KRSSG

Kharagpur Robosoccer Students' Group

Project Proposal

Content

Introduction

Kharagpur Robosoccer Students' Group (KRSSG) is a research group sponsored by Sponsored Research and Industrial Consultancy (SRIC). It is dedicated to building autonomous soccer playing robots. We aim to promote research in the field of robotics and make an impression in this field through various international participations. Our past achievements include participation in international competitions like, FIRA Robo World cup 2010 (Bangalore, India), 2013 (Kuala Lumpur, Malaysia), 2014 (China) and 2015(Daejeon, South Korea). We won Bronze in FIRA 2015, South Korea, with being the first ever Indian team to get a podium finish. For the first time we participated in 3D-Simulation league in Robocup, Germany 2016 . This year we aim to participate in RoboCup SSL 2017 to be held in Japan.

Faculty Members

We are guided by the following professors from various verticals -

- Prof. Jayanta Mukhopadhyay
  Department of Computer Science and Engineering
- Prof. Sudeshna Sarkar
  Department of Computer Science and Engineering
- Prof. Alok Kanti Deb
  Department of Electrical Engineering
- Prof. Dilip Kumar Pratihar
  Department of Mechanical Engineering

Team Description

The team is divided into four groups comprising of over 40 students from various departments and over various years all for their similar interests in research in the field of robotics -

1. Artificial Intelligence Team
2. Mechanical Design Team
3. Embedded and Control Systems Team
4. Techno-Management Team

Research Objective

Problem solving in complex domains often involves multiple agents, dynamic environments, and the need for learning from feedback and previous experience. Robot soccer is a similar complex task in which multiple agents need to collaborate in an adversarial environment to achieve specific objectives. Robot soccer offers a challenging research domain to investigate a large spectrum of issues of relevance to the development of complete autonomous agents.

Projects

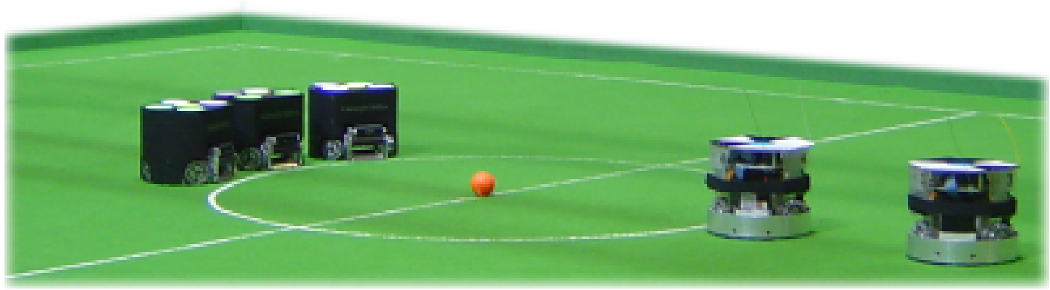
The team further is divided into several profiles which demand it's own area of expertise. This gives the team efficiency to deliver maximum quality output in a limited time. These projects are -

1. Small Sized Robosoccer League
    a. Kicker Team
    b. FPGA Development Team
    c. Controls Team
    d. Artificial Intelligence Team
2. Humanoid League
    a. Design and Simulation Team
    b. Artificial Intelligence Team
3. Code-O-Soccer - A strategical robotics competition at national level
    a. Hardware Team
    b. Organising Team

The rest of the report is organized on the basis of projects that we are proposing to work on and complete in the following years. The team works on all the projects simultaneously to ensure growth and development in the relevant research areas.

1. Small Sized League

a. Overview



A Small Size robot soccer game takes place between two teams of six robots each. The omni-directional robot must fit within an 180mm diameter circle and must be no higher than 15cm. The robots play soccer with an orange golf ball on a green carpeted field that is 6.05m long by 4.05m wide.

All objects on the field are tracked by a standardized vision system that processes the data provided by two cameras that are attached to a camera bar located 4m above the playing surface. The vision system - called SSL-Vision - is an open source project maintained by the league's community.

Off-field computers are used to communicate referee commands and position information to the robots. These computers also perform most of the processing required for coordination and control of the robots. Communications is wireless and uses dedicated commercial FM transmitter/receiver units.

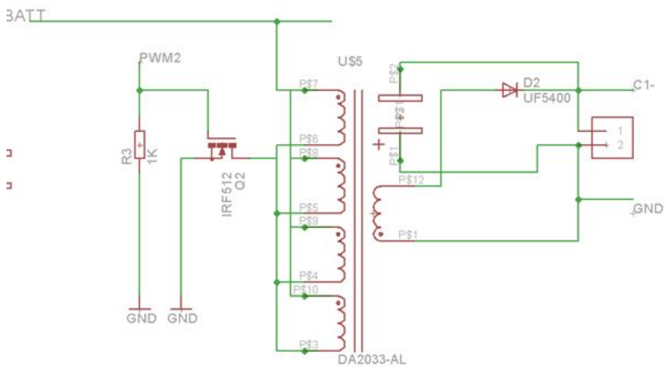
Building a successful team requires clever design, implementation and integration of many hardware and software sub-components into a robustly functioning whole making Small Size robot soccer a very interesting and challenging domain for research and education.

b. Development of Chip and Straight Kicker

Straight and Chip Kicker Circuit Design Based on Flyback converters

The aim is to develop robust charging and discharging circuits to implement straight and chip kicker mechanism. The charging circuits based on Switch Mode Power Supply (SMPS) is shown below.

Charging circuit is based on flyback converters. DC-to-DC techniques use transformers that work at much higher frequencies. Power MOSFETs are used for switching which are controlled by a PWM signal from Atmega328 micro-controller with a frequency of 200 Khz. The PWM duty cycle is optimized for fast charging of capacitors. The capacitors can charge up to 150V in 6-7 seconds. The continuous feedback of charged capacitors is monitored by microcontroller using ADC peripheral. Two capacitors are connected in series with specifications of 2200uF each with a rating of 250V.

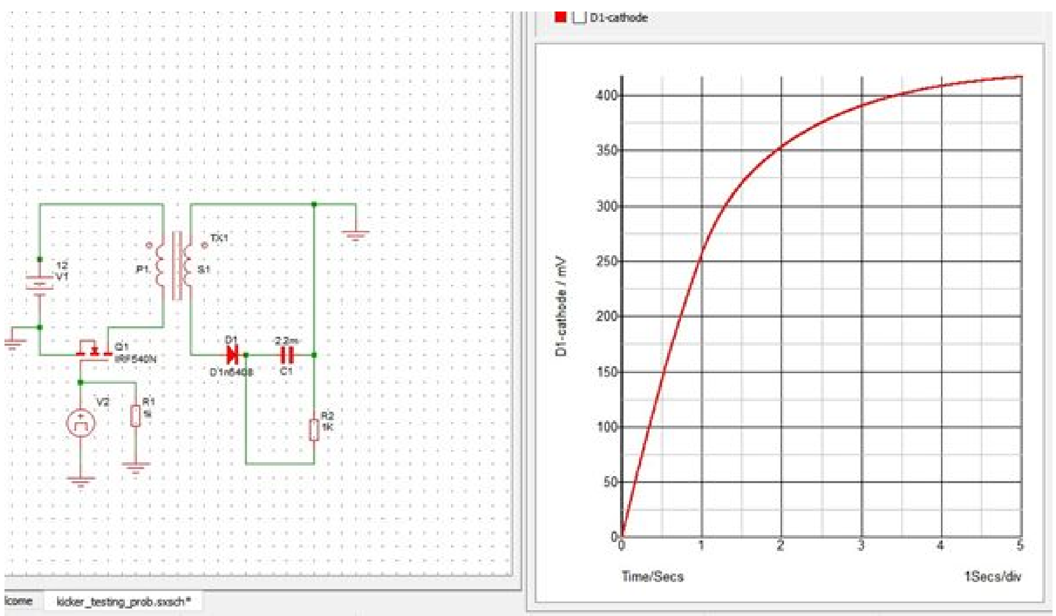


In the **Discharging circuit,** the solenoid is connected in parallel to capacitors in series with relays for switching the discharging circuit. The relays are also controlled with GPIO of microcontroller. Two relays have been used in parallel for handling the high discharging current.

Simulation of Charging Circuit in SIMPLIS

The charging circuit is simulated by the varying all possible parameters i.e no. of turns of transformer, duty cycle of switching and the plots are obtained. The best results obtained are taken into account while fabrication of circuit.

Topologies used for switching MOSFET in charging circuit:



Constant duty cycle :-Circuit takes comparatively large amount of time to reach steady state but less complex in implementation.

Discrete Steps duty cycle:- Circuits converges to steady state for only particular steps of duty cycle and that is quite random. Thus hit and trial is required.

Proportional control: The duty cycle is varied by taking feedback from capacitor voltage using a Proportional control. Theoretically, its best way but charging time is almost same as in case of constant duty cycle.

Elements used for discharging circuit so far are :

IGBT: Integrated Bipolar Junction Transistor is chosen because of its capability of fast switching along with large collector current. But it requires robust protection circuits i.e Snubber. Due to improper snubber design, IGBTS stop working just after 5-6 cycles.

Relays: More robust than IGBT's and simple to control. But large switching time thus variable discharging is not exactly controlled.

Challenges Faced so far:

Till now the first stage of kicker circuit have been fabricated and tested. The difficulties faced by us are as follows:

i. Variable discharging using relays is not been fully controllable due to large switching time of Relays while IGBT's stop working after 5-6 cycles.
ii. The other major challenge is refilling the charge after very time stamp due to continuous charge loss from capacitors to environment when sitting idle. Thus it requires large power consumption.
iii. Also the other difficulty face is the interference of charging circuit with radio frequency modules - transmitter/receiver on the other of the board.

