A PROJECT REPORT ON COMPETITION ROBOCON'18

FACULTY ADVISORS: Dr.Anand D.Darji
Prof. Pinalkumar J.Engineer
Dr. Sankita J.Patel

MEMBERS

Yash Patil
Ajay Ghori
Parth Patel
Bhagyaraj Beladiya
Abdullah Rangwala
Mahesh Birajdar
Manthan Shah
Parth Panchal
Rahul Ashar
Rutvik Patel
Shubham Muke
Vatsal Rathod

Kiran Prajapati
Animesh Panara
Deepeshwar Kumar
Abhinav Jain
Atharva Kalsekar
Dhruv Patel
Himanshu Kumar
Sandeep Ashiwal

ACKNOWLEDGEMENT

We , team DRISHTI would like to thank every individual and whole Drishti members in the successful happening of Robocon 2k18 for our college .the team thanks faculty advisors Dr.Anand D.Darji , Prof. Pinkalkumar J.Engineer ,Dr. Sankita J.Patel and the Dean Academic of SVNIT for their kind support and guidance at every step. It also thanks DRISHTI - A REVOLUTIONARY CONCEPT for providing lab workspace to the team.

ABSTRACT

The document mainly focuses on the development, timeline , methods and the challenges faced to develop two robots (manual + autonomous) that was completed by Team DRISHTI .

TABLE OF CONTENT

Arena

- The Arena was made with dimensions according to what was mentioned in the rule book of ABU Robocon 2018. As the arena was diagonally symmetric, only one of the portion was developed.

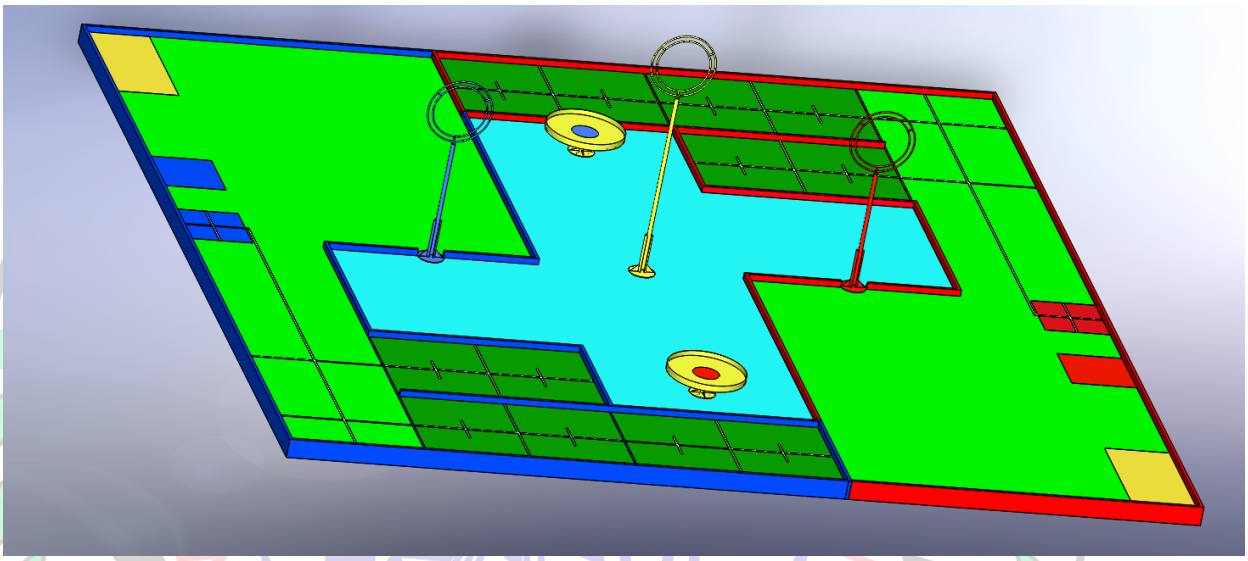
Pole Making :-

- Pole was to be made of mild steel (which was purchased from allarakhu fakirahamad).For maintaining strength and restrict bending and shear, it was necessary for it to be welded.
- Welding requires a jig which provides advantages such as
  1.After bending both the circular rings of pole should be concentric.
  2.For accurate dimensions
  3.During welding to hold the rod to be bent.
- The mild steel rod purchased was bent(from lucky welding and engineering works).
- This rod was welded after bending to form concentric rings.
- The supporting rod was attached to rings by welding.
- Flanged to support the rods are welded at the bottom of rods.
- Height of the rod was kept less than specified because it was to be given extra support of plywood which increased the height.
- Plywood was fixed to pole with the help of foundation bolts that were countersunk inside the wood.

Problems faced and Solutions

- Earlier, a pole design with detachments was thought of. In this the pole height would be divided into two and joined through flanges and the ring was to be fixed through inserting in the grove in the pole. This would

increase the feasibility of transportation of the poles. But there was a problem of linear alignment during welding.Also there was a risk of ring falling off the groove. Thus this idea was dropped and whole pole was welded.



CAD MODEL

Floor making :-

- Floor was decided to be made of vinyl sheets(purchased)
- These vinyl sheets were to be applied with proper technique.
- Straightening of vinyl sheet requires squeezy for pressing.
- Before applying vinyl sheets floor was cleaned to avoid bubble causing impurity.
- The vinyl sheets were applied accurately according to the markings on the floor and also with respect to colour codes.
- White and red vinyl strips were applied at appropriate distances from the edges according to the given arena.

Problems faced and Solutions :-

- Earlier it was thought to make the flooring from cemented sheets.This idea was cancelled as the required amount of cemented sheets were not available.Also there was a problem of painting the sheets.This would have been time consuming and costly.

- While applying vinyl sheets, many creases were formed on the floor due to improper applying techniques. This problem was minimised by carefully applying the Vinyl sheets with just the required force on the squeezy.
- The creases formed were removed by cutting the portion of creases from the floor and then applying another cutting of vinyl sheet in the place of cut portion on the floor

FINAL ARENA

Wall Making

- Wall was initially made by combining plywoods to achieve appropriate thickness.This idea was discarded as plywood does not give uniform thickness throughout.Also alignment of this wall in a straight line using technique of header joint technique used in bricks alignment in civil engineering gave a big problem.Wall was fixed to floor using M-Seal which did not provide enough strength.

- It was thought of making wall from teak wood.But there would be impressions on teak wood after lots of and also would be costly.Teak wood is also brittle reducing the strength.
- To get strength and straightness it was decided to form a wall from cement which had mixture of sand,cement and water.This was too weak to lift it up.For strengthening purpose mixture of mortar,white sand and cement in ratio of 1:2:5 .This structure had its own problems of cracking,shrinking and finishing.This wall used to break from weaker sections.So this idea from cement was discarded.
- The next idea of using box section combinations instead of plywood was implemented.This idea gave uniform thickness but there was bending of some box sections which created problem of height.
- Further a box section was bought which was a single piece and not combination of box sections.This reduced the height variation problem.The bending problem was solved by placing appropriate weight on box sections.This wall was fixed on floor using a groove made of teak wood sections placed at appropriate distances.

FINAL WALL

AUTONOMOUS BOT

- Two mechanisms have used in autonomous bot. Throwing mechanism for throwing shuttlecock whereas reloading mechanism to pass extra golden shuttlecock to throwing mechanism.

THROWING MECHANISM

- Rotating Link Mechanism
- Pantograph Mechanism
- Catapult Mechanism
- Extension of Catapult Mechanism

Rotating link Mechanism

- Centrifugal force is the force arising from the body's inertia, which appears to act on a body moving in a circular path and is directed away from the centre around which the body is moving. Thus this application was used under this mechanism, which would throw the shuttlecock through a rotating link.
- The mechanism consisted of a Motor mounted at a specific height above the ground on a stand. The motor shaft is connected to the rotating link indirectly via two spur gears.
- The end of the link had special type of plate fixed. The plate had a v-shaped groove with end having a drilled hole at the end of the ‘v’, which would support the string attached to the shuttlecock and block the knot.

Different Motors were used for having the best results possible and eliminating errors.

Trial 1: Using Johnson motor (300 rpm)

Problems faced:

- Coiling of string on the shaft (less centrifugal force due to less rpm)
- Low starting torque

Trial 2: Arm at an angle using same Johnson

- Problem of coiling of string is solved!

Problem faced:

- Unbalance in mass of the rotating system
- Failure of gear of Motor shaft due to high load
- Release angle of each shuttlecock was different even starting position was same

Trial 3: Motor having an Encoder (Planetary) to set same starting position and to calculate rotations

- The problem of setting initial condition was solved.

Problems faced:

- Due to less starting torque, it was not giving same results after each try, even remaining parameters are constant
- Problem of Release angle was still there
- Failure of Gear box of motor due to high inertia

- To manufacture a clutch on rotating link with electronics (last year we tried this for about 2 months, but still it was not giving desired output although this year we tried but we didn't get desired output)

Trial 4: Maxon motor (high starting torque as well as rpm) having Mechanical clutch.

Problems faced:

- Different release angle at each time
- CG of mechanism was shifted upward, so it was creating problem in stability of bot which was the most dangerous problem
- Fire rate was very low
- Chances of failure of Gear box of Maxon motor (which is too expensive)
- So finally, we concluded that it was next to impossible to make a throwing mechanism by using Motors.
- Thus we had to shift our idea towards something that would provide better results. The major errors were discussed and finally the outcome was to make a mechanism that would throw the shuttlecock in a single pass, which means in a single impact. This decision was taken as major problems which were previously faced could be eliminated.
- So, we shifted to pneumatics (Giving impact at the end of stroke)

Benefits of Pneumatics based mechanism:

- Repeatability was high at a given pressure. (Which is not possible by using mechanical pressure regulator)
- Manufacturing is quite easy

Problems of Mechanical Pressure regulator (FRLV- Flow Regulator Lubrication Valve):

- We need pressure range of 1.5 to 3 bar. According to the characteristics of Mechanical pressure regulator, it gives constant pressure output in the range of 3.6 to 7.5 bar even in the new one. So, between 1.5 to 3 bars, it's giving fluctuating pressure output.

Pneumatic Mechanisms

A) THE CATAPULT

- A catapult is a ballistic device used to launch a projectile a great distance without the aid of explosive devices.
- In use since ancient times, the catapult has proven to be one of the most effective mechanisms.
- In modern times the term can apply to devices ranging from a simple hand-held implement (also called a “slingshot”) to a mechanism for launching aircraft from Space Ship.
- Thus keeping in mind the following factors, the focus was targeted on designing an effective catapult.
- The aspects to be kept under the attention were the total length of the bot allowed under the rulebook and achieving the required velocity which was enough to provide the given range.
- The repeatability also was a major criteria since the bot remains autonomous.

Problem faced:

- Range was not enough
- After each stroke, pressure remained uncertain in mechanical pressure regulator

B) PANTOGRAPH MECHANISM

Pantograph is a mechanical linkage connected in a manner based on parallelograms. The major use of this is done because of the property of multiplication. The multiplication takes place in the opposite links of parallelogram. So, this mechanism was used for moment multiplication. The mechanism thus designed under this Pantograph design showed a multiplying factor of 5. The force was applied using pneumatics.

Problem faced:

- Constant pressure output which is mentioned above.
- Then we tried different ways for constant pressure output.
- Electrical flow regulator, (we need constant pressure, not flow rate) which was not giving desired output. For that by changing flow rate (at constant pressure) we tried to throw, but it repeated the same error mentioned above.
- Using different electrical pressure switches and different mini compressor (which consumes more power which is not desirable. Total pricing of all these components are quite high. Although it was not giving constant output as lowest pressure difference in switch is 0.7 bar)
- Inquiry for new Mechanical pressure regulator of having higher accuracy, but it also didn't give desired pressure output (information given by JANATICS which is a manufacturing company of pneumatics)
- (For reference:http://janatics.com)

Videos :-

- https://drive.google.com/folderview?id=0B1p3SBvO3qI1WFBlUG95U0hMeE0

- https://drive.google.com/folderview?id=0B1p3SBvO3qI1OXlHMER0OGd6Q1U

C) MODIFIED CATAPULT MECHANISM

- This mechanism consists of fixed horizontal pneumatic cylinder which is connected to single hinged throwing link with connecting link.
- The idea to move towards this mechanism mainly was to occupy the smallest possible space and involving easier manufacturing.

Trial 1:

- Pneumatic specification: Airmax stroke length=200 mm

  Bore diameter=25 mm

  Material Used : 1*1 Aluminium primary box section.

Length of link :

- Throwing link : 800mm
- Hinged point (respect to ground) : 185 mm
- Hinged point in throwing link : 200 mm
- Connecting link :150 mm
- Distance of pneumatic from hinged point : 175mm

Working

- The Pneumatic cylinder is open in initial state. When pneumatic cylinder is closed by 5/2 valve , the trajectory was created by throwing link was projectile

Problem faced:

- Total length of this mechanism is not reliable for the autonomous bot.
- The force generated by pneumatic is not capable enough to throw the shuttlecock on required place.
- Very high pressure is required, approx 5-6 bar.

Trial 2:

- Pneumatic specification: Janatics, stroke length=150 mm

  Bore diameter=32 mm

  Material Use: 1*1 Aluminium primary box section.

Length of link :

- Throwing link : 590 mm
- Hinged point (respect to ground) : 165 mm
- Hinged point in throwing link : 70 mm
- Connecting link :100 mm
- Distance of pneumatic from hinged point : 75mm

Working:-

- The Pneumatic cylinder is close in initial state. When pneumatic cylinder is open by 5/2 valve , the trajectory was created by throwing link was projectile.

Problem we have faced:-

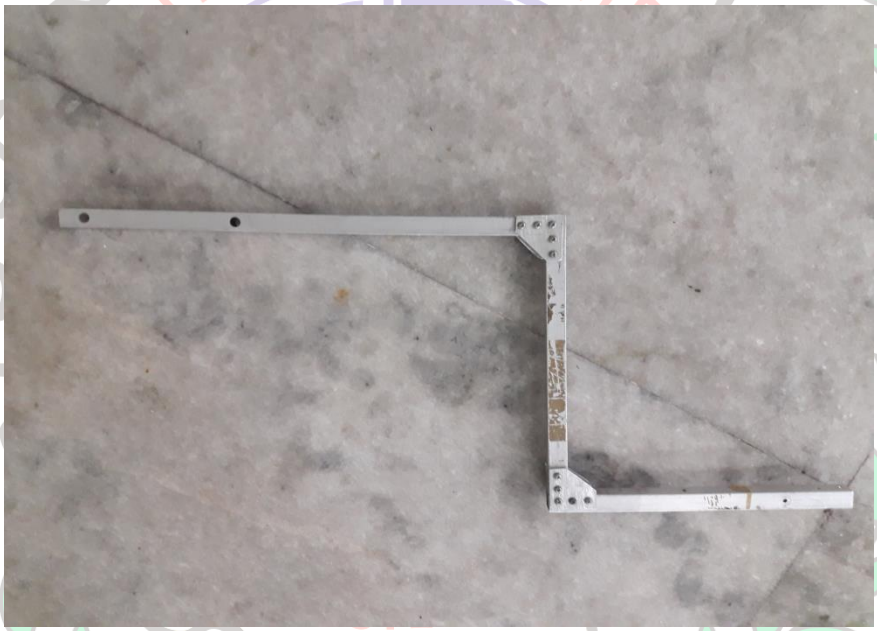
- The output results were good in terms of requirements but the dimensional limits were crossing as per the rulebook, so this mechanism had to be discarded.

