A Dissertation

Entitled

Real- Time Simulation of Autonomous Vehicle Safety Using Artificial Intelligence Technique

The University of Toledo

College of Engineering

Department of Electrical Engineering and Computer Science

By

Ahmed Tijani

Committee Members:

Dr. Richard Molyet (Chair)

Dr. Mansoor Alam (Member)

Dr. Daniel Georgiev (Member)

Dr. Mohammed Niamat (Member)

Dr. Junghwan Kim (Member)

Dr. William Evans (Member)

Spring 2020

Copyright© 2020

This document is a copyrighted material. Under copyright law, no parts of this document may be reproduced without the expressed permission from the author.

**Contents**

1. Introduction. …………………………………………………………………………………………..…………….…… 4.

2. Background. ……………………………………………………………………………………………….……………... 7.

2.1 Bayes Theorem. ……………………………………………………………………….…………………… 10.

2.1.1 Naïve Bayes Classifier. …………………………………………………………………..……..……... 11.

2.2 Artificial Neural Networks. ………………………………………….….……………….…..……….14.

2.2.1 Perceptrons. ………………………………………………………………………………………………… 16.

2.2.2 Backpropagation Algorithm. …………………………………………………………..…………….19.

2.3 Programmable Logic Controllers (PLCs). ………………………………………………….......20.

2.3.1 Motion Control. …………………………………………………………………………………………….22.

2.3.2 Stepper Motors. ……………………………………………………………..……….……….………….. 23.

2.4 SIMATIC Safety Fail- Safe System. ………………………………..…….…………………………..24.

2.4.1 Acknowledgment. ……………………………………………………..……………..…………………….24.

2.4.2 Emergency Stop. …………………………………………………….…………..………………………….24.

2.5 Kinematics in 2-D & 3-D. …………………………………………….…………………..……..………25.

3. Research.

3.1 Part 1: Collision Warning System Using Naïve Bayes Classifier. …..………..……... 26.

3.2. Part 2: Obstacle Avoidance System Using Artificial Neural network and Fail- Safe PLC System. ……………………………………….…..………………………….………………………..41.

References. …………………………………………………………………….…………………………………………59.

1. **Introduction**

An autonomous vehicle is a vehicle that can guide and pilot itself without human intervention. Autonomous vehicles are also known as robotic cars, self- diving cars, or driverless cars [1]. In 1939, N. Geddes created an electric vehicle that used radio electromagnetic fields with magnetized spikes planted in the roadway [1]. In 1958, General Motors started developing that concept. Pick-up coils sensors were installed in the car’s front end, and a wire was embedded in the road [1]. The sensors operated on detecting the current flowing through the wire [1]. In 1977, the Japanese devolved that idea by implementing a camera system that captured images of the roads [1]. The camera system transferred the data to the computer to process the images [1]. Today, developing fully autonomous vehicles is on the rise. Vehicles are equipped with Advanced Driver Assistance System (ADAS) that can guide themselves without human assistance [2]. Autonomous vehicles will provide a large consideration about safety, convenience, and ability. Around 1.3 million people worldwide die in car accidents every year [3]. In 2016, around 34,000 fatal car accidents occurred in the United States, a 5.8% increase from last year [3]. Approximately 38% of all fatal car accidents result from collision with another vehicle [3]. Every day, more than 9 people are killed and 1,000 injured due to distracted driving according to U.S Fatal Car Accidents Statistics [3]. Developing fully autonomous vehicles that can drive themselves under any traffic condition without human drivers, can reduce, and perhaps eliminate, the death rate caused by car accidents and improve overall the road safety. They are designed to end the human error of driving actions [2]. One advantage of autonomous vehicles is to provide a convenient way to travel. According to the U.S. Department of Transportation, drivers in the U.S. spend more than 42 hours in delays and traffic congestion [2]. Human error causes 94% of crashes, which is one reason of the traffic congestion and car accidents. [2]. Another benefit of having autonomous vehicles is independence. They are a great option for individuals with disabilities, medical needs or elderly because they have difficulties getting around. This technology possibly will revolutionize the entire transportation system and the concept of operating vehicles.

An autonomous vehicle relies on advanced technologies, such as sensors, cameras, GPS capabilities, laser beams, and Artificial Intelligence (AI) that can observe and collect data from its surrounding [4]. Advanced control systems interpret that information and determine the most appropriate navigation path for the vehicle [4][5]. According to the National Highway Traffic Safety Administration (NHTSA), autonomous vehicles will integrate onto the road by six levels of ADAS [4][5]:

**Level 0**: No autonomy, the human driver operates the vehicle.

**Level 1**: The driver controls the vehicle, but the ADAS on the vehicle sometimes can assist.

**Level 2**: The ADAS can control part of the vehicle, such as braking, acceleration, but the human driver must remain and monitor the driving tasks all the times.

**Level 3**: limited self-driving automation, ADAS operates the vehicle in limited special areas, no human assist needed

**Level 4**: Automated, no human intervention is needed.

**Level 5:** fully- Automated, the human driver is considered as a passenger.

The approach of real-time simulation of autonomous vehicle safety using artificial intelligence technique is to improve the safety on the roads and control the direction of the vehicle to avoid obstacles. Collision warning system is presented at first as a method to assist drivers. Human drivers will be replaced by the obstacle avoidance system, which controls the directions of the vehicle. The Naïve Bayes classifier from supervised machine learning was applied for collision warning system. Artificial Neural Networks (ANNs) and the Fail- Safe PLC system was implemented to control the direction of the vehicle. Fail-safe PLC system offers a safety concepts in a field of machine and personnel protection.

1. **Background**

**Machine Learning**

Machine learning (ML) is one of the main information technology topics that is going to become part of our life. Machine Learning is a branch of artificial intelligent (AI) that allows computers to learn without being programmed. It studies algorithms which makes computers operate similarly to humans [6]. Machine learning algorithms use statistical analysis to allow computers to train on data inputs to predict values of the outputs that set in a specific range [7]. The techniques of machine learning are used to discover complex that cannot be achieved by humans [8].

**Types of Machine Learning Algorithms:**

* **Supervised Learning**

Supervised machine learning is an algorithm that uses outcome data that is known to define a predictive model. It adopts the model to recreate outputs known from a training example. After the system received input data with the output data, it builds rules that map the input to the output. The training procedure should progress until the performance level reaches a high standard. The system can assign new output parameters after the training. The algorithms implement optimization routines to minimize error function [9]. The model can be trained by several learning algorithms. The most common algorithms are decision trees, Linear Regression, K-nearest Neighbor, Naïve Bayes, and Neural Networks. [9]

**Types of Supervised Machine Learning:**

* **Classification**: Classification is a method that is used to recreate class assignments. It predicts the model outcome when the output parameter is in the form of categories. The classification output has discrete values. It separates data into classes and predicts the response value. Predicting spam messages, the weather monsoon, and a person gender are examples of classification [10].
* **Regression**: Regression is a technique that is implemented to predict model outputs when the outcome parameters are in the form of real value. It is used to define and predict some specific values, such as weight, prices, and stocks. The regression output has continuous values [10].
* Decision Trees:

Decision tree is an algorithm that has applications spanning several different areas. The construction of the decision trees is built to identify ways to split data that is relied on different conditions. A decision tree is a tree-like chart that consists of nodes, edges, and leaves. The nodes identify an attribute location. The edges show the results of the classification. The leaves illustrate the class label. The classification of the examples occurs by sorting them down the tree from the root to the same leaf node. Each node preforms a test case in the tree for some attribute. Each edge connecting from a node corresponds to one of the possible solutions to the test case. This procedure is repeated for every subtree rooted at the new nodes. They are used for tasks, such as classifications and regression. The objective of using decision trees is to build a model that predicts the target values by learning decision rules inferred from the features of the data [12].

* Linear Regression:

Linear recession is a method type of predictive analysis. It examines the performance of a set of predictor examples in predicting an outcome. Also, it determines the predictor examples that are significant predictor of the outcome in a way the magnitude and sign of beta estimates. The regression estimates are applied to illustrate the relationship between a dependent parameter and an independent or more parameters. Multiple linear regression, simple linear regression, and logistic regression are the types of linear regression [11].

* K-nearest Neighbor:

K-nearest neighbor (kNN) is one of the most fundamental and basic classification methods. The algorithm is a pattern recognition model that is applied for classification and regression. The input of either classification or regression composes of the k closest trading examples within a space. K-nearest neighbor was developed to implement discriminant analysis when reliable parametric estimates of probability densities are unknown.

* **Unsupervised Learning**

Unsupervised machine learning is an algorithm that detects and analyzes unlabeled data. It interduces a subset of machine learning tasks that are using unlabeled training data. The data does not carry any kind of label designation to its classification. The methods of implementing unsupervised machine learning must learn the correlation between the elements in a dataset without labeling the data as any specific classification. Unsupervised learning is a method to solve hidden data structures that could not be evident [7,11].

* **Semi-supervised Learning**

Semi-supervised machine learning is a combination between supervised machine learning and unsupervised machine learning. The algorithm is based on training a labeled dataset that contains the outcome information, which allows the algorithm to understand the patterns and recognize the correlation between the target parameters and the rest of the dataset. Semi-supervised learning algorithm learns from a dataset that has no outcome parameters. Both labeled and unlabeled datasets generate the learning algorithm. The learning algorithm can be applied to some cases related to a learning process in which the predicted output parameters are not available. Applying semi-supervised learning techniques are needed to gain effective outcomes. Medical applications are an example of applying semi-supervised learning [10,12].

* **Reinforcement Learning**

Reinforcement learning (RL) is a training algorithm for machine learning models that make a sequence of decisions. The learning of RL agent is based on the consequence of its actions. It chooses its actions based on previous knowledge. The RL receiving signal is a numerical reward that encodes an action outcome, and the agent requires to learn to choose actions that maximize accumulated reward [13].

* 1. **Bayes Theorem**

Bayesian learning methods are among the most practical approaches to some types of learning problems. In machine learning, the objective is to define the best hypothesis that is the most probable hypothesis from the space *H* giving the observed training data *D*. The approach of the Bayes theorem is to provide a way to compute probabilities. It computes the probability of a hypothesis based on its prior probability. Bayes theorem is an essential part of Bayesian learning methods because It provides a way to compute the posterior probability from the prior probability [14].

**Bayes Theorem:**

Where, P(*h*) is the initial probability that hypothesis *h* holds before observing the training data. P(*D*) is the prior probability of the observing training data *D*. P(*D|h*) is the probability of observing data *D* in which the hypothesis h holds. *P(h|D)* is posterior probability of *h*. *P(h|D)* reflects the training data *D* in contrastto the prior probability *P(h)* thatis independent of D [6].

According to the Bayes theorem, P(h|D) increases with P(h) and P(*D|h*). Also, P(*h|D*) decreases as P(*D*) increases because the *D* will be observed independent of h. To determine the MAP hypostasis, Bayes theorem can compute the posterior probability of every candidate hypothesis [14].

Therefore, the *hMAP* provides  *hMAP* = argmax P(h|D)

= argmax P(D|h) P(h) (2)

*h ϵ H*

* + 1. **Naïve Bayes Classifier**

The Naïve Bayes classifier is one of the commonly used supervised machine learning algorithms. It is based on Bayes theorem. The performance of the Naïve Bayes is comparable to some types of supervised machine learning, such as the decision tree and neural networks. The Naïve Bayes classifier is applicable to learning tasks in which each conjunction of parameter value *x* is defined by a set of attribute values where the target function *f(x)* can assume any value from some finite set *V.*  To classify the next instance, the Bayesian approach is applied to specify the most probable target value (*vMAP*), giving the attribute values <a1, a2, a3, …an> that show [14]:

VMAP= argmax *P(vj | a1,a2 … an).*

vjϵv

Using Bayes theorem, the above equation ~~as~~ becomes:

VMAP= argmax *P(a1,a2 … an | vj) P(vj).* (3).

vjϵv

The naïve Bayes classifier is based on the hypothesis that the attribute values are conditionally independent given the target value, which is *P(a1,a2 … an | vj) =∏****i*** *P(ai|vj).* Substituting the equation into (3), we get the naïve Bayes approach [14].

Therefore, the Naïve Bayes classifier is:

VNB = argmax *P*(*vj*) ∏ *P*(*ai*|*vj*). (4).

vjϵv *i*

where VNB is the target value output by the classifier.

**Classification Accuracy:**

Classification accuracy is the ratio of the number of the correct predictions to the sum of the number of input samples [15].

Accuracy (5)

**Confusion Matrix:**

A confusion matrix or an error matrix defines the performance of a classification model on a known dataset with true variables. It enables the visualization of the performance of an algorithm. Also, it recognizes confusion between classes in which a class is commonly mislabeled as the others. A confusion matrix summarizes predicted outcome on a classification problem. The true number of predictions and false numbers are classified by each class. It provides an insight into the errors that are made by the classifier. There are two classes of a confusion matrix: positive and negative [15].

**Definition of the classes [15]:**

* Positive (P): true observation
* Negative (N): False observation
* True Positive (TP): Both the prediction and observation are true.
* True Negative (TN): Both the prediction and observation are false.
* Fales Positive (FP): The prediction is true. However, the observation is false.
* False Negative (FN): The prediction is false. However, the observation is true.
* Accuracy (6)
* Error Rate (misclassification Rate) . (7)
* True Positive Rate: The true positive rate is also known as recall or sensitivity. It measures the frequency of the positive predictions.

Sensitivity . (8)

* False Positive Rate: Measures the frequency of negative prediction
* False positive rate . (9)
* True Negative Rate (Specificity): The frequency of negative predictions
* True Negative Rate. (10)
* Precision: The correct prediction of the actual positive.
* Precision. (11)
* Prevalence: The positive condition happened in the sample:
* Prevalence. (12)
  1. **Artificial Neural Networks**

**Neural Network**

The human brain has 100 billion neurons. Every neuron has 104 connections. Every millisecond the neurons fire with a massive parallel processing [16]. A neuron consists of three parts: a cell body, the dendrites, and an axon [17]. The cell body controls all activities of the neuron. The dendrites work on receiving messages from other neurons and pass those messages to the cell body. The axon’s job is to transfer the messages from the cell body to dendrites of other neurons. Messages are moving from a neuron to another neuron with incredible speed every moment. Synapses are the place where the neurons communicate. A neuron releases a neurotransmitter into the synaptic cleft in order to send a message. The neurotransmitter crosses the synapse and attaches to receptors on the next neuron in line. Changes occur inside the receiving neuron when the neurotransmitters attach to the receptors. As a result, the message is delivered [9].

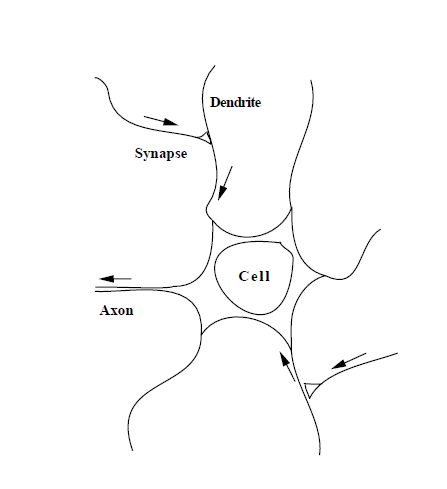


Figure 1 [16]: The Biological Nervous Cell (the neuron).

