

EKLAVYA 3.0

Indian Institute of Technology Kharagpur
The Autonomous Ground Vehicle Research Group (Team AGV)
Faculty Advisor: Dr. Debashish Chakravarty
Email: dc@mining.iitkgp.ernet.in

INTRODUCTION

Team AGV, under the ambit of Center for Robotics, IIT Kharagpur, has been pioneering the autonomous ground vehicle technology with the ultimate aim of developing the first self-driving car of India. The team has been participating in IGVC since its inception in 2011 with the Eklavya series of vehicles. Eklavya 3.0 is the third prototype developed for the IGVC 2014 by the team and it features many improvements over the earlier design – Eklavya 2.0 – especially in mechanical performance, electrical efficiency and robustness of software. The model is twice as fast as its predecessor (section Motors and Wheels) while weighing almost the same, has reaction times over 0.3 times lesser than that of 2.0 and is capable of adapting itself to changing environments (section Lane Detection). This report describes in detail the design and development process of Eklavya 3.0.



Figure 1. Eklavya 3.0

DESIGN PROCESS

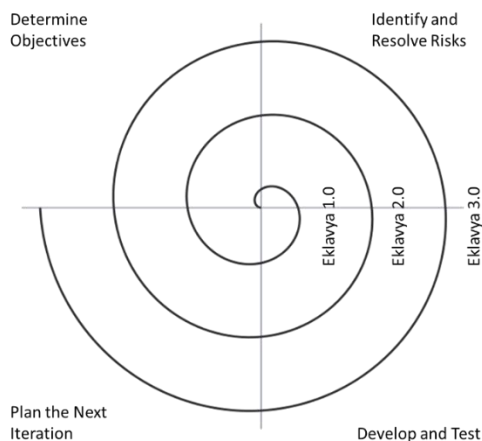


Figure 2. Spiral Model

The best suited design process for Eklavya series is the spiral model and is illustrated in Figure 2. It allows for incremental releases of the product via incremental refinement through each iteration around the spiral. In a spiral model, the estimates (budget, risk etc.) become more realistic as work progresses, as more important issues are discovered earlier. This lets the developers and designers to get started with the model and code at an early stage of development.

In 2011, Team AGV participated for the first time with Eklavya 1.0 in IGVC 2012. Eklavya 1.0 was a skid-steer vehicle but the less powerful motors meant that its maneuverability was limited. The software

consisted a single process and development was carried out on Microsoft Visual Studio 2008. Eklavya 2.0 has been a tremendous improvement over Eklavya 1.0 in all three divisions – mechanical, electrical and software. The chassis has been built using reconfigurable T-slot aluminum rods which made it easier to try various prototypes. Through iterative prototyping, the team has learned to switch from skid-steer drive to single differential, single caster wheel drive which led to maximum maneuverability, especially in rough outdoor terrains like the one in IGVC. Much powerful motors and a state of the art controller have been used to achieve precision for the low level locomotion. The software stack consisted one ROS node which implemented a threaded design which ran all the various modules in parallel. This improved the reaction time a lot in comparison with Eklavya 1.0.

During Eklavya 3.0's design and development, major focus was on finding solutions to the problems faced while participating in IGVC 2013 with Eklavya 2.0. Speed was a major issue in Eklavya 2.0 – the vehicle barely managed to pass the minimum speed test. The electronics could be made more robust with respect to safety. Threaded code is prone to blocking – a common problem due to mutexes. All these and other related issues were addressed in Eklavya 3.0.

TEAM COMPOSITION

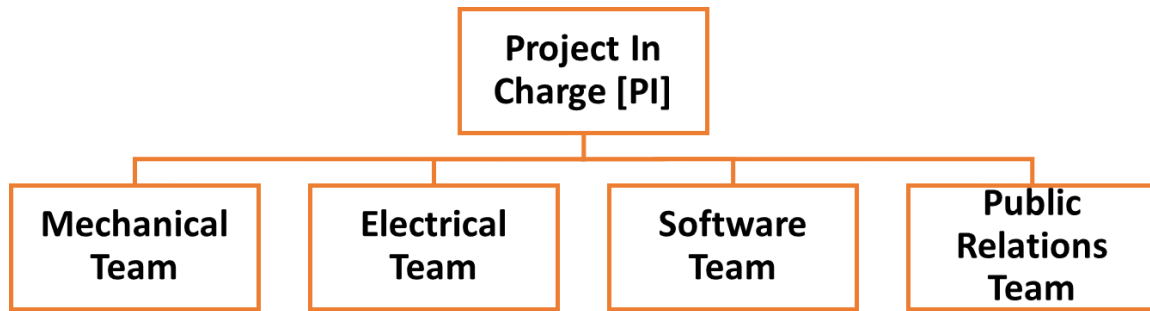


Figure 3. Team Composition

Recently, Team AGV was recognized by the administration of IIT Kharagpur by making it officially a part of the Center for Robotics, IIT Kharagpur. The team consists of more than 40 undergraduate and postgraduate students belonging to interdisciplinary backgrounds viz., Computer Science, Electrical and Mechanical Engineering and is mentored by Professor Debashish Chakravarty of the Department of Mining Engineering, IIT Kharagpur. The team members are pursuing this project out of their passion for robotics as an extra-academic activity. The team spent approximately 2070 person hours in design and development of Eklavya 3.0.

INNOVATIONS

Vehicle Information Display [VID]

Eklavya 3.0 features a display just beneath the switch panels to provide critical but minimalistic information about the state of the robot. It is equipped with anti-glare protection to work properly in sunlight. The VID is interfaced with a Raspberry Pi interfaced to the main processor via Ethernet cable. The display proves vital in debugging where critical responses are appropriately displayed. The displayed information can be one of the following.

1. Steering direction: pointed by an arrow. The speaker would announce the direction of rotation “Taking Left” or “Taking Right”.
2. Current vehicle speed: displayed in *m/s*. The speaker would announce the speed.

3. Special cases in the control flow. These include the cases when no path is found, or an obstacle is too close while starting. In each of these cases the speaker announces an appropriate message.
4. The current battery levels.

