International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

Robot Navigation Using a Neural Network Extracts From the Decision Tree

Amira. A. Elsonbaty
High Institute of Engineering & Technology in New Damietta

Abstract— *The paper will present an artificial neural network model extracts from the decision tree. This paper shows how representing decision trees into a multilayer neural network approach for mobile robot navigation. Two types of models were used in this paper, the first model will design a decision tree architecture that is able to configure itself automatically according to the structure of the mobile robot under consideration, and the second model converts the previous decision tree of robot navigation into an artificial neural network Also here, the kinematics of the differential drive of the wheeled mobile robot were discussed.*

Keywords— *Decision trees, Neural Network, kinematics of differential drive of wheeled mobile robot.*

I. Introduction

Lately, many works were presented about a machine learning, and how it has been applied to help mobile robots to improve their operational capabilities. The robots must be clever with sensitivity, data processing, recognition, learning, reasoning, interpreting, making decision, and action capacities. So, one of the most important issues in the design of an intelligent mobile robot is a navigation problem [1]. Robot Navigation is presented here to show that it is possible to restructure decision tree mapping as a multilayered neural network. To explain kinematics for differential drive robots in current algorithms follows either of two basic approaches, one is decision tree methods whose leaves are labeled with the predicted classification, on the other hand, apply neural network learning algorithms such as the perceptron algorithm and the error backpropagation algorithm that learn through incremental changes of weights in a network consisting of elementary units called neurons. The rest of this paper is structured as follows, section 2 will study the kinematics of differential drive, section 3 will design the decision tree of mobile robot motion, section 4 will show how converting an example decision tree into an equivalent neural network, and section 5 will summarize the conclusions and will give the notes for future research in this area.

II. The Differential Drive Mobile Robot Kinematics

Kinematics is the study of the mathematics of motion without considering the forces that affect the motion, it gives a relationship between the control parameters and the behavior of a system in the space [2]. A differential drive is made of two similar drive wheels on either side of the robot, powered separately, and one or more casters (pivoting wheels) which help maintain the load but that have no active role, figure 1.

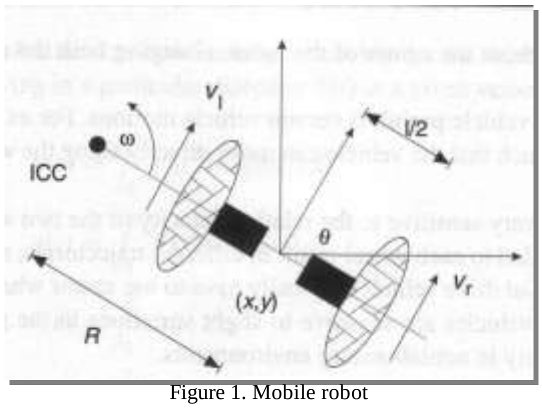

Figure 1. Mobile robot

The kinematic equations for the differential drive mobile robot in world frame are given as:

$$\omega\,(R + l/2) = V_r \tag{1}$$

$$\omega\,(R - l/2) = V_l \tag{2}$$

$$R = \frac{l}{2}\,\frac{V_l + V_r}{V_r - V_l}\,;\quad \omega = \frac{V_r - V_l}{l}; \tag{3}$$

Where,
VR(t) = linear velocity of the right wheel,
VL(t) = linear velocity of the left wheel,
ωR(t) = Angular velocity of the right wheel,
ωL(t) = Angular velocity of the left wheel,
r = Nominal radius of each wheel,
L = Distance between the two wheels,
R = Instantaneous curvature radius of the robot trajectory, relative to the midpoint axis,
ICC = Instantaneous Center of Curvature,
(R – L/2) = Curvature radius of trajectory described by right wheel,
(R + L/2) = Curvature radius of trajectory described by left wheel.

