Lane Keeping Assistance of Autonomous Vehicles using Reinforcement Learning and Deep Learning on ROS

Supervised by : Dr. Hazem M.Abbas
Eng. Mohammed Abdou

Done By
Mohamed Alaa El-din Farghaly
Mohamed Ahmed Mohamed Ali
Khaled Hefnawy Mohamed
Rohanda Hamed El Sayed
Karim Mohamed Ibrahim

Problem Definition

- Autonomous lane keeping using **ROS** as a simulation environment and using 3 different algorithms for machine learning .

Reinforcement learning
- Q-learning
- DDPG

Supervised learning

CNN

- Then comparing their performance based on selected criteria

Our Motivation

- We chose **lane keeping** project because it is a basic building block for other autonomous functions and hot area of interest for autonomous industry.

✓ **Why ROS ?**

A realistic environment ,real time simulation ,flexible, with lots of packages and plugins, open source, Modular, very powerful

✓ **Why RL?**

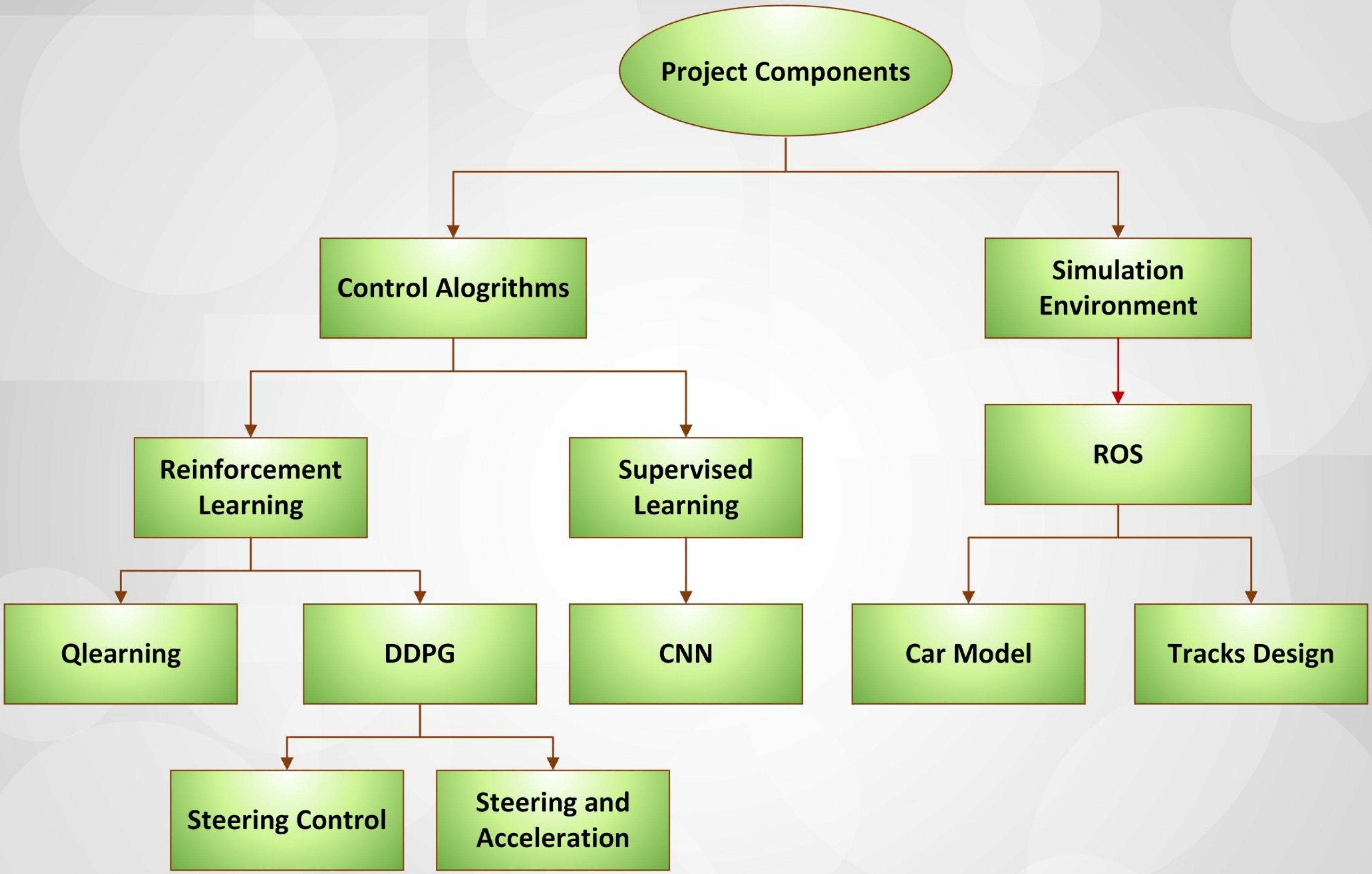
Model-free learning, hot area of research.

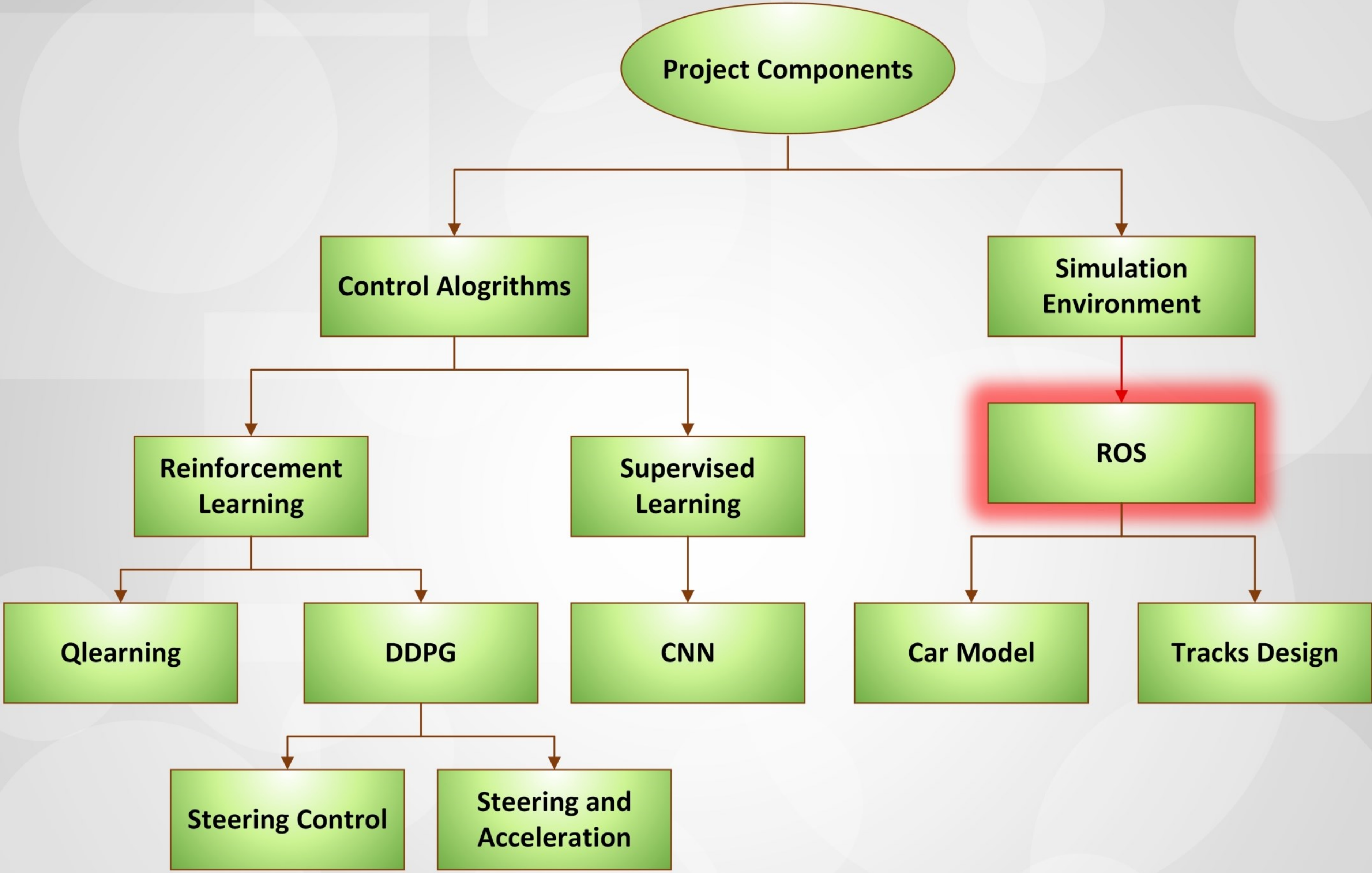
✓ **Why CNN?**

State of the art in autonomous driving algorithms (NIVDIA)