**Artificial Neural Networks**

Neural network learning methods are a way to estimate real, discrete, and vector valued target functions. They are one of the most efficient learning methods for specific types of applications, such as interpreting complex real-world sensor data. The algorithms of artificial neural network show successful results in many practical problems, such as recognizing handwritten characters, voices, and faces. The artificial neural networks are inspired by the observation of biological learning systems that rely on very complex webs of interconnected neurons, and the structure of the artificial neural networks is based on a densely interconnected set of units. Every unit takes a number of real-valued inputs and generates a single real-valued output [14].

* + 1. **Perceptrons**

The perceptron is a type of ANN system [14]. A perceptron is an algorithm that classifies the given input data [18]. It is a linear binary classifier that is used in supervised machine learning [18]. A single perceptron is used to implement many boolean functions [14]. The perceptron assigns a set of weight to its real and Boolean inputs along with a bias value [14]. Many of the primitive boolean functions, such as AND, OR, NAND, and NOR can be implemented by the perceptron. The perceptron consists of four layers [13]: an inputs layer (*xi*), bias (bi) and weights (*wi*), a summation net, an activation function (*f*), as shown in figure 2. Every vector of real-valued input (*xi*) is multiplied by its weight (*w*i). The weighted values are added, and then applied to the activation function (*f*), which generates the perceptron output. The role of the weight is to indicate the relative importance of a particular node. The bias determines the amount of vertical displacement of the activation function. The activation function maps the input between desired values, such as (-1,1), (0,1), etc [18]. For example, the signum activation function, which produces an output of 1 if the result exceeds the threshold, or -1 otherwise [10].

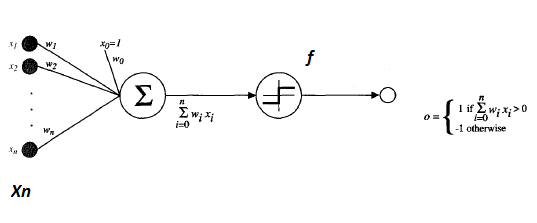


Figure 2 [10]. The perceptron.

 Figure 2.

*O (x1…, xn)* = 1 if w0+w1x1+w2x2+…+wnxn > 0. (13)

-1 Otherwise.

**The Training Rule of the Perceptron**

The learning problem is to devise of a weight vector that makes the perceptron generate the correct output for every training example that is given. The perceptron rule and the delta rule are two algorithms that are used to solve the learning problem, and which converge to satisfactory hypotheses. Starting with a random value, each weight is adjusted iteratively by using the perceptron training rule until the output of the perceptron matches the output of the training example. This process is shown in equation 14 [14].

*w****i*** *← w****i****+ ∆w****i*** (14).

where,

*∆ w****i****=Ꞃ (t-o) x****i***

* ***t***is the training example target output.
* ***o***is theperceptron output.
* ***Ꞃ*** isthe learning rate.

The learning rate adjusts the degree to which the weights are changed at every iteration [10].

**Activation Functions (*f*)**

An activation function is a mathematical relation that is associated with each neuron in the network, and which regulates its output [15]. Neurons can use any differentiable transfer function ***f*** to produce their output. Table 1 compares the properties of three common of activation functions [16].

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Name | Equation | Range | Plot | Notes | | Log-Sigmoid Transfer function (logsig) | **F(x) =** | (0,1) |  | Logsig transfer function produces outputs 0 and 1 as the neuron’s net input goes from negative to positive infinity. | | Tan-Sigmoid Transfer Function (tansig) | **f(x)= Tanh(x) =** | (-1, +1) |  | tansig transfer function produces outputs -1 and 1 as | | Linear Transfer Function (purelin) | **f(x)=x** | **(-∞,+∞)** |  | purelin transfer function produces outputs **-∞** and **∞**. | |  |  |  |  |
|  |  |  |  |  |
| Table 1 [16]. properties of three common activation functions |  |  |  |  |
|  |  |  |  |  |

**2.2.2 The Backpropagation Algorithm**

Several perceptrons may be interconnected to form a multilayer network which is called a neural network. The process of training such a network is more involved than for a single perceptron and is known as backpropagation. It utilizes several algorithms, such as a gradient descent algorithm [16]. A training set of values is applied to the network inputs and the resulting network outputs are compared to the outputs specified for the training set. If the output error is greater than a threshold value, the weights of the network are randomly adjusted, and the network error is reevaluated. This process is repeated until the network error is less than threshold value. That point, the network has been trained and the weights remain fixed. Trained backpropagation networks can identify new inputs that are introduced to the network for the first time. Consequently, the new inputs produce outputs that are similar to the trained outputs [16]. The training process comprises preparing the training data, building the network, training the network, simulation and reaction of the network to new inputs [10].

The errors for networks with multiple output units are different from those for the networks with a single output unit. The errors (*E*) for multiple output units are the sum of the errors of all of the network output units [10].

Hence,

E(w) = ∑ dϵD ∑kϵoutputs (tkd-okd)2  (15).

where, Outputs is set of the output units in the network.

tkd is the target.

okd is output values of the training example *d*.

**3 Programmable Logic Controllers (PLCs)**

In 1968, a group of engineers from General Motors devolved the first programmable logic controller (PLC) system that could replace complex relay control systems. These specifications had to be in the new control system [17]:

* Simple to program
* Reliable and small
* The maintenance cost should be Low
* Program changes without internal rewiring

Programmable logic controller (PLC) is a different form of microprocessor-based controller that stores instructions in a programmable memory to implement functions. Those functions, such as timing, counting, and sequencing are responsible for controlling and operating machines. The PLCs are designed and pre-programmed to have a simple and intuitive form of language, called leader logic, that is easy to program and understand associated with logic and switching operations. The expression logic describes the implementation of logical and switching operations, such as switch *z* on if x or y occurs. Input and output devices are connected to the PLC, and then operators generate certain instructions which send into the PLC memory. The outputs of devices will be observed by the controller according to the program, and the controller implements the control rules for the program [18].

The PLC system consists of the following components as shown in figure 3: Power Supply Unit, Input and Output Interfaces (I/O), Memory, Central Processing Unit (CPU), Communications Interface, and Programming Device as shown in figure 3 [18].

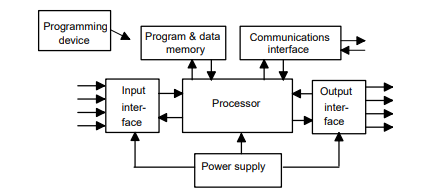


Figure 3[18]. The PLC system.

The microprocessor is located in the processor or central processing unit (CPU). It interprets the input signals of the programs that are stored in the memory and processes the action of the control to the outputs. The power supply unit is to convert the AC-DC voltage that the processor required. The programming device is needed for entering the program into the memory of the processor. The memory unit is for storing programs. The input and output interface units are for receiving and communicating data from devices to other devices. Communication interface is the communication networks for receiving and transmitting information from or to other remote PLCs. The PLCs are rugged and are made to endure hostile environments, such as vibrations, temperature, humidity, and noise [18].

**2.3.1. Motion Control**

Motion Control Functionality of CPU S7-1200.

The motion control functionality supports controlling motors, such as stepper and servo motors with pulse interface. It controls the position axis and pulses of drives and encoders. Operators can control motors, such as the stepper motors and the servo motors using the motion control functionality. The CPU integrates the operations of the PLC with motion control functionality for drives. The devices will be controlled and observed by the motion control functionality. The CPU uses technology objects such as to control devices and motors. Operators can design and configure projects through the TIA portal, and load the configurations into the CPU. The motion control functionality operation occurs in the CPU. Motion control instructions control the technology objects in the user program. Other functions, such as diagnostics, commissioning, and optimization are provided to monitor programs after they are downloaded [19].

* Technology objects for motion control.

Position axis (TO\_PositioningAxis): configures the positioning axis technology object with parameters such as:

* Selecting the Pulse Train Output (PTOs)
* Mechanics and gear transmission of the drive
* Position limits and position monitoring
* Control loop

The configuration is stored in the data block. The data block designs the interface between the operator program and the CPU firmware. The current axis will be saved in the data block. A summary of possible tasks for controlling the axes is given in Appendix A [19]:

**2.3.2. Hybrid Stepper Motor**

Hybrid step motors provide a high performance in many industrial applications. They have high static and dynamic torque. They also run at very high step rates. They show an excellent quality of variable reluctance and permanent magnet step. Hybrid step motors are reliable and digitally controlled motors. The motor performs through the electrical pulses that control the operation of the motor. The electrical pulse changes the direction of the current flowing through the winding, and it is transformed into shaft rotation in step of an angle. Therefore, it creates an open loop system with the driver [20].

One advantage of the stepper motors is that they are low in price compared to servo motors. Also, their systems do not require tuning, which allow them to have a very high torque density. Their high stability torque makes them stable for small quick movements and for driving high inertia loads [20].

There are several types of different stepper motors, such as: Permanent Magnet Stepper Motors, Variable Reluctance Stepper Motors, and Hybrid Stepper Motors. The Permanent Magnet Stepper has a permanent magnet rotor. The Variable Reluctance Stepper Motors have a geared rotor (Non -magnetic). The Hybrid Stepper Motors combine characteristics from both the Permanent Magnet and the Variable Reluctance motors [21]. The hybrid stepper motor was selected for the obstacle avoidance system because of its versatile characteristics.

**SIMATIC Fail- Safe System**

The fail-safe system is engineered to fail in a safe state, so that it reduces danger to humans and environment. It offers a safety concepts in a field of machine and personnel protection. Fail-safe automation systems control processes that obtain a safe state immediately if an unexpected failure or operation occurred. The immediate shutdown to safe state happened to protect humans or the environment. The systems cover more than the traditional safety engineering systems. They enable far reaching intelligent systems, such as measuring systems and electrical drives. F-System follows the safety requirements safety integrity level SIL3 and performance level (PL) e category 4. S7 F/FH executes the safety functions when a risky event occurs in order to restore a safe state in the system. Safety processing of field information such as Emergency stop button and motor are controlled by F-I/O. they are equipped with the required hardware, software, and safety class. The safety function for the process could be provided by a fault reaction function. If the F-system in a fault situation could not execute an actual safety function, it executes the fault reaction function. For instance, when the associated outputs are disabled, and the F-CPU switches to STOP mode [21].

F-runtime group:

A safety program composes of one or two F-runtime groups. An F-runtime group is a logical consists of some related F-blocks that are internally formed by the F- block system. Each runtime group have F-blocks that are automatically added and are used FBD or LAD. The main safety block is the initial F-block of the safety program that is available for programing.

**Global Acknowledgment (ACK\_GL):**

The global acknowledgment generates an acknowledgment for the simultaneous reintegration of all F-I/O or channels of the F-I/O of the f-runtime group after communication errors, F-I/O errors, or channel faults. A positive edge ACK\_GLOB input is required for reintegration. The acknowledgment happens analogously to the user acknowledgment. However, it operates simultaneously on all F-I/O of the F-runtime group.

**Emergency Stop:**

Emergency stop (Estop) instruction applies an emergency stop or emergency shutdown with acknowledgment for stop categories 0 and 1. When the E\_STOP input signal is 0, it enables the output signal and reset it to 0. If the E\_STOP input signal is 1, it enables the output signal after an acknowledgment happened. Every call of the E\_STOP instruction must select a data area in the instruction data storage. When the instruction is installed in her program, the call options dialog is opened automatically.

**2.4 Kinematics in 2-D**

Kinematics is the examination of the geometry of motion. The motion of a body is affected by its surroundings, such as a gas, liquid or solid or by electric or magnetic fields. This motion could be the result of direct contact or gravitational forces and corresponds to physical laws. Kinematics determines the motion of a moving body when that motion is limited because that body is connected to other elements. Two methods can be used to determine the position of a body. The first method calculates the position by using three points within the body which are not oriented in a straight line. The second method calculates the position by using a single point without the body and the orientation of the body with respect to a reference frame. Kinematics studies can be made in one, two, or three dimensions. Single dimension kinematics concerns motion in one direction, and utilizes variables such as position, velocity and time. Two-dimension kinematics concerns motion in two directions that lie in a plane. Position can be expressed in either cartesian or polar coordinates [22].

**Displacement (∆x):** The movement of an object’s position to a reference frame. This change is defined as displacement [23].

**Displacement of an object:**

***∆x= xf-x0*.** (16)

where,

***∆x*** is displacement,

***xf*** is the final position

***x0*** is the initial position.

**Pythagorean Theorem:**

The length of the legs of a right triangle is presented as ***x*** and ***y.*** **z** is length of the hypotenuse. The sum of the squares of the lengths of the legs is equal to the square of the length of the hypotenuse [24].

Given,

*x2+y2=z2* (17)

Motion in two dimensions uses the Pythagorean Theorem to find the straight-line distance [23].

**Obtuse Angle:**

An angle that is between 90 degrees and 180 degrees. It is larger than 90 degrees but smaller than 180 degrees [25].

