Development of an Autonomous Vehicle for High-speed Navigation and Obstacle Avoidance

Jee-Hwan Ryu, *Member, IEEE,* Dmitriy Ogay, Sergey Bulavintsev,
Hyuk Kim, and Jang-Sik Park

***Abstract*— This paper introduces the autonomous vehicle Pharos, which participated in the 2010 Autonomous Vehicle Competition organized by Hyundai-Kia motors. Pharos was developed for high-speed on/off-road unmanned driving avoiding diverse patterns of obstacles. For the high speed traveling up to 60 Km/h, long range terrain perception, real-time path planning and high speed vehicle motion control algorithms are developed. This paper describes the major hardware and software components of our vehicle.**

I. Introduction

The first autonomous vehicle competition in South Korea organized by Hyundai-Kia Motors took place on November, 2011 [11]. The mission of the competition was unmanned traveling about 4 Km on/off road clearing 7 different patterns of obstacles. Lane keeping and stop within 1 m of crosswalk were also included. Eleven universities in South Korea were participated in the main competition out of 20 initial applicants, and "Pharos", developed by our team, finished the course in 8 min 52 sec clearing all the missions except crosswalk, which got 5 min penalty, and Pharos took 4^{th} place.

Recently there have been many research activities in autonomous vehicle area. Especially, DARPA Grand Challenge [1], [8] and Urban Challenge [10] made a big progress on the area of autonomous vehicle. However, there are still many open issues for high speed unmanned traveling such as long range terrain perception [9], real-time obstacle avoidance and trajectory planning [3], [6], and high speed vehicle motion control [5] et al.

In this paper, autonomous vehicle based on real Sport Utility Vehicle (SUV) is introduced motivated by the competition. The main challenging issue in the development of the vehicle was to build a reliable system, able to traveling high speed up to 60 Km/h through on-road and unstructured off-road while avoiding different types of obstacles. To satisfy these requirements, new methods were developed and extended based on existing methods in the field of autonomous navigation such as long-range obstacle detection and mapping, real-time collision avoidance and trajectory planning, and stable vehicle control on slippery and rugged terrain.

This work was supported partly by the R&D program of the Korea Ministry of Knowledge and Economy (MKE) and the Korea Institute for Advancement of Technology (KIAT). (Project: 3D Perception and Robot Navigation Technology for Unstructured Environments, M002300090)

Jee-Hwan Ryu is with the School of Mechanical Engineering, Korea University of Technology and Education, Cheonan, South Korea jhryu@kut.ac.kr

Dmitriy Ogay is with the School of Computer Science and Engineering, Korea University of Technology and Education, Cheonan, South Korea cactus@ kut.ac.kr

Sergey Bulavintsev, Hyuk Kim, and Jang-Sik Park are with the School of of Mechanical Engineering, Korea University of Technology and Education, Cheonan, South Korea {sergey,perseus,ganggai}@kut.ac.kr

II. Hardware Design

Fig. 1 shows the outlook of Pharos at the competition. Pharos is based on a Gasoline-powered Hyundai Santafe CM. Reinforced front bumper has been installed to protect the vehicle from the environmental impact.

Fig. 1. Pharos was standing at the finishing line on the competition track

A geared DC motor is attached to the steering column to allow electronic steering control. Required torque and rotational speed was estimated through the experimental analysis. We found that 4.41 Nm continuous torque with 3000 rpm was required for controlling the steering column. Smart motor from Animatic Co. with 1:4 gear ratio was selected as a steering motor. With this motor, 5.8 Nm continuous torque with 4000 rpm can be achieved at 24V, which is sufficient to control the steering column.

Fig. 2 shows the assembled 3D CAD model of the steering actuation mechanism. one-to-one pulley powered transmission mechanism was used, and steering angle sensor was moved to the motor output axis to measure the absolute steering angle.