(Bojarski et al., 2016)

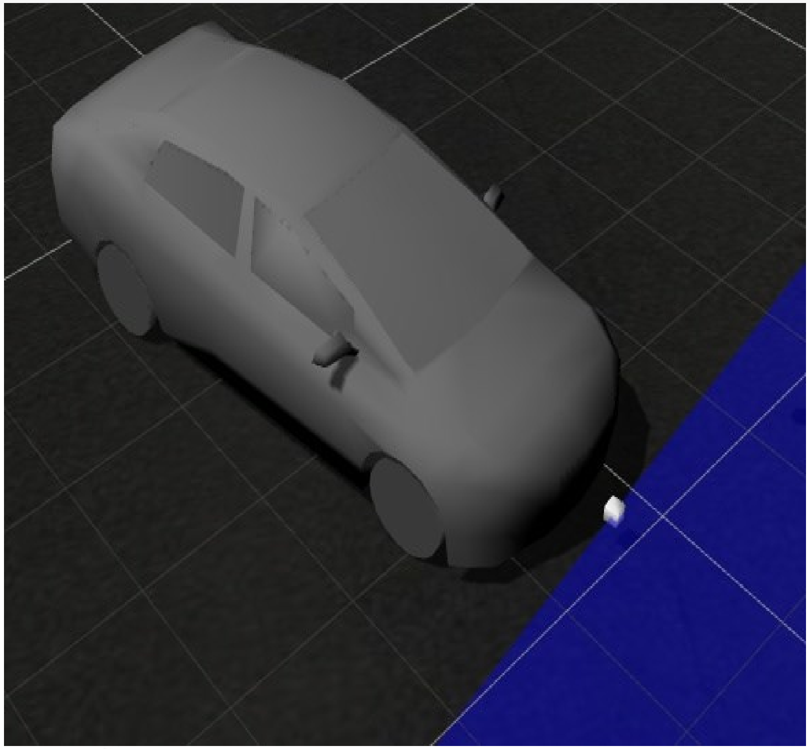
Bojarski, M., Testa, D. D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Zieba, K. (2016). End to end learning for self-driving cars


Project Components
Control Alogrithms
Simulation Environment
Reinforcement Learning
Supervised Learning
ROS
Qlearning
DDPG
CNN
Car Model
Tracks Design
Steering Control
Steering and Acceleration


Project Components
Control Alogrithms
Simulation Environment
ROS
Reinforcement Learning
Supervised Learning
Qlearning
DDPG
CNN
Car Model
Tracks Design
Steering Control
Steering and Acceleration

Simulation Environment

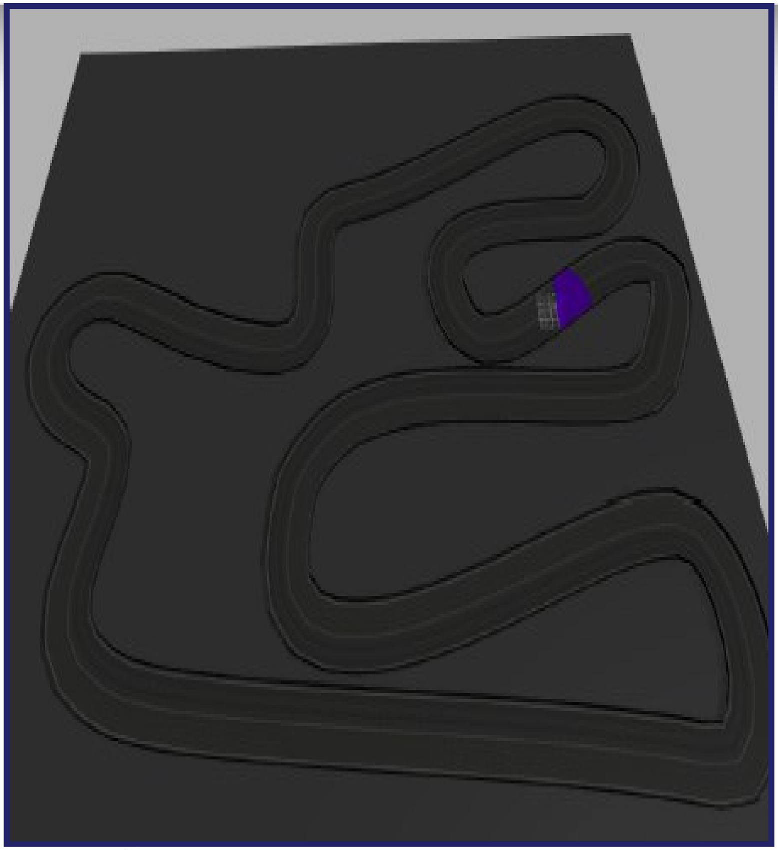
CAR MODEL



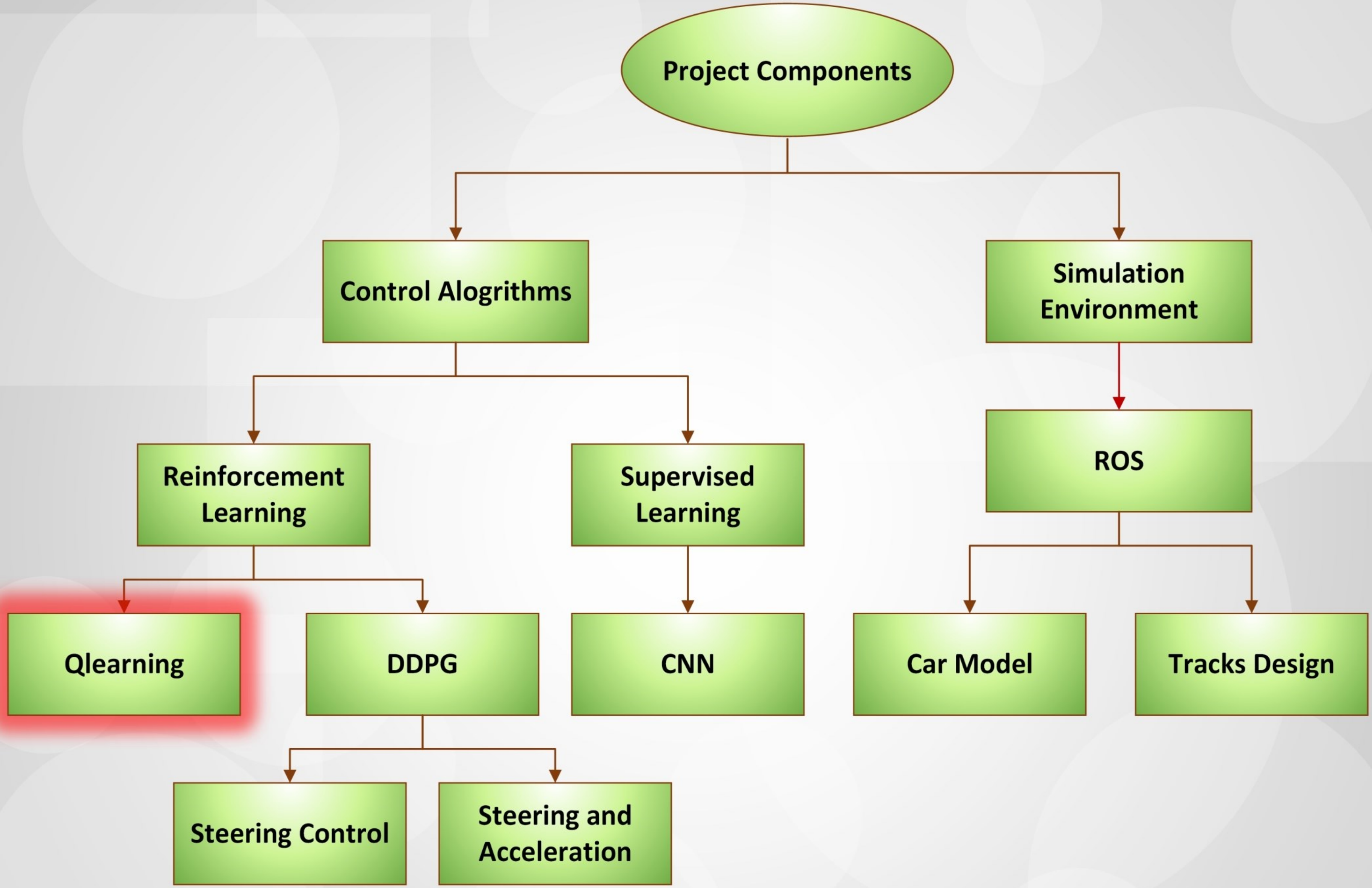
TRACK DESIGN

- **Two tracks** one for testing and other for training
- Made using **Blender**
- **Design considerations**:
  - ➢ Varying patterns
  - ➢ Sharper curves in test track