Battery Management System [BMS]

Conserving the battery power has been a prime concern while designing the power system for Eklavya 3.0. The charging procedure has been modified with a single plug outlet. Physically detaching the batteries during test runs is not a concern anymore. Three batteries are housed inside the bot with only two working at a time along with an extra redundant charged battery. When there is a requirement of high torque, a relay circuit automatically switches the top two charged batteries. The least two charged batteries are used during normal run, unless, one of the batteries is drained critically when the remaining two batteries are used. The battery level indication and the currently used batteries are displayed on the VID.

Power Management System [PMS]

Eklavya 3.0 features an intelligent power management board that uses custom made circuitry providing power just as much required by the individual modules to operate properly. The custom made ICs helped us improve the efficiency of the power consumed, thus increasing the average run-time of the robot. The power requirements are listed in the form of a chart depicted in Figure 10.

Adaptive Grass Removal

Lane detection algorithm used on Eklavya 2.0 was not robust with respect to sparse lanes. Eklavya 3.0 tries to tackle this problem by deploying a machine learning based approach which focuses on removing the grassy regions in the image and then trying to detect the lanes. The simple SVM based classifier proved to be effective against intensity changes and shadows.

Auto Calibration of IPM

During IGVC 2013, a good portion of efforts and time was spent in calibrating the lane detection algorithm for correcting the inverse perspective transform used to map the image coordinates to the world coordinates. This process has to be repeated whenever the camera position changes and thus was quite annoying. A solution to this was developed and being used during the development stages of Eklavya 3.0. The idea is to automatically generate the warp matrix used by IPM.

The traditional approach is to keep a rectangular shape whose dimensions are already known at a known distance and orientation from the bot and manually detect its corners from the camera feed. The automation script would try to detect the corners of a checkerboard (instead of a plain rectangle), since it can be detected with a higher probability and try to match the distances of obstacles placed at the corresponding corners using the laser scan data which is enough to calculate the warp matrix.

Quick Response and Lattice Planners

In the pursuit of developing the highest reaction rates, the path planner module of Eklavya 3.0 takes a two stage approach. The slower but complete second stage will be triggered only when the greedy yet lighter first stage fails to find a feasible and walkable path. Since the complexity of the first stage is constant for all practical scenarios and the probability of second stage being triggered is quite less considering that the vehicle would be strictly following lanes most of the time, this approach would make the vehicle highly reactive, limited only by the perception's critical delay in addition to mechanical lags.

MECHANICAL DESIGN

Vehicle Design Specifications

Table 1. Vehicle Design Specifications

Design Parameter	Specification
Payload Capacity	20 kg
Dimensions	62.3 cm x 80 cm x 109 cm
Weight	45 kg (excluding payload)
Peak speed possible	4 mph
Average running speed	2 mph
Drive mechanism	Single Caster Differential Drive
Drive	Front wheel drive
Power source	Battery

Drivetrain

The things in the drivetrain that have to be decided before the design process are as follows:

1) Number of powered wheels

A bot can either have 4 powered wheels (skid-steer drive) or 2 powered wheels. Based on the study of several research papers, it was understood that using four powered wheels, i.e. having the skid-steer drive mechanism led to the problem of having indefinite yaw angle which can be eliminated only by having two powered caster wheels where as a 2 powered wheels would not have any such problem. A skid steer drive with 4-powered wheels would have a larger skid friction. This would have to be overcome by the driving wheels and this requires that the motors have enormous torque. Thus, such a model cannot make turns easily. Instead, 2-powered wheels with castors would be a more efficient alternative. Thus 2 powered wheel design was chosen.

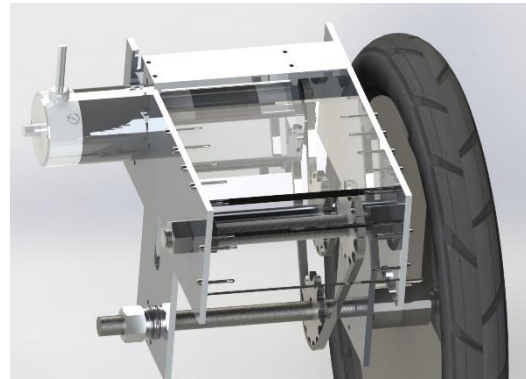


Figure 4. Motor Hub

2) Number of castors

A single caster design was developed on the basis of the observation made on the previous robot Eklavya 1.0 which perfectly showed that a two caster wheel design is prone to swiveling of the two castors in different directions simultaneously in rough terrain. Furthermore, for a bot to be stable it needs to have a 3 point contact. Due to the uneven terrain there is a high possibility that one caster would be in air and it would lead to wheel slips which are not preferred.

3) Placement of Castors

A single caster can be at the front or rear of the powered wheels. When the caster is kept at the front of the vehicle, the bot would go in the direction the caster would lead it to go. Although the direction of the caster is controlled by the motion of the powered wheels, under rough terrain conditions and undulated surfaces, the caster generates its own direction at every moment which makes the robot deviate from the desired path. So, the caster in Eklavya 3.0 is attached to the rear of the vehicle.

Motors and Wheels

The decision of motors and wheels are interdependent due to iterations and availability of motors for the targeted weight and speed the wheels must have a 16 in. diameter and the motor for 15% inclination should have a maximum torque of 10Nm and to achieve the peak speed of 4mph should have 94 rpm. So, we are using a 16 in. diameter wheels and Midwest Motion motors with continuous torque of 12.5 Nm and has 80 rpm. In the construction of the motor hubs there is a gear reduction of 1.2 to get the desired speed.

Chassis

1) Targets

The targets for the chassis of Eklavya 2.2 are:

- Maximum weight of 10kgs.
- Design has to be modular for easy replacement of the damaged parts.
- Accommodate all the sensors inside the bot's chassis frame and be strong enough to carry 2 batteries and a payload weighing a total of 30kgs.
- Waterproofing for the sensors and providing tilt platforms for lidar and camera.

2) Choice of materials

For making chassis under 10kgs weight and be strong to carry 30kg weight the chassis is built using lightweight materials like galvanized 20 x 20 mm extruded aluminum bars, Polycarbonate sheets and Polypropylene sheet. Polypropylene sheets are used for making electrical control boards and electrical circuitry. These sheets were chosen as they are resilient and easy to manipulate. Polycarbonate sheets are used for waterproofing the sensors and as a covering for the bot.

