# INDOOR NAVIGATION OF SMALL VEHICLES USING RECURRENT NEURAL NETWORKS

**Krithika Jaisankar[1], Sai Deshmukh[2], Aishwarya Kanetkar[3]**

[1,2,3]*(MBA-IT, Symbiosis International University,Pune,India)*

## ABSTRACT

*We have tried to analyze the indoor navigation of a small vehicle for which we have used a Data set with time series. The data is collected as Radio Signals. This is an intuitive approach and by understanding it deeply we will make navigation better.*

***Keyword:*** *Recurrent neural network, vanilla neural network, back-propagation.*

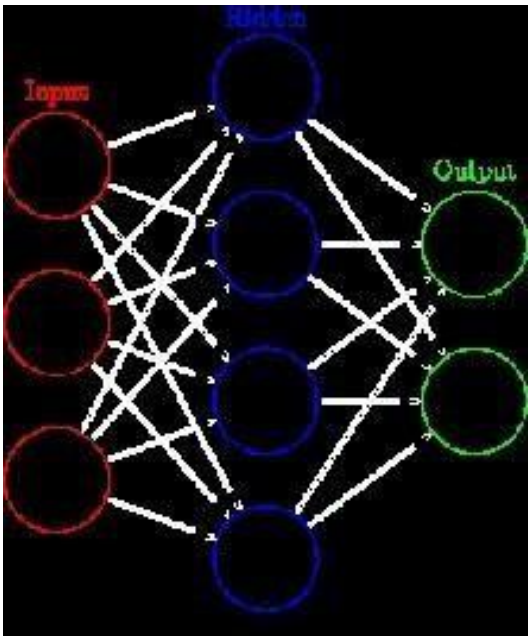
## I.INTRODUCTION:

We have imported the dataset from the UCI repository. We used the dataset of ambient assisted Living. Though, originally the data is collected by predicting movements of a user within an office, we have tried to fit it as per our convenience. As an input, we have Radio Signals. It consists of a time series that is collected from a network of Wireless sensors. Taking one room into consideration, we will have four signal emanators fixed at four corners of the room which will serve as anchors. Then the fifth sensor will be fixed on the small vehicle, say a car. The data is collected at a frequency of 5Hz. We then convert the data to a scale between [0,1], i.e. rescaling. We solely use the data coming from each of the anchors only. Then, we have the target data which is acts like an inference table. It has a labeled class that will indicate whether the vehicle's trajectory will change spatially or not. The entries alternate between -1 and +1. The class +1 relates to change of trajectory and Movement and the class -1relatestocontinuinginthesamepathi.e.preserving.

## II.THEORY OF THE PAPER:

### * Neural Networks*:*

There can be infinite points and hence infinite pathways in a room. So pre-feeding every single path in the system will be difficult. So, we need a model based on which other paths can be predicted. For that, we need to understand the concept of Neural Networks. We have to imagine how the messages to different parts of our body is transmitted as impulses via the nerve cells also called neurons. An impulse enters the nerve cell and then transmits the message to the adjacent nerve cell it is connected to with the help of neural transmitters. We will now consider the endless paths from any source to any destination and the entire system can be visualized as a network of paths that respond in accordance with the radio signal. Also, every nerve cell or unit can have predict if the vehicle will continue in its path or change direction. The neural network systems are always Trained and used for solving difficult problems whose solution cannot be easily found by normal coding or Computation. It learns all by itself i.e. self-learning. These system of neural networks can be in cubical shape or may comprise of different multiple layers. The signal path moves back and forth. The Back-propagation Algorithm is used where the movement or stimulation in the forward direction is used and it accordingly changes the weights present in the front nerve units and is also at times done alongside training where the valid result is known. The recent systems a more free than before as there are more interactions between connections. The most recent dynamic networks are more advanced as they can follow rules and form completely new connections and nerve units. Research is still ongoing on making these systems better such as establishing connections with not just adjacent nerve units but the various layers of processing as well. These networks are based on real values with the axon value ranging between 0 and 1 along with the value of the core. The greatest advantage of these networks is its ability to self-learn. Varied functions like summation where all input values are added up before transmitting it to a new nerve unit.



