Fundamentals and Development of Self-Driving cars

Yoganandhan Anbu

Department of Mechatronics Engineering, Chennai Institute of Technology, India

[praviinvj@gmail.com](mailto:praviinvj@gmail.com)

**Abstract** – In real-time Autonomous cars which are driven by digital technologies without using human intervention. They can also navigate their roads, and detects the traffic signs and avoid hitting obstacles such as other cars and pedestrians. Based on the fundamentals of developing self-driving cars are totally based on Sensing and automating tasks. In our proposed system the localization and perception are to makes define and steers the car, to control that autonomous system certain algorithms are used for steering functions.

Keywords – Driving assistance system, Fundamentals of self-driving cars, Introduction to Self-driving cars, Localization, Perception, Intelligent vehicles

**Introduction**

The self-driving car, are also known as Autonomous driving system or driverless cars, a vehicle that is capable of sensing its environment and moving with define lane without human interaction. Self-driving cars technologies mostly involve computer system by automating the vehicle control parts. These technological parts posses a range of competences, from forward-collision warning and antilock brakes to lane keeping and adaptive cruise control, to fully automated driving, Autonomous cars combine the variety of sensors, actuators and cameras. Benefits of automated cars are predicted to increase traffic flow and provide enhanced mobility for all users. The basic fundamentals are in High definition maps, Localization, Perception, Prediction, Planning and Control of vehicle as follows.

**Origin**

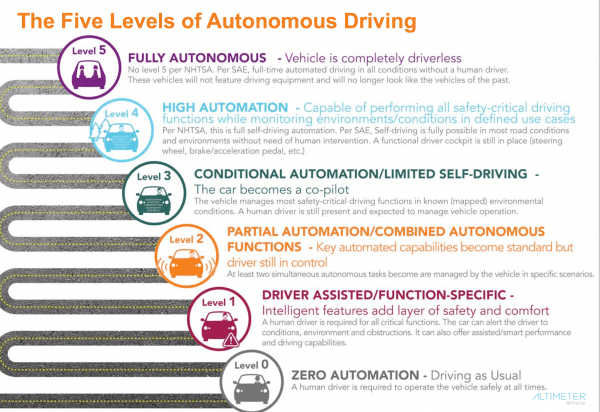
The first era of self-driving cars started in 1920’s. There are a lot of development and creating new technology in later 1960’s. The ALV projects were conducted by the Robotics Institute of Carnegie Mellon University NavLab. By 1994, the double robot vehicles called Vita-2 and VaMP of Daimler-Benz and Ernst Dickmans demonstrated autonomous driving in free lanes.

In 2004 the DARPA (The Defense Advanced Research Projects Agency) conducted the challenge

To self-driving cars to complete the course, But no one did that. In 2005 the second challenge was conducted in that Sebastian Thrun led his team have completed the course. The fully efficient self-driving car was introduced by Toyota Prius modified with Google’s experimental driverless technology was licensed by the Nevada Department of Motor Vehicles in May 2012. The first license issued in the United States for a self-driven car.

**Levels of Automation**

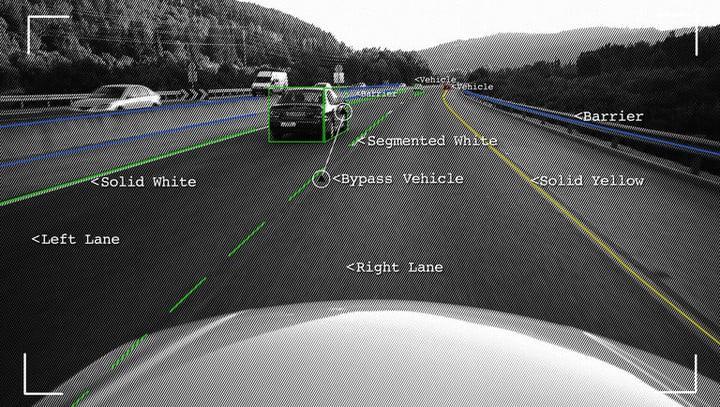
There are five levels of automation system that enhanced the self-driving system.

