Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Neural Network Controllers for Trajectory control

Mobile Autonomous robots

Mozzam Motiwala

University of Hamburg
Faculty of Mathematics, Informatics and Natural Sciences
Department of Informatics
Technical Aspects of Multimodal Systems

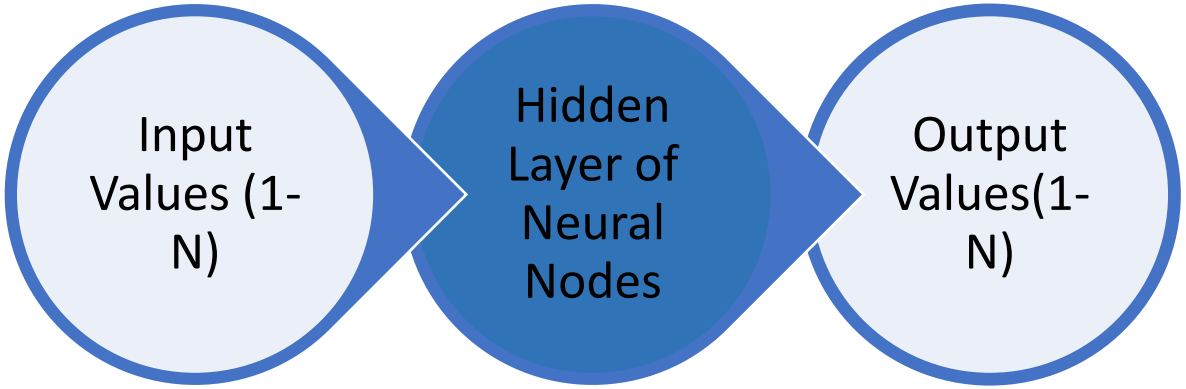
Motivation

- Getting an insight of Neural Network.
- Outlining basic expectation from a Neural Network system.
- Explore few different design of Neural Network Controllers and compare their results with classical Non-linear Non-adaptive Controllers.

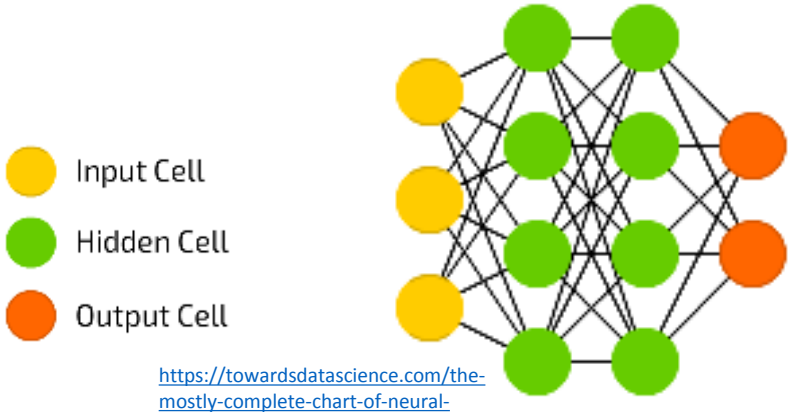
Content

Neural Network in a Nutshell

- The basic expectation



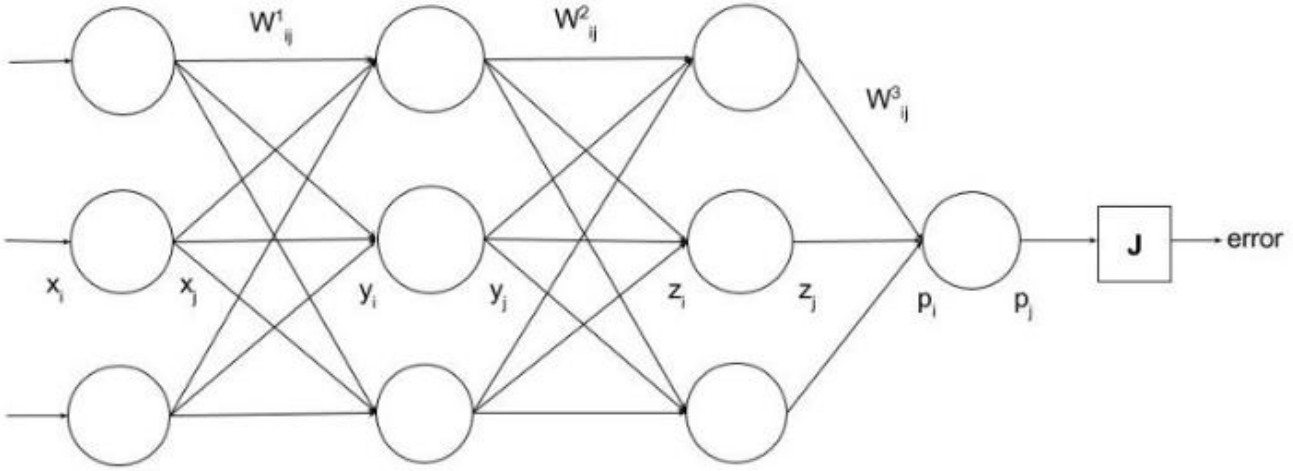
Function Approximation



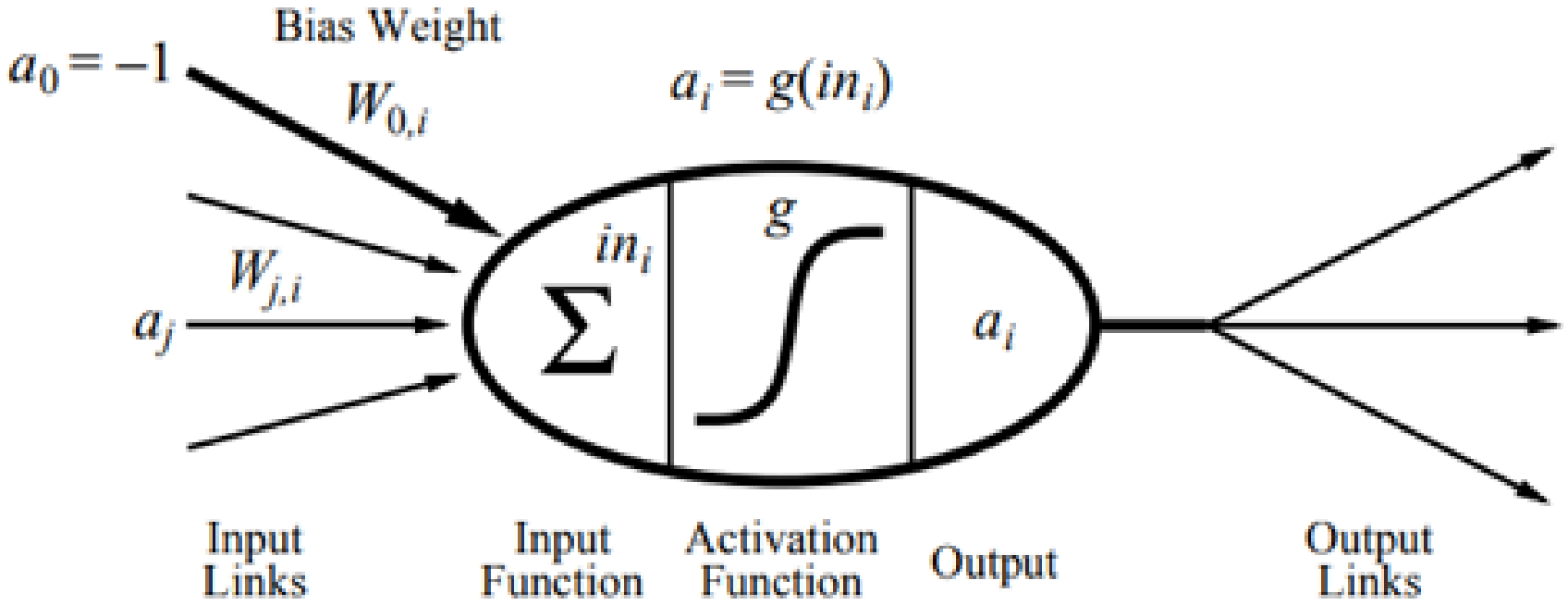
https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464