Extruded aluminum bars were chosen for the following reasons:

- An extruded aluminum bar weigh same as an aluminum bar with square cross section of 3mm thickness but has the strength of a solid aluminum bar.
- The extruded bars have the flexibility of fixing any object at any place as the come with inbuilt grooves (Figure 6) along the length of the bars. The grooves have a 2° bend which when fastened to the right extent, the groove keeps pressing against the M-5 and M-6 T-nuts used thus increasing the grip.
- The flexibility of fixing helped create a modular design for the bot. The placement of the lidar, camera mounts can be varied and the size of the castor and other parameters became independent of the design of the chassis.

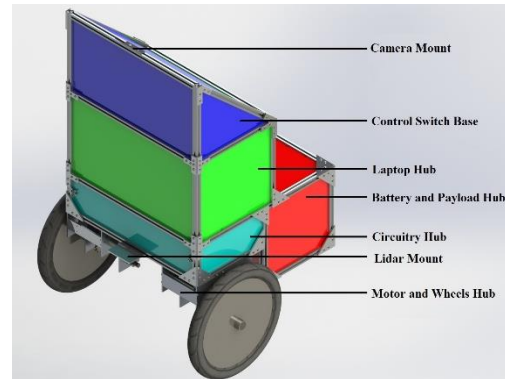


Figure 5. Eklavya 3.0 Chassis

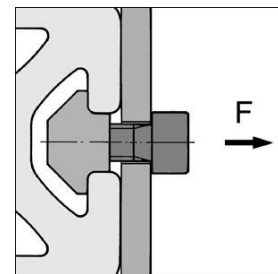


Figure 6. T-Slot

3) Chassis Design

The chassis can be completely divided in the following parts:

- Motor and wheel hubs
- Battery and payload hub
- Laptop hub
- Control switch base
- Circuitry hub

a) Motor and wheel hubs

The 16 in. diameter wheels are kept outside the chassis of the bot as keeping them inside the chassis would raise the center of mass of the bot significantly. The motors and the wheels are connected using a chain and sprocket system so as to transfer the bot's complete weight on to the wheel axles made of solid mild steel. The wheel axles have been simulated under specific the loads and then the dimensions (diameter and length) were decided (Figure 7 and Figure 8). It was found that the under 40kgs weight on a single wheels axle it would have a maximum bend of only 0.25mm and has a factor of safety of 5.5. In each hub the chain passes through 4 sprockets out of which 2 are idlers so as to increase the chain and sprocket contact angle thus increasing the efficiency of load transfer.

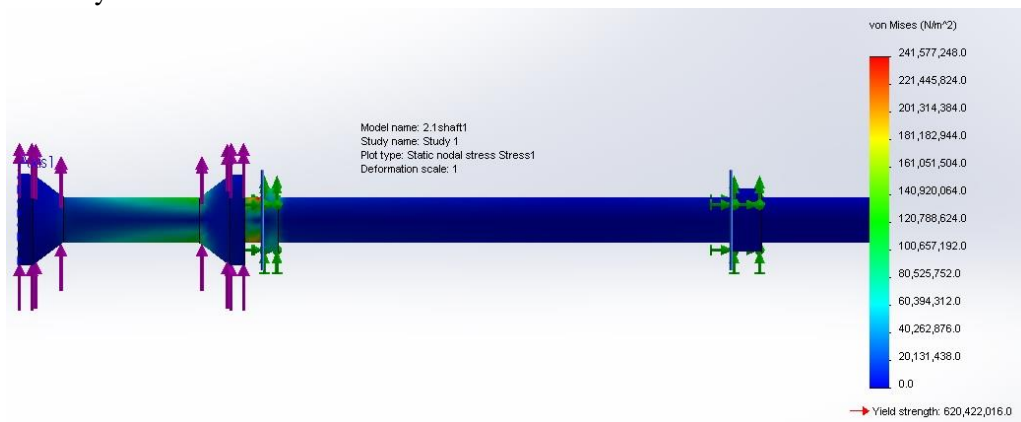


Figure 7. Stress Analysis of Wheel Axle (factor of safety = 5.5)

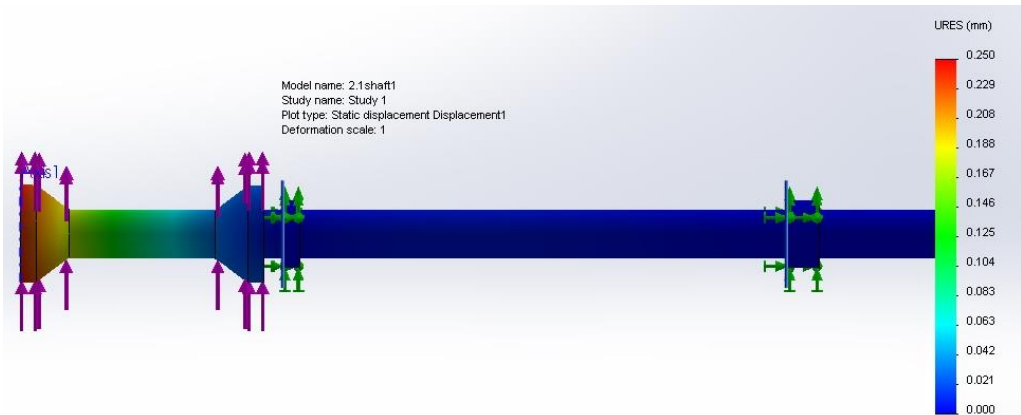


Figure 8. Displacement Analysis of Wheel Axle (maximum displacement = 0.246 mm)

b) Centre of Mass

As a three-wheeled robot with a trailing caster, Eklavya 3.0 turns around the axis passing through the geometric center of the frame, halfway between the front powered wheels. As the wheel hubs weigh a little less than that of the rear part which shifts the center of mass of the vehicle towards rear part, so in order to bring the center of mass close to the center of the vehicle, a laptop hub and a wedge-shaped structure is made on top of the front part of the chassis. The wedge houses the switch control board, cameras, E-Stop switch and safety indicator lights.