Training Track



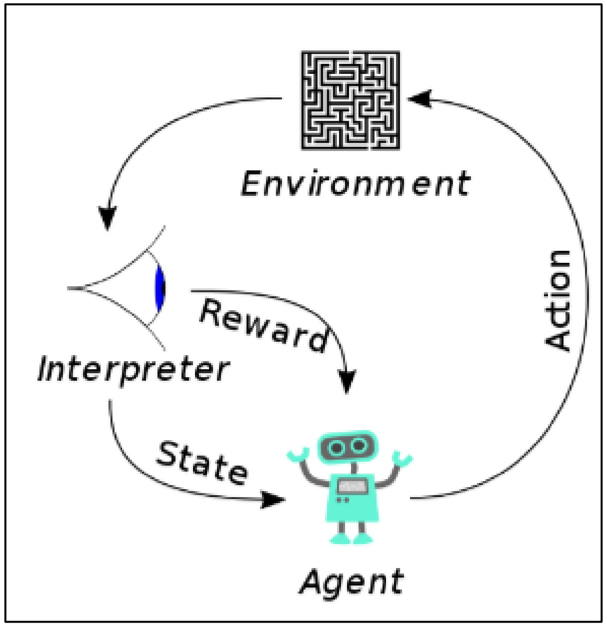
Test Track



Project Components
Control Alogrithms
Simulation Environment
Reinforcement Learning
Supervised Learning
ROS
Qlearning
DDPG
CNN
Car Model
Tracks Design
Steering Control
Steering and Acceleration

RL in a nutshell

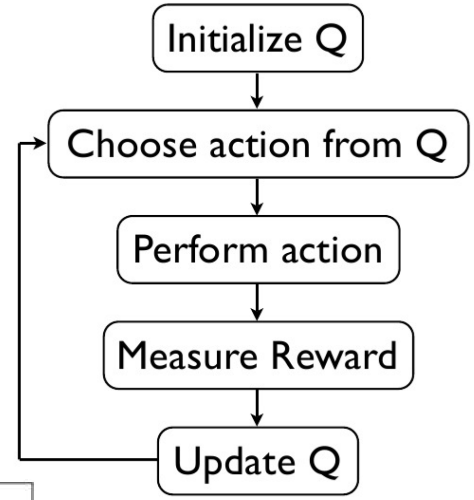
- RL is capable of model-free learning.
- Agent takes action a based on policy π.
- Action a moves the agent from state s to s'
- Agent is rewarded based on a designed Reward function R_t
- Agent's objective is to find the policy that maximizes the cumulative discounted reward.
- Action-value function $Q(s, a)$ is an indication of how good it is to take action a while being in state s based on estimation of future Rewards.



*Loiacono, D., Prete, A., Lanzi, P. L., & Cardamone, L. (2010). Learning to overtake in torcs using simple reinforcement learning. *IEEE Congress on Evolutionary Computation.*

Q-learning

- Discrete Off-policy RL algorithm.
  - Agent's policy is updated using experience sampled by following a different policy.
- It can only deal with finite discrete states.
- It constructs a Q-table that
  - Includes action-value function $Q(s,a)$ of all possible state-action pairs.
  - Serves as a trained agent that can be used in different environments.
- $Q(s,a) = Q(s,a) + \alpha[R_t + \lambda max_{a'} Q(s',a') - Q(s,a)]$



	State 1	State 2	State 3	State 4	State 5
Action 1	-8.85	35.35	28.44	-0.312	-106.4
Action 2	-9.95	25.10	28.44	-0.659	-106.1
Action 3	-8.09	33.25	28.31	-0.039	-104.7
Action 4	-7.06	28.09	27.44	-0.111	-101.6

fppt.

*Karavolos, D. (2013). Q-learning with heuristic exploration in simulated. *Artificial intelligence. Amsterdam: University of Amsterdam*

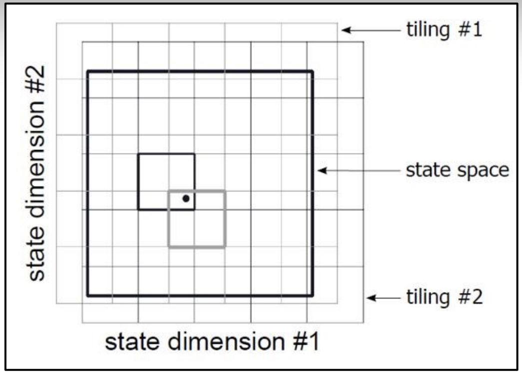
Discretization

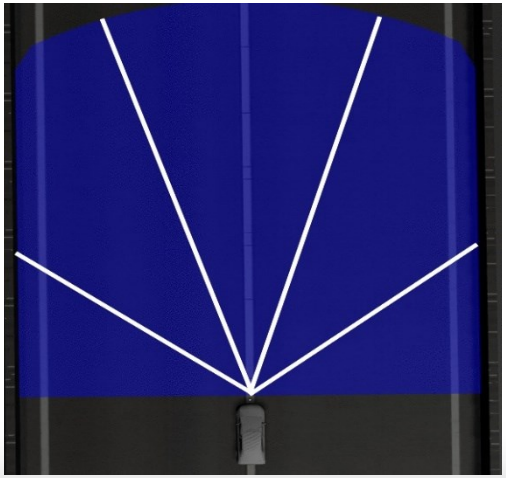
- Tile coding
  - Discretization of multi-dimensional state-space
  - Every 10 laser arrays are averaged into one representative reading.

- State-space dimension
  - 5 laser readings
  - Each reading is approximated to one of 3 possible values.

- Action-space dimension
  - 7 possible steering values.

- Final Q-table
  - 1701 cells

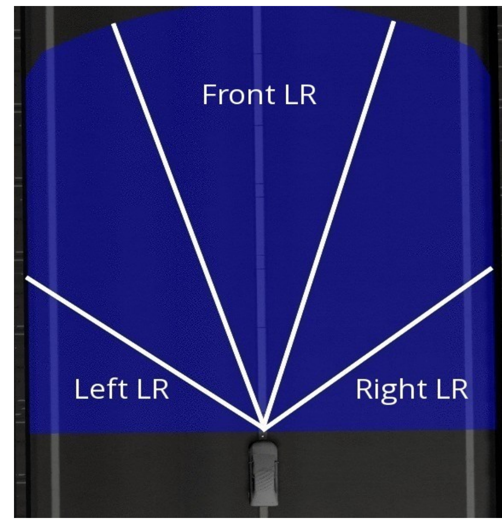




Reward Function

- Agent receives proportional positive reward when
  - It sees no obstacle ahead of it.

- Agent receives proportional negative reward when
  - There is a obstacle ahead of it.
  - There is a difference between leftmost and rightmost laser-readings.
    - Indicative of how successful the agent is in sticking to the center of the lane.



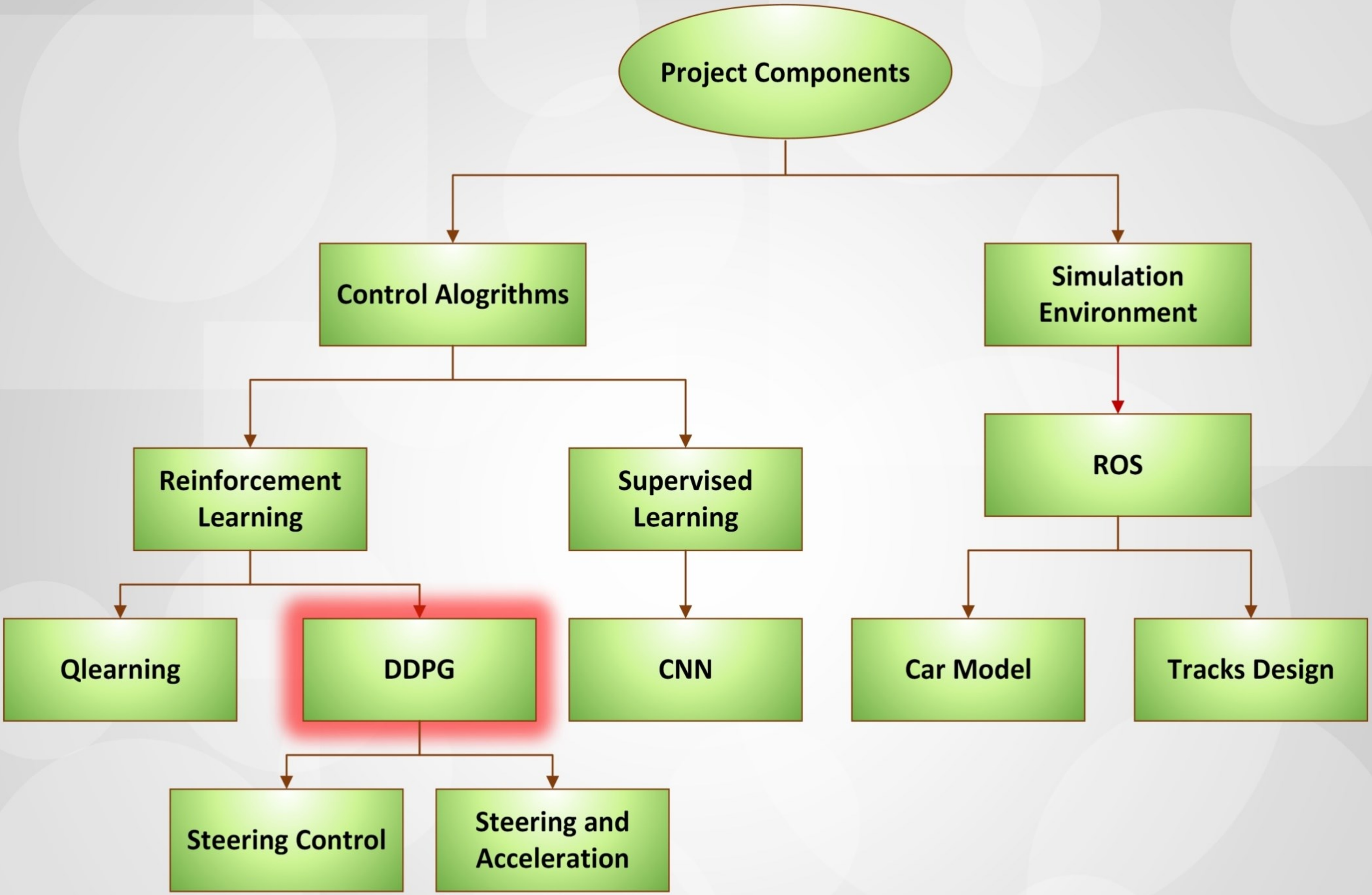
- Agent receives big negative reward when
  - The vehicle is not moving.
  - The vehicle crashes.
  - The vehicle loses contact with the ground.

