the power to the motor is supplied by the track, and we use this power supply to control the train speed, so it is not a constant voltage. Therefore, the wifi and transmitting module which moves along with the train has to have its own independent power supply system. That is the reason why we use a Li-Ion battery.

2.2.1.1 On-train Power Supply

Since all the on-train hardwares require a relatively low voltage to work, we would use rechargeable Li-ion batteries to supply power. To be more specific, we plan to use 18650 3.7V 5000mAh Rechargeable Li-ion Batteries and a voltage regulator to stabilize the output voltage to 3.3±0.1V and 5±0.1V.

| Requirement | Verification |
|---|---|
| Capable of providing stable 5±0.1V and 3.3±0.1V | 1. Setup the power supply module<br>2. Use an oscilloscope to measure the output voltage<br>3. Plot the output voltage and measure the ripple of voltage |

Table 1. Requirement and verification for on-train power supply

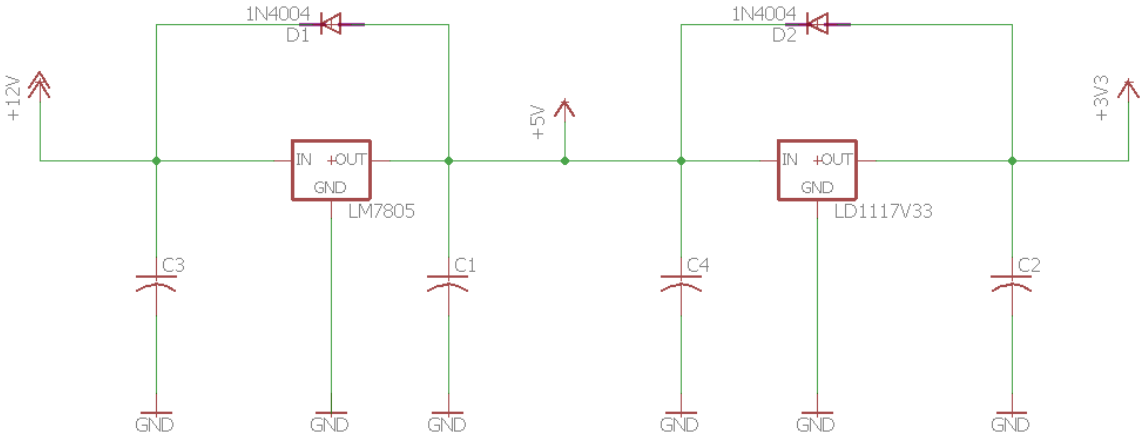


Figure 2: Voltage regulator

### 2.2.2 Computing Module

This module is divided into the hardware side and the software side. On the hardware side of the computing module, there are two parts: camera and microprocessor. Camera is responsible for input images and is directly plugged in the microprocessor. The microprocessor is responsible for down sampling the input images from 14~16 fps to 2~5 fps. The computing module would output a control signal to the transmission module. The software side will implement an algorithm to process images.

#### 2.2.2.1 Microprocessor

A microprocessor is placed on the train connected to the camera described above and takes input images and down samples those images to a lower rate. We plan to use Raspberry Pi 3 [3] purchased at the machine shop. This microprocessor is responsible for communicating with camera and wireless module, processing input images and generating a control signal to the transmission module. The microprocessor can down sample the images by only processing the first image among an amount of images taken by the camera.

| Requirement | Verification |
|---|---|
| The buffer should be able to handle 1 image for every 0.5 seconds | 1. Take pictures for three minutes at different camera fps configuration setup<br>2. Record the free space of buffer through the time<br>3. If there are always free space in the buffer, the requirement is verified |
| Clear up memory used for storing historical information every 5 seconds | 1. Program the microprocessor to clear up its memory every 5 seconds<br>2. Setup camera frame rate to be above the maximum image number that the algorithm can process within 0.5 seconds<br>3. Record the usage of the memory |

Table 2. Requirement and verification for microprocessor

#### 2.2.2.2 Camera

This part will be purchased online and connected to our circuit on the train to take the pictures of the track. In our design, we plan to use Raspberry Pi Camera module V2 powered by the microprocessor connected to it.

| Requirement | Verification |
|---|---|

| | |
|---|---|
| Maintain at least 15 fps | 1. Take pictures for three minutes at different camera frame rate setups<br>2. Check the total images taken<br>3. Divide the total images taken by the duration of time |
| Capable of taking images that are qualified for the image processing algorithm to process above 300 Standard Maintained illuminance(lux) | 1. Setup environment with different lighting condition using a HDE LX-1010B Digital Luxmeter Light Meter ranging from 0 to 3000<br>2. Let the camera take 100 images under every lighting condition<br>3. Let the computing module process the images and calculate the accuracy of recognizing obstacles<br>4. If the program can recognize obstacles above 80% when lux is greater than 300, this requirement is verified |

Table 3. Requirement and verification for camera

Figure 3. Raspberry Pi Camera module V2 connection with Raspberry Pi[10]

2.2.2.3 Image Processing Algorithm

The algorithm inputs images sent from the camera. To detect whether there is a potential hazard in front of the train, the algorithm (raspberry pi 3 model B) will use the segmentation process to recognize the wanted patterns: the curvature of the track and the obstacle. The specific algorithm is introduced in the flowcharts as figure 4 and figure 5 below. Generally, the image shot from the camera at the locomotive should be divided into two regions: the region between the

track, and the background. As the assumption we made, there will be some color contrasts between the track and its background. Then we can threshold the grey level and divide the pixels into two groups: the track and its background. Now, let's consider two situations: 1) there is no obstacle on the track; 2) an obstacle is stuck at the track.

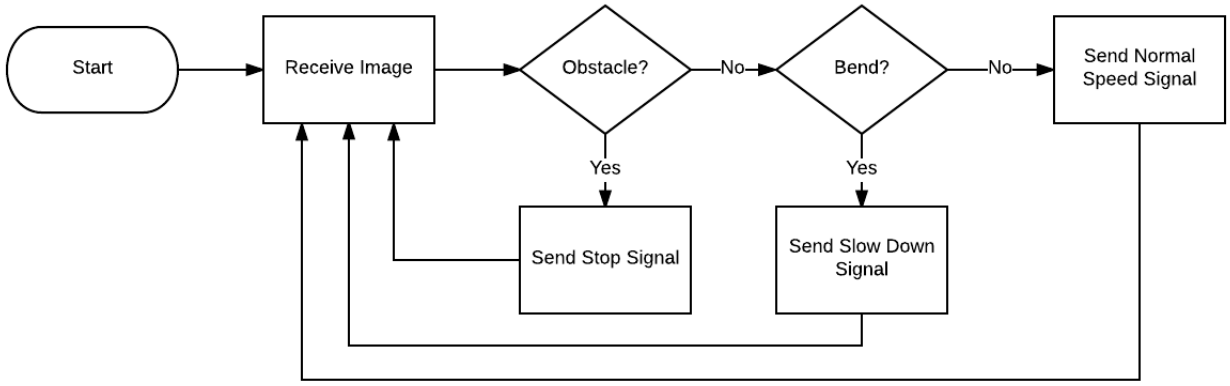


Figure 4. Flowchart of Central Control

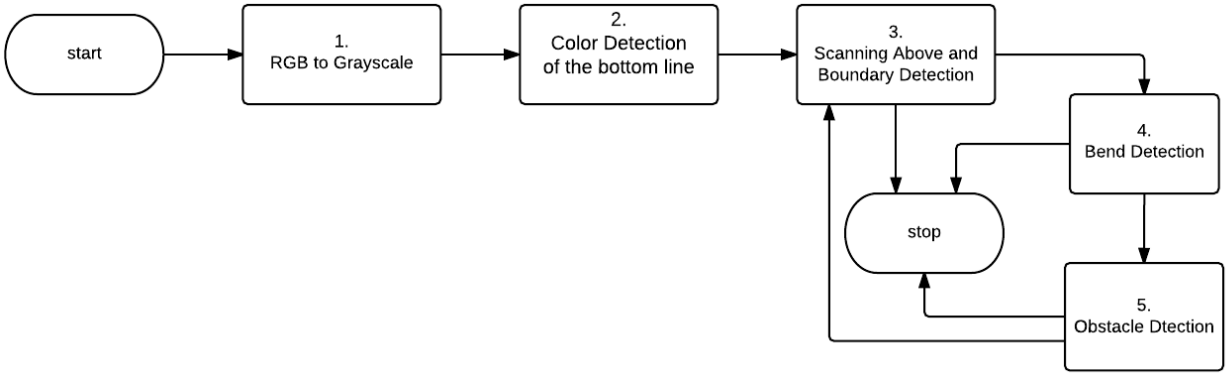


Figure 5. Flowchart of Image Processing

For the first situation, because the track is a consistent line without much of color changing, we can utilize this feature to recognize the track. The image we are going to use is 640x480 pixels.

First, for each pixel, we will convert its RGB value into grayscale by the formula

$$Grayscale = 0.2989R + 0.5870G + 0.1140B. \qquad \text{Eq 1}$$

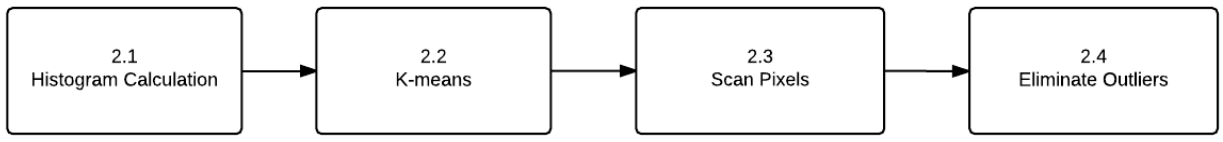


Figure 6. Flowchart of Color Detection

Second, from the bottom line of pixels in the image (figure 7), find the histogram of grayscales of that line. As we assumed, the background is mainly light colored, so the color of the dark track will concentrate on a certain spectrum of the histogram.