## III.MODEL CHOICE:

Totally dependent on the representation of data along with its varied applications. The more complex the model, the more challenging it is Algorithm used: There are many algorithms for learning and will work accurately with the training of a fixed dataset and correct parameters.

_ Durability: With appropriate selection of model, cost function and algorithm used for learning, it can make the entire neural network powerful.

*B1* Recurrent Neural Network:

There are various classes under Artificial Neural Networks. The one that fits our problem statement is the Recurrent Neural Network. Here the various connections between the neural units form a directed cycle. An internal network state is created and it thereby exhibits temporal behavior. This internal memory of the networks are used to process the random series of inputs math.

## IV.TYPES OF RECURRENT NETWORKS:

Full or Partial – it focuses on feed-forward which means that this network moves only in one direction with no loops or cycles. It is also compared with relaxation neural networks.

- Elmann Training – They are basic recurrent networks, and mostly use the standard Back propagation
- training.
- The BPTT – also known as Back-propagation through time
- generally can learn further back,for which it must pick depth Equations.

## APPLICATIONS:

Enlisted below are some of the applications of these networks:

- Approximation of functions, regression, time series and modelling. E.g.: radio signals prediction for temporal data like ours.
- Recognition of patterns and sequences that help in classification. It also helps in collision detection etc.
- Processing of data using various technques like filtering, clustering, compression etc.

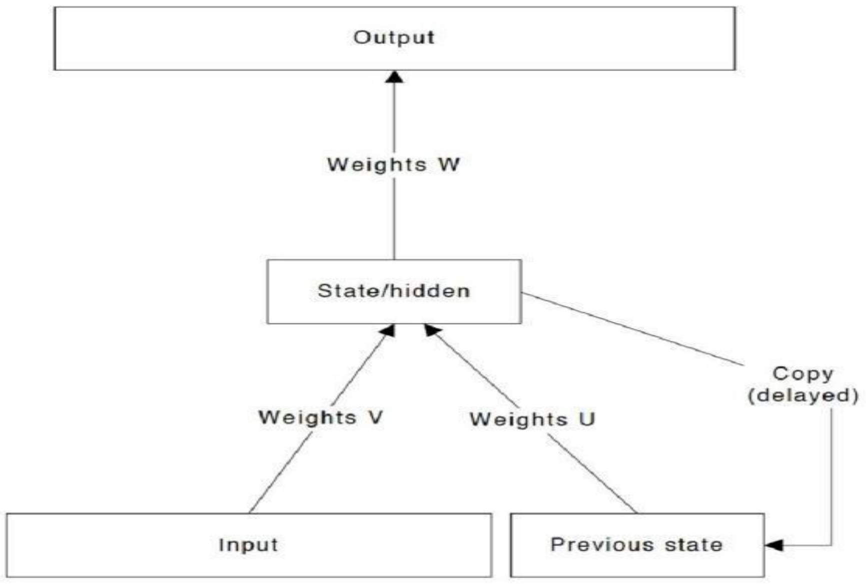
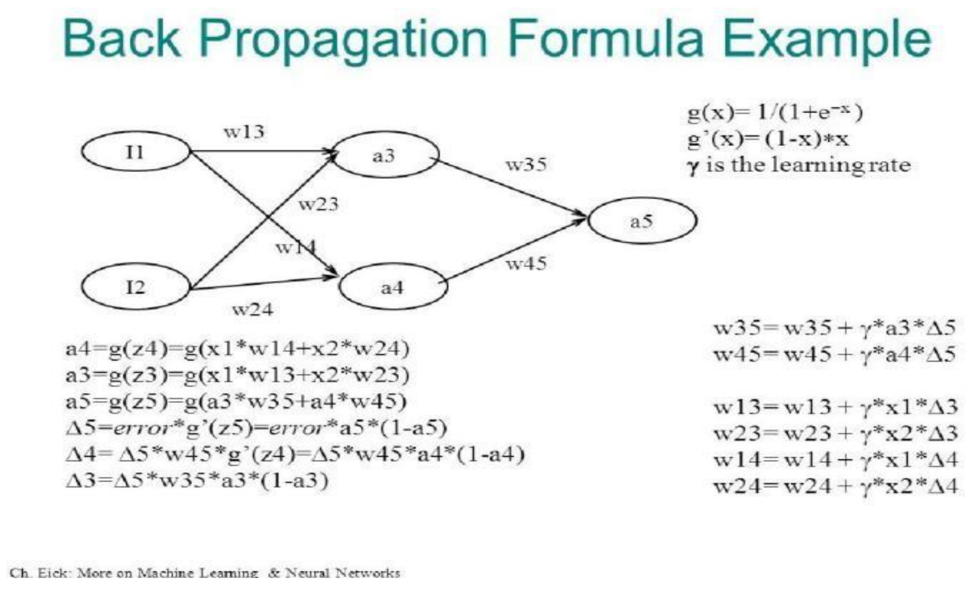