To motorize braking system, another geared DC motor is installed near by the braking pedal. Through the initial experiment, we found that 60 mm travel distance with 8 N pressing force is required max. for controlling the breaking

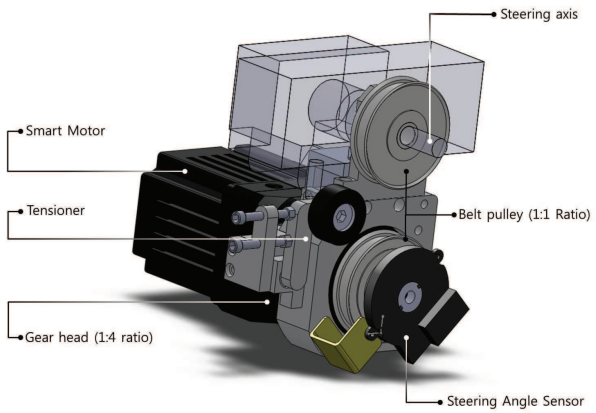


Fig. 2. Assembled 3D CAD model of the steering actuation mechanism

system. one-to-five gear ratio was used to give enough force. It also allows fine position control of the breaking pedal.

Throttle is also modified to be controlled electronically through APS signal modification. APS signal is modified to control the throttle. From 0.8 V to 5 V was given for idling to maximum acceleration.

Vehicle data, such as steering angle, vehicle speed, APS signal and et al. are automatically sensed and communicated to the computer system through Ethernet interface.

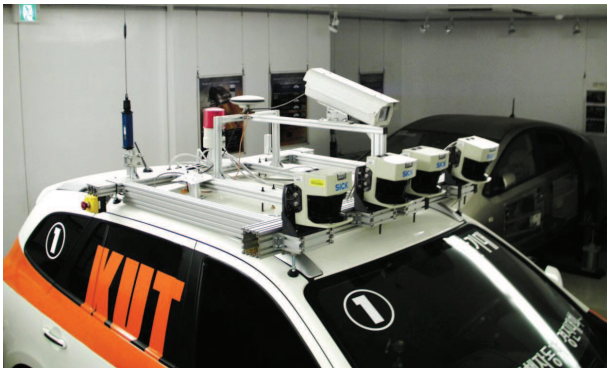


Fig. 3. View of custom made roof rack with sensors

Fig. 3 shows the custom-made roof rack. Most of the sensors are installed on it. Four SICK laser scanners are installed on the roof rack since it can provide best visibility of the terrain. All scanners are facing forward along the driving direction of the vehicle, but with slightly different angles. A single CMOS camera with housing is also installed on the roof rack for crosswalk detection. GPS receiver, RF modem and a radio antenna for E-stop are also installed on the roof rack. The E-stop system is provided by Hyundai-Kia motors for allowing the chasing vehicle safely stop the vehicle in emergent situation with wireless link. Three manual E-stop buttons are installed also on the roof rack and backside of trunk.

Pharos's controllers and communication system are located inside of trunk as shown in Fig. 4. Six Compact PCI, NI-CompactRIO, Gigabit Ethernet switches and various

Fig. 4. Controllers and communication system in the trunk of the vehicle

types of interfaces for physical sensors and actuators are installed on a shock-mounted rack. Custom-made power system with backup batteries are installed underneath of the rack for supplying power to all the electrical components. A six degree-of-freedom IMU is rigidly attached to the vehicle frame underneath the vehicle roof.

The added instrument is required approximately 2.4 KW in addition. Two 12V alternators are installed in addition to supply power, and it provides 24V, 3.4 KW. Therefore, Pharos can even run all the system more than an hour without recharging.

III. Software Architecture

A. Overview

The main principles that we wanted to follow when developing the system architecture were: reliability of a system, ability to distribute software over several independent computers, easy configuration, simulation and development. To reach these requirements we have chosen an architecture, where each module was an independent program that could be run stand alone in a separate process of an operating system. Communication between modules is done through publish-subscribe mechanism.

There is no central process, which encapsulates modules, as it could reduce reliability of the system, if such a process dies. All communication is done through a lightweight messaging server, which supports connectivity through network protocols. It allowed us to easily distribute modules over several computers, and even gave opportunity to run different modules on different operating systems.

This approach also makes development of the modules simpler, because each module is an independent program and can be run stand alone in an environment, where all inputs and outputs can stubbed. For example we could substitute actual data from the sensors by data from file for simulation.