Learning:
Takes Input and output -> Identifies relationship between the 2

Neuron and Networks

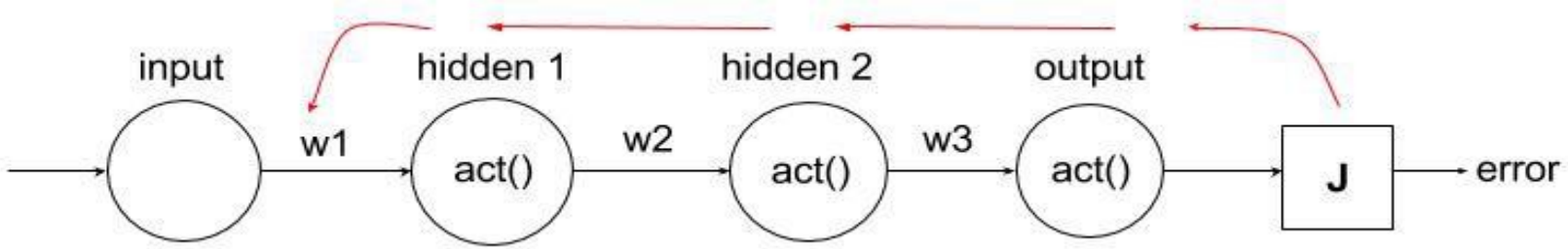


https://ayearofai.com/rohan-lenny-1-neural-networks-the-backpropagation-algorithm-explained-abf4609d4f9d



http://aima.eecs.berkeley.edu/slides-pdf/chapter20b.pdf

Backpropagation



$$error = J = \frac{1}{2}(\vec{p} - \vec{a})^2$$

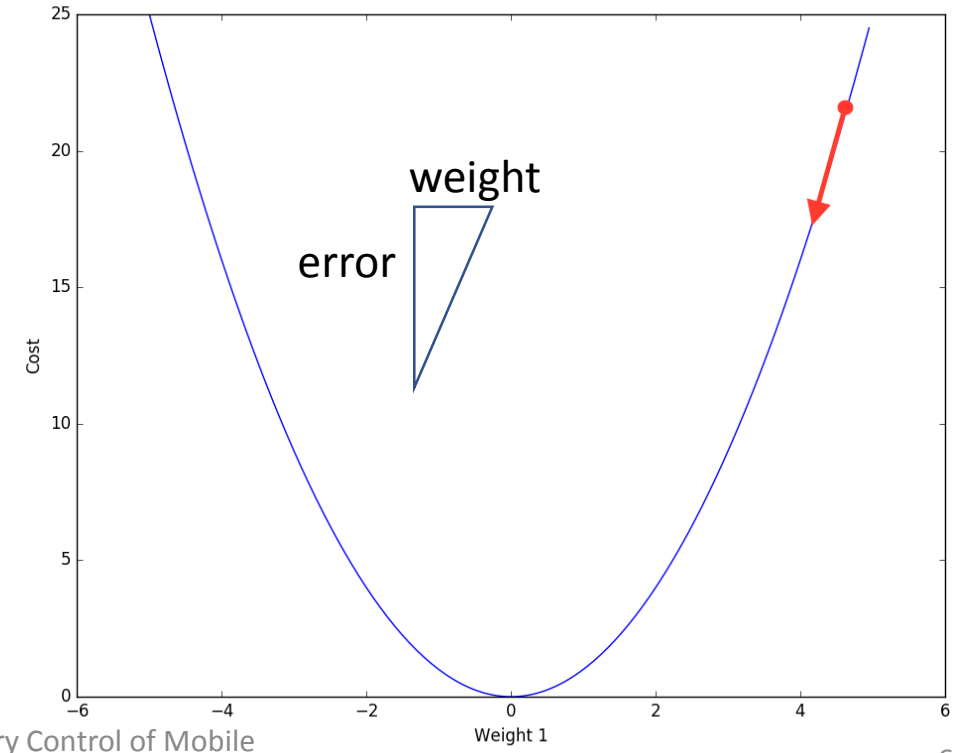
vector **p** = predicted output,
vector **a** = actual output

$$\frac{\partial error}{\partial w1} = \frac{\partial error}{\partial output} * \frac{\partial output}{\partial hidden2} * \frac{\partial hidden2}{\partial hidden1} * \frac{\partial hidden1}{\partial w1}$$

$$output = act(w3 * hidden2)$$

$$hidden2 = act(w2 * hidden1)$$

$$hidden1 = act(w1 * input)$$



https://ayearofai.com/rohan-lenny-1-neural-networks-the-backpropagation-algorithm-explained-abf4609d4f9d

Optimization algorithm

https://skymind.ai/wiki/neural-network

Some examples of optimization algorithms include:

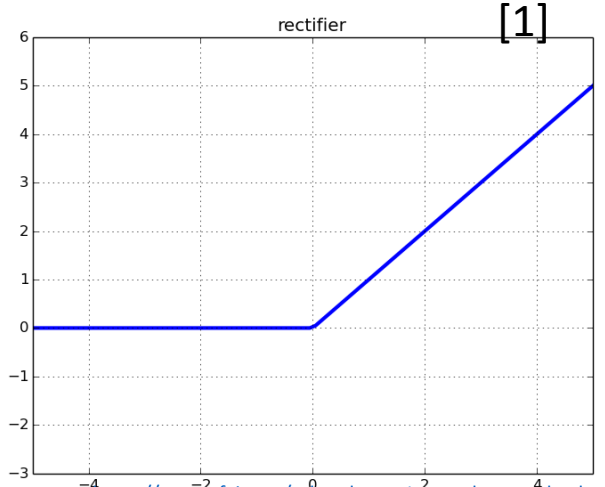
- ADADELTA
- ADAGRAD
- ADAM
- NESTEROVS
- NONE
- RMSPROP
- SGD
- CONJUGATE GRADIENT
- HESSIAN FREE
- LBFGS
- LINE GRADIENT DESCENT

