ZJUNlict Extended Team Description Paper for Robocup 2020

Zheyuan Huang[1], Haodong Zhang[1], Dashun Guo[1], Shenhan Jia[1], Xianze Fang[1], Zexi Chen[1], Yunkai Wang[1], Peng Hu[1], Licheng Wen[1], Lingyun Chen[1], Zhengxi Li[1], and Rong Xiong[1]

Zhejiang University, Zheda Road No.38, Hangzhou, Zhejiang Province, P.R.China
rxiong@zju.edu.cn
http://zjunlict.cn

Abstract. ZJUNlict has won the champion of the Small Size League of RoboCup 2019 because of the great effort made in hardware and software. In this paper, we detailedly describe the major improvements that have contributed to our success. In hardware, we optimize our robots' mechanical structure and electronic board for better stability and stronger ball control ability. Also, we increase our robots' control frequency to achieve more accurate and stable control. In software, we develop a dynamic passing strategy and an off-the-ball running module which help us gain a high possession rate and offensive threat in the game.

1 Introduction

ZJUNlict has been participating in the Small Size League of RoboCup since 2004. We seek innovation and progress in software and hardware every year, which improves our competitiveness in the game and also brings us the champion of the Small Size League of Robocup 2019[1]. Our teammates come from different majors and have done excellent work in the software and hardware groups. This paper presents our work and is organized as follows: In Sects.2 and 3, we introduce our main optimization on hardware, including mechanical structure and electronic board. In Sects.4, we described how we increase the robot control frequency to achieve better motion control. In Sects.5 and 6, we discuss the dynamic passing strategy and the off-the-ball running module respectively which helped us gain a ball possession rate[1] of 68.8% during 7 matches in RoboCup 2019. In Sect.7, we analyze the performance of our algorithms at RoboCup 2019 with the log files recorded during the matches.

[1] The possession rate is calculated by comparing the interception time of both sides. If the interception time of one team is shorter, the ball is considered to be possessed by this team.

2 Modification of Mechanical Structure of ZJUNlict

2.1 The position of two capacitors

During a match of the Small Size League, robots could move as fast as 3.25 m/s. In this case, the stability of the robot became very important, and this year, we focused on the center of the gravity with a goal of lower it. In fact, there are already many teams got there hands busy with lowering the center of the gravity, eg, team KIKS and team RoboDragons have their robot compacted to 135 mm, and team TIGERs have their capacitor moved sideways instead of regularly laying upon the solenoid [2].

Thanks to the open source of team TIGERs [2], in this year's mechanical structure design, we moved the capacitor from the circuit board to the chassis. On the one hand, this lowers the center of gravity of the robot and makes the mechanical structure of the robot more compact, On the other hand, to give the upper board a larger space for future upgrades. The capacitor is fixed on the chassis via the 3D printed capacitor holder as shown in Figure 1, and in order to protect the capacitor from the impact that may be suffered on the field, we have added a metal protection board on the outside of the capacitor which made of 40Cr alloy steel with high strength.

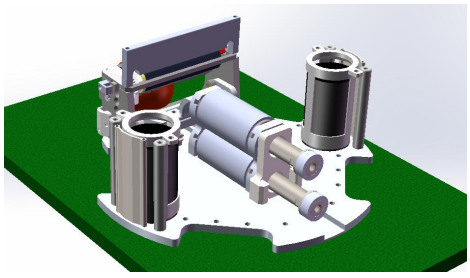


Fig. 1: The new design of the capacitors

2.2 The structure of the dribbling system

The handling of the dribbling part has always been a part we are proud of, and it is also the key to our strong ball control ability. In last year's champion paper, we have completely described our design concept, that is, using a one-degree-of-freedom mouth structure, placing appropriate sponge pads on the rear and the lower part to form a nonlinear spring damping system. When a ball with certain speed hits the dribbler, the spring damping system can absorb the rebound force of the ball, and the dribbler uses a silica gel with a large friction force so that the ball can not be easily detached from the mouth.

The state of the sponge behind the mouth is critical to the performance of the dribbling system. In RoboCup 2018, there was a situation in which the sponge

fell off, which had a great impact on the play of our game. In last year's design, as shown in Figure 2, we directly insert a sponge between the carbon plate at the mouth and the rear carbon plate. Under frequent and severe vibration, the sponge could easily to fall off[3]. In this case, we made some changes, a baffle is added between the dibbler and the rear carbon fiberboard, as shown in figure 3, and the sponge is glued to the baffle plate, which made it hard for the sponge to fall off, therefore greatly reduce the vibration.

