Anorak Team Description Paper

Hamza Munir, Aamir Sheikh

Anorak Artificial Intelligence and Controls,
Center for Advanced Studies in Engineering,
Islamabad, Pakistan
saud.hamza.munir@gmail.com
anorakrobotics@gmail.com
www.athode.com/anorak

Abstract. This paper describes the hardware and software that will be employed by the Anorak team for RoboCup 2016, Small Size League. We have maintained our focus on implementing a modular AI approach with agent awareness and spontaneous real-time decision algorithms. In hardware, we've made significant modifications on our last year's design, providing better performance than before. Our efforts in low-cost design are to enable teams to operate on lower budgets, thus making it possible for more people with valuable ideas in AI to participate.

1 Introduction

Anorak was formed in 2014 with the aim of developing connected AI systems. We are working on making systems such that individual members of a team are "aware" of the circumstances the team is working under collectively. This includes acknowledging the deficiencies of underperforming members (due to technical faults and such), and taking appropriate measures to reduce the gaps in team performance caused by them. This involves modifying strategy and/or reprioritizing team objectives.

Parallel to our research objective in artificial intelligence, Anorak builds robotic machines that are low cost, reliable and easily serviceable.

The 2016 RoboCup will feature our new robot design named "Athlete".

2 Team Targets

Our participation in the 2015 RoboCup was cut short because of travel delays that were beyond our control. This year will hopefully present no such hurdles and we are looking forward to an engaging contest in Leipzig this summer.

In terms of preparedness, our robots are in a much better position to perform at an acceptable level in matches. We've made important structural changes to overcome issues in the robots intended for RoboCup 2015. Although the ball actuation systems need additional work before they can be installed in the new robots, the lab tests have shown positive performance. We are confident that we will have the dribbler, kicker and chipper installed in the new design by March.

Our targets this year for the software performance are the same that were set out last year. This is because we have not yet had the chance to play against other teams and analyse performance in a real match. The section detailing the targets from last year is given below:

"To evaluate software performance, we will be logging game data of our matches on our server and evaluating success rates of each component of our software. This includes targets such as having collision rates below 15% in all instances where collisions were possible, shoot on target rate of 67% weighted against hardware shortcomings and having no more than 30% of zone allocation errors. Zone allocation is explained in the software section later on."

3 Hardware

3.1 Mechanical Design

The current summary of mechanical performance is given in the table below:

Table 1. Mechanical Information

Max Mass	1.8 kg
Dimensions	Dia: 178 cm; Height 145cm
Centre of Gravity	Through central axis, 75mm above ground
Max velocity	3.5 m/s
Max Angular velocity	8 rps
Acceleration	4 m/s^2
Ball coverage	18%

We've reduced our robot weight from 2.3kg to 1.8kg. The weight reduction is due to the revised design and also a switch from Nickel based batteries to LiPo. The kicker solenoid has also been reduced in size and the capacitor bank now consists of more compact capacitor elements than before.

3.1.1 Low Cost Strategies

Continuing with our experimentation with acrylic design, we have managed to improve performance while using the same materials and fabrication processes. Compared with metals and injection moulded plastics, the acrylic fabrication methods using laser cutting techniques takes less time and keeps costs low. This has allowed us to quickly test design iterations and is helpful in getting a larger number of design ideas experimentally evaluated.

The table below from our previous TDP outlines the cost comparison.

Table 2. Cost Comparison for Materials

	6065 Aluminium	**Cast Acrylic Sheet**
Density	2.72g/cm^3	1.18g/cm^3
Price per sq ft, 0.6mm thickness	$3	$3
Fabrication cost per robot*	$250 - $300	$14 - $25

**specific to the team's country of origin*

The weight savings gained by using acrylic allows us to raise weight limits in components elsewhere, such as in the use of larger brushed DC motors. This reduces motor costs by over 24 times leading to massive cuts in per unit cost.