Activation Function

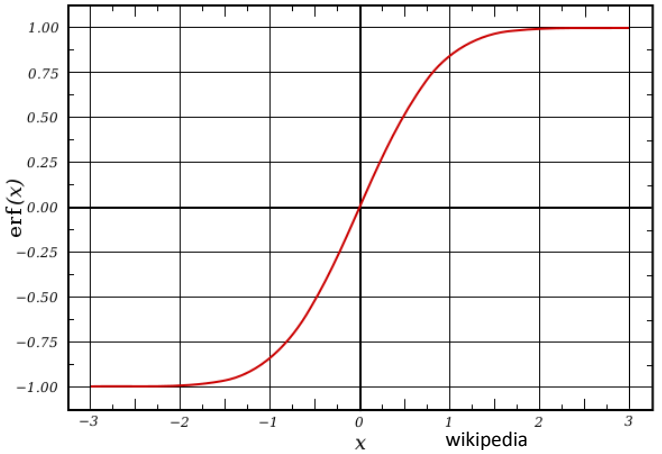
Some examples include:

- CUBE
- ELU
- HARDSIGMOID
- HARDTANH
- IDENTITY
- LEAKYRELU
- RATIONALTANH
- RELU
- RRELU
- SIGMOID
- SOFTMAX
- SOFTPLUS
- SOFTSIGN
- TANH

[1]



https://ayearofai.com/rohan-lenny-1-neural-networks-the-backpropagation-algorithm-explained-abf4609d4f9d



Content

1. Neural Network in a Nutshell
2. Robot Trajectory Control : Problem Statement
3. Neural Network controller
   - A Basic Model (on Robot Manipulator)
   - Model Based on Quadrotors(Focus Model)
   - Additional Model on Mobile wheeled Robot
4. Conclusion
5. References

Problem Statement

Employing Neural Network in Trajectory control of Robot(Mobile autonomous).

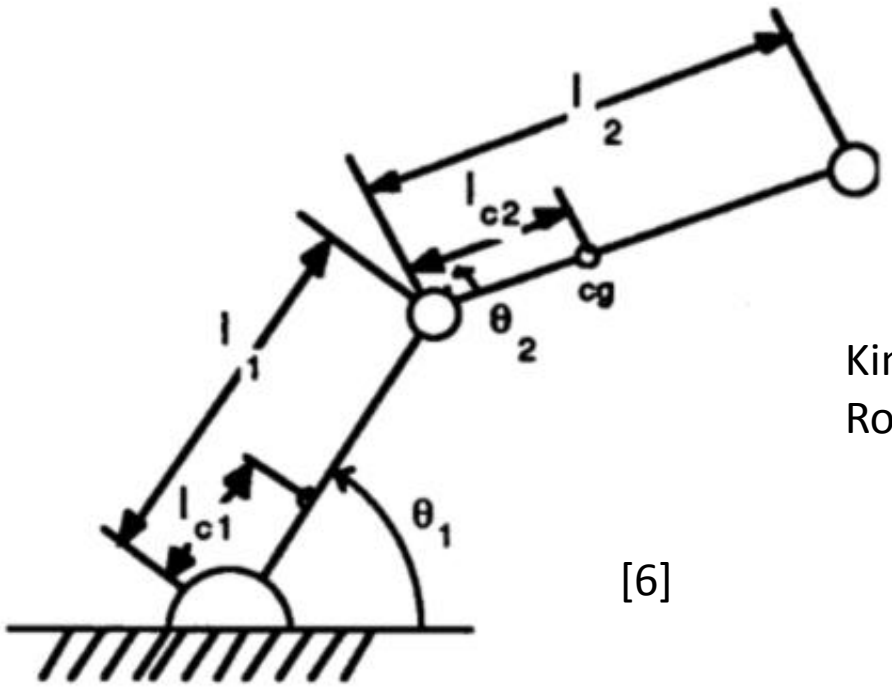
https://robotacademy.net.au/lesson/paths-and-trajectories/

Content

1. Neural Network in a Nutshell
2. Robot Trajectory Control : Problem Statement
3. Neural Network controller
   - A Basic Model (on Robot Manipulator)
   - Model Based on Quadrotors(Focus Model)
   - Additional Model on Mobile wheeled Robot
4. Conclusion
5. References

Trajectory control of End effector of Robot Manipulator

- Inverse Kinematics(Jacobian) by Neural Network
- FNN with Back propagation



Kinematics of Robot Arm

[6]

PARAMETER	VALUE
l_1	0.39 m
l_2	0.42 m
l_{c1}	0.35 m
l_{c2}	0.20 m
M_1	10.0 kg
M_2	3.0 kg
I_1	0.0597 kg-m^2
I_2	0.0494 kg-m^2

$$J^{-1} = \frac{1}{l_1 l_2 \sin\theta_2}\begin{bmatrix} l_2\cos(\theta_1+\theta_2) & l_2\sin(\theta_1+\theta_2) \\ -(l_1\cos\theta_1 + l_2\cos(\theta_1+\theta_2)) & -(l_1\sin\theta_1 + l_2\sin(\theta_1+\theta_2)) \end{bmatrix}$$

