NEUIslanders Team Description Paper RoboCup 2020

Prof. Dr. Rahib H. ABIYEV, Dr. Pavel A. MAKAROV, Ahmet CAGMAN, Ersin AYTAC, Gokhan BURGE, Ali TURK, Nurullah AKKAYA, Tolga YIRTICI, Gorkem SAY, Berk YILMAZ

NEU Robotics Lab., Department of Computer Engineering, Department of Electrical and Electronics Engineering, Department of Mechanical Engineering
Near East University (NEU)
Lefkosa, TRNC

Homepage: http://robotics.neu.edu.tr
Contact Email: info@robotics.neu.edu.tr

Abstract. NEUIslanders team participates at RoboCup Small Size League since 2012-present. Two years ago in Montreal, Canada became SSL Division B champion. Below, it is explained in detail how NEUIslanders team improved their robots and AI from the past years.

1 Introduction

NEUIslanders is a robotics soccer team that launched under the Robotics Lab. of the Near East University (NEU). Since 2012 the team has been one of the active members of RoboCup through hard working efforts of undergraduate and graduate students, and researchers who work in a multidisciplinary manner. Until now, the team has obtained several achievements such as 3rd place in 2016 European Championship and 1st place in Division B RoboCup Championship in Canada, 2018. This year, electronics team has designed a new daughter board and has made some adjustments on the main circuit. As for the software team, we rewrote our firmware for better adjustment to work with teensy 3.6. Other than that we implement the K-means clustering method and reinforcement learning for robots, and improve the tactics layer of AI. We, the NEUIslanders team, are confident that these new changes are going to let us become champion once again after Canada. Below you can find detailed information about the above mentioned changes.

2 Electronics

This year, several upgrades have been made in the electronics section. A new daughter board has been designed to increase multifunctionality of the system. Current measurement system, voltage measurement system, DC-DC boost converter control system and LEDs have been connected to the daughter board. The daughter board contains SMD atmega328p microcontroller. Schematics and PCB layout of the circuit have shown in Fig. 1. FT232RL was added to achieve USB to serial UART interface. Therefore, board can be programmed via USB.

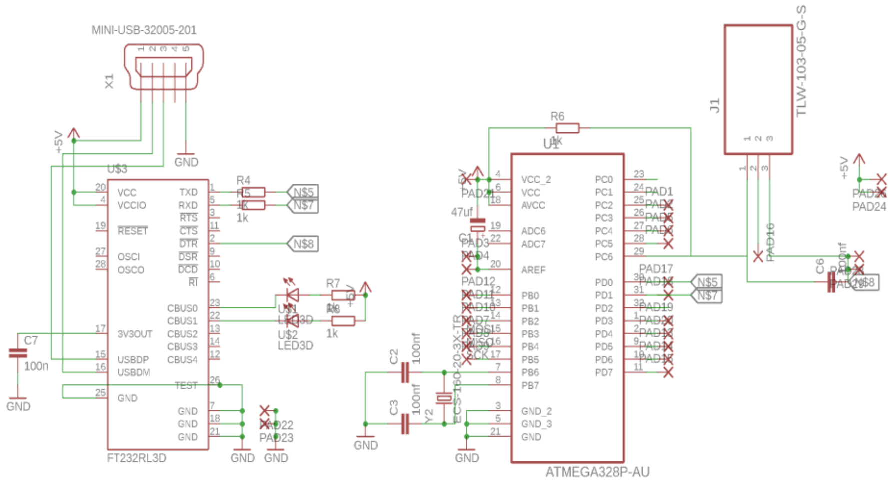

Fig. 1. Daughter board schematic.

Atmega328p ICs comes without bootloader. Therefore, Atmega328p ICs have to be burned bootloader before using them. In order to upload a bootloader, the schematics had to be built [1]. Second step was to remove 0.1 uF capacitor from reset pin. This step is very important, if capacitor is not removed from reset pin, bootloader process will fail. After that, USBtiny (AVR programmer) has to be connected to the SPI pins of Atmega328p. After SCK, MOSI, MISO, RESET, 5V and ground pins are connected, then Arduino IDE software opened. These settings are done in Arduino IDE;

<u>Tools</u>:

Board: Arduino nano

Processor: Old bootloader

Port: Choose your own port

Programmer: Arduino as ISP

When these settings are completed, burn bootloader option is clicked in tools menu. After few seconds, burning process is done. Once burning process done, 0.1uF capacitors are needed to be add reset pin after bootloader process. Regular programming process can be done if bootloader process is successful.

