Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 10, No. 2, 2024

The Intelligent Decision-making and Collaboration Mechanism Design of the Soccer Robot System

Huangkun Chen, Liangxu Sun *

University of Science and Technology Liaoning, Anshan Liaoning, 114051, China
*** Corresponding author:** Liangxu Sun

Abstract: The robot soccer competition integrates technologies such as vision, wireless communication, control, multi-sensor fusion, strategy, and simulation, with a focus on the intelligent soccer robot path decision-making subsystem. This paper briefly analyzes decision-making methods such as role allocation, time-region control, and self-tactics, with a primary focus on the top-down hierarchical reasoning model, emphasizing robot collaboration and reflecting the concept of multi-agent cooperation. During the implementation process, the code and functions of the decision subsystem are described in detail, and examples are provided to demonstrate how to achieve robot decision-making, tactical applications, and movement. Additionally, the effectiveness and performance of the system are verified through algorithm testing.

Keywords: Decision-making; Collaboration; Soccer Robot.

1. Introduction

The robot soccer system is an interdisciplinary high-tech field that combines mechanical electronics, robotics, intelligent control, and computer vision, driving research in artificial intelligence and robotics[1]. It involves vehicle mechanics, motor integration, data fusion, and image recognition, while promoting advancements in multiple technologies and potentially impacting social, economic, and cultural development.[2]

2. System Architecture of Soccer Robots

The robot soccer system involves multiple robots working together, similar to human soccer teams' perception, decision-making, communication, and action[3]. The dynamic and unpredictable environment demands advanced technologies [4] [5]. With uneven development, robots cannot instantly achieve human-level intelligence and need gradual improvement.

2.1. Classification and Implementation Solutions of Soccer Robots

The robot soccer system mainly consists of robots, vision systems, computers, and communication systems. Its operational modes can be classified into three types based on the perception system implementation and strategy decision-making location:

1.Vision-based Remote-Controlled Robot Soccer System: Each robot is equipped with a driving agent, communication module, and CPU board, while visual data processing, strategy decision-making, and position control are handled by the main computer.

2.Vision-based Intelligent Robot Soccer System: Robots are equipped with functions such as speed control, position control, and automatic obstacle avoidance. The main computer makes decisions based on visual data and sends commands to the robots, which are equipped with sensors.

3. Autonomous Robot Soccer System: Robots exhibit autonomous behavior, with all computations and decisions made by the robots themselves. The main computer only processes visual data and transmits positional information.

Implementation solutions for robot soccer include: robot soccer simulation competitions, vision-based remote-controlled robot soccer systems, vision-based robot soccer systems, and fully autonomous robot soccer systems[5]. Simulation competitions implement all intelligent functions through software on a computer, emphasizing thinking and collaboration. The vision-based remote-controlled robot soccer system involves a team of robots performing specific functions, with particular hardware components responsible for certain tasks, such as in FIRA micro-robot competitions[7]. This vision-based robot soccer system serves as an intermediate solution toward full autonomy, where the decision-making subsystem is loaded onto the onboard micro-controller, addressing multi-agent coordination challenges[8]. In the fully autonomous robot soccer system, all functions are performed independently by the robots, with information fusion and wireless communication networks emerging as key technologies.

2.2. Design of the Soccer Robot System Architecture

The robot soccer system is typically composed of four parts: the robot chassis subsystem, the vision subsystem, the wireless communication subsystem, and the decision-making subsystem[9][10]. The vision subsystem implements closed-loop control, emphasizing computer vision and mechanics issues, laying the foundation for robot soccer competitions.

1. Robot Chassis Subsystem: Composed of communication, CPU, power supply, control, and execution modules, it is primarily responsible for task execution. The design needs to balance speed and quality, ensuring that the robot does not deviate from its planned path due to being too light in weight during the competition.

2. Vision Subsystem: Composed of cameras, image capture cards, and computers, it is responsible for real-time acquisition and processing of field images, identifying the position of the robots and ball, and transmitting the information to the decision-making subsystem. Its task is to quickly recognize images and provide precise positional and

orientation data for the entities.