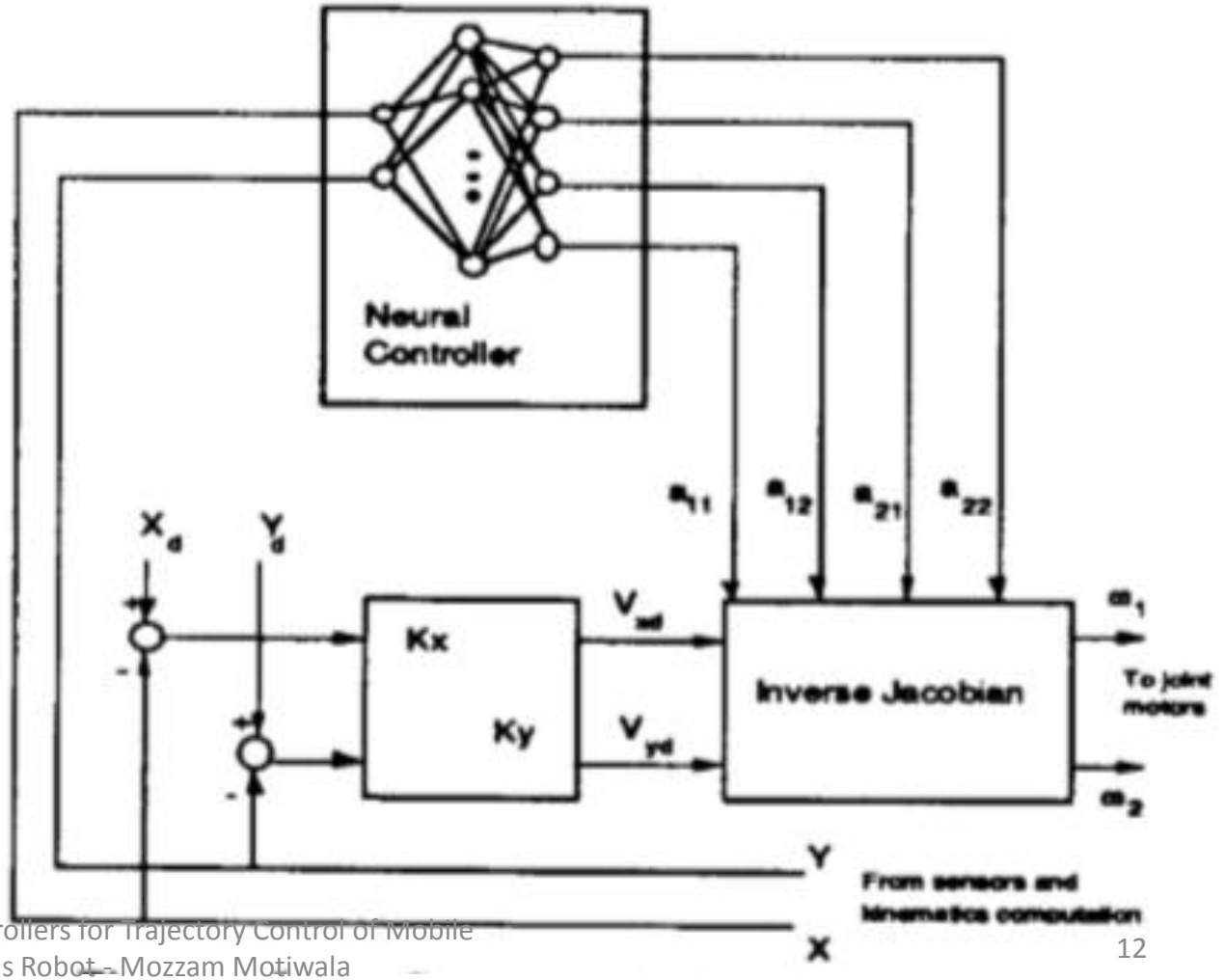
Neural Network design

- 2 angles as input and 4 components of Jacobian as output

TABLE II
VALUES OF θ_1 AND θ_2 USED TO CONSTRUCT TRAINING SET USED TO TRAIN THE INVERSE JACOBIAN NEURAL NETWORK

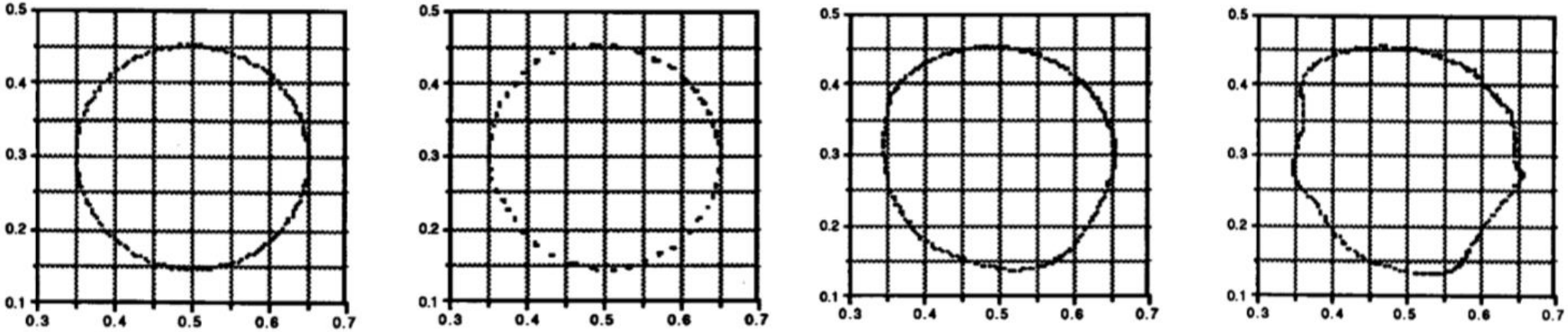
θ_1 (degrees)	θ_2 (degrees)
10	-135
30	-105
50	-75
70	-45
90	-15
110	15
130	45
150	75
170	105
	135

[6]



Functionality and result

- Desired position is compared with the actual position which can be inferred from joint angle measurements. The position error vector is used to scale a desired velocity which gives the direction in which the robot must move to drive the error to Zero.
- 2 input, 4 output, 2 hidden layers of 32 neurons each. 5000 cycles for upto 2.5 – 3 % RMS error.



[6]

0.26 m/s

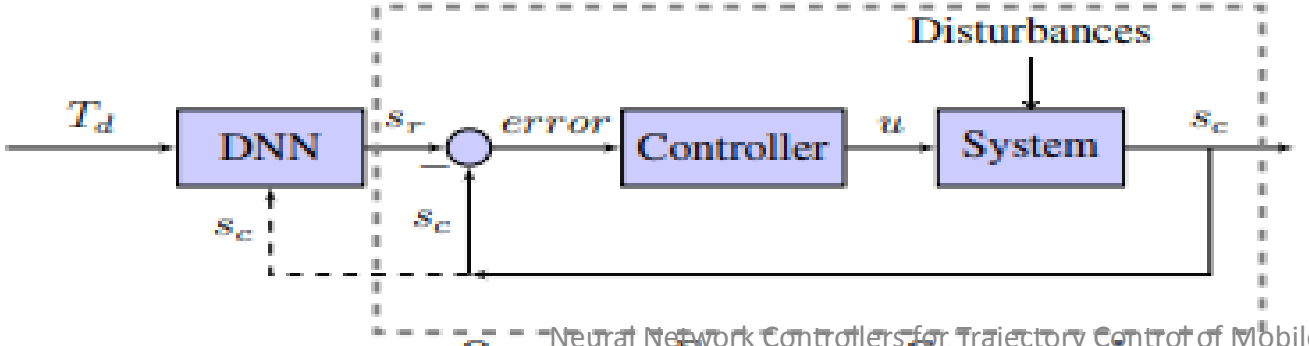
0.59 m/s

Content

1. Neural Network in a Nutshell
2. Robot Trajectory Control : Problem Statement
3. Neural Network controller
   - A Basic Model (on Robot Manipulator)
   - Model Based on Quadrotors(Focus Model)
   - Additional Model on Mobile wheeled Robot
4. Conclusion
5. References

Trajectory Control of Mobile Autonomous Robot(Deep Neural Networks for Quadrotor trajectory)[Video]

- **WHY** : precision of quadrotors can be affected by many factors like
  - Including uncertainty in the turn-rate-to-thrust map
  - unquantifiable time delays
  - aerodynamic effects
  - other unpredictable factors such as friction in the actuators etc.
- **What is does**: Approximate relationship between desired and actual trajectory (highly nonlinear) of baseline controller to reduce it.
- **How:** The proposed method modifies the control system design by adding a DNN block in front of the controller
- A feed-forward DNN with rectified linear units (ReLU) as activation Function



ORIGINAL FEEDBACK CONTROL LOOP

[7]

Experiment Setup

- **The full state of the quadrotor(S)** consists of 13 components.
  - translational position of the quadrotor's centre of mass(X,Y,Z)
  - attitude, represented by Euler angles roll, pitch and yaw(ϕ, θ, ψ)
  - translational velocity($\dot{x}$, $\dot{y}$, $\dot{z}$)
  - rotational velocity, ω = (p, q, r).
  - translational accelerations ($\ddot{z}$)

Input: Current State + 'L' Future Desired State

Output: 1 Reference state (fed to Baseline PD controller as current state by freq 1 /10)