**Sine Rule:**

(18)

**Cosine Rule:**

*a2=b2+c2-2bc cos A* (19)

**Part 1: Collision Warning System Using Naïve Bayes Classifier**

This study investigated a method of determining the potential for a rear-end collision between highway vehicles.A set of training examples was generated to be performed under ideal weather conditions between two vehicles, V1 & V2. The first vehicle (V1) was pursued by the second vehicle (V2). The probabilities of potential collisions were calculated by applying the Naïve Bayes classifier. An alarm would activate, and a text message would display on the dashboard of V2, if there was a potential for a collision. The training examples consisted of conjunctions of three variable parameters: speed, distance, and acceleration. The speed was considered high, if it was between 75-85 MPH; within the speed limit, if it was between 45-74 MPH; and low, if it was between 35-44 MPH. The legal safe distance between two vehicles under ideal weather conditions is 3m/10 feet according to FMCSA [26]. Therefore, the distance between V1 and V2 was safe, when it was 3m; far, when it was longer than 3m; and close, when it was shorter than 3m. Also, the acceleration for both vehicles was determined. True indicated that the vehicle was accelerating, and false indicated that it was not. The collision system calculated whether there was a potential for a collision between the two vehicles on a highway. The calculations were based on the data that the system received, and the system determined whether there was a potential for a collision by applying the Naïve Bayes theorem. The collision warning system is designed to assist drivers avoid potentially dangerous driving situations. The results showed that the collision warning system successfully predicted and responded correctly to different driving scenarios. Also, the collision warning system has 89% accuracy of predicting correctly collisions. Below, is a table containing a set of training examples.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Direction | Speed V1 | Speed V2 | Distance | Acceleration V1 | Acceleration V2 | PC |
| 1 | One Way | High Speed | Speed Limit | Legal | T | T | No |
| 2 | One Way | High Speed | Speed Limit | Legal | T | F | No |
| 3 | One Way | High Speed | Speed Limit | Legal | F | T | Yes |
| 4 | One Way | High Speed | Speed Limit | Legal | F | F | No |
| 5 | One Way | High Speed | Speed Limit | Close | T | T | Yes |
| 6 | One Way | High Speed | Speed Limit | Close | T | F | Yes |
| 7 | One Way | High Speed | Speed Limit | Close | F | T | Yes |
| 8 | One Way | High Speed | Speed Limit | Close | F | F | Yes |
| 9 | One Way | High Speed | Speed Limit | Far | T | T | No |
| 10 | One Way | High Speed | Speed Limit | Far | T | F | No |
| 11 | One Way | High Speed | Speed Limit | Far | F | T | Yes |
| 12 | One Way | High Speed | Speed Limit | Far | F | F | No |
| 13 | One Way | High Speed | High Speed | Legal | T | T | No |
| 14 | One Way | High Speed | High Speed | Legal | T | F | No |
| 15 | One Way | High Speed | High Speed | Legal | F | T | Yes |
| 16 | One Way | High Speed | High Speed | Legal | F | F | No |
| 17 | One Way | High Speed | High Speed | Close | T | T | Yes |
| 18 | One Way | High Speed | High Speed | Close | T | F | Yes |
| 19 | One Way | High Speed | High Speed | Close | F | T | Yes |
| 20 | One Way | High Speed | High Speed | Close | F | F | Yes |
| 21 | One Way | High Speed | High Speed | Far | T | T | No |
| 22 | One Way | High Speed | High Speed | Far | T | F | No |
| 23 | One Way | High Speed | High Speed | Far | F | T | Yes |
| 24 | One Way | High Speed | High Speed | Far | F | F | No |
| 25 | One Way | High Speed | Low Speed | Far | T | T | No |
| 26 | One Way | High Speed | Low Speed | Far | T | F | No |
| 27 | One Way | High Speed | Low Speed | Far | F | T | No |
| 28 | One Way | High Speed | Low Speed | Far | F | F | No |
| 29 | One Way | Speed Limit | Speed Limit | Legal | T | T | No |
| 30 | One Way | Speed Limit | Speed Limit | Legal | T | F | No |
| 31 | One Way | Speed Limit | Speed Limit | Legal | F | T | Yes |
| 32 | One Way | Speed Limit | Speed Limit | Legal | F | F | No |
| 33 | One Way | Speed Limit | Speed Limit | Close | T | T | Yes |
| 34 | One Way | Speed Limit | Speed Limit | Close | T | F | Yes |
| 35 | One Way | Speed Limit | Speed Limit | Close | F | T | Yes |
| 36 | One Way | Speed Limit | Speed Limit | Close | F | F | Yes |
| 37 | One Way | Speed Limit | Speed Limit | Far | T | T | No |
| 38 | One Way | Speed Limit | Speed Limit | Far | T | F | No |
| 39 | One Way | Speed Limit | Speed Limit | Far | F | T | Yes |
| 40 | One Way | Speed Limit | Speed Limit | Far | F | F | No |
| 41 | One Way | Speed Limit | High Speed | Legal | T | T | Yes |
| 42 | One Way | Speed Limit | High Speed | Legal | T | F | Yes |
| 43 | One Way | Speed Limit | High Speed | Legal | F | T | Yes |
| 44 | One Way | Speed Limit | High Speed | Legal | F | F | Yes |
| 45 | One Way | Speed Limit | High Speed | Close | T | T | Yes |
| 46 | One Way | Speed Limit | High Speed | Close | T | F | Yes |
| 47 | One Way | Speed Limit | High Speed | Close | F | T | Yes |
| 48 | One Way | Speed Limit | High Speed | Close | F | F | Yes |
| 49 | One Way | Speed Limit | High Speed | Far | T | T | Yes |
| 50 | One Way | Speed Limit | High Speed | Far | T | F | Yes |
| 51 | One Way | Speed Limit | High Speed | Far | F | T | Yes |
| 52 | One Way | Speed Limit | High Speed | Far | F | F | Yes |
| 53 | One Way | Speed Limit | Low Speed | Legal | T | T | No |
| 54 | One Way | Speed Limit | Low Speed | Legal | T | F | No |
| 55 | One Way | Speed Limit | Low Speed | Legal | F | T | yes |
| 56 | One Way | Speed Limit | Low Speed | Legal | F | F | No |
| 57 | One Way | Speed Limit | Low Speed | Far | T | T | No |
| 58 | One Way | Speed Limit | Low Speed | Far | T | F | No |
| 59 | One Way | Speed Limit | Low Speed | Far | F | T | No |
| 60 | One Way | Speed Limit | Low Speed | Far | F | F | No |
| 61 | One Way | Low Speed | Speed Limit | Legal | T | T | Yes |
| 62 | One Way | Low Speed | Speed Limit | Legal | T | F | Yes |
| 63 | One Way | Low Speed | Speed Limit | Legal | F | T | Yes |
| 64 | One Way | Low Speed | Speed Limit | Legal | F | F | Yes |
| 65 | One Way | Low Speed | Speed Limit | Close | T | T | Yes |
| 66 | One Way | Low Speed | Speed Limit | Close | T | F | Yes |
| 67 | One Way | Low Speed | Speed Limit | Close | F | T | Yes |
| 68 | One Way | Low Speed | Speed Limit | Close | F | F | Yes |
| 69 | One Way | Low Speed | Speed Limit | Far | T | T | Yes |
| 70 | One Way | Low Speed | Speed Limit | Far | T | F | Yes |
| 71 | One Way | Low Speed | Speed Limit | Far | F | T | Yes |
| 72 | One Way | Low Speed | Speed Limit | Far | F | F | Yes |
| 73 | One Way | Low Speed | High Speed | Legal | T | T | Yes |
| 74 | One Way | Low Speed | High Speed | Legal | T | F | Yes |
| 75 | One Way | Low Speed | High Speed | Legal | F | T | Yes |
| 76 | One Way | Low Speed | High Speed | Legal | F | F | Yes |
| 77 | One Way | Low Speed | High Speed | Close | T | T | Yes |
| 78 | One Way | Low Speed | High Speed | Close | T | F | Yes |
| 79 | One Way | Low Speed | High Speed | Close | F | T | Yes |
| 80 | One Way | Low Speed | High Speed | Close | F | F | Yes |
| 81 | One Way | Low Speed | High Speed | Far | T | T | Yes |
| 82 | One Way | Low Speed | High Speed | Far | T | F | Yes |
| 83 | One Way | Low Speed | High Speed | Far | F | T | Yes |
| 84 | One Way | Low Speed | High Speed | Far | F | F | Yes |
| 85 | One Way | Low Speed | Low Speed | Legal | T | T | No |
| 86 | One Way | Low Speed | Low Speed | Legal | T | F | No |
| 87 | One Way | Low Speed | Low Speed | Legal | F | T | Yes |
| 88 | One Way | Low Speed | Low Speed | Legal | F | F | No |
| 89 | One Way | Low Speed | Low Speed | Close | T | T | Yes |
| 90 | One Way | Low Speed | Low Speed | Close | T | F | Yes |
| 91 | One Way | Low Speed | Low Speed | Close | F | T | Yes |
| 92 | One Way | Low Speed | Low Speed | Close | F | F | Yes |
| 93 | One Way | Low Speed | Low Speed | Far | T | T | No |
| 94 | One Way | Low Speed | Low Speed | Far | T | F | No |
| 95 | One Way | Low Speed | Low Speed | Far | F | T | Yes |
| 96 | One Way | Low Speed | Low Speed | Far | F | F | No |

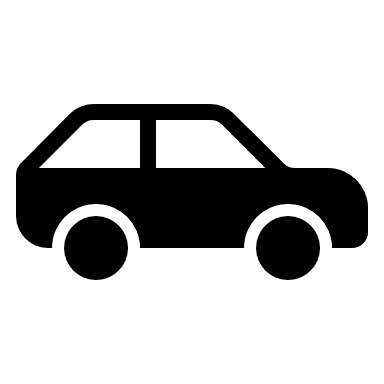
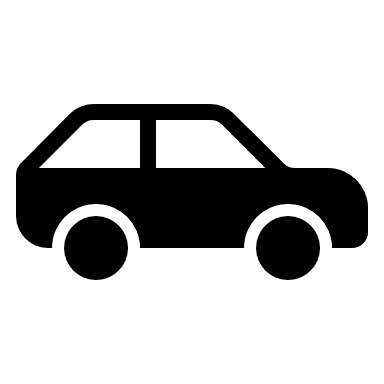
Table 2.

Three scenarios were examined for V1 and V2. The three scenarios are depicted in figures 4, figure 9, and figure 10 respectively.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | One Way-Two Lanes | Speed | Acceleration | Distance | Potential Collision |
| **Scenario 1** | Vehicle (V1) | High Speed | True | Far | No |
| Vehicle (V2) | Low Speed | False |
| **Scenario 2** | Vehicle (V1) | Speed Limit | False | Close | Yes |
| Vehicle (V2) | High Speed | True |
| **Scenario 3** | Vehicle (V1) | Speed Limit | False | Legal | No |
| Vehicle (V2) | Speed Limit | False |

Table 3. Three possible training example scenarios.

**Scenario 1:**



V1

V2

Figure 4. A safe scenario for V1 and V2.

In figure 4, V1 is moving at a high speed of 100 MPH with acceleration, and V2 is moving at a low speed of 30 MPH with no acceleration. The distance between the two vehicles is 8m/26ft, which is far. The speeds introduced in this scenario are considered as a new data to the system.



Figure 5. The probability for scenario 1.

Figure 5 shows the results of applying the Naïve Bayes theorem. After entering the values of speed, distance, and acceleration for scenario 1, the probability of a potential collision is P (Yes) = 0.0003312, and the probability of no potential collision is P (No) = 0.0209. Therefore, *P(No*) has a higher probability than *P(Yes),* and there will be no collision warning. The collision warning system provided the correct prediction even though there was a new data given to the system that is outside the training data. The collision system will send a message to the driver dashboard stating *(Safe, there is no Potential for Collision).* This is because the distance between the two vehicles meets the road safety requirement, V2 is not accelerating and is moving at a lower speed than the V1.



Figure 6. Vehicle Speed for Scenario 1.



Figure 7. Vehicle Acceleration for Scenario 1.

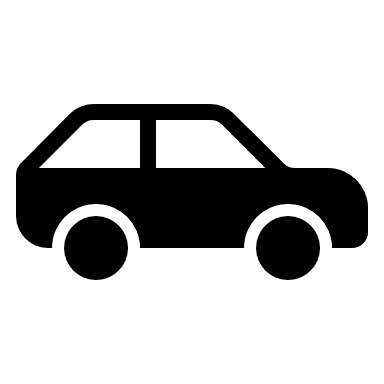
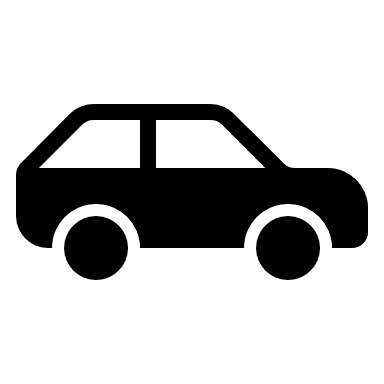
Figure 6 shows the speeds, and Figure 7 shows the acceleration for the two vehicles. A ‘1’ means there is acceleration, and ‘0’ means there is no acceleration. This scenario tested the case where V1 is accelerating and is moving at a high speed (80 MPH), and V2 is moving at a low speed (40 MPH) with no acceleration.



Figure 8.

Figure 8 shows the conditional probabilities of speed, distance, and acceleration for the two vehicles. P(High Speed| Yes), P (Low Speed| Yes), and P (Distance| Yes), etc. P(Yes)= 3.311e-04, P(No) = 0.0209, the conditional probability of the speed for V1 is P(High Speed| Yes) = 0.2105. The conditional probability of the speed for V2 is (LowSpeed |Yes) = 0.070, and so on.

**Scenario 2:**



V1

V2

Figure. 9 an unsafe scenario for V1 and V2.

In figure 9, V1 is not accelerating and is moving within the speed limit of 70 MPH. However, V2 is accelerating and moving at a high speed of 75 MPH. The distance between the two vehicles is close 1.5m/ 5ft.



Figure 10. The probability for scenario 2.

Figure 10 shows there is a potential collision. The value of P(Yes) = 0.0137, and the value of P(No) =0.000729. An alarm will activate, and a message will be sent to the dashboard stating *(Caution, there is a potential for collision).*  This is because the distance between the two vehicles is close, V2 is accelerating and is moving at a higher speed than V1.



Figure 11.



Figure 12.

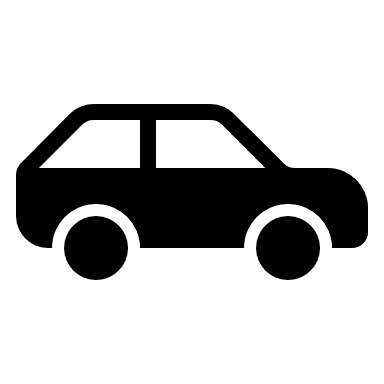
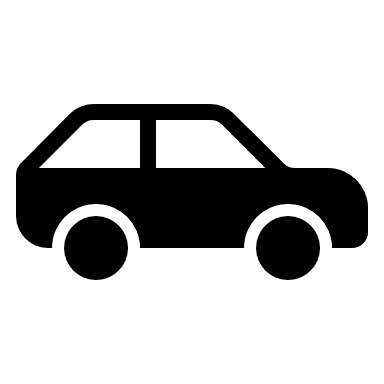
Figure 11 shows the speed, and Figure 12. shows the acceleration for the two vehicles. A ‘1’ means there is acceleration, and ‘0’ means there is no acceleration.



Figure 13

Figure 13 shows the conditional probabilities of speed, distance, and acceleration for the two vehicles. P(High Speed| Yes), P (Speed limit| Yes), and P (Distance| Yes), etc. P(Yes)= 0.5938, P(No) = 0.4063, the conditional probability of the speed for V1 is P(Speed Limit| Yes) = 0.2982. The conditional probability of the speed for V2 is (High Speed |Yes) = 0.5263 and so on.

**Scenario 3:**



V1

V2

Figure 14. A safe scenario for V1 and V2.

In figure 14, V1 & V2 are not accelerating and are moving within the speed limit of 65 MPH, 60MPH. The distance between the two vehicles is safe 3 m/ 10ft.



Figure 15. The probability for scenario 3.

Figure 15 shows there is not a potential collision. The value of P(Yes) = 0.0035, and the value of P(No) = 0.0053.



Figure 17. Vehicle Speed for Scenario 3.



