Synchronizing a Modular Robot Colony for Cooperative Tasks Based on Intra-Inter Robot Communications

José Baca, Manuel Ferre, Matias Collar, Jose Fernandez and Rafael Aracil
Universidad Politécnica de Madrid
Centro de Automática y Robótica
Jose Gutierrez Abascal,2 , 28006, Madrid, Spain
jbaca@etsii.upm.es, m.ferre@upm.es

Abstract

The implementation of robotic cooperative tasks such as pushing an object toward a desired destination or manipulating an object using mobile robots or robotic arms requires motion coordination between the robot colony. When a robot is built by the union of several robots, such as modular robot systems, it is critical to have the complete coordination of each robot configuration within the colony and also overall robot coordination of the colony. The paper presents a demonstration of parallel motion for modular robot configurations through the combination of two types of communications, i.e., Inter-robot and Intra-robot communications. The two types of communications are described and implemented in a real modular robot system. Experiments are executed to show the performance of the robot colony synchronization.

1. Introduction

In recent years there has been increasing interest in modular robot systems: Systems which aim to carry out multiple tasks using cooperative robot teams. Modular robot systems are less task-specialized in comparison to industrial robots; however a few of them are necessary for cooperative task execution.

Modular robot systems are capable of forming different robot configurations made up of n-modules, which have to work in a coordinated fashion to show uniform behavior. Their ability to rearrange their modules and to adapt to different circumstances, allows them to cope with multiple tasks such as different types of displacement and manipulation, a feature ensuring that their performance level excels in circumstances as such. The main idea of modularity is that the functionality of an entire system is greater than the sum of its components.

Control architecture is a key factor to successfully perform cooperative tasks. Particular features concerning communication and synchronization, amongst the modular robot systems, will affect the planning of modular robot architecture in several ways, e.g., if coupling among modules is to be carried out mechanically, then a hierarchical or centralized architecture would normally be implemented [1] [2]. On the other hand, if the module coupling is not mechanical, such as with networked robots [3], then distributed and decentralized architectures would be used. Examples of this are robot systems based on colonies [4]. There are a variety of published works relating to the synchronization of systems such as, [5] [6] [7]. Synchronization is of great importance when two or more robots have to cooperate. e.g. multi finger robot-hands, multi robot systems [8] and master-slave systems [9].

Synchronization may be defined as the mutual time conformity of two or more processes [1]. This conformity can be induced through artificial interactions in the system, e.g., input control feedback, resulting in what is known as controlled synchronization. In controlled synchronization distinction should be made between inter-synchronization, whereby interconnections between all the systems occurs, such as swarm systems, and intra-synchronization, where there are only interconnections from the leader or dominant system to the non-dominant ones e.g., master-slave systems. In tasks that cannot be carried out by a single robot, either because of the complexity of the task or the limitations of the robot, the use of a modular robot system has proved to be a good alternative. The paper proposes a method to synchronize modular robots within the colony and also overall robot coordination of the colony through the combination of two types of communications, i.e., Inter robot and Intra robot communications. The general setup of this paper is as follows. The synchronization methods presented are implemented on actual modular robots, i.e., SMART. These robots have been built in order to study the advantages of

using such a system to execute highly complex and cooperative tasks. Consider a colony formed by M-Robots with the purpose of performing simultaneous movements to accomplish a given task, such as pushing or manipulating an object. To accomplish the task, there will need to be synchronization between the M-Robot modules and eventually synchronization between M-Robots.

The paper is organized as follows: The SMART architecture is briefly presented in Section 2. The synchronization method within the M-Robot using intra-robot communication is presented in Section 3. Section 4 presents the synchronization method performed by the inter-robot communication. The experimental results of the synchronization methods applied to the colony are presented in Section 5. Conclusion remarks are offered in Section 6.

2 SMART Architecture

The system architecture is divided into modules, M-Robots and colonies. *Modules* are base system components and are classified in three types of hardware modules, i.e., *power/control module (P/C)*, *joint module (J)* and *specialized module (S)* as shown in Fig. 1 and also software modules, i.e., *master module* and *slave module*.

The design of the SMART modules aims to balance complexity with functionality. The goal is to build robots (M-Robots) that offer movement flexibility whilst allowing a variety of locomotion modes and reconfiguration capabilities.

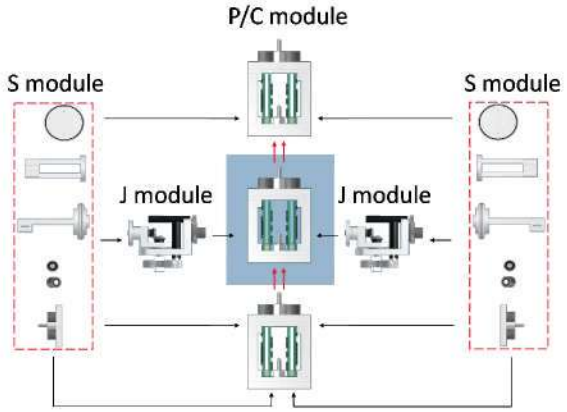

Figure 1. Three types of modules may be combined to form different robot configurations

The desired goal when dealing with communication architecture is for M-Robots to behave as robots that cooperate with each other. Moreover, that M-Robot configuration can change during task execution, either by docking or undocking modules to/from itself. For such an objective, synchronization and communication mechanisms are essential. Figure 2 shows a scenario where two M-Robots cooperate as a colony to execute a common task. There are 3 communication channels. The first channel involves modules that belong to the same M-Robot. The second and third channels involve M-Robots and/or control stations. The three communication channels are enabled through CAN bus, Bluetooth (BT) and RF-video technologies.