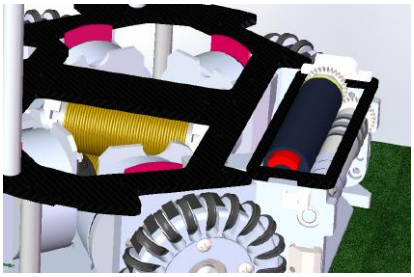


Fig. 2: ZJUNlict 2018 mouth design

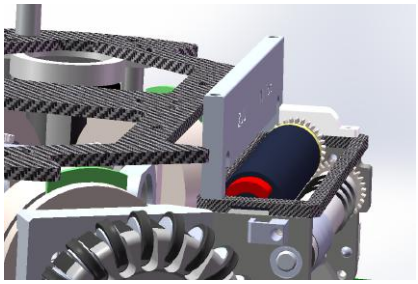


Fig. 3: ZJUNlict 2019 mouth design

3 Modification of Electronic Board

In the past circuit design, we always thought that the board should be designed into multiple independent boards according to the function module so that if there is a problem, the whole board can be replaced. But then we gradually realized that instead of giving us convenience, it is unexpectedly complicated, on the one hand, we had to carry more spare boards, and on the other hand, it was not conducive to our maintenance.

3.1 The new motherboard design

For the new design, we only kept one motherboard and one booster board, which reduced the number of boards, making the circuit structure more compact and more convenient for maintenance. We also fully adopted ST's STM32H743ZI master chip, which has a clock speed of up to 480MHz and has a wealth of peripherals. The chip is responsible for signal processing, packet unpacking and packaging, and motor control.

Thanks to the open source of TIGERs again, we use Allergo's A3930 three-phase brushless motor control chip, simplifying the circuit design of the motor drive module on the motherboard. The biggest advancement in electronic this year was the completion of the stability test of the H743 version of the robot. In the case of all robots using the H743 chip, there was no robot failure caused by board damage during the game.In addition, we replaced the motor encoder

from the original 360 lines to the current 1000 lines. The reading mode has been changed from the original direct reading to the current differential mode reading.

3.2 The new attitude transducer: IMU

To increase the motion performance of our robot, we add an IMU on our motherboard. The IMU can measure the acceleration in three directions and the angular velocity of the robot. Then it calculates the angular velocity integral and gets the real time heading angel. It is a MEMS device and can be put on the PCB. To ensure the stability of the measurements of the angular velocity we tested the IMU and got a satisfying result. The temperature drift and the time drift are low. We put the robot on the level ground and let it stay static. The deviation of the heading angel in 5 minutes is less than 0.5 degree.

With the real time heading angel data got, we can control the heading angel in the lower computer and increase both the control frequency and the accuracy. This will be further discussed in Sects.4.

4 Increase the Control Frequency

On our research platform, ZJUNlict small size soccer team, the frequency of the global vision system is 75 Hz, which determines the frequency of coordination decision, motion planning and other control instructions.

As Figure 4 shows, after obtaining the target position and target orientation from the strategy layer (only the target orientation is concerned here), the motion planner plans next step according to the current orientation and speed obtained from the global visual system, and then sends the next speed instructions to the robot.

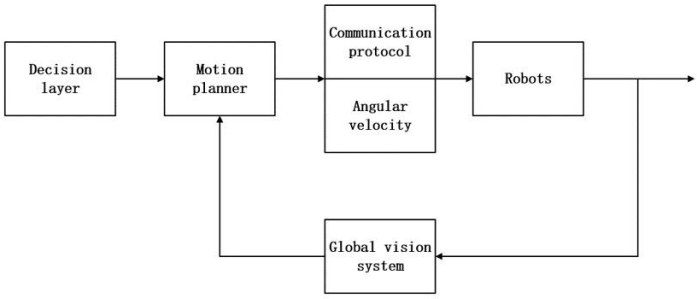


Fig. 4: Control scheme based on global vision system feedback frequency

There are four major problems with the current vision feedback system. Firstly, the frequency of the global vision system, 75Hz, is far enough for decision and planning, but it can not meet the requirements of fast and accurate

motion control. Secondly, the global vision system information feedback has much noise. According to the experimental measurement, the amplitude of the vision information noise is up to 1 degree and the error of the feedback information seriously affects the precision of orientation control. Thirdly, the vision information we obtained has been processed by vision module. It takes about 3-4 frames (40-60ms) from collecting the original vision information to obtaining the vision information. The feedback delay which makes the control precision significantly reduced (at present, it is solved by filtering, prediction and other methods) is not negligible. Fourthly, the frame rate of vision system is very unstable. When the communication is disturbed or the real-time vision processing cannot be fully guaranteed, the frame will be lost, which makes the control frequency unstable.

In order to solve these problems, the robot steering control is transferred to the slave computer, and the feedback information can be directly obtained from sensors such as gyroscope and coding plate. This improvement has achieved good results.