****

The zero automation is base level automation system it was a soul decision matter Human driving system. The first level automation is driver assistance, some intelligent aspects were included in it, and driver was semi-engaged. The second level of automation is partial the automatic cruise control and Automatic lane keeping system. The third level is conditional automation, Human interface in needed whenever necessary. The fourth level is High level automation, there is no human interface. Fifth level is fully autonomous Vehicle that no human and his interface are not needed.

**High-Definition Maps**

The High-Definition maps are not like Normal maps or route maps. The HD maps are more important for self-driving cars. They have a higher level of accuracy of objects, lanes and locations up to 10cm. It contains a huge amount of driver assistance information, Three-dimensional representation of road network, layouts intersection and location of the sign boards. It helps to solve the localization problem, figuring out where exactly the car in the world. It also recognizes the shape of objects. HD maps are core of self-driving cars.



The pre-processing and coordinate transformation is needs to collect data and compare with HD maps. The uniform coordination system is used in most vehicles. The *Region of Interest*, the purpose of this section is to build a program that can easily identify the lane lines in a picture or a video frame from camera. In that we have to convert this image in grayscale is that processing single channel is much faster than three channels RGB and is less computation intense. Planning with maps and planner identify possible routing options. The Maps are also containing information related to the source of data which sensor was used to get the information when the map was last updated.

**Localization**

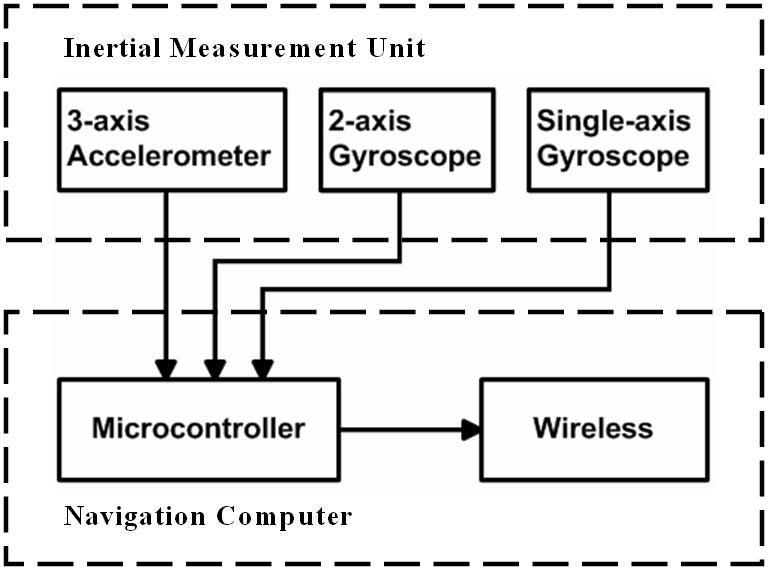
Localization in means the car was where in an exact location. Sensor and the maps are collecting the data to find an exact location. Vehicle coordinate frame and map coordinate frame are vice-versa.

**A) GNSS-RTK**

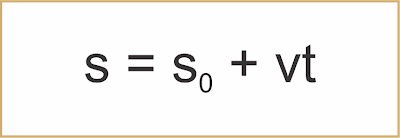
The GNSS is known as *Global Navigation* *Satellite System,* There are 30 GPS satellites operating in outer space in given time each was located on 20,000 kilometers away from the surface of the earth. The control system is specified around the earth for controlling satellites. The RTK is called as *Real Time Kinematic* positioning system is also a satellite navigation system used to precision position data from satellite-based positioning system. But the RTK based system was being issues with tall buildings. It was also low frequency updates like 10MHz. The GPS that equipped in the car can update its location in 0.1 seconds.