Figure 4: A simple recurrent network.



Back Propagation Formula Example

$g(x) = 1/(1+e^{-x})$
$g'(x) = (1-x)*x$
$\gamma$ is the learning rate

$a4 = g(z4) = g(x1*w14+x2*w24)$
$a3 = g(z3) = g(x1*w13+x2*w23)$
$a5 = g(z5) = g(a3*w35+a4*w45)$
$\Delta5 = error*g'(z5) = error*a5*(1-a5)$
$\Delta4 = \Delta5*w45*g'(z4) = \Delta5*w45*a4*(1-a4)$
$\Delta3 = \Delta5*w35*a3*(1-a3)$

$w35 = w35 + \gamma*a3*\Delta5$
$w45 = w45 + \gamma*a4*\Delta5$

$w13 = w13 + \gamma*x1*\Delta3$
$w23 = w23 + \gamma*x2*\Delta3$
$w14 = w14 + \gamma*x1*\Delta4$
$w24 = w24 + \gamma*x2*\Delta4$

Ch. Eick: More on Machine Learning & Neural Networks

**Plain Vanilla Neural Network:**

The back propagation algorithm is used which is a very common algorithm and it updates the weights after every training pattern. It is also called online back propagation.

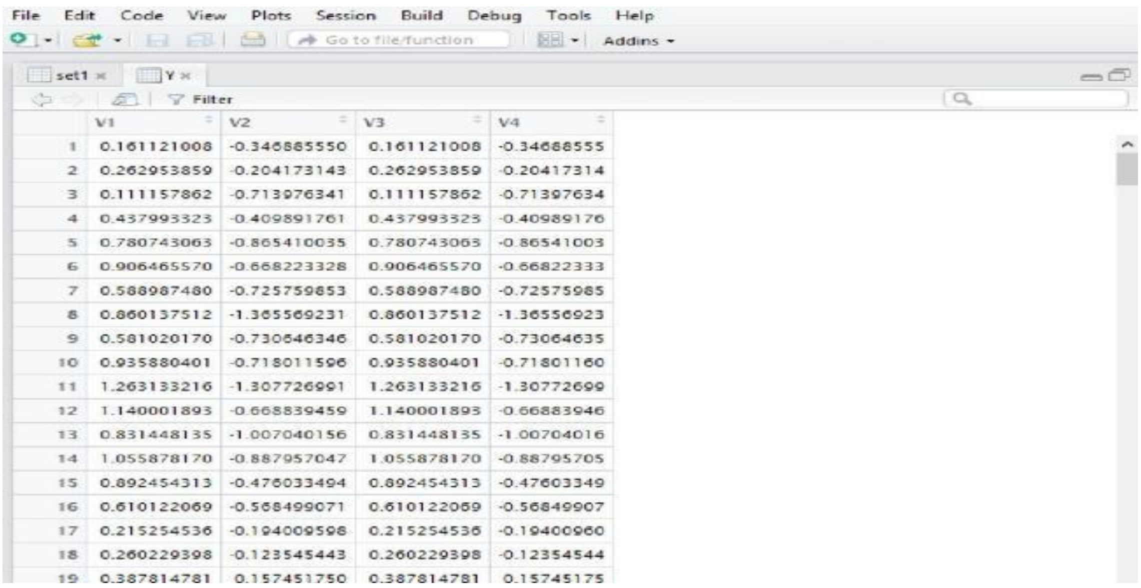
Essentially the back propagation

uses gradient. It generally requires a known output for every input value for calculation of the loss function gradient. It is used in auto encoders. For our model, we have used this algorithm because, we need not have to be aware of the path to the destination always. This algorithm computes gradients for each network layer iteratively. Its main motivation lies in accurate mapping of input to output. It takes input as a series of training values and subsequently producing a series of Weights. We start with a random weight W0. (Our source) accordingly we calculate Wi (destination).

A new function is computed through the old function. We find w1 by applying variable weight W and applying gradient loss function.

OUR METHODOLOGY: We have used the dataset that has records of radio signal strengths. The propagation or movement of our vehicle will have the following phases. Forward movement of Vehicle in the training set's input through the network system for activation of its output. Using training values the vehicle can also have backward movement through the network systems. Then it helps generate the difference