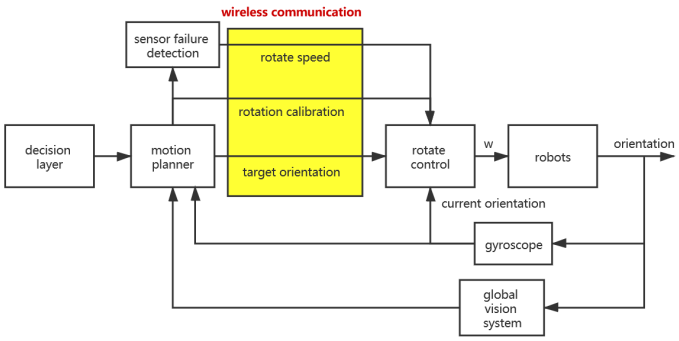


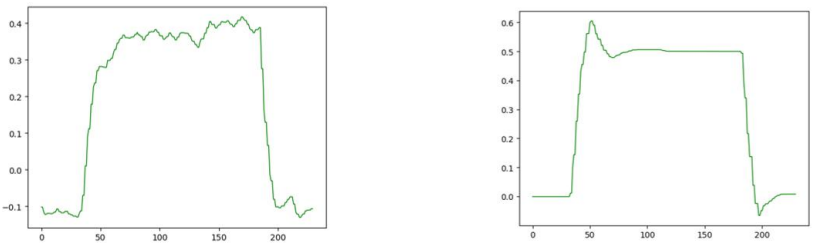
Fig. 5: Orientation control based on sensor feedback

The improved control scheme is shown in Figure 5. The host computer obtains the vision information at the frequency of 75Hz, and the robot will also upload the sensor information to the upper computer's motion planning layer by polling the packet (there are 11 robots in the field, and the communication protocol limits only four robots at a time). The host computer decides to send information of global vision or gyroscope to the slave computer according to the feedback and the instruction is target orientation, current orientation (for sensor calibration) or rotation speed ω (Sensor is broken).

The framework shown in the Figure 5 is divided into three cases: in the first case, the host computer planning layer determines that the sensor works normally and transmits the target orientation to the robot through wireless communication; when the robot gets the target orientation, it will calculate the

rotation speed to be executed according to the current orientation and speed feedback by the gyroscope, and the control frequency of this process can reach to 500Hz. In the second case, when there is big difference between the global vision and the feedback information of the gyroscope, the motion planning layer thinks that it is necessary to calibrate the sensor, that is to say, "tell" the robot its current direction, then the host computer will send the filtered angle to the robot. In the third case, when there is no gyroscope information returning or the angle of gyroscope returning is obviously wrong, the planning layer adopts the original control scheme: according to the image feedback, the rotation speed is planned and issued, and the robot steering control automatically adjusts back to 75Hz.

The new scheme greatly improves the control frequency of steering. As shown in Figure 6, the effect of steering control is significantly improved. Figure 6 is the comparison of the effect before and after the scheme improvement, and the orientation step test of 0.5rad is carried out for the robot respectively. Figure (a) is the low-frequency steering response curve based on the global vision. We can easily find that the robot is unstable to the point, and there is a steady-state error of about 0.03rad. Due to the low control frequency, the angle change curve of the robot has "sawtooth"; on the other hand, the angle of the image will also shake slightly when the robot reaches the target point. Figure (b) shows the high frequency response curve based on the sensor. It can be shown from the figure that although there is a little overshoot (the attenuation ratio is still in the acceptable range of 4:1 to 10:1), there is no steady-state error, and the response is rapid and stable.



a b

Fig. 6: Step response of robot orientation control before (a) and after (b) (horizontal axis unit: frame; vertical axis unit: rad)

As shown in the experiment data in Figure 7, the feedback angle of the sensor is very stable. Since the maximum communication accuracy of the small football robot platform is about 0.087° (12 bit signed number), the feedback angle of the gyroscope obtained at this time is stable, so it can be determined that the maximum amplitude value of the angle noise of the gyroscope is strictly less than 0.17° (the accuracy given in the parameter table is 0.01°), and the maximum image noise is The value is about 1.2°.

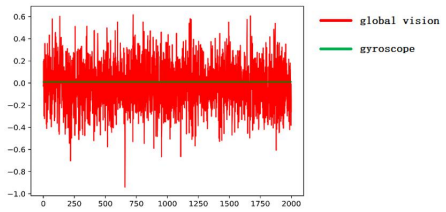


Fig. 7: Sensor and gyroscope angle feedback information

The difficulty of omni-directional wheeled robot motion control lies in the nonlinearity of dynamic model (there is coupling between robot translation and rotation). Intuitively, the rotation control and translation control of the robot are mutually disturbed. It is impossible to solve the nonlinear coupling problem by simply increasing the rotation control frequency. However, when the translation control frequency is kept unchanged and the update frequency of the rotation speed is increased to 500Hz, it can be seen that the path tracking effect of the robot is significantly improved.

In Figure 8, (a) is the track tracking effect at 75Hz control frequency and (b) is the path tracking effect at 500Hz. The reference path is three meters long and one meter wide. Both of them are the data collected by the robot running 4-5 laps in the real field. It is interesting that when the frequency is controlled at 500Hz, the trajectory of the robot in several laps almost coincides. The possible explanation is that the rotation speed of the four driving motors (four wheels) of the robot is obtained by kinematic decomposition of the motion instructions, so the frequency of updating the rotation speed is increased, which means that the frequency of updating the rotation speed of the driving motor is increased, and the uncertainty of the robot motion is also decreased accordingly.

