AUTONOMOUS DRIVING

Aman Hasan Shaik[1], **Tejavanth Hari Velivelli**[1], **Mubashir Khan Mohammed**[1] **and Bindu Gutta**[1]
[1]Computer Science, Arkansas State University, Jonesboro, Arkansas, USA

Abstract - *Autonomous vehicles are one of the developing technologies now-a-days. These autonomous vehicles research is becoming more flexible to develop and deploy break through artificial intelligence for self driving. Automotive research institutions and startups also play a role in developing this technology. Artificial intelligence coupled with GPU's in the data center accelerate the creation and updating of highly detailed maps for autonomous vehicles. Using cameras, system incorporates deep learning algorithms to detect lanes, signs and other landmarks and can be used to both create maps and determine environment change. Data from the multiple cameras are converted into detailed 3-D mapping using GPU's.*

Keywords : Detection, Localization, Planning, Visualization, CUDA,

1 Introduction

Rushing around, trying to get errands done, thinking about the things to be bought from the nearest grocery store has become a part of our daily schedule. Driver error is one of the most common cause of traffic accidents, and with cell phones, incar entertainment systems, more traffic and more complicated road systems, it isn't likely to go away. With the number of accidents increasing day by day, it has become important to take over the human errors and help the mankind. All of this could come to an end with self-driving cars which just need to know the destination and then let the passengers continue with their work. This will avoid not only accidents but also bring a self-relief for minor day to day driving activities for small items.

Nvidia has already introduced what it calls DRIVE PX 2, "the world's first in-car artificial intelligence supercomputer development platform," which is powered by a software suite they call DriveWorks. While many companies, including Google, are working on self-driving car technology, and companies like Uber hope to have large autonomous fleets active in just a few short years, the development of platforms which can be used by any manufacturer will inevitably mean a faster push toward AI-car dominance. Baidu doesn't only want its tech to be involved, though. It has broader goals: a taxi service of its own. Baidu has already received permission from California to test its AI cars. It would seem yet another major Uber competitor is in the works. The company's foray into automobiles does not stop there, though. They also, along with Ford, recently made a large investment in a company which is trying to supply key autonomous vehicle parts at scale for lower costs.

2 CUDA

CUDA is a parallel computing platform and programming model invented by NVIDIA. It enables dramatic increases in computing performance by harnessing the power of the graphics processing unit (GPU). With millions of CUDA-enabled GPUs sold to date, software developers, scientists and researchers are finding broad-ranging uses for GPU computing with CUDA. Today, the CUDA ecosystem is growing rapidly as more and more companies provide world-class tools, services and solutions. If you want to write your own code, the easiest way to harness the performance of GPUs is with the CUDA Toolkit, which provides a comprehensive development environment for C and C++ developers.

With CUDA, you can send C, C++ and Fortran code straight to GPU, no assembly language required.

Developers at companies such as Adobe, ANSYS, Autodesk, MathWorks and Wolfram Research are waking that sleeping giant the GPU to do general-purpose scientific and engineering computing across a range of platforms. Using high-level languages, GPU-accelerated applications run the sequential part of their workload on the CPU – which is optimized for single-threaded performance – while accelerating parallel processing on the GPU. This is called "GPU computing."

GPU computing is possible because today's GPU does much more than render graphics: It sizzles with a teraflop of floating point performance and crunches application tasks designed for anything from finance to medicine. CUDA is widely deployed through thousands of applications and published research papers and supported by an installed base of over 375 million CUDA-enabled GPUs in notebooks, workstations, compute clusters and supercomputers.

The CUDA Toolkit includes a compiler, math libraries and tools for debugging and optimizing the performance of your applications. You'll also find code samples, programming guides, user manuals, API references and other documentation to help you get started. NVIDIA provides all of this free of charge, including NVIDIA Parallel Nsight for Visual Studio, the industry's first development environment for massively parallel applications that use both GPUs and CPUs.

3 Localization

Localization is one of the most basic and important problems in autonomous driving. Particularly in urban areas,localization precision dominates the reliability of autonomous driving. We use the Normal Distibution Transform(NDT) algorithm to solve this localization problem.To be precise,we use the 3D version of NDT to perform scan matching over 3D point-cloud data and 3D map data. As a result,localization can perform at the order of centimeters,leveraging a high-quality 3D Lidar sensor and a high-precision 3D map. We choose the NDT algorithms because they can be used in 3D forms and their computation cost does not suffer from map size(the number of points).

Localization is also a key technique to build a 3D map. Because 3D Lidar sensors produce 3D point-cloud data in real time, if our autonomous vehicle is localized correctly, a 3D map is created and updated by registering the 3D point cloud data at every scan. This is often referred to as simultaneous localization and mapping.Geometric features (curbs, berms, and bushes) provide one source of information for determining road shape in urban and off-road environments. Dense LIDAR data provide sufficient information to generate accurate, long-range detection of these relevant geometric features. Localization is a key enabling factor for urban robotic operation. With accurate localization, autonomous cars can perform accurate lane keeping and obey traffic laws (e.g., stop at stop signs).

3.1 End to End HD mapping by NVIDIA

NVIDIA offers a end-to-end mapping framework for self-driving autos, intended to help automakers, outline and new businesses to quickly make HD maps and keep them redesigned. This procedure that used to take weeks can now happen in close constant. For car designers, a similar engineering used to make maps and stay up with the latest can likewise empower self-driving autos.