between the actual and predicted values(delta) and accordingly gives a visual representation in the form of a plot.Where we multiply the delta value of output and activation value of the input to obtain the weights' gradient. We then subtract the ratio from the gradient. The quality and accuracy of learning is influenced by the ratio obtained. The bigger the ratio, the faster the nerve unit trains, but with a lower ratio the Accuracy of the training is increased.

So by constantly repeating the above-mentioned process we can achieve the desired performance.



| | V1 | V2 | V3 | V4 |
|---|---|---|---|---|
| 1 | 0.161121008 | -0.346885550 | 0.161121008 | -0.34688555 |
| 2 | 0.262953859 | -0.204173143 | 0.262953859 | -0.20417314 |
| 3 | 0.111157862 | -0.713976341 | 0.111157862 | -0.71397634 |
| 4 | 0.437993323 | -0.409891761 | 0.437993323 | -0.40989176 |
| 5 | 0.780743063 | -0.865410035 | 0.780743063 | -0.86541003 |
| 6 | 0.906465570 | -0.668223328 | 0.906465570 | -0.66822333 |
| 7 | 0.588987480 | -0.725759853 | 0.588987480 | -0.72575985 |
| 8 | 0.860137512 | -1.365569231 | 0.860137512 | -1.36556923 |
| 9 | 0.581020170 | -0.730646346 | 0.581020170 | -0.73064635 |
| 10 | 0.935880401 | -0.718011596 | 0.935880401 | -0.71801160 |
| 11 | 1.263133216 | -1.307726991 | 1.263133216 | -1.30772699 |
| 12 | 1.140001893 | -0.668839459 | 1.140001893 | -0.66883946 |
| 13 | 0.831448135 | -1.007040156 | 0.831448135 | -1.00704016 |
| 14 | 1.055878170 | -0.887957047 | 1.055878170 | -0.88795705 |
| 15 | 0.892454313 | -0.476033494 | 0.892454313 | -0.47603349 |
| 16 | 0.610122069 | -0.568499071 | 0.610122069 | -0.56849907 |
| 17 | 0.215254536 | -0.194009598 | 0.215254536 | -0.19400960 |
| 18 | 0.260229398 | -0.123545443 | 0.260229398 | -0.12354544 |
| 19 | 0.387814781 | 0.157451750 | 0.387814781 | 0.15745175 |

## CODE:

```
>install.packages("rnn")
>library(rnn)
>require(rnn)
>set.seed(10)
> f <- 5
> w <- 2*pi*f
> x <- sin(t*w) + rnorm(200, 0, 0.25)
> y <- cos(t*w)
> X <- matrix(X, nrow =300)
> Y <- matrix(Y, nrow =300)
```

