Research on Self-adjustment Method of Time Domain Signal Conflict in Multi-Robot Communication

Jianzi Jin

Zhejiang Gongshang University, China

Keywords: Multi-Robot Communication; Time Domain Signal Conflict; Conflict Self-Adjustment.

Abstract: When multiple robots communicate at the same time, a robot accepts instructions from multiple other robots at the same time, which may cause conflicts in time domain command signals, resulting in a low success rate of communication between robots. In order to solve this problem, a self-tuning method for multi-robot communication conflict is proposed. The algorithm fills the idle time slot caused by the departure command by the new arrival command and solves the time domain signal conflict. The next time slot is adjusted in advance by using the piggyback detection technology, and the idle time is reserved to reduce the generation of the idle time slot. It can reduce the idle time slot and solve the problem of low communication success rate. The experimental results show that this method can coordinate the communication relationship well, avoid command conflict and improve the communication success rate of multiple robots when they communicate at the same time.

1. Introduction

Since 1952, the rapid development of industry has put forward new application requirements for robotic technology, and robotic automation has also begun a rapid development stage. However, early robotic technology focused on single robots or the control and action of single robots [1]. With the rapid development of industrial technology and information technology, the demand of human for robots cannot be satisfied only by single robots. Under the impetus of more and more complex working environment and more stringent working conditions, the concept of a multi-robot system has been put forward accordingly. The so-called multi-robot system is not the simple sum of the number of multiple robots, nor the linear sum of the effects of multiple robots, but the coordination and cooperation between multiple robots [2].

Nowadays, in the process of multi-robot coordination, there is a problem that in order to coordinate tasks, intensive and random communication between robots is needed. By sending instructions to complete the communication work, there is a problem of command conflict in time [3]. For example, when Robot A receives instructions from Robot B, it also receives instructions from Robot C. At the same time, it will cause the conflict of instructions in the time domain signal. Once the conflict increases, it will affect the communication effect, reduce the correct rate of communication, and affect the normal operation of the whole work. Therefore, how to coordinate the smooth communication of multi-robots in the same period becomes a difficult problem in multi-robot coordination work [4-10].

In order to solve the problem of command conflict in multi-machine communication of robots, a new self-tuning multi-robot communication method based on conflict detection is proposed in this paper. This method can not only avoid the conflict between the detained instructions but also avoid the conflict between the detained instructions and the new arrival instructions. It can also make use of the piggyback instruction detection technology to adjust the next slot in advance and reduce the generation of idle slots. The algorithm fills the idle time slot caused by the departure instruction with the new arrival instruction. It can reduce the idle time slot, shorten the communication delay and improve the communication success efficiency without increasing the conflict time slot. The simulation results show that the proposed anti-collision algorithm can reduce collision time slot and

idle time slot more effectively, thus reducing communication delay, improving communication efficiency and success rate.

2. Principle of Robot Communication

In the robot communication system, the robots send each other commands with command characteristics by radio, which are recognizable by the robots, to complete the communication in the task, so as to facilitate better collaborative work. The principle of robot communication is as follows:

First, when a robot needs to communicate with another robot, it needs to get the number of another robot, according to the number, like it sends instructions. The principle is as follows:

Calculate the serial number of the robot that needs to communicate:

$$NUM = \frac{1}{N}\sum_{i=1}^{N}|x_i| \tag{1}$$

Where x_i represents the characteristics of the robot and N represents the number of features.

Then, a radio command with command characteristics recognized by the robot is sent.

$$L(t) = sqrt(\frac{\sum_{i=1}^{n} x_i^2}{n})^* t \tag{2}$$

where L is a command with command characteristics in the time domain, x_i is the command characteristics issued, n is the number of command features.

Finally, the robot receiving the instruction executes the instruction and completes some column operations. The principle is as follows:

$$f(t) = \frac{NUM_t^* L_t}{\sum_{i=1}^{N-1}|x_{i+1} - x_i|} \tag{3}$$

NUM is the number of the received sending robot. L is the parsed command received by the sender. The denominator is the wavelength feature of the command. According to the analytic f (t), further operations can be performed to complete the communication between the two robots.

According to the above principles, the communication of robots depends on the commands with time standard sent in time domain. However, this creates a problem, if in the same time period, multiple robots want a robot to send instructions at the same time. This will result in multiple instructions arriving at the same time in Formula (2), inconsistent instructions in the molecules of Formula (3), and the instructions of Robot A will be issued as those of Robot B. The conflict of instructions results in the error of instruction feedback and execution, and the communication success rate is not high.