Figure 16

Figure 16 shows the conditional probabilities of speed, distance, and acceleration for the two vehicles. P(High Speed| Yes), P (Low Speed| Yes), and P (Distance| Yes), etc. P(Yes)= 0.5938, P(No) = 0.4063, the conditional probability of the speed for V1 & V2 are P(Speed limit | Yes) = 0.2982.



Figure 17

Figure 17, shows the performance of a classification model using Confusion Matrix Terminology as the following: The accuracy of the collision warning system is 89%, the error rate is 0.10, the false positive rate is 0.27, the precision is 85, sensitivity is 94, and true negative rate is 73%.

**Part 2: Obstacle Avoidance System Using Artificial Neural network and Fail- Safe PLC System.**

A collision avoidance system is one of the automobile safety systems that have been designed to prevent road collisions. The system relies on many different advanced components, such as cameras, sensors, radar, LIDAR, and GPS. This part of the project discusses a real time obstacle avoidance system simulated using the MATLAB ANN -Traintool and implementing a PLC (Siemens S7 1200) and SIMATIC Fail-Safe PLC to control the position of the steering wheel. PLCs are designed to control tasks in robotics and industrial fields. The PLCs control a wide array of applications, such as lighting functions, environmental systems, and chemical processing plants. The SIMATIC safety fail-safe system provides a safety concept in the system. Fail-safe automation systems control processes that obtain immediately a safe state if an unexpected failure occurred. The fail-safe system is connecting directly to the ANNs program. It is responsible for shutting down the ANNs in cases where the ANNs get attacked, a cyberattack for instance, or the output of the ANNs started providing false results that could danger humans or properties. Also, it serves as a back up to the neural network system. The PLC can react to a situation where the neural network stopped sending signals, making the system steadier and more robust. Therefore, the vehicle is capable of detecting and avoiding obstacles by using several systems and controllers. The PLC controls the position of the steering wheel. Four sensors were embedded in the vehicle. These sensors were placed in four different positions on the vehicle: in the left side, right side, front side, and back side of the vehicle. By deploying those sensors around the vehicle, we have covered 360 ֯ angle view around the vehicle. The ANN (Backpropagation) is trained to make decisions on the directions that the vehicle should take. A stepper motor is used to simulate the steering wheel. The PLC controls the position of the stepper motor after receiving signals from the neural network. The PLC uses the motion control functionality to control the position of the stepper motor.

**The components used in this project were:**

* LIDAR &Ultrasonic Sensors
* MATLAB Deep Learning ToolBox (R2019a)
* Arduino Uno R3 Board
* Siemens SIMATIC Fail-Safe 1200 PLC
* Siemens S7 1200 PLC
* High Torque Hybrid Stepper Motor (SY42STH38-0406A).

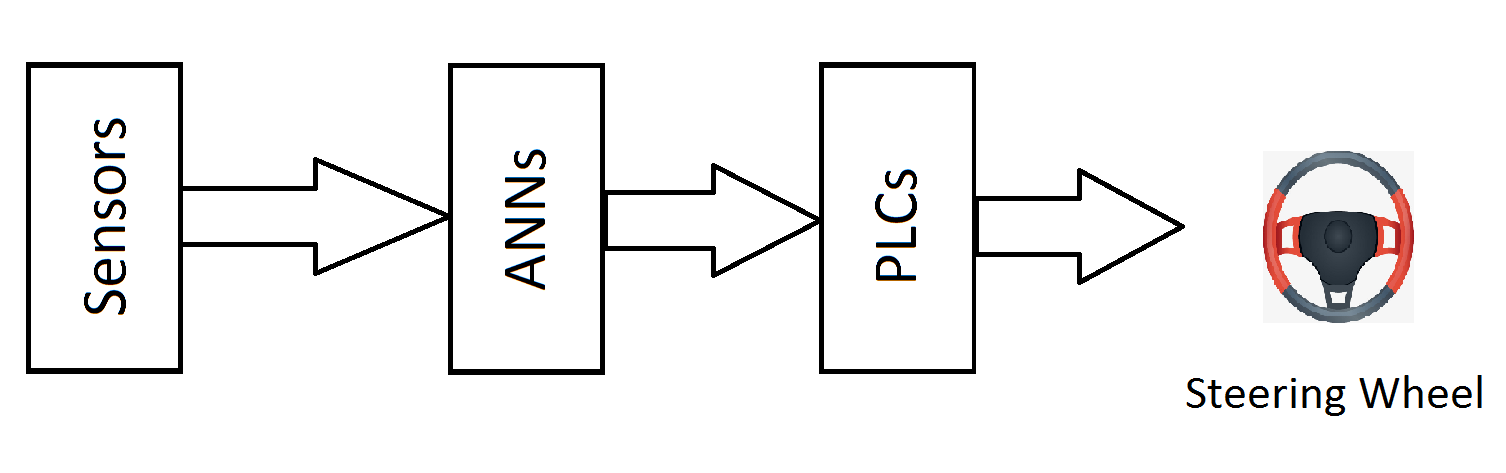


Figure 18 [28]. Proposed Obstacle Avoidance System diagram.

* **Obstacle Avoidance System Operation:**

**Step1:**

The sensors are the inputs of the system. They are responsible for detecting obstacles around the vehicle and measuring the distance between the obstacle and the vehicle. Also, the sensors collect data from the vehicle surrounding. The sensors are connected to the inputs of the neural networks. When the sensors detect obstacles, they will generate signals. The signals will be sent to the neural networks system.

**Step2:**

After training the neural network is done, it learns how to react to different scenarios. The neural network will receive the signals from the sensors and make decisions on which direction the vehicle should move. The decisions are based on the training that the network received. Also, the measurements of the distance or the time period for the vehicle to make a complete maneuver are presented.

**Step 3:**

The outputs of the neural network are connected to the inputs of the fail-safe PLC. The fail-safe PLC reacts to situations when the ANNs stop working properly. There will be an immediate shut down to the ANNs system, which allows the PLC program to operate. The fail-safe program applies the safety functionality. Emergency stop (Estop) and global acknowledgment (ACK) were used on the system. When the emergency stop is unlocked and the ACK is acknowledge the output is set to true. When the start button is activated, the system turns to safe state. If the Estop or stop button is activated, it switches off safely. The PLC controls the positions of the stepper motor and the brakes. The motion control instruction ‘’MC\_Power’’ is used to enable and disable the steering wheel axis Stepper motor. Also, the motion control instruction ‘’ MC\_Home’’ is used to set the axis to home. Motion control instruction ‘’MC\_MoveAbsolute’’ is used in this project to start an absolute movement of the axis. The position of the ’MC\_MoveAbsolute’’ is computed prior the execution. The counter (CTU) is connected to the position of the MC\_MoveAbsolute. The position values of the MC\_MoveAbsolute can be modified by using the counter. When the neural networks send a signal to the PLC, the position of the MC\_MoveAbsolute will be calculated and the PLC executes the position of the MC\_MoveAbsolute. The execution of the position of the axis of the motion control MC\_MoveAbsolute occurs after 5ms from receiving the signals. The Kinematics 2-D formulas, including the Pythagorean Theorem, define the angles, displacement, acceleration, and velocity of the motor axis. To keep the steering wheel at zero angle all the time, an output of the fail-safe PLC is connected to the second MC\_MoveAbsolute that has zero angle or home position. The angle and direction for the vehicle are computed after they receive a signal from the ANNs.

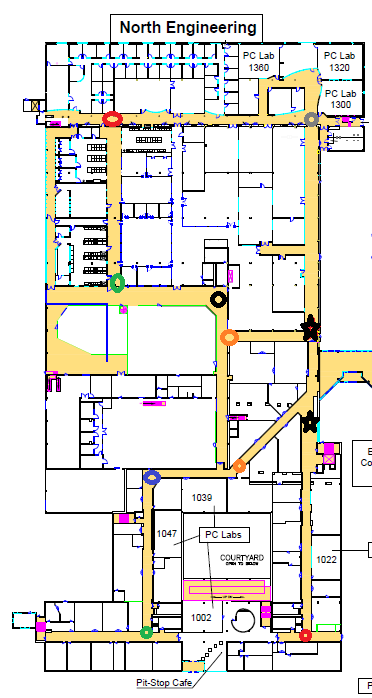
* The following is the dataset table for obstacle avoidance system.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | (X1) Front | (X2) Left | (X3) Right | (X4) Rear | Direction |
| 1 | 0.10 | 0.10 | 0.10 | 0.10 | STOP |
| 2 | 0.20 | 0.20 | 0.20 | 0.20 |
| 3 | 0.30 | 0.30 | 0.30 | 0.30 |
| 4 | 0.40 | 0.40 | 0.40 | 0.40 |
| 5 | 0.50 | 0.50 | 0.50 | 0.50 |
| 6 | 0.60 | 0.60 | 0.60 | 0.60 |
| 7 | 0.70 | 0.70 | 0.70 | 0.70 |
| 8 | 0.80 | 0.80 | 0.80 | 0.80 |
| 9 | 0.90 | 0.90 | 0.90 | 0.90 |
| 10 | 2.13 | 0.79 | 3.40 | 3.50 | R-TURN |
| 11 | 2.13 | 0.97 | 3.45 | 3.44 |
| 12 | 2.43 | 0.76 | 3.39 | 3.42 |
| 13 | 2.65 | 3.55 | 1.82 | 3.45 |
| 14 | 3.40 | 0.42 | 1.03 | 3.46 |
| 15 | 3.51 | 0.48 | 1.06 | 3.42 |
| 16 | 3.41 | 0.61 | 0.91 | 3.53 |
| 17 | 3.56 | 0.67 | 1.28 | 3.15 |
| 18 | 3.21 | 0.79 | 1.09 | 3.62 |
| 19 | 3.95 | 1.52 | 2.13 | 3.85 |
| 20 | 3.20 | 1.03 | 0.42 | 3.15 | L-TURN |
| 21 | 3.19 | 1.06 | 0.48 | 3.55 |
| 22 | 3.45 | 0.91 | 0.48 | 3.20 |
| 23 | 3.52 | 1.28 | 0.67 | 3.20 |
| 24 | 3.12 | 1.09 | 0.79 | 3.45 |
| 25 | 3.45 | 2.13 | 1.52 | 3.65 |
| 26 | 3.17 | 0.68 | 0.68 | 3.85 | FWD |
| 27 | 3.62 | 0.73 | 0.73 | 3.45 |
| 28 | 3.12 | 0.76 | 0.76 | 3.33 |
| 29 | 3.87 | 0.79 | 0.79 | 3.96 |
| 30 | 3.25 | 0.91 | 0.91 | 3.45 |
| 31 | 3.22 | 0.97 | 0.97 | 3.25 |
| 32 | 3.89 | 1.82 | 1.82 | 3.11 |

Table 4

The task is to place the vehicle at three different buildings. The first building is North Engineering at The University of Toledo. The above dataset was collected from the North Engineering building. The dataset was used to train the ANNs. After the training was done, the ANNs was ready to make decisions on the vehicle maneuver. Obstacle avoidance system allows the vehicle to move around the building without any risks. One benefit of applying ANNs algorithm is that can identify new data when it is introduced to the system. Placing the vehicle at second location Nitschke Hall, the data from that building was introduced to the ANNs system for the first time. The results showed that the ANNs was able to detect the obstacles. The third task is to place the vehicle at the back of North Engineering building. The ANNs identified the obstacles. If there are no obstacles in front of the vehicle or the front sensor doesn’t sense any obstacles, the vehicle should move straightforward, which means the steering wheel is at 0 ֯. The vehicle should move on the middle of the hallway. Both left and right sensors should have the same value in order to keep the vehicle moving on the middle of the hallway. The ANNs will decide on which way should the vehicle move if the value of one side sensor exceeded another. When an obstacle is detected, the vehicle should avoid it and then goes back to its track. The vehicle should stop under the following conditions: all four sensors detect obstacles in close range, the distance between the front sensor and the obstacle is very close, or the front and other side sensors detect close obstacles.

**Map of Engineering Complex**



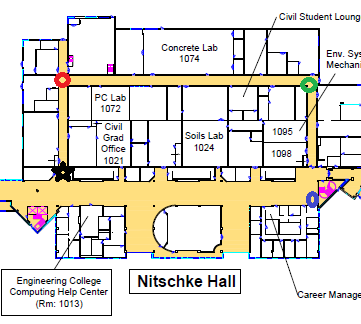


Figure 19.

The MATLAB AAN-Traintool is used to train the network. Figure 20 shows results after the training was done:

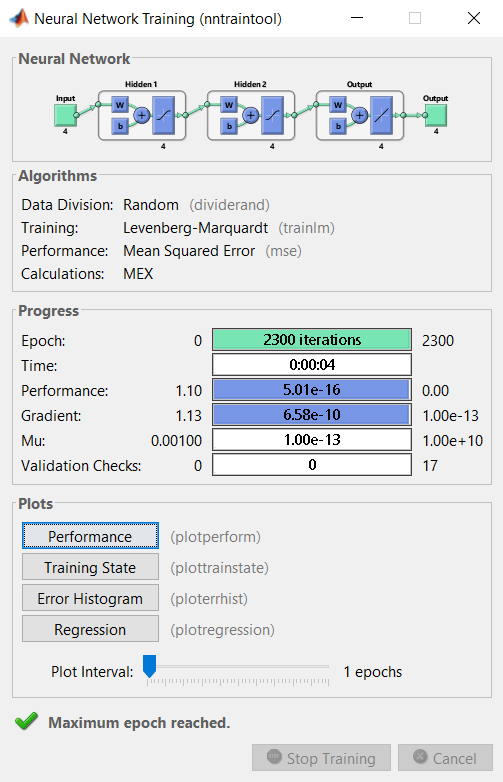


Figure 20.

**Training Process:**

Figure 20 shows the neural network training progress. Divide random (Dividerand) function was applied to divide the data, and Levenberg-Marquardt backpropagation training method also was used with the mean square error performance function. The performance, the magnitude of the gradient of the performance, and number of validation checks are constantly updated during the training. The system used magnitude of the gradient and number of validation checks to complete the training. The gradient decreases when the training performance reaches a minimum level. The training would stop if the magnitude of the gradient below 1e-13. The number of validation checks indicates the number of sequential iterations which the validation performance could not decrease. The training would stop when the number of validations reaches 17 checks. The network consists of 4 inputs, 4 outputs, and two hiding layers. Each hiding layer has four neurons. The weights and the bias are selected randomly. The performance goal is set to zero. The training was stooped after it completed the 2300 epochs. System has 93% accuracy.