Fig3.1.1

This best in class innovation utilizes a NVIDIA DRIVE™ PX 2 AI supercomputer, combined with NVIDIA Tesla® GPUs in the server farm, to quicken the creation and redesigning of very nitty gritty maps for self-sufficient vehicles. Conventional mapping procedures have required various costly sensors in the auto to gather enormous volumes of information that were recorded and afterward prepared disconnected. Then again, this HD mapping framework is exceptionally effective, moving information handling information into the vehicle, and minimizing correspondence with the cloud. The procedure that used to take weeks can now happen in close constant.

NVIDIA's open mapping stage is based on the NVIDIA DriveWorks programming toolbox for self-governing driving, and intended for carmakers, delineate and new companies to quicken advancement. At the point when simply utilizing cameras, the framework fuses profound learning calculations to identify paths, signs and different milestones, and can be utilized to both make maps and decide when nature has changed.

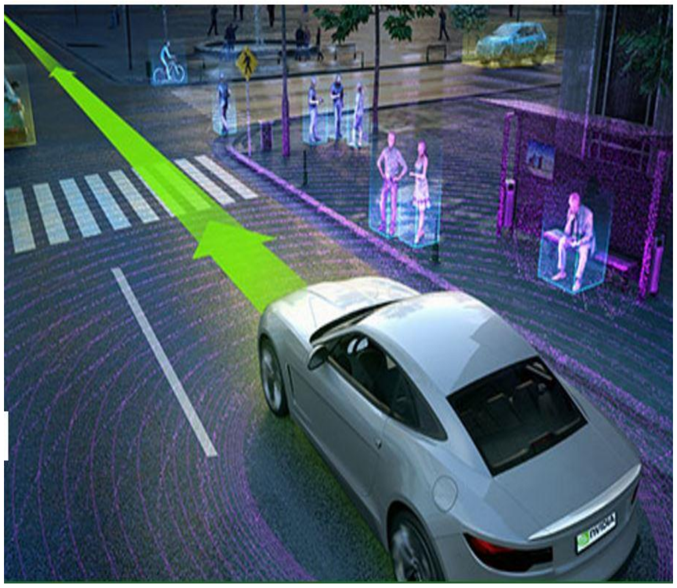


Fig. 3.1.2

For limitation, structure-from-movement calculations empower information from various cameras to be changed over into point by point 3D mapping data. Consolidating information from different inertial sensors in the auto, alongside GPS information and cameras, empowers exact situating of key landmarks.If sought, lidar data can be used to make considerably wealthier maps with more noteworthy detail. A blend of computerized reasoning and VSLAM (Visual Simultaneous Localization and Mapping) handle all phases of guide creation.

Fig.3.1.3

The end-to-end framework is exceedingly adaptable, joining both an in-vehicle AI supercomputer, and a cloud segment in light of Tesla GPUs. For car designers, this same engineering used to make maps and stay up with the latest can likewise empower self-driving autos.

3.2 Road shape estimation

To robustly drive on roads where the geometry is not known a priori, the road shape estimator measures the curvature, position, and heading of roads near the vehicle. The estimator fuses inputs from a variety of LIDAR sensors and cameras to composite a model of the road. The estimator is initialized using available prior road shape data and generates a best-guess road location between designated sparse points where a road may twist and turn. The road shape is represented as the Taylor expansion of a clothoid with an offset normal to the direction of travel of the vehicle. This approximation is generated at 10 Hz.

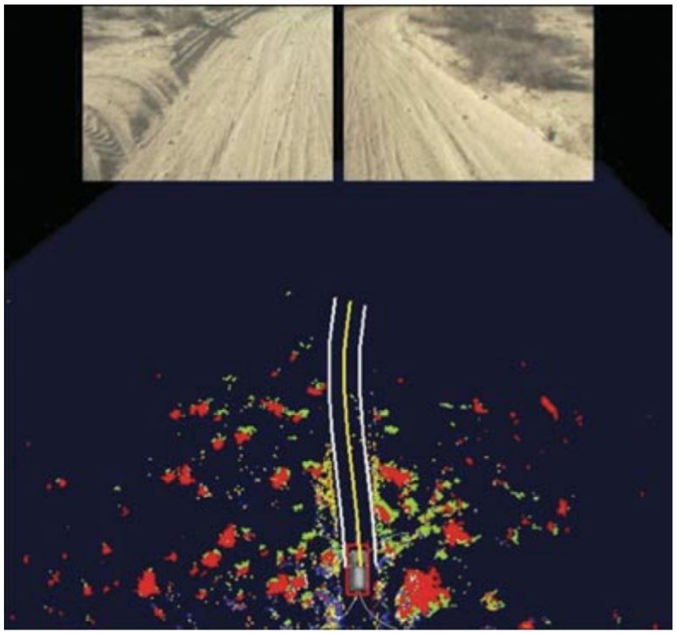


Fig.3.2.1

Two essential components were utilized to decide street area. Checks speak to the edge of the street and are identified utilizing the Haar wavelet (see Static Obstacle Discovery and Mapping segment). At the point when checks are recognized, the estimator endeavors to adjust the edge of the parametric model with the recognitions. Deterrents speak to territories where the street is improbable to exist and are recognized utilizing the obstruction identification framework. The estimator is more averse to pick a street area where deterrent thickness is high.

3.3 Behavioural Planning

Since the road geometry has been considered in the previous layer, the behaviour planner focuses on handling on-road traffic, including moving obstacles and static objects. It takes the traffic-free reference, moving obstacles and all road blockages as input. It outputs controller directives including lateral driving bias, the desired leading vehicle we should follow, the aggressiveness of distance keeping and maximum speed. There are three primary steps: candidate strategy generation, prediction and cost function-based evaluation as shown below.