Throwing Mechanism Link

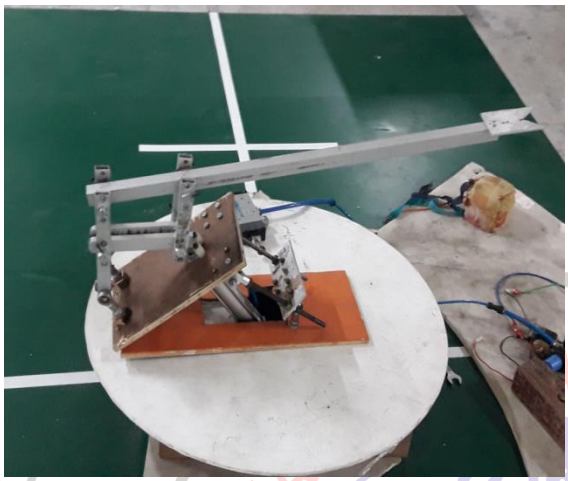
Problems faced and Solutions

- The height of the autonomous bot was exceeding the given limitations because of the required length of throwing link.

  Solution :- The throwing link was modified and made step-like. This solved the height restriction issue.



- The Throwing link exercises high impulse due to jerk produced by pneumatics. Due to this the link tends to bend in the forward direction. This was due to lack of strength of the throwing link in the throwing mechanism.
- Solution :- Different materials were experimented for required strength such as :-
- Aluminum 1”x1” Secondary Box Section - Failed
- Aluminum ¾”x ¾” Primary Box Section -Failed
- Aluminum ¾”x ¾” Primary Box Section with FRP coating at the joint - Failed
- Stainless Steel ¾”x ¾” Box Section – Failed



- Aluminum ¾”x ¾” Primary Box Section with FRP coating on whole link.

Reloading Mechanism

Requirement:

- The idea behind the use of an extra mechanism was to pass one extra shuttlecock at an instant which would save the time for collecting other shuttlecock from manual bot.
- It also allows autonomous to throw more than one shuttlecock which would increase the probability of golden shuttlecock to achieve rongbay.

Design

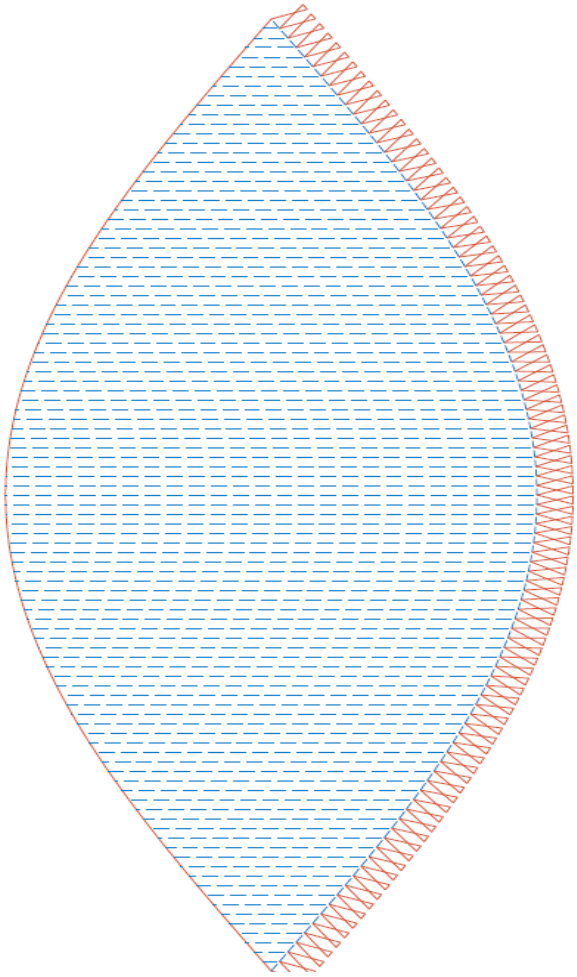
- It consist of four links based on simple four link mechanism.
- Motor was connected to one link which is called rocker through a thrust bearing.Here the motor used was Machtex.
- For loading shuttlecock on this reloading mechanism a V groove similar to throwing mechanism was mounted on the connecting link between two rockers.
- The V groove was bend a little in upward direction so that transfer of shuttlecock from manual to autonomous occurs successfully
- The motion of both the rockers were restricted using limit switches which were mounted on both extremities.
- These positions were fixed and should be accurate so as to get proper passing of shuttlecock from reloading mechanism to main throwing link.
- This reloading mechanism was mounted on a plywood which acted as a base.
- A pneumatic piston was attached to the base of this reloading mechanism to change the height at the time of reloading shuttlecock.
- The reason for this was to ensure that shuttlecock is successfully passed to throwing mechanism.
- Its actuation followed these steps in continuation:-
- Actuation of pneumatic that increased height of mechanism.

- Actuation of Machtex motor to get the desired angle of rotation.Rotation which was limited with the help of limiters.

Shuttlecock

- Different shuttlecocks were made with different sizes,shapes and filling material.
- The first try for making a shuttlecock was a simple sphere and the filling material was black foam.The sphere shape was covered by like tennis ball manufacturing. Problem faced was shuttlecock was too heavy and foam is not easily available.
- Followed by that shuttlecock having sphere shape was made.In that shuttlecock the covering of cotton fibre was stitched with high density flexible foam.Then six sides of the sphere were stitched together and the filling material was only cotton.The shuttlecock was landing properly but problem faced that shuttlecock was bouncy in nature and no damping of shuttlecock and jumping out of golden cup.
- Followed by that shuttlecock having cone shape was made.The shape was covered by calculating desired diameter of shuttlecock.Filling material was only cotton.Problem faced was trajectory of the shuttlecock was uneven and that was not giving constant result.
- After that shuttlecock having hexagon shape was made.The shape was covered by stitching six sides and top and bottom.Filling material was cotton.Problem faced was trajectory was not as desired and it was not satisfying dimensions in all directions.
- Followed by that shuttlecock was made having sphere shape with filling material as cotton and different concentrated load.Firstly a wheel having mass of 43gm as concentrated load was filled. Proper trajectory as well as range were achieved but we have to maintain both constraints weight and diameter. To maintain diameter the weight is exceed over the limit.Still damping problem was there.Different concentrated load like piece of teak wood,rice bag,rolled ball of cotton strip of around 40 gm weight were filled and almost same results were achieved as above in wheel.Problem faced were same as above.

- After that shuttlecock having cube shape was made with filling material cotton with candy sticks. Better result among all of the above shuttlecocks were achieved by this shuttlecock .But still there was problem of dimensions.
- After that the shuttlecock having cube shape was made with filling material as paper pieces.Problem faced was more drag due to distributed load and less weight of filling material.
- After that we switched to sphere shape. Sphere shape was covered by stitching six pieces of specific nature.The nature of the one piece can be get by online simulator.For desired diameter and number of pieces of sphere we can get the nature the curve by simulator.



For 145mm Diameter

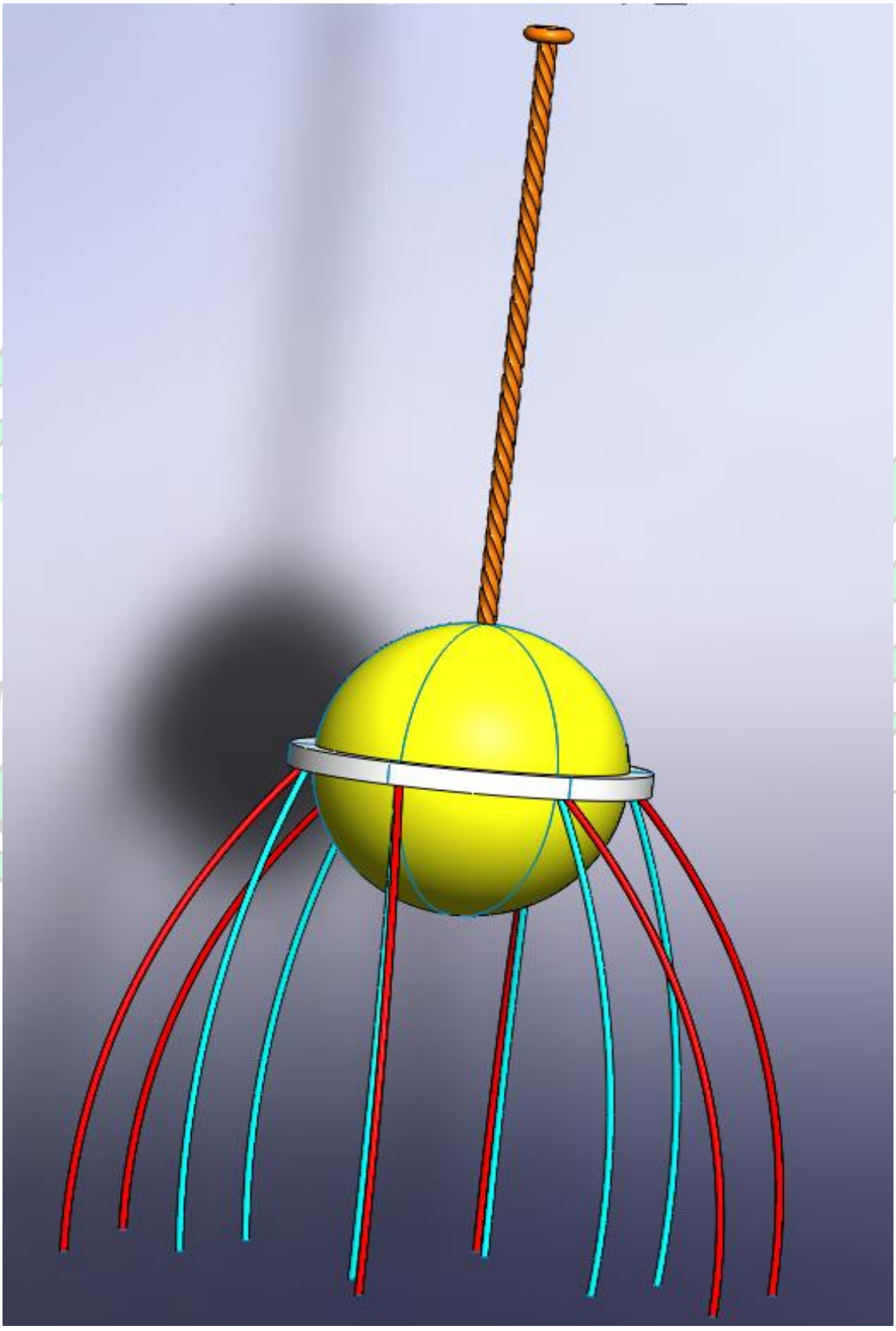
- After that the shuttlecock was made by velvet cloth with filling material as candy sticks and cotton.The problem faced was more drag due to velvet cloth and desired height was not approached.
- After that the shuttlecock was made with specific type of cloth called “Dhupiyan” and filling material as candy sticks and cotton.
- Still there is problem with dimensions.So that ring having material as foam was stitched around the shuttlecock.

Problems Faced:

- This ring was not allowed in the Super League matches.
- Length of fringes were not satisfying the standard rules.
- Number of fringes were less than 5.

Solution:

- Ring was removed with addition of cotton sticks equivalent to weight of the foam in shuttlecock.
- This added to damping of shuttlecock along with dimensional increase.
- Fringe length was increased by attaching cloth strip to original fringes by adhesive(vetra).
- Additional fringes of appropriate length were stitched to shuttlecock.



CAD model of final shuttlecock

Tail:

- Firstly tail of shuttlecock was made by three cotton cloth strips.Problem faced was cotton string was not proper because while releasing shuttlecock string tends to bend and didn't get proper result.
- Then string having polyester material was used to overcome the problem which is faced by cotton string.Desired results were achieved by this material of string.
- To achieve multicolor property of tail laces of three colour were wound on nylon fibre. This method created the problem of unwinding of laces. Stitching these laces created bending of string. Also this created non uniform thickness of tail.

Multicolour tail

Problems faced:-

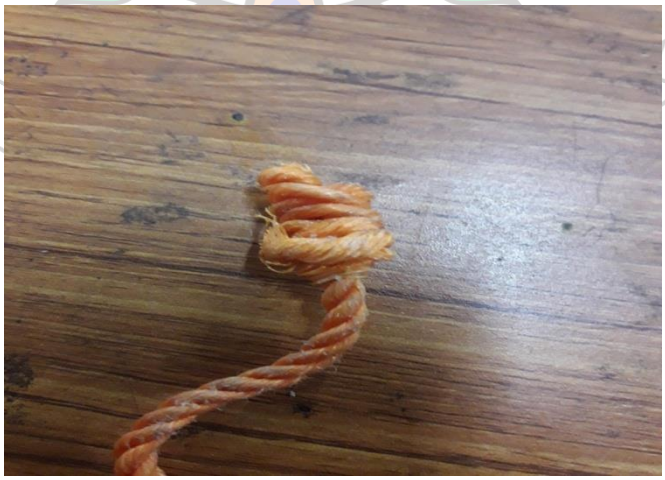
- These tails were not so rigid so they changed their keeping point structures over a period of time.
- Keeping point structure change resulted in random throwing angle of shuttlecocks.

Keeping Point:-

- A special keeping point design was finalised for holding the shuttlecock. The keeping was made such that its lower surface is uniform and smooth.
- First, a small length string of same material as of the tail is cut.
- Then a single wound is made with this cut string on the tail at the required distance(260 mm) from the shuttlecock.
- The string is then sewn to the tail to temporarily hold the string in that position.
- The free ends of the string are now held together through paper tape.
- A single drop of vetra(an adhesive) is then applied on the sown part and is allowed to dry for a few minutes.
- Then a plier is used to grip the string properly around the tail and vetra is applied all over the string and is then allowed to dry for a few minutes.
- The plier is removed and the extra portion of string is cut off.
- Again vetra is applied on the cut portion and allowed to dry.
- Then the cut portion is made smooth with the help of cutter.

Problems faced:

- At first, a plain knot was tied at the keeping point to hold the shuttlecock. This idea was rejected because each knot had different uniformness thus giving different results during testing.
- Then a different type of knot was tested. This knot gave better results than the previous but still the results were varying and the knot was not uniform.



Manual Bot:-

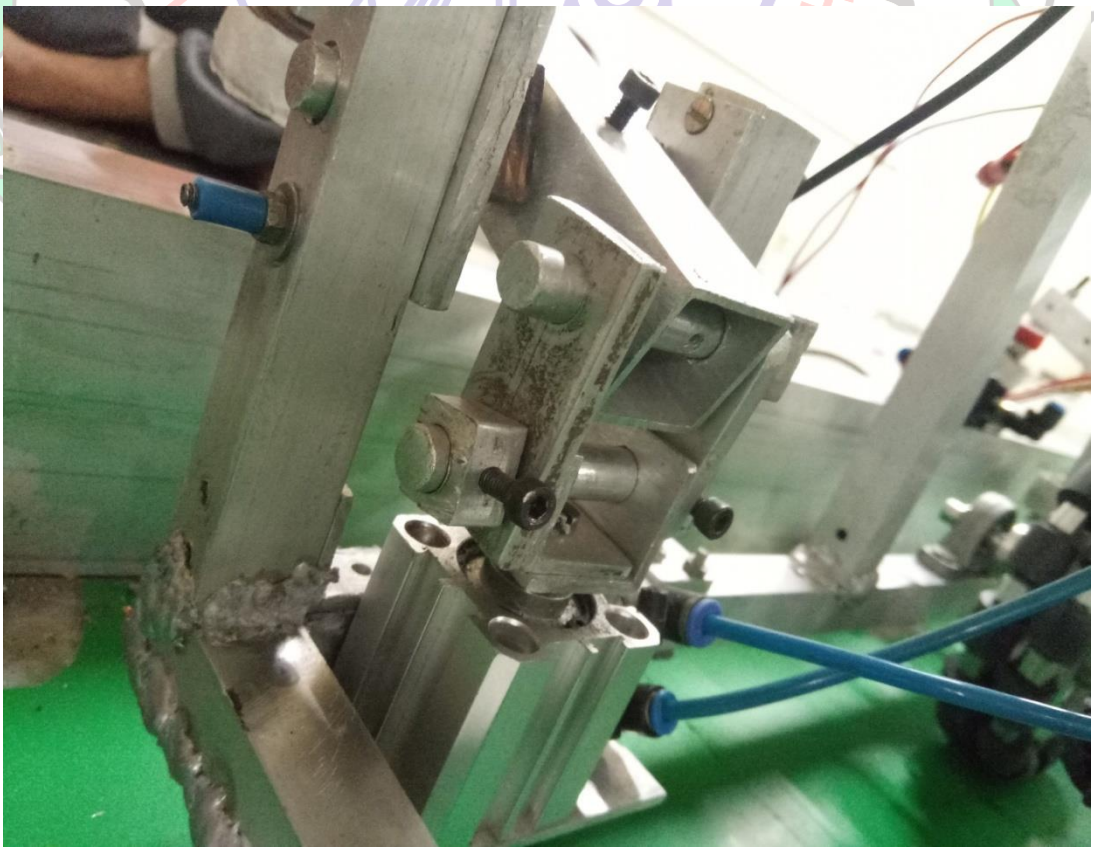
- 3 wheel omni drive with Maxon motors is used for manual bot. The angle b/w the wheels is kept 45-90-45.