**B) Inertial Navigation**

An Inertial navigation system is a navigational device that uses a computer, motion sensors (accelerometers) and rotational sensors (gyroscopes).

****

That can be continuously calculated by dead the action or process of calculating the position, the orientation and the velocity (direction and speed of movement of vehicles)



**S0 = Initial Location**

**V = Velocity**

**T = Time Taken**

We are using 3-axis accelerometer to define the acceleration of the car at any point of the time and also measure the velocity of the current position. Gyroscopes are used to measure the relative position of the spin axis and the three external Gimbals to measure *initial measurement unit* IMU.

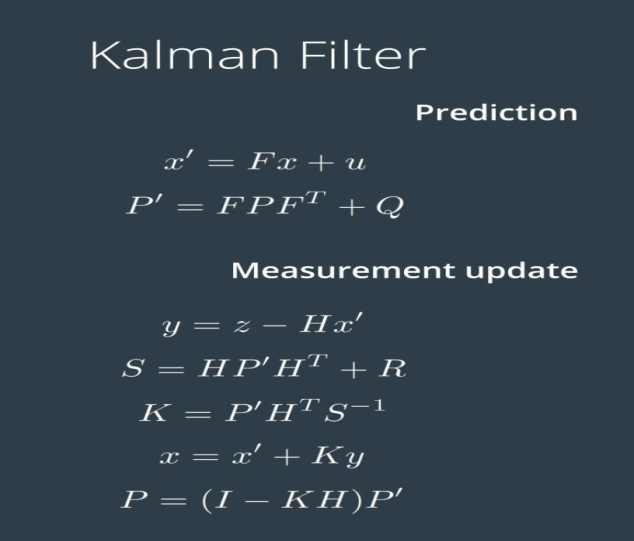
**C) LIDAR Localization**

LiDAR (Light Indication Detection and Ranging) a means of point cloud matching. This method continuously matches with the detected data from Lidar sensor with an HD maps. There are many algorithms to matching point of clouds. *Iterative closest point* ICP is a first approach, filter algorithms are another approach of Lidar localization.

**Kalman Filter** is an algorithm is used to find assume state which was based on the last state in new sensors measurements.



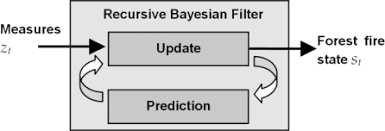
An extended *Kalman filter* algorithm is to pose by registering 3D point clouds against *Gaussian* mixture multiple solution-maps. This method was proposed on two driverless cars in terrible weather conditions and presented localization estimation errors of about 0.15 m.



In statistical method of control system the Kalman filtering is also called as *Linear Quadratic* *Estimation* (LQE), this type of algorithm that uses a series of measurements observed over time, and contains other inaccuracies and unknown variables.

**E) Camera-Based Localization**

In camera based localization in self-driving cars is used to estimate the location of the car and relative to map. A *Recursive Bayesian* filter algorithm is used to perform to find inferences in graph by exploiting its structure and the model of how the car moves, as measured by the visual odometry.



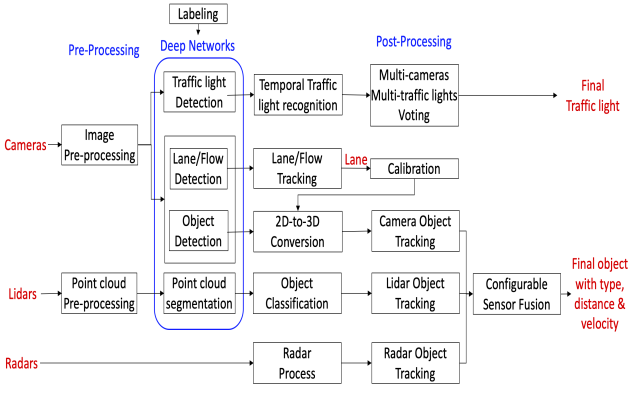
This algorithm is able to point-out the car’s position in the graph and increasing the probability that the current pose lies in a point graph that is correlated with latest car movements.