Table 3. Motor Comparison

	Maxon EC45 Flat	**Johnson 550**
Cost per unit	$85.79	$3.50
Power	30 W	24 W
Torque (Peak Eff)	55 mNm	62.4 mNm
No load rpm	4370	14500
Weight	-	218 g

We have had a positive experience with using the Johnson 550 Motors. The use of brushed motors is by no means even comparable to Maxon's precision design and reliability, but provides an adequate cost effective alternative. The motors do use more battery resources and generate noise. However, the major drawback of these motors is the precious volume they take up in the already compact design.

3.1.2 Mechanical Components and Descriptions

Wheels: The wheels underwent a thorough redesign because of a number of issues with the previous version. The O-rings on the rollers were susceptible to premature wear and developed cracks. This was observed on reserve wheels as well. Secondly, the rims of the rollers would easily distort under acceleration causing the O-rings to slip out. At speed, this would cause severe imbalance to the robot and cause damage to the gears and motor shaft due to vibrations.

The new wheels feature acrylic rollers and a sturdier design. With a larger diameter of 60mm and 16 rollers, the wheel provides higher top speed, better traction and allows us to put through more torque because a larger driven gear can be accommodated.

Images of the previous design and the new design are show below.

Image 1: Previous Omni-Wheel

Image 2: Redesigned Wheel

Power Transfer: After experimentation, we have a reliable perpendicular gear mechanism to transfer power from the vertically mounted motors to the horizontally mounted wheels. With a gear ratio of 1:5, the mechanism provides sufficient torque and is able to handle side impacts during the game. The gears need to be replaced after each full match. A process which takes no more than 5 minutes due to the special design of the assembly.

Structure and Mounts: We have improved the structural integrity of our robots by adding a support ring that connects all four wheel and motor mounts along the circumference of the robot. The previous design had the tendency to deform if excessive centrifugal force was experienced during manoeuvres or due to sudden impact with other robots and external objects. Even in the absence of such circumstances, the stress from the wheel vibrations would damage mount fasteners on the baseplate. The support ring solves these issues by limiting the degree to which individual wheel units can vibrate. This reduces the unsynchronized vibrations which would cause damage to the structure. The image below shows the support ring.

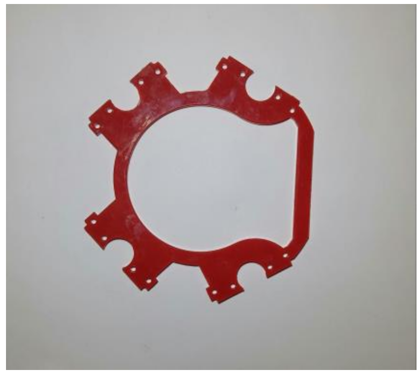

Image 3: Support Ring

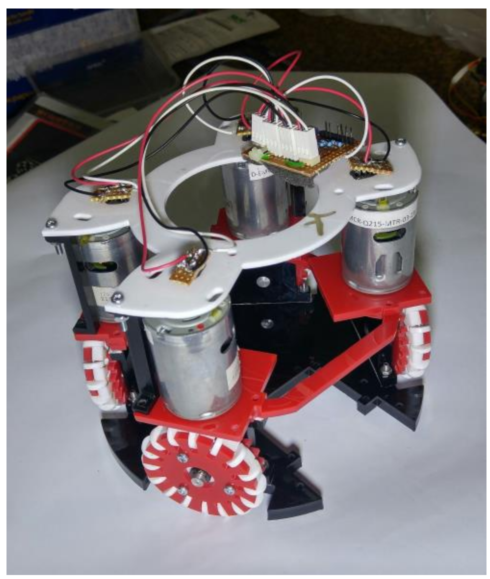

Image 4: Support Ring Mounted onto Athlete chassis

3.2 Electronic Components and Description

Most of the electronic components are same as last year. The summary of the electronic components from the last year's TDP is repeated below.