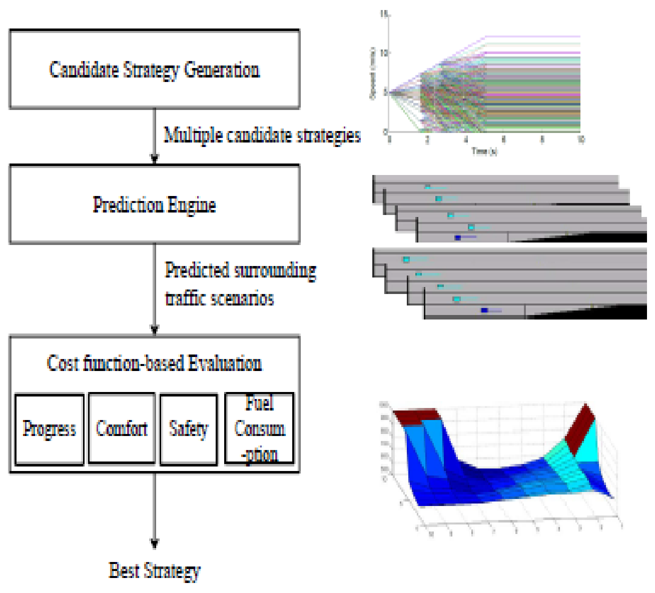

Candidate Strategy Generation
Multiple candidate strategies
Prediction Engine
Predicted surrounding traffic scenarios
Cost function-based Evaluation
Progress
Comfort
Safety
Fuel Consum -ption
Best Strategy

Fig.3.3.1. The block diagram of Prediction and cost function based algorithm(PCB)

3.4 Egomotion(SFM Visual Odometry)

Visual odometry deals with the estimation of the movement of a platform equipped with a camera or a camera rig based on video input alone. No further information sources as wheel encoders or GPS are used. Typical setups include monocular cameras or rely on stereo rigs to increase robustness. Full independence of translation and rotation constraints and a minimized drift is a research goal, in connection with the fulfillment of real-time constraints. A frequent first step is the detection of feature points in each frame of the video input. The found features are matched with subsequent frames based on their descriptors and tracked, if possible. In non-static environments, multiple solutions might occur. In order to decrease their influence, Random Consensus schemes or inlier detections based on stereo constraints are employed . A robust estimation of the movement of the camera based on the tracked features follows. As the relative motion between two frames might be too small, integrating the information over more frames is advantageous. The use of online bundle adjustment to increase robustness is an approach which also seems favorable. All cited solutions exhibit real-time capabilities. Despite their short term accuracy, current solutions suffer from long term drift, starting at drives of more than a few hundred meters. Without other reliable information sources for absolute positioning, such as GPS, only locally smooth results can be obtained. A correction of small angular errors, e.g. in tight bends, is in principle not possible, this leads to large absolute positional errors without a global reference. As the goal is only relative localization, no map is produced. Only if the complete trajectory were kept for bundle-adjustment, could this influence be bounded and a map could be created. Nevertheless, a method based on visual odometry could be used as the starting point of an incremental mapping process or as a smoothing part in combination with GPS, delivering the relative positions for the elements of the map.

4 Planning-VehicleControl, understanding and Path Planning.

4.1 Planning

We all know that an autonomous vehicles advance toward handling real time road traffic. The street outline may differ depending on the traffic and other criterion must be considered explicitly. Some of these situations may include everyday driving methods like merging into traffic flow, passing with on-coming traffic, changing lanes or avoiding other vehicles. This is where trajectory concept come into play, which explicitly account for the time on the planning and execution level. The presented method uses this strategy and transfer velocity and distance control to the planning level. Additionally, the algorithm provides for good obstacle avoidance by the combined usage of steering and breaking / acceleration.

The problem of trajectory tracking is formulated in an optimal sense to take advantage of optimal control theory asserting consistency in the choice of the best solution for trajectory over time. To a car, this means that it follows the remainder of the previously calculated trajectory in each planning step. Bellman's Principle therefore asserts convergence.

While our main criterion in choosing a cost functional is compliance with Bellman's Principle of optimally, trajectories minimizing it must still be close to the desired traffic behaviour of the autonomous car. At the same time, the best compromise has to be found in the longitudinal direction in an analogy manner. Assuming that the car drives too fast or too slow to the vehicle in front, it has to slow down noticeably but without excessive rush. Ease and comfort can be best described by the lateral or longitudinal acceleration : jerk (t) := a(t).

A well known approach in tracking control theory is the moving frame method. Here, we use the Frenet-Serret formulation and apply the moving frame method for combining different lateral and longitudinal cost functional for different tasks as well as to mimic human like driving behaviour. The moving reference frame is given by the tangential and normal vectors at a certain point of some curve referred to as the centre line. The centre line represents either the ideal path along the free road, or the resilt of a path planning algorithm for unstructured environments. Instead of formulating the trajectory generation problem directly in Cartesian Coordinates we switch to the above mentioned dynamic reference frame and seek to generate a one dimensional trajectory for both the root point along the centre line and the perpendicular offset.

4.2 Lateral Motion

Since we seek to maximize comfort and therefore minimize the squared jerk along the resulting trajectory, we chose the start of our optimization according to the previously calculated trajectory. The cost functional

with d1 as the lateral deviation at the end state of the current planning horizon and the weighting factors kj, kt, kd > 0, is independent of the longitudinal movement and therefore velocity invariant. Quintic polynomials can be found to satisfy this cost functional. Instead of calculating the best trajectory explicitly and modifying the coefficients to get a valid alternative, we generate in a first step, a trajectory set by combining different end conditions. In the second step we can pick the valid trajectory with the lowest cost. Notice that , as we continue in each step along the optimal trajectory the remaining trajectory will be the optimal solution in the next step. At extreme low speeds, this strategy above disregards the non-holonomic property of the car, so that the majority of the trajectories would be rejected due to invalid curvatures.For this reason, the behavioural layer can switch below a certain velocity threshold to a slightly different trajectory mode generating the lateral trajectory in dependence on the longitudinal movements.