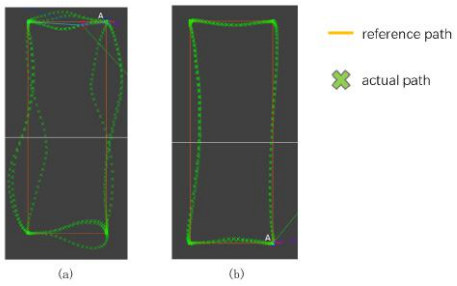


Fig. 8: Comparison of robot track tracking effect

5 Dynamic Passing Strategy

5.1 Real-time Passing Power Calculation

Passing power plays a key role in the passing process. For example, robot A wants to pass the ball to robot B. If the passing power is too small, the opponent will have plenty of time to intercept the ball. If the passing power is too large, robot B may fail to receive the ball in limited time. Therefore, it's significant to calculate appropriate passing power.

Suppose we know the location of robot A that holds the ball, its passing target point, and the position and speed information of robot B that is ready to receive the ball. We can accurately calculate the appropriate passing power based on the ball model shown in Figure 9. In the ideal ball model, after the ball is kicked out at a certain speed, the ball will first decelerate to 5/7 of the initial speed with a large sliding acceleration, and then decelerate to 0 with a small rolling acceleration.

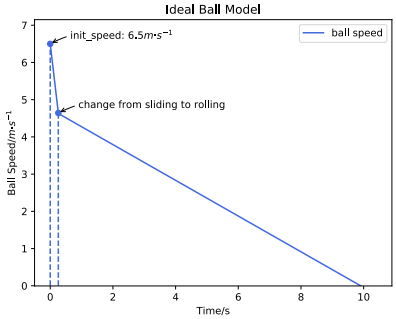


Fig. 9: Ideal ball model

Let μ, g, m, r, M, β, v_0, v_1, t be the friction coefficient of the field, the gravity coefficient, the mass of the ball, the radius of the ball, the resultant moment of the ball, the angular acceleration of the ball, the initial speed of the ball, the speed at which the ball starts to roll, and the total time the ball slides respectively. Then the speed at which the ball changes from sliding to rolling can be calculated by the following equations.

During the slide,

$$M = \mu mgr = \frac{2}{5}mr^2\beta \tag{1}$$

$$\beta = \frac{5\mu g}{2r} \tag{2}$$

When the ball changes from sliding to rolling,

$$v_0 - \mu g t = \frac{5\mu g}{2r} r t \tag{3}$$

$$\mu g t = \frac{2}{7} v_0 \tag{4}$$

$$v_1 = v_0 - \mu g t = \frac{5}{7} v_0 \tag{5}$$

Based on this, we can use the passing time and the passing distance to calculate the passing power. Obviously, the passing distance is the distance between robot A and its passing target point. It's very easy to calculate the Euclidean distance between these two points. Passing time consists of two parts: robot B's arrival time and buffer time for adjustment after arrival. We calculate robot B's arrival time using last year's robot arrival time prediction algorithm. The buffer time is usually a constant (such as 0.3 second). Since the acceleration in the first deceleration process is very large and the deceleration time is very short, we ignore the moving distance of the first deceleration process and simplify the calculation. Let d, t and a be the passing distance, time and rolling acceleration. Then, the velocity of the ball after the first deceleration and the passing power are given by the following:

$$v_1 = (d + \frac{1}{2} a t^2)/t \tag{6}$$

$$v_0 = v_1 / \frac{5}{7} \tag{7}$$

According to the capabilities of the robots, we can limit the threshold of passing power and apply it to the calculated result.

5.2 *SBIP*-Based Dynamic Passing Points Searching (DPPS) Algorithm

Passing is an important skill both offensively and defensively and the basic requirement for a successful passing process is that the ball can't be intercepted by opponents. Theoretically, we can get all feasible passing points based on the *SBIP (Search-Based Interception Prediction)* [3][4]. Assuming that one of our robots would pass the ball to another robot, it needs to ensure that the ball can't be intercepted by opposite robots, so we need the SBIP algorithm to calculate interception time of all robots on the field and return only the feasible passing points.

In order to improve the execution efficiency of the passing robot, we apply the searching process from the perspective of passing robot.

As is shown in Figure 10, we traverse all the shooting power in all directions to apply the SBIP algorithm for all robots on the field. According to the interception time of both teammates and opponents under a specific passing power and

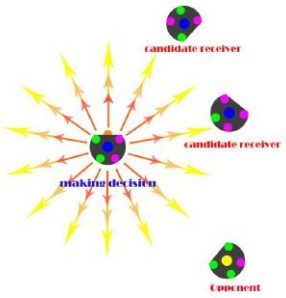


Fig. 10: Dynamic passing points searching process

direction, we can keep only the feasible passing directions and the corresponding passing power.