While the velocity of each wheel can vary, for the robot to make rolling motion, the robot must rotate about a point that lies along their common left and right wheel axis. The point that the robot rotates about is known as the ICC - Instantaneous Center of Curvature. They run smoothly and advance some feet, going straight, keep the axle in the middle. The reason is that when two parallel wheels turn, their paths must have different lengths, the outer one having a long distance to cover. This section will discuss the kinematic & a dynamic model of the robot, the definition of required motion speed control, position control, and control law that satisfies the requirements. The differential rotates at the same speed by varying the velocities of the two wheels. Because the rate of rotation ω about the ICC must be the same for both wheels [3]. Mobile robot kinematics aim to description of mechanical behavior of the robot to design and control as show in table [1].

TABLE 1. Robot Navigation

No	Left Wheel	Right Wheel	Robot
1	Stationary	Stationary	Rests stationary
2	Stationary	Forward	Turns counterclockwise pivoting around the left wheel
3	Stationary	Backward	Turns clockwise pivoting around the left wheel
4	Forward	Stationary	Turns clockwise pivoting around the right wheel
5	Forward	Forward	Goes forward
6	Forward	Backward	Spins clockwise in place
7	Backward	Stationary	Turns counterclockwise pivoting around the right wheel
8	Backward	Forward	Spins counterclockwise in place
9	Backward	Backward	Goes backward

III. Differential Drive Robots Decision Tree

Decision trees represent decision procedures for determining the most proper output from a range of application possibilities. Decision trees can be applied to many fields such as pattern recognition, classification, decision support system, expert systems. A range of procedures is available that allow them to summarize a decision procedure automatically from labeled instances [4]. Here be present some rules that can be used to represent knowledge. Propositional If-Then / If-Then-Else rules/Decision tree: These rules contain logical IF part condition which can hold several logical operands like conjunction, disjunction and negation. Classification decision trees can be reduced to If-Then rule sets. From table [1], basically such rules enable making classification decisions by making data splits orthogonal to input data dimensions.

TABLE 2. Rules of differential drive robots.

No	Rule
1	IF (Vr= stationary). AND. (Vl= stationary). THEN rests stationary
2	IF (Vr= forward). AND. (Vl= stationary). THEN. Turns counterclockwise pivoting around the left wheel
3	IF (Vr= backward). AND. (Vl= stationary). THEN. Turns clockwise pivoting around the left wheel
4	IF (Vr= stationary). AND. (Vl= forward). THEN. Turns clockwise pivoting around the right wheel.
5	IF (Vr= forward). AND. (Vl= forward). THEN. Goes forward.
6	IF (Vr= backward). AND. (Vl= forward). THEN. Spins clockwise in place.
7	IF(Vr= stationary).AND.(Vl= backward).THEN. Turns counterclockwise pivoting around the right wheel.
8	IF (Vr= forward) .AND. (Vl= backward). THEN. Spins counterclockwise in place.
9	IF (Vr= backward) .AND. (Vl= backward). THEN. Goes backward.