ELECTRICAL SYSTEM DESIGN

Embedded System Architecture

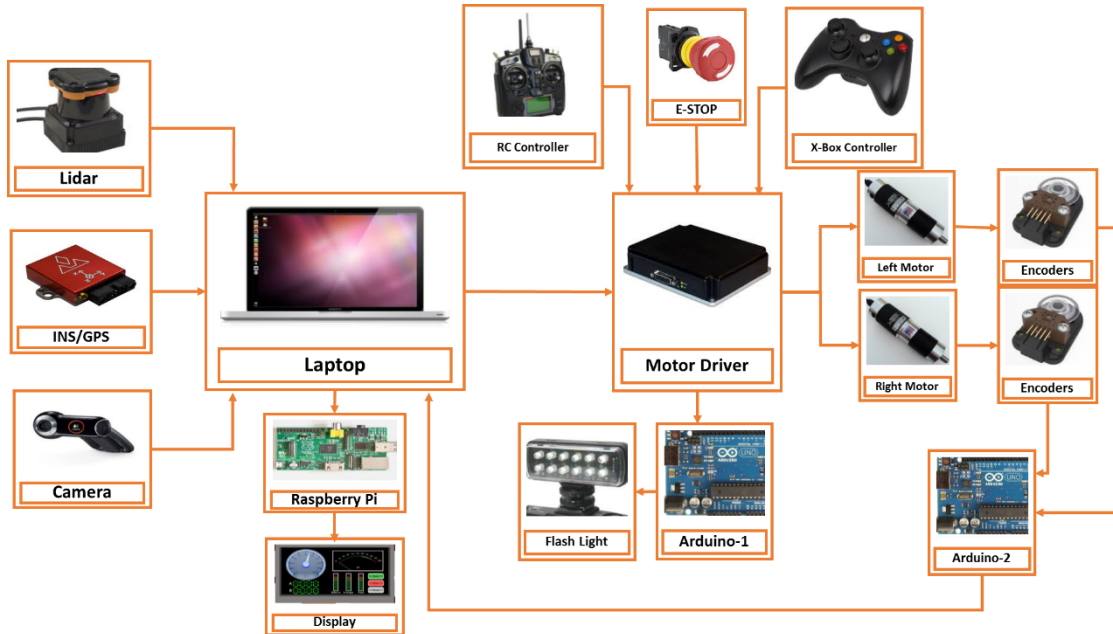


Figure 9. Embedded System Architecture

Control Panel

The control panel of Eklavya 3.0 is both informative and interactive. It features several modules:

- Switch Board: An array of switches to power individual sensors/modules, to electrically decouple the motors to prevent unwanted back-current which might damage the main circuitry, to charge the batteries or to use them in operate mode.
- Safety Lights: A panel consists of 5X10 LEDs to indicate the autonomous or manual drive mode as the rules of IGVC mandate.
- Mechanical E-Stop: Located at the center of the dashboard at approximately a reachable height of 1.2m to bring the vehicle to an immediate halt.

Sensors

Eklavya 3.0 is equipped with high-end robust sensors that have been phenomenal in accuracy, high-speed and reliable automation.

1) Hokuyo Lidar

The vehicle uses a long range laser scanner from Hokuyo with a view angle of 240° and gives 652 readings which are spread equally throughout the range at rate of 40 scans per second. The range of laser scanner is 30m which is sufficient for our planning algorithm. The large scan rate enables us to implement Hector slam to obtain accurate pose, just from the lidar data.

2) VectorNav GPS/INS

VN-200 is a miniature high-performance GPS-Aided Inertial Navigation System that combines MEMS inertial sensors, a high-sensitivity GPS receiver and advanced Kalman filtering algorithms to provide optimal estimates of position, velocity and orientation for industrial applications. We used VectorNav's C/C++ software library to fetch data through the INS solution provided by the sensor which gives us thermally and magnetically corrected pose with a horizontal dynamic accuracy of 2.5 meters. The yaw data that we need for target location was obtained from VN-200 having static accuracy of $\pm 2^\circ$ and a dynamic accuracy of $\pm 0.75^\circ$.

Extended Kalman Filter (EKF) was applied on the odometry data, IMU orientation and latitude longitude from the GPS to accurately localize the vehicle.

3) Logitech Camera

The Logitech QuickCam Pro 9000 color camera is installed which gives high definition video (1600×1200) at 30 frames per second. The camera has inbuilt auto focus and blur removal which maintains the video quality when the vehicle is in motion. It has a high horizontal field of view (600) which makes it possible to detect both the lanes simultaneously.

Power

1) Power Source

Eklavya 3.0 gets its whole power supply from a 2x26AH Lead Acetate battery supported by a 1x8Ah redundant Lithium Polymer battery. The LiPo battery is used only when the motors demand incremental torque which happens primarily during inclines or while maneuvering thick grassy terrain. The power is primarily consumed by three components: motor along with motor controller, sensors and the cooling fans. As one of the many safety precautions, fuses have been installed to prevent component damage.

2) Battery Switching

The Battery Management System (BMS) has been a prime innovation in Eklavya 3.0. The battery level is constantly monitored and a relay circuit switches to the redundant LiPo battery source in two cases: either when one of the lead acid batteries is drained out or when the other motors require more torque. This increasing demand in torque is assumed to arise mainly due to inclines. The incline is detected by a change in pitch sensed by a VectorNav Inertial Measurement Unit installed on the robot and also by the instantaneous battery life remaining.

3) Power Distribution

The power distribution diagram is shown in Figure 9. The battery charger has been incorporated in the vehicle itself with a power outlet for charging the batteries. The power flow from the battery goes directly to the motors and the custom made power distribution circuit board to the sensors, motors, flashlight, mini control board (Raspberry Pi) and the main computing platform (laptop). The power circuit board IC is an innovation this year which features a Buck/Boost converters to cater to the demands of the 19V laptop battery charger, the 12V sensors (e.g., lidar). 5V powered USB hub, Raspberry Pi controllers and the Vehicle Information Display.