**Perception**

Perception is a tough task in controls of self-driving cars. The perception module has been upgraded completely to handle comprehensive sensor fusion of sensor.

The perception module incorporated the capability of using multiple cameras, radars and Lidar to recognize obstacles and fuse their individual tracks to obtain in final tack list from the controller. The obstacle, sub-module detects, classifies and tracks obstacles. The sub modules are also predicts obstacle motion and position information. For lane keeping we had a lane instances by post-processing lane parsing pixels and calculate the lane relative location to the vehicle.

In the core concepts of self-driving cars are Detection, Classification, Tacking and Segmentation. The *Detection* in the means of detects the object the capture images by cameras or the Lidar inputs. The *Classification* is a process that was done by some Neural Network algorithm and classify in certain manners. The *Tracking* in the means of the tracks the objects from the car like and their velocity, distance and some other aspects. The *Segmentation* in the means of clarifies the each pixel form the camera images and semantic category.



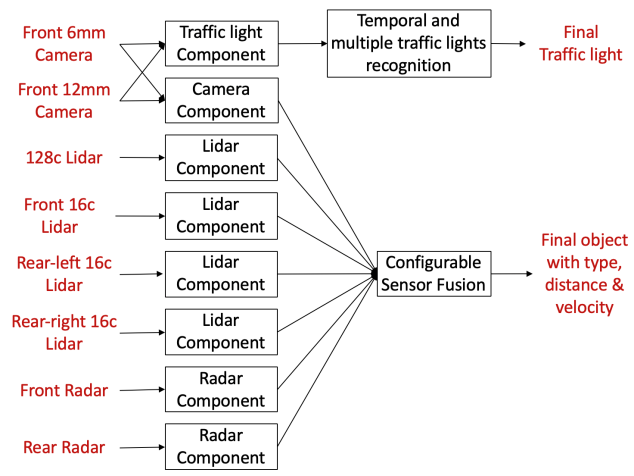
**A) Camera Images**

The camera images are the common data; the images are comprised of pixels. Which is called small units of color, in every pixel of an image is just a numerical value, that values are comprised into an image matrix.

Color Images are more complex. Color images are constructed as Three-Dimensional cubes of values each cube is a Height, Width and Depth of the value.

**B) LIDAR Images**

The Lidar images are getting from the sensor which creates the point cloud on the environment and defines the objects around it. The Lidar works by the laser is coming out of it and getting back with the modified frequency that makes it measure distance.



**C) Machine Learning**

Machine Learning is extremely used to find out the solution to various problems that arise in the manufacturing of self-driving cars. With the inclusion of sensor data processing in an Electronic Control Unit (ECU) in a car, it's essential to enhance the utilization of machine learning to accomplish new tasks.

The **Supervised Algorithms** make a training dataset to learn and they continue to learn till they get to the level of confidence they aspire to reduce the probability of error. Supervised learning is also sub-categorized into regression, classification, and detection or dimension reduction.

**Unsupervised Algorithms** are another set of machine algorithms that fall between unsupervised and supervised. There is a target label in supervised learning; there are no labels in unsupervised learning, **Reinforcement learning** consists of time-delayed and sparse labels for future rewards.