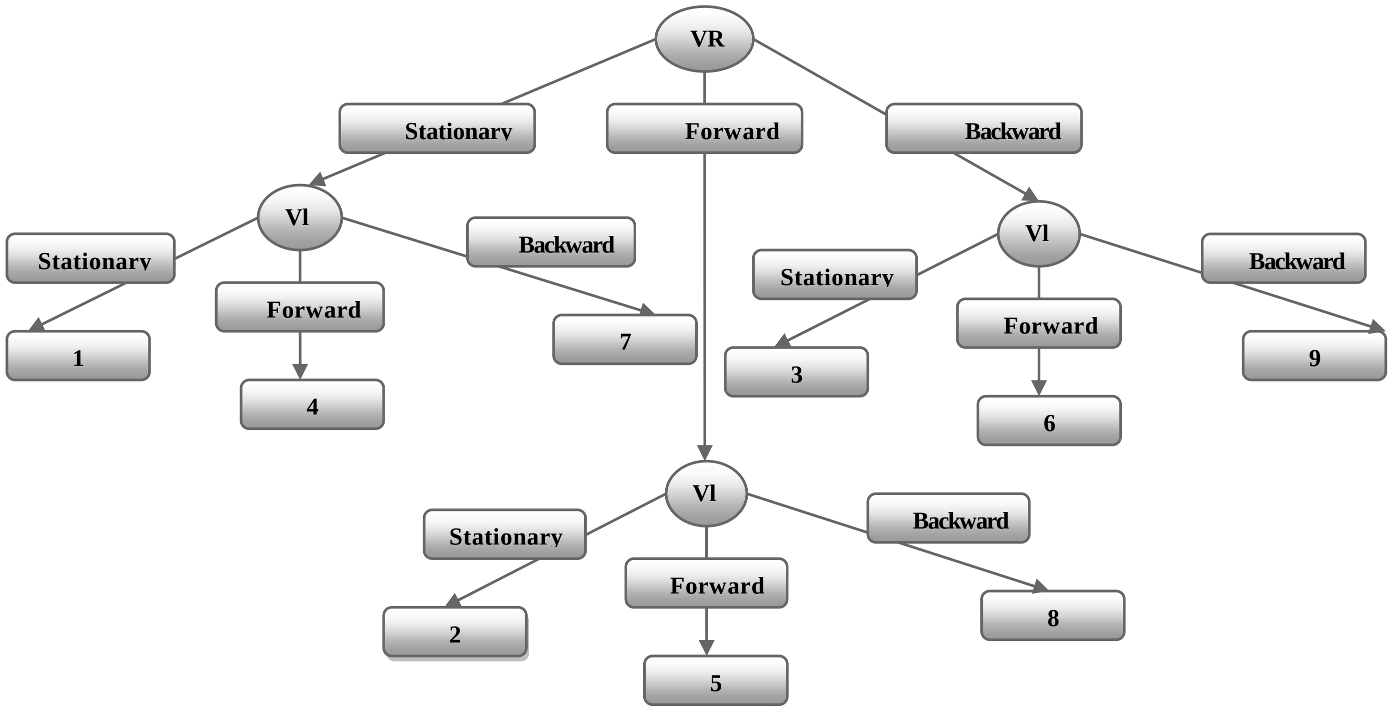

Figure 2. A typical decision tree of robot's navigation

IV. Neural Networks

4.1. Artificial Neural Networks

ANNs are the new computational implements that have established the general utilization in solving many complex real-world problems, due to owing to their ability to represent nonlinear relationships that are difficult to model by means of other computational methods, moreover, neural networks are very good classification and regression models. An activation function takes the neuron input value and produces a value which becomes the output value of the neuron [5]. This value is passed to other neurons in the network Every network has a single input layer and a single output layer. The number of neurons in the input layer equals the number of input variables in the data being handled. The number of neurons in the output layer equals the number of outputs related to each input.

aj	Activation value of unit j
wj,i	Weight on the link from unit j to unit i
ini	Weighted sum of inputs to unit i
ai	The activation value of the unit i (also known as the output value)
g	Activation function

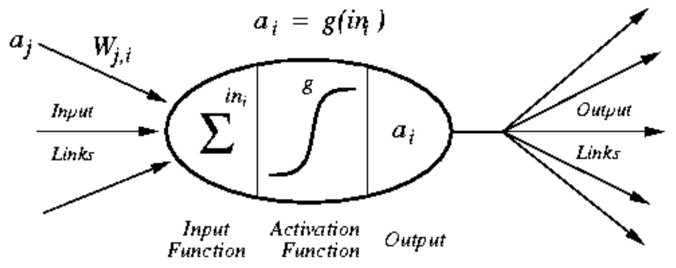

Figure 3. Artificial neural network

The interfacing layer on the input side of the network is called the sensory layer; the one on the output side is the output layer or the motor control layer. All the intermediate layers are called hidden layers. Associated with each unit is a transfer function which determines how that unit's value (or activation) is updated. Typically, the transfer function multiplies each weight projecting to the unit by the activations of the (input) units which the weights project from. The sum of the weight inputs is added to a baseline or bias value to calculate the net input to the unit. Then a very simple activation function is applied to the net input. This function is generally used in the input layer where the inputs to the neural network are delivered into the network unaffected.