3. Wireless Communication Subsystem: Uses radio communication technology, with the robot and host connected via wireless communication. The choice of communication method is crucial to the system's performance, ensuring efficient communication speed and interference resistance.

4. Decision-Making Subsystem: Responsible for formulating strategies and issuing commands based on the match situation, similar to the role of a coach. It makes decisions based on real-time data and controls the robot's technical movements, such as smooth transitions and action combinations, ensuring the system has good robustness, control, and fault tolerance, while also offering flexibility for scalability and self-learning capabilities.

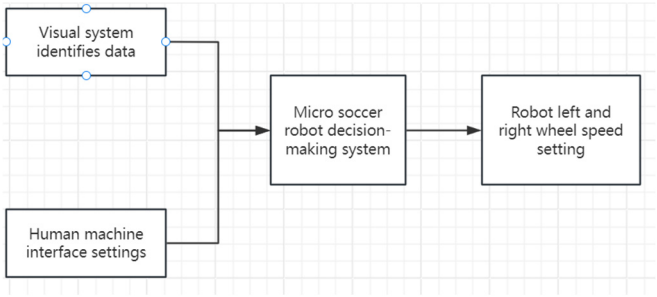

Figure 1. Decision-Making Subsystem Input-Output Module Diagram

3. Research on Decision Subsystem Model

As the intelligent core of the soccer robot system, the decision subsystem must be capable of autonomous task planning and decision-making, reacting quickly to the dynamics of the game, avoiding obstacles, and reaching target positions.

3.1. Role Assignment Method

The role assignment method can be divided into two types: based on the on-field situation and based on regions. The situational-based assignment determines the robot's role, such as defense or attack, depending on factors like the opponent's strength. The region-based assignment divides the field into different zones, with higher-priority robots leading actions and others coordinating to avoid conflict. The advantage of this method is that each robot has a fixed role and clear tasks, but the drawback is that the strategy becomes rigid, especially in region-based assignment where robots are confined to specific areas, limiting their flexibility in responding to changes on the field.

3.2. Time-based Region Control Method

This method determines the strategy based on the ball holder. By calculating the time and path of the ball within the control area, it determines control over the ball. When no one holds the ball, the nearest robot will chase it. This method can accurately assess ball control, but due to the robot's shape and the ball's movement characteristics, predicting the ball's trajectory is difficult, and it adds computational time for strategy implementation.

3.3. Individual Combat Strategy

This method determines the transition between offense and defense based on ball possession: if the ball holder gains control, the team is on offense; if the possession is lost, the team switches to defense. The advantage of this method is its simplicity and ease of implementation, but frequent transitions between offense and defense can lead to increased energy consumption for the robots and coordination issues.

3.4. Top-Down Hierarchical Decision-Making and Inference Model

In robot soccer, the environment is complex, and information is uncertain, making stability, reliability, and efficiency in the decision subsystem critical. Unlike human coaches who rely on intuitive thinking, robot decision-making depends on precise data and programmatic language, with the reasoning process divided into multiple levels, relying mainly on logical thinking to make decisions.

3.4.1. Six-Step Reasoning Model

Professor Xu Xinha and his team proposed a six-step reasoning model, which represents a top-down hierarchical deductive reasoning model for the decision subsystem. The six steps are as follows:

The first step is field information prepossessing, which involves calculating the linear and angular velocities of various entities (robots and the soccer ball) and the relative distances between these entities. This step provides the foundational dynamic information for subsequent decision-making.

The second step is offensive and defensive situation analysis and strategy selection, where, based on the information obtained in the first step, the decision is made on whether to adopt an offensive or defensive strategy. Factors such as the ball's position, movement direction, and the relative positions of players need to be considered in order to determine the appropriate strategy.

The third step is robot formation determination and role assignment. To ensure coordination and successful task completion among robots, the formation needs to be determined, and the robots' roles should be adjusted based on the current task. Role assignment must take into account the information on the field and dynamically adjust according to the situation. For example, in an offensive strategy, the role of players may be adjusted based on their position on the field.