When considering that there is about 3 degree's error between the accurate orientation of the robot and the one obtained from the vision, we set the traversal interval of direction as 360/128 degree. And the shooting power, which can be considered as the speed of ball when the ball just kicked out, is divided equally into 64 samples between 1 m/s and 6.5 m/s, which means the shooting accuracy is about 0.34 m/s. Because all combinations of passing directions and passing power should be considered, we need to apply SBIP algorithm for 262144 times (we assume there are 16 robots in each team, 32 in the field), which is impossible to finish within about 13ms by only serial computing. Fortunately, all of the 262144 SBIPs are decoupled, so we can accelerate this process by GPU-based parallel computing technique[6][7][8], and that's why the numbers mentioned above are 128, 64 and 32.

5.3 Value-based best pass strategy

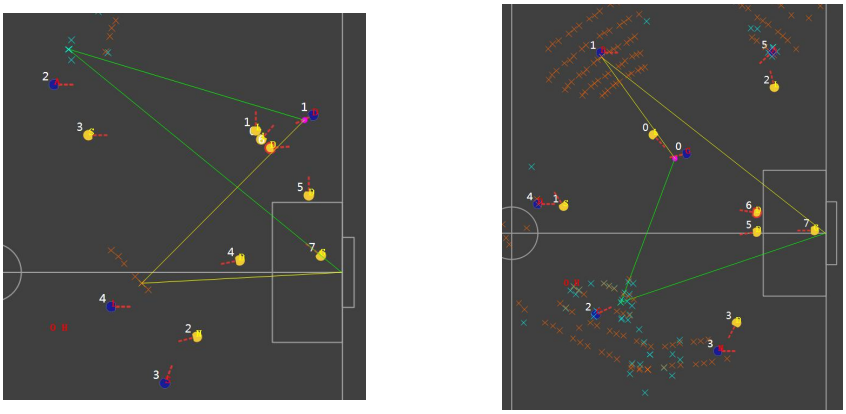
After applying the DPPS algorithm, we can get all optional pass strategies. To evaluate them and choose the best pass strategy, we extract some important features $x_i (i = 1, 2, ..., n)$ and their weights $(i = 1, 2, ..., n)$, at last, we get the scores of each pass strategy by calculating the weighted average of features selected by Equation 8 [9][10]

$$\sum_{i=1}^{n} \omega_i \cdot x_i \tag{8}$$

For example, we chose the following features to evaluate pass strategies in RoboCup2019 Small Size League:

- Interception time of teammates: close pass would reduce the risk of the ball being intercepted by opposite because of the ideal model.
- Shoot angle of the receiver's position: this would make the teammate ready to receive the ball easier to shoot.

– Distance between passing point and the goal: if the receiver decides to shoot, short distance results in high speed when the ball is in the opponent's penalty area, which can improve the success rate of shooting.
– Refraction angle of shooting: the receiver can shoot as soon as it gets the ball if the refraction angle is small. The offensive tactics would be executed smoother when this feature is added.
– The time interval between the first teammate's interception and the first opponent's interception: if this number is very small, the passing strategy would be very likely to fail. So only when the delta-time is bigger than a threshold , the safety is guaranteed.



a b

Fig. 11: Feasible pass points and best pass strategy

After applying the DPPS algorithm, evaluating the passing points and choosing the best pass strategy, the results will be shown on the visualization software. In Figure 11, the orange cross is the feasible passing points by chipping and the cyan cross is the feasible passing points by flat shot. The yellow line is the best chipping passing line, and the green line is the best flat shot passing line.

According to ***a*** in Figure 11, there are few feasible passing points when teammates are surrounded by opponents. And when the passing line is blocked by an opponent, there are only chipping passing points. According to ***b*** in figure 11, the feasible passing points are intensive when there is no opponent marking any teammate.

5.4 Implementation of Passing Skill

Based on the good dribbling ability of our robot, we developed a "break" skill to find enough space for our robot to pass or shoot while dribbling.

During the dribbling, we want to find the best point to move to, where we can ensure the feasibility to pass the ball to our target point P_{target}.To find such point, we developed a search-based algorithm. In one actual game, according to the rule, a robot must not dribble the ball further than 1 meter, so we define the point P_{start} where a robot start dribbling as the reference point. Then we use a vector $\boldsymbol{v}$ added to P_{start} to define the point to be searched. We search the length and angle of $\boldsymbol{v}$ at equal intervals with a fixed minimum interval of Δl and Δa(the max length is limited according to the rule above). At certain l_i and t_k, we can get a point P_{ik}, as shown in the Figure 12. To assess the feasibility, we make a list of the opponent robots near P_{start}, which may have chance to intercept the ball. As shown in the Figure 13, assume that we are on the blue side, location M is the location of the robot dribbling the ball, location P is one of the searched points, location E is one of the opponent robots, location T is the target point we want to pass the ball to and location N is the projection point of E on the segment PT. For each opponent robot, we take the length of segment MN, EN and PE into consideration and decide whether we can finish a pass or shoot when we move to that point (If we can, we call the point "shootable point"). For each point, we record whether we can shoot and the minimum length of PE. Among those shootable points, we will choose the one that has the maximum length of PE and then the minimum length of PT as the best point, since longer PE shows less ability of opponent to obstruct our robot and shorter PT makes it easier to pass our ball to target point. Even if none of the points is shootable points, we will still choose one according to the same rule, which seems to have more possibility to finish a pass.