**Result**:

The following table shows the output of the trained neural network:

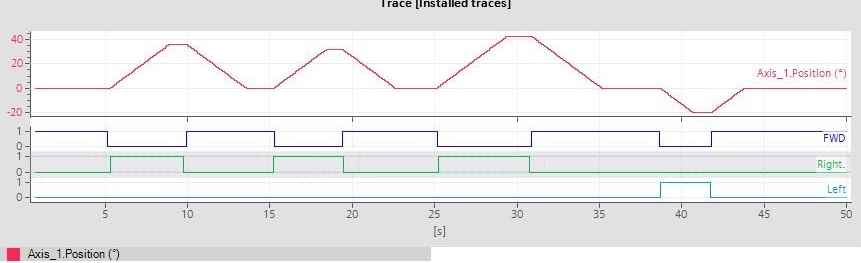


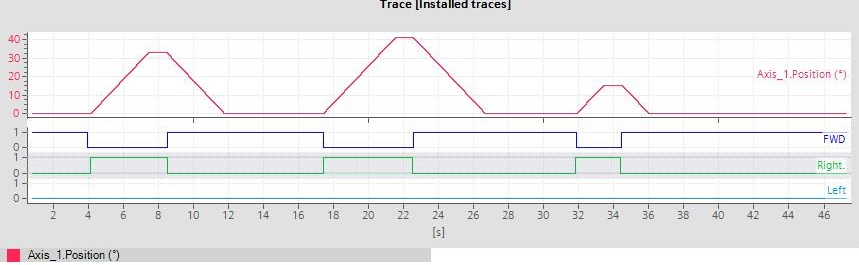
**The training performance**



Figure 21.

Figure 21 shows the training performance. The best training performance is at 5.56x10-14. The validation and the test curves are similar which indicates the overfitting problems have not occurred.



Figure 22

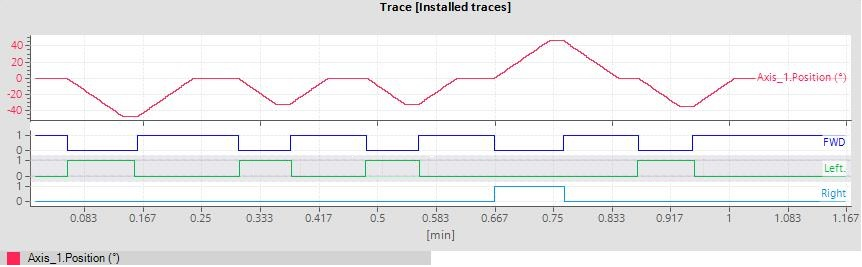
Figure 23

Figure 24.

Figure 22 shows the position of the axis that is set to be at a zero angle. The vehicle should avoid obstacles based on the information that is received from the ANNs. The red curve shows that there are four obstacles. The vehicle successfully managed to avoid the first obstacle the red circle. It’s required 36 degrees (right) to avoid the first obstacle. Then, the position of the axis return home, which is at a zero angle. The second obstacle is at the green dot. The system recognized the second obstacle, and it took 32 degree to the right to completely avoid it. The third obstacle is the blue dot. The system processed the 42 degrees angle to the right. An adjustment was made to the left (-18 degrees) to correct the vehicle path. The system avoided successfully the all the obstacles at the first building.

The second task is to interduce new data to the system and examine the effectiveness of the system. Figure 23 depicts the second path. The system detected the first obstacle which is the red circle. It is required 32.8 degree (right) to avoid the first obstacle. The second obstacle is the green circle. It required 41.18 degrees to the right, and after that 15 degree to right to avoid the third obstacle (blue circle). The system was successfully able to detect and avoid all the obstacles that are at the second building.

The final task is to interduce a different data to the system. Figure 24 shows the maneuver of the vehicle. The system detected the first obstacle which is at the gray circle. It required -47.72 degrees to the left. In order to avoid the second obstacle at the red circle, the system needed -32.22 degrees to the left. The third obstacle is the green circle, and it needed -32.88 degrees to the left. The fourth obstacle (black circle) is required a right turn, so the angle for the right turn is +46.84 degrees. The final obstacle is at the orange circle. It needed -35.75 degrees to the left. Obstacle Avoidance system was successfully able to avoid all the obstacles.

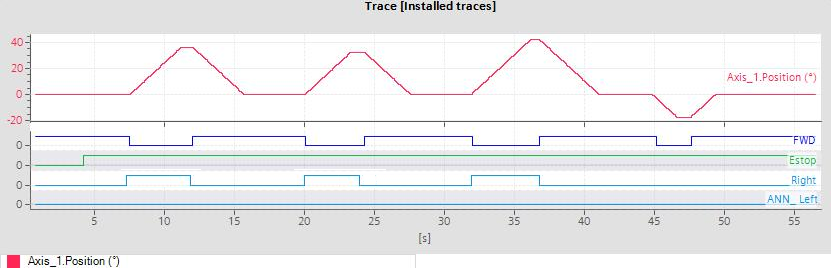
 Figure 25.

Figure 25 shows how the system operates after the E\_STOP was activated. It disabled the ANNs which allowed the PLC program to operate the system.

**Related work**

Tai, Li, and Liu studied “A Deep-Network Solution Towards Model-less Obstacle avoidance.’’ They examined the effectiveness of a hierarchical structure that fuses a convolutional neural network (CNN) with a decision process for an obstacle avoidance system. An advance compact network structure that processes raw depth images as inputs and provide control commands as network outputs. The authors mentioned that the overall accuracy of the system was 80.2% and the class accuracy was 79.7%. The system has a low misclassification. Therefore, there is a low chance for the system to provide opposite decision. The results showed the performance of the system is similar to human reactions under the same circumstances.

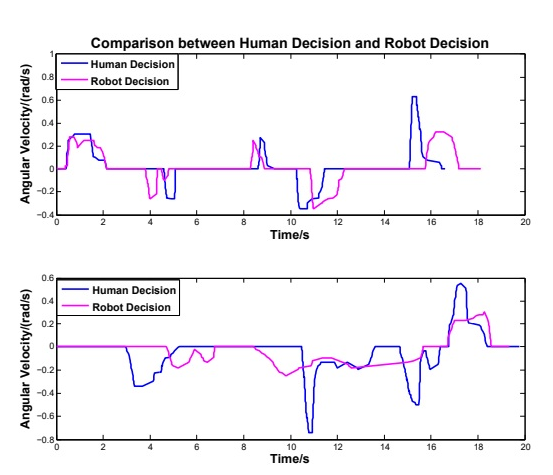
 .

Figure 25.

Figure 25 shows the differences between human decision and robot decision over a time series. The positive values indicate left turns and the negative value means right turns. a sample of 500 points were tested for both human and robot decisions. The mean absolute difference in the second case had 0.1408 rad/s, which indicates that the robot decision super passed the human decision [24].

**Result:**

|  |  |
| --- | --- |
| A Deep-Network Solution Towards Model-less Obstacle avoidance | **Obstacle Avoidance System Using Artificial Neural network and Fail- Safe PLC System.** |
|  |  |
| * Mean absolute difference: 0.1114 rad/s. (Human). * Mean absolute difference: 0.1408 rad/s (Robot) | Mean absolute difference: 0.204 rad/s (PLC) |

Table 4 illustrates the different between the two systems.

**Appendix A**

* **MC\_Power**

The motion control instruction ‘’MC\_Power’’ is for enabling and disabling the axis, and it must be enabled before starting commands for the axis.

* **Home axes (MC\_Home)**

The motion control instruction ‘’ MC\_Home’’ is used to set the axis to home.

* **Position axis absolutely (MC\_MoveAbsolute)**

The motion control instruction ‘’MC\_MoveAbsolute’’ is to start an absolute movement of the axis.

* **Position axis relatively (MC\_MoveRelative)**

The motion control instruction ‘’MC\_MoveRelative’’ is for starting a relative move of the axis.

* **Move axis with velocity set point (MC\_MoveVelocity)**

The motion control instruction ‘’MC\_MoveVelocity’’ is to set the velocity of the axis.

* **Run axis as movement sequence. (MC\_CommandTable)**

The motion control instruction ‘’ **MC\_CommandTable**” makes a combination of multiple induvial axis control commands in one movement sequence.

* **Move axes in jog mode (MC\_MoveJog)**

The motion control instruction “**MC\_MoveJog**” moves the axis in jog mode with constantly at specific velocity.

* **Stop axis (MC\_Halt)**

The motion control instruction “MC\_Halt” is to stop the axis movements.

* **Read motion data of the axis continuously (MC\_ ReadParam)**

The motion control instruction “**MC\_ ReadParam**” enables continuous reading of the motion data and status messages of an axis.

* **Write tag of positioning axis (MC\_WritePrarm)**

The motion control instruction” MC**\_WritePrarm**” in the user program, it enables the writing of tags of the positioning axis.

* **Acknowledge error (MC\_Reset).**

The ‘’MC\_Reset’’ is to acknowledge errors.

Operators decide the command parameters that are input parameters of the motion control instaurations and axis configuration. The output parameters of the instruction provide the operators with information about the status, error of the command.

References:

[1] Techopedia “Autonomous Car.” <https://www.techopedia.com/definition/30056/autonomous-car>

[1] B. Gringer, ‘’History of Autonomous Cars.’’ Title Max, <https://www.titlemax.com/resources/history-of-the-autonomous-car/> (accessed May 8, 2014).

[2] A. Jarvis. “An Introduction to Autonomous Cars: How They Will Improve the Cost, Convenience, and Safety of Driving.” <https://velodynelidar.com/newsroom/an-introduction-to-autonomous-cars-how-they-will-improve-the-cost-convenience-and-safety-of-driving-part-one-of-a-three-part-series/> (accessed September 19, 2018).

[3] B. Beltz. “100+ Car Accident Statistics for 2019.” <https://safer-america.com/car-accident-statistics/> (accessed October 25, 2018).

[4] ‘’MBN.‘’ <https://marketbusinessnews.com/financial-glossary/autonomous-vehicle/>

[5] ‘Union of Concerned Scientists.’ <https://www.ucsusa.org/clean-vehicles/how-self-driving-cars-work#targetText=Self%2Ddriving%20vehicles%20are%20cars,navigate%2C%20and%20drive%20the%20vehicle.> (accessed January 26, 2017).

[6] “Geeks for Geeks.” Machine Learning, <https://www.geeksforgeeks.org/machine-learning/>

[7] L. Tagliaferri ‘’DigitalOcean.’’ An Introduction to Machine Learning, <https://www.digitalocean.com/community/tutorials/an-introduction-to-machine-learning> (accessed September 28, 2017).

[8] G. Edwards ‘’Towards Data Science.’’ Machine Learning | An Introduction <https://towardsdatascience.com/machine-learning-an-introduction-23b84d51e6d0> (accessed November 18, 2018).

[9] R. Shaw. “DataQuest”, The 10 Best Machine Learning Algorithms for Data Science Beginners <https://www.dataquest.io/blog/top-10-machine-learning-algorithms-for-beginners/> (accessed June 26, 2019).

[9] C. Kumar. “Towards data Science.” Machine Learning Types and Algorithms <https://towardsdatascience.com/machine-learning-types-and-algorithms-d8b79545a6ec>

[10] G. Mrukwa. “Netguru.” Types of Machine Learning Algorithms: Supervised and Unsupervised Learning (accessed October 8, 2018).

[11] “Statistics Solutions.” What is Linear Regression? <https://www.statisticssolutions.com/what-is-linear-regression/>

[12] “HackerEarth.” Decision Tree <https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/ml-decision-tree/tutorial/>

[14] T. M. Mitchell, Machine Learning, McGraw-Hill Science, 1997.

[15] ‘’Geeksforgeeks.’’ Confusion Matrix in Machine Learning <https://www.geeksforgeeks.org/confusion-matrix-machine-learning/>

[10] P. Kelly, J. Self, M. Stutz, J. Taylor, W. Freeman, Autoclass Bayesian Classification System, Proceedings of AAAI 1988.

[10] G. Herskovits, A Bayesian method for the induction of probabilistic networks from data. Machine Learning, 1992.

[10] P. Luby, Approximating probabilistic reasoning in Bayesian belief networks is NP-hard, Artificial Intelligence, 1993.

[10] A. P. Laird, N. M. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society, 1977.

[10] P. Pazzani, Beyond independence, Conditions for the optimality of the simple Bayesian classifier, Proceedings of the 13th International Conference on Machine Learning, 1996.

[10] R. O. Hart, Pattern Classification and scene analysis. New York, John Wiley & Sons, 1973.

[10] M. Hirsh, Machine Learning in Information, [http://www.parc.xerox.com/ist~ projects/mlia/](http://www.parc.xerox.com/ist~%20projects/mlia/) (Accessed March 25-27, 1996).

[10] D. Geiger, D. Chickering, Learning Bayesian networks, The combination of knowledge and statistical data, Machine Learning, Kluwer Academic Publishers 1995.

[10] F. V. Jensen, An introduction to Bayesian networks, Springer Verlag, 1996.

[10] T. Joachims, A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization, Carnegie Mellon University, 1996.

[10] K. Lang, learning to filter Netnews, Proceedings of the 12th International Conference on Machine Learning, San Francisco, Morgan Kaufmann Publishers, 1995.

[10] L. Maisel, Probability, statistics, and random processes. Simon and Schuster Tech Outlines, New York, Simon and Schuster, 1971.

[16] J. Larsen, Introduction to Acritical Neural Networks, 1st ed. Lyngby, Denmark, Technical University of Denmark, 1999.

[17] ‘National Institute on Drug Abuse (NIDA).’ <https://www.drugabuse.gov/publications/teaching-packets/power-science/section-i/3-how-neuron-works> (accessed 2007).

[18] S. Sharma, “Toward Data Science.” <https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53> (accessed September 9, 2017).

[14] A. L. Chandra,” Toward Data Science.” <https://towardsdatascience.com/perceptron-learning-algorithm-d5db0deab975> (accessed August 22, 2018).

[15] “Missinglink.ai.” <https://missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right/>

[16] Neural network Toolbox User’s Guide, MathWorks Web Site, 2002.

[17] R. Bliesener, F. Ebel, CLoffler, B. Plagemann, H. Regber, E. v. Terzi, A. Winter, Programmable Logic Control, Festo Didactic GmbH &Co. 2002.

[18] W.Bolton, Programmable Logic Controllers, 4th ed. Burlington, MA, Elsevier Newnes, 2006.

[19] The Technology Objects (TO) of SIMATIC S7-1500 (T), Siemens, 2017.

[20] Dings’ Electrical& Mechanical Co. Ltd, Shanghai, China.

[21] R. Condit, & Dr. D. W. Jones, “Microchip.” Stepping Motors Fundamentals, <http://ww1.microchip.com/downloads/en/AppNotes/00907a.pdf> (accessed 2004).

[21] “SIMATIC S-71200 Functional safety manual.” Siemens AG 2016

[22] J.S.Beggs, Kinematics, Hemisphere Publishing Corporation, 1983.

[23] OpenStax, Rice University, 2019.

[24] L. Tai, S. Li, M. Liu, “A Deep-Network Solution Towards Model-less Obstacle avoidance” (accessed October, 2016)

[24] Monterey Institute for technology and education, The Pythagorean Theorem, <http://www.montereyinstitute.org/courses/DevelopmentalMath/COURSE_TEXT2_RESOURCE/U07_L1_T4_text_final.html>

[25] “Basic mathematics.com.” <https://www.basic-mathematics.com/types-of-angles.html>

[26] “CMV Driving Tips.” <https://www.fmcsa.dot.gov/safety/driver-safety/cmv-driving-tips-following-too-closely> (accessed March 31, 2015).