Figure 7. Track image for Building Histogram

Third, we will use K means to find the means of two classes ($M_1$ and $M_2$, and $M_1 < M_2$). The smaller mean grayscale ($M_1$) represents the class containing the pixels from the track area, because the track is of dark color which has small grayscale value. The reason we use K-means to divide the classes of pixels is that we want the algorithm able to detect the track color threshold under different lighting conditions. The flowchart of the K means algorithm is shown in figure 28.

Forth, scan the pixel from the left to the right of the bottom line; if the grayscale of a pixel is closer to $M_1$, store the x-coordinate of that pixel to `array[]`.

Fifth, use the quantile measurement to eliminate the outliers in `array[]` made by the fourth step. We can use the following formula to decide the outliers [4].

$$x < Q1 - 1.5(Q3 - Q1) \text{ or } x > Q3 + 1.5(Q3 - Q1) \qquad \text{Eq 2}$$

$x$ is the x-coordinate of an outlier.

$Q1$ Is the first quantile number of the data in `array[]`.

$Q3$ Is the third quantile number of the data in `array[]`.

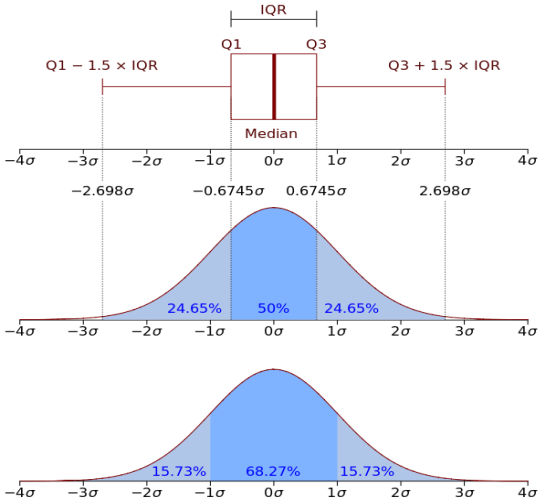


Figure 8. Quantile Measurement [4]

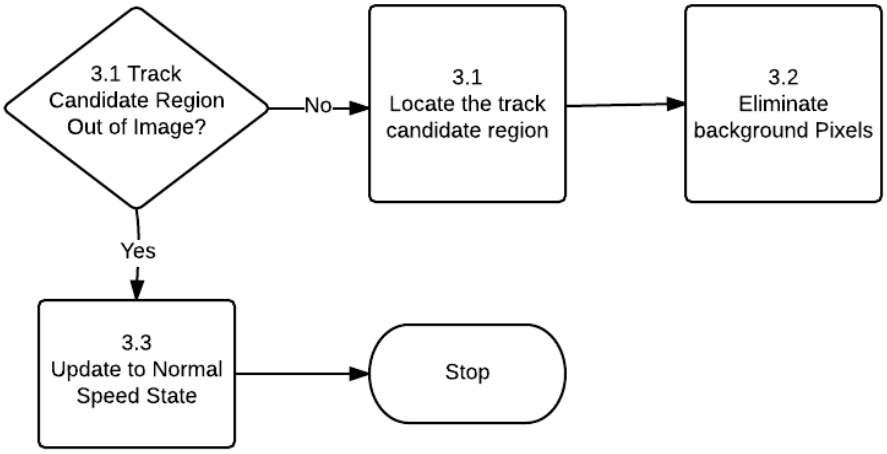


Figure 9. Flow chart of Scanning Above and Boundary Detection

Now, we get the x-coordinates of the track pixels in the bottom line. Let's say the range of these pixels' x-coordinates is $[x1, x2]$.

Sixth, in the line above the previous line (as circled by the blue rectangle in figure 10), find the pixels in x-coordinate range

$$[x1 - c(x2 - x1), x2 + c(x2 - x1)]$$

$c$ is a constant.

that have grayscales closer to $M_1$ than $M_2$.

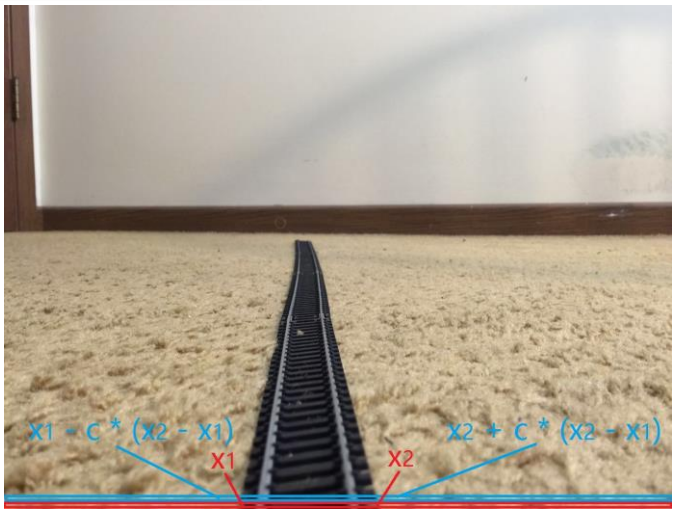


Figure 10. Track Image scanning the above line

The x-coordinates of these pixels represents the track location in the line circled by the blue rectangle. The reason why we have the term $c(x2 - x1)$ is that the track may be bend, and thus we need greater range to detect the track location.

Seventh, calculate the new mean values of the track grayscale and the background of this line, and store them to $M_1$ and $M_2$ separately.

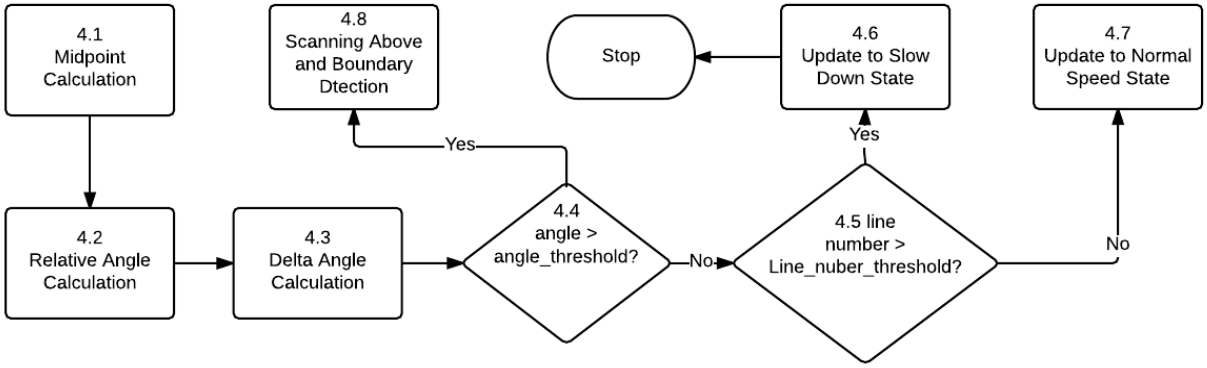


Figure 11. Flowchart of Bend Detection

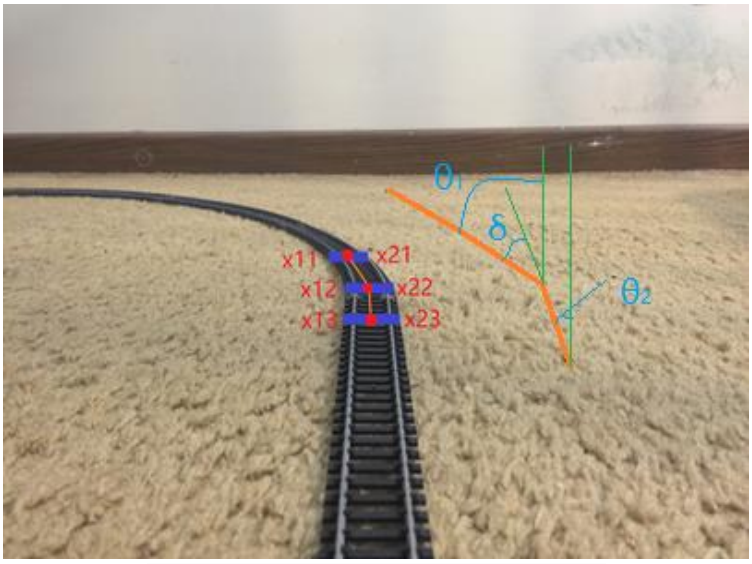


Figure 12. Track Image Detecting the Bend

Eighth, repeat sixth and seventh steps. If the recognized track area reaches the boundary of

$$[x1 - c(x2 - x1), x2 + c(x2 - x1)]$$

or reaches the left or up or right edge of the image, stop repeating the sixth and seventh step. To decide whether the track is bent, we can compare the difference of the midpoint x-coordinates of the track in different lines. For example, in figure 12, we can first compute the midpoints coordinates of the track in the three lines by

$$x_{1mid} = (x_{11} + x_{21})/2$$
$$x_{2mid} = (x_{12} + x_{22})/2$$
$$x_{3mid} = (x_{13} + x_{23})/2 \qquad \text{Eq 3}$$

$x_{11}, x_{12}, x_{13}$ are the left x-coordinates of the track in each line, respectively;
$x_{21}, x_{22}, x_{23}$ are the right x-coordinates of the track in each line, respectively.

If

$$(x_{1mid} - x_{2mid}) - (x_{2mid} - x_{3mid}) > threshold \qquad \text{Eq 4}$$

$threshold$ is a number defined in program.

the algorithm will trigger a signal saying that the bend is detected, and send this signal to the motor control module.