4.3 Longitudinal Movement

In contrast to previous discussion where time or travelled distance was the key criterion, we will focus here on comfort and contribute at the same time to safety at high speeds, as smooth movements adapt much better to the traffic flow. We also take the longitudinal jerk into account in our optimization problem. Since distance keeping, which describe the transfer from the current position to a longitudinal, possibly moving, target position, we generate a longitudinal trajectory with the following cost functional:

With the distance to the leading vehicle along the centre line s_d.Again, a quintic polynomial satisfies the cost functional.The movement of the leading vehicle has to be predicted within the considered time horizon.Similarly, we can define a target point which enables the autonomous car next to a pair of vehicles before squeezing slowly in between during a tight merging maneuver. For, stopping at intersections due to the red light or a stop sign, the target distance becomes the distance to the stop sign and the targets velocity and acceleration are set to zero. In situations without a vehicle or a stop line directly ahead, the autonomous car does not necessarily have a certain position but it needs to adapt to any desired velocity given by a behavioural level.For this case, the cost functional

Is satisfied by a quartic polynomial. Before combining the lateral and longitudinal trajectory sets, each one is checked against outsized curvatures and acceleration values. The remainders in each set are then brought together in every combination.

4.4 Combining Lateral and Lonigtudinal Curves

In a last step, the conjoint costs of each trajectory is calculated as the weighted sum

Ctot = klatClat + klonClon

As far as our experience goes, it is sufficient for highway trajectory generation to classify all traffic scenarios as merging, following, keeping certain velocity, stopping at a certain point and all combinations thereof, which are conflicting most of the time.In control theory, override control is a well known technique, which chooses among multiple control strategies according to a scheme, prevalently the most conservative one via amax or min operator.An example for a generated smooth trajectory set is shown in below fig.

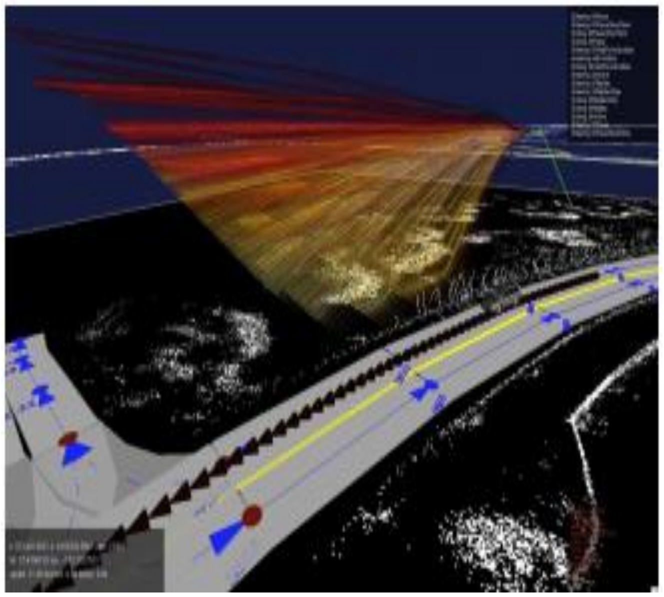


Fig.4.4.1.Smooth trajectory set,the z axis shows the velocity; overall costs are indicated by the color.

4.5 Motion Planner

Motion planning for autonomous vehicle has been developed and substantially improved over the decades. At present, most autonomous vehicles have their own motion planning to directly command a desired trajectory, which includes both desired path and speed. In the Urban Challenge, CMUs Tratan Racing Teams motion planner considered kinematic and dynamic constraints of the vehicle, as well as the moving and static obstacles. It produces number of feasible trajectories from a given sample point with different offsets from the centre of the road with a distance of 10 to 30 meters away from the vehicle and then select the best trajectory from all of them. This mechanism proved to be an efficient way to avoid static obstacles. However, due to the short planning horizon, it was difficult for it to perform smooth lane changes or avoid dynamic obstacles at highway speeds.

To overcome the shortcomings of this planner, multiple layer of trajectories to produce a longer horizon and search more candidate trajectories is proposed. Benefiting from the large number of trajectories, this achieves better performance in obstacle avoidance and circumventing slowly moving obstacles. However, the planner only choose from the limited number of candidate trajectories. Their endpoints have different lateral offsets, but the same heading as the road. In case with sharp turns and in obstacle avoidance maneuvers, sometimes no trajectories provides smooth and human-like performance.

An alternative approach to sample-based planners is based on the path optimization. The trajectories is defined to be the function of control points that are limited.The initial control points are set to be the centre of the road. Then the parameters of the control points are changed based on the multiple cost terms, includes curvature, length of the trajectories and lateral acceleration. The drawback is that optimization is vulnerable to local optima, which can cause very poor performance. It is also computationally quite intensive, with non-deterministic computation time.