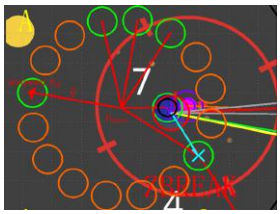


Fig. 12: Testpoints for "break" skill

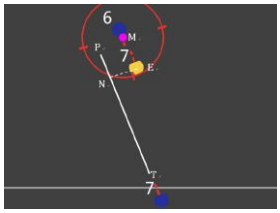


Fig. 13: Analysis of the feasibility of passing or shooting

When our robot executes the "break" skill, as long as the ball is in the robot's control, we will calculate and update the point in real-time until the robot find enough space to shoot or pass.

The application of this skill allows us to keep the ball in our possession and finish the pass or shoot in a relatively safe condition, thus improve our control of the game.

Algorithm 1 Search Move Point

Require: Δl,$\Delta\alpha$,dribbling initial point P_{start},our dribbling robot point P_{me},pass target point P_{target},threatening oppent robots $enemy[k]$,vector$\overrightarrow{v}$
maximum dribbling length l_{max},total number of threatening enemy k_{max}, $m \leftarrow 0$,$n \leftarrow 0$,$k \leftarrow 0$
repeat
  $n \leftarrow n + 1$
  repeat
    $m \leftarrow m + 1$
    repeat
      $v_{mn} \leftarrow testVector(m\Delta l, n\Delta\alpha)$
      $P_{mn} \leftarrow testPoint(P_{start}, \overrightarrow{v_{mn}})$
      $P_{enemy} \leftarrow enemy[k]$
      Analysis passing feasibility according to P_{mn},P_{me},P_{target} and P_{enemy}
      $k \leftarrow k + 1$
    until $k \geq k_{max}$
    $d_{min} \leftarrow$the minimum distance between P_{mn} and P_{enemy}
    $d_t \leftarrow$the distance between P_{target} and P_{mn}
    $P_{best} \leftarrow$ if "can shoot",choose the can shoot one
      then choose the one with farther d_{min}
        then choose the one with shorter d_t
  until $m\Delta l \geq l_{max}$
until $n\Delta\alpha \geq 2\pi$

5.5 Shooting Decision Making

In the game of RoboCup SSL, deciding when to shoot is one of the most important decisions to make. Casual shots may lead to loss of possession, while too strict conditions will result in no shots and low offensive efficiency. Therefore, it is necessary to figure out the right way to decide when to shoot. We developed a fusion algorithm that combines the advantages of shot angle and interception prediction.

In order to ensure that there is enough space when shooting, we calculate the valid angle of the ball to the goal based on the position of the opponent's robots. If the angle is too small, the ball is likely to be blocked by the opponent's robots. So, we must ensure that the shot angle is greater than a certain threshold. However, there are certain shortcomings in the judgment based on the shot angle. For example, when our robot is far from the goal but the shot angle exceeds the threshold, our robot may decide to shoot. Because the distance from the goal is very far, the opponent's robots will have enough time to intercept the ball. Such a shot is meaningless. In order to solve this problem, the shot decision combined with interception prediction is proposed. Similar to the evaluation when passing the ball, We calculate whether it will be intercepted during the process of shooting the ball to the goal. If it is not intercepted, it means that this shot is very likely to have a higher success rate. We use this fusion algorithm to

avoid useless shots as much as possible and ensure that our shots have a higher success rate.

5.6 Effective free kick strategy

We generate an effective free kick strategy based on ball model catering to the new rules in 2019[5]. According to the new rules, the team awarded a free kick needs to place the ball and then starts the game in 5 seconds rather than 10 seconds before, which means we have less time to make decisions. This year we follow our one-step pass-and-shoot strategy, whereas we put the computation for best passing point into the process of ball placement. Based on the ball model and path planning, we can obtain the ball travel time t_{p-ball} and the robot travel time $t_{p-robot}$ to reach the best passing point. Then we make a decision whether to make the robot reach the point or to kick the ball firstly so that the robot and the ball can reach the point simultaneously.

Results in section 7 show that this easy-executed strategy is the most effective strategy during the 2019 RoboCup Soccer Small Size League Competition.