- $R = 0.5 * (FrontLR - 15) - 0.5 * abs(LeftLR - RightLR) - 5 * slowMove$

Exploration by ε-greedy

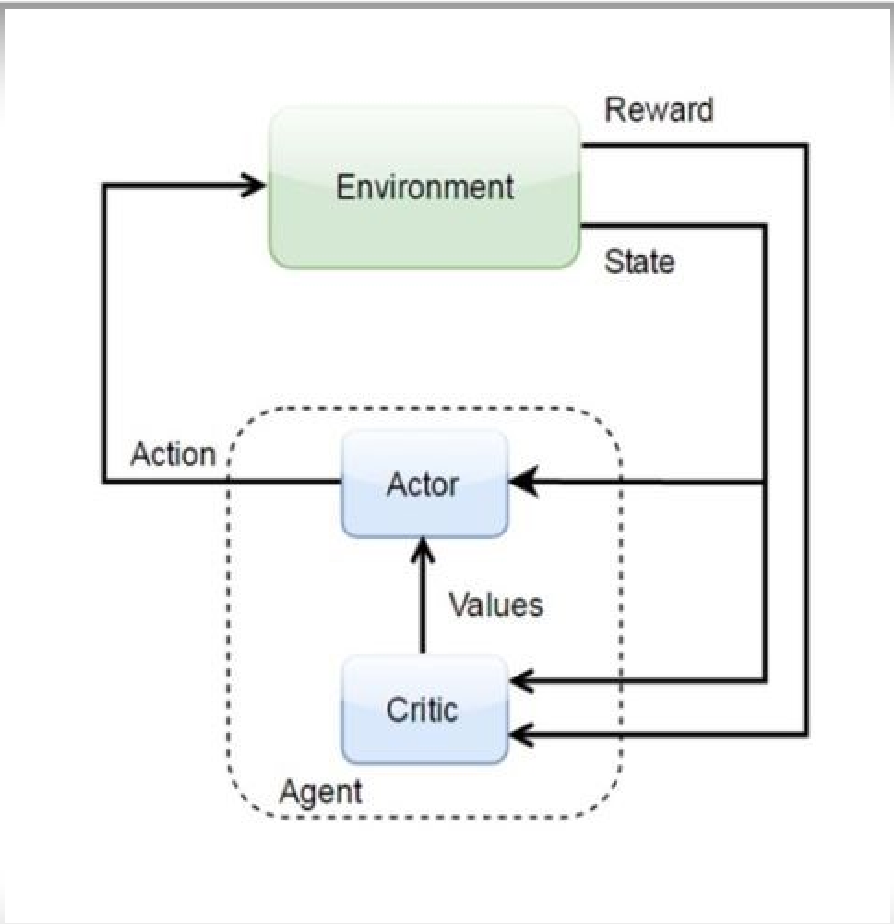
- Infamous exploration technique
  - Adds a small probability ε of taking a totally random action to the policy followed by the agent.
- Advantages
  - Forces the agent to visit as much unknown states as possible
  - Avoids overfitting of the trained model.
  - Ensures better convergence to a more general model.
- $$\pi(s_t) = \begin{cases} Random\ action\ a \in A(s_t) & if\ (\rho < \varepsilon) \\ argmax_a\ Q(s_t, a) & otherwise \end{cases}$$

- Agent successfully performed well on both training and testing tracks


Project Components
Control Alogrithms
Simulation Environment
Reinforcement Learning
Supervised Learning
ROS
Qlearning
DDPG
CNN
Car Model
Tracks Design
Steering Control
Steering and Acceleration

Deep Deterministic Policy Gradient (DDPG)

- Algorithm Combined of 3 techniques:
  1. Deterministic Policy-Gradient
  2. Actor-Critic Methods
  3. Deep Q-Network
- DQN is better for Continuous state-action space problems
- The Actor-Critic Algorithm is a hybrid method of the policy gradient method and the value function method.

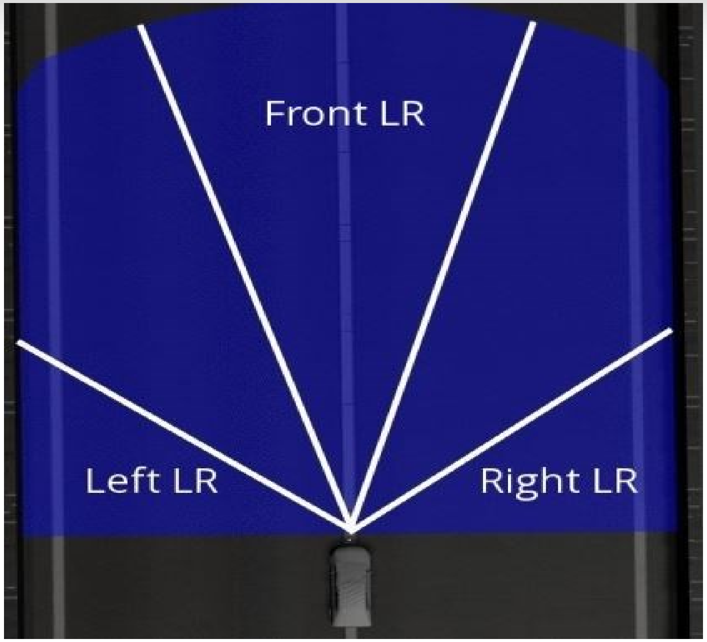


*Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2015). Continuous control with deep reinforcement learning. CoRR, abs/1509.02971

Control Steering angle

- Constant speed
- States (Dim=7)
  - Laser sensors ranges 5
  - Current speed
  - Current angle

- Actions(Dim=1)
  - Steering angle (- 0.5:0.5)



$$Reward = c_1 * (FrontLR - 15) - \mathrm{c}_2 * \mathrm{abs}(\mathrm{LeftLR} - \mathrm{RightLR}) - \mathrm{c}_3 * (\mathrm{slowMove})$$

Control Acceleration and Steering (1/2)

- States (Dim=31)
  - Laser sensors ranges 25
  - Current speed
  - Current angle
  - wheel-ground contact normal 4

- Actions(Dim=2)
- Steering angle (- 0.5:0.5)
- Added Velocity (-1 : 1)

Control Acceleration and Steering (2/2)