Future Research:

In future we aim to use a more robust controller for controlling switching duty cycle (variable switching) of charging circuit such as LT3750 for which we'll require DSO for obtaining waveforms. Also we aim to incorporate more steps variable discharging through relays using interrupt based mechanism rather polling. Break beam sensors will also be incorporated to ensure presence of ball to the bot , this will act as a double check mechanism and make our kicking system more robust.

c. Development of FPGA based control system

PID and Fuzzy Based Control loop Implementation using FPGA

To control the velocities of the BLDC motors fitted with encoders, a PID and Fuzzy based control loop is formed. The detailed flow diagram of the control loop implementation is given below.

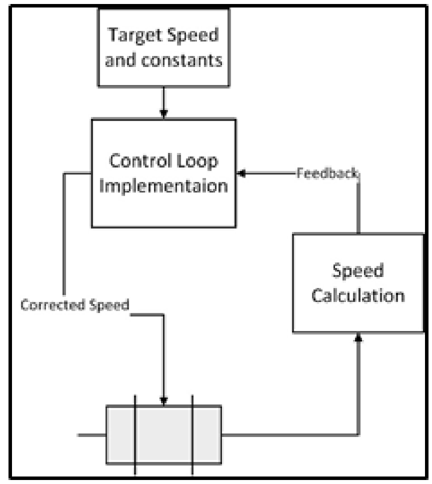
Speed calculation involves the estimation of current speed and direction of the motor using the pulses coming from the high resolution encoder fitted with motor. The control loop implementation block monitors the speed of the motor and calculates corrected speed using target and precomputed constants. This corrected value is fed to the motor using a motor driver. With proper synchronization of all the events a real time monitoring and controlling of the motor speed is achieved.

There are multiple architectures we have used to implement the same, which are mentioned as follows:

Architecture 1: A single microcontroller common speed calculation and implementing control loop

Architecture 2: Multiple inferior microcontrollers for calculating the speed of motors and a central superior microcontroller

Architecture 3: A FPGA controller for calculating the speed and implementing PID using the constants precomputed from a central microcontroller



Challenges Faced with different architectures:

1. In architecture 1 when a single microcontroller is used for both speed calculation and control loop implementation an external interrupt is triggered in the system for each incoming tick from encoder. This load the system heavily and degrades the performance destabilizing the control loop at higher speed. (Implemented in FIRA 2014)

2. In architectures 2, when a separate microcontroller is used for speed calculation, the load on the central microcontroller is divided thereby making it possible for the system to control the motor for higher speed. The slave microcontrollers communicate with the master (central) microcontroller using UART communication protocol. The overhead caused due to this communication comes into action when target speed is above around 75 % of the maximum speed and the control loops become unstable. (Implemented in FIRA 2015)

3. While in architecture 3, when a FPGA is used for the same process, parallel processing is exploited and the system becomes capable of reaching maximum possible speed for the motors (capable of handling the ticks at maximum 6MHz). (To be implemented in SSL 2018)

The performance analysis of the different architectures is given below:

Architecture	Maximum achievable Speed	Maximum Ticks Frequency
Architecture 1	50% of maximum (5000 rpm)	32 KHz
Architecture 2	75% of maximum (7500 rpm)	48 KHz
Architecture 3	100% of maximum (10000 rpm)	6 Mhz

Future Plans:

Architecture 1 and 2 are completely tested and implemented by fabricating custom circuits, while circuits are to be fabricated for the architecture 3. In the third architecture, separate SoC are used for microcontrollers and FPGA. Few SoCs are available in market which house the power of both microcontroller and FPGA with dedicated area. Development of the system using these SoCs will improve performance.

d. Mosfets and Gate Driver based BLDC Control

The circuit comprises of three gate drivers driving three independent discrete P and N MOSFETs inverter circuit. Moreover the basic power circuit consists of fuse (3 Amp) and a main switch along with an LED. Trapezoidal control using hall sensors feedback is used to drive the BLDC motor. The commutation table is generated using a microcontroller feeding the PWM signal to the gate drivers input which in turns switches the MOSFETs depending on the hall sensors values.

Challenges faced

i. Initially we used a gate driver with a peak current of 1.5 A and maximum voltage rating of 18 V. Keeping delay time short and equal rise and fall time, we used Fairchild's Dual N-Mos P-Mos IC with high voltage and current ratings up to 40 V and 6.2 A current. The combination did not worked out well. The Dual N-Mos P-Mos IC burned out even when used with 18 V supply. We used current limiting resistors to prevent the current in the gate terminal of the MOS but the problem was not resolved, we believe that the surge current may be an issue.

ii. FInally, we switched to discrete power mosfets by International Rectifiers. We designed the circuit again using the discrete mosfets and finally we were able to drive the motor using the ATMEGA2560 Microcontroller which was used to generate the commutation using the hall sensor feedback. The gate driver used initially were rated at exactly 18V and they burned out if we used a lower PWM value to drive the motor for longer amount of time.

iii. Ultimately we decided to change the gate driver IC. Since we were using two discrete P and N channel mosfet, we faced the problem of *Deadtime* insertion. The rise and fall time of both the transistors are different resulting in a situation when both the mosfets are in ON state which results in a huge Spike of current from VCC to GND. This time was about 100 ns approximately. We looked for certain deadtime insertion method and finally ended up in using the RC Delay and AND -Gate propagation delay. We observed the possible gate driver inputs and made a logic level interpretation using AND gate and thus used its propagation delay. Then we tried to implement PID control on the motor but the RC Delay is creating some problems and the ticks obtained didn't showed any significant change when the RPM was changed.

iv. Now, we decided to change this method of deadtime Insertion and thought to inject deadtime using the microcontroller. We are successful in implementing closed loop PID Control with ATMEGA2560 but the frequency of ATmega2560 is not enough to inject the DEADTIME. Presently, we are trying to use the STM32f4 microcontroller to implement the closed loop control with proper deadtime.

Future Plans

We have planned to test the circuits on ARM processors with proper PID control and deadtime insertion and then design the circuit on PCB and continue the integration with other parts of the bots. Moreover, we want to use different control methods namely the sinusoidal and the back EMF control methods. We need to implement the sensorless back EMF method of BLDC motor control for the Dribbler Motor as they do not have any encoders. We will implement more robust control loop on the motor using the Fuzzy logics and other control Algorithms.

e. Path planning and optimization

Architecture:-

We use a 3 layered architecture to generate responsive team behavior which is adaptive to dynamic changes in the field during the execution of the soccer match. We have a centralised system, which tries to incorporate the benefits of a de-centralised system with the ease of a centralised system. For that we shifted our code-base from normal C++ architecture to ROS(Robot Operating System) based architecture.

ROS has been used in various other robotic arenas but is a new addition to the scene of robot soccer. Not many teams have tried to use this as a base for their architecture. Most of the teams use a C++ or Java based architecture on the lower level followed by a higher level architecture in Python to incorporate machine learning techniques.

Current Architecture:

Basler ACA2000 165uc, USB-3 cameras have been integrated with SSL-Vision which did not have the support for USB-3 cameras earlier. Using the drivers provided by Basler and some changes in the source code of SSL-Vision provided by Robocup, we have been able to integrate these cameras.

The Nodes :

There are separate packages in ROS, which execute on separate threads to generate a concurrent teamwork.

SSL-Vision publishes the vision information (Position of robots and ball) using Google Protobuf messages which is received by our the node **Vision_Comm**. This is a ROS node responsible for receiving the Vision Message packets and converting them into ROS messages which are published to **vision_data topic** for further processing. This node runs in two modes, one for testing in Simulator Mode which receives the data from GRSIM(a simulator provided by ROBOCUP for testing of the codes), and secondly in vision mode.

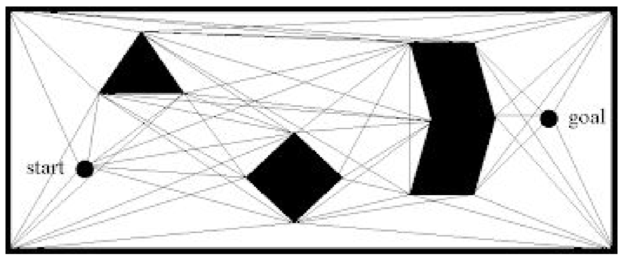
The **belief_state node** is another node which subscribes to the vision_data topic and is responsible for creation of a world model from the raw data. A low pass filter is first applied to raw data to smooth out the inconsistency in the in incoming data. After trying out Kalman Filter and Savitzky Golay, we found out the later one to be faster to converge, which is needed in a highly dynamic situation as is in our case. Then calculations like robot and ball velocities, various field predicates(ball_in_our_possession, free_ball, etc.) are made from the filtered data. This data is published on the **belief_state topic** for other nodes to use.

This belief_state topic is subscribed by the **6 robot nodes** and the **play_node.** The play node is responsible for the higher level strategy formation of the robots. The play node uses the play library to generate roles for each robot based on the state of the game. These roles are published on the the **robot_id topic**. Each robot node subscribes to its corresponding robot_id topic and executes the roles assigned to it using the tactic library which in turn uses the skill library. Each robot node publishes the velocities the calculated velocities (vx, vy and w) to **bot_comm topic**.

The **nodes bot_comm/grSim_comm** subscribe to the bot_comm topic and convert the vx,vy and w values into the velocities of the 4 omnidirectional wheels. These wheel velocity values are finally transmitted from the robot_comm node via the communication modules to the robots.

Path Planning :

Visibility Graph:-



In terms of path generated this planner gives the most optimized path. It uses the corners of all the obstacles which includes HomeTeam Bots, Opponent Bots and all corners in the boundary and goal lines of the field. Using these points A* or Dijkstra's algorithm is used to find the shortest path. After the generation of the path curve fitting is done to generate a curve that uses the points to form a trajectory. It was tested and found that it was very slow and unstable in dynamic environment.