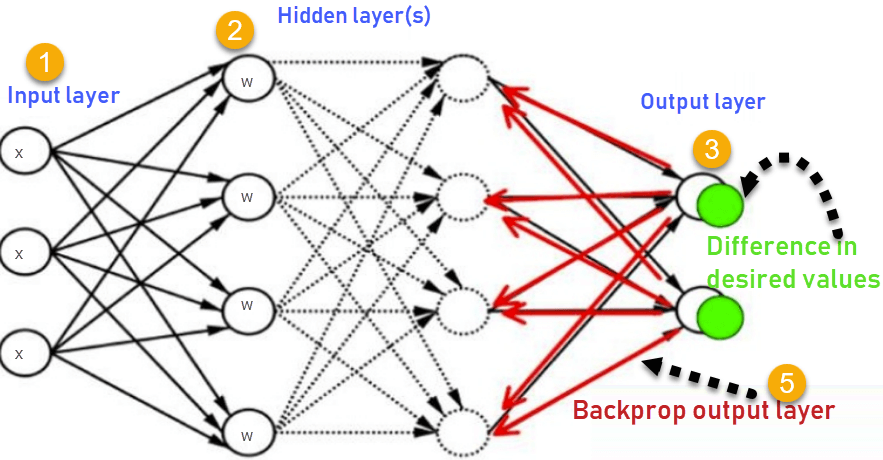
**Regression** is also a kind of algorithm to predict functions. The Regression Analysis evaluates the relationship between two or more variables to collate the effects of variables on distinct scales and is driven mostly by the metrics.

**Neural Networks**

An Artificial Neural Network is a tool to learn complex patterns of data. Neural Network is comprised of a large number of neurons. For neural networks the most basic representation of an image “The Pixel value of the Image”.

**A**) **Back Propagation**

The learning is also called Training. It consists of the step cycle.

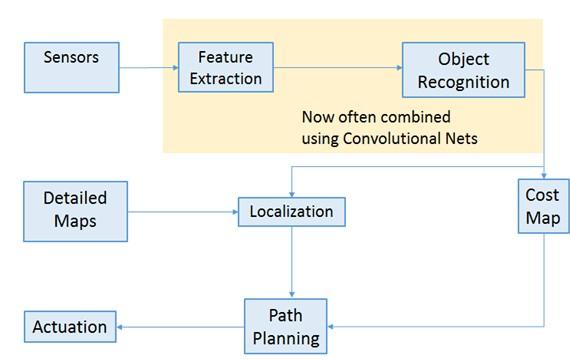


1. **Feed Forward** – Feed each image to Neural Network (n.n) to generate the output value.
2. **Error Management** – The difference between Ground Truth and generate the output value.
3. **Back Propagation** – We sent the error to back through the Neural Network feed to forward on the reverse.

B) **Convolutional Neural Network**

The Convolutional Neural Network is perfect solution to the Perception problem. The input values for CNN are multi-dimensional values, Including two, and three-dimensional shapes the act defines most of the sensor data. Reshaping the Image Matrix into the Vector Columns into a Giant Row.

C) **Region-Based Convolutional Network**  The Region-based Convolutional Network (RNN) gets excellent object detection accuracy by the deep convolutional network to classify the object's proposals. R-CNN has notable drawbacks. I. Training is a multi-stage pipeline R-CNN is work to fine-tune the ConvNet on object proposals using log loss. Then, it fits Support Vector Machines (SVM) to ConvNet features. These SVMs are acting as object detectors, replacing the softmax classifier learned by fine-tuning. In the third training stage, bounding-box regressors are learned.

****

1. **Training is expensive in space and time**

For SVM and bounding box are the regressor training, features are extracted from each object proposal in each of the image and written to the output. The very deep neural network, such as the VGG16, in this process it takes a 2.5 GPU-days for the 5k images of the VOC07 trainval set. These features require hundreds of gigabytes of memory and storage.

1. **Object detection is slow**

At last test-time that features are extracted from each object proposal in each test image, Detection with VGG16 takes 47s / image on a GPU.

**D) Tacking**

After detect the object it was been in tracking. If we detect the every object and every frame and identity each of the object with the *Boundary Box*. If the identity gets the conformation if we match all the objects detected in previous frame with objects. That objects detects in the frame by finding objects with the higher similarity.

**E) Segmentation**

The semantic segmentation involves the classifying on each pixel of the image.  *Fully* *Convolutional Network* (FNN). In that FNN is replaces the flat layers at the end of a traditional CNN architecture with a convolutional layers. The first part of the network is called an encoders and fetches on input image. The second half is a decoder it applies to output.