For the second situation, because the obstacle has an obvious color contrast with the track, we can utilize this feature to detect the obstacle on track.

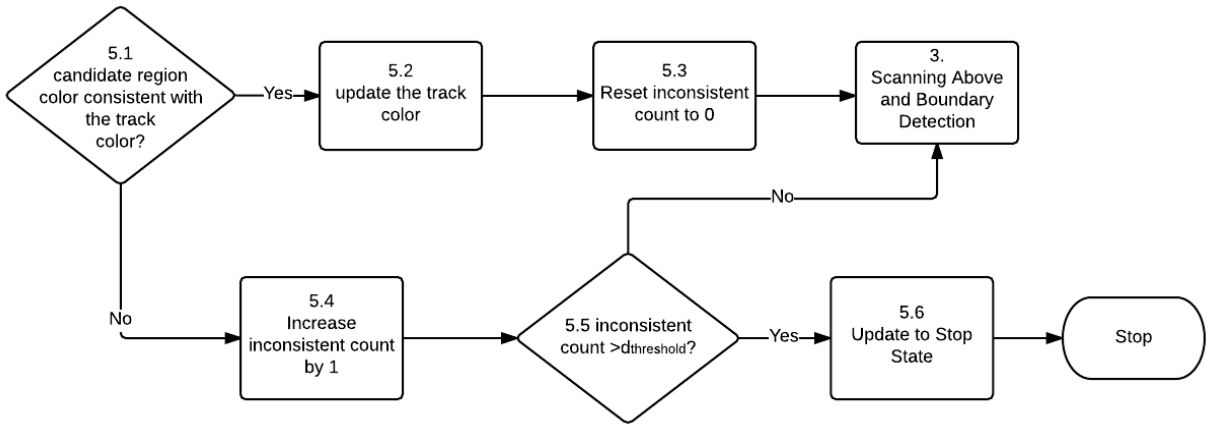


Figure 13. Flowchart of Obstacle Detection

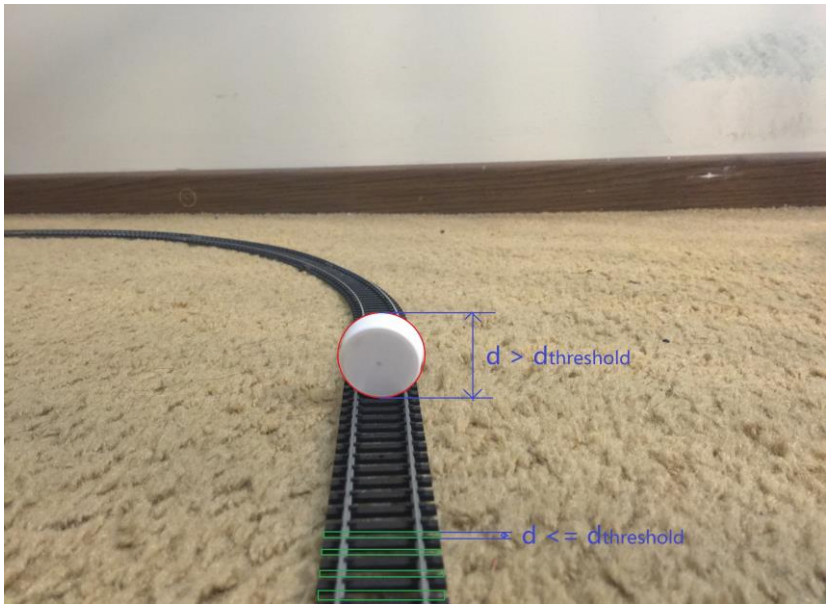


Figure 14. Track Image Detecting the Obstacle

First, we have to deal with some exceptions. Specifically, in figure 14, the area circled by the green rectangle is the gap between the railway sleepers, of which the color is very similar to the background color. Therefore, when we scan from the bottom to the top of the picture, if $M_1$ > 100, store $M_1$ into `gap[]`; if $M_2$ <= 100, store $M_1$ into `sleepers[]` (we assume the grayscale of the sleepers must be less than 100).

Second, decide whether the color contrast is caused by the obstacle or by the gap of sleepers. We set a threshold $d_{threshold}$ for the elements number in `gap[]`. The elements must be consistently filled in `gap[]`. That means, if $M_1$'s of consistent several lines are above 100, all these

$M_1$'s have to be put into `gap[]`. If $M_1$ is less than 100, the algorithm should first empty `gap[]`, and then add this $M_1$ to `sleepers[]`. If the number of elements in `gap[]` is greater than $d_{threshold}$, the algorithm will trigger a signal saying that an obstacle is detected, and send this signal to the motor control module.

Also, we have to add another restrict to the sixth step of the first situation. Instead of using $M_1$ calculated from the last line to determine the sleepers area, we should always use the last element added to `sleepers[]` to determine the sleepers area. Because the newly calculated $M_1$ may be less than 100, it is inaccurate to decide the sleepers area with that as the grayscale of sleepers is always below 100.

For specific implementations of states in the flowchart, please refer to table 1 in the appendix.

Assumptions we make:

1) The lighting condition is consistently above 100 lux of luminance. That is, the luminance stays at a relatively fixed level above 100 lux.

2) The obstacle qualified for detection is at least as large as a cube having 2 cm of its side, and has at least 150 units of grayscale.

3) The bend qualified for detection is at least 2 $m^{-1}$ of its curvature.

4) The grayscale of the track is below 100 units. The grayscale of the background is above 150 units.

5) The obstacle is initially put at least 50 cm away from the locomotive for detection.

6) The bottom part has ground truth of track color.

| Requirement | Verification |
|---|---|
| For steps 2 in the flow chart, the algorithm is able to divide the pixels in bottom line of the image, which includes the sleeper, into the class of the track and the class of the background. | 1. Input an image of the track as figure to the program. Run the program.<br>2. The program should output an array (array 1) of the grayscale value of the bottom line pixels, and an array (array 2) of grayscale value of its recognized track pixels.<br>3. Compare the two arrays, array 2 should be a subset of array 1.<br>4. Either the head or the end element of array 2 should at least has a difference of 50 units when compared to its adjacent element in array 1 but not in array 2. |
| For steps 3, the algorithm can detect the obstacle on track at least as large as a cube having 2 cm of its side, having at least 150 units of grayscale and placed 10 cm to 70 cm away | 1. Place a cube onto the track. The cube has at least 2 cm of its side, at least 150 units of grayscales (probably white), and is placed at least 10 cm to 70 cm away from |

| | |
|---|---|
| from the locomotive. This is finished by step 6 in the flowchart. | the locomotive. Use a camera on the locomotive to take a picture of its front.<br>2. Input this image to the program and run the program.<br>3. The program should output a string “0000” to indicate the train should go to stop state. |
| For steps 4, the algorithm can detect the bend of the track at least 2 $m^{-1}$ of its curvature and having 10 cm to 50 cm away from the locomotive. | 1. Set a bent track having at least 2 $m^{-1}$ of its curvature at a place 10 cm to 50 cm away from the locomotive. Use a camera on the locomotive to take a picture of its front.<br>2. Input this image to the program and run the program.<br>3. The program should output a string “0101” to indicate the train should go to stop state. |
| The algorithm can run under sufficient luminance condition. | Under a lighting condition which has at least 100 lux of luminance (this is satisfied by the lighting condition of a classroom), the program can finish the first verification. |
| The algorithm can finish processing an image at most in 0.4 s. | 1. Include a time library in the program. Measure the time taken by processing each image of the program using the time() function.<br>2. Input an image of the vision in front of the locomotive to the program and run.<br>3. The program should output a variable saying the time taken to finish processing one image. |

Table 4. Requirements and Verifications of Image Processing Algorithm

### 2.2.3 Transmitting module

This module consists of two wifi chips that can communicate with each other. The wifi chip on the train takes input from the microprocessor and transmit it to the wireless receiver connected to the motor control module. We decide to use ESP8266 wifi chip in our projects for a few reasons. This chip is cheap enough comparing to other wifi chips and can still fulfill our requirements for data transmission rate at the same time.

#### 2.2.3.1 Wireless transmitter and receiver

The wireless sender is a purchased chip connected to the Raspberry Pi on the train to receive commands from our computer vision algorithm and transmit data to the wireless receiver

connected to the microprocessor. Since the wireless transmitter will send out commands to the motor control module and we change the duty cycle to control the speed of the train, the command would contain information about duty cycle.

The wireless receiver would then receive the commands and output it as digital signals and send it to the motor control module.