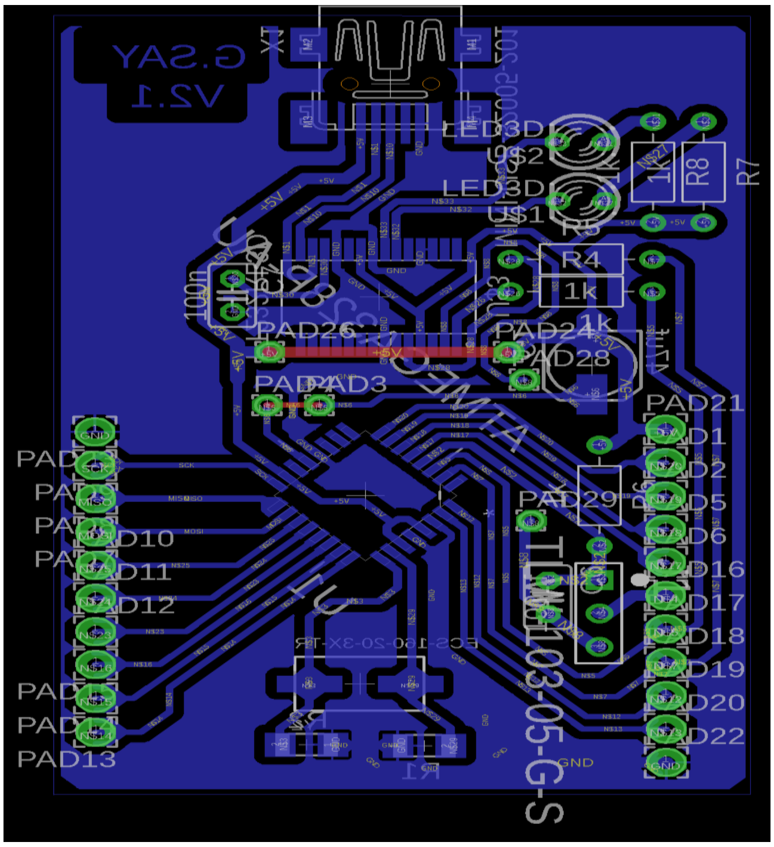

Fig. 2. Daughter board PCB layout.

3 Software

Past year the software team focused on the following areas:

1. New Firmware
2. K-Means Clustering
3. Reinforcement Learning

3.1 New Firmware

Our new circuit contains a Teensy USB Development Board which is a complete USB-based microcontroller development system. Version 3.6 features a 32 bit 180 MHz ARM Cortex-M4 processor with floating point unit. In our old system all processing was done on the central computer and only simple commands are sent to the robots. No processing other than simple radio communication was handled on the robot. With the new CPU on the robot we are gradually moving more processing to the robots such as filters and holomonic calculations. We believe this will help us with the latency, since we are now transferring much less data via the radio link and the robots themselves become more autonomous.

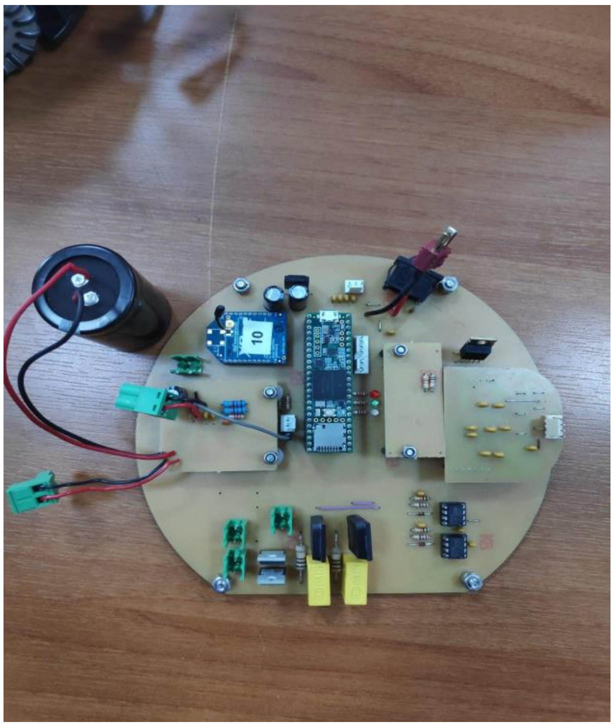

Fig. 3. Board look from above.