RRT Connect:-

It is used by a majority of teams (including modifications) in Robocup SSL League. The algorithm is probabilistically complete. RRT algorithm grows a tree with its root at the starting position by sampling random points from the search space (i.e. arena). As each sample is drawn, a connection is attempted between it and the nearest state(node) of the tree. If the connection is feasible then a new state is added at a certain fixed distance from

the nearest state. Due to uniform sampling of the search space, the probability of expansion of the tree towards a large unsearched space is higher. The downside is that uniform sampling produces unnecessarily long paths.
Future Scope**:** Better heuristics and implementations(VF-RRT, RRT* etc.) of the above algorithm can be used to produce better paths.

Recursive Planner:-

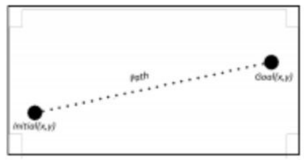
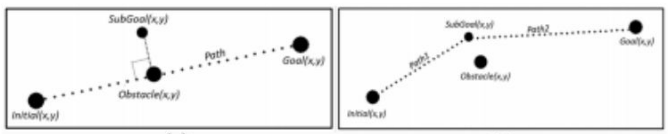


Fig. 3. Path without obstacles



(a) (b)

Fig. 4. (a) Subgoal Selection. (b) New paths created with subgoal

Since there was a requirement of a fast path planner that gives shorter paths than RRT, we tested Recursive Planner. It first tests if there is an obstacle in between the HomePos and destination. If no, then it directly joins the points else it samples a point that lies on the perpendicular bisector of the line joining HomePos and destination at a fixed distance k. Then the algorithm is recursively applied to the 2 subsections generated (i.e. - HomePos, generated point and generated point, destination). Upon testing it was found that algorithm was very fast and gave short paths.
The downside of this planner is that their was a huge probability of replanning and the path changed frequently. Though path generation was quick but the algorithm performed slow due to the problem of replanning at every step.
Future Scope: The algorithm can be modified to ensure that the frequency of replanning is restricted.

Integrating learning and planning: We integrate learning and motion planning for a soccer playing robot using Bayesian optimisation. Trajectories generated using end-slope cubic Bezier splines are first optimised globally through Bayesian optimisation for a set of candidate points with obstacles. Then, trajectories and the corresponding prior are stored in a database to be queried for real time path optimisation. The closest planning situation is identified using k-Nearest Neighbour approach with robot and obstacle position, and velocity as features. It is further optimised online through reuse of prior information, enabling optimised trajectories with obstacle avoidance in high-velocity dynamic soccer game. A velocity profile for the trajectory is generated incorporating kinodynamic constraints of the robots, using arc-length reparametrization through a secondary spline. Trajectories are tracked, which triggers trajectory replanning in case of large displacements. Extensive testing is done on developed simulator, as well as on our robots. Our method shows marked improvements in mitigating tracking error, and reducing traversal and computational time over competing techniques under the constraints of performing tasks in real time. This work in under review at Advances in Robotics 2017.

Future plans:

There are some major changes that we are about to bring in the architecture.

1. We are aiming to shift on a response based system for implementing smooth passing and receiving. One of the major challenges in Robosoccer is intelligent and adaptive passing. The passing is currently implemented in the play layer. But this restricts the roles of the robots and binds the system in a way. We plan to make the passing a different response based function which will be triggered once a passer chooses a robot to receive and assigns the receiving task to it.
2. Detection of chipped ball and interception of a chipped pass by the opponents is an unsolved problem. We plan to detect the chipped ball using the curvature of the path caused by the projection of the chipped ball with respect to the direction towards which the robot is facing. Further to intercept the ball we will be fitting a 2 degree general curve on the collected points till that frame. Other teams have used kalman filter to get the predicted path but that method is slower to converge, the parameter determination is tedious and results have not been very satisfactory.
3. We plan to shift the play layer to python due to the vast array of packages available to us to implement machine learning algorithms for play selection and role assignment. Python gives us the advantage to quickly implement and test various learning algorithms. Other SSL teams have also experimented with the use of Python with success. Since, ROS has the ability to support Python packages besides C++ it would be helpful for us to shift to Python.
4. Besides testing individual path planners we propose to implement an algorithm that would select a path planner from an array of pre-implemented planners according to the positions of Home Bots and Opponent Bots (i.e. the state of the play arena). This would allow us to get well-optimized paths and would reduce average computation time.

f. Mechanical Design

Dribbler

A dribbler bar made of Aluminium is used. Dribbler bar is directly driven by a brushless Maxon EC16 (16W) motor through two external spur gears with ratio of 4:1. After calculations we concluded that the chip kicker plate should be parallel to the common normal of the ball and the dribbler. Angle of chip kicker plate: **50°.**

Analysis of dribbler design

The design has slots or grooves on the dribbler’s surface to prevent slippage of the ball during turning of the bot. The equation for dribbler height was determined using the MATLAB code for different percentage of area coverage. We found out the various heights of the dribbler. The MATLAB code was written and practically verified to optimize the height of the dribbler. We also had to consider the angle of the chip kicker in determining the height as the normal to plate of the kicker should be parallel to the common tangent between ball and the dribbler. Thus the optimum height of the dribbler = **3.97 cm** (from base to center of dribbler)

% area covered = **18 %**

Solenoid

The solenoid mainly consists of 3 parts: the plunger, the shield and the coil. The plunger is the movable part which delivers the force to the leg. The shield makes sure that the magnetic field doesn't influence other systems in the robot, like the motors or the laptop and also makes sure that the other systems don't influence the magnetic field built by the coil. The shield also decreases the reluctance at the outer positions of the solenoid. To be sure most energy will be available inside the coil, the reluctance outside the coil has to be as little as possible. The reluctance of a steel shield is much less than the reluctance of air. So the shield makes the solenoids also more efficient.

Straight Kicker

The straight kicker is powered by two 250V 2200uF capacitor, which can be charged by a step-up converter upto 200V each but we have limited the voltage to 150V to limit the kicking speed to 6 m/s to comply with SSL rules. After electromagnetic analysis using FEMM 4.2 and MATLAB, we found that 23 AWG wire gave the optimum results. So 23 AWG wire is wound with 870 turns.

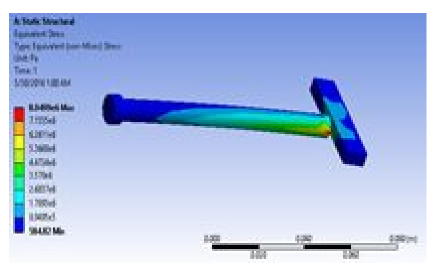
Estimation of maximum Force on straight kicker

For the estimation of the maximum force on the kicker during the collision with the ball, ADAMS simulation was used. We got the maximum deceleration of the front part of the kicker and multiplying it with the mass of kicker we got the required force.

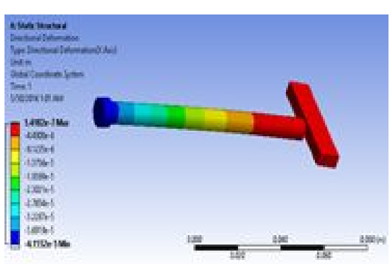
Maximum deceleration = 4.1×10^{3} ms^{-2} (approx)

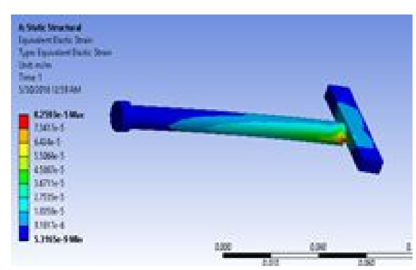
Mass of the kicker = 64.35 gm

Maximum force = 263.835 N



Total deformation





Analysis of chip kicker

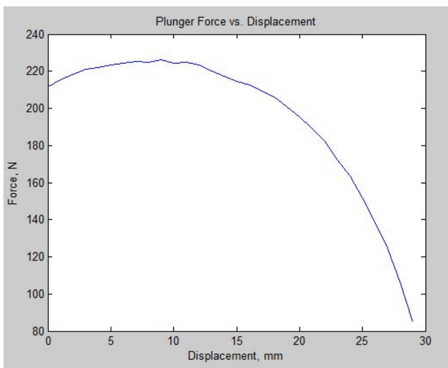
The new chip kicker with pull type mechanism was designed in the previously used flat solenoid was replaced by cylindrical solenoid and pulling action was achieved by the plunger of the straight kicker.

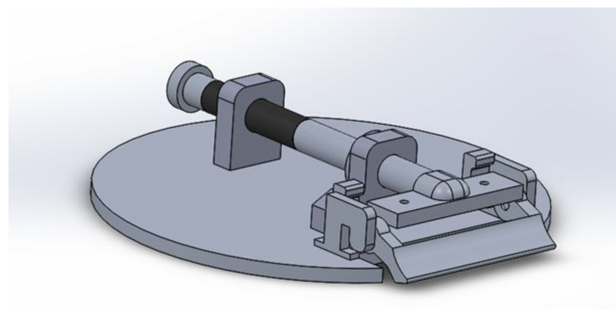
FEMM MODEL OF SOLENOID

Due to increase in space in the new model we increased the width of the solenoid and consequently increased the no. of turns. In the new solenoid the **no. of turns = 1400.** The above no. is for 24 AWG wire. Following is the FEMM model of the solenoid. For a constant current of 40A in the solenoid. The results are as follows

Total energy transferred to the solenoid = 5.726 J

The plot of plunger force vs displacement graph is as follows





After obtaining the graph we got the expression of force with respect to displacement of plunger.

Feeding this expression in ADAMS we obtained the range and max height of the ball. Following are the results of the kicker in ADAMS.