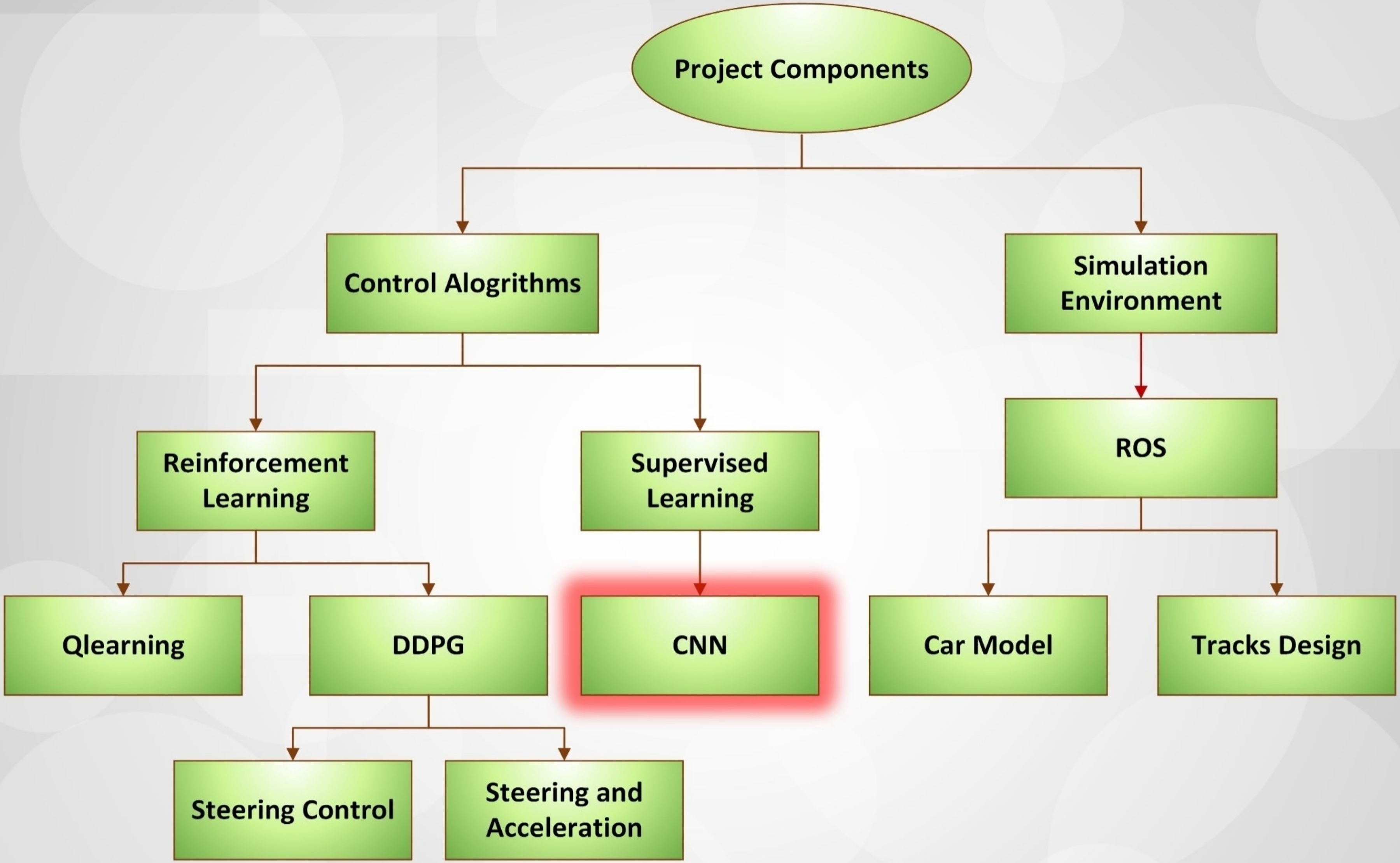
- $Reward = normalized\ velocity\ (c_1(FrontLR - 15) - c_2 * abs(LeftLR - RightLR) - c_3 * angle) - c_4(slowMove)$

- Reward -=1 if a wheel lost contact with ground

Exploration

for steering and acceleration control

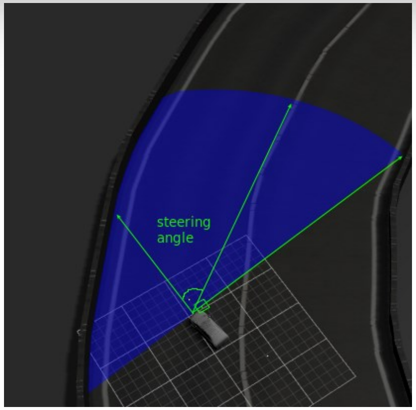
- Steering only control *Exploration steps* = 7,000 steps
- Steering and acceleration control = 200,000 steps
- Explore the track using Ornstein-Uhlenbeck (OU) process
- $\varepsilon_t = \varepsilon_{t-1} - \frac{1}{Exploration\ steps}$; $\varepsilon_0 = 1$
- $dx_t = \theta(\mu - x_t)dt + \sigma dW_t$: $dW \sim N(0,1)$, $dt=1$
- $X_t = x_t + \varepsilon_t \, dx_t$
- Steering ,mean=0
- Acceleration, mean=0.3

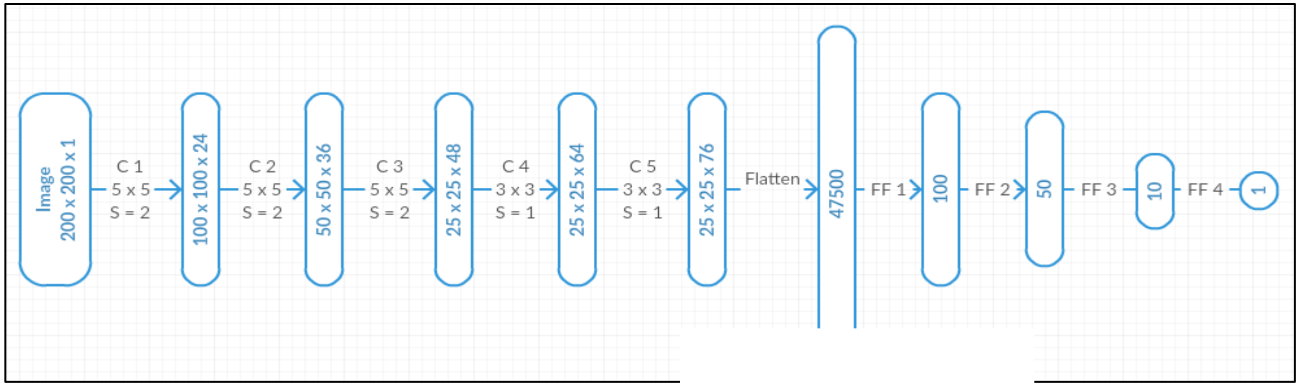

Project Components
Control Alogrithms
Simulation Environment
Reinforcement Learning
Supervised Learning
ROS
Qlearning
DDPG
CNN
Car Model
Tracks Design
Steering Control
Steering and Acceleration

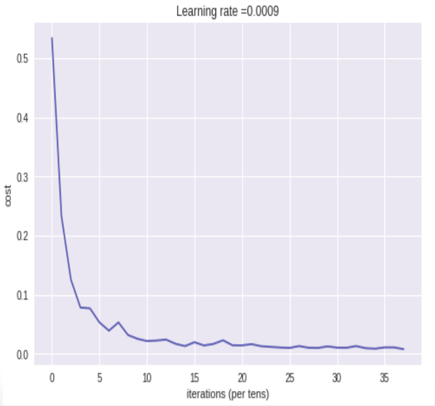
CNN Algorithm

- The train and test data are extracted from informed action algorithm.
- The cost function is Root Mean square Error.

$$Cost = \sqrt{\frac{\sum_{i=1}^{n}(predictedOutput_i - exactOutput_i)^2}{n}}$$







*Innocenti, C., Linden, H., Panahandeh, G., Svensson, L., & Mohammadiha, N. (2017). Imitation learning for vision-based lane keeping assistance.

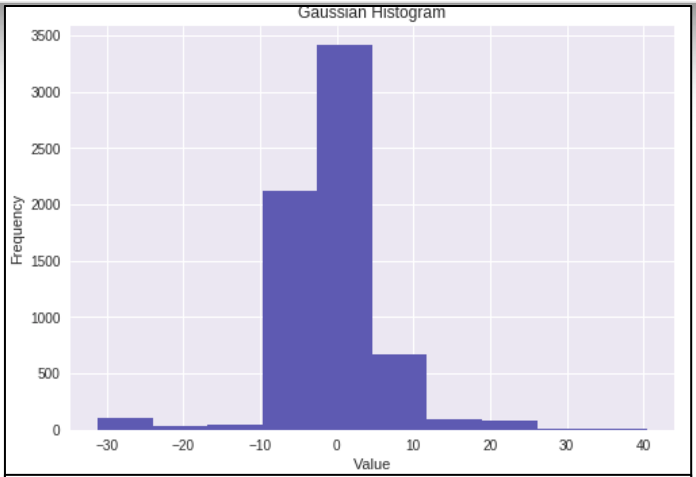
CNN Enhancements (1/2)

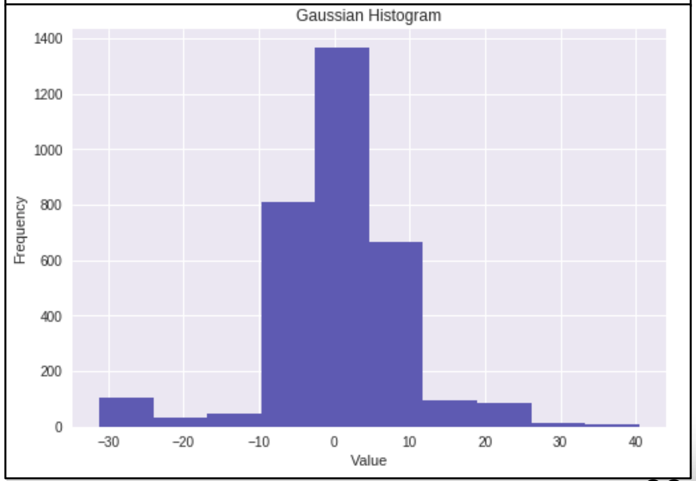
- Temporary Evaluation Criteria → if the car has more than 10 critical errors without interruption, the car will be exposed to an accident.