Main Processing Unit: For the main processing unit, each robot has an Arduino Mega 2560 R3 board. The board provides 54 digial I/O pins, 16 analog inputs and 4 hardware serial ports. The processor has a clock speed of 16 MHz.

Communication Module: We have opted for a 2.4GHz radio transceiver which uses the nRF24L01 IC from Nordic Semiconductors. The choice was made because of the quick switching speed between transmission and receiving modes, allowing us to set up a two-way communication system with the robots. The chip also offers better transfer rates and extremely low errors. As compared to other modules, the low cost of the nRF24L01 chip makes it easier for us to keep costs low.

Motor Drivers: Each motor is driven by a L298N dual full-bridge driver by STMicroelectronics. The four ICs are mounted together with a cooling fan. This allows us to operate the motors at higher currents for longer periods of time. Each motor driver receives PWM signals from the main CPU. The CPU manages feedback control via a PID implementation.

Wheel Encoders: Each wheel axle has a slotted spinner. A mounted slot sensor detects the slots and sends input signals to the CPU via a signal amplifier. The count signals are used to determine the rpm of the wheels and is used for PID control of the motors. We are in the process of designing more accurate encoders using a hall sensor IC.

Kicking Circuit: The kicking circuit uses a capacitor bank of 4400μF which is charged using a booster circuit. The CPU controls the discharge trigger through a transistor circuit. Charging takes around 2 to 3 seconds depending on configuration.

Dribbler: The dribbling motor is controlled by a L298N chip which is fed PWM signals from the CPU.

Power Supply: A 3 Cell 3200mAh Lithium Polymer battery bank supplies the motor circuits and the booster circuit for the kicker. A separate power source with two 9V batteries is used to power the main CPU, slot sensors, radio module and IR sensor for the ball. The main battery gives around 30 minutes of play time per charge.

Note: For the 2015 RoboCup, we had settled for Nickel based batteries because of availability. We have now shifted to using LiPo batteries as planned.

4 Software

In the software domain, very little has changed in terms of design from our previous TDP. We are working to make more of the theoretical work implementable on our current Robot Processing Module. Compared to 2015, we have good performance in eliminating noise from the SSL data and applying proper identification to detected objects. We are working to better implement our real-time decision algorithms to demonstrate improved gameplay during the 2016 RoboCup.

Our outline for the software architecture from the previous TDP is repeated here for reference.

The high level architecture of our software is illustrated below.