The package is installed. The obtained radio signals can be visually represented as a sine wave with necessary modifications in its electric and magnetic field.So here, we have set the frequency as 5.The seed value helps us in prediction.These signals are normalized and the waves are fit in the matrix form for our dataset.

plots

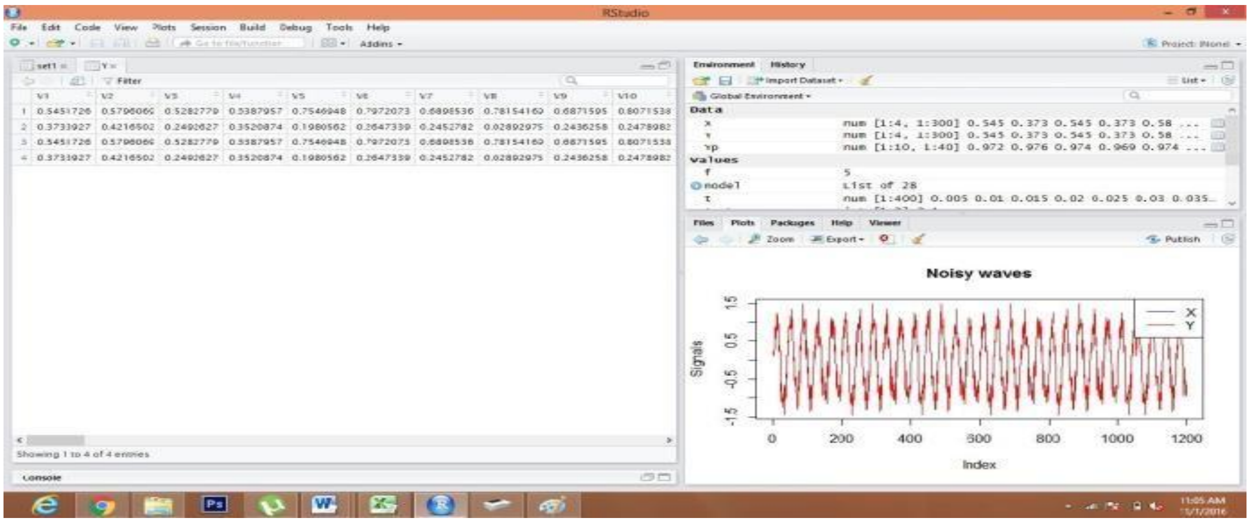
A regular signal has number of disturbances and that gives us the noisy waves. >plot(as.vector(X), col='blue', type='l', ylab = "Signals", main = "Noisy waves") lines(as.vector(Y), col = "red") legend("topright", c("X", "Y"), col = c("blue","red"), lty = c(1,1), lwd = c(1,1))

The dataset is divided into train and test which enables us in comparing the reults and reaching an effective conclusion.

```
>train <- 1:2
>test <- 3:4
>model <- trainr(Y = Y[train,], X = X[train,],learninggrate = 0.05,hiddendim = 16,numepoch =
1500)
>Yp <- predictr(model, X)
>plot(as.vector(t(Y[test,])), col = ’red’, type=’l’, main = "Actual vs predicted: testing set", ylab
= "Y,Yp")
```

Real vs Predicted Values

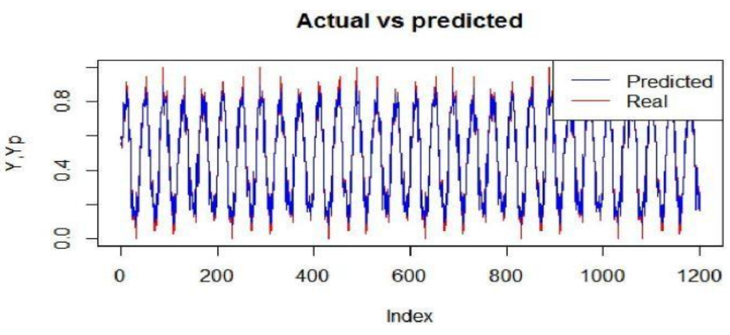
```
plot(as.vector(t(Y)), col = ’red’, type = ’l’, main = "Actual vs predicted", ylab =
"Y,Yp")lines(as.vector(t(Yp)),
type = ’l’, col = ’blue’)
legend("topright", c("Predicted", "Real"), col = c("blue","red"), lty = c(1,1), lwd = c(1,1))
```