**F) Region of Interest**

The region of interest which was based on the object detection on the real-data input to a point-cloud data.

1. **Single Shot Detector**

The SSD approach is based on the feed-forward convolutional network that produces a fixed-size collection of *bounding boxes* and scores for the presence of the object with the classes instances in those boxes, it followed by a non-maximum suppression step to produce the final detections. The Early network layers are based on a standard architecture used for the high quality image classification technology, which we will call the base network. We then add auxiliary structure to the network to produce detections with the following structures.

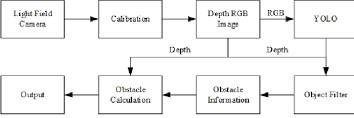
**Multi-scale feature maps for detection,** in that we add convolutional feature layers to the end of the truncated base network.

**Convolutional predictors for detection**, in this approach each added feature layers can produce a fixed set of detection predictions using a set of convolutional filters.

1. **You Only Look Once (YOLO)**

In this YOLO, a new type of a new approach detection technology in the self-driving cars. YOLO predicts multiple bounding boxes per grid cell. At training time we only want one bounding box predictor to be responsible of each product. First, YOLO is extremely fast in object detection technology. Since we frame detection as a regression problem we don’t need a complex pipeline. We simply run our neural network on an image at a test time to predict detections.

Our base network runs at the 45 frames per second with no batch processing on the Titan X GPU a fast version runs more than 150 fps.



**Prediction**

The prediction module studies and predicts the behavior of all the obstacles detected by the perception module. Perception receives obstacle data along with basic perception information including positions, headings, velocities, accelerations, and generates predicted trajectories with probabilities for those obstacles.

Prediction needs to be real time, latency as small as possible Accuracy, Predictions is also been value on learning a new behaviour of vehicles.

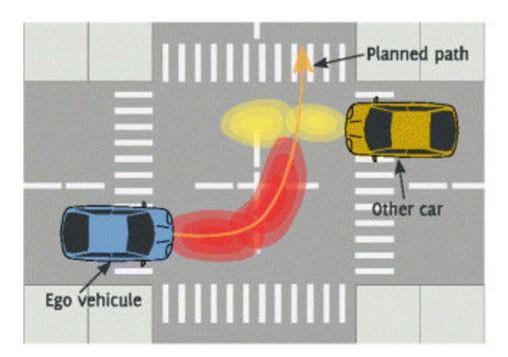
1. **Model-based Prediction**
2. **Driven-based Prediction**

**Model Based Prediction**, one model is describes the moment of a vehicle turning right. The another one that describes the movement of the vehicle on continuing straight.

**Data Driven Prediction**, It was used by the machine learning to train a model based on the observations once on the model is trained and make predictions in the real world.

**A) Lane Sequence Based Predictions**

The lane sequence based predictions we have to divide the path into multiple segments.



Autonomous vehicles are equipped with many advanced sensors that allow them to perceive other vehicles, obstacles, and pedestrians in the environment. If any obstacle status, we have known on predict the state, Initially we have to know the state of an object.

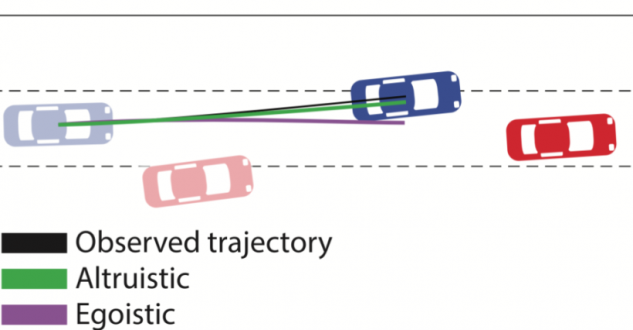
The classical approach of probabilistic graphical models, such as *factors graph* *spatiotemporal graphs*, and the dynamic Bayesian networks, which bring graphical models into the sequential modelling space, is widely used in self-driving cars community for many reasons including their interpretability and the high level structure, which can capture various relationships between features to modelling temporal sequences.