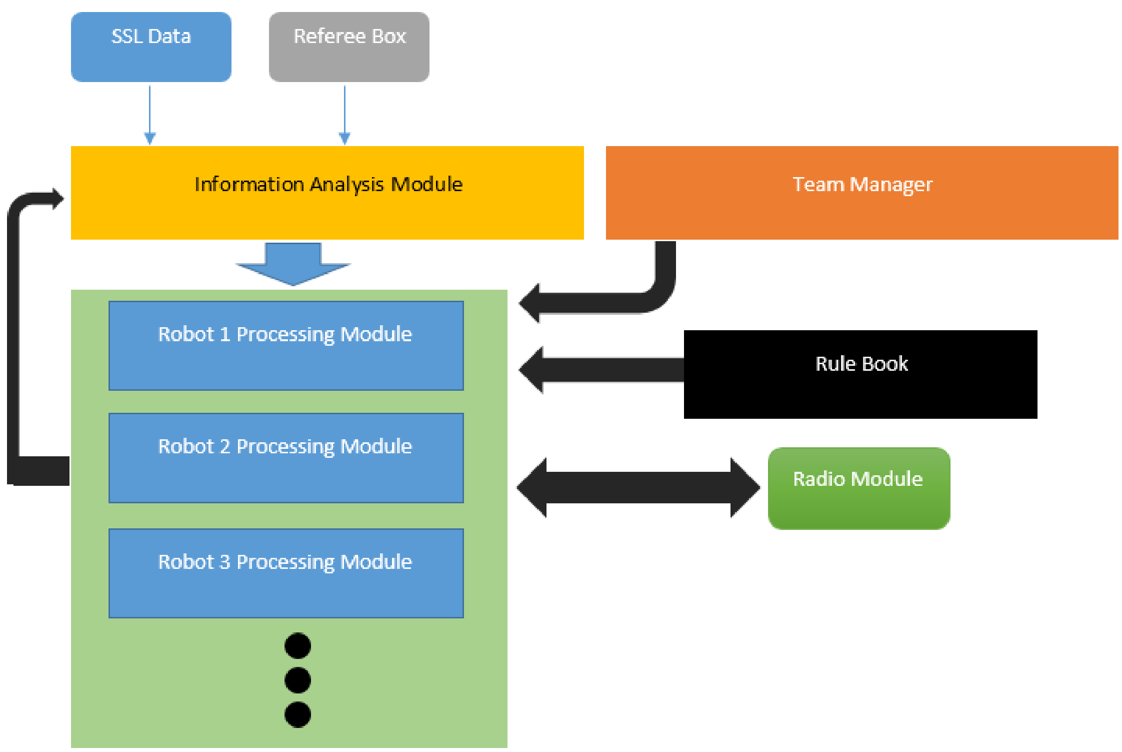

Figure 1. High Level Software Architecture

The set of Robot Processing Modules are the central unit of our software architecture. The units are fed information from the other modules as shown. The Information Analysis Module, shown in Yellow, receives data from the SSL setup and from the referee box. The module contains a database for storing all current and past information about the game and forwarding actionable data to the set of Robot Processing Modules. The Information Analysis Module also receives feedback information from the robots which includes physical information such as battery level and system health.

The Team Manager Module handles overall game strategy and guides the individual Robot Processing Modules regarding formation and objectives.

All Robot Processing Modules update their limit guidelines according to the rule book which contains F180 specifications and specifications of the equipment mounted on the robots, such as motor torque data and size of capacitor banks for the kicker. This allows robots to remain within limits of the game and calibrate themselves according to the equipment on board.

4.1 Information Analysis Module

The Information Analysis Module is responsible for preparing actionable data for the Robot Processing Modules. It receives location data from the SSL Vision setup and game status from the Referee Box input. In addition, the module receives robot status information from each Robot Processing Module.

Data from the SSL Vision system is processed through a Kalman filter. Compared to our use of the previous year's median filter, the Kalman filter provides much accurate estimates of the location of agents on the field. This is especially useful for ball tracking. Because during the game there are often momentarily detections of extra balls, the Kalman filter allows proper identification for the ball. Once filtered, the data is passed to a linked list. The Actionable Physical Data module then processes the smoothed location data to calculate primary information including position, orientation, velocity and acceleration of each game agent. This information is then accessed by the Robot Processing Modules to use as input data for their decision algorithms.

The Information Analysis Module also forwards information from the Referee Box and robot statuses so each playing robot can behave according to the game state.