|  | V1 | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | V10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.5451726 | 0.5796069 | 0.5282779 | 0.5387957 | 0.7546948 | 0.7972073 | 0.6898536 | 0.78154169 | 0.6871595 | 0.8071538 |
| 2 | 0.3733927 | 0.4216502 | 0.2492627 | 0.3520874 | 0.1980562 | 0.2647339 | 0.2452782 | 0.02892975 | 0.2436258 | 0.2478982 |
| 3 | 0.5451726 | 0.5796066 | 0.5282779 | 0.5387957 | 0.7546948 | 0.7972073 | 0.6898536 | 0.78154169 | 0.6871595 | 0.8071538 |
| 4 | 0.3733927 | 0.4216502 | 0.2492627 | 0.3520874 | 0.1980562 | 0.2647339 | 0.2452782 | 0.02892975 | 0.2436258 | 0.2478982 |

Showing 1 to 4 of 4 entries

```
Data
  X        num [1:4, 1:300] 0.545 0.373 0.545 0.373 0.58 ...
  Y        num [1:4, 1:300] 0.545 0.373 0.545 0.373 0.58 ...
  Yp       num [1:10, 1:40] 0.972 0.976 0.974 0.969 0.974 ...
Values
  f        5
  model    List of 28
  t        num [1:400] 0.005 0.01 0.015 0.02 0.025 0.03 0.035...
```



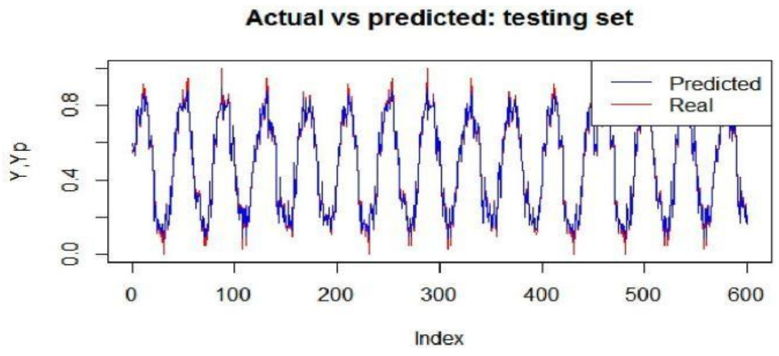
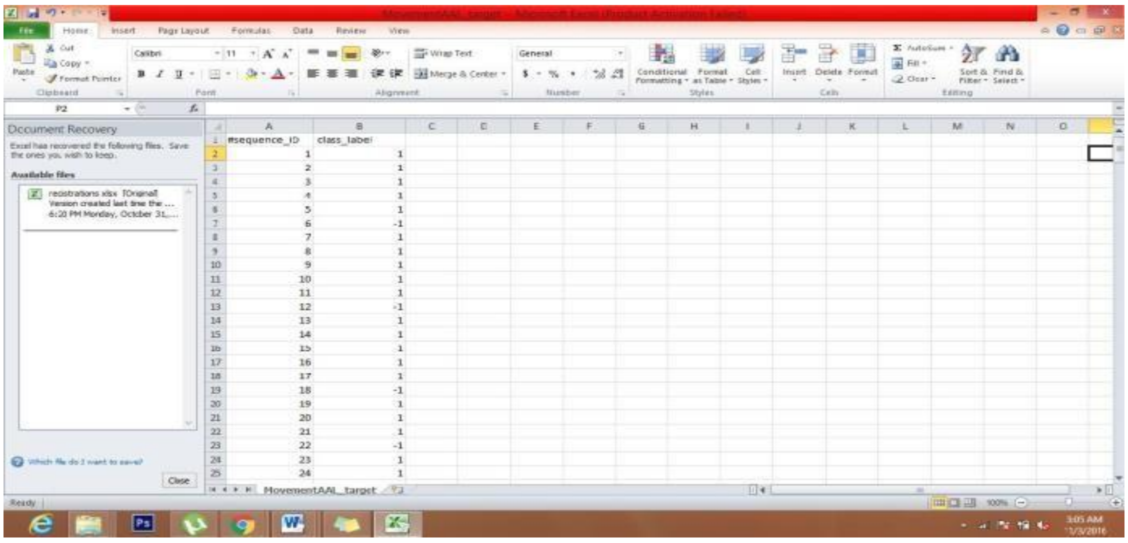