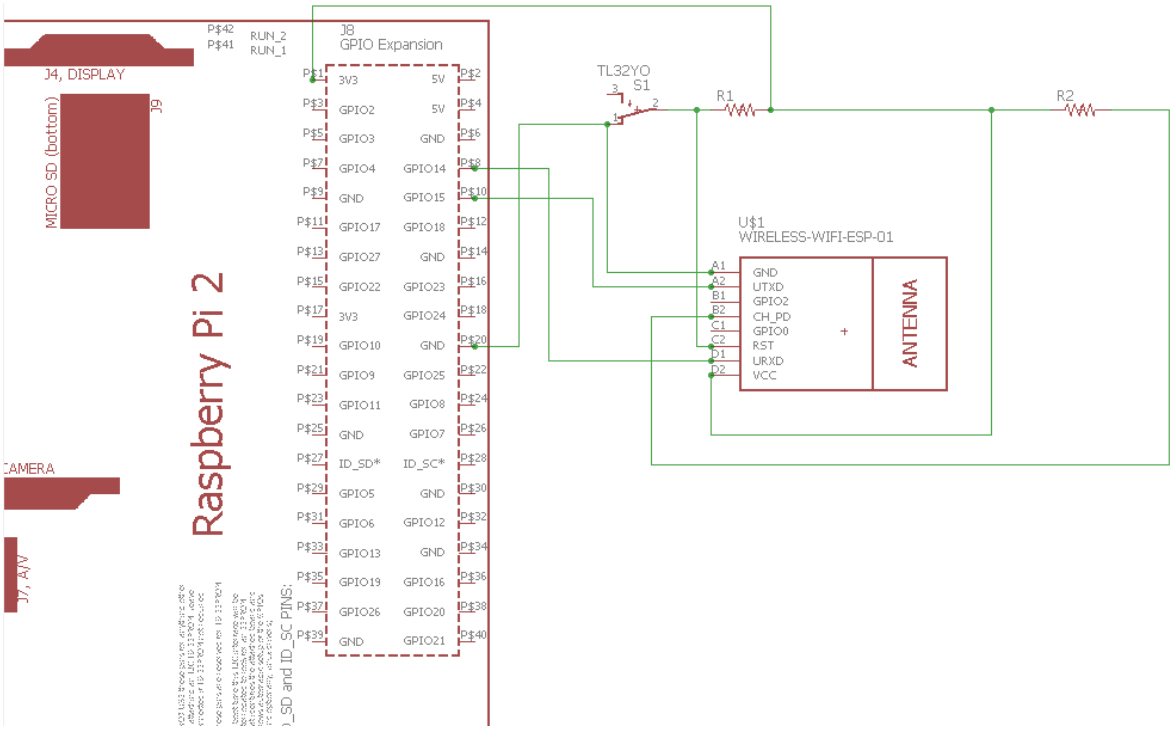


Figure 15. Connect ESP8266 to Raspberry Pi

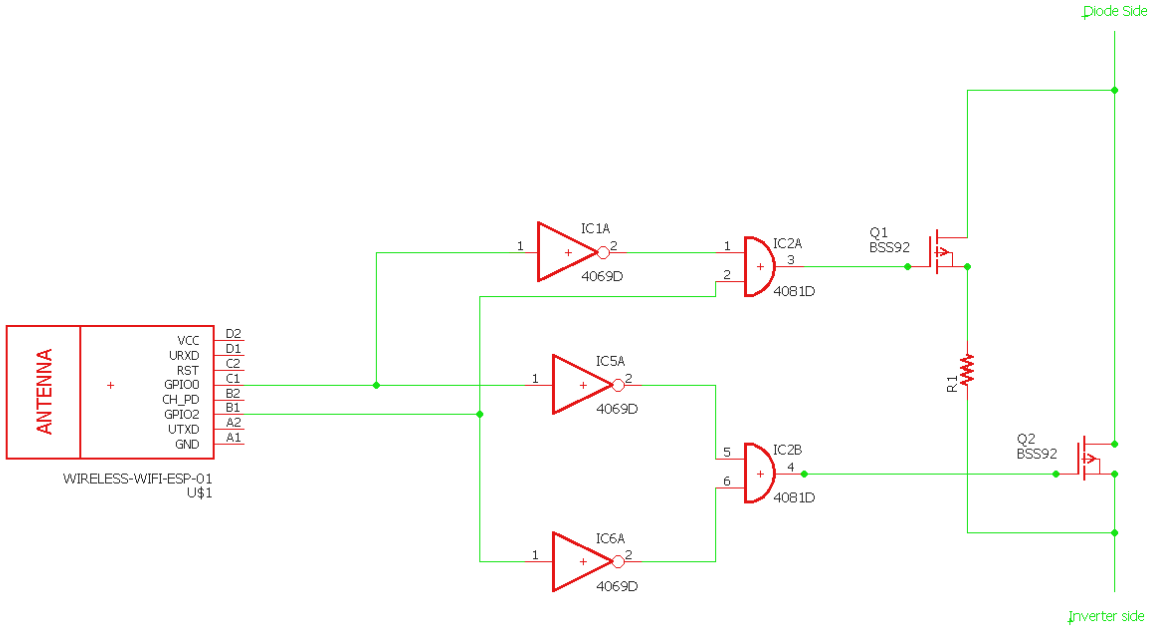


Figure16. Off-train wifi chip connection

| GPIO0 | GPIO1 | MODE |
|---|---|---|
| Low | Low | Stop |
| Low | High | Slow down |
| High | Low(High) | Normal |

Table 5. State Signal and Corresponding State

| Requirement | Verification |
|---|---|
| Capable of transmitting data above 0.1KB/s | 1. Setup the ESP8266 chip<br>2. Send out 10KB commands and record the timestamp for sending and receiving<br>3. Measure the speed of data transfer rate |
| Capable of transmitting and receiving data within 5 meter omni-directional | 1. Setup the ESP8266 chip<br>2. Change location of the sender chip within 5 meter omni-directional for 10 times<br>3. Send out commands for 3 minutes at each location<br>3. Measure how many commands are received by the receiver |

Table 6. Requirements and Verifications of wifi transmission

### 2.2.4 Motor Control Module

The motor control module utilizes PWM (pulse-width modulation) method for motor speed control. Basically it controls the duty cycle of a square wave (with the high voltage corresponding to the turn-on voltage of a certain BJT transistor discussed later, and the low voltage at around 0~0.5V), therefore managing the time interval in which the motor is driven by the power supply, regulating the motor speed. More details about circuit component selection and circuit layout will be introduced later.

#### 2.2.4.1 Duty Cycle Controller

The purpose of the duty cycle controller is controlling how long the transistor in the motor driving part is turned-on within each voltage cycle. The duty cycle controller circuit is shown below.

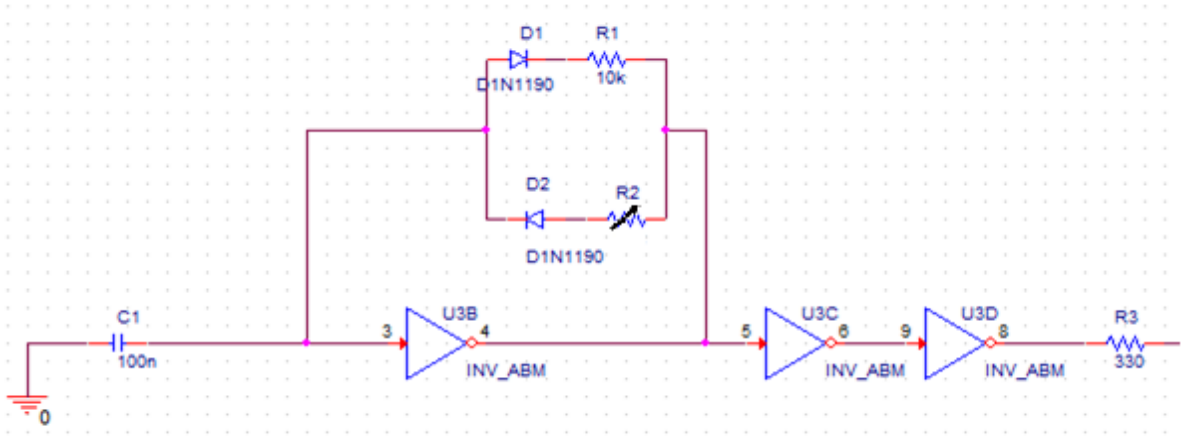


Figure 17. Duty Cycle Control Circuit using PMW

For inverters, we will use HEF40106B chip, which needs to be driven by 5±0.1V DC voltage. We simulated the input-output behavior of the inverter in PSPICE, as attached below:

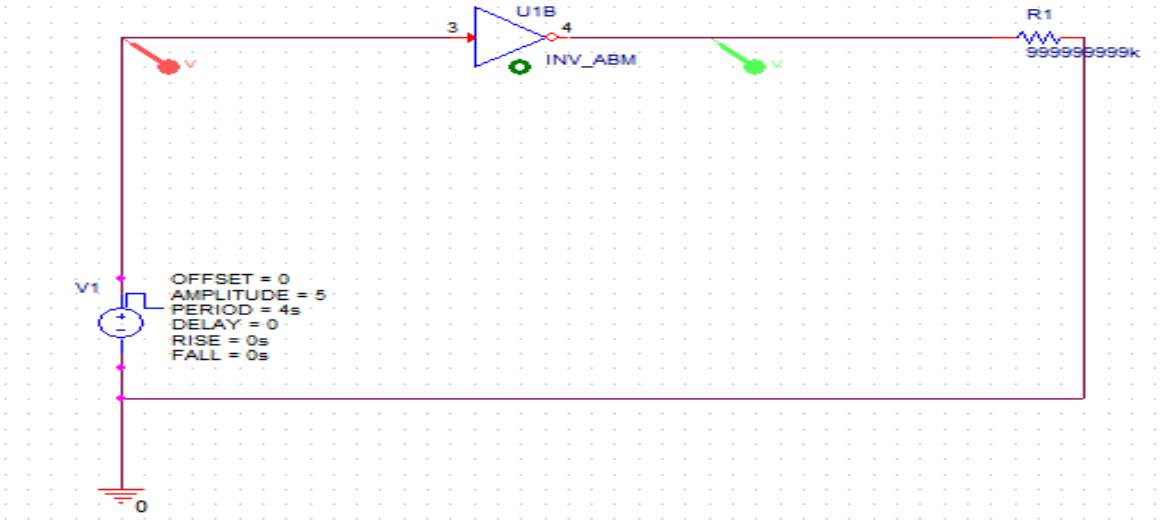


Figure 18. Inverter Simulation Circuit in PSPICE

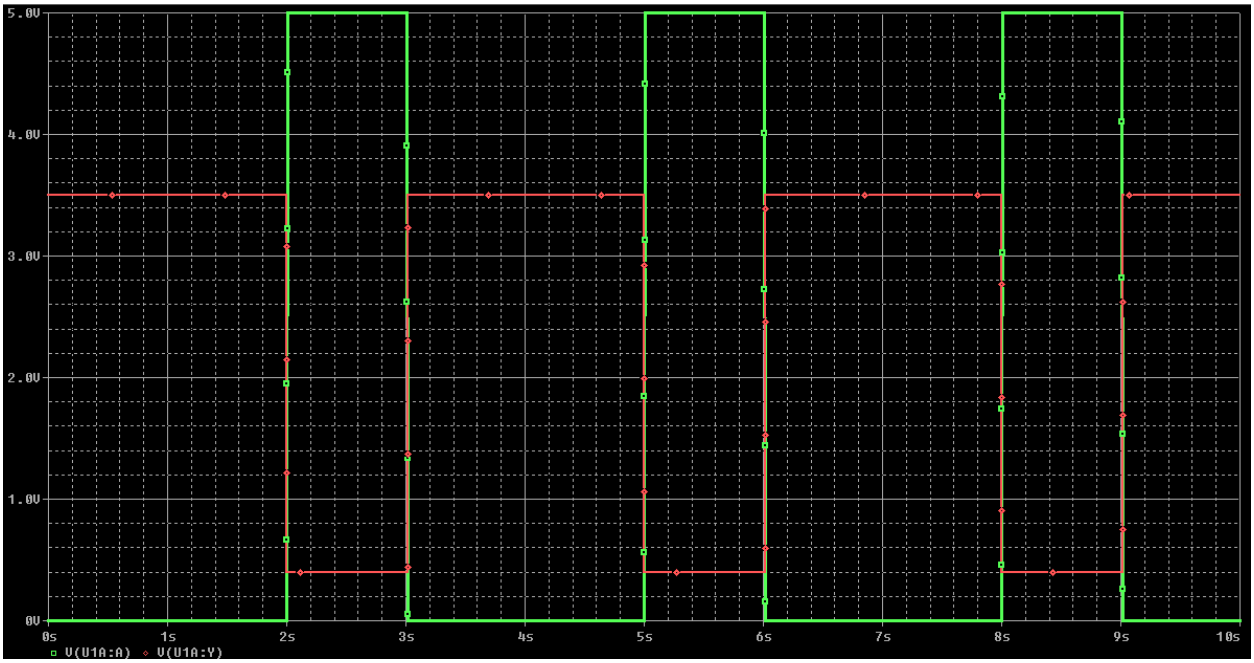


Figure 19. Simulation Result of Inverter Input Voltage (Green line) and Output Voltage (Red line)

In the duty cycle control circuit as shown in Figure 17, R1 is a resistor with fixed value, and the value of R1 determines the discharging time of the capacitor. R2 is a resistor whose value can be determined by the input signal from computer based on various situations, and the value of R2 determines the charging time of the capacitor. In the proposed duty cycle control circuit above, the duty cycle is determined by the time ratio between the charging time and the sum of charging time and discharging time of the capacitor. In this situation, the charging time and discharging time can be evaluated based on the values of R1 and R2 and capacitance C as the formula below.

Based on the I-V correlation of the capacitor:

$$I_C(t) = C\frac{dV_C(t)}{dt} \qquad \text{Eq 5}$$

We can evaluate the duty cycle to be:

$$t_{Charing} = R_2C \times log_e(\frac{V_{out}-V_{low}}{V_{out}-V_{high}}) \qquad \text{Eq 6}$$

$$t_{Discharging} = R_1C \times log_e(\frac{V_{high}}{V_{low}}) \qquad \text{Eq 7}$$

$$\text{Duty Cycle} = \frac{t_{Charging}}{t_{Charging}+t_{Discharing}} \qquad \text{Eq 8}$$

$$\text{Duty Cycle} = \frac{R_2C\times log_e(\frac{V_{out}-V_{low}}{V_{out}-V_{high}})}{R_2C\times log_e\left(\frac{V_{out}-V_{low}}{V_{out}-V_{high}}\right)+ R_1C\times log_e(\frac{V_{high}}{V_{low}})} \qquad \text{Eq 9}$$

Where $V_{low}$ and $V_{high}$ are certain threshold voltages of inverter chip. In most cases we have the following relation to be valid:

$$V_{out} - V_{low} \approx V_{high} \qquad \text{Eq 10}$$
$$V_{out} - V_{high} \approx V_{low} \qquad \text{Eq 11}$$

So as a result, plugging equation (6) and (7) into equation (5), the duty cycle can roughly be estimated as:

$$\text{Duty Cycle} \approx \frac{R_2}{R_1+R_2} \qquad \text{Eq 12}$$

Therefore, we need to carefully control the value of R2 through the computer command to properly control the duty cycle that allows the motor to be driven. Figures 20~22 below show the simulated output voltage (pulse wave) with certain duty cycle:

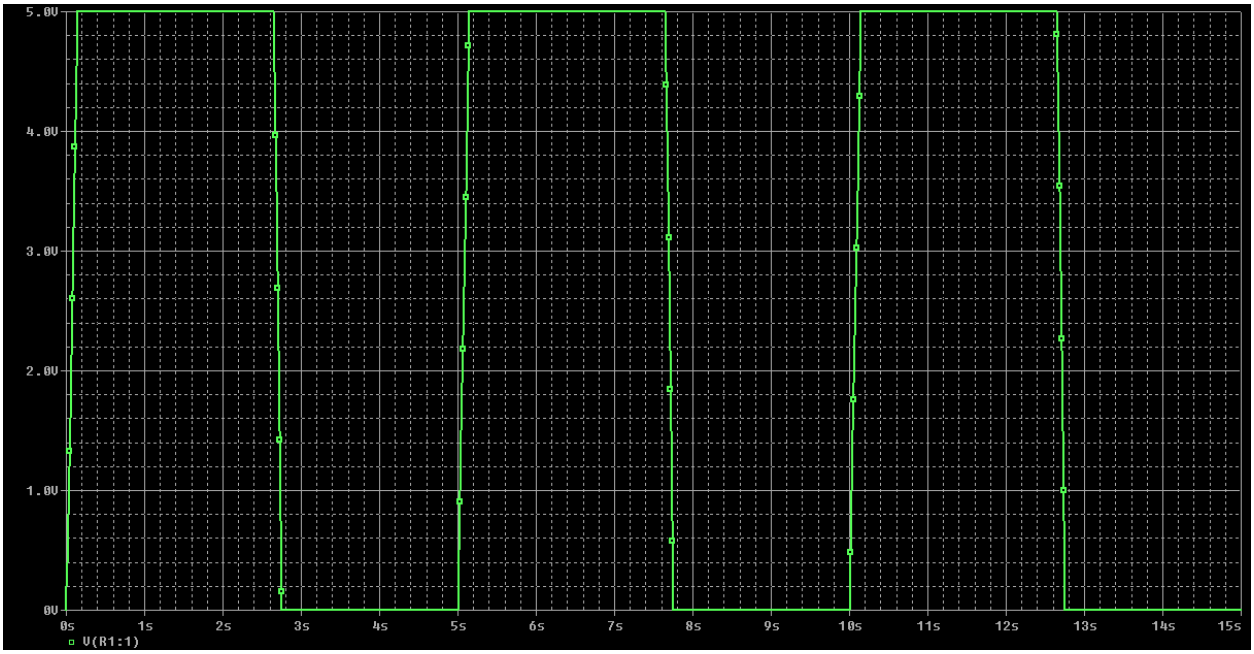


Figure 20. Output Pulse Wave with a 50% Duty Cycle (R1=R2)

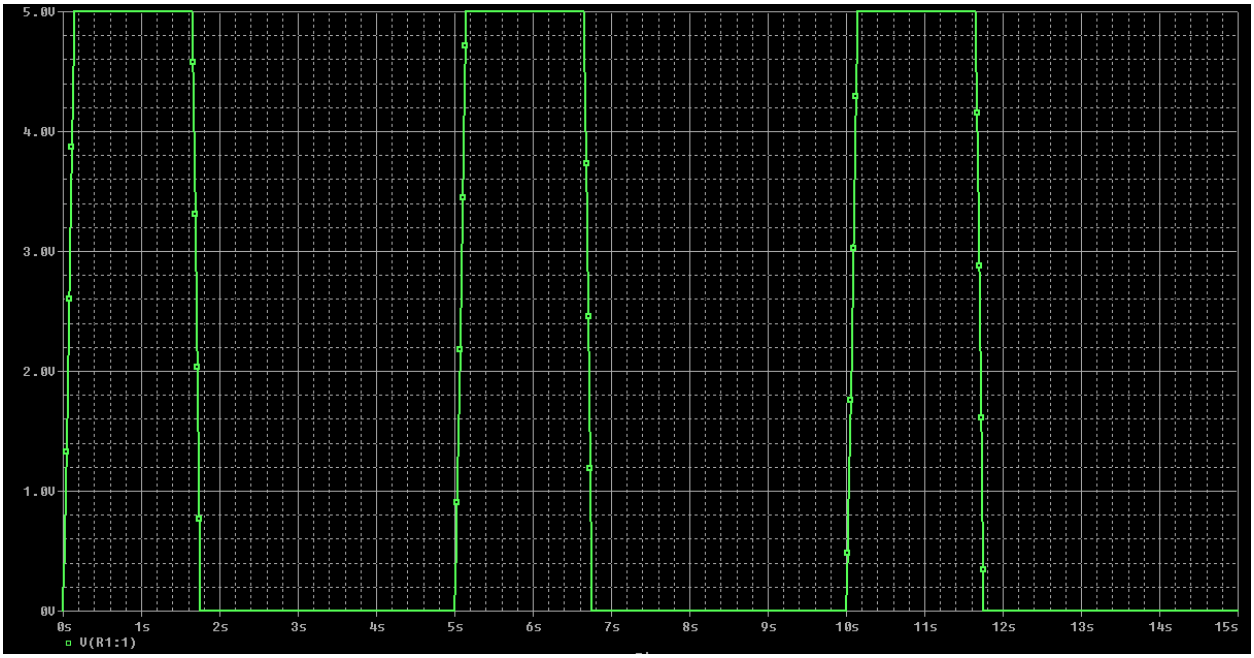


Figure 21. Output Pulse Wave with a 30% Duty Cycle (R2 / (R1 + R2) = 30%)