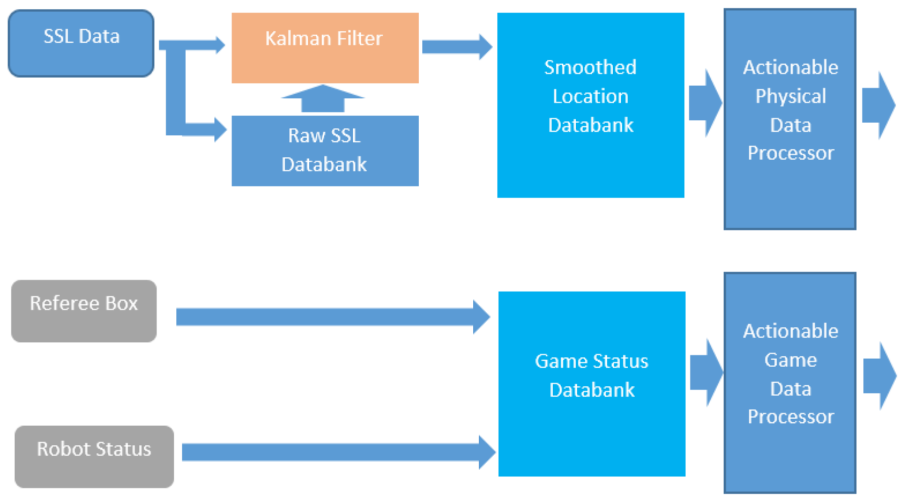

Figure 2. Information Analysis Module

4.2 Team Manager Module

Like in any real game of football, the overall strategy and style of play is determined beforehand by the team manager. Our module mimics the real world role of the team manager. The module contains a range of match and strategy variables that can be edited prior to a match. Through the several previous team description papers that we studied, almost all teams have a pre-determined playbook. The concept of a playbook is analogous to our Team Manager Module. However, they differ in that the Team Manager Module does not contain hardcoded plays. What it does is defines the team formation and responsibilities of each member prior to kick-off. It also dictates the team mentality that the players are to display during each assumed situation during the match. Team mentality dictates the aggressive or defensive behaviour a robot demonstrates during the game. An aggressive behaviour will soften decision constraints during passes and shooting. This means that in an aggressive setting, a robot will attempt a shoot even if there is greater probability of not scoring.

Additional to overall strategy, the Team Manager also sets pre-defined formation setups for free-kicks and throw-ins depending on the location where each is awarded.

In our current implementation, configuration of the Team Manager does not change during a match and is only pre-programmed before the start.

Figure 3. Team Manager

4.3 Robot Processing Module

We have approached the AI problem of autonomously playing a game of football in a modular manner. This means that the game is not controlled by one central decision module, but consists of separate autonomous modules that process decisions based on situational data. Central to our AI design is the Robot Processing Module. The Robot Processing Module contains a master algorithm that defines how a player should behave on the football field. The behaviour is governed by situational data. Each behaviour is triggered by a set of situational stimulus and is executed through its respective algorithm chain.

The Team Manager defines the overall strategy. As a blunt example, a statement in the strategy could be to switch to ultra-defensive mode after taking a lead of one goal against the opponent. The manner in which each player conducts their game is determined solely by their respective Robot Processing Module.

At the server side, a separate dedicated instance of the Robot Processing Module is run for each player on the field. The module receives actionable physical information, game state and status of other robots from the Information Analysis module. At each change in game state, the Robot Processing Module of each robot refers to the Team Manager configuration to determine any changes in formation, mentality and play style. It then updates the values of these variables accordingly.

The figure below illustrates the structure and components of the Robot Processing Module.

Figure 4. Robot Processing Module

4.3.1 Zone Planning

Each robot views the playing field as zones of responsibility. The size of each zone of responsibility varies depending on player roles. Attacking members of the team have larger zones of responsibility as there is less risk of conceding a goal at the far end of the field. Because preventing the opponent from scoring has a higher priority, the zones at the defending side are smaller and thus need more members to fill them as compared to attacking.

Additionally, zone sizes for underperforming members (as in case of low battery) will be lowered by the weightage coefficients generated by the robot status portion of the Information Analysis Module.

Each zone has a risk rating. The risk rating of the zone determines how many members of the team it needs. If the risk rating of a zone is greater than another zone, the member of the lower risk zone will leave its position and arrive at the high risk zone to assist the member there. This decision of course involves interzone distances. Members of adjacent zones to the high risk zone have higher priority to join the high risk zone.

Zone risk is a function of ball location and density of opponent members in that location. Currently risk zones are mapped as isosceles triangle patches with the ball as the far vertex and the sides of the triangle proportional to distance of the closest opponent member to the ball.