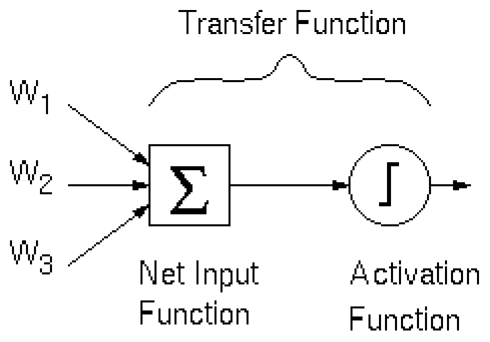

Figure 4. The Transfer Function

Here be defined some common activation functions as follows.

Step(x) =	1 >= t	if x	Else 0
Sin (x) =	+1 >= 0	if x	Else –1
Sigmoid (x) =	1/(1+e-x)		

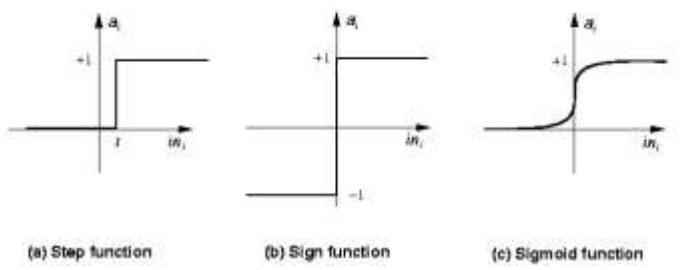

Figure 5. Some common activation functions

4.2. Multilayer Perceptron

A multiple layer artificial network (ANN) building is a sort of neural network that is comprised of at least 3 nodes, which is able of carrying out arbitrary input-output mappings. The nodes of the multilayer perceptron are arranged in layers are the input layer, the output layer, and hidden layers. A multi-layer perceptron has a linear activation function in all its neuron and utilizations backpropagation for its training [6]. Here, the key task is knowing the number of hidden layers and their neurons. In view of the data, draw a predictable decision boundary to isolate the classes, express the decision boundary as a set of lines. Note that the combination of such lines must yield to the decision boundary, the number of selected lines represents the number of hidden neurons in the first hidden layer. To connect the lines created by the previous layer, a new hidden layer is added. Note that a new hidden layer is added each time you need to create connections between the lines in the previously hidden layer, and the number of hidden neurons in each new hidden layer equals the number of connections to be made.

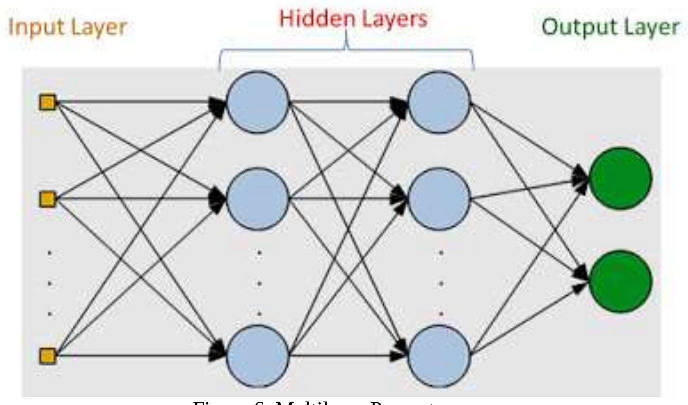

Figure 6. Multilayer Perceptron

4.3. Neural Network of Mobile Robot Navigation

In this strategy, the initial network performs almost as well as the decision tree on which it is based, and the methodology that converts the decision tree into a neural network. The main idea is creating a decision tree and translate it into an equivalent neural system, and after that modify, adjust the network by training it over the equivalent data set for a short period of time. This area will clarify the algorithm which converting a motion control of wheeled robots decision tree into a proportional multilayer perceptron neural network.