**B) Recurrent Neural Networks**

An approach that takes special advantage of time-series data (Back Propagation). Apart from its standalone utility. Input monolithic and relativity simple SSD model provides a useful building block for larger systems that employ an object detection component. A promising future direction is to explore its use as part of a system using Recurrent Neural Networks to detect and track objects in Video simultaneously.

**C) Trajectory Generation**

The Trajectory planning was a final step of the prediction process. We can get constraints that will eliminate most of the candidate trajectories. We make an assumption that the car will align with the center to the target lane.



In the above figure the Path planning for autonomous vehicle becomes possible after technology considers the urban environment in a way that enables it to search for a path. Put simply the real-life physical environment is transformed into a digital configuration or a state space. Path planning technology searches for and detects the space and corridors in which a vehicle can drive.

**Planning**

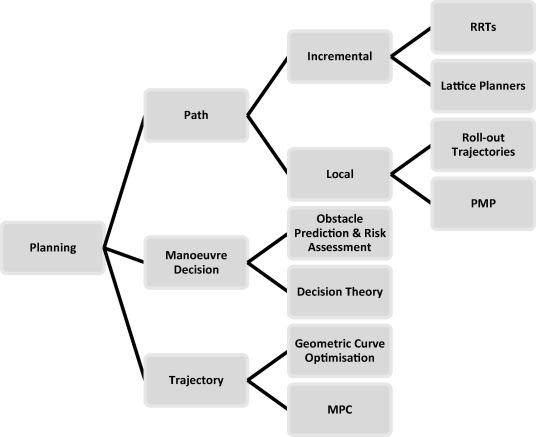
The Planning is a base of Routing. The routing takes the map data as input and output a navigable path.

**Route Module,** Trajectory planning how we make the subtle decisions to avoid obstacle and create a smooth ride for the passengers.

**A)Routing**

Which was planning to go from A to B. It needs three inputs.

* **Map**
* **Current position on the map**
* **Destination**

****

**B) Graph Analysis**

Graph is not the state-space graph, In fact, unlike the state-space graph in which a plan is a path through the graph. Planning graph is essentially a flow in the network flow sense. Planning Graph are closer in spirit to the *Problem Space Graphs* (PSG).

Nodes – Section of Road

Edges – Connection between on those sections

**Constraints,** In the real-world it was plenty of constraints, that was a major use of trajectory to a collision free, obstacle free passengers makes to feel more comfortable.

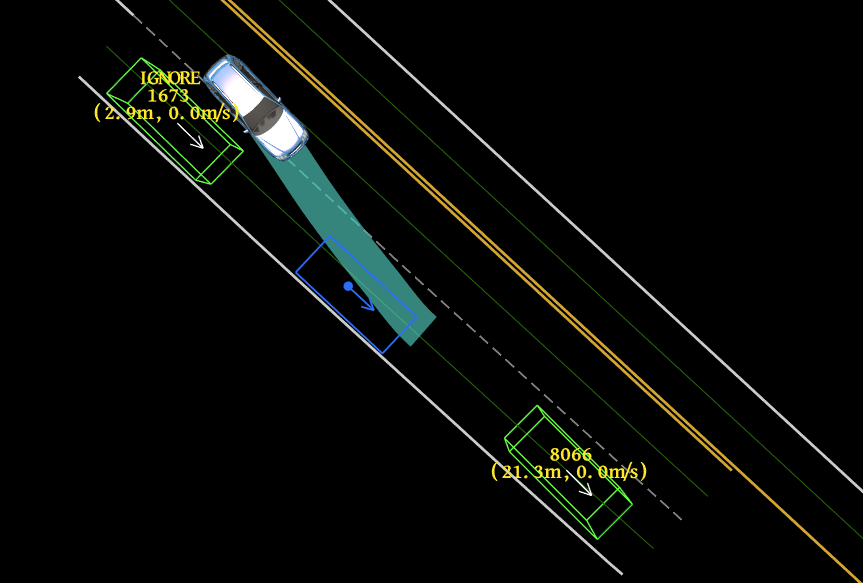
**Frenet coordinates,** It was help us to describe the position of the cars respect to road.