Different mechanisms used in manual bot are

- **Wall Clutch**
- **Striker Mechanism**
- **Slider Mechanism**
- **Rack Clutch**
- **Rack Holding**

Wall clutch:-

- The idea behind this mechanism is to guide the manual bot using wall.This mechanism is basically a clutch type mechanism.It usually grips the wall from both the sides with the help of castor wheels.The castor wheels rolls over the wall and also act as constraints for the manual bot which makes the bot to follow a straight path.
- This mechanism consist of one pneumatic cylinder(stroke length 20 mm and bore diameter 25 mm) each and two casters each on either side of the wall. The mechanism the Piston rod of cylinder is connected to a box section through a connecting link which contains a caster on the other end. This box section is connected by means of shaft which is free to rotate on two columns vertically brazed to the manual bot.When piston is fully extended or is at top most position the caster will make contact with the wall.



- This position will remain as it is until it is unclutched. This gives constraint to the bot from outer side but the bot still cannot follow a linear path as it is lacking another constraint from inner side. To do this another caster which is bolted to the chassis is used and mounted at a particular distance from the chassis.

- This mechanism is attached to one side of the manual bot and pair of each of these mechanism is brazed on corner sides.

Advantages:-

- By using wall clutch the manual bot can be of open loop driving system which would be simple and efficient as compared to close loop. The use of this mechanism makes easier for driver to drive the bot, as there is no feedback from the system.

Problems faced:-

- Mechanism is to set at a definite distance so that bot should maintain a safe distance from the wall as well as it should perfectly mate with the autonomous bot waiting at junction.
- Initially the dimensions of wall clutch was such that the axis of castors were parallel to wall.In this the problem was as it was difficult to make proper contact with the wall and there was also problems of slipping of castor surface.
- As a solution the design was changed in which the axis of castors was kept at an angle to the wall. Due to this the mechanism could also contact easily with the wall and is able to maintain appropriate distance.

Why Wall Clutch was REJECTED?

- There was uploaded FAQ's by ABU robocon international which states that we can not use wall clutch.

Striker Mechanism

- Shuttlecock has to be passed from rack to the autonomous bot so that autonomous bot will throw it through rings.For this purpose striker mechanism is made which projects the shuttlecock from rack to the holding plate of throwing mechanism.
- Pneumatics is used in striker mechanism.Pencil piston cylinder of 10 cm stroke and 20 mm bore dia is used for this.Main purpose of using such piston is we required less pressure for striking.

Piston mounting:-

- Piston cylinder is to be mounted at 3 degree angle for initial trial. Purpose of such angles is to keep the shuttlecock steady when the manual bot is moving and prevent it from falling down.after mounting the cylinder at 3 degree a L clamp was attached to the front of piston by boring a hole of 8 mm dia on one face of clamp.

- Striking Plate Mounting:-

  A plate which was given a V shape was cut.At the vertex of v a 10mm dia was drilled for the knot to keep.This plate is attached to the non bored face of L clamp.

- Temporary Rack:-

  A temporary Rack was cut of U shape. It is mounted on padding (in a row along the movement of piston)of appropriate height with two bolts below the striking plate.Angle of rack was managed with the help of washers on the front bolt of front padding.

Problems faced

- If we had mounted piston perfect horizontally, shuttlecock may slip from rack .As solution we have tilted the piston along with rack by 3 degrees.
- Bolts on striking mechanism used to hit the shuttlecock on rack.
- The striking plate rotates due to no constraint on rotation.

Solution:-

- Angle measured with help of compass.
- Wider striker clamp than rack.Bolts were placed away from rack.
- Mechanism modified with zero degree inclination.

Striker mechanism side view

Pneumatic Mounting:-

- Pneumatic cylinder of 20cm stroke length and piston dia 8mm is used.This length gives sufficient stroke to project the shuttlecock.
- Pneumatic cylinder is mounted on aluminium box section of 1”x1/2” primary(giving sufficient strength for bearing bending stress).
- In this trial cylinder is mounted horizontally at 0 degree. After lots of practice it was concluded that angle had no effect on the steadiness of shuttlecock.
- So the cylinder was mounted horizontally without any inclination with the help of L clamps.Back L clamp of 3 mm thickness and front L clamp of 5 mm thickness. This thickness variation gives shear force resisting strength.

Striker Clamp:-

- This L clamp has both the features as of striker plate and clamp used in trial 1.This clamp has got two holes on one face and V cut on other face.
- This provides combined advantage of reduction of weight and reduction in assembly.The two holes on a face are of 8mm and 10mm for piston and guiding rod respectively.
- They are at a same distance as in front clamp in piston mounting.Thickness of this clamp is 3 mm. Small thickness is provided as its functioning does not include force bearing.

Guiding Rod

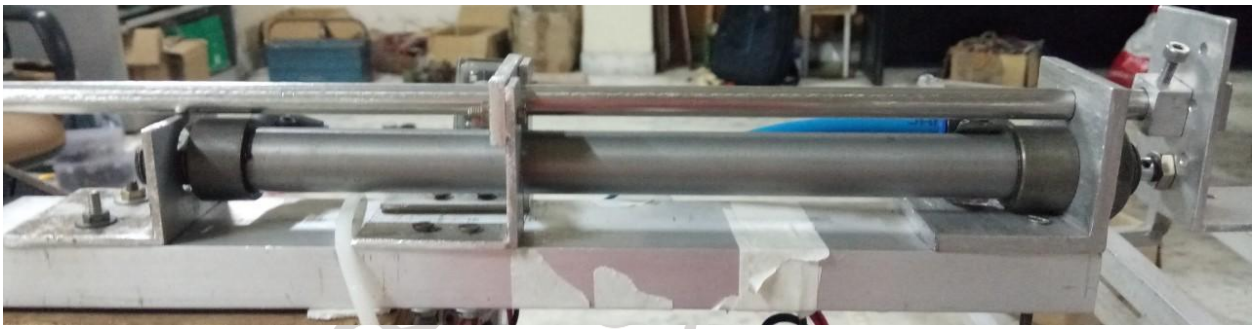
- This is a 10mm dia mild steel rod.Its function is to give guidance to pneumatic piston without rotation.for fixing this guiding rod to striker clamp collars with tappings are used to hold the striking plate at a position with guiding rod.It is specially provided for avoiding rotation problem arise in trial

Problems Faced :-

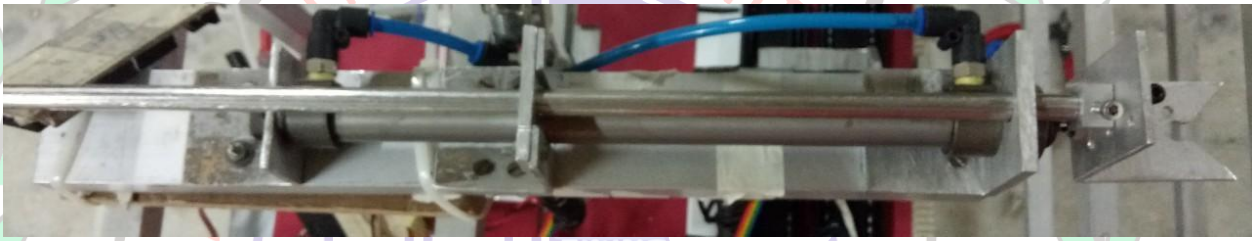
- Holes in the striker clamp are not equidistant as in front clamp of pneumatic mounting.
- There was no L clamp to manufacture striking clamp of required dimension.
- Guiding Rod was not stable and moved due to its weight sideways.This caused rotation of striker clamp to some extent.
- Guiding rod touched the back plate of the pneumatic mounting there by restricting its motion backward.
- The mounting of striker created the problem of height difference between the striker front L clamp and the rack upper face.This height difference was of about 1 cm.

Solution:-

- Made a striker clamp using front clamp of pneumatic mounting as formo.
- A clamp was reused which had holes already drilled in it.This reduced the weight along with accomplishing the required task.As strength was of least importance here it worked else there was a need to buy a new L clamp section for a small piece.
- Rotation of striker clamp to some extent was mainly due to single constraint on guiding rod.This was modified by applying another constraint at the back of guiding rod.This was done by mounting a plate having a hole of 10mm dia (same as mild steel rod) on two L clamps.Due to this second restriction the length of mild steel rod was increased with length being equal to summation of stroke length and distance between two constraints.
- Reduced the height of back plate to provide sufficient clearance from the hole.
- The striker mounting problem was solved by adding the hinge to striker.Addition of this hinge made auto adjustment of height for any variable height difference between striker L clamp and rack upper face.
- As we got the desired result from this trial ,this design was finalled.



Striker Mechanism side view



Striker Mechanism top view

Rack Clutch Mechanism :-

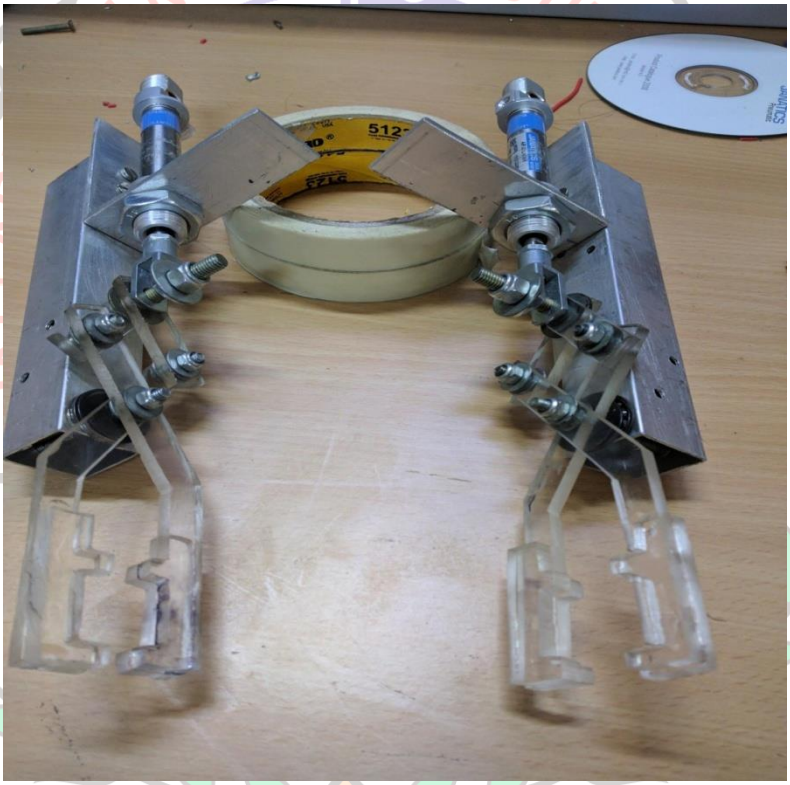
- First of all we have cut all links from acrylic plates according to dimensions.Then we made 4 mm and 6 mm diameter holes with appreciate dimensions of distances.
- We made one L clamp for mounting of pneumatic piston.

Problems faced:-

- Gripping mechanism was possible only if the rack was allowed to touch the game arena outside loading zone.
- Gripping of this mechanism was not strong enough to lift the 1kg rack.
- It was difficult to mount the two pneumatics of gripping mechanism, on slider, at a distance equal to the length of the rack.
- The weight of rack created an effective moment on slider, which could bend the slider at mounting point.

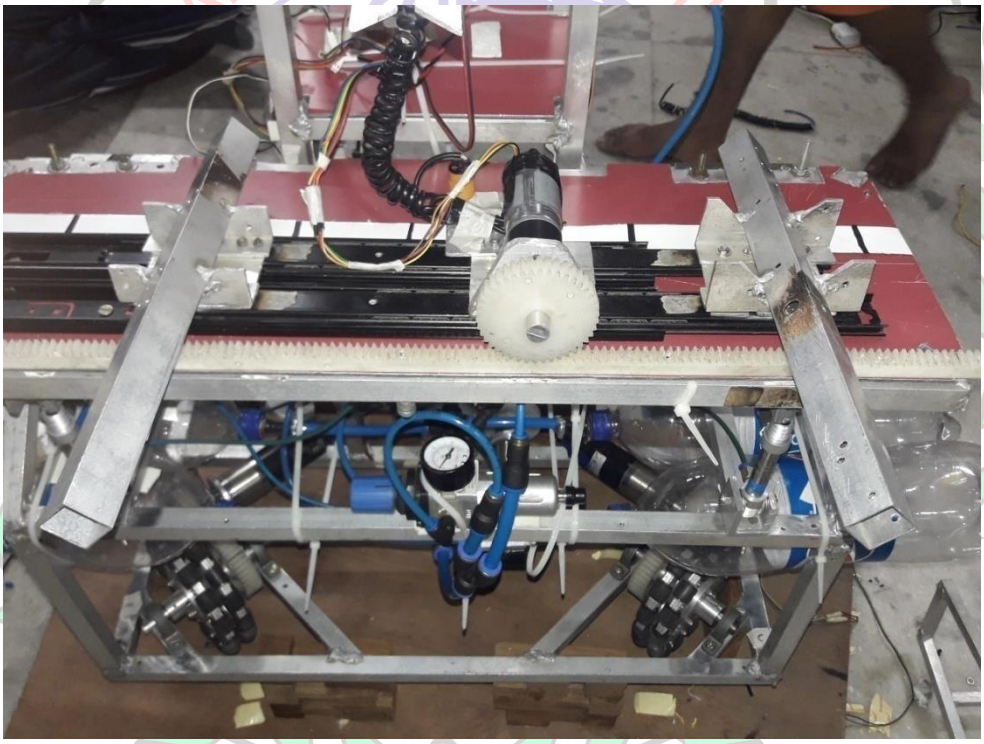
Solution :-

- Gripping mechanism was replaced by rack lifting mechanism.This mechanism solved the problem of rack touching the arena by lifting it to some height above the ground.
- It also solved the problem of moment by reducing the distance between rack and the manual bot.