- **First Enhancement** → modifying cost function.

$$Cost = \sqrt{\frac{\sum_{i=1}^{n}(predictedOutput_i - exactOutput_i)^2}{n}} + \max(error^2)$$

- **Second Enhancement** → modifying the histogram of train data.



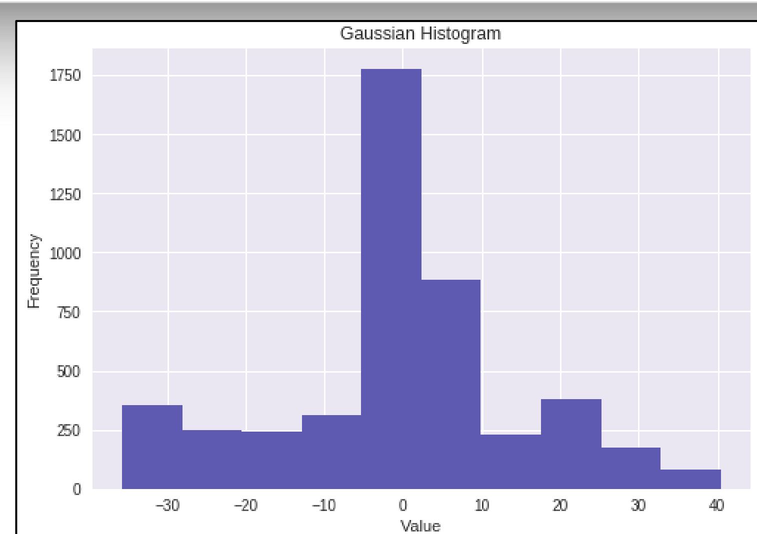


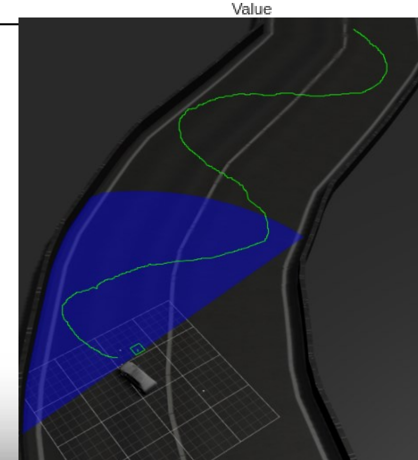
CNN Enhancements (2/2)

The CNN trained on a very little data with steering angle above 25 degrees and below 15 degrees:

1. Any angle above 25 degrees or less than 15 degrees cannot be predicted due to the lack of data.
2. the CNN is not trained to behave properly when the car is going to crash.

- **Third enhancement** → crash avoidance data is extracted from the simulation.





CNN Advantages and Results

1. The CNN algorithm is trained using only one camera (front camera).

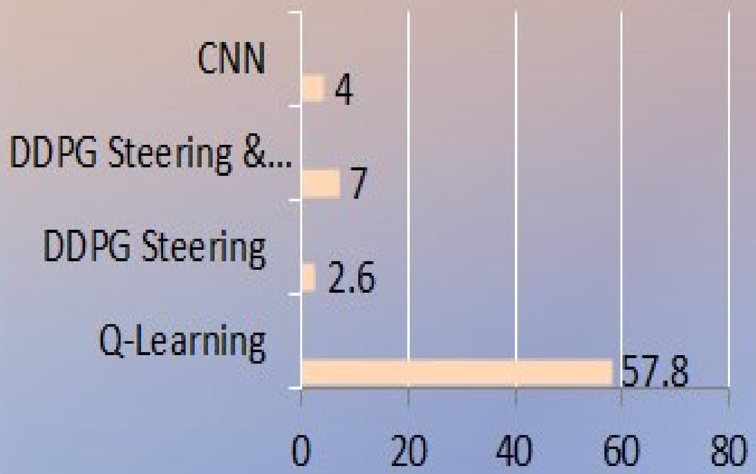
2. The CNN model is capable of finishing test track without any crash.

3. If the car gets out of the lane, the model will be able to get back to the lane and avoid crash.

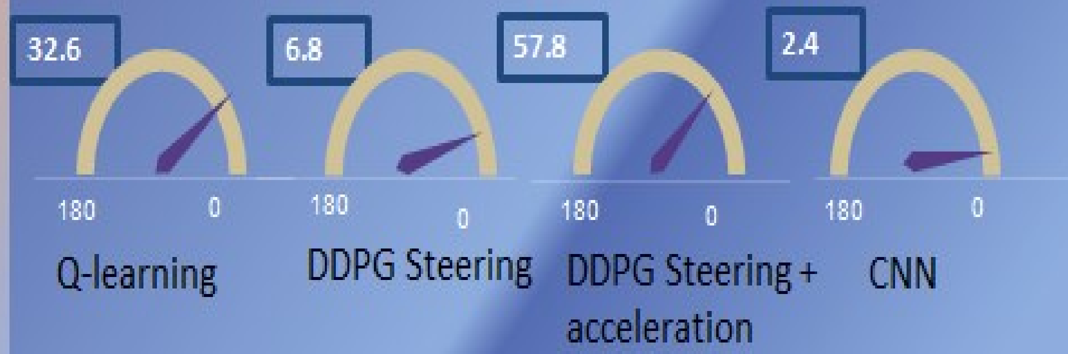
Evaluation Criteria

- Training time
- Number of completed laps
- Mean lap time
- Mean absolute change in steering angle per second
- Mean deviation per lap

Mean Deviation per lap



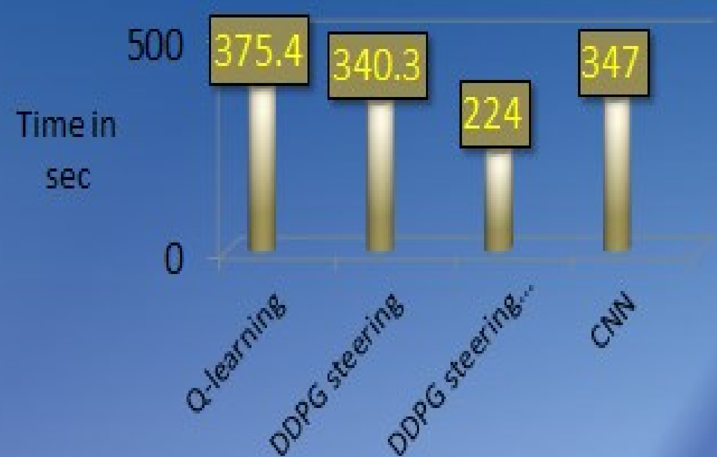
Mean absolute change in steering (deg/sec)