In a running integrated system all blocks are put on watchdogs, so they are restarted if fail, or they can be controlled remotely from the user interface or health monitoring system.

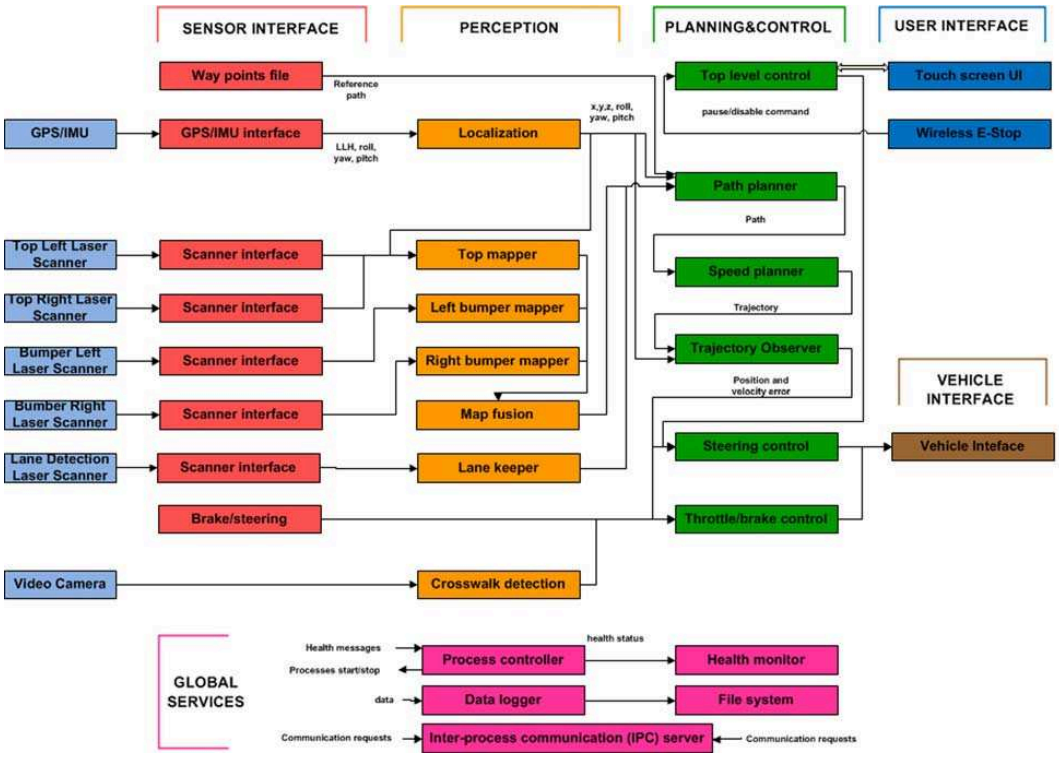


Fig. 5. Overview of the software architecture

B. System Components

Our system has multiple processing layers, when data flow from sensors through consecutive layers, until it reaches steering and braking-acceleration actuators. These layers are: sensor interface, perception layer, planning layer, and control layer. Each layer consists of several modules as shown in Fig. 5.

Sensor interface layer consists of modules that receive data directly from sensors, using sensor protocols, and then publishes the data in a format used inside our system. They run on frequency determined by corresponding sensors, which are 75Hz for laser scanners and 100 Hz for GPS/IMU.

Perception layer has several independent parts that are: localization for estimating vehicle state, laser mapping for generating obstacle occupancy grid map from laser scanner data, and vision mapping for detecting road markings. Data to the path planner are sent at 10 Hz frequency.

Planning layer consists of path planning module that uses 2D obstacle map to generate traversable path, speed planning layer that plans speed taking into account current vehicle speed, speed limits imposed by competition requirements, and shape of the generated path.

Control layer consists of a Trajectory Observer, which calculates difference between planned path and current vehicle position and sends it to the motion control module, which is run in a separate real-time system. Unlike path planner, which is non-deterministic, Trajectory Observer is run at 100 Hz frequency, monitors vehicle position and stops vehicle if no data from path planner come, which can occur in case of a process failure.