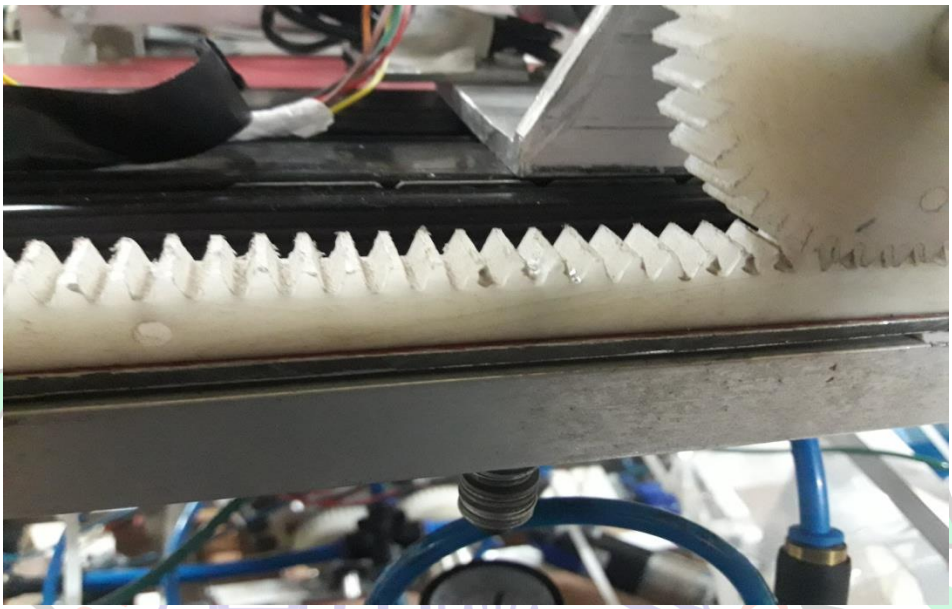
SLIDER MECHANISM:-

- In Slider, a rack and pinion mechanism is used .
- This is used to move the upper rack stand horizontally for positioning of shuttlecocks for loading by striker.
- Two sliders are used instead of one for more stability. Care is to be taken that both sliders should be parallel to each other.



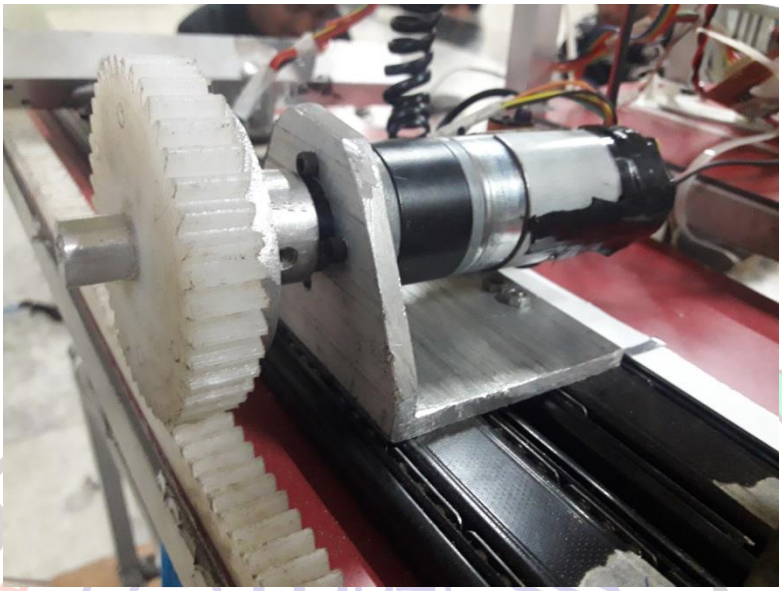
Rack and Pinion Mounting:-

- The rack is fixed and mounted on the chassis.
- The pinion is fixed to a planetary motor which is mounted on sliders by means of an L section.



Motor Mounting:-

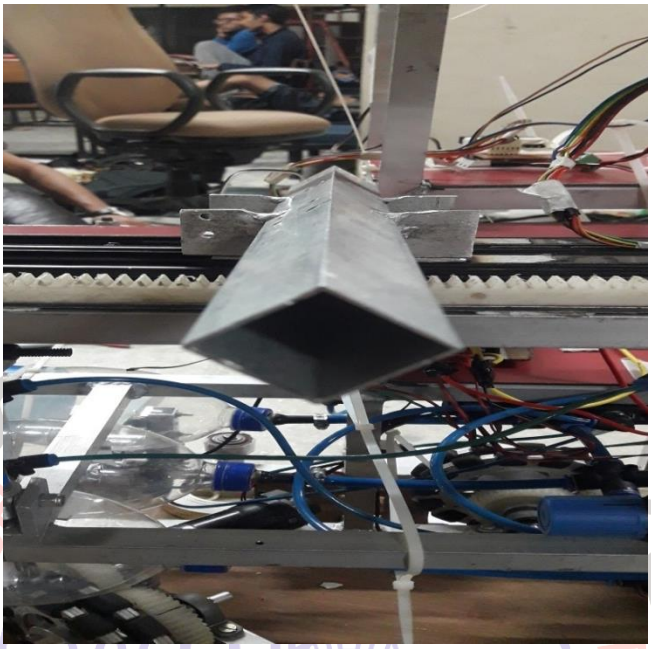
- While doing so, care should be taken regarding the height of motor fixed on slider for proper meshing of the rack and pinion.
- Accordingly, a hole is to be drilled on the L section. The L section used is 5 mm thick to avoid bending of L section and thus preventing any misalignment.



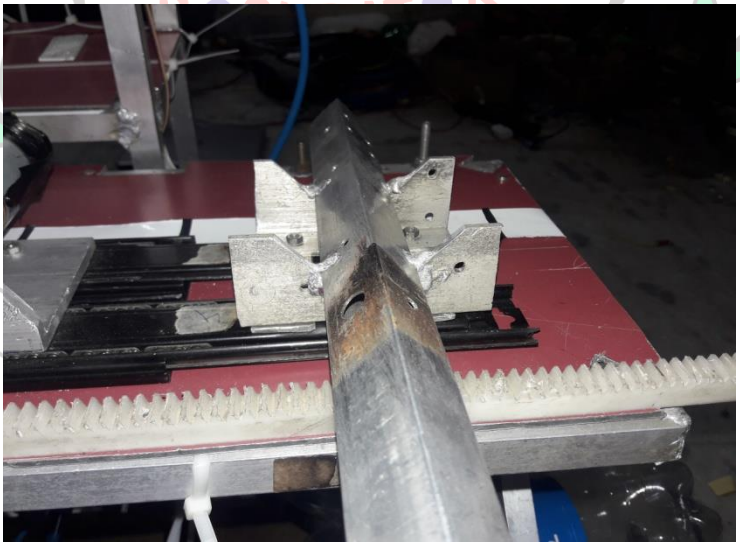
Upper Rack Stand Lifters Mounting:-

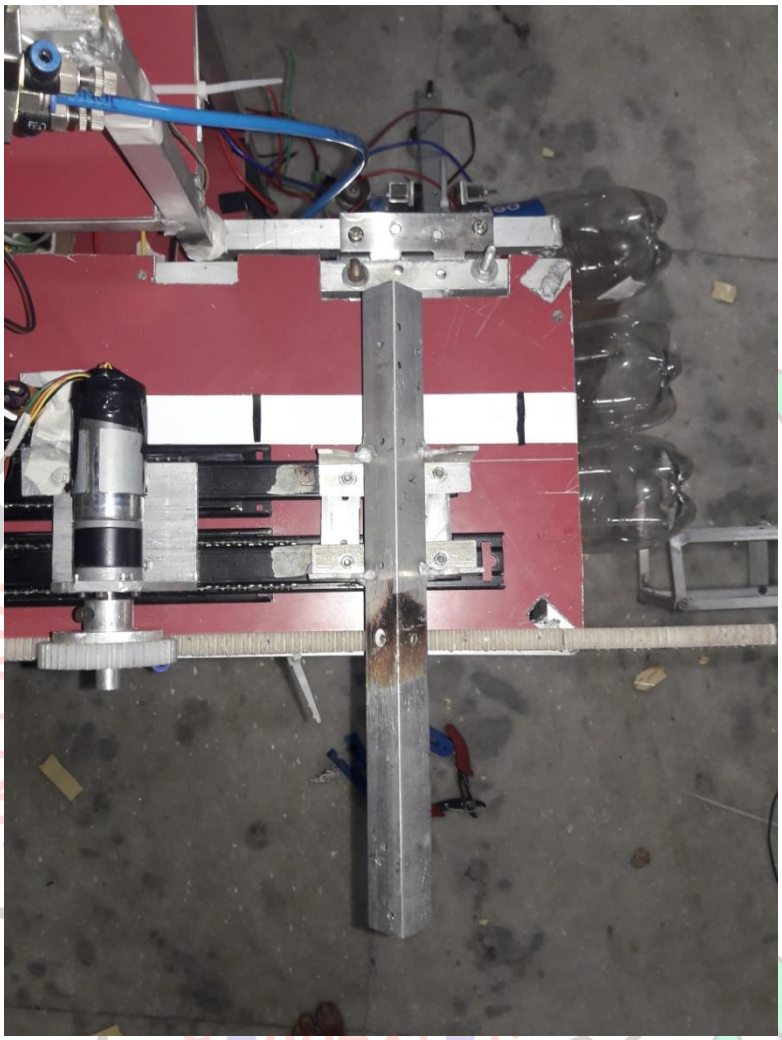
- Secondary 1”×1” aluminum box sections are used for lifting the upper rack stand.

- These were fixed on slider such that their faces made 45 degrees with the horizontal plane. This is done so that the upper rack stand can be quickly and easily lifted as somewhat horizontal misalignment will not matter. Due to its V shape the rack will autoalign.



- L sections with V grooves are fixed on each end of each slider. L sections used are 3mm thick as they are to be used for moderate loading. L sections are fixed on the two sliders with perfect alignment such that both links be parallel to each other and perpendicular to the slider. The links are then brazed to the L sections.





- Care is to be taken and sufficient clearance is to be kept between the upper rack stand lifting links and the rack. This is done to avoid any obstruction in the slider movement.

Slider Mounting:-

- The sliders were first mounted on composite and then the composite mounted on the chassis.
- The slider should be mounted such that the centre of its stroke length should match with the centre of the bot(chassis).

Problems faced and Solutions:-

- Stability of the motor fixed was very less.
- Solution : Thus two sliders are used to minimize the vibrations of the motor and increase its firmness. Thus increasing its stability.

- Not enough holes present on sliders.
- Solution : Extra holes had to be made on sliders with proper calculation of the spacings required.

- Sliders had just forward stroke, but both sided stroke is required as the sliders should be within the centre stroke length.
- Solution : For this some major modifications are to be made in the sliders.
- The end constraint of the slider is removed so that the slider stroke is increased and extended to both sides, forward and backward.
- The removal of end constraint leads to removal of the ball bearings during the return stroke. Thus new constraints need to be made for the ball bearing strip. This was done by cutting a small section in the slider and bending it such that it creates a barrier for the the ball bearing strip.

- The rear slider experienced vertical shifting due to loading of upper rack stand.
- **Solution:** This problem was eliminated by increasing the perpendicular distance between the sliders. This was done so that the rear slider would experience lesser torque.

RACK HOLDING MECHANISM:-

- It is the mechanism which holds the rack at a specific distance from the slider.
- It consists of a pneumatic,a slotted groove and a connecting link.
- The pneumatic drives the motion of connecting link forward and backward.
- Part of backward and forward motion of connecting link is converted into slight upward and downward motion.
- Pneumatic is actuated with the help of solenoid with appropriate exhaust valve opening.
- This is a pulling mechanism that pulls a link of the rack towards the manual bot.
- To constraint its pulling 2 L clamps are used on the 2 rack lifters.

UPPER RACK STAND :-

- An idea was thought of to save the time of rotation of the autonomous bot while loading and throwing. Shuttlecock loading was thought to be done with autonomous bot aligned just as while throwing.In this way, time consumed for two rotations would be saved.
- For this type of loading, the shape of grooves on the upper rack stand was thought to be modified. The grooves instead of being straight were made curved. Along with this the direction of striker was changed and made along the curve.In this way, the shuttlecock could be loaded when the autonomous bot was aligned as in throwing position.

Problem faced :-

- There were a few problems faced during loading with this method.
- The manual bot would have to move perpendicular to the wall to load the shuttlecock.
- This would instead consume more time in alignment of manual bot. Hence this idea was dropped.

UPPER RACK STAND LIFTING MECHANISM

- The upper rack stand is to be lifted through pneumatics. Three pneumatic cylinders were used for this purpose. Composite having slider is fixed on a rectangular platform made up of ¾× ¾ in Al box section.This platform is hinged on one edge and on other edge inverted c-section one for each cylinders are fixed. Cylinders are connected to C-sections through bolts. The pistons arþt5e connected to C-sections using fork end on piston and then inserting the bolts.

Problems faced

- Earlier only two cylinders were to be used. The weight of slider assembly was too much to carry by just two cylinder. Hence a third cylinder was added in the centre. This helped in smooth lifting of the upper rack stand

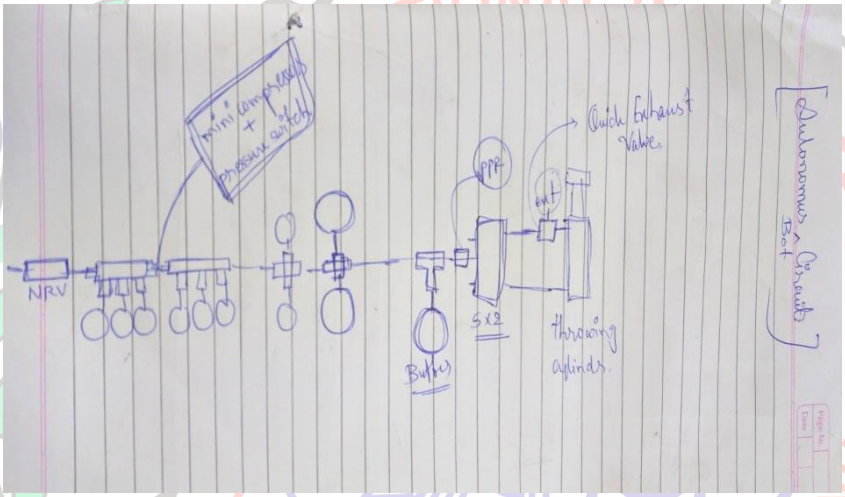
Pneumatics

Making of air tank :-

- To make air tank, 2 liter Coca-Cola bottles are used.
- 10 mm hole is drilled into bottle cap and rubber valve is pulled out from hole of bottle cap.
- Before that non return valve mechanism is removed from rubber valve.
- Then bottle neck is covered with PTFE tape and bottle cap is fixed on bottle neck.
- Fevibond is applied on bottle neck and between bottle cap and rubber valve.
- Wait for 40 minutes to dry and strength.

Autonomous Bot

- Pneumatic circuit diagram

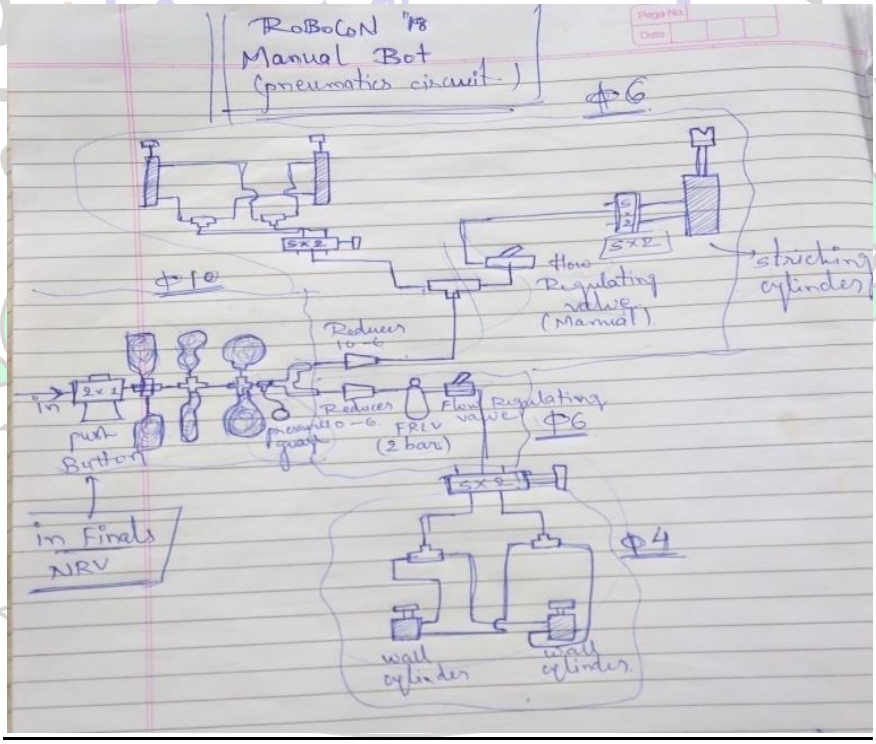


- The basic pneumatic circuit uses an air tank, valves, pressure regulator, pneumatic cylinder, and tubes.
- The circuit used consists of
- an Air Tank which is made from the Plastic bottles which are completely sealed to prevent leakage.
- A pressure regulator which maintains a constant pressure, which is the required throwing pressure.
- A 5x2 valve used to actuate the forward and backward stroke of the pneumatic cylinder.
- A quick exhaust valve used to get quick exhaust.