The **MAX HEIGHT OF THE BALL = 5.07m (approx.)**

The **RANGE OF THE BALL = 20m (approx.)**

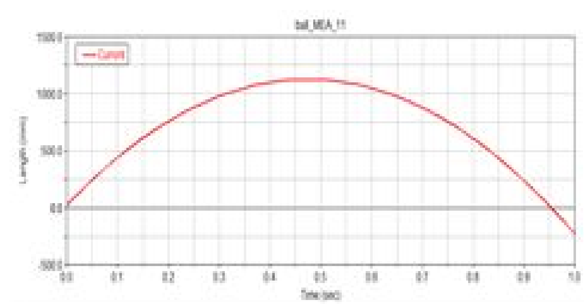
We changed the **24AWG** wire with **22AWG** wire. So the no. of turns reduced from **1400** to **930.** The plot obtained from MATLAB was a little better and then it was practically tested. Energy transferred to the plunger = **3.727J.**

ADAMS SIMULATION RESULTS

Results of current design yields:-

Range of the ball = 6.4 m (approx.)

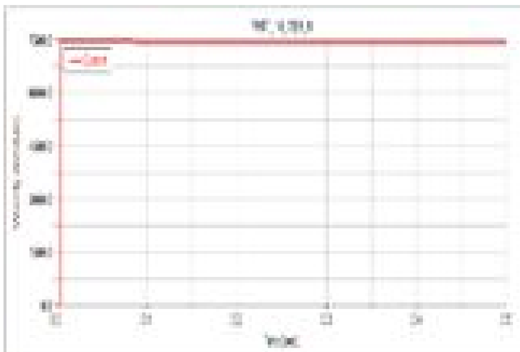
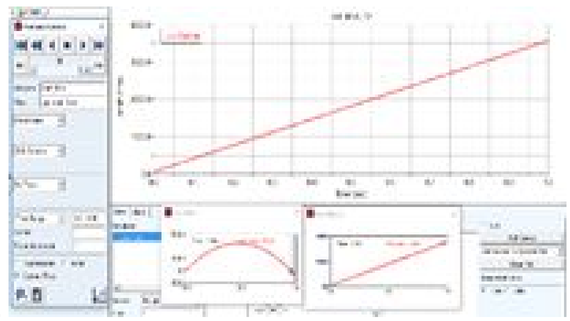
Height of the ball = 2.7 m (approx.)



FEMM AND MATLAB MODEL

The **24 AWG** wire is replaced by **26 AWG** wire.

The no. of turns increased from **1300** to **2200 .**
Energy = 9.146 J
Range = 9.1 m.
Max height = 2.75 m.

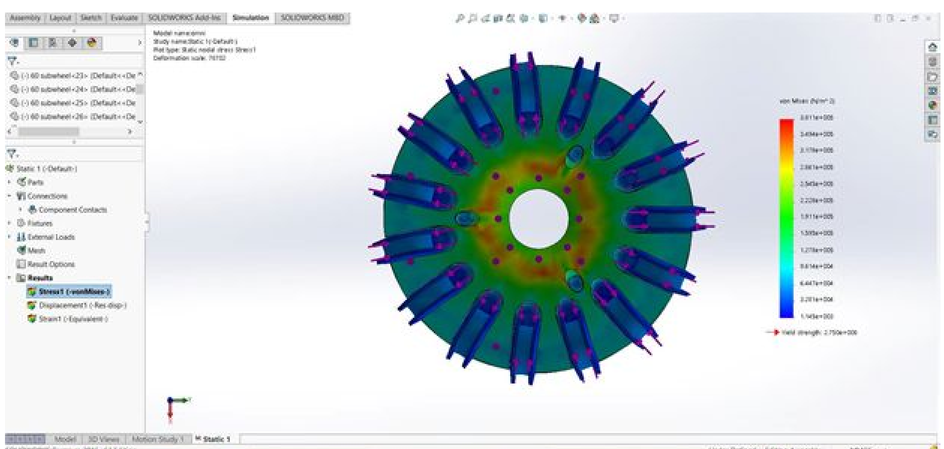



Wheels

The number of sub-wheels in the omni wheel was optimised by analyzing the stress sub-wheels might experience during a collision. Another major change in our older robot is that the screw joining the wheel and the shaft used to loosen after few minutes of continuous motion of wheel. So we redesigned the washer and implemented a new square locker mechanism to prevent the loosening of screw

Stress Analysis

Analysis of Omni Wheel in SolidWorks



Stress Analysis of the body was performed prior to manufacturing and it was ascertained that the stresses were coming under tolerable limits , even when collisions occurred .

g. Timeline

Current	• 6 bots are ready with initial mechanical design and circuits based on ARM controllers using an industrially built controller. • It is being used for testing various path planners and controls of the bot. • The code is in complete running condition. SSL vision configured to work with camera used for FIRA. MergeS curve is used as the path planner for the SSL team.
Year 1	• We aim to shift from industrially built motor controllers to an in-house built BLDC controller using Gate Drivers. • We will also optimize kicker charging time and incorporate variable kicking. This includes the completion of mechanical prototyping of chip kicker. • We aim to configure SSL vision to work with the new Bassler camera before the month of May. • Code architecture will be changed to accommodate multiple path planners. An improved ROS library OMPL will be used to bring this change before the month of May. • Reinforcement learning algorithms will be implemented for Play selection and Role assignment using python libraries in the month of May.
Year 2	• By this year, we aim to complete the prototype circuit for FPGA. The microcontroller onboard would be replaced by a softcore processor on Spartan 6 FPGA with only the relevant digital logic required to drive run the integral components. • We aim to get a new set of robots with proper space optimization and chip kicker implemented • Further development of game play
Year 3	• We aim to prepare circuits based on FPGA and extend that research to optimize the computation the spartan board

h. Budget Required

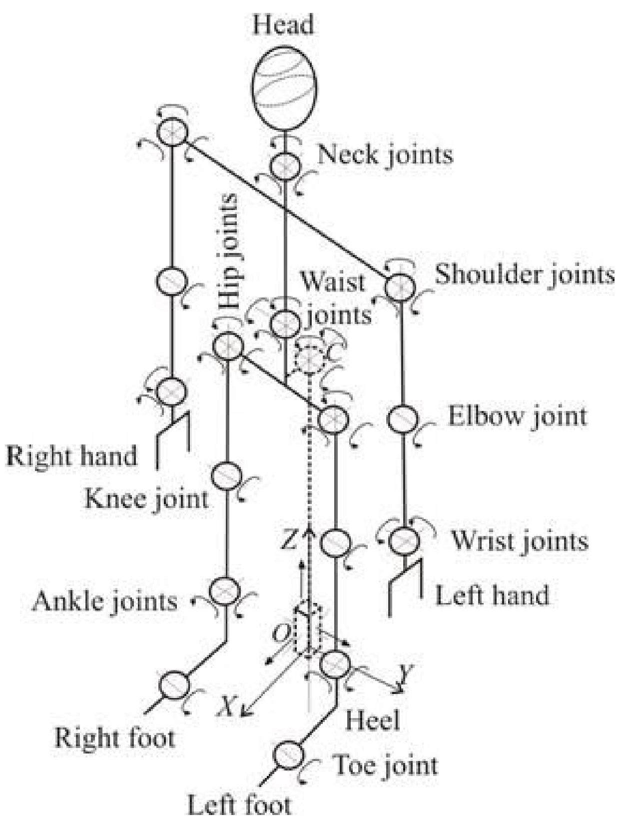
Purpose	**Particulars**	**Quantity**	**Price per unit**	**price**
	Spartan 6 (FPGA Ics)	15	3,456	51,480

FPGA Development	Circuit fabrication	15	2,000	30,000
	Circuit Components	15	2,000	30,000
	Zynq development Board (zybo)	1	23,737	23,737
	Oscilloscope (50 MHz 2 Channel)	1	24,270	24,270
	Total			1,59,847
Kicker Development	2200uF Capacitors	15*2	1,250	37,500
	Circuit fabrication	15	2,000	30,000
	Circuit Components	15	3,000	45,000
	Break Beam sensors	15	1,000	15,000
	LT3750 controller	5	600	3,000
	Total			1,30,500
Driver Development	Gate Driver	150	130	19,500
	Mosfet p channel	150	65	9,750
	Mosfet N channel	150	90	13,500
	STM32F407	50	1,100	55,000
	Circuit Components	15	3,000	45,000
	Circuit Fabrication	15	2,000	30,000
	Total			1,72,750
Mechanical Design	Prototyping body	-	-	1,55,000
	Prototyping (dribbler and kicker)	-	-	70,000
	Final Bot MAnufacture	7	1,40,000	9,80,000
	Total			12,05,000
Overall	Including all above			16,68,097

2. Humanoid League

a. Overview

Increased demand for use of various types of robot have been of prime interest to us. Among the wide range of variety of robots, humanoid robots are the focus of researches nowadays. We aim to build a fully autonomous humanoid robot which can play soccer and has the ability to reproduce human like movements, skills and strategies.