All modules also generate health monitoring messages, which are listened by a health monitoring block.

IV. Obstacle Detection and Mapping

To make safely avoid obstacles, Pharos must be able to create a precise terrain map with sufficient range of view for undertaking appropriate actions. The map should contains accurate position of the obstacles to maintain the movement of the vehicle even in the narrow path, comparable with the width of the vehicle. Detection range of the obstacles is considered as well since that faster speed of the vehicle results in the longer perceiving distance. To satisfy these main criteria, medium range laser scanners were chosen. It has 75 *Hz* update rate with accuracy 1cm on the 30 m distance. Laser scanner system of the Pharos consists of three laser scanners, mounted on the roof, tilted downward and two scanners, installed on the front bumper with 45 degree yaw angle with respect to the vehicle to provide wider area of view. Different obstacle classification algorithms are used for scanners installed on the roof and on the bumper. Each laser scanner acquires distance information in its own local coordinate frame. By using estimated position and orientation of the vehicle, data from scanners are transformed into points in global coordinate frame and represent 3D point cloud. Each point in 3D point cloud can be described as (X_m^i, Y_m^i, Z_m^i) where m is the index of the individual scanner and i is the index of each measurement in one scan.

Deterministic algorithm is used for the roof laser scanners because it has shown more reliable performance compared

to other algorithms that had been tested. The classification method is based on the occupancy grid map [2]. The map represents 2D map consists of cells, and each cell contains position information and one of the possible conditions: occupied or unknown. The size of the cells is decided considering maximum resolution at the distance 30 m. To classify a cell as obstacle, two nearby points in 3D point cloud must be found and their vertical distance must satisfy condition $|Z_k^i - Z_l^j| > \varepsilon$, where i and j are indices of two closest points in 3D cloud point from two different laser scanners and ε is the threshold, assumed to being obstacle. If no such points can be found or no value is acquired, this cell is assigned as unknown. Unknown zone considered as drivable. Fig. 6 shows the occupancy grid map, drawn in our campus. Gray color means unknown area or drivable and white color represents obstacles.

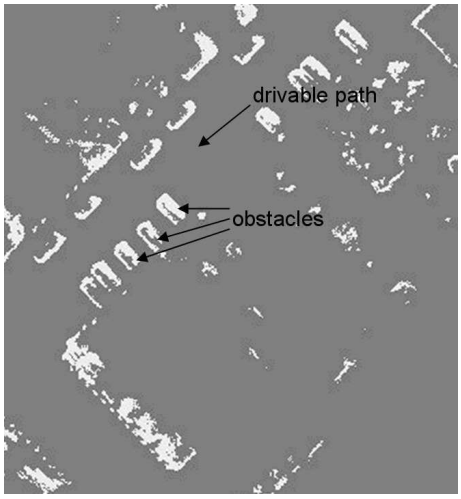


Fig. 6. An example of occupancy grid map

The advantage of this algorithm is that the measurements from two different scanners are used to detect obstacles. This excludes possibility to detect fake obstacles due to the error in the orientation estimation of the vehicle. In addition, it is possible to make six combinations when three scanners are used, therefore reliability of the algorithm can be increased. On other hand, algorithm can be possibly less reliable in case of fault in laser scanner hardware. Bumper laser scanners use different way for obstacle classification. Those are mounted on some known height relate to the ground and scans in horizontal plane. Cell is assigned to be an obstacle if points with same X and Y location satisfies condition $\sum_{j=1}^{n} p_j(Z_j > |h-\delta|) \cdot w > 0.5$, where j is the time index for the series of height measurements acquired for same cell, w is weight of the possibility that point is obstacle and δ is vertical distance that determines size of the obstacle. If same point has been classified as obstacle several times, likelihood of the obstacle presence in this location increases and obstacle is detected.

In practice both algorithms have shown reliable performance and allow our vehicle avoiding obstacles on the distance around 30 m ahead with speed up to 60 km/h.