- NRV(Non Return Valve) used to pump air into air tank made of bottles.
- In Autonomous bot 10 bottles are used to make air tank.
- To pump air between match a mini compressor is used if pressure is not enough in air tank.
- Which is not used in final bot.

Manual Bot

- Pneumatic circuit diagram



JIGS

- When bot is in standstill condition the whole weight of bot would be acting on motor shafts. If shafts is not rotated for much time it would bend. so it is necessary to rotate wheel after 10-15 min.
- Jigs are solution for this problem. Jig is basically a platform on which bot is kept when not in use to prevent motor shaft bending.
- Jigs are made up of teakwoods and ply.
- on ply teakwood of 10 cm height is attached using fevicol. teakwood is attached only to support where load of bot is going to act i.e binary and ternary joint.
- Main problem came across was levelling. All teakwoods were of not same height. so we have to adjust height using tapes so that all joints touches bot.

ELECTRONICS

Omni Drive

Why Omni?

Omnidirectional wheels are wheels which can slip in the direction of motor axis and rotate perpendicular to it. The use of these wheels lets the bot move in any direction without turning i.e without changing its heading. Moreover, translational movement along any desired path can be combined with a rotation, so that the robot arrives to its destination at the correct angle. This is a quicker alternative to other systems like differential driving as the time required to turn is removed. Instead the bot just slides in the direction we want it to move.

For Robocon 2018, we used a 3 wheel Omni drive.
With a four wheeled system we can get between 50% (when translating along the axis of a wheel) to 71% (when translating 45º from an axis of a wheel) of your total force of your four motors. Therefore, you are harnessing between 2 to 2√2 (~2.83) motor's worth of force.

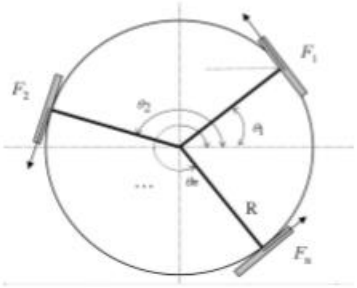
With a three wheeled system you get between 47% (when translating

along the axis of a wheel) to 68% (when translating 30º from an axis of a wheel) of your total force of your three motors. Therefore, you are harnessing between √2 (~1.41) to 2 motor's worth of force.
The four wheel drive also gives more stability to the base, however, still we decided to use a 3 wheel drive as

1. It is easier to manufacture as it is harder to keep all points in the same plane in a 4 wheel drive
2. Uses a motor less.

Omni Math

To calculate the RPM of each wheel, a mathematical model was used whose input was the desired velocity and angle.
Solving simple kinematics equation the following model was developed in terms of velocity in x direction and velocity in y direction vx, vy respectively.



The mathematical model consists of simple equations.

$$\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} -\sin\theta_1 & \cos\theta_1 & 1 \\ -\sin\theta_2 & \cos\theta_2 & 1 \\ \vdots & \vdots & \vdots \\ -\sin\theta_n & \cos\theta_n & 1 \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ R\omega \end{pmatrix}$$

This can be further reduced in terms of angle of displacement of the bot to the general equation
wheel_rpm = velocity * sin(wheel_angle-angle_of_displacement)

Here for each wheel rpm will be different according to wheel_angle being theta1, theta2….. and angle of displacement being the angle you want the bot to move.

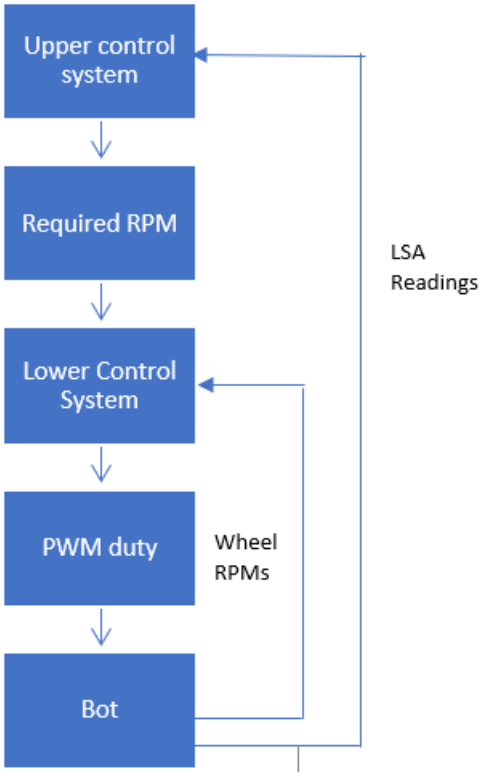
Problems faced:

Slipping of wheels:

The wheels were not able to generate enough traction at certain points with the floor due to high velocity of wheels and smooth floors. This made the robot spin as it started to slip and rotate. Omega control using LSAs (See Line Following) was used to correct this. This can cause problems while doing Odometry.

Required RPM not set:

Due to the effects of loading, motor characteristics RPM cannot be linearly mapped to PWM duty cycle hence we used a separate controller Arduino Due to maintain the required RPM for the wheels. The controller received feedback from the encoders as wheel RPM. A PID controller was applied to maintain the RPM. A separate controller was used for this purpose due to the high amount of interrupts from the three wheel encoders. As in an Omni drive perfect RPMs are necessary to drive the bot straight this was required.



This was the closed loop used in the bot.
Upper control system received data from the LSA08, this data was used to calculate the line correction and Omega correction (See line following) and the required RPM for each wheel was calculated.
This RPM was then sent to the lower control system which set the desired RPM by use of the feedback from the motors and PID controller.
The error was the difference in the required and the current RPM.

Actuation:-

We wanted to actuate the pneumatic for throwing and the lifting mechanism in autonomous and the rack-lifting mechanism, rack-clutching mechanism, and the striker mechanism in manual. For this we used solenoid valve as the actuators and the IC ULN2003 as a relay.

Actuating the solenoid valve:-

The solenoid valve has two wires coming out of it which we have to give the input voltage. The rated voltage for the solenoid valves we used is 24v below which they do not do not work properly.

Solenoid valve

The solenoid valve is used to control the movement of the piston it has only two states on and off .

By doing so we can control the pressure as when solenoid valve is on it will allow the air or pressure to release else it will stop it.

Problems faced

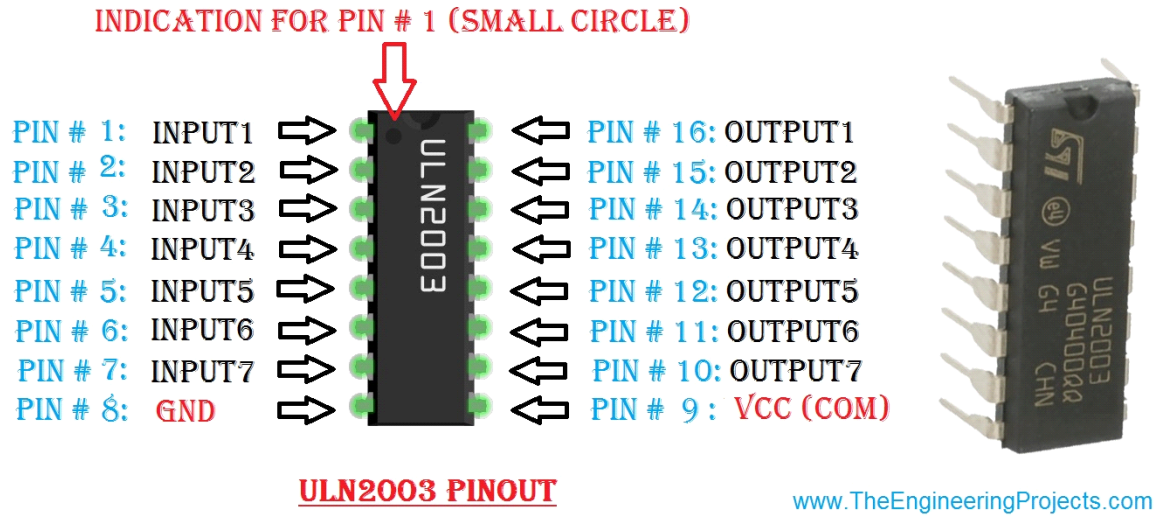
Due to being in continuously in ON state the solenoid valve gets heated up .

Solution

We connected two solenoid valve(double pilot) so that we can have four different states . Hence to actuate the piston first we make the Solenoid ON and after some delay we make it off but it remains at the same position. To close the piston we have to make other solenoid valve on and after some delay make it off too that is how double pilot works.

Working of IC ULN2003:-

The input is given from the micro-controller in our case it was the Arduino from any of its GPIO pins.



And the output signal is taken from the corresponding output pin of the IC ULN2003. Output signal is then send to the solenoid valve.

Problems Faced

- It gets heated up if we connect multiple number of solenoid valves to it .
- As we have connected solenoid valves which is an inductive load hence the back e.m.f.

Solution

We connected a Schottky diode in parallel to the load that is solenoid valve in our case. The anode of diode is connected to the output coming from ULN2003 and cathode to the 24 v supply positive terminal.

The IC we have used :-

- MCP4725
- ULN2003
- LD3v3
- LM 2596 (BUCK converter)

- LLC(logic to logic converter)

Connections:-

- For connecting the motor with the motor drivers we used 3mm diameter of multistrand wires as they need high current to flow throgh it .

- For connecting the signal pins from microcontroller to the motor driver we used 0.5 mm multistrand wires.

- For connection where 24v supply is needed we have used 3mm diameter wires .

- Power and signal for the LSA08 is given by the FRC cable through FRC connectors connected on both sides.
- We have made same power supply circuit to give power to all the devices on the robot.
- By connecting two 12v batteries in series we can have 24v and by taking the direct connection from the base battery we can have 12 v and giving that 12v to LM2596 buck converter we can have 5v .
- Initially we used to make screw connectors in power supply circuit as well as in the GCB for the micro- controllers but we found that it was very slow to remove and connect wires with screw connectors as we have to always unscrew first and then remove the wires and the screw it again .

- So we used Plugin connectors on both sides of a wire. It made the connection process faster and if in case of emergency we have to remove any wire so it is easy to remove too.

GCB :-

- GCB for the upper control of Autonomous robot

  It contains following components:-

1. 12v and signal given to the LSA08 through FRC connector.

2. 5v and signal given to the DAC IC MCP4725 through female connector 6 pin .

3. 24v and signal given to the PPR thorough 4 pin Tamiya connector.

4. 5v given to the microcontroller and the male headers connected for inserting the microcontroller to GCB.

5. Communication between the upper and lower control system through I2c protocol and USART protocol.

6. 5v given to the led for the camera detection.

7. 5v given to the led for the indication of throw.

8. 5v and communication signal given to the raspberry pi.

9. 5v and signal given to the two limit switches for the reloading mechanism.

10. Signals given to the motor drivers from Microcontroller.

11. 24v given to the motor driver of the reloading mechanism.24v given to the solenoid valve for throwing mechanism.

12. 24v given to the solenoid valve for reloading mechanism.

- GCB for the upper control of Autonomous robot

It contains following components:-

2.1. signal given to three motor drivers of the wheel motors.

2.2. 5v and signals from the encoders of the wheel motors.

2.3. communication between upper and lower control using I2c protocol and USART protocol.

2.5. LLC are been kept for converting the signal coming from the microcontroller which is of 3.3v logic to 5v logic.

2.6. microcontroller is given 5v.

2.7.supply of 3.3v using ld3v3 IC.

1.Task

The Bot had to traverse throughout the arenato reach different loading and throwing zones. The arena had predefined lines marked throughout which can be used as feedback.

2. Solution

Line following:

Line following was one of the most important part of the robot and had to be done accurately as well as fast so as to reach spots in minimum time.

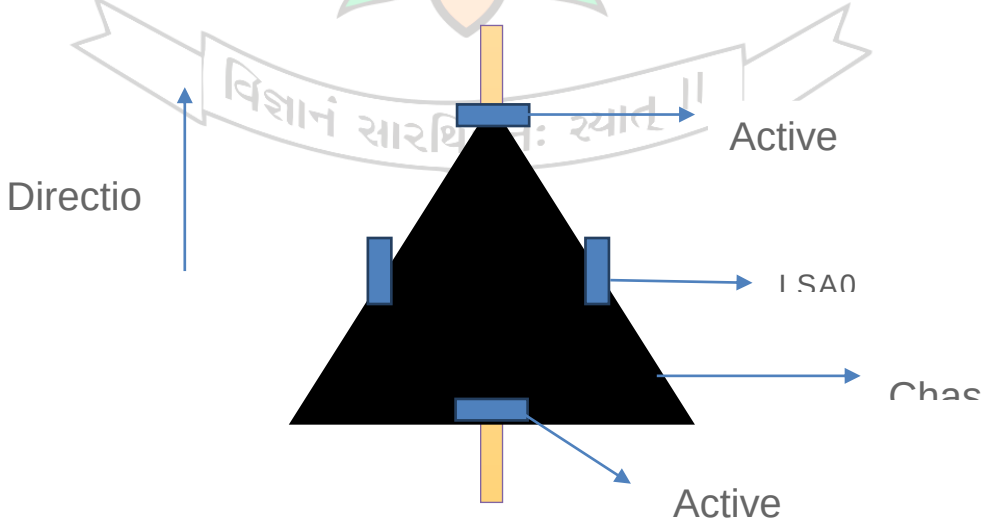


Figure 1

Line following was done using 4 LSA 08 sensors. The sensors were placed in an asymmetric plus shape, at all time the reading of the two line sensors on the line was taken. For perfect line following two things were required.

1. The Forward sensor (Active Line) be at the center of the line
2. The Backward sensor (Active Omega) be at the center of the line

This was achieved by varying two aspects of the bots' movement. As the bot was an omni drive, the translational velocity, the angle of movement and the speed of rotation of the bot could be altered.