6 Off-the-ball Running

6.1 Formation

As described in the past section, we can always get the best passing point in any situation, which means the more aggressiveness our robots show, the more aggressive the best passing point would be. There are two robots executing "pass-and-shot" task and the other robots supporting them[11]. We learned the strategy from the formation in traditional human soccer like "4-3-3 formation" and coordination via zones[12]. Since each team consists of at most 8 robots in division A in 2019 season[5], a similar way is dividing the front field into four zones and placing at most one robot in every part (Figure 14). These zones will dynamically change according to the position of the ball (Figure 15) to improve the rate of robot receiving the ball in it. Furthermore, we rasterize each zone with a fixed length (e.g. $0.1m$) and evaluate each vertex of the small grids with our value-based criteria (to be described next). Then in each zone, we can obtain the best running point x_R in a similar way described in section 5.5.

There are two special cases. First, we can't guarantee that there are always 8 robots for us on the field for yellow card and mechanical failure, which means at this time we can't fill up each zone. Considering points in the zone III and IV have more aggressiveness than those in the zone I and II, at this time we prefer the best point in the zone III and IV. Secondly, the best passing point may be located in one of these zones. While trying to approach such a point, the robot may be possibly interrupted by the robot in this zone, so at this time, we will avoid choosing this zone.

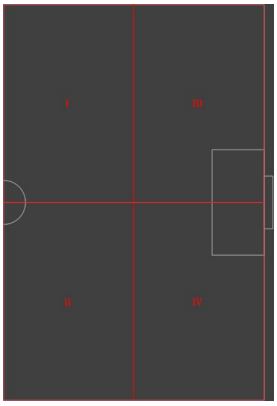


Fig. 14: Four zones divided by front field

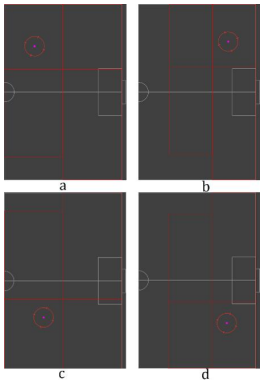


Fig. 15: Dynamically changed zone according to the position of the ball

6.2 Value-based running point criteria

We adopt the similar approaches described in Section 5.3 to evaluate and choose the best running point. There are five evaluation criteria $x_i(i = 1, 2, ..., n)$ as follows. Figure 16 shows how they work in common cases in order and with their weights $\omega_i(i = 1, 2, ..., n)$ we can get the final result by Equation 8 showed in f of Figure 16 (red area means higher score while blue area means lower score).

- **Distance to the opponent's goal.** It is obvious that the closer robots are to the opponent's goal, the more likely robots are to score.
- **Distance to the ball.** We find that when robots are too close to the ball, it is difficult to pass or break through opponent's defense.
- **Angle to the opponent's goal.** It doesn't mean robot have the greater chance when facing the goal at 0 degree, instantly in some certain angle range.
- **Opponent's guard time.** Guard plays an important role in the SSL game that preventing opponents from scoring around the penalty area, and each team have at least one guard on the field. Connect the point to be evaluated to the sides of the opponent's goal, and hand defense area to P and Q (according to Figure 17). Then we predict the total time opponent's guard(s) spend arriving P and Q. The point score is proportional to this time.
- **Avoid the opponent's defense.** When our robot is further away from the ball than the opponent's robot, we can conclude that the opponent's robot will approach the ball before ours, and therefore we should prevent our robots being involved in this situation.

6.3 Drag skill

There is a common case that when our robot arrives at its destination and stops, it is easy to be marked by the opponent's robot in the following time. We can

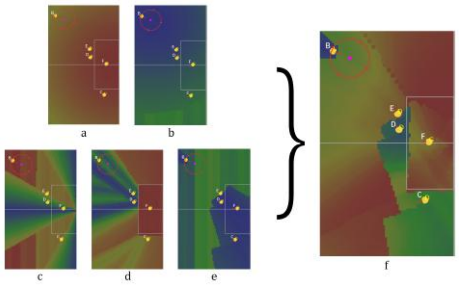


Fig. 16: How Individual evaluation criterion affects the overall

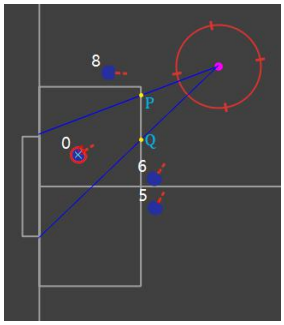


Fig. 17: Method to get location P and Q

call this opponent's robot "defender". To solve this problem, we developed a new "Drag" skill. First of all, the robot will judge if being marked, with the reversed strategy in [4].Assume that the coordinates of our robot, defender and the ball are $(x_{me}, y_{me}), (x_{defender}, y_{defender})$ and (x_{ball}, y_{ball}). According to the coordinate information and Equation(9) we can solve out the geometric relationship among our robot, defender and the ball, while they are clockwise with $Judge > 0$ and counterclockwise with $Judge < 0$. Then our robot will accelerate in the direction that is perpendicular to its connection to the ball. At this time, the defender will speed up together with our robot. Once the defender's speed is greater than a certain value v_{min}, our robot will accelerate in the opposite direction. Thus there will be a huge speed difference between our robot and defender, which helps our robot distance defender and receive the ball safely.