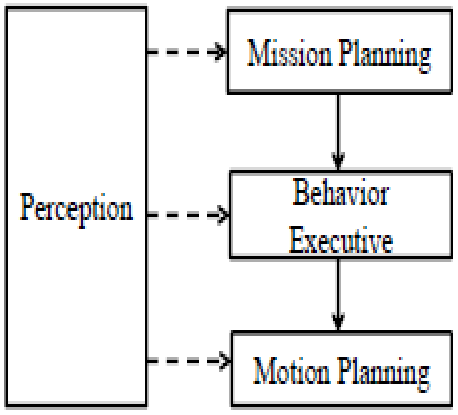


Fig.4.5.1.Hierachical framework for an autonomous vehicle

4.6 Social Behaviour

Although kinematic and dynamic constraints, smoothness of path and obstacle avoidance are frequent concerns in motion planning, the autonomous vehicle ability to socially cooperate with surrounding moving vehicle is usually overlooked. In circumstances such as entrance ramps, lane changers etc. The human driver will make decision based on his/her understandings of what other human drives are thinking or are likely to do. He can also adjust his speed to show other drivers his own intention. This kind of cooperation happens intuitively between human drivers. Not understanding this behaviour can result in incorrect decision making and motion planning with undesirable outcome.

4.7 Hierarchical Autonomous Vehicle Architecture:

To enable autonomous vehicle to finish long term work missions and reduce the workload of the motion planning, a hierarchical architecture is widely used. For each layer of architecture, the input higher-level mission is decomposed into sub-missions and passed on to the next-lower level. Using this framework, many complicated problems becomes solvable. However, the drawback is that layered can cause some performance problems.

One of the shortcomes of this framework is that the higher level decision making module usually does not have enough detailed information and the lower level layer does not have authority to re-evaluate the decision. For instance, when the autonomous vehicle wants to use an oncoming lane to deviate a slow-moving obstacle, the behaviour layer make decision.

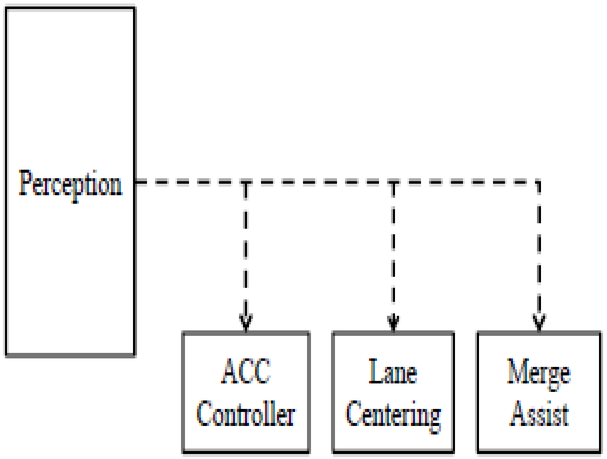


Fig.4.7.1.Parallel framework for an autonomous freeway cruise system

Then it outputs the correct path window and speed car should use to perform this behaviour without interfering with the traffic in the opposing lane. However, the lower-level motion planner may find out it cannot drive at the desired speed because of an upcoming sharp turn, which is not considered in the behaviour layers decision making, so it has to slow down to perform the obstacle.
The other shortcoming is that in this architecture most information is processed in the motion planner, including road geometry, vehicle dynamics and surrounding moving objects and static obstacles. This make planner problem very complicated and computationally expensive. The planner usually needs to sacrifice performance to meet real-time constraints.

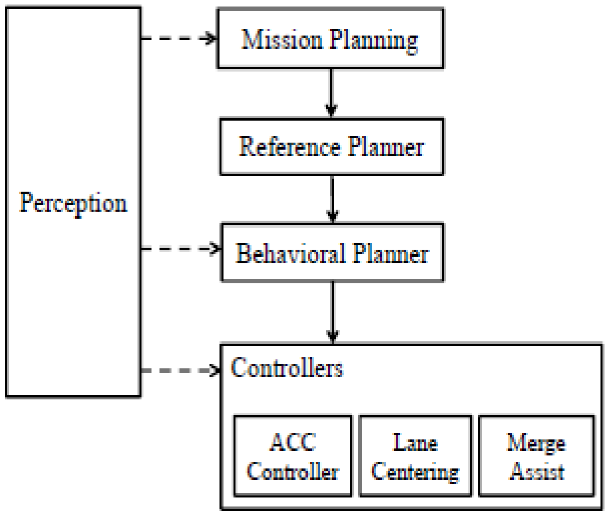


Fig.4.7.2.The proposed behavioural planning framework

4.7.1 Mission Planning

The mission planning module takes charge of dividing driving mission such as “go from point A to point B” into lane level sub mission such as which lane we should be in, whether we need to stop at the intersection or not etc. It takes a human drivers desired destination and computes the shortest path to it from the robots current position. It outputs a set of future lane-level sub-mission which describe the desired lane and turn of the vehicle at each intersection. In addition, this module also controls the transition of goals. When the vehicle completes the current lane-level sub-mission, it will automatically send the next set of goals to the lower layers. When there are intersections with stop signs, traffic lights or yielding requirements, this module directly talks to the perception system to decide whether the car can proceed to next sub-mission.

4.7.2 Traffic-free Reference Planning :

The reference planning layer takes the lane-level sub-mission and outputs a path and speed profile for the autonomous vehicle to drive. In this layer, planner make an assumption that there is no traffic on the road. It uses non-linear optimization to find a smooth and human-like path and speed with consideration of the kinematics and dynamics of the autonomous vehicle, as well as the geometry of the road.

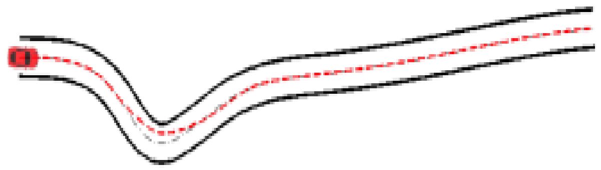


Fig.4.7.2.1

Reference planners result the desired path cut corners at the turn to minimize path length and generate better handling

For example, if the road is not straight as shown in above fig instead of driving exactly in the centre of the road, a human driver will drive slightly offset from the centre line to minimize the required steering, which is emulated in this module. In addition, traffics rules such as speed limits are applied in this layer.