V. Road Boundary Estimation

One of the problems that can occur during autonomous driving is reference path drifting, due to the shifting of the base coordinates of the DGPS, which varies depend on the various conditions. To avoid it, we develop algorithm, which helps to determine the position of the vehicle with respect to the road boundary and removes the effect of the drifting. On the asphalt road, where static obstacles usually aren't encountered, vehicle has to keep traveling right side of the road. Therefore our task simply was to define lateral distance between car and one of the road sides. To distinguish actual road from off-road, we use reflection properties of the road. Flat asphalt surfaces have lower reflection value than rough terrain. Based on this reflection, occupancy grid can be built. The cell is assumed as non-drivable if reflection value at that location higher than certain limit. Accumulated average reflection value is used. The reference path assumed to be parallel to the road side, but usually lateral offset changes over the time and it involves the unstable behavior of the vehicle when unfiltered offset is used to correct car position. To find lateral distance, one-dimensional low-pass Kalman filter is used. The state of the Kalman filter is distance between reference path and the road boundary. The Kalman filter searches for largest offset along discrete search pattern orthogonal to the reference path. The Fig. 7 shows an example of discrete search pattern. In that case the road boundaries change slowly. Sudden increases in reflection, for example when obstacles appear which normally have higher reflection value, result in small affect in the boundary road estimation. Output from Kalman filter defines offset which is used by path planner, which necessary to prevent road boundary crossing and removing influence of the DGPS drift.

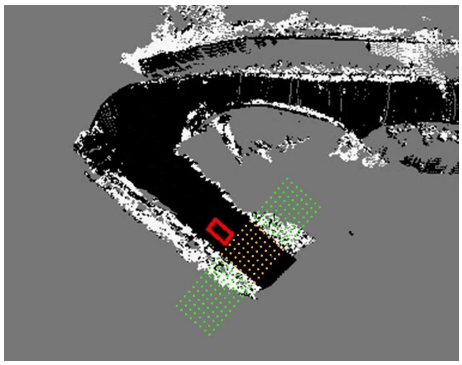


Fig. 7. An example of road boundary estimation

VI. Path and Speed Planning

A. Mission

The main mission that car should accomplish is the following of a pre-recorded reference path. Artificial and natural obstacles can be present on a reference path, so vehicle should avoid them, and also edges of a road can be natural obstacle, because of GPS localization drifting.

The reference path is stored in a text file. Before being given as an input to the path planner, the reference path

is smoothed, and such parameters as tangential vector and curvature of the path are calculated for each reference point.

B. Path Planner

For our implementation, we have chosen RRT [7] based approach, as it can drive vehicle through a very narrow path and can take kinematic constraints of a vehicle into account. But computation speed is a challenge for the development of real-time algorithms [4]. We wanted our vehicle to response faster to the environment changes, because the range of car's perception was limited. And fast path planner response would allow us to increase car's speed.

To cope with those challenges we made some modifications to original RRT algorithm and also developed a new method to increase computing time efficiency of the path planner. It should be mentioned that our path planner is run in a single thread and has re-planning time about 100 ms, which is enough to drive vehicle at speeds up to 60 km/h, given car's perception range.

Original RRT algorithm is modified in several ways, including biased sampling, which is made along lines perpendicular to the reference way points. The state consists of three components, which are: x and y coordinates, and a heading angle of a vehicle.

To plan kinematic behavior of a vehicle we assumed that vehicle travels in short arcs. This let us use curvature of a planned path to calculate lateral acceleration and determine maximum speed, given limited lateral acceleration, which is the safe speed. Then we use time component based on this safe speed in our cost function, which measures distance between states in RRT. The main optimization criteria are fast and smooth motion. Fig. 8 shows an example of the path planning inside of an artificially made narrow tunnel. The area beyond the walls on Fig. 8(a) is valid for sampling in original RRT algorithm, but that area is not reachable by a car. Fig. 8(b) shows that the proposed planner allows us to achieve computing time efficiency by not propagating path to the points that are potentially not reachable.

Before planned path is sent as output it is run through a Savitzky-Golay low-pass filter at first to be smoothed, but mainly to calculate tangential vectors and curvatures of the path at every point.

C. Speed Planner

The main task for a speed planner is to follow maximum safe speed, while taking into account speed limit zones, that are recorded together with reference path, and maximum braking deceleration to avoid collisions if there is some sudden change in a safe speed. As it was mentioned before, safe speed is determined by a curvature of a planned path. And, if there is a sudden change in a path curvature, speed planner should react in advance.