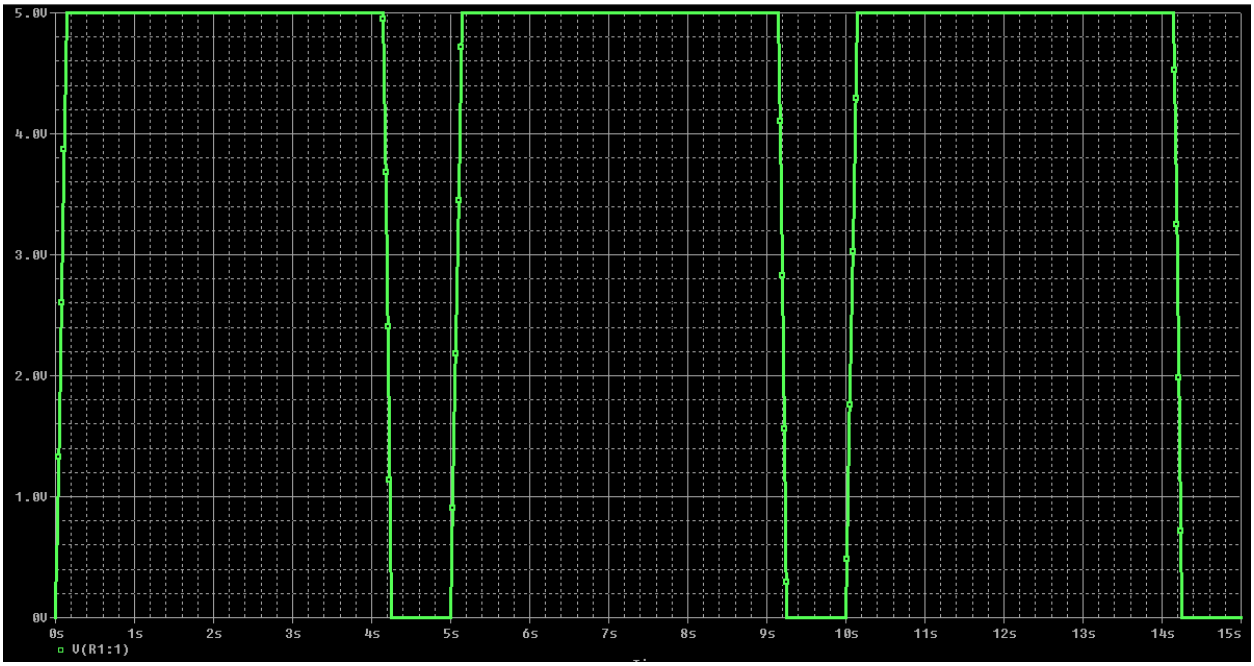


Figure 22. Output Pulse Wave with a 80% Duty Cycle (R2 / (R1+R2) = 80%)

| Requirement | Verification |
|---|---|
| The value of the command-controlled resistor (represented as R2 in Figure 17) should be | 1. Monitor the value of command-controlled resistor R2 under |

| | |
|---|---|
| able to respond to command signal from computer sensitively. | oscilloscope during test.<br>2. If we measured the responding time of R2 for a change in command signal is within 0.01s, this requirement is satisfied. |
| For the high voltage of produced square wave with certain duty cycle, it may not be perfectly constant DC value all the time during the duty cycle. However, it can fluctuate within at most 5% of the expected voltage (5±0.1V). | 1. Monitor the output voltage pulse of the PMW control circuit precisely using oscillator during the test.<br>2. Analyze the tiny ripples reside in the high output voltage of PMW control circuit using oscilloscope.<br>3. Check whether the largest ripple exceeds 5% of expected high voltage (which is around 5V). |

Table 6. Requirement and Verification of Duty Cycle Control Module

2.2.4.2 Motor Driving Circuit

Based on the duty cycle controller introduced above, we come up with the motor driving circuit as shown below, which connects the output port of the duty cycle controller and the motor with a transistor, which functions as a switch controlled by voltage:

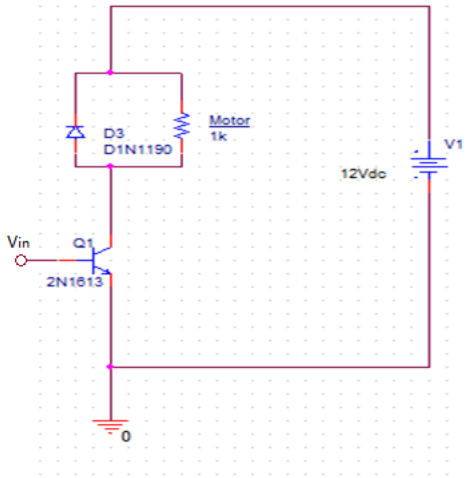


Figure 23. Motor Driving Circuit in PSPICE

Based on the operation region of BJT transistor, when the transistor is turned-on (on duty time range), the collector current is allowed to go into emitter, therefore allowing the motor to be driven by power supply. While the transistor is off (not in duty time range), there is an open circuit between collector and emitter, implying that no current flows through the motor and the motor does not operate. Note that the power supply driving the motor should always be the rated voltage of the motor (possible to be below that, if the voltage across collector and emitter is not negligible when the transistor is turned-on). Further bench test about how sensitive the motor driving circuit is according to the duty cycle is needed.

| Requirement | Verification |
|---|---|
| When the transistor is in off-region, there should be zero collector current or very small value of collector current flowing through the motor. | 1. We connect the base of BJT to ground.<br>2. Measure the current flowing through the collector of BJT (the same current flowing through the motor).<br>3. If the measured collector current is 0~0.1mA, this requirement is satisfied. |
| When the transistor is turned-on, the voltage across the collector and emitter should be very small (close to 0). | 1. Apply a stable DC voltage 5±0.1V to the base of BJT.<br>2. Measure the voltage across collector and emitter.<br>3. If the measured voltage drop between the collector and emitter is 0~0.1V, this requirement is satisfied. |

Table 7. Requirement and Verification of Motor Driving Circuit

2.2.4.3 DC Motor

The DC motor directly connects to the wheel, taking in its rated voltage to run (12±0.1V). It does not support running in the backward direction, and it is not supposed to take in negative voltage to operate. The whole motor control circuit is shown as below [11]:

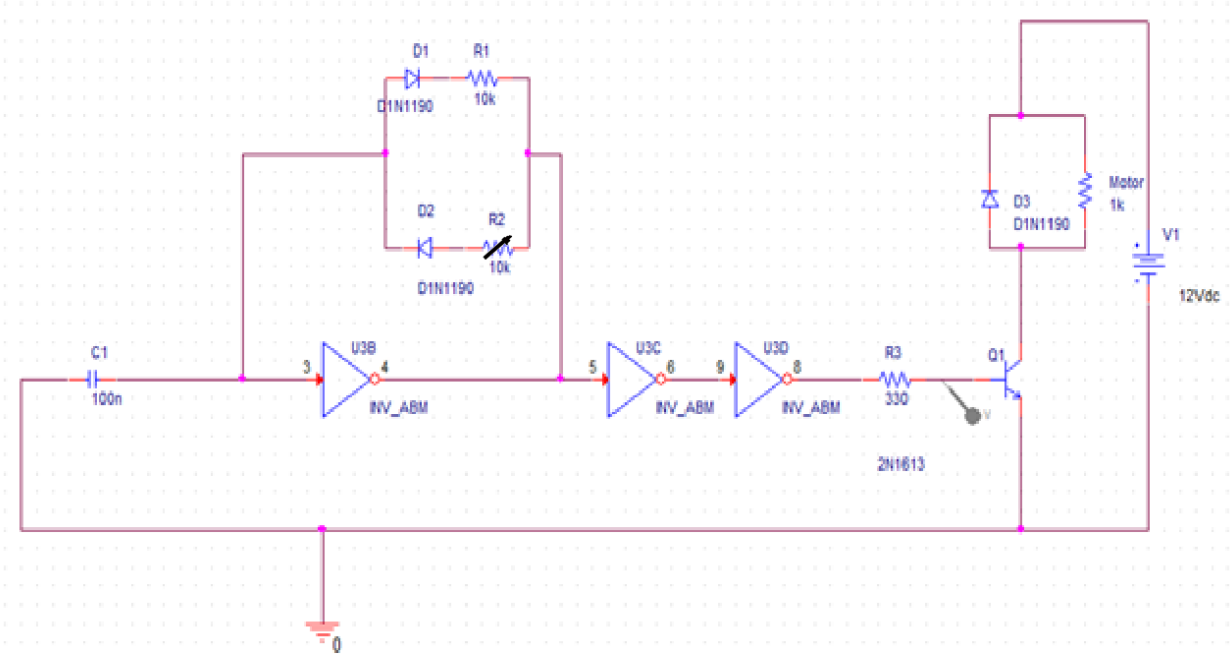


Figure 24. Whole Motor Control Circuit in PSPICE

## 2.3 Risk Analysis

The computability of the computer to implement the algorithm we choose is a crucial determinant of the success of our project. We propose that the computer has to finish processing each image within 0.3 seconds. Even though we implemented our draft algorithm on PC which finishes one image processing in 0.04 seconds [7], the computability difference of different computers is apparent. Therefore, there is a risk that implementing the algorithm on Raspberry Pi may not be fast enough. Under such a condition, we can use a more power computer or slow down the normal speed of the train in order to enrich the processing time as a remedy. The specific calculation of the requirements for the processing speed is included in tolerance analysis.