Error Function: Root-mean-square (RMS)

$$E(T_c, T_d) = \sqrt{\frac{1}{N}\sum_{t=1}^{N} \|\mathbf{p}_{c,t} - \mathbf{p}_{d,t}\|^2},$$

Accuracy Measure:

$$I(T_{\text{w/ DNN}}, T_{\text{w/o DNN}}) = \left(1 - \frac{E_{\text{w/ DNN}}}{E_{\text{w/o DNN}}}\right) \times 100\%,$$

[7]

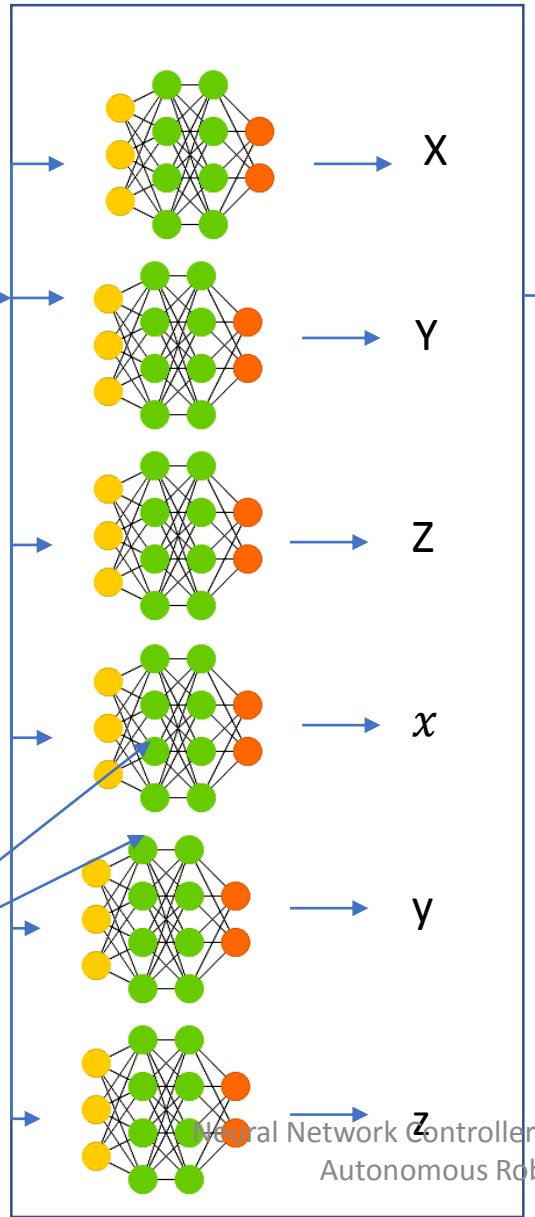
DNN Network design

2L+1 Inputs

1 current state +
L future desired state +

L sets of desired positions relative to the current current position {pd,t+Δ1 −pc,t, . . . , pd,t+ΔL −pc,t}

4 Fully connected Hidden Layer with 128 Neurons Each.



https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464

1 output for each component

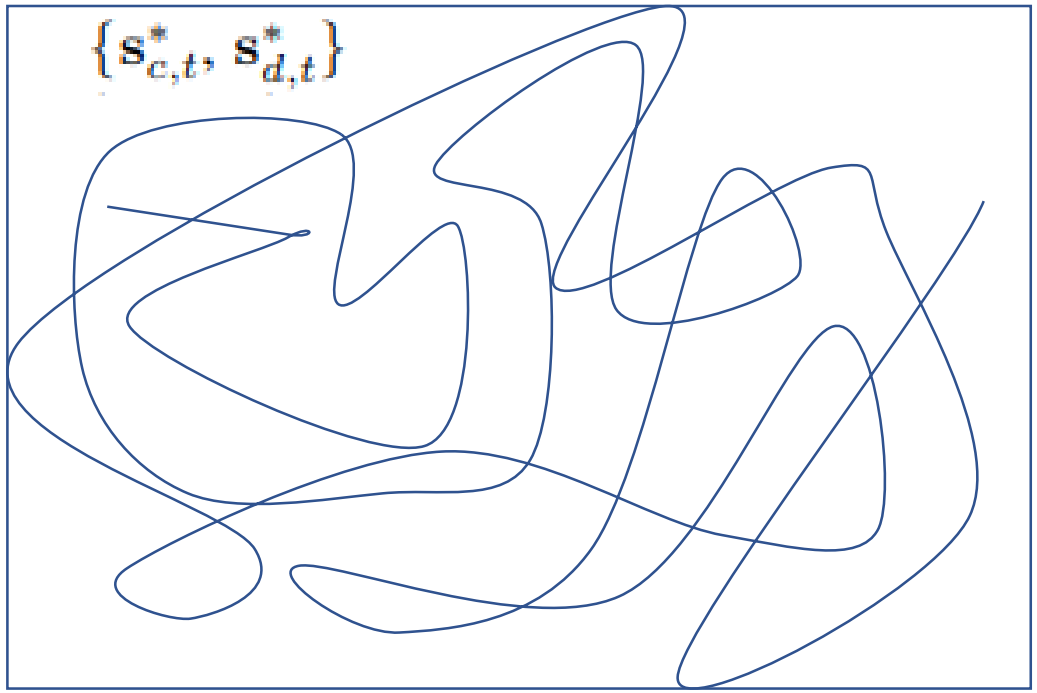
One Neural Net for each of the position and velocity elements of the reference state.

Other elements , {ϕ, θ, ψ, p, q, r, z¨} -> Constant.

Adam optimizer is used to tune the weight and bias parameters in the DNNs,

[7]

DATA Collection and DNN Training



[7]

- a 400-second trajectory of Baseline controller recorded.
- Approximately 10,000 ordered raw data pairs are collected at a 7 Hz rate

- **Input 1** : the current State at time t.
- **Input 2**: the L actual states already traced as L desired states (t+1 -> t+L+1)