The fourth step is targeting position determination and action selection. Once the strategy is decided, robots need to move quickly to the designated positions and perform tasks. Actions are divided into three levels: basic actions (such as turning and moving), technical actions (such as goalkeeping, shooting), and tactical actions (such as team coordination). This structure ensures that each robot can complete its assigned task.

The fifth step is robot trajectory planning, where the movement path of the robot is designed. This includes decisions on when to travel in a straight line, when to turn, and how to avoid obstacles. Various path-planning methods, such as the artificial potential field method, genetic algorithms, and others, are used in this step.

The sixth step is determination of the left and right wheel speeds. After determining the movement trajectory, the speeds of the left and right wheels are calculated to ensure the robot moves precisely along the planned path. The wheel speed calculations must take into account the mechanical and platform characteristics of the robot, as well as the underlying motion control methods.

Overall, the first, fourth, fifth, and sixth steps belong to deductive reasoning, which can be computed using mathematical models. The second and third steps rely on

heuristic reasoning, such as fuzzy logic and expert systems, to implement intelligent decision-making.

3.4.2. Simplified Reasoning Model

Step 1: Prepossessing of Input Information
Step 2: Situation Assessment
Step 3: Role Assignment
Step 4: Action Execution

This simplified model makes the subsystem structure more compact and easier to implement while saving system time. This hierarchical model for soccer robots has two notable characteristics: for control, the precision increases from top to bottom, and for information feedback, the feedback becomes increasingly coarse from bottom to top, with the level of intelligence rising correspondingly. As a research platform for multi-agent systems, soccer robots must enable cooperation among multiple players. Thus, the team's organizational capability is more important than individual player skills. This top-down hierarchical reasoning model fully reflects the collaborative spirit of the decision-making system.

3.4.3. Decision Subsystem Reasoning Framework

Before the decision system selects between attack and defense, the system must complete visual prepossessing. This includes calculating the linear velocity, angular velocity, direction angle, and the relative distance between each robot and the ball. The structure of the soccer robot decision system, constructed based on the simplified model, is shown in Figure 2. In the diagram, each robot selects the corresponding action based on its role, coordinating and collaborating to complete the match.

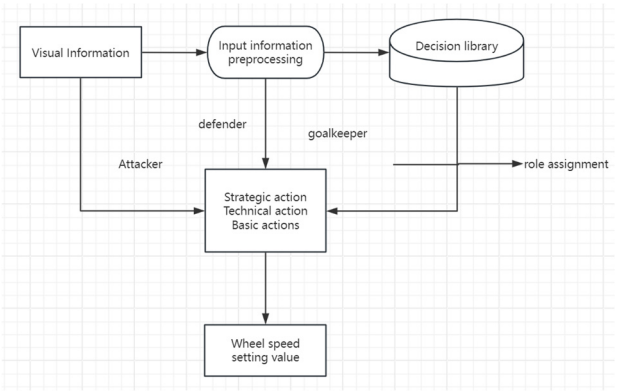

Figure 2. Decision System Reasoning Structure

4. Decision Subsystem Design and Implementation

In human soccer matches, the strategic thinking and tactical arrangements of the coach are crucial. Similarly, the decision subsystem is the strategic thought behind soccer robot matches. It interfaces with the visual subsystem above and the communication subsystem below, serving as the central hub of the entire system. The design of the decision subsystem directly impacts the outcome of the match. The position coordinates, direction angles, and other information received from the visual subsystem serve as inputs, and after processing, the decision subsystem generates motion control commands for the robots and sends them through the communication subsystem.