In order to solve this problem, this paper proposes a conflict self-tuning multi-robot communication method. The algorithm fills the idle time slot caused by the departure command by the new arrival command, adjusts the instructions in the next time domain in advance, and reserves the idle time to solve the problem that the communication success rate caused by the command conflict is not high.

3. Implementation of Anti-Conflict Algorithms for Robot Communication

3.1 Instruction Conflict Feedback

In multi-machine communication, each robot instruction receiver has a counter with an initial value of 0. When the counter value is 0, the receiver responds to the request of the reader and sends the ID to the instruction reader. The first time the instruction reader sends a request, the counter of all

instructions has an initial value of 0, so all receivers respond to the reader request. The reader detects the instruction signal and sends feedback to the sender. If the feedback information is "conflict", it means that two or more instructions are sent at the same time, resulting in signal conflict. At this point, a random number "0" or "1" is generated by the instruction in conflict and added to the counter. If the feedback information is conflict-free, the counter value of all instructions is reduced by 1, until the counter value is less than 0, the instruction receiver no longer responds to the reader request. Complete pair identification and feedback, as shown in Figure 1.

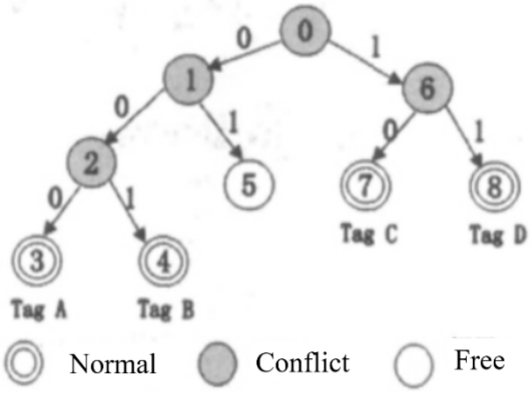

Fig. 1 Instruction Conflict Feedback

3.2 Adjusting Idle Time Slots to Prevent Real-Time Instruction Collision

Instruction reader and each instruction have two counters. The reader has counters PS and TS and the counter of instruction is PS and AS. PS represents the number of instructions that have been completed in the current cycle; AS determines which slot the instruction sends its main characteristics; TS records the maximum AS for the end of the instruction execution cycle. And the PS of all instructions and the reader always keeps the same value during the execution of instructions. The first receiving cycle begins (a receiving cycle refers to the time from the reader to receive instructions within its readable range until all instructions are executed, expressed in C_i); PS, AS and TS are initialized to 0, and when the instruction AS = PS, the instruction sends the ID to the reader. The AS < PS instruction has been executed and no longer responds to the instruction execution request until the next cycle of instruction execution is required. There are three states of the current slot: idle slot (I), no instructions to respond to execution requests; executable slot (R), which refers to only one instruction to respond to execution requests; conflict slot (C), which refers to two or more instructions to respond to execution requests. The instructions adjust the values of PS and AS according to the returned status information. The specific adjustment process is as follows:

I: If the instruction AS > PS, then the instruction AS-1;

R: PS + 1 of instruction;

C: If the instruction's AS = PS, that is, the conflicting instruction, select a random number 0 or 1 and add it to your AS. If the instruction's AS > PS, AS + 1.

Instruction reader can adjust PS and TS easily. Free time slot, TS = TS-1; Readable time slot, PS = PS + 1; Conflict time slot, TS = TS + 1. When TS < PS, the current cycle recognition ends.

In the C_i (i = 2, …, n) cycle, TS saves the value at the end of the last cycle, and the value of the detained instruction AS remains unchanged. The new instruction AS is a random number from 0-TS. The variable adjustment process of the instruction and the reader is the same as that of the first cycle.

Instructions are executed in the C_1 cycle. In C_1 cycle, there are four instructions A, B, C and D. The initial values of TS and PS are all 0, and the initial values of AS of all instructions are also 0. In the 0th slot, the read-write sending request "0" responds to the reader's request, and the reader detects more than one instruction to send data. In the first slot, a conflict signal is sent to the command receiver of the robot. The reader TS+1 receives the conflict signal and instructions A and B select "0", C and D select "1". In the second slot, instruction A and B conflict, then A choose "0" and B chooses "1", and in the third slot, only instruction A sends ID, then A is successfully executed, so repeat the above instructions and reader adjustment process until all instructions are executed. The

C_1 cycle ends and enters the C_2 cycle. In C_2 cycle, instructions A and D are detained instructions with $AS_A = 0$ and $AS_D = 3$. The newly arrived instructions E and F choose their AS as 2, G and H choose their AS as 3, and begin to execute all instructions. The reader sends the request first, and the instruction chooses which slot to send the ID according to its AS and current PS.