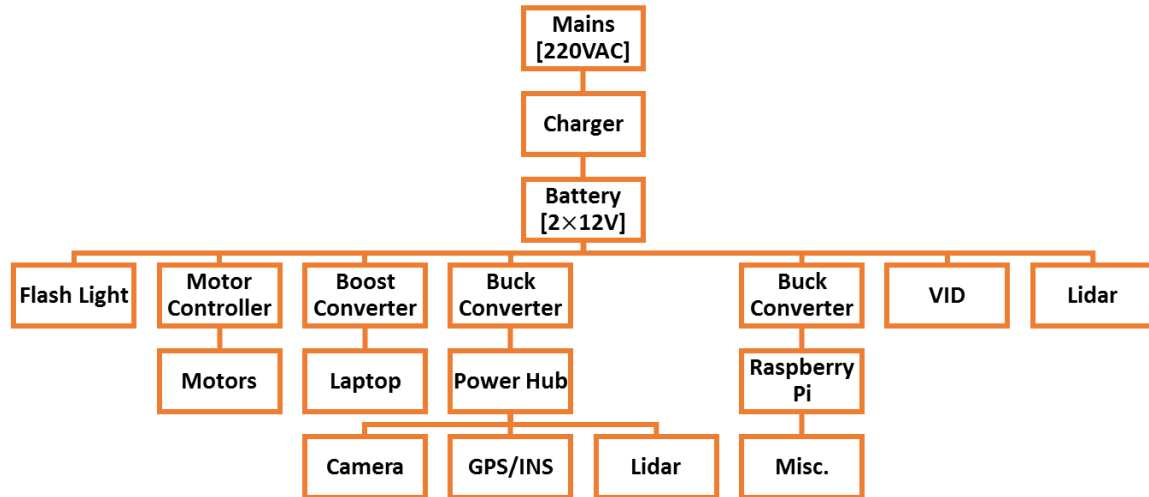


Figure 10. Power Distribution Diagram

4) Power Distribution Table

The power consumed by the different elements are listed in Table 2. Based on these power calculations, the average runtime of the robot comes out to be: 1.5 hours (approx.) at full load.

Table 2. Power Distribution Table

Sl. No.	Components	Voltage (V)	Current (A)	Power (W)
1.	Roboteq (+Motor)	12	15*2	360
2.	Hokuyo LIDAR	5	0.7	3.5
3.	Camera	5	0.25	1.25
4.	GPS/INS	5	0.5	2.5
5.	IMU	5	0.1	0.5
6.	Kinect	12	1.08	12.96
7.	Cooling Fans	12	0.42*4	20.16
8.	Flash Lights	12	1.4	16.8
Total Power Requirement				417.67 W

Motors and Controller

1) Roboteq Controller

Eklavya 3.0 uses the brushed DC motor controller MDC2230 from Roboteq. Roboteq controllers are used in automated guided vehicles and have built-in features supporting some common utilities such as encoder data acquisition, current sensing, PID, RC remote integration, E-Stop etc. The support for the RC remote enables ease of switching to manual control. The priority of the manual control through wireless remote is set greater than serial data transmission, enabling the bot to swiftly switch to manual mode, once the RC controller is activated. It is a 2*60A, 30V micro-controller which receives input from RC or PC and converts it into high voltage and high current output to drive two DC motors. There are various safety measures incorporated in Roboteq which ensures safe and reliable operation.

2) Motors and Encoders

Eklavya 3.0 uses reversible DC geared motors equipped with an IP-65 optical quadrature encoders. The maximum output speed is 4615 RPM delivered at an internal gear head ratio of 34.97:1. The rated continuous torque provided is 164 in-lbs. The motors run at 12V DC and have a rated continuous current of 28.6 A. As per the requirements of IGVC, we needed our robot to move at a speed of 1.5 mph. With a wheel diameter of 8 inch, the required angular velocity of wheels came out to be 71 RPM. The continuous torque requirement, hence, came out to be 110 in-lbs.

Speed Control Algorithm

The control algorithm implemented is the proportional integrative and derivative control (PID) on the speed of the motors. In the Robotiq controller, although a high frequency inbuilt PID controller attempts to achieve the desired set speed, it was required to tune the constants as per our requirements to get the optimum response. The tuning was done using intuitive observations by following the standard Ziegler-Nichols Rule

SOFTWARE SYSTEM DESIGN

Software Architecture

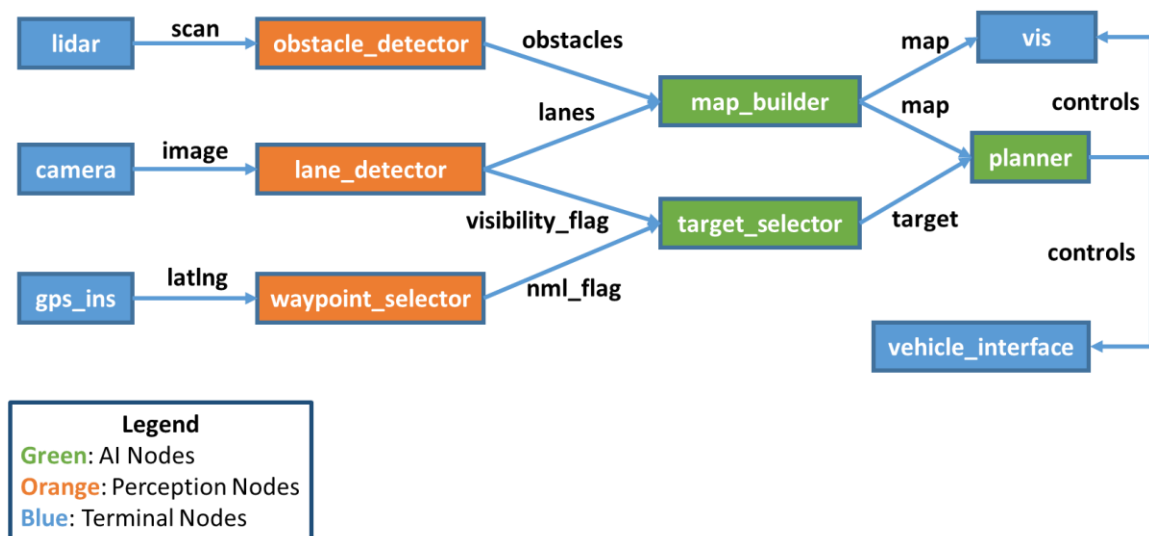


Figure 11. Software Architecture

The software architecture for Eklavya 2.0 was based on threads. Each module was running on its own thread and inter-thread communication was implemented by accessing shared global variables via mutexes. This led to long waiting times when the number of modules increased and so did the communication between them. This prompted the inception of a publisher-subscriber based mechanism, closely following the ROS framework while developing the architecture for Eklavya 3.0.

The ROS framework allows for seamless scalability when changing the number of sensors and other hardware thanks to the hardware agnostic code. Each node runs as an individual process, thus

providing more freedom over its creation and deletion during test runs. Many other features like the rosbag and rosparam helped dynamic reconfiguration of code during run time which helped ease the testing process.