## 2.4 Tolerance Analysis

The critical component of this project is the speed of the algorithm. As we assumed the normal speed of the train is 10 ± 3 cm/s. With its maximum speed and an acceleration of $-0.01g$, to stop the train, we need

$$d_{braking} = 0.5at^2 = 0.5a(V_{max} / a)^2 = 0.5\,(0.01g)\,(0.13 / (0.01g))^2 = 0.086\,m = 8.6\ cm$$

Eq 13

for the decelerating distance. That means, if we want the train to stop whenever the obstacle is within 50 cm away from the locomotive, the whole system only has

$$(d_{total} - d_{braking}) / V_{max} = (50 - 8.6) / 13 = 3.18\,s \qquad \text{Eq 14}$$

to finish the whole image processing and transmitting process. Eliminate the time for transmitting module and motor control module to react to the signal sent from the computer. The Computing unit only has around 3 seconds to finish the image processing process.

We can use the following formula to compute the success rate for the system stop the train before it hits the obstacle:

$$P_{success} = 1 - (1 - P_{recognize})^{t_{allow} / t_{process}} \qquad \text{Eq 15}$$

$P_{success}$is the success rate to stop the train before it hits the obstacle

$P_{recognize}$ is the success rate for the algorithm to recognize the hazard in front

$t_{allow}$ is the time allowed for processing

$t_{process}$ is the processing time for each image

Let's assume the lowest success rate for recognition of the hazard in each image is 0.2 (it could be much higher). Then, the graph $P_{success}$versus $t_{process}$ is as figure 25:

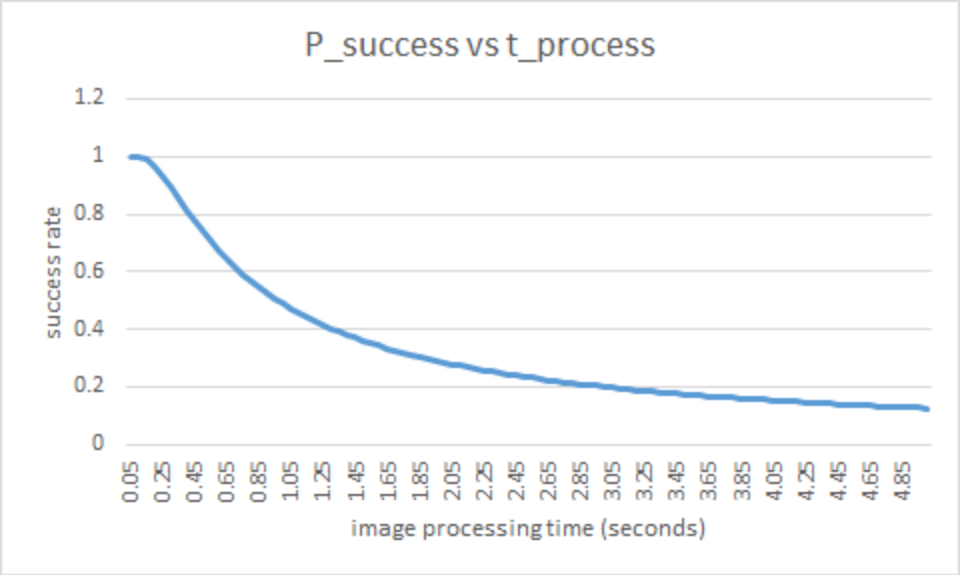


Figure 25. Relation between P_success and t_process

As we can see, to maintain around 90% success rate, the processing time for each image should be at most 0.3 seconds.

3. Ethics and Safety

There are several potential safety hazards in our project. The first safety issue we have thought of is the usage of an AC-DC converter to convert the existing 120V to a smaller voltage for our use, the circuit might heat up and burn if some of the connections of our circuit are broken. Considering the 1st rule of IEEE code of ethics, "accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment" [8], we will do our best to avoid causing unsafe factors. We will check all the connections between AC-DC convertor and our circuits before turning on the operation.

Second, the motor running condition also arouses safety issues. The motor is supposed to run with the rated voltage. However, it is possible that the DC voltage supplied to the motor is not perfect and may not be stable over a long period of time. As a result, there might be some time that the motor will be expected to run with a voltage under or above the rated voltage. This may cause potential damage to the motor, resulting in a crash-down of motor after a long time of usage or testing. Just like we state above, we will do our best to avoid these hazards considering the 1st rule of IEEE code of ethics [8]. To avoid this happening, we would explore how the motor of the toy train works, including output power under rated voltage and lower voltage than the rated voltage. Based on the output power, we will threshold the minimum and maximum voltage supplied to the motor. We would do some experiments to fully justify what we found on the internet about the characteristics of the motor and during this process, the motor would suffer from a risk of heating up as well because the main goal of our project is to brake the train so if we can't control the power of the motor, part of the energy that should have transformed to dynamic energy would be transformed to thermal energy. We would do lots of researches online for instructions

to build up the hardware frame that takes images taken by the camera and transmit it to the other microprocessor so it would be very important to keep academic integrity and put everything we cited clearly in reference.

Third, the Li-Ion battery stores a large amount of energy. If it is operated under inappropriate conditions, it may explode. Considering the 1st rule of IEEE code of ethics [8], we would try our best to avoid this happening by strictly operating the regulatory standard. We will charge the Li-Ion battery with the nominal voltage of 3.2~3.65 volts [10]. We will clean the contacts of the Li-Ion battery before usage [11]. Also, we will use the Li-Ion battery in the temperature range 0 ℃~ 45 ℃[12].

4. Cost

For our labor salary, we think a reasonable salary would be $30/hr, 10hr/wk for three people. Thus we have the costs of labor:

$$3 \bullet \frac{\$30}{hr} \bullet \frac{10hr}{wk} \bullet 6wk \bullet 2.5 = \$13,500 \qquad \text{Eq 16}$$

Since our PCB manufacture and prototype of our circuits on breadboard can be completed in the lab and we can get all resources we need in the lab, our parts and manufacturing costs would be:

| Part | Cost |
|---|---|
| Microcontroller(Raspberry Pi 3 Model B) | $46.92 |
| ESP8266 ESP01 wifi chip (Makefire 4pcs ESP8266 Serial Wireless Wifi Transceiver Module Esp-01 ) | $15.99 |
| Raspberry Pi Camera Module V2 – 8 Megapixel,1080p | $29.96 |
| HO 1:87 Scale Southern Pacific Daylight F2-A Diesel Locomotive Model Power 96812 | $43.74 |
| Total | $136.61 |

Table 8. Cost of parts

In total, the cost of parts and our labor add up to our development cost of $13,636.61.

5. Schedule

| **Week** | **Zhonghao** | **Jianghuai** | **Jin** |
|---|---|---|---|
| **02/27/2017** | Finish the code for steps 1-5 in the algorithm flow chart. | Study the motor characteristics and decide the exact rated voltage. | Complete wifi module with power supply prototype on breadboard |
| **03/06/2017** | Finish the whole algorithm and make it work on different test cases. | Collect all the required circuit components (resistors, capacitors, BJT and etc.) and test their I-V behaviors. | Verification on the speed of wifi module and camera fps |
| **03/13/2017** | Figure out how Raspberry Pi works, and run the algorithm on Raspberry Pi. | Connect all components together on breadboard, and figure out whether they work properly (check with the simulation diagram). | Integrate wifi module with motor control module |
| **03/27/2017** | Connect the computing module to other modules. | If the circuit schematic on breadboard works as expected, convert them into PCB circuit. | Complete PCB prototype for wifi chips |
| **04/03/2017** | Debug and complete integration of all parts. | Test the PCB circuit by connecting the modulated pulse voltage with controlled duty cycle to the track. Fix any unexpected problems. | Debug and complete integration of all parts |
| **04/10/2017** | Final presentation preparation. | Final presentation preparation. | Final presentation preparation |