This method can solve the real-time conflict of robot communication well.

3.3 Piggyback Detection Technology to Prevent Delay Instruction Conflict

In the practical application of conflict-proof multi-machine communication technology for robots, the reader usually needs to repeat the instructions, and the number of instructions within the readable range of the reader varies. In the process of reading and writing, some instructions leave the readable area of the reader after the reader has executed the current cycle. The next cycle is not required to be executed, which is called leaving instruction. Some of them remain in the readable area of the reader and need to be executed again in the next cycle, which is called detention instructions. The next read-write cycle may also include new instructions that were not recognized in the previous cycle, called new arrival instructions. The method in the previous section cannot avoid the conflict between detained instructions and new arrival instructions. When leaving more instructions, it will cause a lot of idle time slots, which seriously affects the communication efficiency.

In order to solve this problem, a self-tuning conflict prevention method based on piggyback detection function is proposed. It can not only avoid the conflict between the detained instruction and the new arrival instruction but also make use of the piggyback detection technology to adjust the next slot in advance to avoid the generation of idle time slots. The so-called piggyback detection means that in the current slot, the reader reads the ID of the instruction to be read. It can also send a signal to the instruction according to whether it receives the "presence" signal of the instruction to send ID in the next time slot. The instruction can see the signal. If there is no instruction to send ID in the next time slot, the newly arrived instruction can be adjusted automatically, and the instruction that meets the condition can send ID. Using the new arrival instruction to fill the idle time slot caused by the departure instruction can reduce the idle time slot, shorten the execution delay and improve the communication efficiency without increasing the conflict time slot.

The specific idea of the algorithm is as follows: in C_i (i = 2, …, n) Cycle, each instruction has three variables PS, AS and TS_{i-1}. PS indicates the number of instructions that have been executed in the current cycle; AS indicates in which slot the instruction sends its ID, TSC_{i-1}, which identifies the TSC value at the end of the last cycle identification. The reader has three variables PS, TS and TS_{i-1}. The PS and TS_{i-1} variables of the reader and the instruction have the same meaning and the same value. TS is used to identify the maximum AS value. First, the number of new arrivals is estimated as New-count, and the new arrivals are estimated as a random number plus TS in the range of 1~New-count. The detention instruction retains the AS of the previous cycle.

The reader feedback signal is defined as follows:

I, 0: Free time slot, and no new arrival instructions that are not recognized;

I, 1: idle time slots with new arrival instructions that are not recognized;

C: Conflict time slot, with two or more instruction responses;

R, 0: The current slot is readable and there is no instruction response in the next slot.

R, 1: The current slot is readable, and the next slot has an instruction response.

In the process of recognition, the instruction of AS = PS sends ID, and the instruction of AS = PS + 1 sends "existence" signal. The reader detects the signal of the instruction, sends the feedback signal according to the instruction signal, and the instruction adjusts the PS and AS according to the feedback signal as follows:

I, 0: If the instruction AS > PS, AS = AS-1.

I, 1: If the instruction $AS = TS_{i-1} + 1$, AS = PS; if the instruction $AS > TS_{i-1} + 1$, AS = AS-1.

C: Instruction randomly chooses 0 or 1, if instruction chooses 1, when $PS \leq TS_{i-1}$, instruction $AS = TS_{i-1} + 1$ that is in conflict, instruction AS = AS + 1 that is not in conflict and $AS > TS_{i-1} + 1$ instruction AS = AS + 1; when $PS > TS_{i-1}$ instruction $AS \geq PS$, instruction AS = AS + 1 if instruction AS > PS. If the instruction chooses 0, AS does not change.

R, 0: Instruction recognition counter PS = PS + 1, when PS≤TS_{i-1}, if AS = TS_{i-1} + 1, AS = PS, if AS > TS_{i-1} + 1, AS = AS - 1; when PS > TS, AS = AS - 1.

R, 1: Instruction recognition counter PS = PS + 1.

In the operation part of the reader, when PS≤TS, the reader receives the signal of instruction and judges the current state. If there are two or more instructions sending IDs, the reader sends a feedback signal "C". If there is only one instruction sending ID, the reader receives the instruction ID, PS = PS + 1. If PS > TS_{i-1}, TS = TS + 1, the reader detects whether there is an instruction sending a "presence" signal. If there is one, the reader sends a feedback "R, 1". Otherwise, send "R, 0". If there is no instruction to send ID, when idle, if there are new unrecognized instructions, the reader sends "I, 1". If the new arrival instructions have been identified, the reader sends "I, 0" and the TS-1, thus avoiding the interference of the detained instructions.