4.4. Setting Neural Networks

The next phase of decision tree design involves the construction of nodes that correspond to branches in the tree.

This is accomplished by assigning one node in the hidden layer to each branch such that the node replicates the decision of that branch in the tree. The one before the last phase corresponds to the act of grouping together all branches that are assigned to the same output class. This is achieved by allocating one node each in the output layer to a class, and by connecting with it all nodes in the hidden layer that correspond to the branches for that class. In the final phase, a set of weak edges is superimposed on the network, these edges connect each node in a layer to every node in the previous and the next layer that remain disconnected from it after the previous step. In table 1 notices that the robot can easily turn in any direction, and shows the behavior of a differential drive robot according to the direction of its wheels. There are three interesting cases with these kinds of drives, according the equation [3].

TABLE 3. Three cases with the kinds of drives

Case1:C1	IF (Vl = Vr). THEN. -The robot has forward linear motion in a straight line. -R becomes infinite. -There is effectively no rotation ω is zero.
Case2:C2	IF (Vl = -Vr). THEN. - The robot rotates in place. -R = 0. - The robot has rotated about the midpoint of the wheel axis
Case3:C3	IF (Vl = 0). THEN. -The robot has rotated about the left wheel. -R = 0.5. The Same is true if Vr = 0.

TABLE 4. The behavior of a differential drive robot according to the Right Left Wheel direction of its wheels.NO

NO	Left Wheel	Right Wheel	Robot
1	Stationary	Stationary	C1
2	Stationary	Forward	C3
3	Stationary	Backward	C3
4	Forward	Stationary	C3
5	Forward	Forward	C1
6	Forward	Backward	C2
7	Backward	Stationary	C3
8	Backward	Forward	C2
9	Backward	Backward	C1

By studying table [3, 4], the network generated by the technique has exactly two input nodes (Vr, Vl), three output nodes, and two hidden layers. The ANN size is [2 * 4 * 9 * 3] nodes.

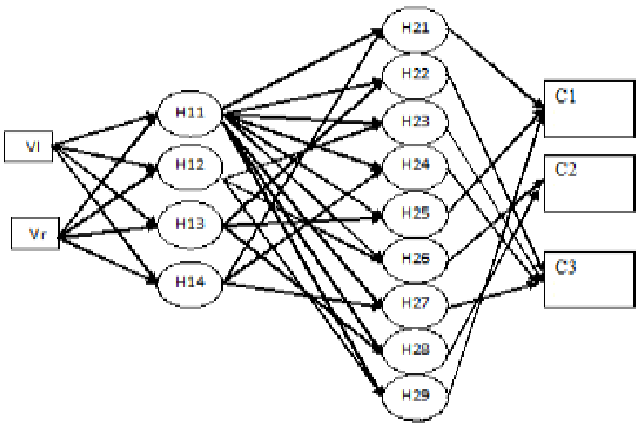

Figure. 7. Mobile Robot Navigation Neural Network

4.5. Training of the Artificial Neural Networks

The learning process of the MLP occurs by continuous adjustment of the weights of the links after each processing. This tuning is based on the error in the output (which is the difference between the expected result and the actual output). This continuous adjustment of the weights is a supervised learning process called 'backpropagation'. The backpropagation algorithm consists of two parts, forward pass, and backward pass. In the forward pass, then expect output corresponding to stated inputs are rated. In the backward pass, partial derivatives of the cost function with respects to the different parameters are propagated back through the network. The process continues until the error is at the lowest worthiness. Backpropagation is the learning algorithmic rule used in neural networks and is a generalization of the least mean squares algorithm used in linear perceptron. Backpropagation need an actual and expected output value for each input value and therefore it is therefore a supervised learning method. Learning is done as the following [7].