Each node can subscribe to topics on which messages are published by other nodes. The architecture is described in Figure 11 and contains terminal nodes, perception nodes and AI nodes. The terminal nodes consist of entry and exit points to the software stack. Perception nodes are tasked with interpreting the sensed data and modeling the environment. AI nodes mainly include planner and its associated nodes which take the ultimate decisions and generate commands. Messages published by each node are labelled against the corresponding outward arrows. The corresponding topic names are prepended by the node names. (e.g., `lane_detector` node publishes the `lane_detector/lanes` and `lane_detector/visibility_flag` topics)

Perception

1) Lane Detection

The lane detection algorithm for Eklavya 2.0 comprised of color-based filtering followed by application of canny threshold and concluding with a Hough transform to give the line equations of lanes. The observed problem with this approach was that the thresholds were heavily dependent on the changes in brightness and contrast of the ambient environment. Eklavya 3.0 features a machine learning based algorithm adaptive to the fluctuations characteristic of the ambient scenery.

The initial approach was to detect lanes via their structural features. However, applying median blur, histogram equalization and other color and/or brightness equalization techniques, followed by edge detection the results were highly dependent on features like color, blur size, filter type, etc. which were not robust to changes in the lightening conditions. The problem was amplified with non-uniform grass and grass containing dead patches. This led to the conception of the negative space approach. The trick is to remove the grassy portions of the image, instead of detecting the lanes themselves via a machine learning algorithm.

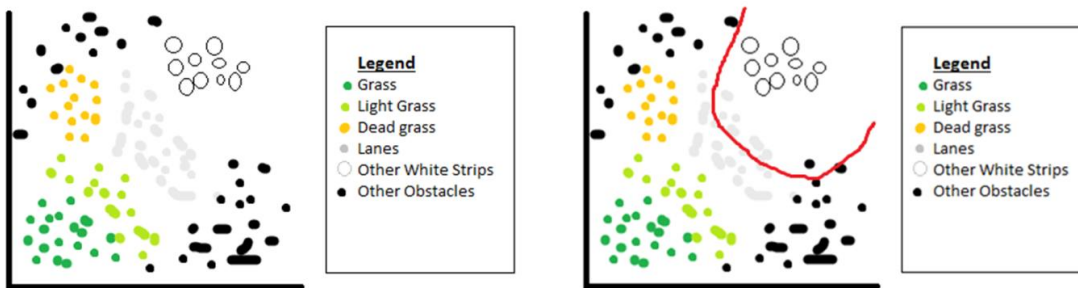


Figure 12. SVM Classification Results for Lane Detection

Features for learning were taken as a kernel of an $N \times N$ ROI of the image. This kernel was classified as grass or non-grass based on a polynomial SVM classifier. This classifier test was used at each pixel of the image to remove the grassy portions. Initially the classifier was trained to detect lanes though, the results were not satisfactory. This was attributed to high saturation and content of the white color in the white strips on the barrels and ladders. The strips caused spurious lanes which led the classifier boundary to shift more toward this region.

As shown in Figure 12, the left image shows the various regions and the right image depicts a red line separating the aforementioned regions using the SVM classifier. Since lanes are not exactly white, some of the lane points lie in the negative region i.e. on the left side of the curve, thus giving false positives. But if the detection is applied on the grassy part we would get a parting line similar to the red curve shown in Figure 14 (Experimental observation. Curves are generated based on classification results over various images). Although this gives a few false positives, most of the lanes are classified as non-grass. Also, grass offered a more uniform patch compared to lanes as the lane portions in the image varied with variations in brightness and lightning conditions. Lanes also exhibit non-uniform thickness.

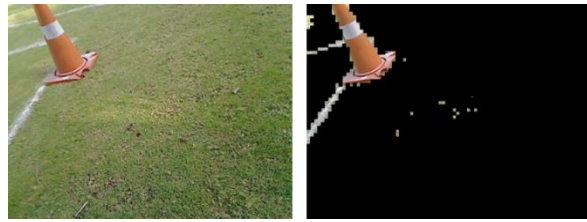


Figure 13. Grass Removal Results

The next task was determining the number N , size of the kernel. It was observed that the code's FPS increased with N and the classification accuracy decreased with N . By experimentation the value of N was kept between 7 and 12 when tested on a 640×480 image with a first generation i5 processor.

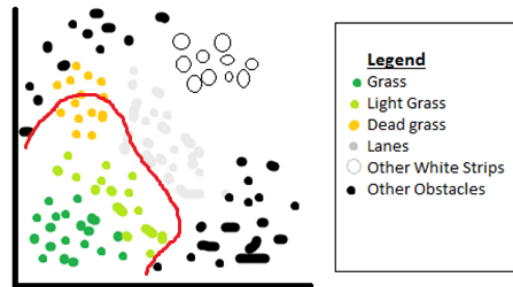


Figure 14. SVM Classification Results for Grass Removal

2) Obstacle Detection and Fusion

The white strips in the obstacles and the white ladders interfere with the lane detection algorithm as they occur as false positives and thus have to be removed before lane detection. Initially a color based algorithm was applied to extract the non-white portions of the barrels and dilate them so that they cover the white strips lying in between them. Still, the problem prevailed in certain cases, especially with the white ladders. Thus, instead of a color based filter, the lidar data was used to erase parts of the image which coincided with the lidar readings.

After this step, the image contained only the lanes and some random noise. The lane was further filtered by a color based threshold algorithm followed by an edge detection algorithm which resulted in high lane detection probabilities with very few false positives.

3) Map Fusion

Output of the lane detection algorithm is subjected to an Inverse Perspective Transform which, with proper selection of the warp matrix, would generate a representation of the lanes in the world frame with a desired resolution. Lidar data is already in a similar world frame, albeit

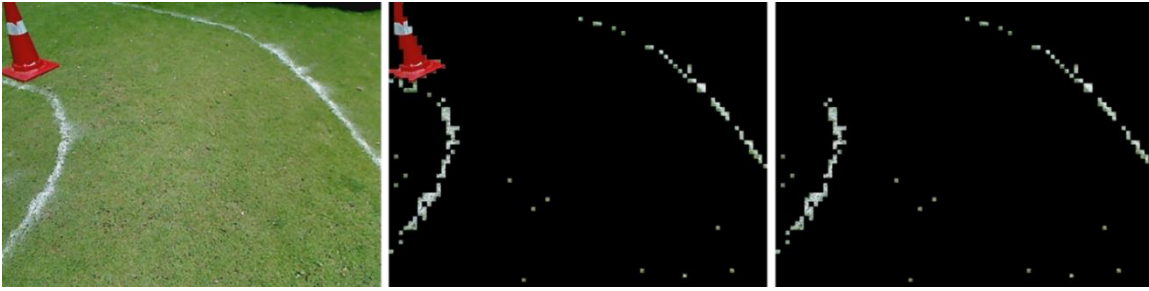


Figure 15. Lane Detection via Grass Removal Results

with some offset (x, y, θ) compared to the lane map. Both these maps are corrected for the relative offset and their union is taken as the final fused map. Note that the lanes are being concerned as non-walkable for all future purposes.