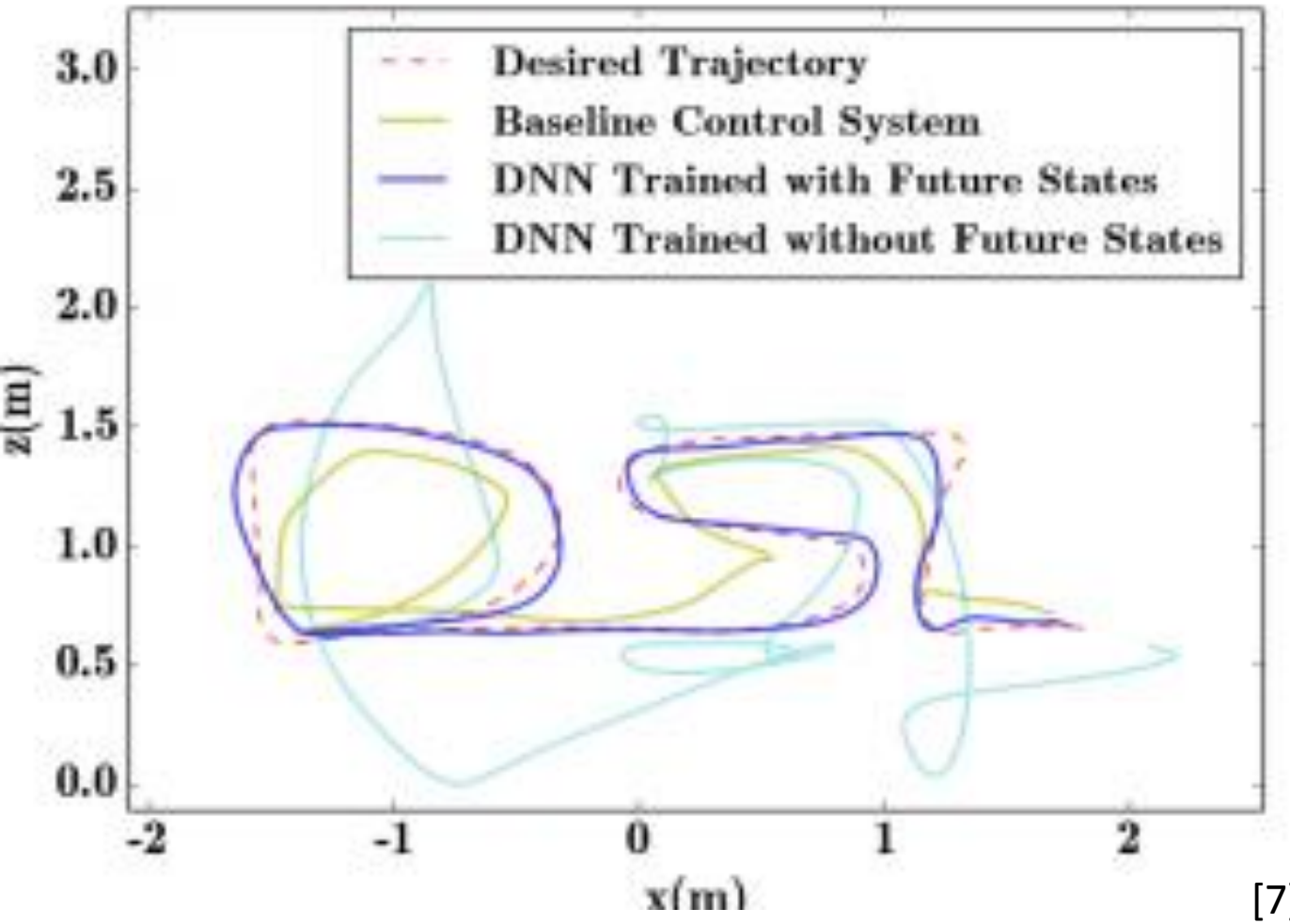
$$\{\mathbf{s}^*_{c,t}, \mathbf{s}^*_{c,t+\Delta_1+1}, \cdots, \mathbf{s}^*_{c,t+\Delta_L+1}\}$$

- **Output** : Current Desired State of Controller

$$\mathbf{s}^*_{d,t}$$

Approximation between Actual states based on desired states of Baseline controller

Results



[7]

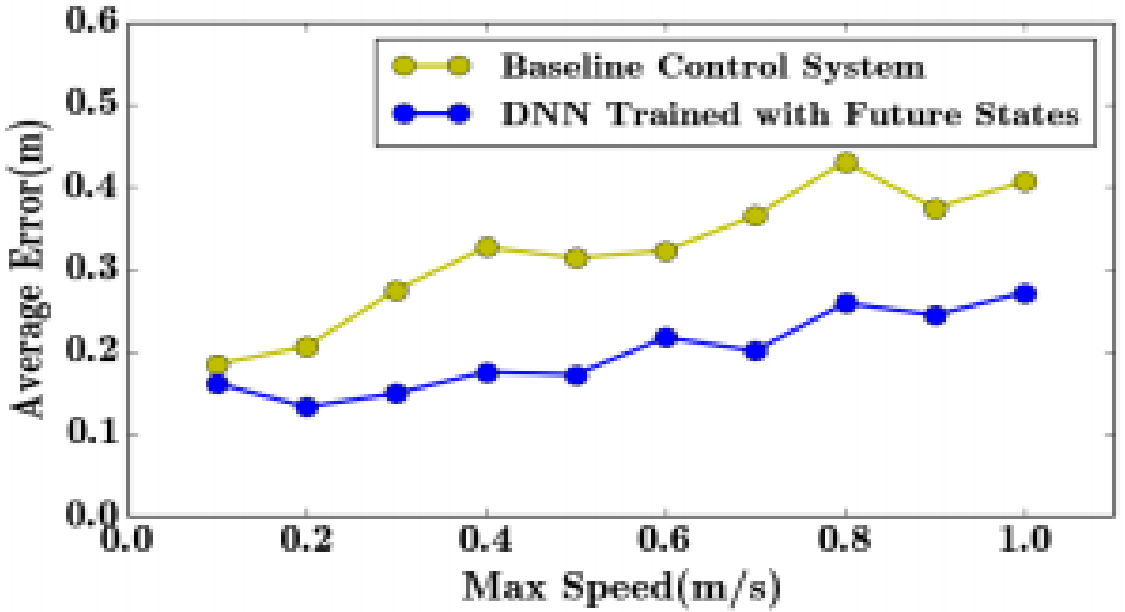


TABLE I

TRACKING ERRORS ON ONE TRAJECTORY FOR THREE DIFFERENT CONTROL SYSTEMS

	Baseline Controller	DNN without Feedback	DNN with Feedback
RMS Error, E (m)	0.360	0.232	0.144
Peak Error (m)	0.605	0.497	0.356
Improvement, I (%)	-	35.6	59.9

Additions done in next paper on this topic

To Identify

- conditions in which the “add-on DNN” can improves performance
- rules for the DNN feature selection.
- conditions to improve efficiency of the training.