[27] “Sccppre.” Blue Car Birds Eye View of Car Clipart, <https://www.sccpre.cat/show/oJhJbJ_blue-car-birds-eye-view-of-car-clipart/>

[28] “Clipart Library.” Steering Wheel Transparent Background, <https://www.cleanpng.com/png-steering-wheel-car-cartoon-steering-wheel-155861/>

[29]”drivers education usa.com.”<https://www.driverseducationusa.com/resources/passing/#targetText=Move%20to%20the%20left%20lane,safely%20return%20to%20your%20lane.>

MATLAB Code

% Collision warning System >>> A.Tijani

% count the number of '' one way''

buffer = [alldata(:,1), repmat({}, numel( alldata(:,1) ), 1)]';

C=count (buffer, 'OneWay');

D= sum (C) ; % the total number of OneWay

% Count the number of ''Yes''

buffer = [alldata(:,2), repmat({}, numel( alldata(:,1) ), 1)]' ;

C1= count (buffer,'Yes');

D1=sum(C1); % count how many yes on '' one way''

% count the number of ''No''

C2=count (buffer,'No');

D2=sum(C2) ; %count how many No on ''one way'

% the prob. of P(Yes) is:

P=(D1/D)

% the prob. of P(No) is:

P1=(D2/D)

%%%%%%%%%%% First the Speed %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%% Speed of the Vehicle #1 %%%%%%%%%%%%%%%%%%%%%%%%%%%

% the prob. of High Speed being Yes P(High Speed |Yes) is :

buffer = [alldata(:,3),alldata(:,4), repmat({}, numel( alldata(:,1)), 1)]' ;

C3= count (buffer,'HighSpeed') ; %count the number of the High speed

S=sum (reshape(C3,97,2))';

[The\_Tatal\_Number\_Of\_HighSpeed]=sum (S);

% count the number of Yes on ''High Speed''

c3= count(buffer,'Yes') ;

S1=sum (reshape(c3,97,2))';

s1= sum (S1);

P2 = (s1/D1) ; % the prob. of P(High Speed |Yes).

% The prob. of High Speed being 'No' P(High Speed |No) is :

c4= count(buffer,'No'); % count the number of ''No'' On high Speed

S2=sum (reshape(c4,97,2))';

s2=sum (S2);

P3=(s2/D2); % the prob. of P(High Speed |No).

% % count the number of Speed Limit :

buffer = [alldata(:,5),alldata(:,6), repmat({}, numel( alldata(:,1)), 1)]' ;

[CountSpeedLimt]= count (buffer,'SpeedLimit');

[SumSpeedLimt]= sum (reshape(CountSpeedLimt,97,2))';

[Speed\_Limt]=sum(SumSpeedLimt);

% %%the prob. of Speed Limit being yes P(speed Limit|Yes) is:

[SpeedLimt\_yes]= count(buffer, 'Yes');

[SpeedLimt]= sum(reshape(SpeedLimt\_yes,97,2))';

[Speed\_Limt\_Yes]=sum(SpeedLimt);

[Prob\_SpeedLimit\_Yes]= (Speed\_Limt\_Yes/D1);

% Prob. Speed Limit is being No P(SpeedLimit|No) is:

[SpeedLimit\_No]= count(buffer, 'No');

[SpeedLimitNo]= sum(reshape(SpeedLimit\_No,97,2))';

[Speed\_Limit\_No]=sum(SpeedLimitNo);

[Prob\_Speed\_Limit\_No]= (Speed\_Limit\_No/D2);

% count the number of Low Speed

buffer = [alldata(:,7),alldata(:,8), repmat({}, numel( alldata(:,1)), 1)]' ;

[CountLowSpeed]=count (buffer, 'LowSpeed');

[SumLowSpeed]= sum(reshape(CountLowSpeed,97,2))';

[Low\_Speed]=sum(SumLowSpeed);

%Prob. Low Speed is being Yes P(LowSpeed|Yes) is:

[LowSpeedYes]= count (buffer,'Yes');

[SumLowSpeed]=sum (reshape(LowSpeedYes,97,2))';

[Low\_Speed\_Yes]= sum(SumLowSpeed);

[Prob\_Low\_Speed\_Yes]=(Low\_Speed\_Yes/D1);

%Prob. Low Speed is being No P(Low Speed|No)is:

[LowSpeedNo]= count (buffer,'No');

[SumLowSpeed\_NO]=sum(reshape(LowSpeedNo,97,2))';

[Low\_Speed\_No]=sum(SumLowSpeed\_NO);

% Prob. of the Low Speed being No P(Low Speed|No) is:

[Prob\_Low\_Speed\_No]=(Low\_Speed\_No/D2);

%%%%%%%%%%%% Speed of the vehicle #2 %%%%%%%%%%%%%%%%%

buffer = [alldata(:,11),alldata(:,12), repmat({}, numel( alldata(:,1)), 1)]' ;

[CountTheHighSpeed2]= count (buffer,'HighSpeed') ; %count the number of the High speed2

[SumTheHighSpeed2]=sum (reshape(C3,97,2))';

[The\_Tatal\_Number\_Of\_HighSpeed2]=sum (SumTheHighSpeed2);

% count the number of Yes on ''High Speed2''

[HighSpeed2\_Yes]= count(buffer,'Yes') ;

[HighSpeed2]=sum (reshape(HighSpeed2\_Yes,97,2))';

[High\_Speed2\_Yes]= sum (HighSpeed2);

[Prob\_HighSpeed2\_Yes] = (High\_Speed2\_Yes/D1); % the prob. of P(High Speed2 |Yes).

% The prob. of High Speed2 being 'No' P(High Speed2 |No) is :

[HighSpeed2\_No]= count(buffer,'No'); % count the number of ''No'' On high Speed

[HighSpeed22]=sum (reshape(HighSpeed2\_No,97,2))';

[SumOfHighSpeed2\_No]=sum (S2);

[Prob\_High\_Speed2\_No]=(SumOfHighSpeed2\_No/D2); % the prob. of P(High Speed |No).

% % count the number of Speed Limit 2 :

buffer = [alldata(:,9),alldata(:,10), repmat({}, numel( alldata(:,1)), 1)]' ;

[CountthespeedLimit2]= count (buffer,'SpeedLimit');

[SumtheSpeedLimit2]= sum (reshape(CountthespeedLimit2,97,2))';

[Speed\_Limit2]=sum(SumtheSpeedLimit2);

% %%the prob. of Speed Limit2 being yes P(speed Limit2|Yes) is:

[SpeedLimit2\_yes]= count(buffer, 'Yes');

[SpeedLimit2]= sum(reshape(SpeedLimit2\_yes,97,2))';

[Speed\_Limit2\_Yes]=sum(SpeedLimt);

[Prob\_SpeedLimit2\_Yes]= (Speed\_Limit2\_Yes/D1);

% Prob. Speed Limit2 is being No P(SpeedLimit2|No) is:

[SpeedLimit2\_No]= count(buffer, 'No');

[SpeedLimitNo]= sum(reshape(SpeedLimit2\_No,97,2))';

[Speed\_Limit2\_No]=sum(SpeedLimitNo);

[Prob\_Speed\_Limit2\_No]= (Speed\_Limit2\_No/D2);

% count the number of Low Speed2

buffer = [alldata(:,13),alldata(:,14), repmat({}, numel( alldata(:,1)), 1)]' ;

[CountLowSpeed2]=count (buffer, 'LowSpeed');

[SumLowSpeed2]= sum(reshape(CountLowSpeed2,97,2))';

[Low\_Speed2]=sum(SumLowSpeed2);

%Prob. Low Speed is being Yes P(LowSpeed2|Yes) is:

[LowSpeed2Yes]= count (buffer,'Yes');

[SumLowSpeed22]=sum (reshape(LowSpeed2Yes,97,2))';

[Low\_Speed2\_Yes]= sum(SumLowSpeed22);

[Prob\_Low\_Speed2\_Yes]=(Low\_Speed2\_Yes/D1);

%Prob. Low Speed2 is being No P(Low Speed2|No)is:

[LowSpeedNo2]= count (buffer,'No');

[SumLowSpeed\_NO2]=sum(reshape(LowSpeedNo2,97,2))';

[Low\_Speed\_No2]=sum(SumLowSpeed\_NO2);

% Prob. of the Low Speed2 being No P(Low Speed2|No) is:

[Prob\_Low\_Speed\_No2]=(Low\_Speed\_No2/D2);

% %%%%%%%% The prob. of the distance %%%%%%%%%%%%%%%%%%%

%the prob. of Legal distance being yes P(Legal distance|Yes) is :

buffer = [alldata(:,15),alldata(:,16), repmat({}, numel( alldata(:,1)), 1)]' ;

[Countlegaldistance]= count(buffer,('Legal'));

[Sumlegaldistance]=sum(reshape(Countlegaldistance,97,2))';

[Legal\_distance]=sum(Sumlegaldistance);

% Prob. Of legal distance being Yes P(Legal Distance|Yes) is:

[LegalDistanceYes]=count(buffer,('Yes'));

[SumLegalDistance]=sum(reshape(LegalDistanceYes,97,2))';

[Legal\_Distance\_Yes]=sum(SumLegalDistance);

%Prob. of the Legal Distane being ''Yes'' P(Legal Distance|Yes) is:

[Prob\_Legal\_Distance\_Yes]=(Legal\_Distance\_Yes/D1);

%Prob. of the Legal Distance being No P(Legal Distance |No) is:

[LegalDistanceNo]=count(buffer,('No'));

[SumLegalDistance]=sum(reshape(LegalDistanceNo,97,2))';

[LegalDistance\_No]=sum(SumLegalDistance);

%Prob. of Leagal Distance being noP(Legal Ditance|No) is:

[Prob\_Legal\_Distance\_No]=(LegalDistance\_No/D2);

%Prob. of close distance is: %%%%%%%%%%

buffer = [alldata(:,17),alldata(:,18), repmat({}, numel( alldata(:,1)), 1)]' ;

[CountCloseDistance]= count(buffer,('Close'));

[SumCloseDistance]= sum (reshape(CountCloseDistance,97,2))';

[Close\_Distance]=sum(SumCloseDistance);

%Prob. of the Close Distance being Yes P(Close Distance|Yes) is:

[CountCloseDistanceYes]= count(buffer,('Yes'));

[SumCloseDistanceYes]=sum(reshape(CountCloseDistanceYes,97,2))';

[Close\_Distance\_Yes]=sum(SumCloseDistanceYes);

% Prob. of close distance P(Close Distance|Yes) Is;

[Prob\_Close\_Distance\_Yes]=(Close\_Distance\_Yes/D1);

% %% Prob. of Close Distance being No P(Close Distance|No) is:

[countCloseDistanceNo]= count(buffer,('No'));

[SumCloseDistanceNo]= sum(reshape(countCloseDistanceNo,97,2))';

[CloseDistance\_No]=sum (SumCloseDistanceNo);

%Prob.of the Close Distance P(Close Dist|No) is:

[Prob\_Close\_Dist\_No]=(CloseDistance\_No/D2);

% Prob. of far distance

buffer = [alldata(:,19),alldata(:,20), repmat({}, numel( alldata(:,1)), 1)]' ;

[countFarDistance]=count (buffer,('Far'));

[SumFarDistance]= sum(reshape(countFarDistance,97,2))';

[Far\_Dist]=sum(SumFarDistance);

% Prob. far distance being Yes P(Far|Yes) is:

[CountFarDistYes]= count (buffer,('Yes'));

[SumFarDisYes]=sum(reshape(CountFarDistYes,97,2))';

[Far\_Dist\_Yes]=sum(SumFarDisYes);

[Prob\_Far\_Dist\_Yes]=(Far\_Dist\_Yes/D1);

% % Prob. far distance being No P(Far|No) is:

[CountFarDistance\_No]= count(buffer,('No'));

[SumFarDist\_No]=sum(reshape(CountFarDistance\_No,97,2))';

[Far\_Dist\_No]=sum(SumFarDist\_No);

[Prob\_Far\_Dist\_No]=(Far\_Dist\_No/D2);

%%%%%%%%%%%%%%%%% Acceleration for the vehical #1 %%%%%%%%%%%%

% Prob. of Acceleration1 (True)

buffer = [alldata(:,21),alldata(:,22), repmat({}, numel( alldata(:,1)), 1)]' ;

[CountAcceleration\_Ture]=count(buffer,('T'));

[SumAccelereation\_True]=sum(reshape(CountAcceleration\_Ture,97,2))';

[Acceleration\_True]=sum(SumAccelereation\_True);

% Prob. Acceleration being Yes P(Acceleration|Yes) is:

[countAcceFalse\_Yes]=count(buffer,('Yes'));

[SumAcceFalse\_Yes]=sum(reshape(countAcceFalse\_Yes,97,2))';

[Acceleration\_True\_Yes]=sum(SumAcceFalse\_Yes);

[Prob\_Acce\_True\_Yes]=(Acceleration\_True\_Yes/D1);

% Prob. Acceleration being No P(Acceleration|No) is:

[CountAcce\_True\_No]=count(buffer,('No'));

[SumAcce\_True\_No]=sum(reshape(CountAcce\_True\_No,97,2))';

[Acce\_True\_No]=sum(SumAcce\_True\_No);

[Prob\_Acce\_True\_No]=(Acce\_True\_No/D2);

% Prob. of Acceleration1 (False)

buffer = [alldata(:,23),alldata(:,24), repmat({}, numel( alldata(:,1)), 1)]' ;

[CountAcceleration\_False]=count(buffer,('F'));

[SumAccelereation\_False]=sum(reshape(CountAcceleration\_False,97,2))';

[Acceleration\_False]=sum(SumAccelereation\_False);

% Prob. Acceleration being Yes P(Acceleration|Yes) is:

[countAcceFalse\_Yes]=count(buffer,('Yes'));

[SumAcceFalse\_Yes]=sum(reshape(countAcceFalse\_Yes,97,2))';

[Acceleration\_False\_Yes]=sum(SumAcceFalse\_Yes);

[Prob\_Acce\_False\_Yes]=(Acceleration\_False\_Yes/D1);

% Prob. Acceleration being No P(Acceleration|No) is:

[CountAcce\_False\_No]=count(buffer,('No'));

[SumAcce\_False\_No]=sum(reshape(CountAcce\_False\_No,97,2))';

[Acce\_False\_No]=sum(SumAcce\_False\_No);

[Prob\_Acce\_False\_No]=(Acce\_False\_No/D2);

%%%%%%%%%%Acceleration of the vehicale #2 %%%%%%%%%%%%%%

buffer = [alldata(:,25),alldata(:,26), repmat({}, numel( alldata(:,1)), 1)]' ;

[CountAcceleration2\_Ture]=count(buffer,('T'));

[SumAccelereation2\_True]=sum(reshape(CountAcceleration2\_Ture,97,2))';