Navigation

Navigation is tasked with the responsibility of using the data interpreted by the perception nodes and steer the bot to success. First an intelligent strategy is used to decide the target to follow out of targets suggested by lane and GPS. Finally, a path planner plans a path from the bot's origin to the target position (relative to the bot's origin) while avoiding obstacles.

1) Target Selection

Targets calculated from the lane and GPS data are both represented in the bot coordinate frame with the bot at origin. The selection strategy is to implement lane following (use target from lane) until the first GPS waypoint is reached or the last waypoint has just been traversed (black path in Figure 16). Otherwise, target from the GPS is used (waypoint navigation, green path in Figure 16). During lane following, it can so happen that the lanes are not visible (especially during switchbacks). In this case, the selection strategy is to resort to the next waypoint navigation.

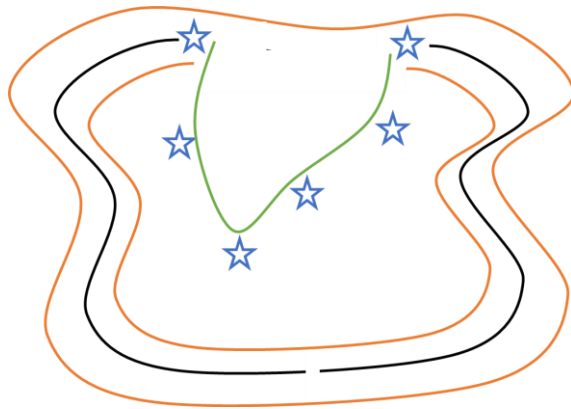


Figure 16. Target Switching Strategy

The binary output of the lane detection algorithm is processed via the Hough transform to give line segments on each lane. Lines are fit which pass through most of these segments and their parameters estimated. Center line of the lane can be obtained by taking the mean midpoint and mean slope of the left and right lines. Lane target is selected as the point lying on the center-line a fixed distance from the perpendicular dropped from the origin to the center-line.

VectorNav GPS/INS gives the latitude and longitude reading of both the target waypoint and the current position of the bot. These are first transformed to the ENU coordinate frame (East-North-Up) and their relative displacement is calculated in meters. A simple yaw correction (obtained from the IMU's yaw reading) yields the position of target in the bot coordinate frame in meters and radians. This is depicted in Figure 17.

2) Path Planning

The path planning algorithm for Eklavya 3.0 has been improved over the last year's A* algorithm to guarantee faster reaction rates, improved control granularity and increased robustness. The planner executes in two stages with the second stage being triggered only when the first stage fails to find a path.

a) Quick Response Planner

The constraint for this stage is to produce a feasible and walkable path with high control granularity quickly. The idea is to spawn a lot of elementary circular arcs, each representing a particular command set (left and right velocities and unit time of execution). After filtering for a basic walkability check, the best arc is selected based on a cost which decreases with the proximity to target and increases with proximity to the nearest obstacle. It may be noted that this planner is not complete; it might not be able to find a path when one truly exists, due to the absence of an exhaustive search function. In such cases, the second stage kicks in.

b) Lattice A* Planner with DT

This is a simple grid A* search algorithm with a slight modification to the grid representation to ensure kinematic feasibility of the planned paths. The inter-grid transitions (seeds) in the grid A* are replaced by circular arcs which always connect two grids. Pose and curvature continuity is preserved at places where the seeds meet, thus ensuring the feasibility throughout a path planned by this method.

In a lattice, each grid center represents the space around it, thus leading to a discrete representation of the environment around the bot. A major advantage of using a lattice is a massive reduction in the size of open list with decrease in the lattice grid resolution. This is possible since two possible paths ending up in the same lattice grid will be mapped to the same lattice grid cell, thus cutting short the search in open list at every step.

The concept of distance transform was adopted from Eklavya 2.0 to produce obstacle avoidance behavior in the planned paths. A safe path would take the bot away from the obstacles as much as possible – probably along the voronoi edges created by assuming the obstacles as the voronoi centers. This can be achieved by adding an extra cost to the cost function which represents the overall distance of the path planned so far to the nearby voronoi edges.