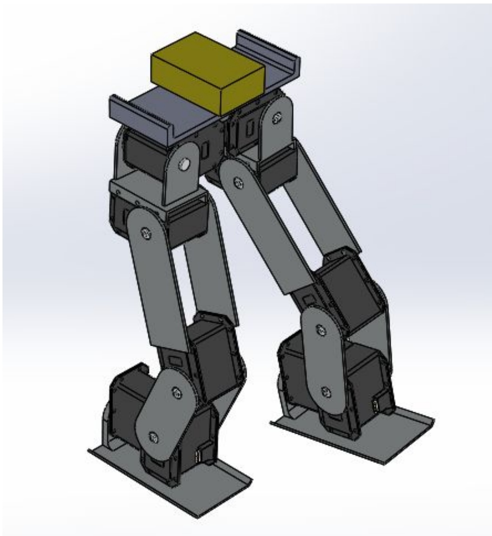
b. Hardware Team

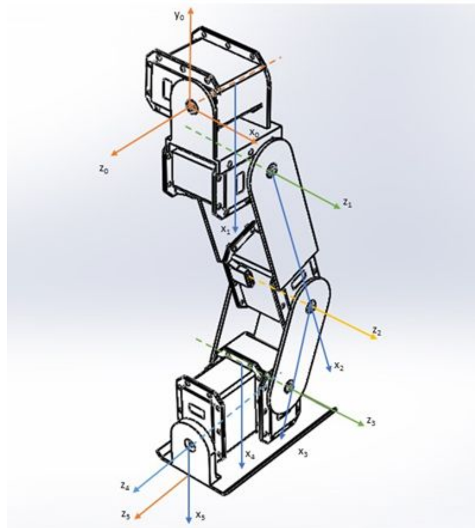
Robot kinematics which basically deals with study of kinematics of open chain mechanisms. It also studies the relationship between the dimensions and connectivity of kinematic chains and the position, velocity and acceleration of each of the links in the robotic system, in order to plan and control movement and to compute actuator forces and torques. The relationship between mass and inertia properties, motion, and the associated forces and torques is studied as part of robot dynamics. To maintain dynamic balance during the walk, a robot needs information about contact force and its current and desired motion. The solution to this problem relies on a major concept, the Zero Moment Point (ZMP) and the Inverted Pendulum Model. Robot Kinematics can be broadly classified into Forward Kinematics and Inverse Kinematics.

The above figure shows the kinematic diagram of a human giving an overview of the degrees of freedoms of various joints and their axis of rotation. We designed a biped robot in solidworks and calculated the D-H parameters. Thus we calculated the transformation matrix for the foot with respect to waist.

Link/ Frame	a_i	α_i	d_i	θ_i
1	4.82	90°	7.20	-90°
2	110.51	0°	0	θ_2
3	50	0°	0	θ_3
4	24.47	-90°	24.44	θ_4

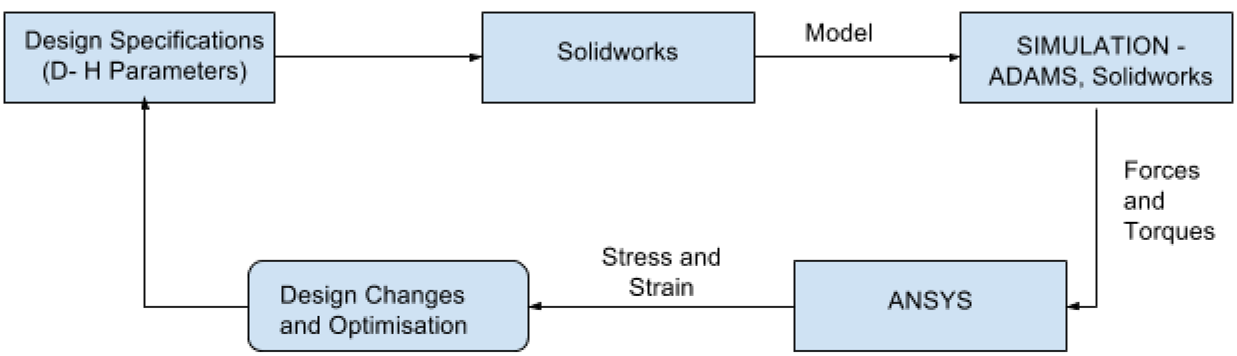
5	23.41	0°	57.94	θ_5





In the next phase, development kick-started the designing and analysis of humanoid robot.

The workflow of designing implemented by us is represented below and one can notice that it is an iterative process, thus we can evolve our design to produce a more optimized and stable design.

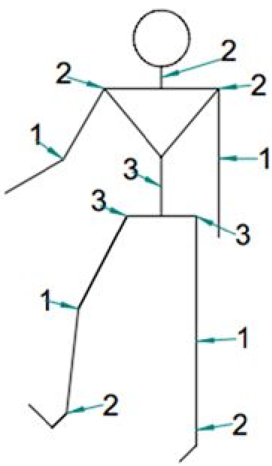


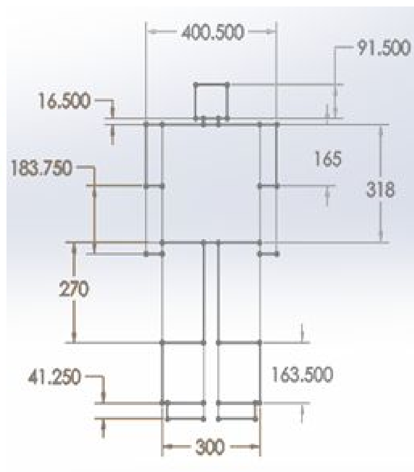
The designing process involves the use of multiple softwares (like SolidWorks, Adams, Ansys, Matlab) to get the desired model. The various steps are explained below-

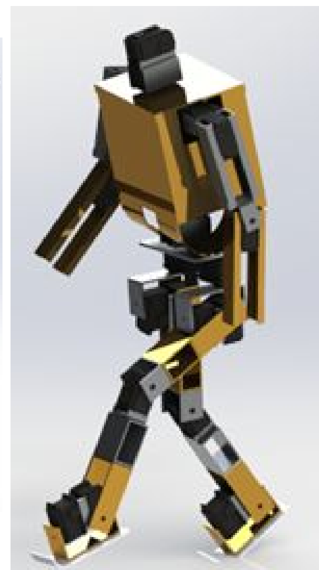
1. **D-H Parameters:** At first the total no of degrees of freedom is decided analogous to human body. The current design specifications are as follows-
    a. Degrees of freedom = 23
    b. Height = 90 cm
    c. Width = 40cm

   d. Weight = 7.5kg (including electronic components)
      Then the various D-H parameters are calculated which further gives us the transformation matrix of each of the individual links.

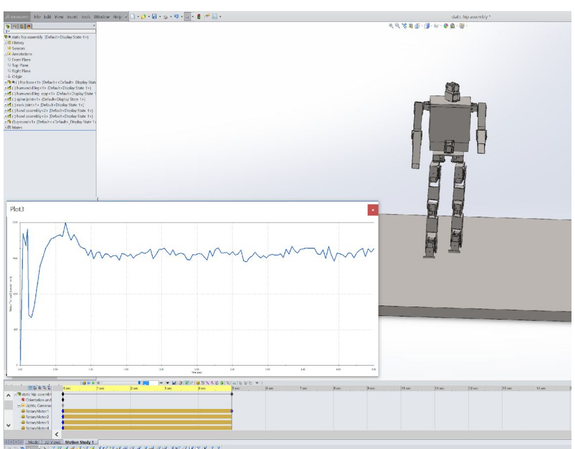
2. **Solidworks Model**: On deciding the basic parameters we designed the model in solidworks. Which would further be used for static and dynamic simulation.
3. **Simulation and Torque Analysis:** The complete analysis was done in solidworks and the optimization of various parameters was done. The optimized results are shown in the pictures. As foot is raised the torque increases but the bot is able to balance itself. The bot is completely stable and it proves the bot is capable of withholding standing stresses.

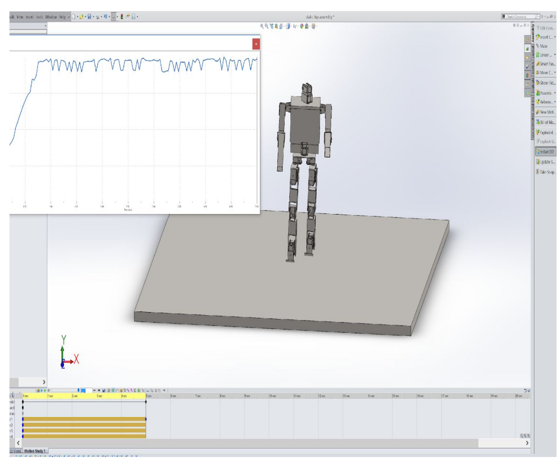






Case 1: Standing Torques

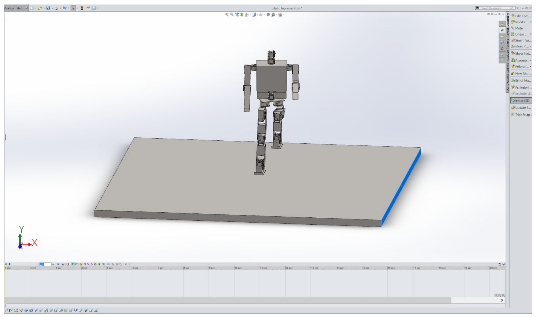


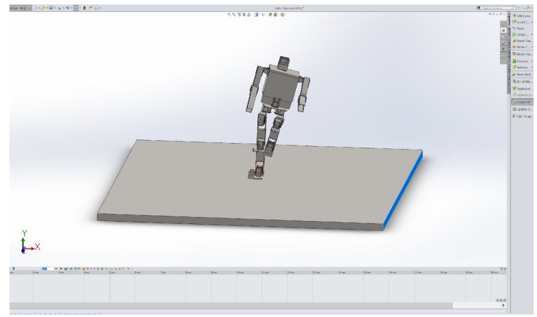


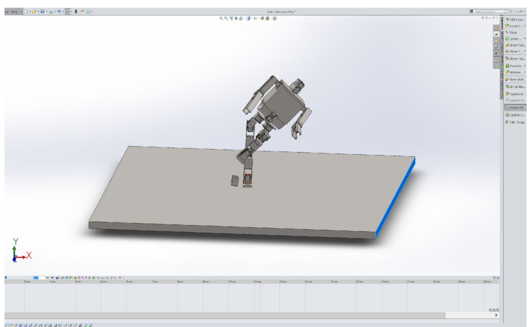
This is the calculated torque for a test Humanoid Robot to keep standing. (for the left leg of the robot). This is the calculated torque for the same position as the first case. (For the right leg).