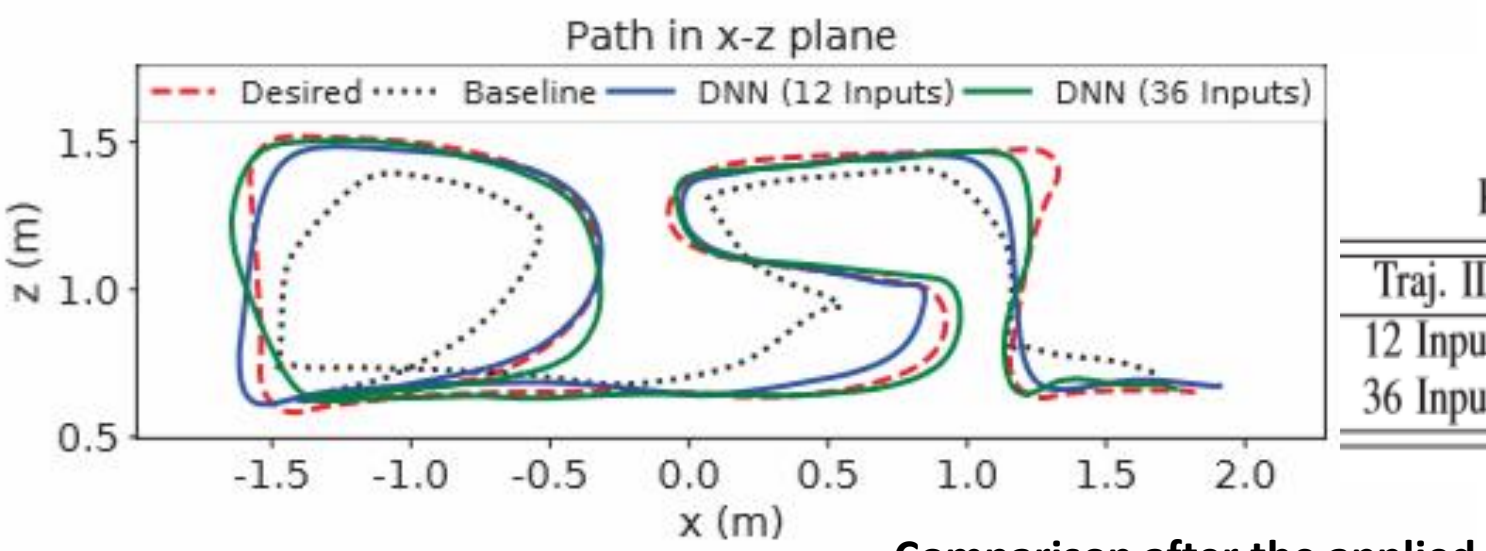


TABLE I

PERCENTAGE REDUCTION IN RMS TRACKING ERROR

Traj. ID	‘DSL’	2	3	4	5	Avg.
12 Inputs	52.4%	48.4%	42.0%	46.6%	36.9%	45.2%
36 Inputs	55.6%	61.4%	39.7%	43.9%	26.6%	45.5%

[8]

Comparison after the applied results with the original system

Future scope and limitation of this Idea

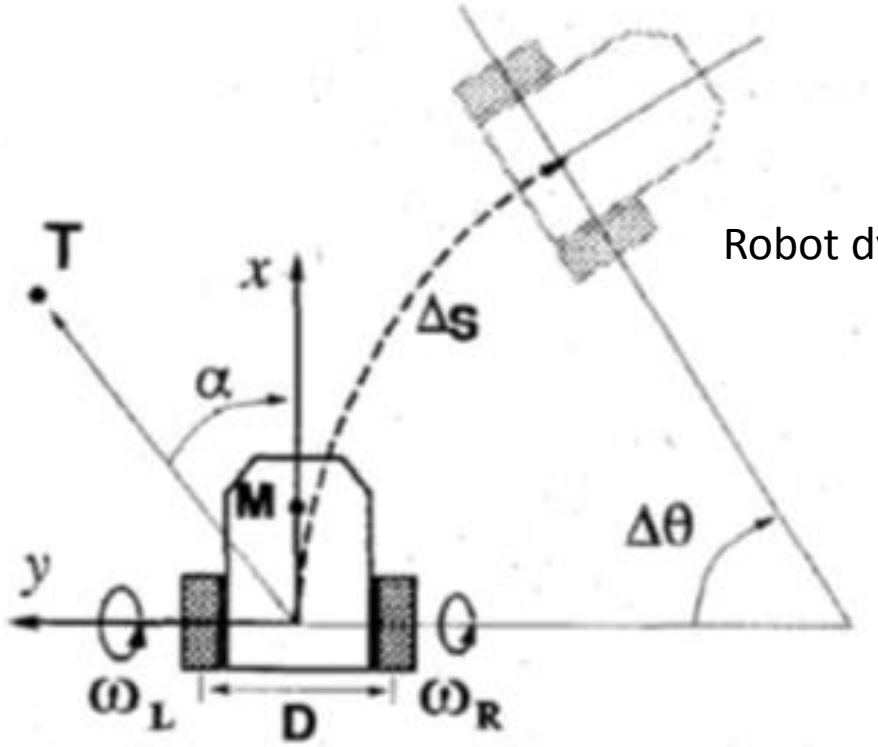
- The DNN serves as a pre-block outside the feedback control loop, the proposed method can be generalized as an add-on to any “Blackbox”, stable feedback control system.
- **More effective Neural Network design can be implemented to enhance the performance.**
- **Improvement mostly depends on the training.**

[7], [8]

Content

1. Neural Network in a Nutshell
2. Robot Trajectory Control : Problem Statement
3. Neural Network controller
   - A Basic Model (on Robot Manipulator)
   - Model Based on Quadrotors(Focus Model)
   - Additional Model on Mobile wheeled Robot
4. Conclusion
5. References

Bonus architecture(in case of extra time)