4.1. Decision Subsystem Environmental Code

The code defines several structures, including the coordinate variable structure `Vector3D`, the boundary structure `Bounds`, the team robot structure `Robot`, the opponent robot structure `OpponentRobot`, the ball structure `Ball`, and the `Environment` structure, which contains various match elements. These structures provide the foundation for the simulator's operation and the simulation group's robot execution. The following code snippet shows an example of the `Environment` structure:

```
typedef struct {
    Robot home[PLAYERS_PER_SIDE];
    OpponentRobot opponent[PLAYERS_PER_SIDE];
    Ball currentBall, lastBall, predictedBall;
    Bounds fieldBounds, goalBounds;
    long gameState;
    long whosBall;
    void *userData;
} Environment;
```

4.2. Algorithm Introduction

1. Potential Field Method:

In this method, repulsive potential fields are created for obstacles, and attractive potential fields are created for the target point. The robot moves toward the target under the influence of both the repulsive and attractive forces. The form of the potential field is set according to the actual situation and is commonly used in global motion planning sub-modules. However, it is not precise and is prone to getting stuck in local minima.

2.Obstacle Avoidance in the Potential Field Method:

Soccer robot obstacle avoidance is complex and involves both static and dynamic obstacles. A dynamic obstacle avoidance strategy is proposed, which transforms obstacle position information into velocity information to process real-time obstacle avoidance in dynamic environments. This strategy, as an important component of the decision subsystem, improves system performance.

4.3. Algorithm Implementation

The soccer robot strategy is implemented based on C++ programming, and the core algorithms include modules for angle adjustment, target positioning, ball chasing, and obstacle avoidance. Each module employs precise mathematical calculations and reasonable motion control logic to enable the robot to autonomously complete target positioning and movement on the field. The following outlines the implementation process of key algorithms.

4.3.1. Angle Adjustment Function

The angle function is key to controlling the robot's precise turning. Its primary task is to calculate the appropriate left and right wheel speeds for steering adjustments based on the angle difference between the robot's current orientation and the target point. First, the difference between the target angle and the current angle is calculated and constrained to the range of [-180, 180]. Then, the rotation direction and speed are determined based on this angle difference.

Specifically, when the angle difference is large, the robot will rotate faster, while when the angle difference is small, the rotation speed will slow down to avoid over-correction. The key angle adjustment process is as follows:

1. Calculate the difference between the target angle and the current angle.
2. Adjust the angle difference to the range of [-180, 180].
3. Adjust the left and right wheel speeds based on the angle difference.

The core code is as follows:

```
theta_e = desired_angle - (int)robot->rotation;
```

```
while (theta_e > 180) theta_e -= 360;
while (theta_e < -180) theta_e += 360;
if (theta_e < -90) theta_e += 180;
else if (theta_e > 90) theta_e -= 180;
if (abs(theta_e) > 50) {
        vl = (int)(-9./90.0 * (double)theta_e);
        vr = (int)(9./90.0 * (double)theta_e);
} else if (abs(theta_e) > 20) {
        vl = (int)(-11.0/90.0 * (double)theta_e);
        vr = (int)(11.0/90.0 * (double)theta_e);
}
Velocity(robot, vl, vr);
```

4.3.2. Car Localization Function

The localization function of the robot is fundamental to its motion control. The localization process consists of two parts: first, the robot must adjust its direction to face the target point, then move in a straight line towards the target. The key to this process is calculating the distance and the direction angle between the robot and the target, and adjusting the movement speed based on these parameters.

To ensure the robot reaches the target point accurately, the angle difference between the target direction and the robot's current orientation is calculated to determine the rotation angle. As the robot approaches the target point, its speed gradually decreases to improve precision. The `exp` function is used for a smooth transition, avoiding sudden speed changes that could cause instability in the movement.

The core algorithm is as follows:

1.Calculate the distance and angle between the target point and the robot's current position.

2.Adjust the steering based on the angle difference.