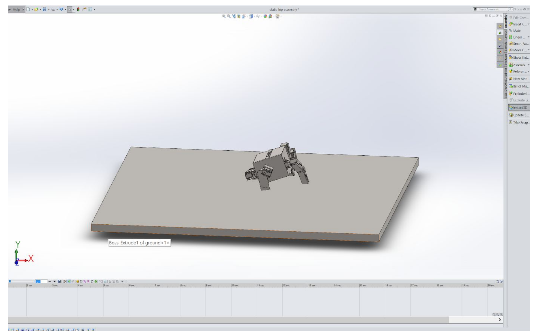
Just in case things are not clear the peak torque is 3.4 N.m and 2.5 N.m respectively. It is not same because we weren't able to make the robot to stand in exact upright manner by freeing all joints due to complications arising for other movements. We will try our best to give values for better positions (more upright).

Case 2: Falling torques









The above pictures represent the falling procedure. The fall was pretty bad . We calculated the torques for the ankle motor which maximum torques in the above fall.

Fig aside

The torque for the initially raised leg was low initially since it was under free fall. It suddenly peaks when it hits the ground whose value was 2.29 N.m.

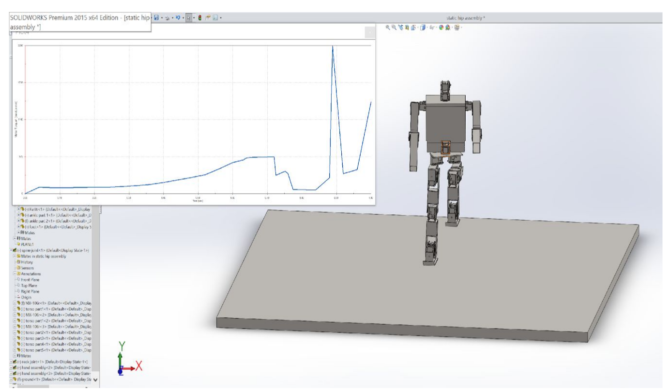
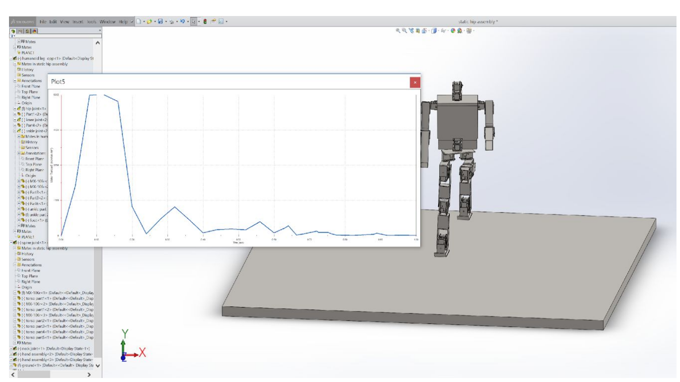


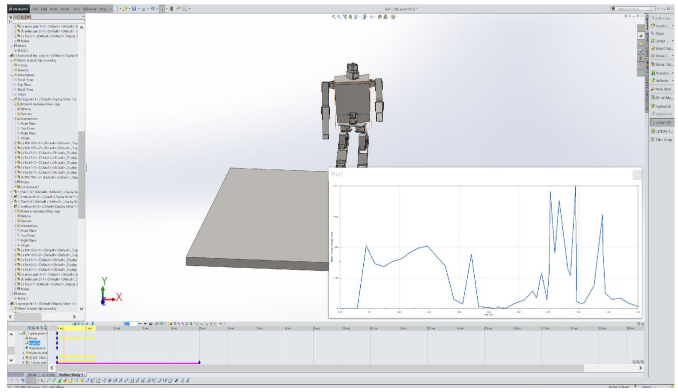
Fig aside

This is the torque of the right leg. It peaks at 5.5 N.m which we can be taken as the total torque on the motor when balanced on 1 foot. It then drops steeply as the bot falls down. For safety the other motor in the foot was checked and the value came out to be a mere 0.3 N.m.



Case 3: Backward Falling

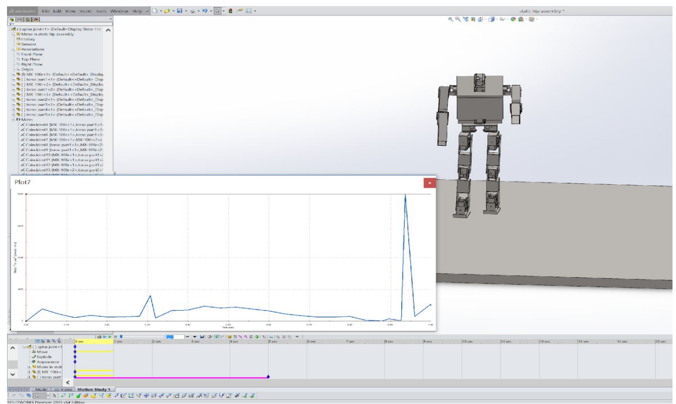
Fig Aside



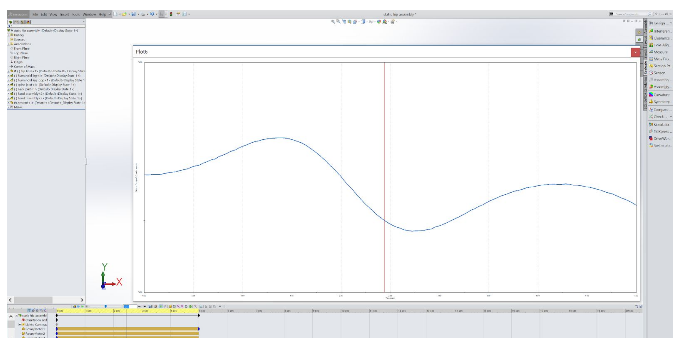
- Maximum torque in knee motor during falling backwards was 4.3 N.m.
- Maximum torque in spinal motor during falling backwards was 2.9 N.m.
- Maximum torque in thigh hip region was about 3.4 N.m.

Case 4: Forward Falling

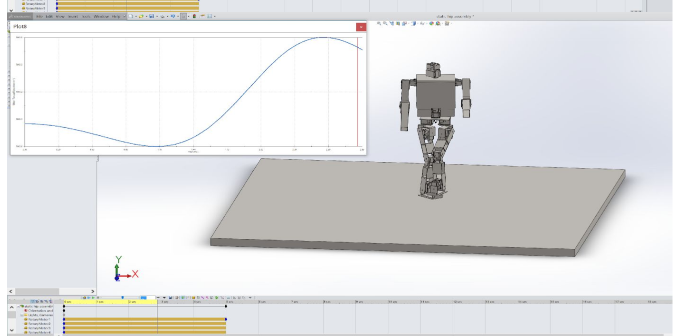
Fig Aside



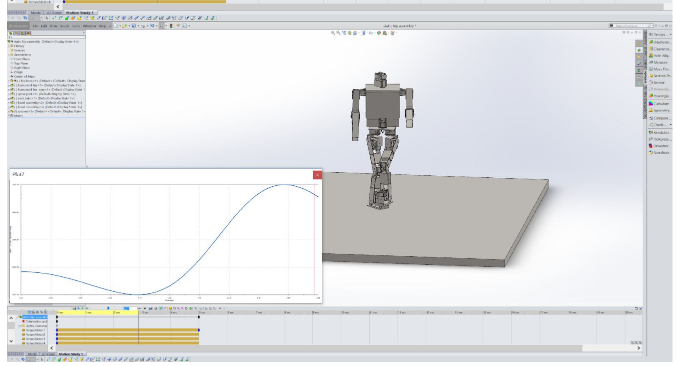
- Maximum torque in hip motor during forward falling was 6.7 N.m.
- Maximum torque in foot motor during forward falling was 5.5 N.m.
- Maximum torque in knee motor = 2.8 N.m



Peak torque of *ankle motor* was 5.096 N.m. The torque remained nearly constant i.e. between 5.095 N.m and 5.096 N.m.



Peak torque of *knee motor* was 3.303594 N.m. The torque remained nearly constant i.e. between 3.303574 to 3.303594 N.m.

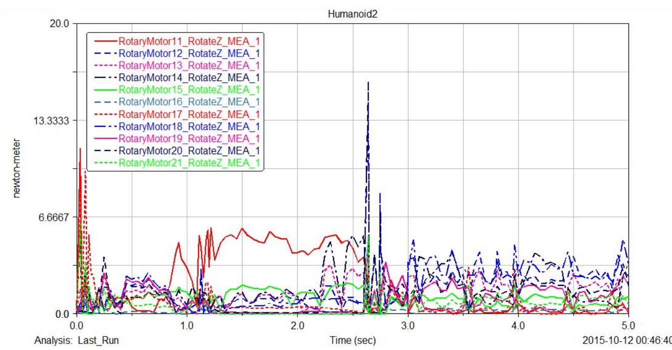


The *hip motor* along pitch gave maximum torque i.e. 7.0635 N.m. The torque remained constant between 7.063-7.0635 N.m

Motor Torques

Based upon the simulation studies we selected the appropriate motors which can withstand the peak torques experienced at the joints.

HIP (YAW)	2.8 Nm
HIP (PITCH)	5.5 Nm
HIP (ROLL)	6.7 Nm
KNEE (PITCH)	5.5 Nm
ANKLE (PITCH)	3.8 Nm
ANKLE (ROLL)	4.1Nm



Dynamic Simulation Torques (Simulation in ADAMS)

Table showing Maximum Torques experienced at various joints.