D. Trajectory Observer

Trajectory Observer is a special block, running at a high frequency which measures difference between current state of a vehicle and planned path. Then it sends the data to the control block. There were two reasons to introduce this block. At first it is an interface between the whole system run on a general purpose operating system, which is linux in our case and a real time system. At second it improves safety of a vehicle, as it is deterministic and does have delays or failures like path planner.

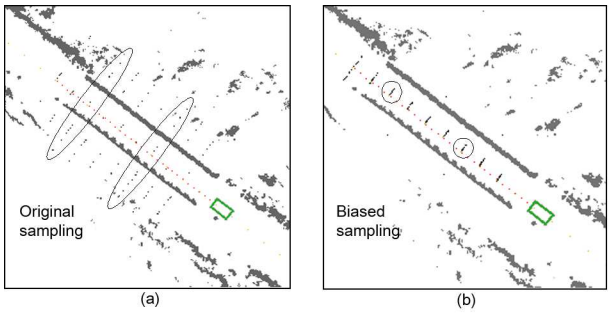


Fig. 8. Comparison of sampling distribution (black dots) of non-optimized algorithm (a), and optimized algorithm (b). Passing artificially made narrow tunnel

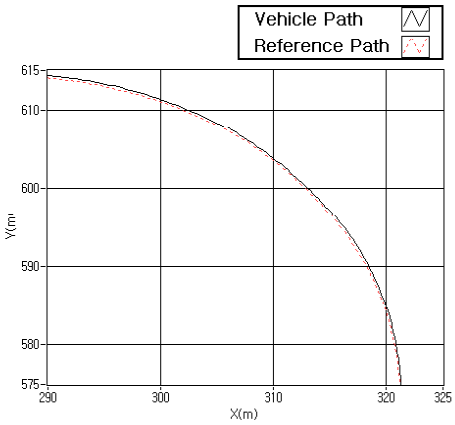
VII. Real-Time Vehicle Motion Control

Once the errors to the reference trajectory of the vehicle is given from the trajectory observer, the motion controller produces appropriate steering, throttle and brake command to achieve the given trajectory. This issue will be described in two parts: steering control and speed control.

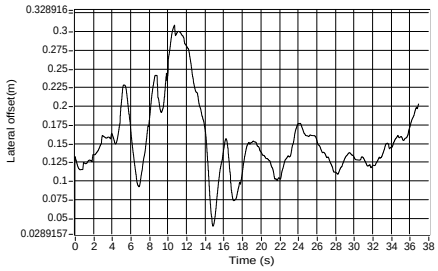
A. Steering Control

Lateral offset, angle difference and curvature of the reference path are given to the steering controller as inputs. The steering controller gives steering commands to the steering motor at a rate of 100 Hz. The basic steering angle control law is very similar to the one in Stanley [8] except look ahead consideration. Simple kinematic based PD feedback controller is used for compensating Lateral offset, which measures the lateral distance of the center of the vehicle's front wheels from the nearest point on the reference trajectory, and angle difference, which is the orientation of the nearest path segment, measured relative to the vehicle's own orientation. Basically, compensating angle difference only allows our vehicle to track the reference trajectory at some level, however without lateral offset consideration it can not compensate steady state error. To copy with the continuous change of the reference trajectory, curvature of the trajectory at an adaptive look ahead point was used as a feed forward control input. Each gains were scheduled to the vehicle velocity, and has proven stable trajectory following on terrain from pavement to off-road, and trajectories with tight curvature.

Fig. 9 shows steering performance while Pharos was making a corner of 40 m radius curved road. Steering controller maintained within 30 cm lateral offset around 60 km/h.



(a) Tracking performance of the steering controller



(b) Lateral offset error of the vehicle

Fig. 9. Steering performance of the vehicle on 40m radius curved road

B. Speed Control

Speed planner and cross walk detection module give speed command to a low-level speed controller. The low-level speed controller translates the speed commands from each modules into actual throttle and brake commands. The minimum of the two recommended speed is always used.