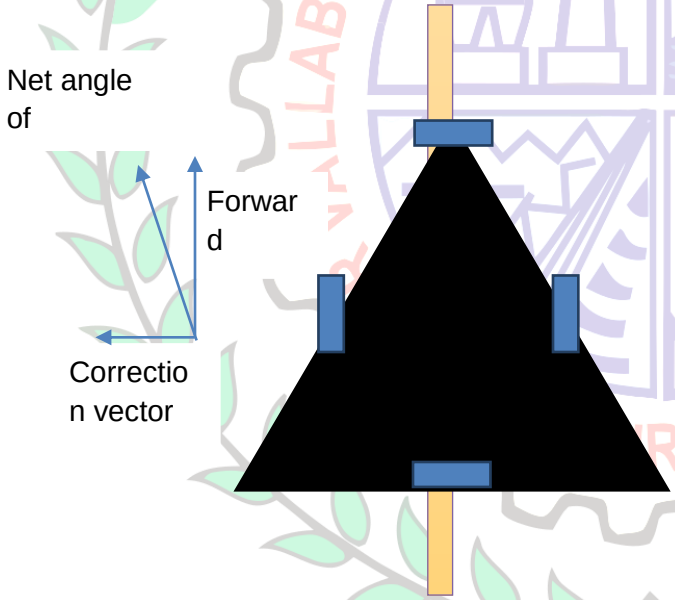


Figure 2

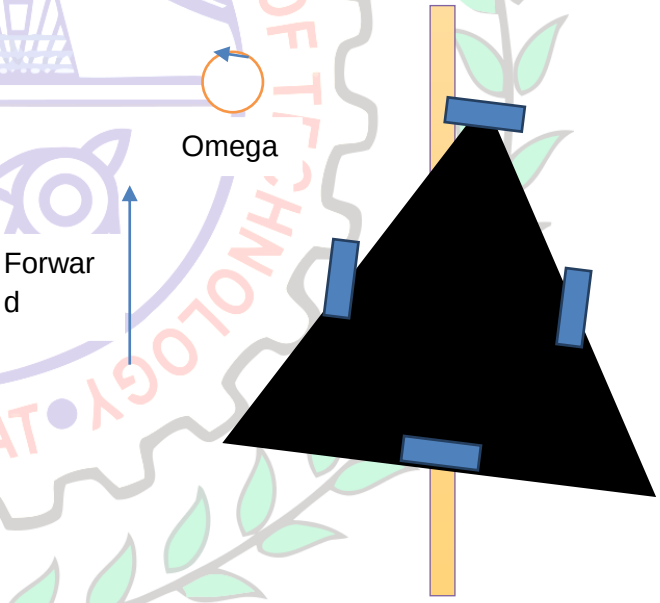


Figure 3

To keep the bot on the line, using the Active Line sensor we determined the angle at which the bot needed to be shifted. The forward sensor determined the shift considering the heading of the bot to be constant always. The resultant angle was calculated as the net vector of forward translational vector and correction vector. See figure 2

To keep the heading constant, the reading of the Active LineSensor and the Active Omega sensor should be the same as they are perpendicular to each other. The difference between the

readings of the sensor signified how much the bot was rotated off the given axis. See Figure 3. This difference was used to correct the heading of the bot.

Note:
Initially an IMU, (number) was mounted to calculate the heading of the bot and correct it but the readings were lagging due to intense calculations. The response of the line sensor was pretty fast and thus was chosen over it. A faster controller than the Arduino Mega would've worked for the IMU.
A simple magnetometer was tested as well and was showing decent results. There was no rigorous testing done as finally it was not required.
To rotate the bot at a certain angle, a simple gyro with noise filtering could've been used. Integrating gyro values would result in giving us the angle however use of a gyro is not suggested for long use due to drift accumulation unless drift elimination is done (Using Magneto and accelero).

Path Planning

The bot had to traverse a predefined constant environment with a few nodes therefore the path from each node to the next was hardcoded as arrays. A simple backtracking algorithm was also thought upon but wasn't used as would've made an easier problem more complex.
The array used was a 3-dimensional array, the first input was the current position, 2nd the final position and the 3rd was automatically updated by the bot and referred to which node the bot was at between the initial position and the final position

```
char arr[6][6][15]={
            {{s},{r,s},{r,f,s},{r,f,f,s},{r,r,f,f,s},{r,r,f,f,f,f,f,s}},
{l,s},{s},{r,s},{f,f,s},{r,f,f,s},{r,f,f,f,f,f,s}},
{{l,l,s},{l,s},{s},{l,f,f,s},{f,f,s},{f,f,f,f,f,s}},
{{b,b,l,s},{b,b,s},{b,b,r,s},{s},{b,b,r,f,f,s},{b,b,r,f,f,f,f,f,s}},
{{b,b,l,l,s},{b,b,l,s},{b,b,s},{b,b,l,f,f,s},{s},{f,f,f,s}},
{{b,b,b,b,b,l,l,s},{b,b,b,b,b,l,s},{b,b,b,b,b,s},{b,b,b,b,b,l,f,f,s},{b,b,b,s},{s}}

};
            };
```

To go from node 0 to node 1, the array index [0][1][currentnode] was called, current node was automatically handled by the bot by the number of junctions it crossed. l, r, f, b, sare enumerations and define the direction of the movement of the bot.
L – left

R – right
F – forward
B - backward
The last s signified that the bot had to stop at that position.
Whenever a new junction was passed by the Active line sensor, it incremented the current node. The value returned by the array was the direction the bot had to move.
E.g. Node 0 to Node 2. Current node = 0

1. Initial value returned by array with respect to [0][2][0] was r. The line sensors pertaining to r got active.
2. Bot moved right till next junction, current node incremented [0][2][1] returned f
3. Bot moved forward till next junction, current node incremented [0][2][2] returned s
4. The bot stopped as s.

Whenever the bot had to turn, sensors perpendicular to the bot were turned on. As soon as they got a reading that implied, the perpendicular sensor was on line, at this moment the Active and Omega sensors were switched to those which were perpendicular sensors before according to the direction of movement.

Rotation

To rotate the bot, we used feedback of LSA only. This caused many problems because as soon as the LSA left the line we had no other feedback thereby we had to reduce the speed of rotation a lot. A better option would have been the use of a gyro, we used LSA for feedback because we every time had to just rotate 90 degrees and not any arbitrary angle.
Even the rotation was PID tuned, Kd was way higher than Kp here because we had to stop the bot as soon as the perpendicular half reached the line, therefore as soon as the error changed a little a lot of opposite force was required.

Alignment

The bot was required to be aligned accurately at a plus, this was done by getting all line sensors on the center.
As all 4 sensors were perpendicular, aligning any 2 sensors would've resulted in proper alignment of the bot thus reading of 2 perpendicular LSAs were taken and an angle corresponding to the error was calculated using atan2() function and the bot was moved at that angle.

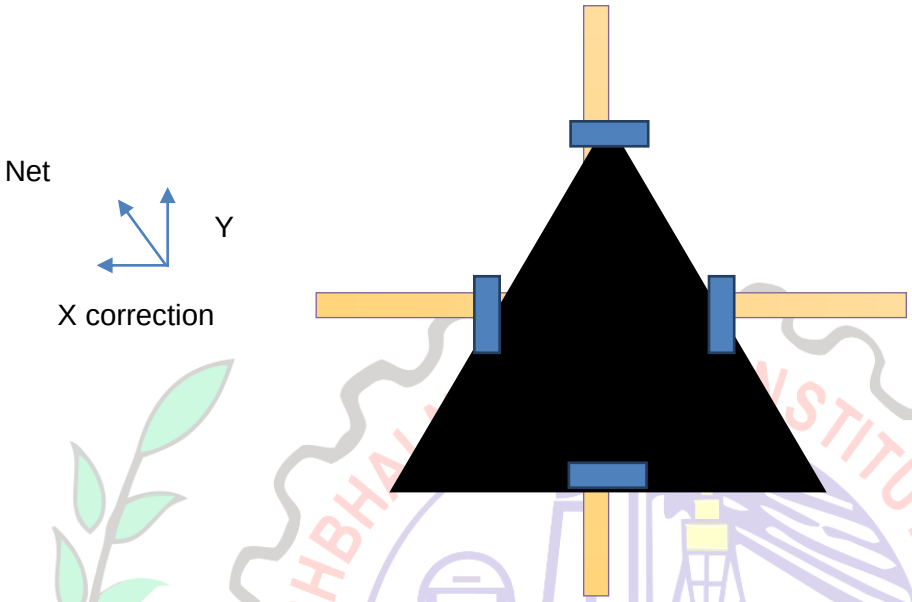


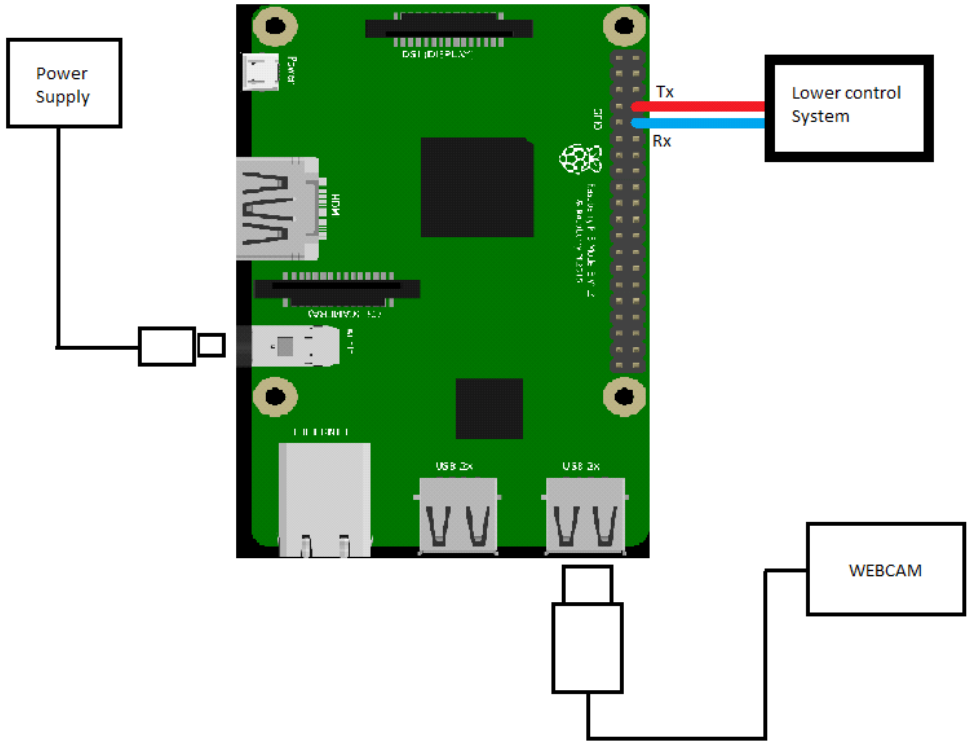
Figure 4

Line following is a pretty fast and accurate way of traversing the bot however the limitation arises that a line is not present everywhere therefore you have to always follow the line even when other shorter paths exist. Instead/Along line following other things such as Odometry, Wall following SHOULD be used.

1.Task

When the autonomous bot throws the shuttle cock from a TZ the shuttle cock can either go through the ring or it may miss it. To check that the shuttle cock has passed or not from the ring image processing is used as according to the problem statement of the competition there cannot be any kind of communication between the autonomous and manual bot and any of the team member. Depending on the result of this value, the bot will travel to another TZ or the same again.

CIRCUIT



2.Solutions

- **Object Tracking (NOT USED)**

  A webcam fitted on the autonomous bot is used to track the thrown object (Shuttle cock in this case) and also to detect the ring. The image of thrown ball has some pixel area. If the pixel area of the image of object is below a certain value and if its center is lying in the region inside the ring detected then it is an indication of the shuttle crossing the ring.

- Components

  1. Raspberry pi

  2. Logitech webcam

- Setup

  The camera is mounted on the top of the bot above the stack of bottles from where clear view of the ring in front is acquired.

- Implementation

- Before the competition begins the camera is calibrated to track the object. The calibration is done by taking each frame from the ongoing visual of the shuttle in front of the camera and adjusting the min, max values for HSV color space of the frame. This is done until we get a clear image in binary of the object only.

- The autonomous bot then starts and reaches TZ1.After reaching there it detects the ring in front of it by Hough Circles function of OpenCV used in the program.

- When the shuttle is thrown, the shuttle is tracked alongside the ring detected. The center of the object, its pixel area in the image and the region inside the ring is known to us.

Case 1: When pixel area of the object < certain value and center of the object is inside the region of ring

Then: object has crossed the ring.

Case 2: When pixel area of the object < certain value and center of the object is not inside the region of ring

Then: object has not crossed the ring.

- Algorithms used

  The C++ programming language and the OpenCV library is used for the code and the Visual Studio 2017 IDE was used for this purpose.

- Object Tracking

- The methods available :-

  - HSV color space
  - Background Subtraction
  - MIL tracker

- Methods not used and its reason :-

- Background Subtraction: This requires stable platform on which camera can rest. Even if there is slight motion of the camera while taking the reading then it would result into much disturbance. As the camera is fitted on the autonomous bot which has to move continuously, this idea cannot be used.

- MIL tracker: This is a built-in feature tracker provided in the “tracking.h” header. This tracker works on continuous evaluation of every frame captured for the region where the object is predicted to be. This predicted region is calculated from the data of the object that is to be detected fed in it prior to everything (sample image of the object) and using previous frame to search for the region in it matching the mathematical feature values of the sample provided. This method is based on Machine Learning which requires a lot of data set for training the algorithm to detect a particular object, which takes much time so this idea was not used.

- Method used :-

  - HSV color space: This method converts every frame captured (which is in RGB format) into HSV format. These HSV values are set by initially calibrating them using six parameters – H(min, max) , S(min, max), V(min, max). These parameters are manually set then the algorithm starts detecting the object. It also calculates the pixel area of object in a image and the coordinate position of its center.

- Ring Detection

- Methods Available :-

  - Hough Circle
  - Canny edge then contours

- Methods not used and its reason :-

  - Canny edge detection and contours : This requires that the image should be filtered much carefully which is difficult task, so this method wasn't used.

- Method used :-

  - Hough Circle Transform : This method is a built-in function provided by OpenCV.

    The image input is made to pass through a number of filtering process predefined and then its edges are detected. This function also returns the co-ordinates of the center of the ring and its radius.

- **Color Detection**

In this case we assign different colored shuttle cocks to various TZs viz. Red for TZ1, Blue for TZ2 and gold for TZ3.The color of the shuttle passed to the autonomous bot is recognized and the output is then given to the appropriate control system that drives the bot to the location.

- **Components**
  - Raspberry pi
  - Logitech webcam
- **Setup**

  The camera is mounted below the throwing link in such a way that its field of view consists only of the shuttle cock when it is passed to it from the manual bot.

- **Implementation**
  - When the ball is received in the autonomous bot the IR sensor placed just below it gives a signal to the raspberry pi for taking the picture.