Motor Selection based upon the calculated Torques

1. Dynamixel MX 106R (12 Nos.)

- Weight : 153g
- Dimension : 40.2mm x 65.1mm x 46mm
- Stall Torque
  - 8.0N.m (at 11.1V, 4.8A),
  - 8.4N.m (at 12V, 5.2A)
  - 10.0N.m (at 14.8V, 6.3A)

2. Dynamixel MX 64R (7 Nos.)

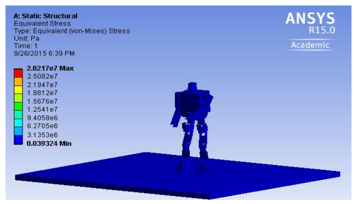
- Weight : 126g
- Dimension : 40.2mm x 61.1mm x 41mm
- Stall Torque
  - 5.5N.m (at 11.1V, 3.9A),
  - 6.0N.m (at 12V, 4.1A)
  - 7.3N.m (at 14.8V, 5.2A)

3. Dynamixel MX 28R (4 Nos.)

- Weight : 72g
- Dimension : 35.6mm x 50.6mm x 35.5mm
- Stall Torque

- 2.3N.m (at 11.1V, 1.3A),
- 2.5N.m (at 12V, 1.4A)
- 3.1N.m (at 14.8V, 1.7A)

4.Static Stress and Strain Analysis on ANSYS: Stress analysis was done in ansys to check the stress bearing capabilities and the structural stability of our model and as can be seen from the pictures the results were very good with a factor of safety of (FOS) = 4.25.



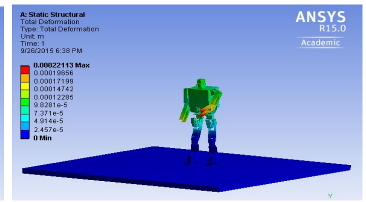
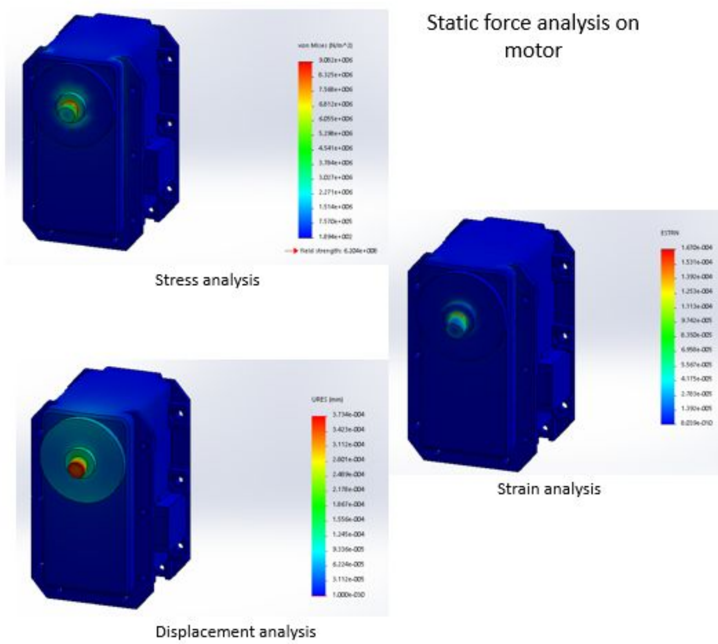
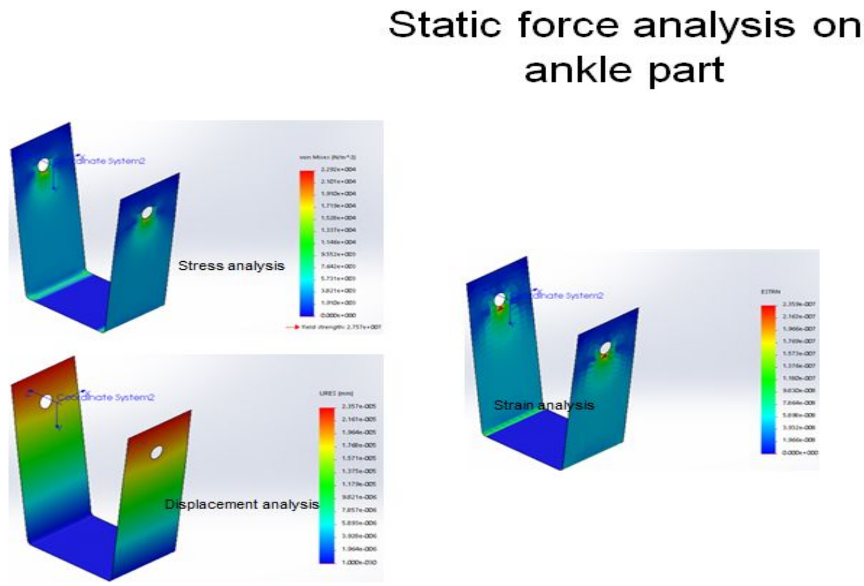


Fig 1: Stress Analysis *Fig 2: Strain Analysis*

We have also performed part-wise static force analysis-





Challenges Faced

While finding the D-H parameters, matlab code stopped due to lack of computation powers and redundant equations. This equations were computed again and again in order to correct the parameters but nothing was achieved till we corrected the transformation matrix.

Static Simulation results were derived first and they showed that the robot can stand only if we increase the foot area by 90%, but these would make it heavy and slow. Optimization of body structure was done in order to accommodate the humanoid in specified dimensions.

Static simulations results showed humanoid was not capable of walking unless hand was moved to an exact degrees. Finding these exact figures along with an equation was challenging and involved a lot time.

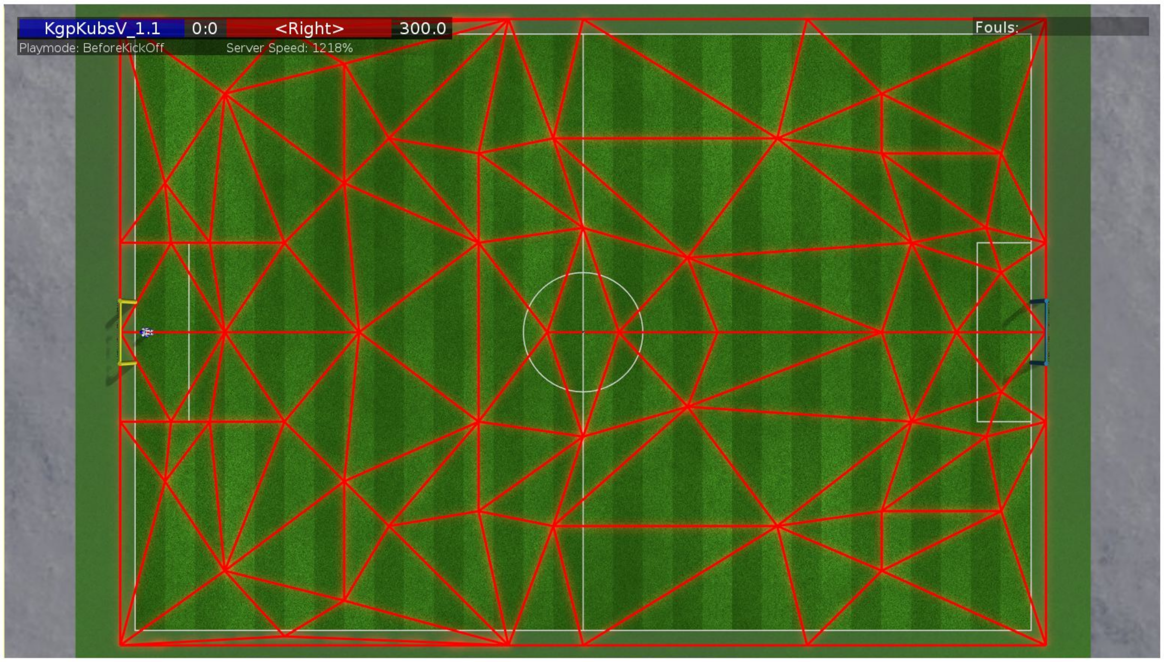
While finding the motor torques, 2 of the motors were in a state of breakage as the force and torque on them was much more than the tolerated limit.

Future Research Plans

1. Prototyping of bipeds first and analyse them: In order to see what the simulation results lead to in real life, we need to implement first only the walking legs in the hardware and see the exact movements and weight balance of them.
2. Static and dynamic analysis of each part individually: Analysis of various parts have been done to ensure they are within the tolerable limit. Designs need to be changed and analysis of each part has to be done carefully to find optimal relationship between them.
3. Reduction of material and looking for alternatives: Weight has to be reduced in order to make them more stable and easy to implement.
4. Modification in design according to real life results: Analysis of real life results and then implement those in simulations and then again modify the prototype.
5. Prototyping the upper body: After we are confident with the lower body, upper body of the

humanoid robot will be made and the entire robot will be put to rigorous testing.

c. Control and Intelligence Team



Overview:
We have a decentralized system in which 11 humanoid bots need to perceive the environment data and act accordingly. A 11 vs 11 Humanoid Football match is run on a simulator and a monitor is connected to the simulator using UNIX sockets. Each Agent(Humanoid) connects to the simulator and runs its code on individual processes. Joint perceptors provide the agent with noise-free angular measurements every simulation cycle (20 ms), while joint effectors allow the agent to specify the speed and direction in which to move a joint. Visual information about the environment is given to an agent every third simulation cycle (60 ms) through noisy measurements of the distance and angle to objects within a restricted vision cone (120◦). Agents are also outfitted with noisy accelerometer and gyroscope preceptors, as well as force resistance perceptors on the sole of each foot. Additionally, agents can communicate with each other every other simulation cycle (40 ms) by sending 20 byte messages.