Once the desired speed is determined, low-level speed controller control the brake level and throttle level exclusively. It treats the break and throttle as two opposing single-acting actuators that produce longitudinal force on the vehicle. The controller computes weighted sum plus weighted integration of the speed error. Based on the sign of the computed value, either brake or throttle level is controlled. By considering dead-zone, the controller is able to avoid the chattering behavior. Fig. 10 shows the control performance of the proposed speed controller. It tracks the desired speed very well.

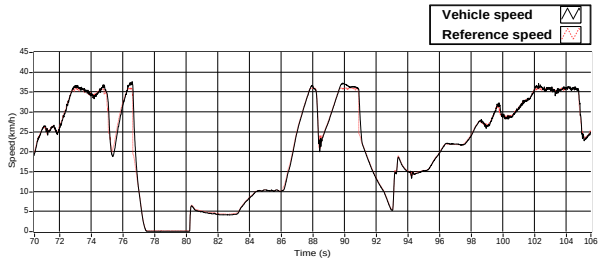


Fig. 10. Control performance of the speed controller

VIII. Discussion

This paper provides an overview of the autonomous vehicle Pharos, developed to participate in the 2010 Autonomous Vehicle Competition in South Korea.

From a broad perspective of view, Pharos's control software mirrors common methodology in the area of autonomous vehicle. However many of the individual modules have developed to achieve the high speed unmanned driving. Long range terrain perception and obstacle detection algorithm, real-time trajectory planning and obstacle avoidance algorithm, and high speed vehicle motion control method are developed and integrated all together. All the developed algorithms are extensively tested in the field to prove the reliability.

Even though our vehicle could successfully finish the race, there are many issues to improve our vehicle further. Only static environment was considered in perception. Pharos is unable to properly interact with moving objects. Perception range was limited within 30 m, which limited the maximum speed of our vehicle. In some case, to avoid an obstacle it was necessary to move up to 6 m to lateral direction from the reference path. If the perception range is limited, vehicle must reduce the approaching speed to avoid an obstacle which has large lateral offset from the reference path. We expect our methodology can allow us to perceive longer range than 30 m.

IX. Acknowledgments

The authors gratefully acknowledge the contribution of Pharos Racing Team for their passion to build our vehicle.

References

[1] M. Buehler, K. Iagnemma, S. Singh, *The 2005 DARPA grand challenge: the great robot race*, Springer Trats in Advnaced Robotics 36, Springer-verlag, Berlin; 2007.

[2] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, Vol 22, No. 6, 1989, pp. 46-57.

[3] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel. "Practical Search Techniques in Path Planning for Autonomous Driving", *in Proceedings of the First International Symposium on Search Techniques in Artificial Intelligence and Robotics*, 2008.

[4] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," *AIAA Journal of Guidance and Control*, Vol. 25, No. 1, 2002, pp. 116-129.

[5] A. Kelly, *A partial analysis of the high speed autonomous navigation problem*, Project report of "Perception for Outdoor Navigation" and "Unmanned Ground Vehicle System," Carnegie Mellon University, 1994.

[6] Y. Kuwata, G.A. Fiore, J. Teo, E. Frazzoli, and J.P. How, "Motion planning for urban driving using RRT", *in Proc. IROS*, 2008, pp. 1681-1686.

[7] S. M. LaValle and J. J. Kuffner. "Randomized kinodynamic planning". *International Journal of Robotics Research*, Vol. 20, No. 5, 2001, pp. 378-400.

[8] S. Thrun and et al., "Stanley: The Robot that Won the DARPA Grand Challenge," *Journal of Field Robotics*, Vol. 23, No. 9, 2006, pp. 661-692.

[9] C. Urmson and et al., "A robust approach to high-speed navigation for unrehearsed desert terrain," *Journal of Field Robotics*, Vol. 23, No. 8, 2006, pp. 427-508.

[10] C. Urmson and et al., "Autonomous driving in urban environments: Boss and the Urban Challenge," *Journal of Field Robotics*, Vol. 25, No. 8, 2008, pp. 425-466.

[11] http://www.hyundai-ngv.com/techcontest/, in Korean.