3.Adjust the forward speed based on the distance.

```
d_e = sqrt(dx * dx + dy * dy);
desired_angle = (int)(180. / PI * atan2(dy, dx));theta_e = desired_angle - (int)robot->rotation;
while (theta_e > 180) theta_e -= 360;
while (theta_e < -180) theta_e += 360;
if (d_e > 100.) Ka = 17. / 90.;
else if (d_e > 50) Ka = 19. / 90.;
else Ka = 25. / 90.;
if (theta_e > 95 || theta_e < -95) {
        theta_e += 180;
        vr = (int)(-vc * (1.0 / (1.0 + exp(-3.0 * d_e)) - 0.3) + Ka * theta_e);
        vl = (int)(-vc * (1.0 / (1.0 + exp(-3.0 * d_e)) - 0.3) - Ka * theta_e);
} else {
        vr = (int)(vc * (1.0 / (1.0 + exp(-3.0 * d_e)) - 0.3) + Ka * theta_e);
        vl = (int)(vc * (1.0 / (1.0 + exp(-3.0 * d_e)) - 0.3) - Ka * theta_e);
}
Velocity(robot, vl, vr);
```

4.3.3. Car Control

To enable the robot to move around the field, it is crucial to control the robot's position precisely. By setting a series of coordinate points and using the car localization function, the robot can be directed to travel to these coordinates, ultimately completing the task of navigating around the field.

The car's control program works by continuously adjusting the robot's position coordinates. By combining the localization algorithm, it ensures precise movement from one point to another. Below is the core code for controlling the robot to navigate around the field:

```
double x = env->home[1].pos.x;
double y = env->home[1].pos.y;
Position(&env->home[1], x, 16.3);
if (y <= 16.3) Position(&env->home[1], 16.8, 6.3);
```

4.3.4. Car Chasing the Ball

The process of the car chasing the ball relies on dynamic tracking of the ball's position. Once the car knows the position of the ball, it automatically adjusts its orientation and begins to chase the ball. This process requires continuously updating the car's position and using the localization function to follow the ball. The core code is as follows:

```
double x = env->currentBall.pos.x;    // Get the x-coordinate of the ball
double y = env->currentBall.pos.y;    // Get the y-coordinate of the ball
Position(&env->home[1], x, y);    // Car chases the ball
```

4.3.5. Car Obstacle Avoidance Algorithm

The obstacle avoidance algorithm is a key module to ensure the car avoids collisions and navigates smoothly. A vector algorithm is used to implement the obstacle avoidance. The car is influenced by two forces: the attractive force from the ball and the repulsive force from an opponent's car. By calculating the resultant of these two forces, the car can adaptively adjust its direction to avoid obstacles.

The algorithm calculates the attractive force between the ball and the car, and the repulsive force between the opponent's car and the car. These two forces are combined by vector to determine the car's steering direction. Below is the core code for the obstacle avoidance algorithm:

```
double bx = env->currentBall.pos.x;    // Ball coordinates
double by = env->currentBall.pos.y;
double px = env->home[1].pos.x;    // Car coordinates
double py = env->home[1].pos.y;
double ox = env->home[2].pos.x;    // Obstacle car coordinates
double oy = env->home[2].pos.y;
double r1 = atan2(py - oy, px - ox);    // Calculate the angle of the repulsive vector
double d1 = 1000 - sqrt((px - ox) * (px - ox) + (py - oy) * (py - oy));    // Repulsive force value
double r2 = atan2(by - py, bx - px);    // Angle of the attractive vector
double d2 = 1000 - sqrt((px - bx) * (px - bx) + (py - by) * (py - by));    // Attractive force value
double x = d1 * cos(r1) + d2 * cos(r2);    // x component of the resultant force
double y = d1 * sin(r1) + d2 * sin(r2);    // y component of the resultant force
Position(&env->home[1], x, y);    // Car turns towards the resultant force direction
```

5. Conclusion

The soccer robot integrates cutting-edge technologies such as computer vision, multi-agent collaboration, motion control, and wireless communication. Originating in the 1990s, it has quickly evolved into an interdisciplinary research field. It is one of the most forward-looking topics in automation and robotics, with significant theoretical and practical value.

In the soccer robot system, the decision-making subsystem plays a core role, similar to the coach in human soccer, responsible for solving collaboration and motion planning problems between robots. The quality of the decision-making

algorithm directly affects the robot's role selection, path planning, and obstacle avoidance in a complex environment, making the design of the decision-making system crucial.