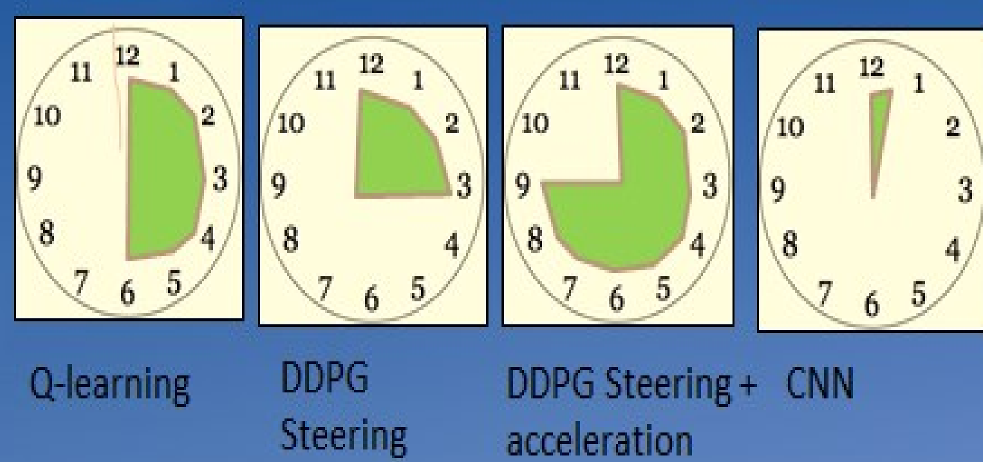
Number of completed laps

All higher than 20 laps

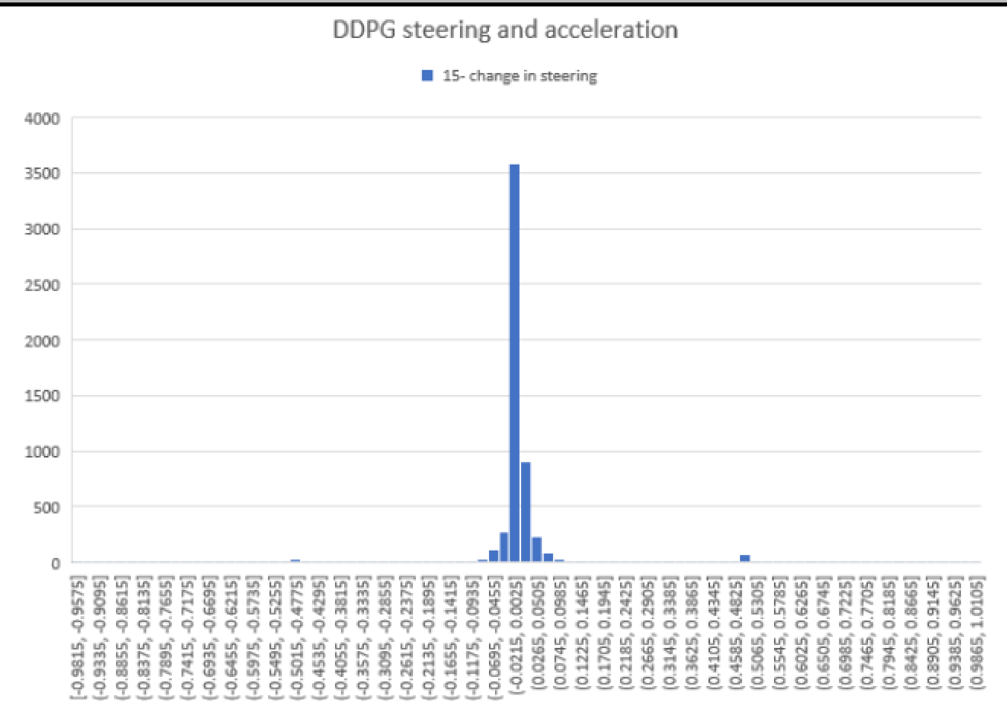
Mean Lap time

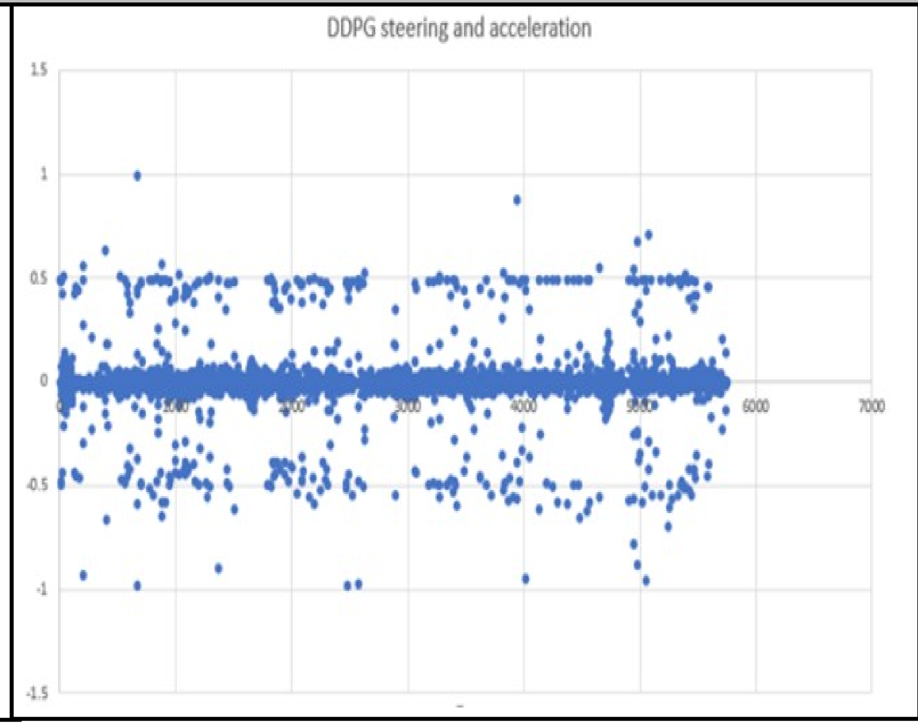


Training time

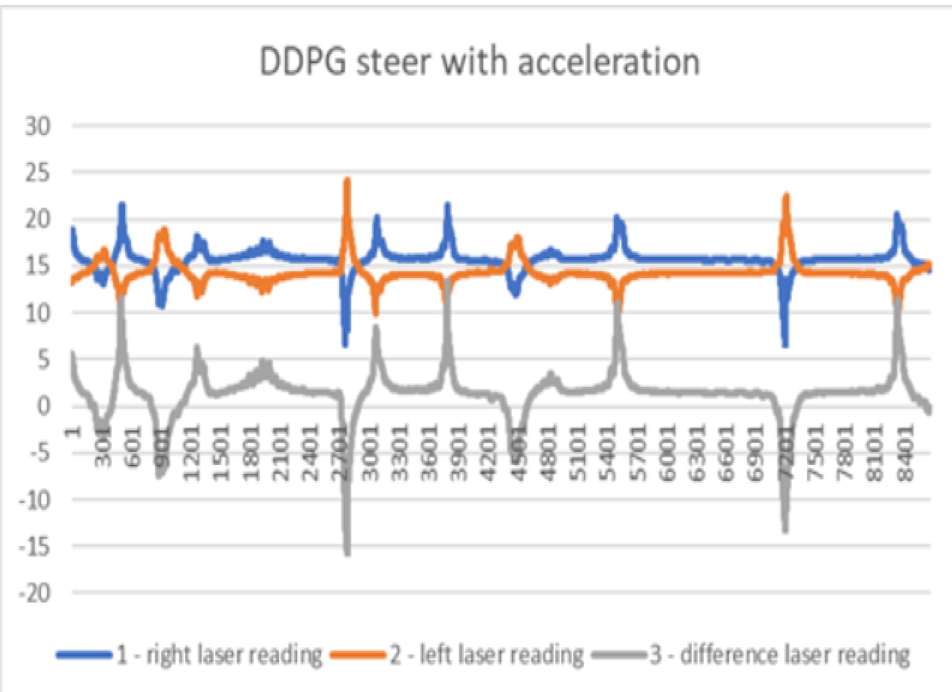


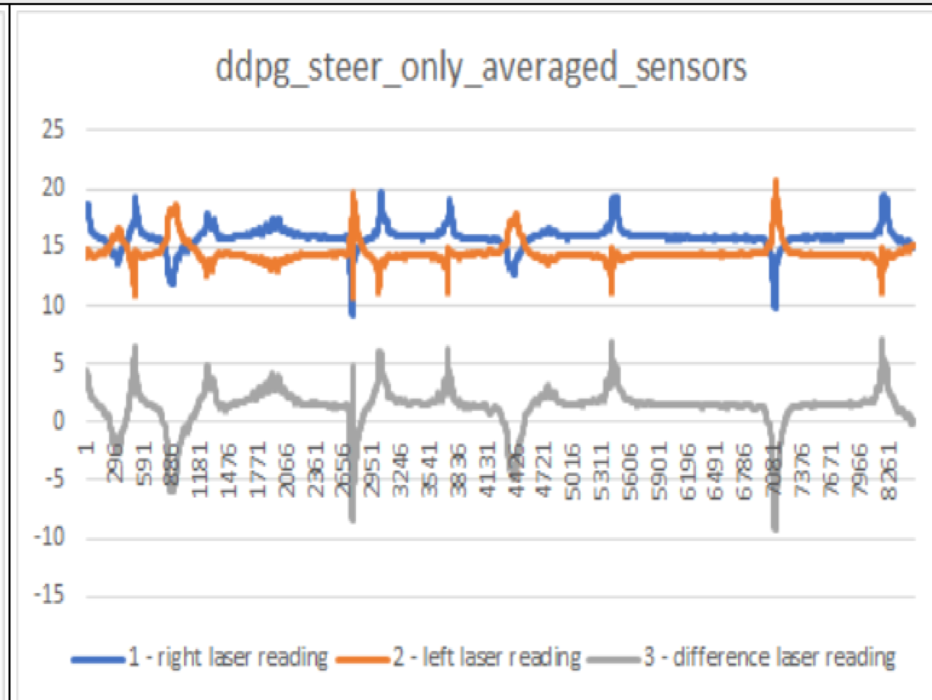
DDPG steering with acceleration graphs



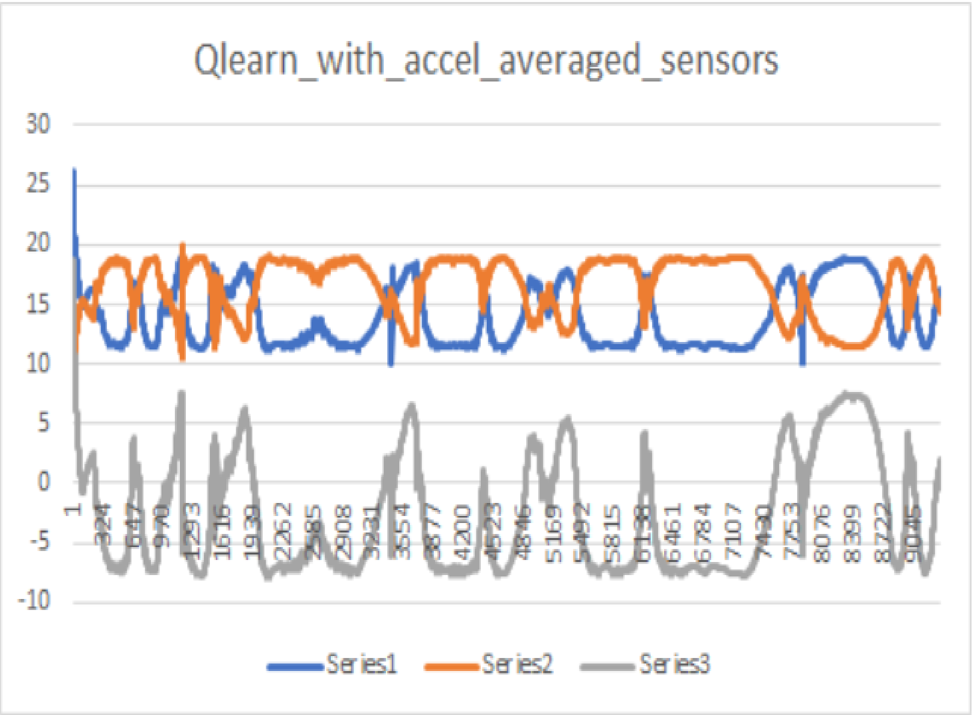


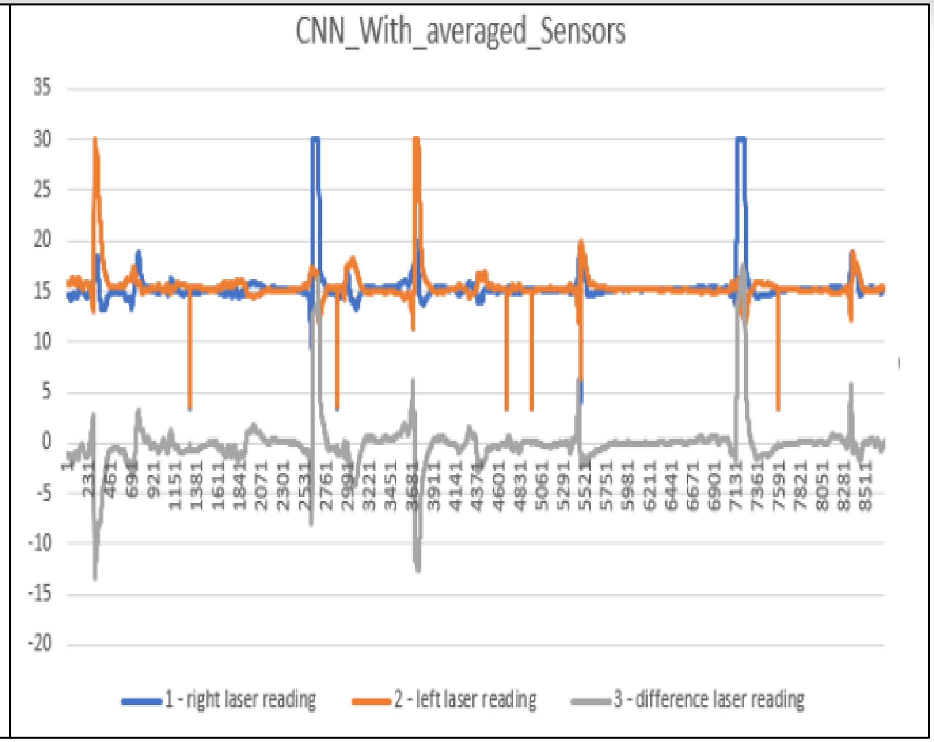
Mean deviation per lap





Mean deviation per lap