[Acceleration2\_True]=sum(SumAccelereation2\_True);

% Prob. Acceleration being Yes P(Acceleration2|Yes) is:

[countAcceTrue2\_Yes]=count(buffer,('Yes'));

[SumAcceTrue2\_Yes]=sum(reshape(countAcceTrue2\_Yes,97,2))';

[Acceleration\_True2\_Yes]=sum(SumAcceTrue2\_Yes);

[Prob\_Acce\_True2\_Yes]=(Acceleration\_True2\_Yes/D1);

% Prob. Acceleration being No P(Acceleration|No) is:

[CountAcce\_True2\_No]=count(buffer,('No'));

[SumAcce\_True2\_No]=sum(reshape(CountAcce\_True2\_No,97,2))';

[Acce\_True2\_No]=sum(SumAcce\_True2\_No);

[Prob\_Acce\_True2\_No]=(Acce\_True2\_No/D2);

% Prob. of Acceleration2 (False)

buffer = [alldata(:,27),alldata(:,28), repmat({}, numel( alldata(:,1)), 1)]' ;

[CountAcceleration2\_False]=count(buffer,('F'));

[SumAccelereation2\_False]=sum(reshape(CountAcceleration2\_False,97,2))';

[Acceleration2\_False]=sum(SumAccelereation2\_False);

% Prob. Acceleration being Yes P(Acceleration|Yes) is:

[countAcceFalse2\_Yes]=count(buffer,('Yes'));

[SumAcceFalse2\_Yes]=sum(reshape(countAcceFalse2\_Yes,97,2))';

[Acceleration\_False2\_Yes]=sum(SumAcceFalse2\_Yes);

[Prob\_Acce\_False2\_Yes]=(Acceleration\_False2\_Yes/D1);

% Prob. Acceleration being No P(Acceleration|No) is:

[CountAcce\_False2\_No]=count(buffer,('No'));

[SumAcce\_False2\_No]=sum(reshape(CountAcce\_False2\_No,97,2))';

[Acce\_False2\_No]=sum(SumAcce\_False2\_No);

[Prob\_Acce\_False2\_No]=(Acce\_False2\_No/D2);

% Potential Collision

% <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

%%%%%%%%%%%%%%%%%%%%%% vehicle #1%%%%%%%%%%%%%%%%%%%%%%%%%

[Speed]=input('Enter the Speed of the Vehicle#1 ');

switch Speed

case {55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71} % Speed Limit >>>>V1

disp('Vehicle#1 is Moving at Speed Limit')

K=(Prob\_SpeedLimit\_Yes);

K1=(Prob\_Speed\_Limit\_No);

%V0=(s3/((S1)+(S2)));

case {72,73,74,75,76,77,78,79,80,81,82,83,84,85} % High Speed

disp('Vehicle#1 is Moving at High Speed')

K=P2;

K1=P3;

% V0=(High\_Speed/((S1)+(S2)));

case {35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54} % Low Speed

disp('Vehicle#1 is Moving at Low Speed')

K= (Prob\_Low\_Speed\_Yes);

K1= (Prob\_Low\_Speed\_No);

% V0=(Low\_Speed/((S1)+(S2)));

end

% %%%%%%%%%%%%% vehicle #2 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[Speed1]=input('Enter the Speed of the Vehicle #2 ');

switch Speed1

case {55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71} %Speed limit of the vehicle#2

disp('Vehicle #2 is Moving at the Speed Limit' )

L=(Prob\_SpeedLimit2\_Yes);

L1=(Prob\_Speed\_Limit2\_No);

case {72,73,74,75,76,77,78,79,80,81,82,83,84,85} % High Speed2

disp('Vehicle#2 is Moving at High Speed')

L=(Prob\_HighSpeed2\_Yes);

L1=(Prob\_High\_Speed2\_No);

case {35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54} %Low Speed

disp('Vehicle#2 is Moving at Low Speed')

L=(Prob\_Low\_Speed2\_Yes);

L1=(Prob\_Low\_Speed\_No2);

end

%

U=input ('Enter the distance in Meters: ');

if U ==3 % legal distnce

disp('The Distance is Legal ''3m''')

%switch U

%case 3

U0=Prob\_Legal\_Distance\_Yes;

U1= Prob\_Legal\_Distance\_No;

% V1=(Legal\_distance/((S1)+(S2)));

elseif U<3 %close Distance

disp('The Distance is Close')

%case {(U) < 3}

U0= Prob\_Close\_Distance\_Yes;

U1= Prob\_Close\_Dist\_No;

% V1=(Close\_Distance/((S1)+(S2)));

% case {3 < (U)}

else U>3 ; % far distance

disp('The Distance is Far')

U0= Prob\_Far\_Dist\_Yes;

U1= Prob\_Far\_Dist\_No;

% V1=(Far\_Dist/((S1)+(S2)));

end

%%%%%%%%%% Acceleration for the Vehicle#1 %%%%%%%%%%%%%%

[Acceleration]= input('Enter the Accelreation of the Vehicale #1 [True=1/False=0]: ');

switch Acceleration

case 1 % Acceleration is true

disp('There is Accerlration')

A0= Prob\_Acce\_True\_Yes;

A1=Prob\_Acce\_True\_No;

%V2=(Acceleration\_True/((S1)+(S2)));

case 0 % Acceleration is false

disp('There is no Accerlration')

A0=Prob\_Acce\_False\_Yes;

A1= Prob\_Acce\_False\_No;

% V2=(Acceleration\_False/((S1)+(S2)));

end

%%%%%%%%%%% Acceleration Of Vehicle #2 %%%%%%%%%%%%%%%%%

[Acceleration1]= input('Enter the Accelreation of the Vehicle #2 [True=1/False=0]: ');

switch Acceleration1

case 1 % Acceleration is true

disp('There is Accerlration')

X= Prob\_Acce\_True2\_Yes;

X1=Prob\_Acce\_True2\_No;

% V2=(Acceleration\_True/((S1)+(S2)));

case 0 % Acceleration is false

disp('There is no Accerlration')

X=Prob\_Acce\_False2\_Yes;

X1= Prob\_Acce\_False2\_No;

% V2=(Acceleration\_False/((S1)+(S2)));

end

% probability of Yes and No>>>>> P22=P(x|yes)\* P(Yes) & P23=P(x|No)\*P(No).

P22=K \*U0 \*A0\*P\*X\*L %\*R22 %

P23=K1\*U1\*A1\*P1\*X1\*L1 %

if (P22)>(P23)

disp('''CAUTION'' There is a Potential for Collision')

clearvars A;

A=arduino ('com3','uno');

for ii=1:10

disp(i);

WriteDigitalPin (A,'D11',0);

pasue (0.05);

WriteDigitalPin (A,'D11',1);

end

configurePin (A, 'D12'); % added on 11-23 @7:40 AM.

time= 50;

while time>0

playTone(A, 'D12', 1200, 1);

writeDigitalPin (A, 'D07',1);

time=time-1;

end

writeDigitalPin (A,'D07',0);

clear A

else

disp('''SAFE'' There Is no Potential for Collision')

% clearvars A; % Arduino code, the green led will be on for 5 seconds

% % and then turns off. % Edited on 11-23-2019 @ 7:55 AM.

A= arduino ('com3','uno');

for ii=1:1;

writeDigitalPin (A,'D4',1);

pause (5);

writeDigitalPin (A, 'D4',0);

end

clear A

%

end

% normalization = the sum equals to one ()

x1 =[P22 P23]; x2=max (x1); x3=min(x1);

x4=(x2/(x2+x3))

x5=(x3/(x2+x3))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Plots %%%%%%%%%%%%%

figure

T= categorical ({'Yes','No'});

T1= [P22 P23];

O=bar (T,T1, 'FaceColor','flat');

O.CData(2,:)=[1 0 0];

O.CData(1,:)=[0 1 0];

% % bar (T,T1, 'g')

% hold on

% bar (T1, [P23],'r')

% % % Q1= plot([P22 P22],'-v'); %Speed V1, Accerleration, and the distance of the vehicle V#1

hold on

% bar (P23)

% Q2= plot([P23 P23], '-o'); % Speed V2, Accerlration and distance of the vehicle V#2

% % % set ([Q1 Q2], 'LineWidth',2 );

% % % legend({'y= P(Yes)','y= P(No)'},'Location','Northwest')

title( 'The Probabilities of a Potential Collision')

% xlabel('Fig.3 The Probabilities of a Potential Collision: Yes/ No')

% xlabel('Figure 2. The probability for scenario 1')

% xlabel('Figure 7. The probability for scenario 2')

ylabel('The Value of The Probabilities')

hold off

figure

T2= categorical ({'V1 Speed','V2 Speed'});

T3= [Speed Speed1];

bar (T2,T3)

hold on

title( 'Plot of Speed for V1&V2')

% xlabel('Figure 3. Vehicle Speed for Scenario 1')

% xlabel('Figure 8. Vehicle Speed for Scenario 2')

ylabel('Speed')

hold off

figure

T8= categorical ({'Acceleration V1','Acceleration V2'});

T9=[Acceleration Acceleration1];

bar (T8,T9)

hold on

title( 'Plot of Acceleration for V1& V2')

ylabel('Acceleration')

% xlabel('Figure 4. Vehicle Acceleration for Scenario 1')

% xlabel('Figure 9. Vehicle Acceleration for Scenario 2')

hold off

% % % Q3= plot ([Speed Acceleration],'O'); % Speed and Acceleration of V1

% hold on

% % % % Q4= plot ([Speed1 Acceleration1],'V'); % Speed and Accerleration of V2.

% % % % set ([Q3 Q4], 'LineWidth', 2);

% % % % axis ([1 2 -15 inf]);

% hold on

% % plot ([U;U])

% % % % legend({'y= V1','y= V2'},'Location','Northeast')

% title( 'Plot of Speed and Acceleration')

% % % % xlabel(' Number of probabilities')

% ylabel('Speed & Acceleration')

% % % % legend({'y= V1','y= V2'},'Location','Northeast')

hold off

figure

T4= categorical ({'P(Speed V1|Yes)','P(Speed V2|Yes)','P(Distance|Yes)','P(AccelerationV1|Yes)','P(AccelerationV2|Yes)'});

T5= [K; L;U0; A0; X];

bar (T4,T5,'r')

% % % plot([K; L;U0; A0; X],'O')

hold on

T6= categorical ({'P(Speed V1|No)','P(Speed V2|No)','P(Distance|No)','P(AccelerationV1|No)','P(AccelerationV2|No)'});

T7=[K1; L1; U1; A1; X1];

bar (T6,T7,'g')

% % % plot([K1; L1; U1; A1; X1],'\*')

% % % title( 'Line plot of probability of potential collision: Yes/ No')

title( 'Conditional Probabilities for Speed, Distance & Acceleration')

% xlabel('Figure 5')

% xlabel('Figure 10') %<<<<<<<<<<<<<<<

ylabel('The value of the probabilities')

% % % legend({'y= Speed,Acceleration&Distance of V1','y= Speed,Acceleration& Distance of V2'},'Location','Southwest')

hold off

**Obstacle Avoidance System Using Artificial Neural network & Fail- Safe PLC System.**

% AHMED TIJANI edited 4-27-2020

close all

clear

clc

p=[[0.10;0.10;0.10;0.10],[0.20;0.20;0.20;0.20],[0.30;0.30;0.30;0.30],[0.40;0.40;0.40;0.40],[0.50;0.50;0.50;0.50],[0.60;0.60;0.60;0.60],[0.70;0.70;0.70;0.70],[0.80;0.80;0.80;0.80],[0.90;0.90;0.90;0.90],[2.13;0.79;3.40;3.40],[2.13;0.97;3.40;3.40],[2.43;0.76;3.40;3.40],[2.65;3.40;1.82;3.40],[3.40;0.42;1.03;3.40],[3.40;0.48;1.06;3.40],[3.40;0.61;0.91;3.40],[3.40;0.67;1.28;3.40],[3.40;0.79;1.09;3.40],[3.40;1.52;2.13;3.40],[3.40;1.03;0.42;3.40],[3.40;1.06;0.48;3.40],[3.40;0.91;0.61;3.40],[3.40;1.28;0.67;3.40],[3.40;1.09;0.79;3.40],[3.40;2.13;1.52;3.40],[3.40;0.68;0.68;3.40],[3.40;0.73;0.73;3.40],[3.40;0.76;0.76;3.40],[3.40;0.79;0.79;3.40],[3.40;0.91;0.91;3.40],[3.40;0.97;0.97;3.40],[3.40;1.80;1.80;3.40]];

t=[[0;0;0;1],[0;0;0;1],[0;0;0;1],[0;0;0;1],[0;0;0;1],[0;0;0;1],[0;0;0;1],[0;0;0;1],[0;0;0;1],[0;1;0;0],[0;1;0;0],[0;1;0;0],[1;0;0;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[1;0;0;0],[1;0;0;0],[1;0;0;0],[1;0;0;0],[1;0;0;0],[1;0;0;0],[0;0;1;0],[0;0;1;0],[0;0;1;0],[0;0;1;0],[0;0;1;0],[0;0;1;0],[0;0;1;0]];

p=[[0.10;0.10;0.10;0.10],[0.20;0.20;0.20;0.20],[0.30;0.30;0.30;0.30],[0.40;0.40;0.40;0.40],[0.50;0.50;0.50;0.50],[0.60;0.60;0.60;0.60],[0.70;0.70;0.70;0.70],[0.80;0.80;0.80;0.80],[0.90;0.90;0.90;0.90],[2.13;0.79;3.40;3.40],[2.13;0.97;3.40;3.40],[2.43;0.76;3.40;3.40],[2.65;3.40;1.82;3.40],[3.40;0.42;1.03;3.40],[3.40;0.48;1.06;3.40],[3.40;0.61;0.91;3.40],[3.40;0.67;1.28;3.40],[3.40;0.79;1.09;3.40],[3.40;1.52;2.13;3.40],[3.40;1.03;0.42;3.40],[3.40;1.06;0.48;3.40],[3.40;0.91;0.61;3.40],[3.40;1.28;0.67;3.40],[3.40;1.09;0.79;3.40],[3.40;2.13;1.52;3.40],[3.40;0.68;0.68;3.40],[3.40;0.73;0.73;3.40],[3.40;0.76;0.76;3.40],[3.40;0.79;0.79;3.40],[3.40;0.91;0.91;3.40],[3.40;0.97;0.97;3.40],[3.40;1.80;1.80;3.40]];

net = feedforwardnet([4 4],'trainlm');

% net = init(net);

net.initFcn = 'initlay';

net.trainParam.epochs = 2300; %Maximum number of epochs to train

% net.trainParam.lr = 0.01;

% net.IW{}=[0];

% net.trainParam.mu=0.004;

% droplayer = dropoutLayer(0.3); % edited in 11-6-2019

net.trainParam.goal =0; % Performance goal

net.trainParam.min\_grad = 1e-13; % Minimum performance gradient

net.trainParam.mu\_dec=0.1; % % mu decrease factor

% net.trainParam.mu=1e+10; % Initial mu

net.trainParam.max\_fail=17; %validation checks. Maximum validation failures

% % net.layerWeights{1,2}; % show the weights added on 11-27

% net.layerWeights{2,1}; % show the layers

% net.trainParam.mu\_inc=10; %mu increase factor

% net.trainParam.mu\_max =1e10; % Maximum mu

[net,tr] = train(net,p,t);

a = sim(net,p);

h0=30; %

format short

p77=a'

p78=a

%actual= t; predicted= p78;

%s0= net.layers{i}.size;

% the end of the training

figure

plot (p,t,'O',p,a,'+')

% ( Actual == predicted )

% Accurcy = ------------------------- \* 100.