4.7.3 Vehicle Controller Layer

The last layer in the framework contains multiple vehicle controllers running in the embedded controllers in the vehicle, including an adaptive cruise controller and vehicle lateral controller as shown in below fig.

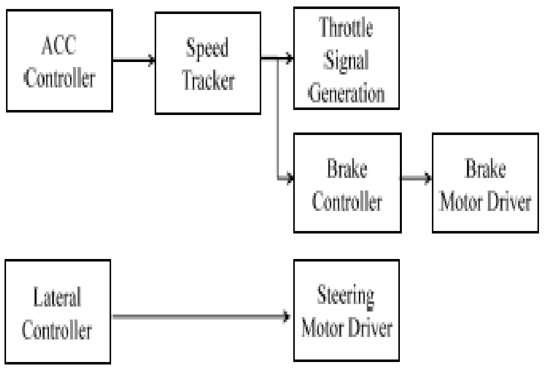


Fig.4.7.3.1.The block diagram of the vehicle controller layer

The lateral controller takes the lateral offset output from the behavioural planner as a directive. It can make the car drive along the road with desired lateral offset to perform lane changes or avoid static obstacles.

The adaptive cruise controller (ACC) uses control laws to compute a desired speed command based on current state of the vehicle, leading vehicle information and control preferences. Then this velocity command is converted to throttle and brake pedal actuation commands by speed tracking controller. Based on the control preferences given by upper layer behavioural planner, it can perform aggressive distance keeping that keeps a closer distance to its leader or more conservation behaviour that stays further away from its leader and applies little acceleration.

5 Experimental results

5.1 Path Planning

The sample-base planners path are usually expressed in polynomials. The heading and curvature for the end points of the trajectory need to be fixed to constrain the polynomial. In cases where the autonomous vehicle drives on the curve and needs to perform evasion methods, the constraints of the sample point limited the flexibility of candidate paths, thereby affecting its performance considerably. Instead of applying these arbitrary constraints, the behavioural planner uses lateral offset to direct the controller to avoid obstacles smoothly as shown in below fig.

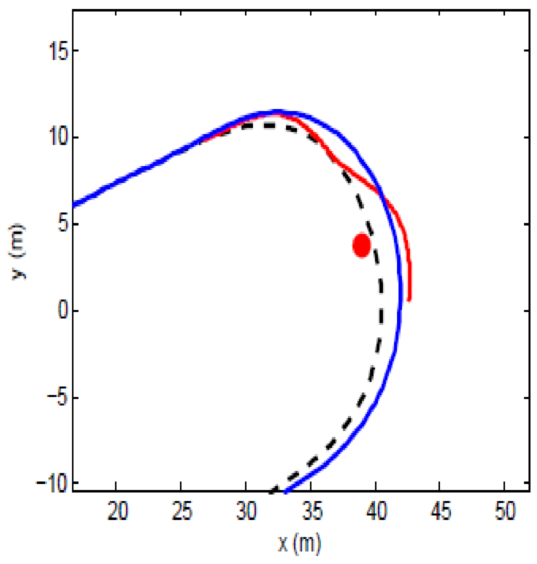


Fig.5.1.1

Comparison of the path planning results: black, centre line of the road; blue, behavioural planners output; red, spatio-temperal sample planners output

5.2 Velocity planning

Comparison between the best speed profile found by the behavioural planner and the spatio-temporal speed-based planne.

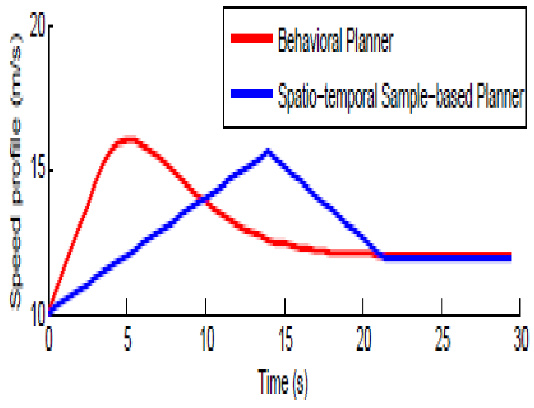


Fig.5.2.1

The above figure shoes the sample space for the normal planner applying searches on spatio-temporal grids, which covers a wide variety of possible speed profiles for the car to execute.

6 Visualization

Visualization is a technique for creating images, diagrams or animations to communicate a message. Visualization through visual image has been an effective way to communicate both abstract and concrete ideas.

6.1 Streaming to Cluster display

Display cluster provides a network-based streaming interface that enables arbitrary applications to be shown on a tiled display. Application provide image buffers which are compressed, sent over the network and receive the data, decompressed and displayed by Display Cluster. This pixel based streaming approach can be integrated into existing application with minimal modification. In addition, unmodified application can be streamed through provided Desktop Streamer application that captures a user-specified region of a desktop. The streaming interface supports resolutions upwards of a hundred megapixels, which makes it ideal for high-resoluiton scientific visualization applications.

Streamed content is displayed in a window on the tiled display in the same way media content is shown. Streams can provide content for a full window or for segments of a window. The set of window segments of a full window can be sent over one or many network connections. This flexibility enables streamed content to be generated and compressed in parallel form a single process or form a set of distributed processes (from below fig.)