Table 9. Schedule Table

## 6. Appendix

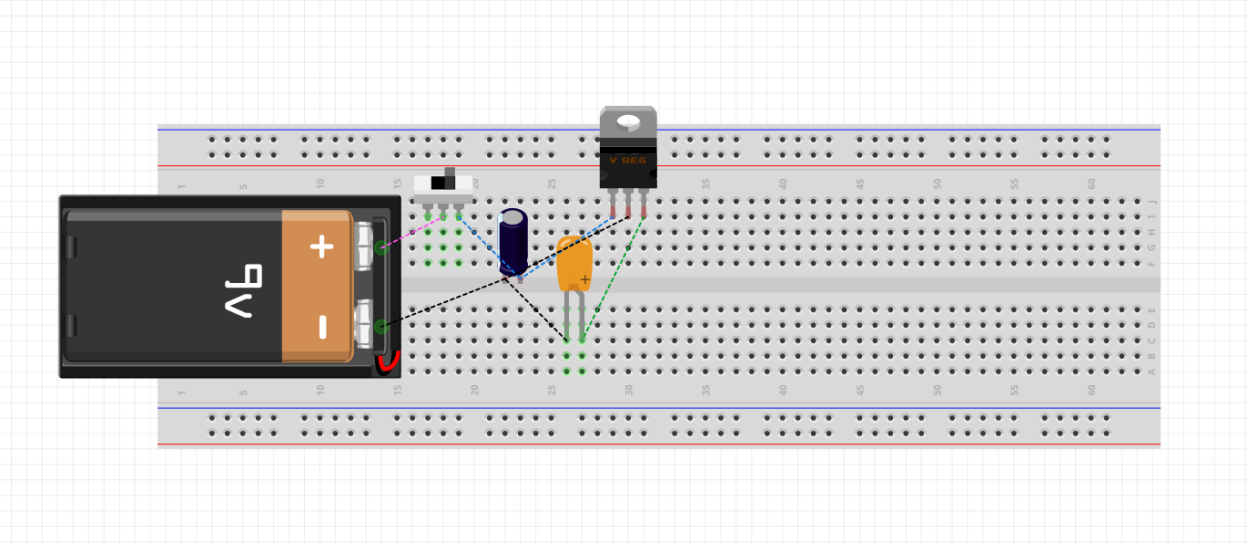


Figure 26. Breadboard connection of voltage regulator

```python
# Changing Raspberry Pi FPS with Python
# allow the camera to warmup and start the FPS counter
print("[INFO] sampling frames from `picamera` module...")
time.sleep(2.0)
fps = FPS().start()

# loop over some frames
for (i, f) in enumerate(stream):
  # grab the frame from the stream and resize it to have a maximum
  # width of 400 pixels
  frame = f.array
  frame = imutils.resize(frame, width=400)

  # check to see if the frame should be displayed to our screen
  if args["display"] > 0:
    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF

  # clear the stream in preparation for the next frame and update
  # the FPS counter
  rawCapture.truncate(0)
  fps.update()

  # check to see if the desired number of frames have been reached
  if i == args["num_frames"]:
    break

# stop the timer and display FPS information
fps.stop()
print("[INFO] elasped time: {:.2f}".format(fps.elapsed()))
print("[INFO] approx. FPS: {:.2f}".format(fps.fps()))

# do a bit of cleanup
cv2.destroyAllWindows()
stream.close()
rawCapture.close()
camera.close()
```

Figure 27. Code for changing Raspberry Pi camera FPS

| state name | state description | pseudo code |
|---|---|---|
| RGB to Grayscale | Convert RGB value to grayscale of each pixel. | pixGray = 0.2989*R + 0.5870*G + 0.1140*B |
| Histogram Calculation | Find the histogram of the bottom line of the image. | for i in [0, 679]:<br>  hito[pixGray[479][i]] += 1 |
| K-means | Use K means to find the mean values M1 and M2 of two classes of the histogram. M1 < M2. M1 represents the class of the track. M2 represents the class of the background | (see figure 28) |
| Scan Pixel | scan the pixel from the left to the right of the line; if the grayscale of a pixel is closer to M1, store the x-coordinate of that pixel to an array. | array = [];<br>for i in [0, 679]:<br>  if abs(pixGray[479][i] - M1) < abs(pixGray[479][i] - M2):<br>  array.append(i) |
| Eliminate Outliers | eliminate the outliers of elements in array[]. | for item in array:<br>  if (item < array[1/4*len(array)] - 1.5*(array[3/4*len(array)]-array[1/4*len(array)])) or (item > array[3/4*len(array)] + 1.5*(array[3/4*len(array) - array[1/4*len(array)])) :<br>  del item; |
| Inconsistency Detection | calculate the mean value M1, M2 in the above line . If M1 > 100, store M1 to gap[]. Otherwise, store M1 to sleepers[] and empty gap[]. | sumTrack = 0;<br>sumBackground = 0;<br>leftBound = array[0] - c*(array[len(array)-1]-array[0]);<br>rightBound = array[array[len(array)-1] + c*(array[len(array)-1]-array[0]);<br>array = [];<br>for j in [leftbound,  rightbound]:<br>  if abs(pixGray[nextLineNum][j] - M1) < abs(pixGray[nextLineNum][j] - M2):<br>  array.append(j);<br>for j in [0, 639]:<br>  if (j < array[0] or j > array[len(array) - 1]):<br>  sumBackground += pixGray[nextLineNum][j];<br>  else:<br>  sumTrack += pixGray[nextLineNum][j];<br>M1 = sumTrack / (array[len(array) - 1] - array[0] + 1);<br>M2 = sumBackground / (639 - (array[len(array) - 1] - array[0] + 1));<br>if M1 > 100:<br>  gap.append(M1);<br>else:<br>  sleepers.append(M1);<br>  gap = []; |

| | | |
|---|---|---|
| Midpoint Calculation | If M1 <= 100, store the midpoint of the track into midpoint[]. | if M1 <= 100:<br>midpoint.append(1/2 * (array[len(array) - 1] + array[0])); |
| Curvature Detection | Detect the curvature of the track. | curv = abs(midpoint[len(midpoint)-1] - 2*midpoint[len(midpoint)-1-n] + midpoint[len(midpoint)-1-2*n]; |

Table 10. States of the Algorithm

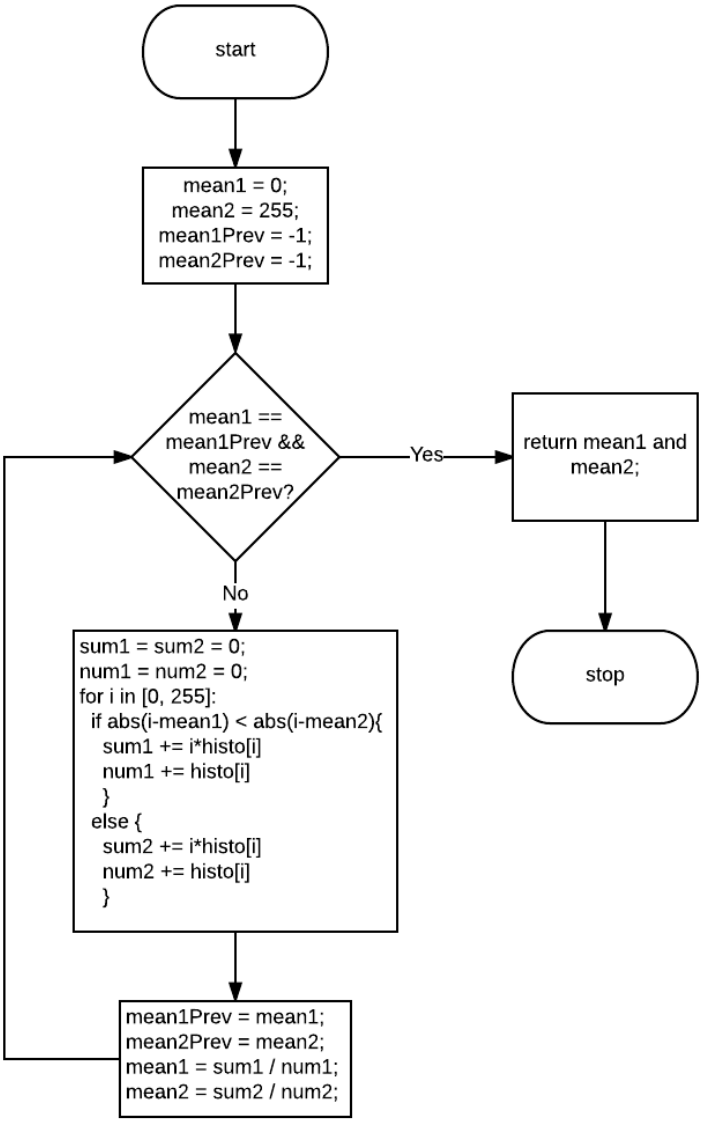


Figure 28. K-means Flowchart

7. References

[1] Model RailRoader, ‘Locomotive Derailment at Turnout’, 2004. [online]. Available: http://cs.trains.com/mrr/f/88/t/250494.aspx. [Accessed: February 8, 2017].

[2] National Model Railroad Association, ‘Trouble Shooting Derailments’, 2001. [online]. Available: http://www.nmra.org/derailments. [Accessed: February 8, 2017].

[3] Raspberry Pi 3 Model B [online]:
https://www.raspberrypi.org/products/raspberry-pi-3-model-b/ [Accessed: Februrary 8, 2017]

[4] Wikipedia [online]:
https://en.wikipedia.org/wiki/Box_plot [Accessed: February 25, 2017]

[5] IEEE, ‘IEEE Code of Ethics’. [online]. Available at:
http://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: February 8, 2017].

[6] “An electromechanical relay.” 2010, University of Surrey. [Online] Available at :
http://info.ee.surrey.ac.uk/Workshop/advice/coils/force.html. [Accessed: February 8, 2017].

[7] PowerStream, “Lithium-ion Battery and Lithium Iron Phosphate Battery Charging Basics”, 2016. [online]. Available at:
http://www.powerstream.com/li.htm. [Accessed: February 8, 2017].

[8] *Safety of Lithium-Batteries*, International standard, IEC 60086-4. [online]. Available at:
https://www.ieee.org/documents/ieeecitationref.pdf. [Accessed: February 8, 2017].

[9] *ESP8266 Arduino tutorial – WiFi module complete review* [online]:
http://www.geekstips.com/esp8266-arduino-tutorial-iot-code-example/ [Accessed: Feburary 13, 2017]

[10] Raspberry Pi Camera module [online]:
https://www.raspberrypi.org/products/camera-module-v2/ [Accessed: Feburary 13, 2017]

[11] PMW Modulation Circuit with Motor Driving [online]:
https://courses.engr.illinois.edu/ece110/content/labs/Experiments/experiment.9.procedures.SP17.v1.pdf

[12] “BU-410: Charging at High and Low Temperatures”, 2016, Battery University. [online]. Available at:
http://batteryuniversity.com/learn/article/charging_at_high_and_low_temperatures [Accessed: Feburary 24, 2017]