1. Input vector x is given to the network
2. Input propagates forward to produce an output y in the output layer
3. The error function is calculated
4. The error is propagated backwards into the network
5. The weights are adjusted accordingly
6. Repeat the process until the final error is minimized

Error = Output perceptron – expected result (4)

Let y, y' be vectors in the R^n

We select an error function E (y, y') which give the difference between outputs y and y' which is given by:

$$E(y, y') = \frac{1}{2}||y - y'||^2 \tag{5}$$

The error function would be given by the square of the Euclidean distance between y and y' as shown below: For n training examples, the average error can would be given by:

$$E = \frac{1}{2n}\sum_x ||(y(x) - y'(x))||^2 \tag{6}$$

The partial derivatives with respects to y and y' would be given by:

$$\frac{\partial E}{y} = y' - y \qquad \frac{\partial E}{y'} = y' - y \tag{7}$$

4.6. Neural Network Simulator

Neural network simulators are software applications that are used to simulate the behavior of artificial or biological neural networks. They focus on one or a limited number of specific types of neural networks. They are typically stand-alone and not intended to produce general neural networks that can be integrated into other software. Simulators usually have some form of built-in visualization to monitor the training process. Some simulators also visualize the physical structure of the neural network [9].

Commonly used artificial neural network simulators include the Stuttgart Neural Network Simulator (SNNS), Emergent and Neural Lab. In the study of biological neural networks, however, simulation software is still the only available approach. In such simulators the physical, biological and chemical properties of neural tissue, as well as the electromagnetic impulses between the neurons are studied. Commonly used biological network simulators include Neuron, GENESIS, NEST and Brian. In the future of the paper, we will use a software simulator technique for training.

V. Conclusion & Future Work

In this paper, we presented an DTs and ANNs to provide the solution of drive differential kinematics problem of a mobile robot. It has been proven that the decision trees can be restructured as four-layer neural nets. A set of mathematical models fails to imitate the behavior of the mobile robot kinematics problem, because Mathematical models depend on the precept that the structural model in advanced, which may head to sub-optimal operation. On the other hand, ANN (artificial neural networks) is based on the data input/output data pairs to define the structure and parameters of the model. Moreover, ANN’s can always be updated to hold better outcomes by delivering new training models as new information becomes available. From the present study it was discovered that the MLP gives the best effects. In this paper the MLP has been offered for the solution of drive differential kinematics problem of a mobile robot. Still, It needs to be observed that the tree-to-network mapping approach is not without any flaws, it causes some restrictions. The paper will try to overcome it in the future, and there are several types of soft computing methods are available which can be used for finding the solution, but this is out of this paper but this technique can be used in the future of the paper.

Refrences

[1] Janglová, D. "Neural Networks in Mobile Robot Motion", *International Journal of Advanced Robotic Systems*, Volume 1 Number 1, pp. 15-22, 2004.

[2] A.V. Chavan, Dr. J. L. Minase, "Design of a differential drive mobile robot platform for use in constrained environments", *Novateur Publications International Journal of Innovations in Engineering Research and Technology*, volume 2, issue 6, June-2015.

[3] http://www.cs.unca.edu/~bruce/Spring05/egm180/Exercise19.doc 6/8/2019

[4] L.Atlas, R. Cole, Y. Muthusamy, A. Lippman, J. Connor, D. Park, M.E Sharkawi, and R.J. Marks II, "A performance comparison of trained multi-layer perceptrons and trained classification trees," *Proc.IEEE*, Vol. 78, No. 10, pp. 1614-1619, October 1990.

[5] Ishwar Sethi,"Entropy nets: From decision trees to neural networks", in *Proceedings of the IEEE* November 1990.

[6] https://kindsonthegenius.com/blog/2018/01/basics-of-multilayer-perceptron-a-simple-explanation-of multilayer-perceptron. html, 7/8/2019.

[7] https://www.cse.unsw.edu.au/~cs9417ml/MLP2/Source.html, 28/8/2019.

[8] https://rdrr.io/cran/RSNNS/f/, 28/8/2019.

[9] https://en.wikipedia.org/wiki/Neural_network_software.10/9/2019.