The application of this skill allows our robots to move off the opponent's defense without losing its purpose, thus greatly improves our ball possession rate.

$$Judge = (x_{ball} - x_{me})(y_{defender} - y_{me}) - (x_{defender} - x_{me})(y_{ball} - y_{me}) \quad (9)$$

7 Result

Our newly developed algorithms give us a huge advantage in the game. We won the championship with a record of six wins and one draw. Table 1 shows the offensive statistics during each game extracted from the official log.

Table 1: Statistics for each ZJUNlict game in RoboCup 2019. The possession rate of Game UR1 is not included in the calculation due to the radio communication interference.

Game	**Possession Rate(%)**	**Goals by Regular Gameplay**	**Goals by Free Kick**	**Goals by Penalty Kick**	**Total Goals**
RR1	66.4	2	2	0	4
RR2	71.6	3	2	1	6
RR3	65.9	0	0	0	0
UR1	–	2	1	1	4
UR2	68.2	1	0	1	2
UF	69.2	1	1	0	2
GF	71.4	1	0	0	1
Total	–	10	6	3	19
Average	68.8	1.4	0.9	0.4	2.7

7.1 Passing and Shooting Strategy Performance

Our passing and shooting strategy has greatly improved our offensive efficiency resulting in 1.4 goals of regular gameplay per game. 52.6% of the goals were

scored from the regular gameplay. Furthermore, Our algorithms helped us achieve a 68.8% possession rate per game.

7.2 Free-kick Performance

According to the game statistics, we scored an average of 0.9 goals of free-kick per game in seven games, while 0.4 goals for other teams in nineteen games. And goals we scored by free kick occupied 32% of total goals (6 in 19), while 10% for other teams (8 in 78). These statistics show we have the ability to adapt to new rules faster than other teams, and we have various approaches to score.

8 Conclusion

In this paper, we have introduced our main improvements on both hardware and software which played a key role in winning the championship last year. Our future work is to predict our opponent's actions on the field and adjust our strategy automatically. Improving our motion control to make our robots move faster, more stably and more accurately is also the main target next year.

References

1. Chen Zexi, et al.:Champion Team Paper: Dynamic Passing-Shooting Algorithm of the RoboCup Soccer SSL 2019 Champion. Robot World Cup. Springer, Cham, 2019.
2. Nicolai Ommer, Andre Ryll, Mark Geiger.:TIGERs Mannheim Extended Team Description for RoboCup 2019. RoboCup Wiki as extended team description of TIGERs Mannheim team, Leipzig, Germany, accessed April 5(2019): 2019.
3. Zheyuan Huang, et al.:ZJUNlict:RoboCup SSL 2018 Champion Team Paper. RoboCup 2018 small size league champion. Robot Soccer World Cup. Springer, Berlin, Heidelberg, 2018.
4. Zheyuan Huang, et al.:ZJUNlict Extended Team Description Paper for RoboCup 2019. RoboCup Wiki as extended team description of ZJUNlict, Sydney, Australia, accessed March 6 (2019): 2019.
5. Technical and Organizing Committee:Rules of the RoboCup Small Size League 2019. RoboCup Soccer SSL Official Web. https://robocup-ssl.github.io/ssl-rules/sslrules.html (2019). Accessed 26 June 2019
6. Sanders J, Kandrot E.: CUDA by Example: An Introduction to General-Purpose GPU Programming[M]. Addison-Wesley Professional, pp.41-114 (2010)
7. Harris M.: Optimizing Parallel Reduction in CUDA[J]. Nvidia Developer Technology. 2(4), 70 (2007)
8. Garland, M., Le Grand, S., Nickolls, J., Anderson, J., Hardwick, J., Morton, S., Phillips, E., Zhang, Y., Volkov, V.:Parallel Computing Experiences with CUDA. IEEE Micro. 28(4), 13-27 (2008)
9. Gyarmati L, Anguera X.:Automatic extraction of the passing strategies of soccer teams[J]. arXiv preprint arXiv:1508.02171 (2015)
10. Guarnizo Marin, Jose Guillermo, Martin Mellado Arteche.: Robot Soccer Strategy Based on Hierarchical Finite State Machine to Centralized Architectures. IEEE Latin America Transactions. 14(8), 3586-3596 (2016)

11. Michael Phillips, Manuela Veloso.:Robust Supporting Role in Coordinated Two-Robot Soccer Attack. In:Proceedings of the RoboCup Symposium, pp.235-246 (2008)
12. Juan Pablo Mendoza, Joydeep Biswas, Danny Zhu, Philip Cooksey, Richard Wang, Steven Klee, Manuela Veloso.: Selectively Reactive Coordination for a Team of Robot Soccer Champions. In:Proceedings of AAAI'16, the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, pp.3354-3360 (2016)