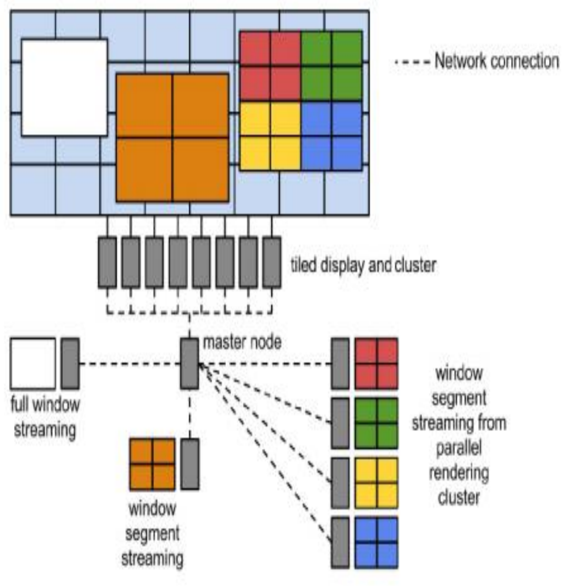

Network connection
tiled display and cluster
master node
full window streaming
window segment streaming from parallel rendering cluster
window segment streaming

Fig.6.1.1

DisplayCluster's streaming capability allows content to be streamed in segments and from one or many network connections.

6.2 Parallel Visualization Applications

Display Cluster provides a modified version of the image composition for Tiles(IceT) library with streaming support. Visualization applications that use IceT can link with this modified version and stream directly to a running Display Cluster instance. No modification of the underlying visualization application is required.

7 Program

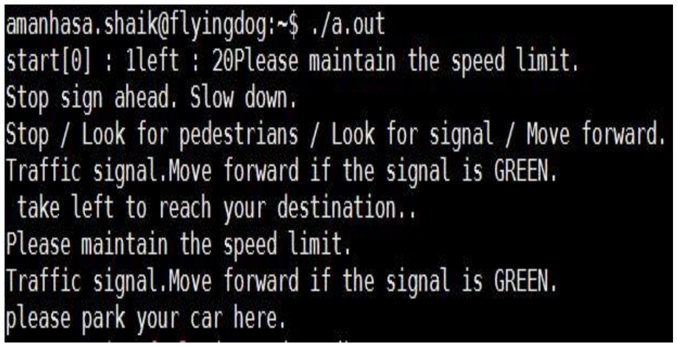
For autonomus car, it senses it ssurroundings using sensors, GPS etc. After sensing them, the data is given to the system of the car. This system analyse the data and forms a 2-D map. Then, the car knows the obstacles in its path. So, using this the autonomous car steers the vehicle in the direction where there are no obstalces.

In our program, we gave a 2-D matrix as input. We also gave a path to follow for the car.

```
#include <iostream>
using std::cin;
using std::cout;

#include <fstream>
using std::ifstream;
using std::ofstream;
#include <cuda.h>
using namespace std;
```

```
#define A 43
#define B 24
//kernel function
__global__ void function(int **map)
{
   int row = blockIdx . x * blockDim . x + threadIdx . x;
   int col = blockIdx . y * blockDim . y + threadIdx . y;
   // Entering the way
   /*for(int i = 0; i < 43; i++)
      {
         printf( "\n");
         for(int j = 0; j < 24; j++)
         {
            printf ( "%d" , map[i][j] , " " );
         }
      }*/

   int start[2], left, right;
   for(int i = 0; i < 2; i++)
   {
      for(int j = 0; j < B; j++)
      {
         if(map[42][j] == 0)
            start[i] = j;
      }
   }
   printf("start[0] : %d", map[42][21 ]);
   left = start[0];
   right = start[1];
   printf( "left : %d" , left,  " right : %d" ,right);
   int temp1 = right + 1;
   int temp2;

   for(int  i = 42; i >= 0; i--)
   {
      if(map[i][right - 1] == 8)
      {
         temp2 = i;
      // printf ("temp2 : %d",temp2 ) ;
         i = 0;
         printf ( "Traffic signal.Move forward if the signal is
GREEN. \n");
      }
      else
      {
         if(map[i][right] == 7)
            printf ("Stop / Look for pedestrians / Look for signal /
Move forward. \n");
         else
         {
            if(map[i][temp1] == 4)
               printf ("Please maintain the speed limit. \n" );
            if(map[i][temp1] == 9)
               printf ("Stop sign ahead. Slow down. \n");
         }
      }
   }
   printf (" take left to reach your destination..\n");
   for(int  i = 15; i >= 0; i--)
   {
      if(map[temp2][i] == 8)
      {
         temp2 = i;
         //cout << "temp2 : "  ;
         i = 0;
         printf ( "Traffic signal.Move forward if the signal is
GREEN. \n");
      }
      else
      {
         if(map[temp2-2][i] == 7)
            printf ("Stop / Look for pedestrians / Look for signal /
Move forward. \n");
         else
         {
            if(map[temp2-2][i] == 4)
               printf ("Please maintain the speed limit. \n") ;
            if(map[temp2-2][i] == 9)
               printf ( "Stop sign ahead. Slow down. \n");
         }
      }
   }
   for(int  i = 21; i >= 0; i++)
   {

      if(map[i][temp2] == 6)
         printf ( "Please maintain the speed limit... parking area
\n" );
      if(map[i][temp2-1] == 3)
         printf ( "please park your car here. \n");

   }


}

int main()
{
    int map[A][B];
   int **dev_map;

   cudaMalloc ((void **) & dev_map, A*B*sizeof(int));

    ifstream infile ("//home//amanhasa.shaik//map.txt");

   if(! infile)
      printf ( "error opening a file : \n");
   else
   {
      for(int i = 0; i < A; i++)
      {
         for(int j = 0;j < B; j++)
         {
            infile >> map[i][j];
         }
      }
/*    for(int i = 0; i < 43; i++)
      {
         cout << "\n";
         for(int j = 0; j < 24; j++)
         {
            cout << map[i][j] << " " ;
         }
      }*/
   }


   //function (map);


   cudaMemcpy(dev_map,       map,              A*B*sizeof(int),
cudaMemcpyHostToDevice);
   function <<< A, B >>> (&&dev_map);

   cudaEvent_t start, stop;
   cudaEventCreate(&start);
   cudaEventCreate(&stop);
   cudaEventRecord(start);


   cudaEventRecord(stop);

   cudaEventSynchronize(stop);
   float milliseconds = 0;
    cudaEventElapsedTime(&milliseconds, start, stop);
   //printf ("\nTotal time taken : %d " ,milliseconds );

    infile.close();
}
```