3.2 K-Means Clustering

This past year one problem we, the software team noticed is, our gameplay is based on man to man defense. Depending on the state of the game our software will try to assign a defending robot to each enemy robot that considered a threat. This becomes a problem if two enemy robots are right next to each other we will waste two robots to cover two enemy robots. In order to fix the problem, we applied clustering so instead of defending the robot we defend the center of the cluster which gives us extra free robots the AI can use as attackers or defenders.

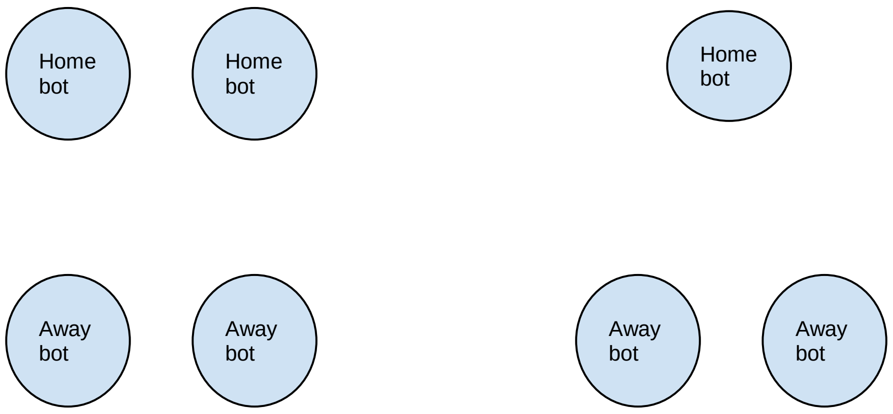

Fig. 4. Old and new defense robots.

Our algorithm of choice is K-means clustering [2] which is a simple unsupervised machine learning algorithm that groups a dataset into a user supplied number (*k*) of clusters. The algorithm clusters the data into *k* clusters, even if *k* is not the right number of clusters to use. Therefore, when using k-means clustering, users need some way to determine whether they are using the right number of clusters. One problem with this approach is system cannot know the number of optimum number of clusters beforehand. In order to detect the optimum number clusters we make use of the elbow method. The idea of the elbow method is to run k-means clustering on the dataset for a range of values of *k* (say, *k* from 1 to 10 in the examples above), and for each value of *k* calculate the sum of squared errors (SSE). Then, plot a line chart of the SSE for each value of *k*. If the line chart looks like an arm, then the elbow on the arm is the value of *k* that is the best.

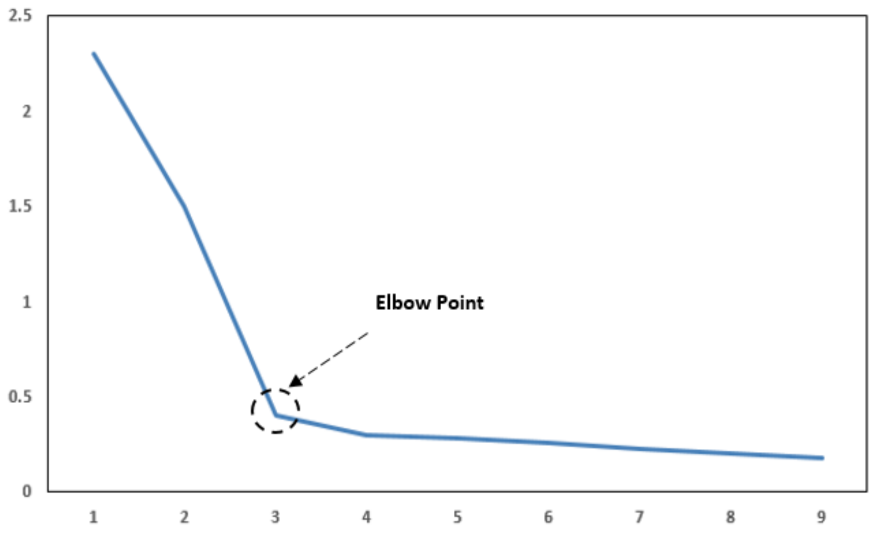

Fig. 5. Elbow Method Graph.