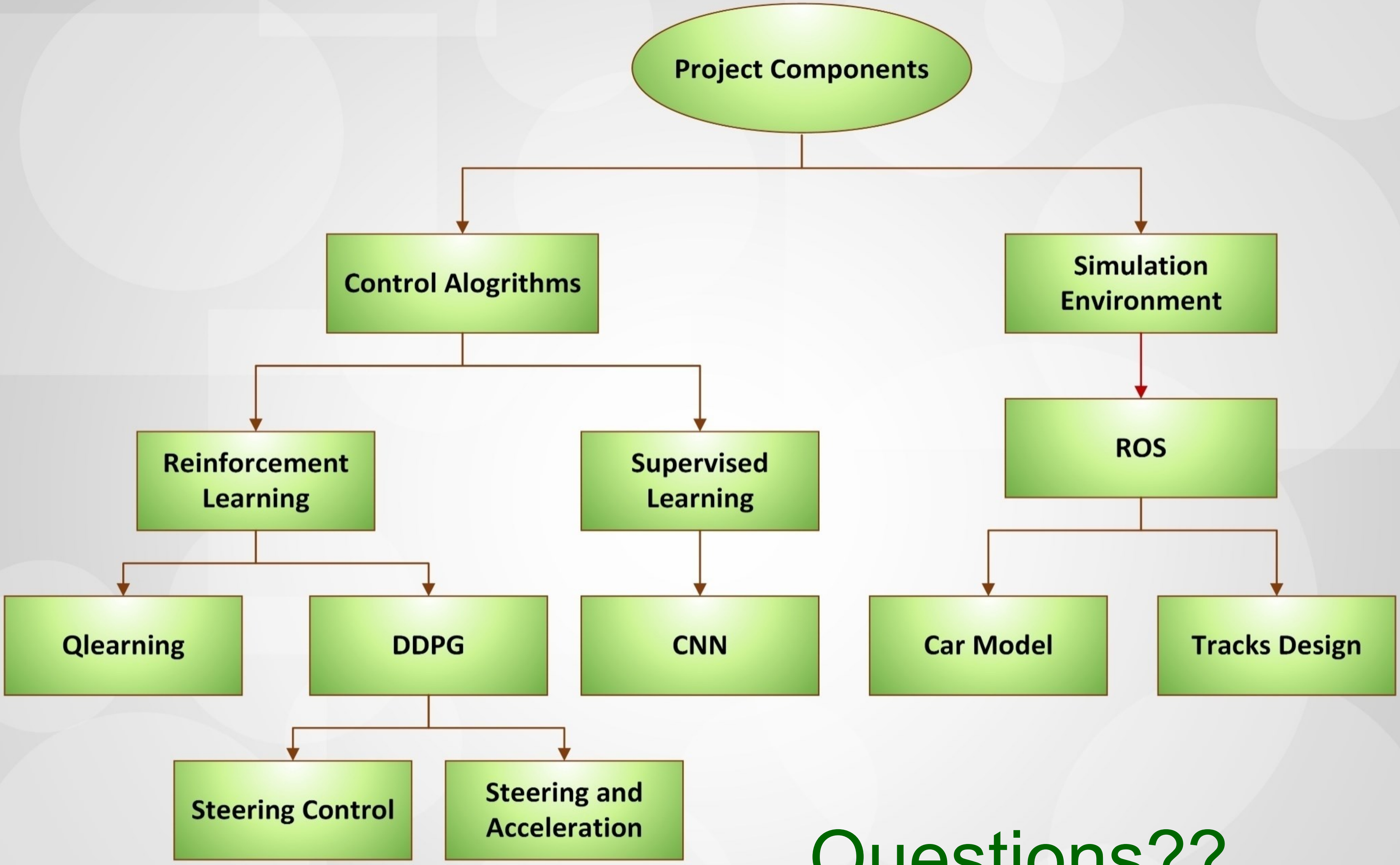


Conclusion and future work

- Managed to build realistic model
- Lane keeping task
- Comparative study for 4 algorithms

Future work

- Crash avoidance
- Cross roads for learning brake


Project Components
Control Alogrithms
Simulation Environment
Reinforcement Learning
Supervised Learning
ROS
Qlearning
DDPG
CNN
Car Model
Tracks Design
Steering Control
Steering and Acceleration

Questions??

Thank you ☺