7.1 Output



```
amanhasa.shaik@flyingdog:~$ ./a.out
start[0] : 1left : 20Please maintain the speed limit.
Stop sign ahead. Slow down.
Stop / Look for pedestrians / Look for signal / Move forward.
Traffic signal.Move forward if the signal is GREEN.
 take left to reach your destination..
Please maintain the speed limit.
Traffic signal.Move forward if the signal is GREEN.
please park your car here.
```

Fig.7.1

This output is the input for the car to follow. The GPS and other sensors collects the data and the data is sent to the GPU to convert the data into a 3D map, the output above is the data generated by the sensors and it formed like a 3D map by GPU and it is given like an input to the car to follow the path or way area.

8 Conclusion

Autonomous driving was an enormously energizing program. The forceful innovation advancement course of events, universal rivalry, what's more, convincing inspirations cultivated a situation that drew out a huge level of inventiveness and exertion from every one of those included. This examination exertion created numerous developments

- a coupled moving obstacle and static obstacle recognition and following framework
- a street route framework that consolidates street limitation and street shape estimation to drive on streets where from the earlier street geometry both is and is not accessible
- a blended mode arranging framework that can both effectively explore on streets and securely move through open regions and stopping parcels
- a behavioral motor that is fit for both taking after the standards of the street and damaging them when fundamental
- an advancement and testing procedure that empowers quick improvement and testing of very fit self-ruling vehicles.

In spite of the fact that this article plots the calculations and innovation that made Boss fit for meeting the challenge, there is much left to do. Urban situations are extensively more confounded than what the vehicles confronted in the Urban Challenge; walkers, movement lights, fluctuated climate, and thick activity all add to this intricacy.

As the field advances to address these issues, we will be confronted with auxiliary issues, for example, How would we test these frameworks and by what means will society acknowledge them. In spite of the fact that resistance needs may give the important to drive these promising innovations, we should endeavor to guarantee our that work is applicable and gainful to a more extensive society. Though these difficulties pose a potential threat, plainly there is a brilliant and non-as well far off future for self-governing vehicles.

9 References

[1] Johann Borenstein & Yoram Koren, Obstacle Avoidance with Ultrasonic Sensors, IEEE JOURNAL OF ROBOTICS AND AUTOMATION, VOL. 4, NO. 2

[2] Yue Wanga, Eam Khwang Teoha & Dinggang Shenb, Lane detection and tracking using B-Snake, Image and Vision Computing 22 (2004) , available at:www.elseviercomputerscience.com.

[3] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski. Self-supervised monocular road detection in desert terrain. G. Sukhatme, S. Schaal, W. Burgard, and D. Fox, editors& Proceedings of the Robotics Science and Systems Conference, Philadelphia, PA, 2006.

[4] S.Kato, E.Takeuchi, Y.Ninomiya, K.Takeda and T.Hamada, An Open Approach To Autonomous Vehicles, IEEE Micro, Vol.35,No.6,pp.60-69-2015.

[5] http://online.qmags.com/MIC1115/default.aspx?pg=62&mode=2#pg62&mode2

[6] Autonomous driving in urban environments: Boss and the Urban Challenge," *Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge* http://www.ri.cmu.edu/pub_files/pub4/urmson_christopher_2008_1/urmson_christopher_2008_1.pdf

[7] http://www.nvidia.com/object/drive-automotive-technology.html

[8] S. Tuohy, D. O'Cualain, E. Jones, & M. Glavin, Distance determination for an automobile environment using inverse perspective mapping in OpenCV, in Proc. Irish Signals and Systems Conference 2010

[9] Ch. Krishnaveni , Ms. A. Siresha , Mr. B. J. Prem Prasana Kumar & Mr. K. Sivanagireddy, Implementation of embedded systems for pedestrian safety using haar features, IJEC: International Journal of Electrical Electronics and Communication,ISN 2048 – 1068,Volume: 06 Issue: 20 l Oct -2014, pp. 761-766

[10] T., Gutierrez, A., Harbaugh. S., Johnston, J., Kato, H., Koon, P.L., Messner, W., Miller, N., Mosher, A., Peterson, K., Ragusa, C., Ray, D., Smith, B.K., Snider, J.M., Spiker, S., Struble, J.C., Ziglar, J., & Whittaker, W.L. (2006). A robust approach to high-speed navigation for unrehearsed desert terrain. Journal of Field Robotics, 23(8), 467–508.

[11] MacLachlan, R. (2005, June). Tracking moving objects from a moving vehicle using a laser scanner (Tech. Rep. CMU-RI-TR-05-07). Pittsburgh, PA: Carnegie Mellon University

[12] Darms, M., Rybski, P., & Urmson, C. (2008b). Classification and tracking of dynamic objects with multiple sensors for autonomous driving in urban environments. In Proceedings of the 2008 IEEE Intelligent Vehicles Symposium, Eindhoven, the Netherlands (pp. 1192– 1202). IEEE