3.3 Reinforcement Learning

Our current sets of robots are our 3rd generation robots. We have been competing with them for 4 years now and they each produced unique mechanical flows due to crashes throughout the years. Our control system assumes all robots are uniform but due to mentioned mechanical flows one robot may have trouble with its number 1 wheel and another may have trouble with its number 4 wheels. This is problematic with our current control system. System can command the same velocity for both robots but they may actually track different trajectories. To overcome this problem we are experimenting with reinforcement learning [3].

Reinforcement learning is based on the concept of an intelligent agent. An agent interacts with the environment it's in by observing some state and then taking an action. As the agent takes an action that takes him between states, it receives feedback about the goodness of its actions using a reward signal. This reward signal is the reinforcement in reinforcement learning. It's a feedback loop that the agent can use to learn the goodness of its choice. It will pick actions that maximize the reward.

We are experimenting with the idea of pre training the algorithm using our classical control methods (PID / Fuzzy controllers) which should give us a good enough working controller then use reinforcement learning to train a dedicated controller for each robot that should be able to take into account flaws within the robot and retune itself during operation.

4 Improvement of the offence tactics

The matches played by NEUIslanders in RoboCup-2019 revealed a weak point of the team's AI part, namely the algorithms determining the behavior of the robots in attack. While in the dynamic capabilities, such as on-the-fly ball interception [4] (that is, interception without the need to place the robot at the interception point in advance), obstacle avoidance, and struggling for the ball possession (even in the absence of the dribbling device), the team appeared quite competitive, and the tactics layer of our AI showed under-elaborated. The under-elaboration was especially clear in the games against teams employing defensive strategy.

Among the drawbacks of the attack behaviors were:

- predictable positioning of the *receivers* (friendly robots which are likely to receive the pass) during a free kick;
- no use of a *bounce shot* (shooting the ball into enemy robot so that it bounces into the goal);
- almost no use of a chip kick;
- De-facto participation of no more than two robots in attack, simultaneously with keeping *idle* defenders (defending robots unengaged with blocking the threatening robots of the opposing team) in front of the penalty line, where they stayed nearly useless.

Taking all the above-mentioned drawbacks, an improved offence tactics was developed, which is currently in the stage of software implementation. The key features of this updated tactics are outlined below.

Similarly with the existing tactics of the team, enemy robots are divided into threatening and non-threatening, according to their position. For each threatening robot, a *blocking* defender (i.e. the one closing the shot line from the enemy robot to the goal) is assigned. All friendly robots unengaged in defense become attackers. According to the ball position and velocity, one friendly robot is assigned as possessing (or chasing) the ball. The chasing robot becomes an attacker as well.

In every moment of the game, starting with kick-off or free kick, four basic opportunities are analyzed, ranked by the priority:

1) direct shot to the enemy's goal (unless it is an indirect kick);
2) pass to a friendly robot, to be followed by the shot to the enemy's goal;
3) bounce shot – a shot to an enemy robot, aimed in such a way that the ball can bounce into the enemy's goal;
4) Chip kick to an area beneficial for the accomplishment of attack (presumably, the area in front of the enemy's penalty zone).

Only the first two opportunities are considered in the existing tactics.

An example of scenario, where bounce shot is preferred, is shown in Fig. 6. The advantage of the bounce shot consists in a significantly smaller time supply for the goalkeeper to react to the bounced ball. Note that in all the figures below only relevant robots are depicted and the illustrations are schematic.

An example of scenario, where both conventional pass and bounce shot are troublesome, is provided in Fig. 7. In such situation, chipping the ball towards the penalty zone can give the best effect, as the ball will likely bounce from enemy defenders and get into the possession of one of the friendly attackers in the vicinity of the zone.

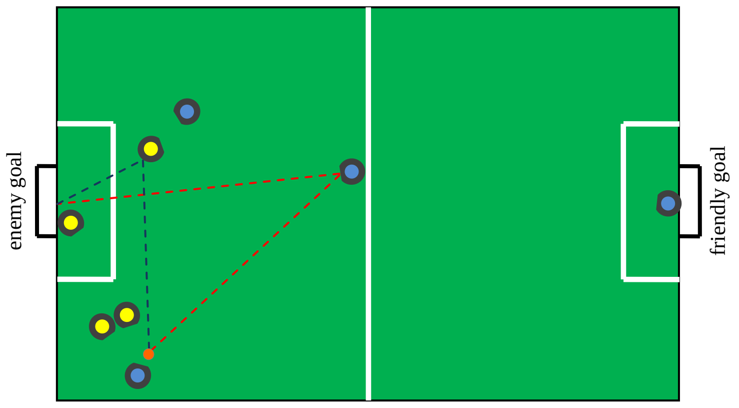

Fig. 6. Pass opportunity (red trajectory) and bounce shot opportunity (blue trajectory). In the existing tactics only the former is enabled.