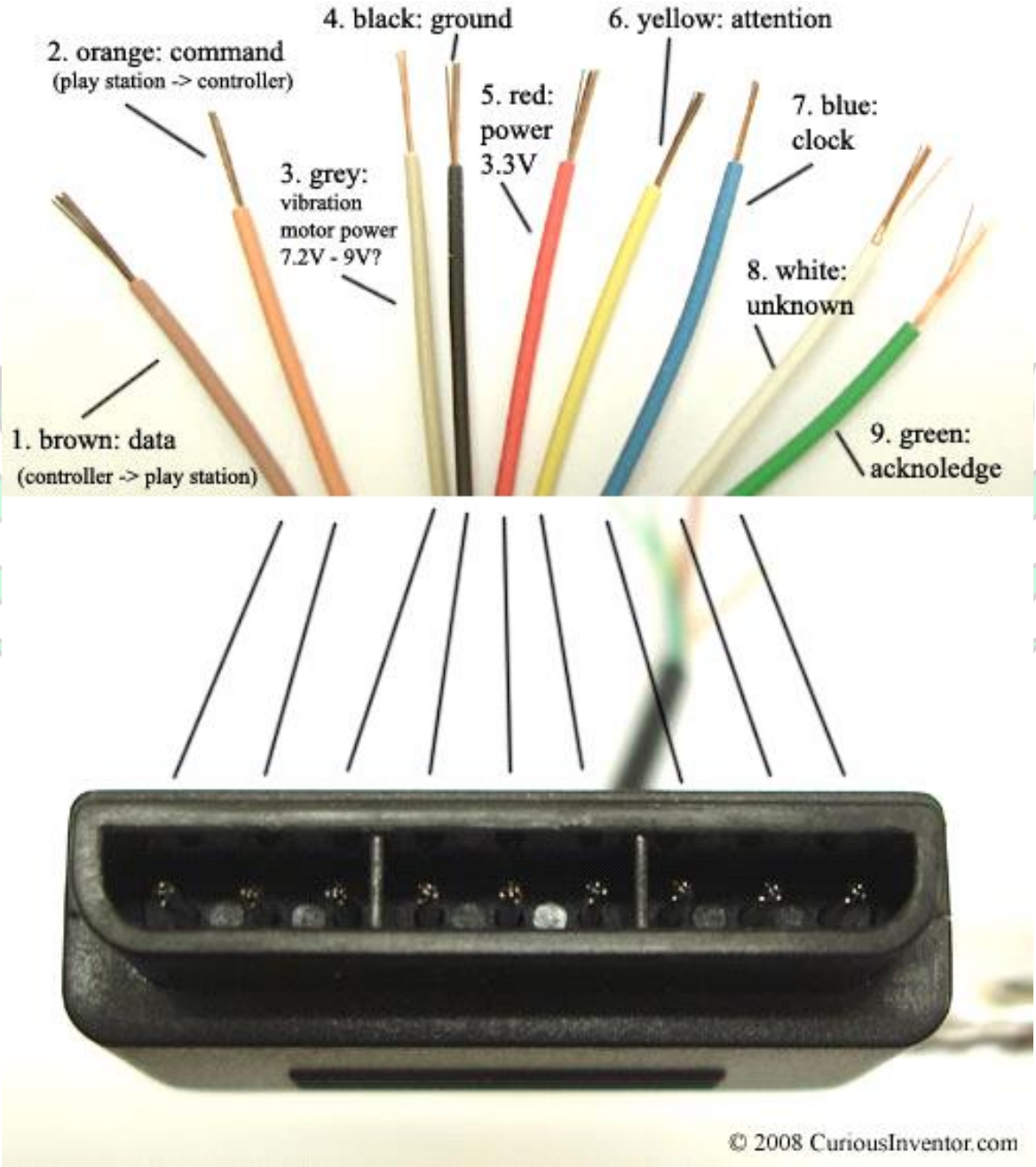
- Meanwhile raspberry pi has the code running in it. As soon as the raspberry pi receives the signal from IR sensor it takes 5 pictures of the shuttle cock, at an interval of 0.25 seconds, in front of the camera.

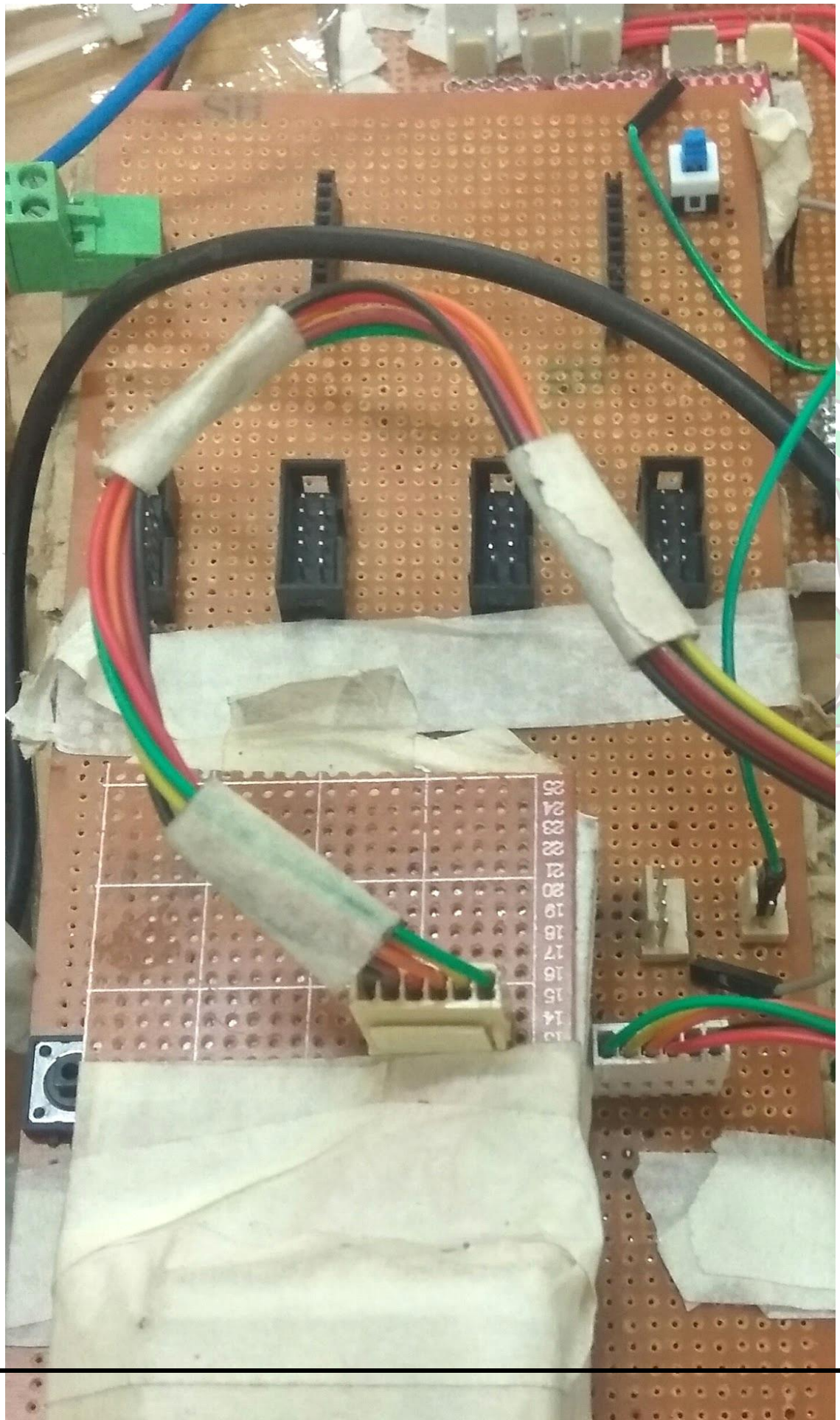
- Each image taken is then processed and filtered.

- For one such image area occupied by red pixels, blue pixels and gold pixels is calculated. This is repeated for all the 5 samples and in the end cumulative area of red, blue and gold pixels is obtained.

- The color with maximum area is then obtained and appropriate signal is sent to the control system for driving the bot to the required TZ.

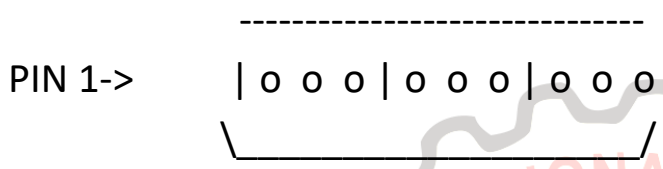
PS2 INTERFACING:





Hardware setup:

LOOKING AT THE PLUG



PIN # USAGE

1 DATA -MISO
2 COMMAND -MOSI
3 N/C (9 Volts unused). -Vibrationmotors
4 GND -Gnd pin
5 VCC -Vcc pin
6 ATT -Slave Select
7 CLOCK -SCK
8 N/C
9 ACK. - acknowledge

Brief description of pins:

1.)**MISO**- Data: Controller -> PlayStation This is an open collector output.

2.)**MOSI-**Command: PlayStation >Controller

3.) **Vibration motors power:** Power pin 6-9V

4.)**GND**

5.)**VCC-** 3.3-5V

6.) **Attention or Slave select:**This line must be pulled low before each group of bytes is sent / received, and then set high again .

7.)**Clock**: 500kHz, normally high on. The communication appears to be SPI bus. We've gotten it to work from less than 100kHz up through 500kHz.

8.)**White:** unknown

9.)**Green-** acknowledge:This normally high line drops low about 12us after each byte for half a clock cycle, but not after the last bit in a set.

How Bytes and Packets are Transferred:

The play station sends a byte at the same time as it receives one (full duplex) via serial communication.

The clock is held high until a byte is to be sent. It then drops low (active low) to start 8 cylces during which data is simultaneously sent and received. When the clock edge drops low, the values on the line start to change. When the clock goes from low to high, value are actually read. Bytes are transferred LSB (least significant bit) first, so the bits on the left (earlier in time) are less significant.

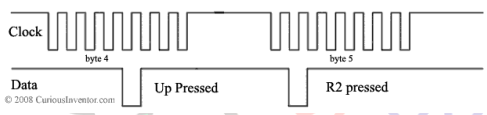
Digital Button state mapping:

Digital Button State Mapping (which bits of bytes 4 & 5 goes to which button):

Button		**Byte.bit**
Select.	-.	4.0
L3.	-.	4.1
R3.	-.	4.2
Start.	-.	4.3
Up.	-.	4.4
Right.	-.	4.5

Down. -. 4.6
Left. -. 4.7
L2. -. 5.0
R2. -. 5.1
L1. -. 5.2
R1. -. 5.3
Right analog(X) -. 6
Right analog(Y). -. 7
Left analog (X). -. 8
Left analog (Y). -. 9

For example:



Commands:

--0x41: Find out what buttons are included in poll responses.
--0x42: Main polling command
--0x43: Enter / Exit Config Mode, also poll all button states, joysticks and pressures
--0x44: Switch modes between digital and analog
--0x45: Get more status info
--0x4D: Map bytes in the 0x42 command to actuate the vibration motors
--0x47: Read an unknown constant value from controller
--0x4C: Read an unknown constant value from controller

--0x4D: Map bytes in the 0x42 command to actuate the vibration motors

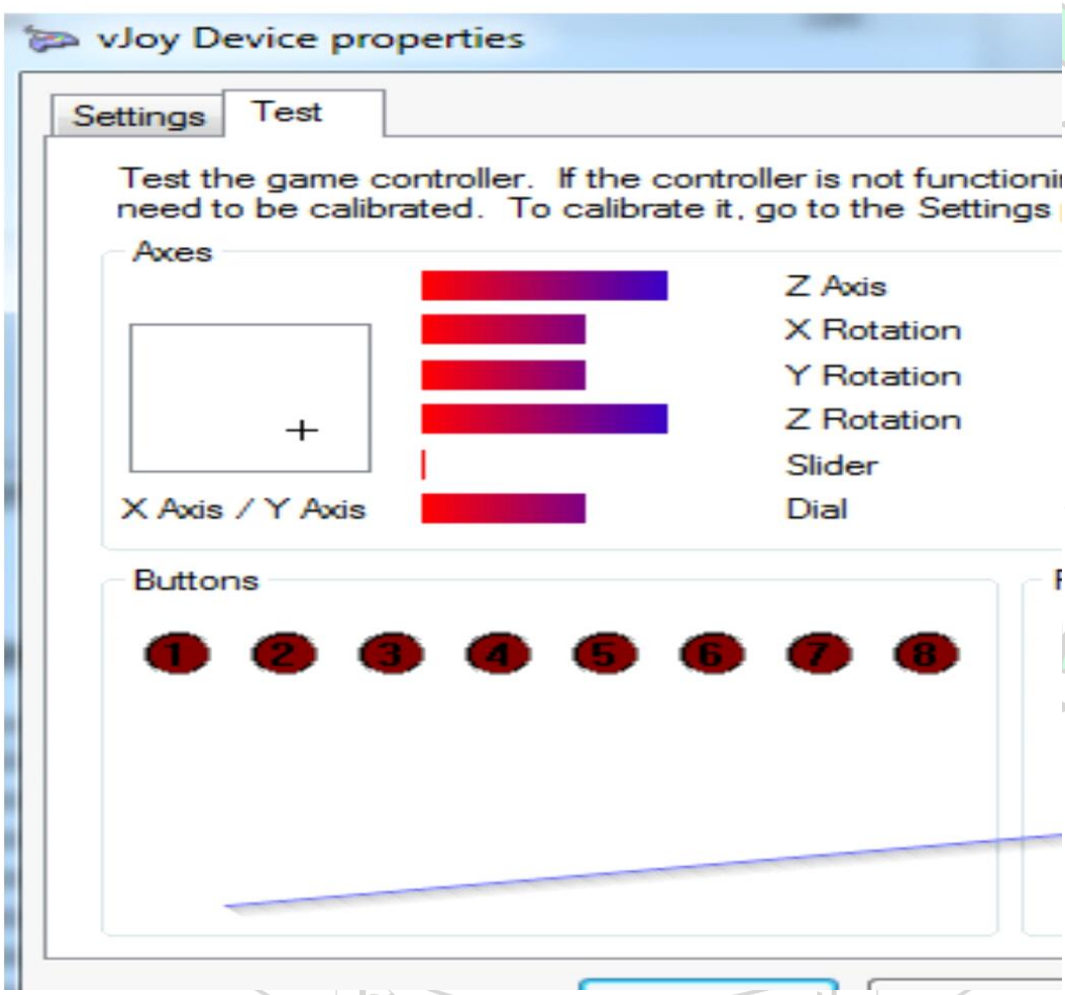
--0x4F: Add or remove analog response bytes from the main polling command (0x42)

Byte Sequence to Configure Controller for Analog Mode + Button Pressure

---- 0x42 Poll once just for fun
---- 0x43 Go into configuration mode
---- 0x44 Turn on analog mode
---- 0x4F Return pressure values
---- 0x43 Exit config mode

Problems faced and their solutions

I refer to the 'Point of attached picture):



The analog joystick in the PS2 microcontroller has a region of square which denote it's X and Y axis as shown in the image above.

But as the joystick in game controller has outer region as circle so the value at perimeter of the circle doesn't remain same.

The value at the diagonal points is √2 times that at any point on periphery of circle as internally it is a square.

But our requirement was to have the same value at all points of periphery of the circle.

Solution- "MAPPING OF SQUARE INTO CIRCLE"

- Solution was the mapping of the

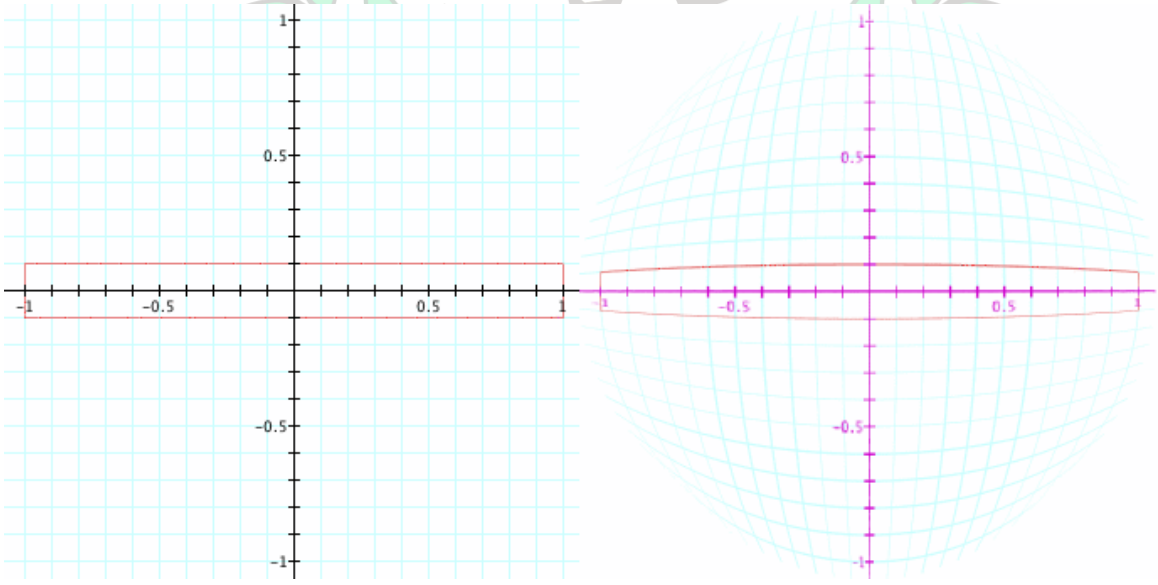
square values into a **circular region .**

We referred on the internet and found one Mathematical formula for transformation of **SQUARE <-> CIRCLE.**

$$u = x\sqrt{(x^2 + y^2 - x^2y^2)} / \sqrt{(x^2 + y^2)}$$

$$v = y\sqrt{(x^2 + y^2 - x^2y^2)} / \sqrt{(x^2 + y^2)}$$

where (u,v) are circular disc coordinates and (x,y) are square coordinates.