4.3.2 Game Status

The game status sub module raises direct flags to the motion decision sub module in event of spot kicks, throw-ins and game stoppage. Secondly, it sends weighting coefficient values to the Action Limits sub module. These coefficients relaxes or tightens clearance values for each motion action and essentially determines the amount of risk the robot takes while shooting, passing or following a dribble path.

4.3.3 Motion Decision

The motion decision sub module is the main decision tree used by the Robot Processing Module to determine the course of action the robot will take. The decision tree consists of dynamically ranked statements. The Path Planner sub module conducts kinematic projections and probability calculations to determine whether a decision will yield success. For example, for the shoot decision, the sub module conducts a linear sweep operation to determine the success rate of each shoot path to the target and returns the orientation and shoot speed for the path with the highest success rate.

4.3.4 Motion Execution

Once the highest ranked motion decision has been selected, the motion execution sub module resolves the corresponding vectors according to pre-coded algorithms and prepares time-coded dispatch packets containing PWM data for each motor (including the dribbler), and discharge trigger value for the kicker and chipper. The data is transmitted to the robot's channel as serial commands in the order of the time-code and is received and executed by the robot's Main Processing Unit.

Creating time-coded packets for future commands of a motion plan at the server side, and then transmitting them in real-time, is part of our development of an Execution Monitor Module and a parallel Contingency Planning Module. The objective of the former is to observe the planned motion against the real-world motion of the robot and provide information on success rate of the planned motion as events unfold. Meanwhile, with the primary planned motion already prepared, the Contingency Planning Module processes an alternative motion plan ready to be used should the Execution Monitor indicate the initial plan's failure. However, these modules are in very early stages and not our focus for the current tournament. We have made our code comply with the requirements of these modules in future to make integration easier.

5 Concluding Statements

Our inability to participate in 2015 left us with very little data to evaluate our current software in a real game against different opponents. Because of this, there has been very little change to our AI. In addition, the redesign of the mechanical structure of the robot delayed proper testing of the software. With a better design now complete, the software is being tested. We hope to update the software section of our TDP by the final TDP deadline and provide fresh material after conducting full practice matches.

References

Cauchy, Augustin (1847). *Méthode générale pour la résolution des systèmes d'équations simultanées*. pp. 536–538.

Mordecai Avriel (2003). *Nonlinear Programming: Analysis and Methods*. Dover Publishing.

Arduino Electronics Prototyping Boards: http://arduino.cc/en/Main/ArduinoBoardMega2560

Ross, I. M. *A Primer on Pontryagin's Principle in Optimal Control*, Collegiate Publishers, San Francisco, 2009.

Ohlmeyer, E.J., Phillips, C.A., Generalized Vector Explicit Guidance Journal of Guidance, Control, and Dynamics 2006; 0731-5090 vol.29 no.2 (261-268)

Survey of Numerical Methods for Trajectory Optimization;John T. Betts Journal of Guidance, Control, and Dynamics 1998;0731-5090 vol.21 no.2 (193-207)

Kalman, R. E. (1960). "A New Approach to Linear Filtering and Prediction Problems". *Journal of Basic Engineering* **82** (1): 35–45.

Wan, Eric A. and van der Merwe, Rudolph "The Unscented Kalman Filter for Nonlinear Estimation"

Minsky, Marvin (1967). *Computation: Finite and Infinite Machines*. Englewood Cliffs, N.J.: Prentice-Hall. ISBN 0-13-165449-7.

Aamodt, A. and Plaza, E. (1994). Case-based reasoning: foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1): 39-59.

Bacchus, F. and Grove, A. (1995). Graphical models for preference and utility. In *Uncertainty in Artificial Intelligence (UAI-95)*, pp. 3-10.

Nordic Semiconductor: http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01

STMicroelectronics: http://www.st.com/web/en/catalog/sense_power/FM142/CL851/SC1790/SS1555/PF63147