**Trajectory Planning,**  It was most crucial moment of planning of the car the **Path-Velocity** decoupled planning.

* **Path-Planning**
* **Speed-Planning**

**C)Path Generation and Selection**

The path Generation and Selection is next process after it defines all constraints, it was based on the position of the car.



In that above figure the Velocity and the distance of the vehicle are measured before the path generation plan.

**D) Lattice Planning**

The trajectory was in implement in 3d representation longitudinal dimension, lateral dimension, time dimension. And it was a two step process,

* SL – Trajectory
* ST – Trajectory

**Control**

The Control is a main strategy of actuating the vehicle to move it towards the road. The control inputs are Steering, Acceleration, Brake. It especially for safety as planning and control smoothness of driving is the main option to control.

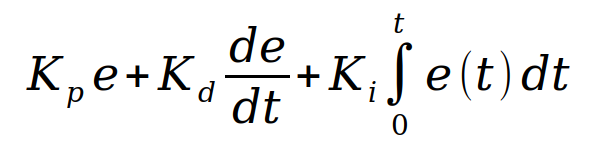
1. **PID – Proportional Integral Derivative**
2. **LQR – Linear Quadratic Regulator**
3. **MPC – Model Predictive Control**

**Control Pipeline**

Two inputs are aspects are *Target* *Trajectory* and the *Vehicle* *state*. The Target Trajectory comes from planning module. Each point of trajectory as designates position (x,y) and velocity(*v)* and the acceleration of the car (a). The vehicle state that determines the position of the vehicle by using localization module. This get data from sensor in the steering, acceleration and brake.

**PID Control**

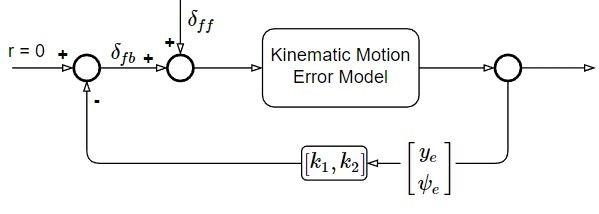
The most important characteristic of autonomous vehicle are their safety and their ability to adapt to various situations and road conditions. We are comparatively comprised three implementations of such controlling methods, a proportional-derivative (PD) controller built with in accordance with the sensors in steering, a PID controller as an extension of the steering control, a controller designed via the most versatile evolutionary computing methods.



In the above equation that defines about the PID controller, where Kp, Ki, Kd refers to to *proportional*, *integral* and *derivative* gains constants respectively. For implementation in discrete form, the controller equation is modified by using the backward Euler method for numerical integration. The term which represents the sampling time ts, is simply eliminated because of multiplication with constants values the PID gains Ki and Kd

**Linear Quadratic Regulator**

In Linear Quadratic Regulator methods for constant and time varying vehicle speed. The latter one is implemented by using a simple gain scheduling method at the grid of the operatings points.



In the design of stabilizing LQR state space control coefficients (*k1, K2*) for the given linear system and obtain the time varying controllers are different vehicle speed as (*k1*(*V*(t)), *k2*(*V*(t))) in matlab.

**Model Predictive Control**

The MPC is an advanced method of process control that is used to control a process while satisfying a set of constraints. It can obtain further speedup by solving planning problems approximately it can also fix barrier parameters and limit the total number of Newton steps. It can also runs on **Kilohertz** rates.

**Conclusion**