Using the above methods, the conflicts between commands can be well solved and the normal communication between robots can be ensured.

4. Experimental Environment

In order to verify the experimental results, this paper designs a unique simulation system platform based on the software idea of "platform + plug-in", which combines entity simulation with network simulation. To accomplish the communication task of group robots, the experimental environment of this paper is a robot for soccer matches.

4.1 Simulation Results and Analysis

In robotic soccer games, it is necessary for multiple robots to communicate with each other. Therefore, the communication frequency is high and the communication randomness is strong. So, the possibility of communication conflict is greater. In order to verify the effectiveness of the algorithm, the number of instructions received by the robot and the number of successful execution instructions are counted. If the ratio of the two is high, the conflict rate is low. Anti-conflict performance is good. In a soccer game, the acceptance and execution of a robot's instructions are shown in Figure 2.

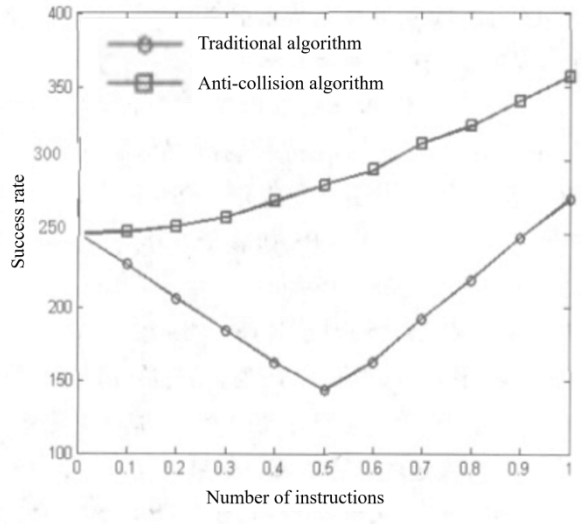

Fig. 2 Implementation of Directives

We can see from the picture above: under the same number of instructions, the success rate of the anti-collision algorithm in this paper is higher than that of the traditional algorithm.

The specific statistical results are shown in Table 1.

Table 1. Conflict statistics

	Number of instructions received	Number of instructions executed	Implementation success rate
Traditional algorithm	100	58	58%
Anti-collision algorithm	100	97	97%

From the table, we can see that this method can solve the problem of command conflict in multi-robot communication and complete the communication in multi-robot.

5. Conclusion

In view of the limitations of traditional multi-robot communication methods, a multi-robot communication algorithm is proposed according to the characteristics of time-domain command interference in multi-robot communication. The algorithm solves the problem of the low success rate of robot communication caused by command conflict. With the increasing degree of industrial automation in China, the application of robots will be more and more. Therefore, this study has good practical value and commercial value.

References

[1] Zhang Bing, Chen Wanmi, Liang Liang, Wei Yanqin. Design of small group robot soccer simulation platform based on OpenGL [J]. Journal of System Simulation, 2008, 20 (3): 724-728.

[2] Qin Fenglin, Li Xiaoming, Wang Songjie. Research on Open Simulation System for Soccer Robot Competition [J]. Electromechanical Engineering, 2009, (12): 62-64.

[3] Mu Zhichun, Guo Wenjie. Research on Human Ear and Face Feature Fusion in Identity Identification [J]. Computer Science, 2009-5.

[4] Sun Peng and Chen Xiaoping. RoboCup Small Robot Simulation System [J]. Computer Simulation, 2006, 23 (4): 128-131.

[5] Feng Yu, Yu Zhulin. Implementation of Wireless Communication System Based on Embedded Mobile Robot [J]. Electronic Science and Technology, 2009, 22 (7): 70-72.

[6] S Stancliff, et al. CMU Hammerheads 2001 team description [J]. Lecture notes in computer science, 2002:631-634.

[7] Meng, Qing-chun, Yin Bo, Xiong Jiao-she. Intelligent Learning Technology Based-on Fuzzy logic for multi-robot Path Planning. Journal of Harbin Institute of Technology [J].2001, 8(3): 222-227.

[8] JHKim, et al. A cooperative multi-agent system and its real time application to robot soccer [R].Institute of Electrical Engineers Inc (IEEE), 1997:638-643.

[9] Liu Yaxin, Wang Jinge, Wang Qiang, Zhang Junjun, Xiangzhongfan. Research on Micro Soccer Robot System [J]. Journal of Xihua University (Natural Science Edition), 2009, 28 (4): 11-15.

[10] Hu Chao, Li Yongxin, Ma Mengchao. RoboCup Small Group Soccer Robot System and Related Technologies [J]. Robot Technology and Applications, 2010, (2): 43-48.