But stil we had problems in getting uniform values of joystick at periphery of the circle.
We searched online libraries for ps2 controller.

We got a online library named **PS2X LIBRARY** for arduino which included **button debouncing** in it.So finally we used this library.

References-

Ps2x library<->https://github.com/madsci101HYPERLINK "https://github.com/madsci1016/Arduino-PS2X"6/Arduino-PS2X

Square<->Circle mapping:-
https://stackoverflow.com/questions/1621831/how-can-i-convert-coordinates-on-a-squHYPERLINK "https://stackoverflow.com/questions/1621831/how-can-i-convert-coordinates-on-a-square-to-coordinates-on-a-circle"are-to-coordinates-on-a-circle

PS2 INTERFACING:- **http://store.curiousinventor.com/guides/PS2/**

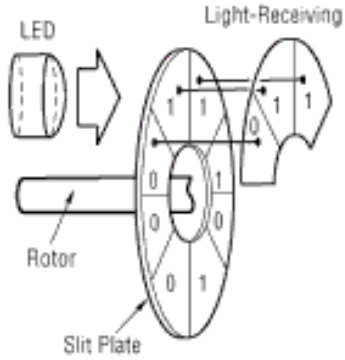
MANUAL BOT

LINEAR ENCODING:

1.) IR SENSORS:

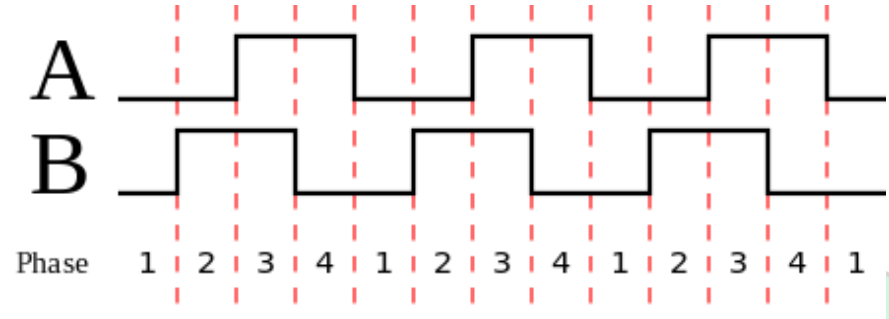
IR sensors work on the principle of emitting IR light and receiving the reflected light from the object it is pointing to.Whenever it detects the object the output of the IR goes **LOW** else it is **HIGH.**

2.)ENCODERS (WITHIN MOTOR):



Sensor Output		Angle
011	➡	45°
001	➡	90°
000	➡	135°
100	➡	180°
010	➡	225°
101	➡	270°
110	➡	315°
111	➡	360°

PHASE A & B OUTPUTS:



A rotary encoder, also called a shaft encoder, is an electro-mechanical device that converts the angular position or motion of a shaft or axle to an analog or digital signal.
We are using a white sheet on which there are black strips on it at a definite distance. Whenever the IR senor on linear encoder motor which moves the rack comes on **white** it gives **0** and on black gives 1.

We use these interrupts from **IR** sensors to actuate the rack movement to move the rack to a desired position set by the controller.On each **BLACK-WHITE** junction it's current position gets updated and reaching to the desired position it stops.

Why Encoder???

We are using the encoders to get rid of the overshoot at the desired position.At each junction we check and set the direction of the motor according to it's position .

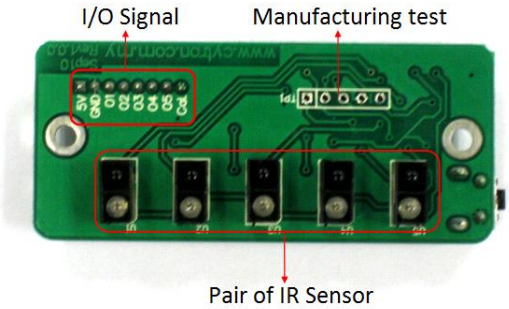
Now at the desired position due to inertia of MOTOR it gets overshoot and moves ahead so through the encoder we check the direction of the motor and update the direction on getting overshoot at the desired position and reverse it's direction at the desired position.

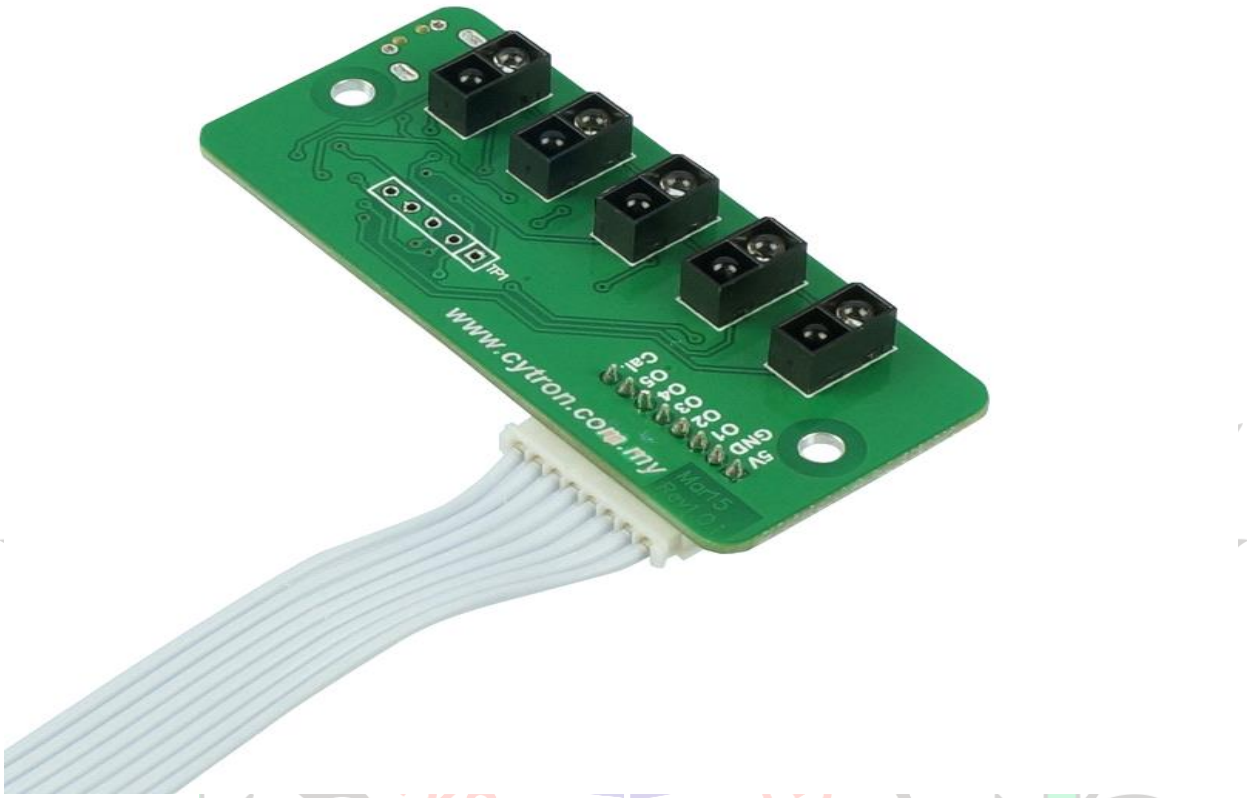
Line Alignment using LSS05:

We tried using the LSS05 sensor to align the bot on the line to stop the Manual bot on junction for passing the shuttlecock using the PID CONTROL on it.

In this aligning of the bot , we assigned a button on PS2 controller. Pressing the button on PS2 will make the bot enter into Semi-————————Autonomous Mode and will align itself through the error set in the PID Control.

LSS05 LINE SENSOR





LSS 05 LINE SENSOR

<u>PROBLEMS FACED:-</u>

Aligning the Manual bot using line sensor on the junction was taking much time though there was PID control due to the inertia of the BOT and due to it's high Speed .Though we made Kp as low as possible but the bot took atleast 2 oscillations due to it's inertia to align itself. So , we decided to align the bot Manually.

STM32

STM32 is a 32 bit microcontroller based on ARM Cortex-M processor manufactured by ST Microelectronics. It offers new degrees of freedom for the MCU users.

SOFTWARES REQUIRED & INSTALLATION

1. *STM32CubeMX*: STM32CubeMX is part of STMicroelectronics STMCube original initiative
to make developers' lives easier by reducing development effort, time and cost. STM32Cube covers the whole STM32 portfolio.
STM32CubeMX is a graphical software configuration tool that allows the generation of C initialization code using graphical wizards. We will be selecting the GPIO, Timers, etc. what we are wanting to use in the GUI. The code can then be generated accordingly. One can download the software following the link given below:

http://www.st.com/en/development-tools/stm32cubemx.html#getsoftware-scroll The installation is normal and simple.

2. *Keil uVision 5*: The µVision IDE combines project management, run-time environment, build facilities, source code editing, and program debugging in a single powerful environment. We will be using it just after we are done with the STM32CubeMX. The main logic will be there in this IDE. One can download the software following the link given below:

https://www.keil.com/demo/eval/arm.htm

Fill up details and download. The installation is normal.

3.*TeraTerm*: **Tera Term** is an open-source, free, software implemented, terminal emulator (communications) program. It supports telnet, SSH 1 & 2 and serial port connections. It also has a built-in macro scripting language and a few other useful plugins. This software we used was

basically like X-CTU. Reason we used this instead of X-CTU is STM32 Serial Communication doesn't work with the latter.

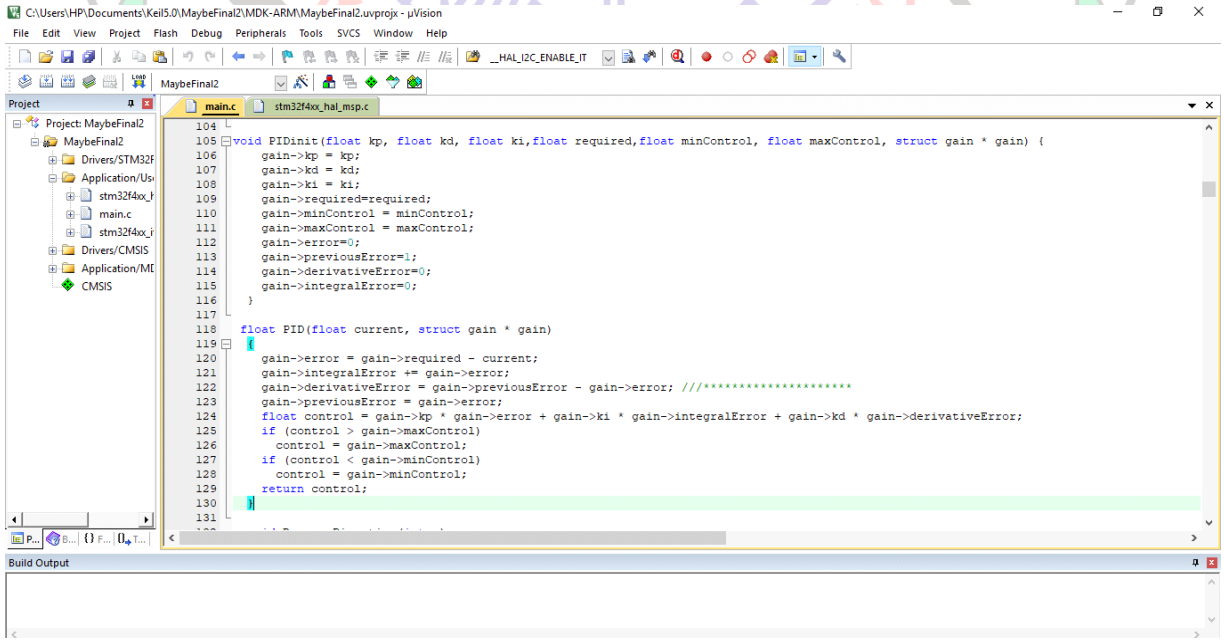
http://download.cnet.com/Tera-Term/3000-2094_4-75766675.html

SOFTWARE + HARDWARE IMPLEMENTATION of SYSTEM

What we planned? The upper control system will calculate the velocity and angle of the Bot. Hence, accordingly Rpm of three wheels is known from Omni Drive equation. These RPM will be sent to lower control with either TWI or USART. Now, this lower control is getting ticks from encoder of respective wheels. We can convert ticks into RPM of Wheel by the following equation:

```
Rpm=(Count- previousCount)*60.0)/(Time*GearRatio* Ppr);
previousCount= Count;
```

Hence, we know the current RPM of the wheels. Again, we know the required RPM of the wheels. So, we know the error. Our PID System was:



After calculating the PID Output, as what output we were supposed to give. We will write analogically at the signal pins of the three motor drivers.

PPR

The proportional pressure regulator is a device used for controlled pressure output. The device takes voltage between 0 and 5V and gives pressure between 0 bar and max pressure as output. It linearly maps the voltage input and gives the output accordingly.

We used the device for throwing mechanism. For three different throwing zones, the throwing link requires three different value of pressure for successful completion of the task.

Problems we faced:

The DAC voltage from microcontroller wasn't that constant. Due to this, the pressure value changes dynamically within some small error.

Solution:

We used a DAC IC for the purpose.

DAC IC

Digital to Analog Converter Integrated Chip is a device which gave constant DC voltage. The device's communication protocol is an inter-integrated circuit (I2C). The master (MCU) pings three different value (three throwing zones) to the slave (DAC IC) when the autonomous bot is in different zones. And the output is the voltage of those values (after digital to analog conversion).

List of microcontrollers

1. **Autonomous Bot**:

   a) *Upper control:* **Arduino Mega**

The upper control is handling all the LSA08s. It is calculating the RPM of the wheels from the feedback of the LSAs and the current position of the bot. The Control systems of the various state of the bot like align, rotate was handled by the Arduino mega. Communication of it with the Lower control.

b) *Lower control*: **Arduino Due**

The lower control was getting RPM from the upper control and this was just the required for the system. We got to know the real current RPM using the encoders. Three feedbacks from encoders gave us three RPMs. So, we have the errors of respective wheels. This is where we applied PID.

2. Manual Bot:

a) *ThreeWheelDrive* : **Arduino Mega**

PS2 was connected to it via SPI communication protocol. RPM was calculated from the PS2, mathematically and makes the bot move. The other buttons were sent to the Actuation control using USART.

b) Actuation Control: Arduino Uno

It receives which button is pressed from PS2 and accordingly it actuates the corresponding pneumatic cylinder. The actuations like Lifting Rack, Holding Rack, Passing shuttlecock were handled by Arduino uno.

Problems of communication between upper and lower

Initially, we were communicating with USART. In the system, upper was sending 9 different characters to the lower control. The problem of this was sometimes the data transfer wasn't complete before new data was ready to be sent. It created problems on the bot.

So, we started communicating with I2C. It too created many problems. If ACK is not sent somehow to the upper control, the bot didn't move or move in a wrong direction. Sometimes the TWI communication failed.

To overcome the above problem, we used a library named "WSWire.h". This library reset the I2C if communication is stopped. This was working good enough. But, the same I2C was engaged in many things.

So, we used another library “Easy Transfer” which could send whole structure instead of individual 9 characters using USART. So, finally our bot used both of them as communication.