Code Base:
Major Modules:

- WorldModel:

It contains various environment variables which are updated by the data received by the humanoid. This include joint angles of all agents, ball position, agent positions, etc. Our work on this includes :

1. Implementation of **Roboviz** debugging tools which includes plotting of various points(Target positions of bots), drawing lines when necessary(for having a visual of the goalie stand line) etc.
2. We also implemented various heuristic based functions :
   a. Finding the best bot to attack the ball.

b. Finding the nearest opponent(important parameters include FALLEN state of agent,direction in which agent is facing,distance from my position).
c. Finding the nearest teammate.

- **NaoBehaviour:**
  - *Walking:* We use a double linear inverted pendulum model omnidirectional walk engine. So, basically at the lowest level we have to send motor speed and angles but using a pre-built walk engine of AustinVilla we also have a set of parameters for walking. Optimizing this would get us a better walk speed. More about this in optimization section.
  - *Attacker:* The Attacker bot is arguably the most dynamic bot on the field.This role is selected based upon certain heuristics like fallen status,distance from opponent etc. The attacker can quickly dodge, dribble, kick(fast and slow) and drive ball to goal.
  - *Goalie:* The Goalie is the last line of defense and is the only agent allowed to purposely dive to try and stop a ball when the opposing team shoots on goal.Our goalie agent is designed to stay on a line .5 meters above its own goal line and always position itself between the ball and the goal so as to minimize the maximum angle between either goal post, ball, and the goalie. We apply **Kalman filter** to track the ball's position and velocity.We equip our goalie with a special set of diving skills including left/right dive , central stretch dive ..in order to effectively use its body to stop a ball that is headed toward the goal.
  - *Passing:* Passing is one of the most essential element for a multi-agent football playing system. We implemented a **fuzzy logic** based passing, which for each team member within a threshold radius takes into account.Following factors are taken into consideration while deciding the Pass.
    - target player's distance from source player
    - distances of opponent players from source player and target player
    - proximity to goal of source player and target player
    - angle to rotate for source player to face target player

    And based on these we compute a score, if the score is above some threshold we deem ourselves in a favourable position to pass, otherwise we continue as it is.
  - Dodge: The Agent plans a plan if an opponent is nearby such that he dodges the opponent.
- **Positioning:**
  - Delaunay Triangulation and Formations:

    Delaunay triangulation is the Dual graph of Voronoi cell plane. Hence, Delaunay triangles ensure that no other focal point lie inside the circumcircle of the Delaunay triangle formed. Also, due to this property it tends to avoid skinny triangles. As a result, interpolating any point inside the triangle yields to a smooth-gradient continuous equation in terms of the coordinates of the vertices of the triangle.

    The algorithm used to generate player positions uses statistical data (bot and ball positions under different conditions of the game) and generates a data set of agent positions with respect to certain ball positions. In all 65 ball positions in strategic locations were identified and triangulated using the incremental algorithm to generate Delaunay triangles . For each ball position we defined formation of agents. Once the triangles are generated, the **Gouraud Shading** algorithm yields the value of bot positions at any given point in terms of the values of bot positions stored at the vertices of the triangle enclosing it.

- **Role Mapping:**
    - Hungarian Role Mapping: We use hungarian algorithm which solves the role assignment problem in polynomial time. Time complexity of this algorithm is O(n^3). At every one second,as per the ball position, we get a set of target Points from voronoi triangulation.These set of target points are then matched to players on field by hungarian algorithm. The cost function used for hungarian algorithm is euclidean distance between bot's current position and target location. It is also easy to see visualize that using this type of role matching has following properties:

      (a) The collisions are mostly avoided.

      (b) Longest distance is minimized

      (c) It is dynamically consistent
    - Prioritized Role Mapping:
        - The role map generated using hungarian is not always reliable for some crucial roles like attacker, defender and support attackers.
        - This roles are to be assigned based on game scenario rather than optimizing the distance travelled. We use heuristics to determine these roles. This gave a lot of advantage in the matches.
- **Optimization:**

  For optimizing low level skills, like walking, it may seems that reinforcement learning is more suitable but on contrary **CMA-ES** performs at par with the RL algorithms. We treat it as a black-box optimization algorithm, and have not interfered with the algorithm itself. For training a different cost function for optimizing different skills was used. For example, commanding the bot to go straight and using the difference in x-coordinate optimizes walking straight as well as improves its speed. Similarly other walk types (like lateral walking) and other skills like kicking could be optimized. Here we provide a brief about the CMA-Evolutionary Strategy:

  CMA-ES is a policy search algorithm that successively evaluates sets of candidates. Each candidate is evaluated with respect to a fitness measure. The next set of candidates is generated by sampling multivariate normal distribution that is biased toward directions of previously successful search steps. Recombination amounts to selecting a new mean value for the distribution. Mutation amounts to adding a random vector, a perturbation with zero mean. Adaptation of the covariance matrix amounts to learning a second order model of the underlying objective function. It is a parallel search algorithm so it can be run on a large server to make the optimization feasible. Some parameters were carefully chosen for optimization keeping others constant to reduce the search space.

  Future Course of Action

  Currently we have used CMA-Evolutionary Strategy for improving low level skills, we aim to use deep reinforcement learning along with exciting new research in asynchronous actor-critic algorithms. :

  1. Deep RL algorithms based on experience replay have achieved unprecedented success in challenging domains such as Atari 2600. However, experience replay has several

drawbacks: it uses more memory and computation per real interaction; and it requires off-policy learning algorithms that can update from data generated by an older policy.

2. Instead of experience replay, one of the key insights is that you can achieve many of the same objectives of experience replay by playing many instances of the game in parallel.
3. Advantages of A3C: As our research problem motivates the parallelizing thing, we must start in this direction only, possible alternative are using evolutionary algorithms but CMA-ES is the state of art nonlinear evolutionary algorithm based optimizer and a huge amount of research has already been done with it. Another alternative being improving on simple RL based algorithms but with the OpenAI providing a simple simulation based environment this has already been looked upon.

Challenges Faced

The CMA-ES algorithm takes 1-2 weeks to train and due to this we can't scale it to every problem given the current resources we are using. With the advent of Deep Reinforcement Learning, the complexity would only increase.

d. Timeline

Current Status	The solidworks design for the humanoid is ready and static and dynamic stress analysis is also complete. The analysis of a biped is complete and ready to go for prototyping.
Year 1	a. Study and implementation of robot kinematics, stability and control b. Simulation and analysis of biped c. Prototyping and testing of biped
Year 2	a. Development of Upper Body (Design and Simulation)
Year 3	a. Full Humanoid Prototyping and testing b. Debugging errors and further developments.

e. Budget

Item	**Cost**
Motors - 23 No	10 Lakh
3 D Printer	2 Lakh
Electronics	3 Lakh
Manufacturing	2 Lakh
1 Nao Bot to be purchased	10 Lakh
Nvidia TItan X(P) or AWS	1 Lakh
Misc	2 Lakh
Total	**30 lakh**

3. Code-O-Soccer

Overview

Code-O-Soccer is a coding competition conducted by Kharagpur RoboSoccer Students' Group. This is a first of its kind competition wherein soccer strategies brewing within one's mind are implemented on robots using techniques of Artificial Intelligence.

The main aim of the event is to introduce the concept of autonomous soccer playing robots in students mind and motivating students to create a challenging strategy using our API on a THREE BY THREE robot match for which robots will be provided by us during the event. The participants will also be provided with a simulator with game environments (playground, robots, score board, etc.) to test their codes.

Current Status

The event and the hardware has been framed on the basis of an international competition - **Federation of International Robotics Association (FIRA).** We have been participating in this competition for the past 4 years and have recently secured bronze position in it. Presently, we no longer intend to continue this participation and use these robots for further research and development. We are also looking forward to encourage the robotics enthusiasts in the country to contribute to our research through their participation.

We have successfully organised two seasons of this national competition with participation from more than 50 colleges and 100+ teams. We need financial support to continue the research and keep the competition alive.

Challenges Faced

1. THe current bots are not working at their optimum speed because of problems at mechanical ends and communication systems. While running at higher speeds, the wheels loosens and comes out limiting the performance of the bots. In our centralized communication system, 9 bytes of data is being sent at 16ms which is too much for the current system to bear. We are facing data loss and latency issues.
2. Kinodynamic path planners are the state of the are planners in 2 wheeled robots where the motion of the robot is highly constrained by its kinematic constraints. The idea is that during replanning of a path the new path should consider the inertia of the last motion

Research Aims

1. We are planning to shift the control board from and ARM processor to FPGA as discussed in the SSL section. We are ready with the closed loop motor control and communication. They have already been tested and we need to implement then on main circuits for 5 boards and optimize the processing.

2. We plan to implement kinodynamic path planners by using Clothoid equations in order to generate smooth paths during path transitions, keeping in mind the current inertia of the bot.

Event Goals

1. Bring coding and robotics enthusiasts on a completely new platform to encourage robotics among students at national level
2. Prepare a completely ready set of robots for robotics enthusiasts across the country to check their algorithms without any cost on hardware.

Timeline

Current Status	Bots for 3 vs 3 match are ready and have been used for 2 seasons of Code-O-Soccer.
Year 1	We aim to improve the communication and shift the controls from ARM controller to FPGA. By this year, we aim to increase the speed of the bots.
Year 2	Prepare the set of bots on FPGA
Year 3 Onwards	Conduct matches at national level on the hardware

Budget

Item	**Price**	**Quantity**	**Cost**
Motors	17000	12	204000
Batteries	1000	6	6000
X Bee	2000	6	12000
ARM Controllers	1000	6	6000
Motor Controllers	500	6	30000
Circuit Components	2000	6	12000
Misc			3000
Total			**2,40,000**

Proposed Budget

Project	Proposed Budget
Small Sized League	16.7 Lakhs
Humanoid League	30 Lakhs
Code-o-Soccer	2.4 Lakhs
Total Budget Required	**49.1 Lakhs**