FIG: comparison of testing and training data



| | A | B |
|---|---|---|
| 1 | #sequence_ID | class_label |
| 2 | 1 | 1 |
| 3 | 2 | 1 |
| 4 | 3 | 1 |
| 5 | 4 | 1 |
| 6 | 5 | 1 |
| 7 | 6 | -1 |
| 8 | 7 | 1 |
| 9 | 8 | 1 |
| 10 | 9 | 1 |
| 11 | 10 | 1 |
| 12 | 11 | 1 |
| 13 | 12 | -1 |
| 14 | 13 | 1 |
| 15 | 14 | 1 |
| 16 | 15 | 1 |
| 17 | 16 | 1 |
| 18 | 17 | 1 |
| 19 | 18 | -1 |
| 20 | 19 | 1 |
| 21 | 20 | 1 |
| 22 | 21 | 1 |
| 23 | 22 | -1 |
| 24 | 23 | 1 |
| 25 | 24 | 1 |

Document Recovery

Excel has recovered the following files. Save the ones you wish to keep.

Available files

reoistrations xlsx [Original] Version created last time the ... 6:20 PM Monday, October 31,...

MovementAAL_target

Fig: target which helps in preserving environment.

## Target dataset:

So, suppose the vehicle wants to travel from A to B. There can be several paths to reach the destination B.B ut our vehicle will solely move in the direction where the signal strength is highest. The various paths are pre-fed in the vehicle s memory. So what happens if the path to reach B is north, and the signal closer to the vehicle is in the direction of south? Well, then according to our analysis, it will check the pre-fed path. So, if there is a change in the direction, it is denotedby+1andpreservingofthedirectionisdenotedby-1.0

## V.CONCLUSION:

There are a number of models that can be fitted for this problem statement like the LSTM-Long Short term Memory which is also a type of Recurrent Neural Network. We have used the Plain Vanilla Recurrent Model as a simple intuitive approach for the indoor navigation of a small vehicle and thus this model can further be improvised using in-set technologies.

## REFERENCES AND FOOTNOTES:

1) Slawomir Grzonka, Giorgio Grisetti, Wolfram Burgard - 'Autonomous Indoors Navigation using a small size Quadrato
2) Davide Bacciu, Paolo Barsocchi, Stefano Chessa, Claudio Gallicchio, Alessio Micheli - 'An experimental characterization of reservoir computing in ambient assisted livingapplications
3) Claudio Gallicchio, alessio micheli, Paolo Barsocchi, stefano Chessa-'User Movements Forecasting by Reservoir Computing using signal streams produced by mote-class sensors'
4) Xiiaoyang Wen, Wennyuan Tao, Chung-Ming Ow and Zhenjiiang Pan-'Article on the Dynamic RSS Feedbacks of Indoor Finger-printing Databases for Localization Reliability Improvement'
5) A.K.Rigler,J.M.Irvine, T.P.Vogl-'Rescaling of variables in back-propogation learning'
6) V.Kandetsky, I Antonyuk-'Signal Processing in defect detection using back-propogation neural networks'

Article

Ralf Solomon, J.L.van.Hemman-'Article on accelerating backpropogation through self-adaptation'