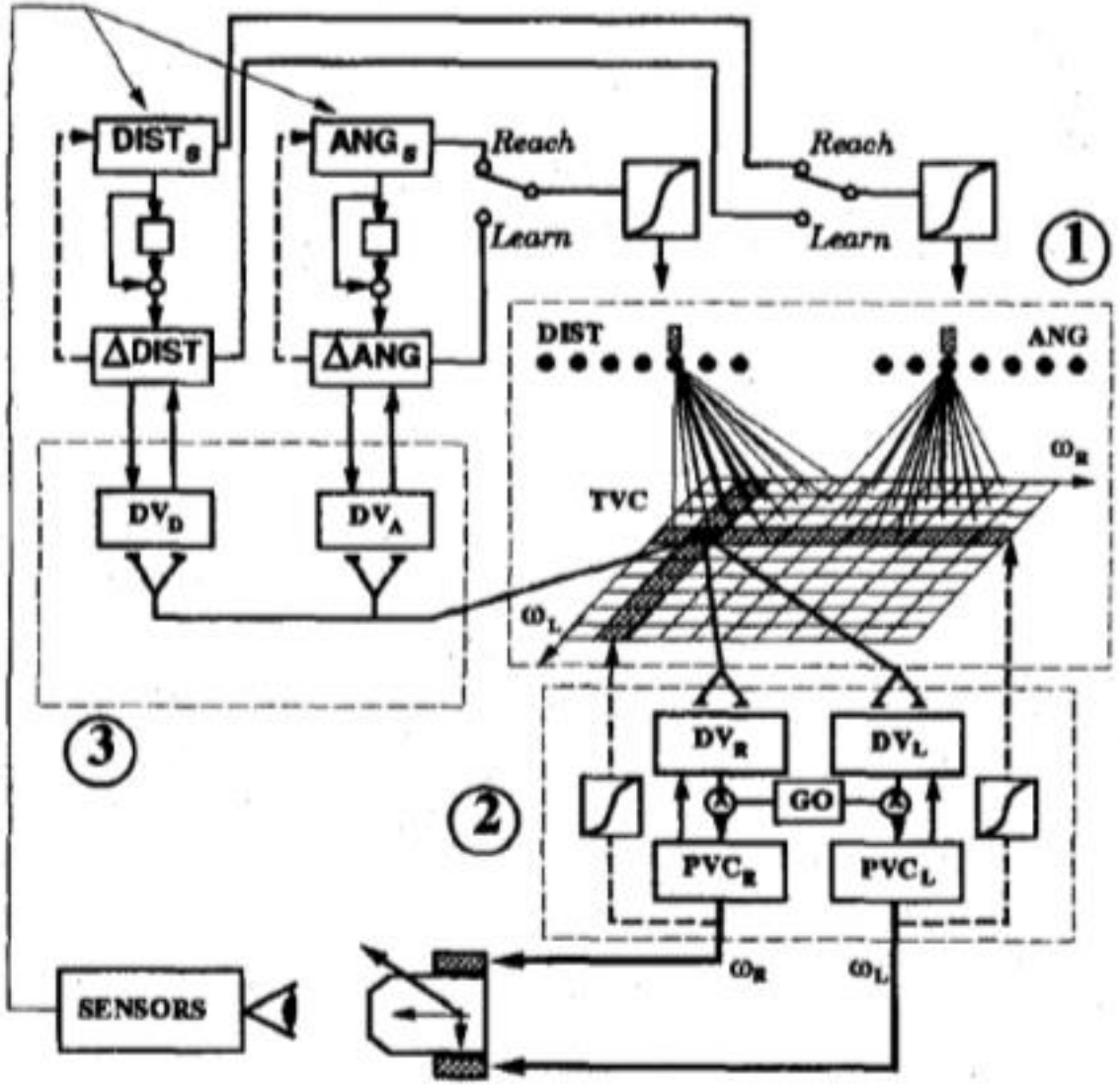


Robot dynamics

1. Adaptive Inverse Odometric Transformation.
2. Velocity Look up Transformation.
3. Adaptive Forward Odometric Transformation.

[9]

Robot Neural Controller Architecture

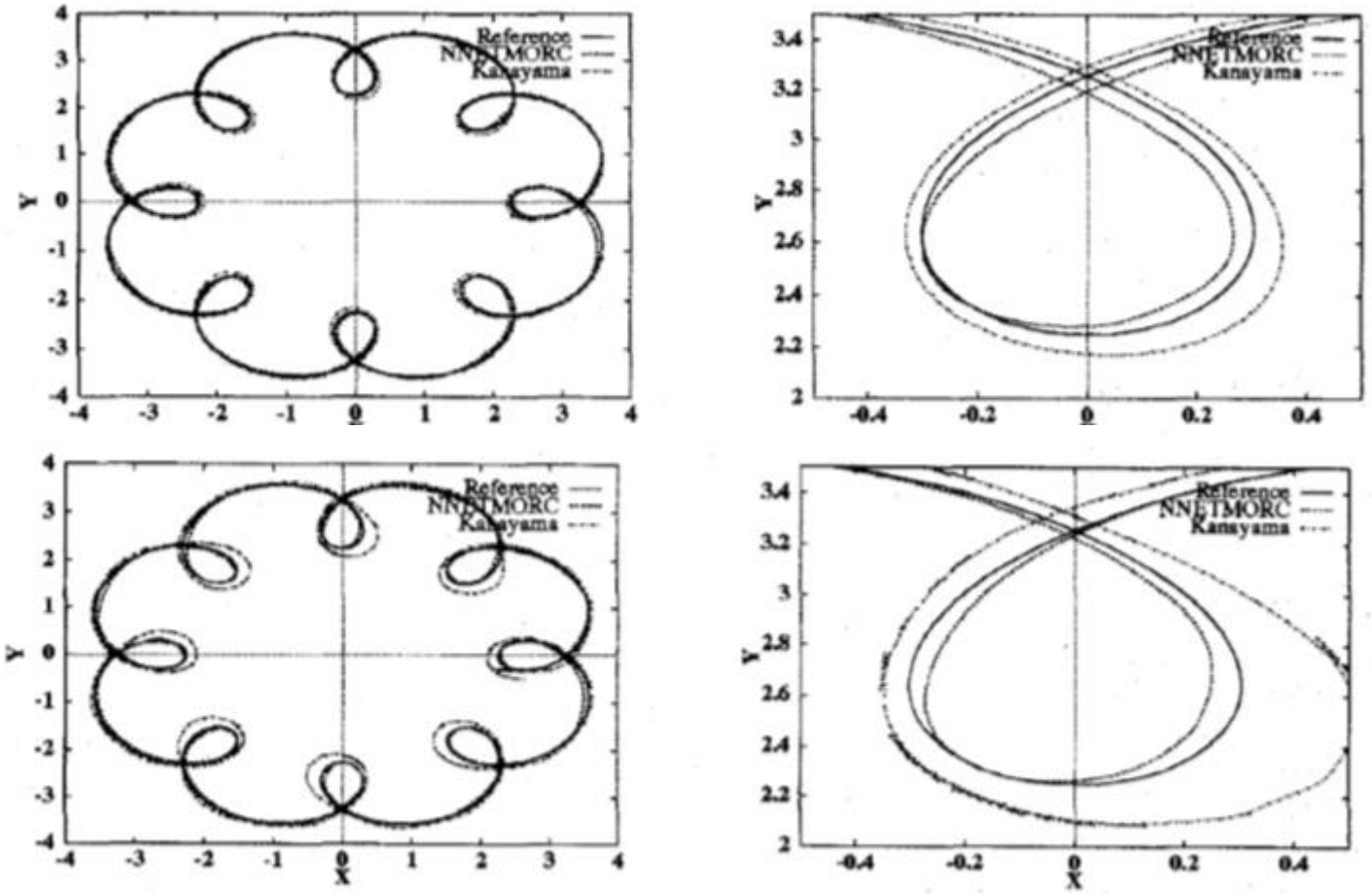


Advantages

- Can operate without sensory information.
- Wheel Slippage
- On line Adaptation.

Without noise

With noise



[9]

Content

1. Neural Network in a Nutshell
2. Robot Trajectory Control : Problem Statement
3. Neural Network controller
   - A Basic Model (on Robot Manipulator)
   - Model Based on Quadrotors(Focus Model)
   - Additional Model on Mobile wheeled Robot
4. Conclusion
5. References

Conclusion

What can be expected out of Neural Network controllers?

- NNs can provide a good approximation of Noise.
- NNs controllers can improve performance as well as accuracy.
- Can provide generalized approximated result even in failure condition.

Cons:

- Their efficacy is directly based on their training in similar environment. Thus, Training has to be comprehensive.

References

Literature:

- [6] Gardner, Brandt, Luecke (1991): “Application of Neural Network for Trajectory control of Robots” in Fifth International Conference on Advanced Robotics 'Robots in Unstructured Environments
- [7] Q. Li, J. Qian, Z. Zhu, X. Bao, M. K. Helwa, and A. P. Schoellig, “Deep neural networks for improved, impromptu trajectory tracking of quadrotors,” in IEEE Intl. Conf. on Robotics and Automation (ICRA), 2017, pp. 5183–5189.
- [8] Siqi Zhou, Mohamed K. Helwa, and Angela P. Schoellig, “Design of Deep Neural Networks as Add-on Blocks for Improving Impromptu Trajectory Tracking”, in 2017 IEEE 56th Annual Conference on Decision and Control (CDC) December 12-15, 2017, Melbourne, Australia
- [9] Paolo Gaudiano, Eduardo Zalama, and Juan L6pez Coronado, “An Unsupervised Neural Network for Low-Level Control of a Wheeled Mobile Robot: Noise Resistance, Stability, and Hardware Implementation ”, in IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS-PART B: CYBERNETICS, VOL. 26, NO. 3, JUNE 1996