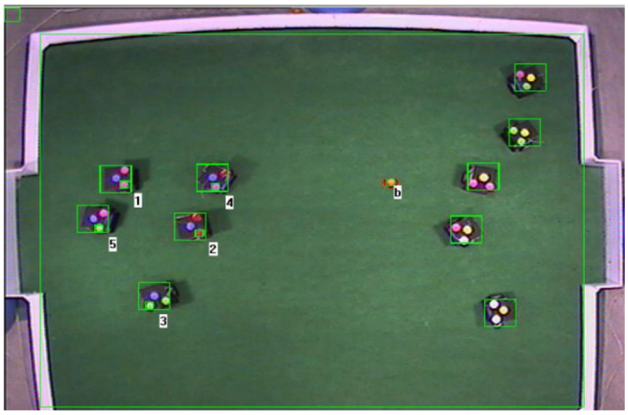

Figure 3. Schematic Diagram of the Soccer Robot System

This article focuses on several common decision-making models, including role assignment, time-zone control, and the "self-for-all" strategy. It also elaborates on the top-down hierarchical decision reasoning model, which emphasizes team cooperation and coordination. This model has been widely applied in soccer robot systems. The implementation of functions such as autonomous movement, obstacle avoidance, and ball chasing through C++ programming deepens the understanding of the decision-making subsystem.

Although progress has been made, there are still many areas that require further exploration, such as the diversity in role selection, improvement in dynamic obstacle avoidance capabilities, and the development of complex tactical actions. Future research will focus on improving the real-time performance of motion planning algorithms and developing more precise obstacle detection and avoidance algorithms to better adapt to real-world environments.

Acknowledgments

The authors gratefully acknowledge the financial support from college student innovation and entrepreneurship project of University of Science and Technology Liaoning in 2025.

References

[1] Wang Qiang, Shi Yinggang, Ye Binghui, et al. (2021). "Arc Shot Algorithm Design for FIRA Simulation 5V5 Robots." *Computer Knowledge and Technology*, 17(16): 12-15.

[2] Wang Yiping, Wang Yingkuan. (2024). "Frontier Agricultural Technology at the 2024 World FIRA." *Agricultural Engineering Technology*, 44(08): 9.

[3] Zhou Qiaojun, Bao Muze, Yu Haiqing, et al. (2024). "Design of a Football Robot Motion Experimental Platform Based on Machine Vision." *Robot Technology and Application*, (02): 37-40+46.

[4] hang Zheng, Jin Zibo, Xiang Xin, et al. (2024). "Design Method of a Multi-functional Football Training Robot Based on Trajectory Model." *Journal of Xi'an University of Technology*, 1-12.

[5] Wang Xuyang, Liang Zhiwei, Gao Xiang, et al. (2023). "Football Robot Local Trajectory Planning Based on Improved DWA Algorithm." *Foreign Electronic Measurement Technology*, 42(08): 1-9.

[6] Chen Jiali, Song Xiaoyan, Chen Weigao, et al. (2024). "Research on i Loboke Simulation Football Robot Based on SOM 3.4.2 Platform." *Science and Technology Innovation and Application*, 14(09): 21-24.

[7] Xu Junjie, Guo Lifeng, Liu Li, et al. (2023). "Feature Extraction for Accurate Positioning of Humanoid Football Robot's Vision System." *Mechanical Design and Manufacturing*, (06): 258-262. DOI: 10.19356/j.cnki.1001-3997. 20230224.002.

[8] Wang Jun, Chen Min, Dong Mingli, et al. (2022). "Research on Target Detection Algorithm Based on Multi-scale Feature Fusion." *Optical Technology*, 48(06): 749-754.

[9] Nie Shangwei. (2022). "Football Intelligent Robot Binocular Vision Goal Angle Positioning Method." *Mechanical Design and Manufacturing*, (12): 260-263+268.

[10] Zhang Zezheng, Wang Jun, Dong Mingli, et al. (2023). "Fast Detection Method of Humanoid Robot Key Points Based on Improved OpenPose." *Laser Journal*, 44(10): 65-70.