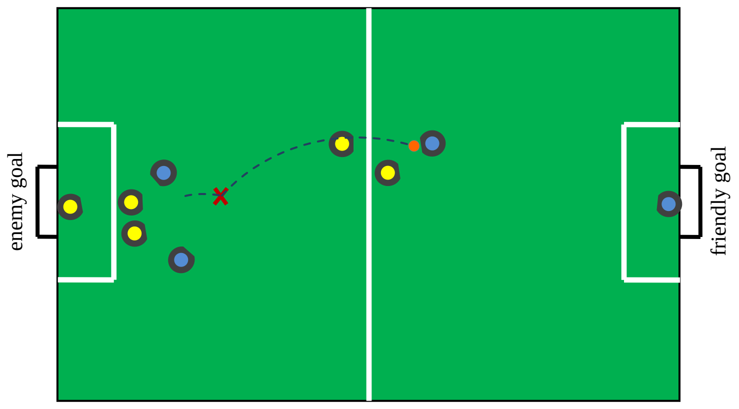

Fig. 7. A typical scenario for the chip kick opportunity.

Another novel feature of the offence tactics is a permanent dynamic search of *open* positions (i.e. positions where conventional pass-and-shoot is enabled) by the attackers which are not chasing the ball at the moment. The search is done over a discrete grid of target points in the enemy's part of the field. If the point the attacker (potential receiver) is already moving to is open, the robot continues its motion to this point, or staying at it if the point is reached. Otherwise, another target point is chosen from the grid. The choice constitutes a tradeoff of the three factors:

1) proximity to the attacker;
2) minimum of the total length of the potential pass-shoot trajectory;
3) avoidance of the situation when two or more attackers have close target points: as a simple condition, certain minimal distance between the target points for different attackers can be set, for example 0.5 m (to be specified experimentally).

The third condition is crucial, as it dictates the behavior of attackers which is likely cover the field efficiently and thus "stretch" enemy's defense. Also, this factor makes the algorithm of search collective for all attackers. The essence of the collective assignment of target points is illustrated in Fig. 8.

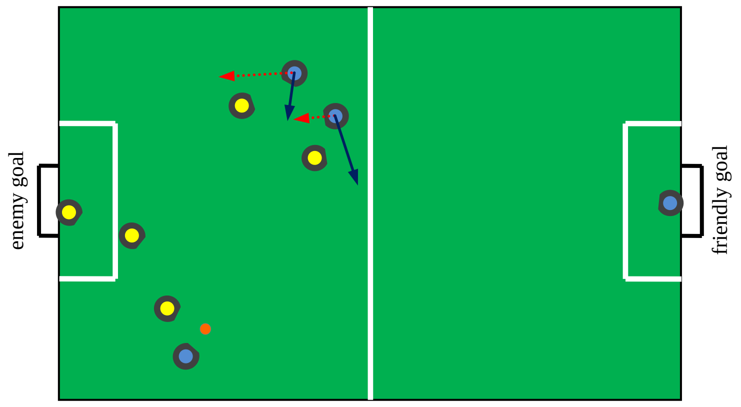

Fig. 8. Cooperative selection of open positions by two receivers. Blue solid and red dashed arrows indicate the selected and burned out target points.

References

1. http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061A.pdf
2. A. Likas, N. Vlassis, Verbeek. J.J., The global k-means clustering algorithm, Pattern Recognit. 36 (2) pp. 451–461 (2003).
3. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S. & Hassabis, D.: Human-level control through deep reinforcement learning, Nature 518 (7540), pp. 529–533 (2015).
4. P. Makarov, T. Yirtici, N. Akkaya, E. Aytac, G. Say, G. Burge, B. Yilmaz, R.H. Abiyev: A Model-Free Algorithm of Moving Ball Interception by Holonomic Robot Using Geometric Approach, LNCS, vol. 11531, pp. 166–175 (2019).