% The Length of Acual

Accurcy= (h0)/(length (a))\*100

Erroe\_Rate= (1-Accurcy)/100;

%s1= [90 91 92 93 100]; s2=0:5;

%figure

%plot (Accurcy,s1,'-',Accurcy,s0,'\*' )

pause

% the new data from the second building (for testing only)

% pn=[[1.37;0.88;3.40;3.40],[1.21;1.06;3.40;3.40],[3.05;0.88;3.40;3.40],[3.40;0.57;1.18;3.40],[3.40;0.76;1.37;3.40],[3.40;2.60;3.21;3.40],[3.40;1.18;0.57;3.40],[3.40;1.37;0.76;3.40],[3.40;3.21;2.60;3.40],[3.40;0.88;0.88;3.40],[3.40;1.06;1.06;3.40],[3.40;2.91;2.91;3.40]];

% % new bitch!

% tn=[[0;1;0;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[1;0;0;0],[1;0;0;0],[1;0;0;0],[0;0;1;0],[0;0;1;0],[0;0;1;0]];

% b= sim (net, pn);

% F=b'

% figure

% plot (pn,tn,pn,b,'o')

% % figure

% % plot (tn,b,'-', 'LineWidth',3)

% pause

% new data from the third building (for testing only)

% pn1= [[1.21;3.40;3.40;3.40],[1.44;3.40;0.91;3.40],[1.76;3.40;1.14;3.40],[1.37;1.46;3.40;3.40],[1.52;3.40;1.10;3.40],[3.40;1.34;1.34;3.40],[3.40;0.91;0.91;3.40],[3.40;1.14;1.14;3.40],[3.40;1.46;1.46;3.40],[3.40;1.10;1.10;3.40],[3.40;1.37;1.37;3.40],[3.40;1.03;1.64;3.40],[3.40;0.60;1.21;3.40],[3.40;0.83;1.44;3.40],[3.40;1.15;1.76;3.40],[3.40;0.79;1.40;3.40],[3.40;1.06;1.67;3.40],[3.40;1.64;1.03;3.40],[3.40;1.21;0.60;3.40],[3.40;1.44;0.83;3.40],[3.40;1.76;1.15;3.40],[3.40;1.40;0.79;3.40],[3.40;1.67;1.06;3.40]];

% tn1=[[1;0;0;0],[1;0;0;0],[1;0;0;0],[0;1;0;0],[1;0;0;0],[0;0;1;0],[0;0;1;0],[0;0;1;0],[0;0;1;0],[0;0;1;0],[0;0;1;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[0;1;0;0],[1;0;0;0],[1;0;0;0],[1;0;0;0],[1;0;0;0],[1;0;0;0],[1;0;0;0]];

% d=sim(net,pn1);

% d'

% figure

% plot (pn1,tn1,pn1,d,'O')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Simulation %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% the following is the data from the Engineering buildings.

%left/ right -- Front

x0=[0.42]; m0=[2.13];

x1=[0.48]; m1=[2.80];

x2=[0.61]; m2=[2.43];

x3=[0.67]; m3=[2.65];

x4=[0.81]; m4=[2.68];

x5=[1.52]; m5=[2.35];

x6=[0.57]; m6=[4.43];

x7=[0.76]; m7=[2.86];

x8=[2.60]; m8=[3.50];

x9=[1.03]; m9=[3.26];

x10=[0.83];

x11=[1.15];

x12=[1.06];

x13=[0.60];

%Left &right nn

w0=[0.68 0.68]; y0=[0;1;0;0]; % front to Right

w1=[0.73 0.73]; y1=[1;0;0;0]; %Front to Left

w2=[0.76 0.76]; y2=[0;0;1;0]; % forward only

w3=[0.79 0.79]; y3=[0;0;0;1]; % stop

w4=[0.91 0.91];

w5=[0.97 0.97];

w6=[1.82 1.82];

w7=[0.88 0.88];

w8=[1.06 1.06];

w9=[2.91 2.91];

w10=[1.34 1.34];

w11=[0.91 0.91];

w12=[1.14 1.14];

w13=[1.46 1.46];

w14=[1.10 1.10];

w15=[1.37 1.37];

% all four sensors

H0=[0.10 0.10 0.10 0.10];

H1=[0.20 0.20 0.20 0.20];

H2=[0.30 0.30 0.30 0.30];

H3=[0.40 0.40 0.40 0.40];

H4=[0.50 0.50 0.50 0.50];

H5=[0.60 0.60 0.60 0.60];

H6=[0.70 0.70 0.70 0.70];

H7=[0.80 0.80 0.80 0.80];

H8=[0.90 0.90 0.90 0.90];

%%%%%%%%%%%%%%%%%%%%%% Activate Arduino uno card %%%%%%%%%%%%%%%%%%%%

clearvars a;

a=arduino ('com3','uno');

% a=arduino('com3','uno','Libraries','Ultrasonic');

% ultrasonicObj = ultrasonic(a,'D2','D3');

for i=1:0.5:1000000; % Loop, the code is good for time shown (10k Times).

disp (i)

% for auto testing only

% distance0= readDistance (ultrasonicObj); % front % reading values from the 4 ultrasonic sensors

% distance1= readDistance (ultrasonicObj); %Left

% distance2= readDistance (ultrasonicObj); %Right

% distance3= readDistance (ultrasonicObj); %Back

%for manual testing

distance0= input ('Enter the Value of the Front Sensors Plz: ')

distance1= input ('Enter the Value of the Left Sensors Plz: ')

distance2= input ('Enter the Value of the Right Sensors Plz: ')

distance3= input ('Enter the Value of the Back Sensors Plz: ')

T=[distance0; distance1; distance2; distance3]

z=[distance1 distance2];

% Speed , distance, time formula:

% Distane

% Speed= ---------.

% Time

Speed= 1.34112; %(m/s) % 3 MPH the vehicle is moving @ 3mph

%if data that is collected by the sensors or the distance between the front sensor

%and the obstacle equals m0-m9, the ANNs would simulate the input data.

%The Output of the simulation would be rounded and it

%must equals to y0 or y1 to activate the right turn or left turn conditions.

%if the output of the ANN matches y0 or y1, the pythagorean theorem would be computed.

% we considered 3 MPH is the speed in this test. the times (time0

% -time9) are calculated by the Speed, distance, time formula. we calculated the time for

%that the steering wheel requires to stay on a certain position.

% The Arduino Uno code activates the pin # D2 for the time period that previously computed

%and then switches off, which complete the task.

%D2 is for the right turns, y0 = right turn. D3 is for the left turns,

%y1=left turns.

%%%%%%%%%%%% The front sensor %%%%%%%%%%%%%

if (distance0== m0) % to the right

c0= sim (net,T)

v0= round (c0)

if (v0==y0)

Q0= sqrt (((distance0/2))^2+(distance2)^2); %Pythagorean Theorem

Time0=(Q0/ Speed); % Speed, distance, time formula.

writeDigitalPin (a,'D2',1); % Turn Right % Arduino uno code, turn pin D2 on.

pause (Time0) % the time pin D2 should be on.

writeDigitalPin (a,'D2', 0); % after completing the time (time0), turn the pin D2 off.

else % to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

if (distance0==m1) % to the right

c1= sim (net, T)

v1= round (c1)

if (v1==y0)

Q1= sqrt (((distance0/2))^2+(distance2)^2); %Pythagorean Theorem

Time1=(Q1/ Speed); % Speed, distance, time formula.

writeDigitalPin (a,'D2',1); % Turn Right % Arduino uno code, turn pin D2 on.

pause (Time1) % the time pin D2 should be on.

writeDigitalPin (a,'D2', 0); % after completing the time (time0), turn the pin D2 off.

else % to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

if (distance0==m2) % to the left

c2= sim (net, T)

v2= round (c2)

if (v2==y1) % to the left

Q2= sqrt (((distance0/2))^2+(distance2)^2); %

Time2=(Q2/ Speed);

writeDigitalPin (a,'D3',1); % Turn left

pause (Time2)

writeDigitalPin (a,'D3', 0);

else

writeDigitalPin (a,'D10', 1); % to the Fail-Safe PLC

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

if (distance0==m3) % to the left

c3= sim (net, T)

v3= round (c3)

if (v3==y1)

Q3= sqrt (((distance0/2))^2+(distance2)^2); %

Time3=(Q3/ Speed);

writeDigitalPin (a,'D3',1); % Turn left

pause (Time3)

writeDigitalPin (a,'D3', 0);

else % to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

if (distance0==m4) %to the right

c4= sim (net, T)

v4= round (c4)

if (v4==y1)

Q4= sqrt (((distance0/2))^2+(distance2)^2); %

Time4=(Q4/ Speed);

writeDigitalPin (a,'D2',1); % Turn left

pause (Time4)

writeDigitalPin (a,'D2', 0);

else %(v4~=y1) % to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

if (distance0==m5) %to the right

c5= sim (net, T)

v5= round (c5)

if (v5==y1)

Q5= sqrt (((distance0/2))^2+(distance2)^2); %

Time5=(Q5/ Speed);

writeDigitalPin (a,'D3',1); % Turn left

pause (Time5)

writeDigitalPin (a,'D3', 0);

else % to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

if (distance0==m6) % to the right

c6= sim (net, T)

v6= round (c6)

if (v6==y1)

Q6= sqrt (((distance0/2))^2+(distance2)^2); %

Time6=(Q6/ Speed);

writeDigitalPin (a,'D2',1); % Turn left

pause (Time6)

writeDigitalPin (a,'D2', 0);

else % to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

if (distance0==m7) % to the left

c7= sim (net, T)

v7= round (c7)

if (v7==y1)

Q7= sqrt (((distance0/2))^2+(distance2)^2); %

Time7=(Q7/ Speed);

writeDigitalPin (a,'D3',1); % Turn left

pause (Time7)

writeDigitalPin (a,'D3', 0);

else % to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin

(a,'D10', 0);

end

end

if (distance0==m8) % to the left

c8= sim (net, T)

v8= round (c8)

if (v8==y1)

Q8= sqrt (((distance0/2))^2+(distance2)^2); %

Time8=(Q8/ Speed);

writeDigitalPin (a,'D3',1); % Turn left

pause (Time8)

writeDigitalPin (a,'D3', 0);

else % to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

if (distance0==m9) % to the left

c9= sim (net, T)

v9= round (c9)

if (v9==y1)

Q9= sqrt (((distance0/2))^2+(distance2)^2); %

Time9=(Q9/ Speed);

writeDigitalPin (a,'D3',1); % Turn left

pause (Time9)

writeDigitalPin (a,'D3', 0);

else % to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Left Sensor %%%%%%%%%%

if (distance1==x0)||(distance1==x1)||(distance1==x2)||(distance1==x3)||(distance1==x4)||(distance1==x5)||(distance1==x6)||(distance1==x7)||(distance1==x8)||(distance1==x9)||(distance1==x10)||(distance1==x11)||(distance1==x12)||(distance1==x13)

K10= sim (net, T); % simulate the ANNs

K10

S10=round(K10)

if S10==y0 % Turn Right

while true

while distance1 > distance2 % if the value of the left sensor larger than the value of the right sensor;

writeDigitalPin (a,'D2', 1) % turn right till both values are even.

break

end

if (distance1==distance2)

writeDigitalPin (a,'D2',0)

end

end

% if the simulated ANNs output doesn’t match the actual output, the

% fail-safe program would be active. pin D10 is connected to the PLC.

else % connected to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

%%%%%%%%%%%%%%%%%%%%%%%%% %Right sensor %%%%%%%%%%%%%%%%%%%%%%%%

% this part is for adjustments. the value of the right sensor matches x0-x13, the steering wheel

% should turn to the left until the value on both sensors are the same.

% this would make the vehicle in the middle of the path.

if (distance2==x0)||(distance2==x1)||(distance2==x2)||(distance2==x3)||(distance2==x4)||(distance2==x5)||(distance2==x6)||(distance2==x7)||(distance2==x8)||(distance2==x9)||(distance2==x10)||(distance2==x11)||(distance2==x12)

K11= sim (net, T);

K11

S11=round(K11)

if S11==y1 %turn right

while true

while distance2 > distance1 % if the value of the left sensor larger than the value of the right sensor;

writeDigitalPin (a,'D2', 1) % turn right till both values are even.

break

end

if (distance2==distance1)

writeDigitalPin (a,'D2',0)

end

end

% if the simulated ANNs output doesn’t match the actual output, the

% fail-safe program would be active. pin D10 is connected to the PLC.

else % connected to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

%%%%%%%%%%%% Left & right sensors together %%%%%%%%%%%%% FWD

% to keep the steering wheel straight forward or at 0 degrees, both values of the left

% and the right sensors must be the same. Also, that make the vehicle move in the middle of

%the path.

if (z==w0)|(z==w1)|(z==w2)|(z==w3)|(z==w4)|(z==w5)|(z==w6)|(z==w7)|(z==w8)|(z==w9)|(z==w10)|(z==w11)|(z==w12)|(z==w13)|(z==w14)|(z==w15)

K12= sim (net, T);

K12

S12=round (K12)

if S12==y2

writeDigitalPin (a,'D8',1); % Straightforward

pause (0.7)

writeDigitalPin (a,'D8', 0);

% if the simulated value does not match the actual value y2, activate the

% fail-safe PLC.

else %to the Fail-Safe PLC

writeDigitalPin (a,'D10', 1);

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

%%%%%%%%%%%% All Sensors Together (Stop) %%%%%%%%%%%%%

% if there are obstacle around the vehicle or all sensors sense obstacles,

% the vehicle should stop until the obstacles are removed.

if (T==H0)|(T==H1)|(T==H2)|(T==H3)|(T==H4)|(T==H5)|(T==H6)|(T==H7)|(T==H8)

K13=sim (net, T);

K13

S13=round (K13)

if S13==y3

writeDigitalPin (a,'D9', 1); % Brakes

pause (1)

writeDigitalPin (a,'D9', 0);

% in a situation when the vehicle should stop, but it does not, the

% Fail-safe program will be activated.

else

writeDigitalPin (a,'D10', 1); % To the Fail-Safe PLC

pause (1.5)

writeDigitalPin (a,'D10', 0);

end

end

end

%the end of the program.

Fail-Safe PLC Program