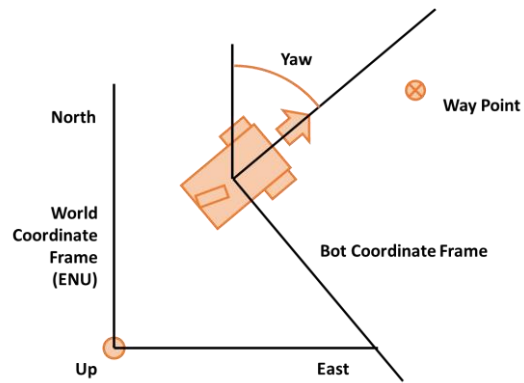


Figure 17. Target from GPS

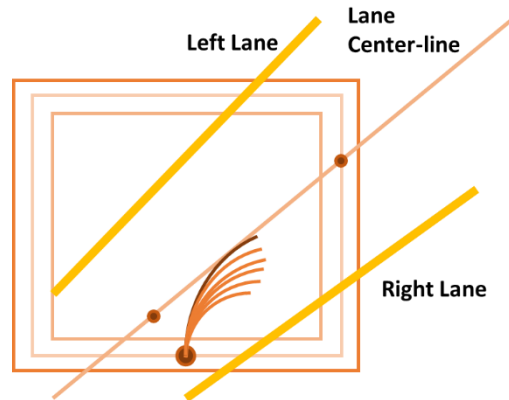


Figure 18. Quick Response Planner

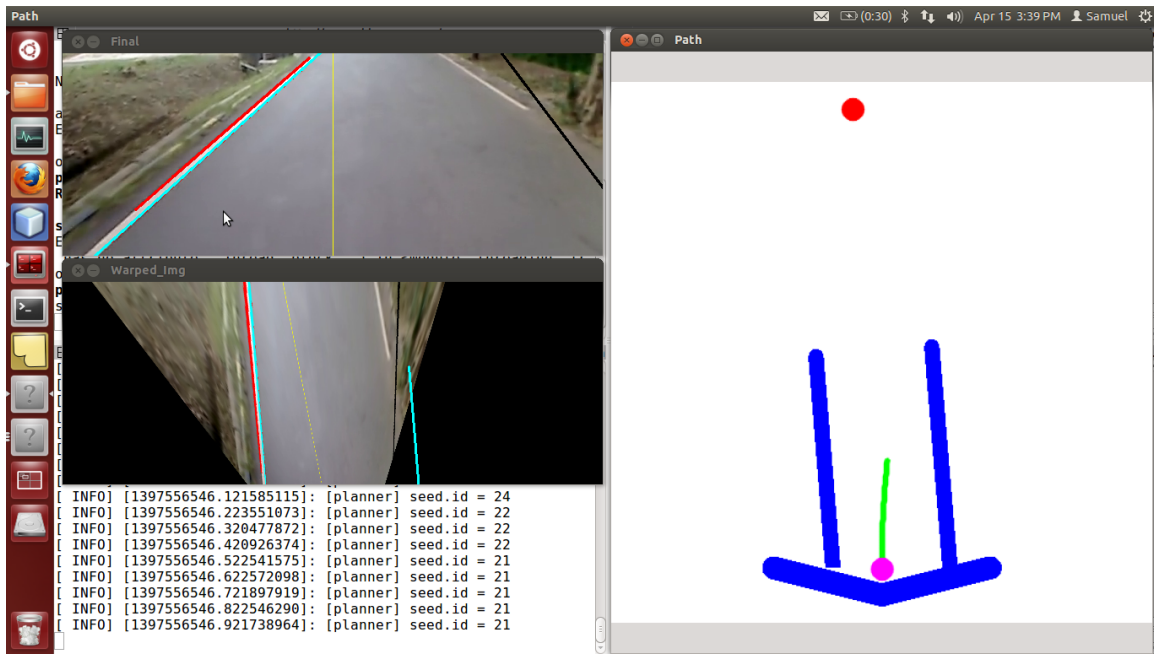


Figure 19. Quick Response Planner with Circular Seeds

SAFETY

“Safety has to be the topmost concern while building any autonomous vehicle” - this was kept in mind while designing Eklavya 3.0. The vehicle is equipped with flashlight indicators that blink to indicate the autonomous run mode. The average speed of the vehicle has been kept at a meagre 2 mph which is not too fast. The mechanical and wireless E-Stops directly cut off the motor driver power bringing the vehicle to an immediate stop. The wireless E-Stops are two X-Bee modules that operate in a secure channel frequency protected by a custom key and operating at a radius of about 100 meters.

PREDICTED PERFORMANCE

Table 3. Predicted Performance

Parameter	Expected Values
Maximum Speed	4 mph
Ramp Climbing Ability	10°
Response Time	0.1 seconds
Battery Life	1.5 hours
Obstacle Detection Range	30 m
Localization Accuracy	0.5 m

CONCLUSION

Team Eklavya designed Eklavya 3.0 with an aim to excel the challenges put forward by the IGVC-2014 based on the important lessons learned in IGVC 2013 with Eklavya 2.0. The team's efforts and innovations are clearly reflected in the improvements on Eklavya 2.0 which represent the humble skill of the people at Autonomous Ground Vehicle Research Group, IIT Kharagpur. Having achieved the goals set at the beginning of the year, the team is looking forward to contributing the ongoing research in the field of mobile robotics and building a driverless car.

ACKNOWLEDGEMENTS

*Team AGV is thankful to Prof. Debashish Chakravarty, Department of Mining Engineering, IIT Kharagpur, for his constant support and valuable guidance. We would also like to express our thankfulness to our sponsors – **VectorNav** and **NAG Inc.** for their gracious contributions.*

TEAM AGV

Software Team

Abinash Meher, Aditya Narayan, Arna Ghosh, Ayushi Mrigen, Harsh Gupta, Krishna Bagadia, Krishna Kumar Agarwal, Rohan Sur, Samuel Anudeep, **Satya Prakash Yadav (Lead)**, Shivam Vats, Shiwangi Shah, Sidakpreet Singh Chawla, Tanmay Patil, Udai Bhardwaj, Yash Shrivastava

Electronics Team

Anshuman Pradhan (Team Lead), Arun Kumar Patro, Ayush Pandey, Bismaya Sahoo, Divesh Hura, Diwakar Paliwal, **Harit Bansal (Lead)**, Kawaljeet Kumar, Madhuri Sangaraju, Manuj Agrawal, Rahul Kumar Singh, Swapnil Shaktawat, Vikram Mohanty

Mechanical Team

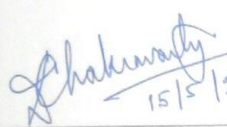
Abhijeet Kant Sinha, Amit Rathi, Ayush Chhaparia, **Naveen Thontepu (Lead)**, Parimal Parth Sarathi, Shubham Kumar, Siddhant Gupta, Siddhesh Keluskar, Suraj Aggarwal

Public Relations Team

Anand Kumar, Dwipayam Kar, Ishan Gupta, **Megha Sinha (Lead)**, Pranshu Jain, Prashant Singh, Rohan Gupta, Shubhi Jain, Soumyadeep Ghosh, Vinayak Mahbubani

FACULTY ADVISOR STATEMENT

This is to certify that the engineering design present in this vehicle is significant and equivalent to the work that would satisfy the requirements of a senior design or graduate project course. Eklavya 3.0 has witnessed significant improvements on the previous bot Eklavya 2.0 in the areas of mechanical design, electrical power distribution, efficient control, system architecture and intelligent navigation. I wish the team all the success for IGVC 2014.


15/5/2014



Associate Professor
Department of Mining Engineering
Indian Institute of Technology
Kharagpur-721302

Professor Debashish Chakravarty,
Dept. of Mining Engineering,
IIT Kharagpur