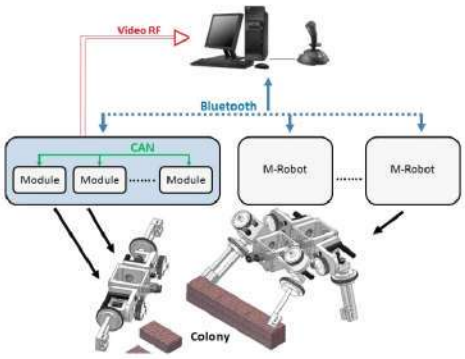

Figure 2. The SMART System

2.1 Intra-robot communication

Intra M-Robot communication refers to communication between P/C modules. Since elements are mechanically linked, it is performed via CAN bus technology. CAN bus is widely used in industrial environments since its application in modular robots offers considerable advantages. One important CAN bus feature is transmission speed, which can reach 1 Mbit/s, hence, all the inner M-Robot communications are executed via CAN bus.

2.2 Inter-robot communication

In some cases, there is no physical contact between elements and the communication has to be carried out by wireless technology. This is the case between the control station and the master module of each M-Robot. Therefore, BlueTooth (BT) protocol is used. Nowadays, BT devices are widely used, mainly in office environments and computer peripheries. Their main advantage when applied to small systems, such as the SMART, is the avoidance of wiring and the ease for its application. However, the main constraint of this technology is its low rate of data transmission, which is about 19.2 kbits/s. Additionally, another limitation lies on the maximum number of BT devices that can be connected within the same network, or piconet. Such a network can have no more than seven nodes.

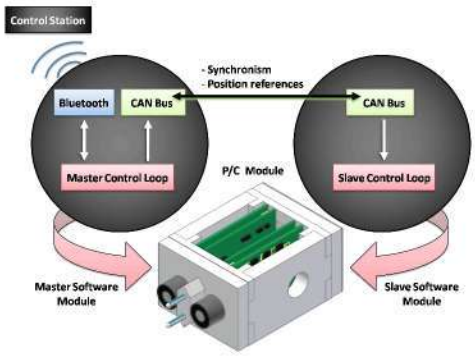

Figure 3. Master and slave modules

2.3 Master and slave software modules

The master software module (MsM) manages external communications (via BT) and synchronizes the slave software modules (SsM) within the M-Robot configuration. The master module of each M-Robot is responsible for preprocessing the commands and dispatching the corresponding command to the rest of the slave modules via CAN bus. The information sent internally within the M-Robot are low-level commands, i.e., references for position control loops, information for synchronization and both sensor and actuator data as shown in Fig. 3 . The slave modules receive the message via CAN bus and execute the desired low-level commands. The MsM and/or SsM are downloaded into the P/C module. The P/C module contains the electronic boards, the system power source, the communication peripherals, the mechanisms which physically connect the other types of modules and, depending on the robot configuration, may contain up to two software modules, i.e., MsM-SsM, MsM-MsM or SsM-SsM.

3 Synchronizing M-Robot modules using Intra-communication

The M-Robot displays robot behavior thanks to module synchronization. This global behavior is required to move the M-Robot as a unit. This is a key feature in implementing collaborative behavior in a robot team. For example, in some modular robot configurations, in order to complete an action such as displacement of the modular robot, it would require simultaneous similar movements across numerous modules. In other words, it would require starting and finishing all joint module movements at the same instant of time. With this type of synchronization it is possible to execute complex tasks.

One of the problems in distributed computational systems, is the lack of a global clock [10]. This is a handicap for carrying out concurrent actions in a coordinated way. Several approaches have been proposed to overcome this problem and the one used most is the message-passing method [11]. From the computational point of view, modular robots are a set of processing units joined by a communication bus. Therefore, message-passing methods are seen as a possible solution to synchronization problems in modular robots. However, many messages are needed to synchronize different processes. We propose a closed-loop discrete time method, which keeps all system clocks in the same phase and period, using a single short message in every cycle. The period of that cycle can be longer (seconds) than the control cycle period (milliseconds, typically the period of timer interruption). The method needs a periodic signal acting as a trigger for the closed loop. This signal is generated by the master module for every N control cycles and consists of a high priority short CAN message. In theory, all modules receive the message at the same time, but this is not correct. Each time the message is received, a local timer (tick counter) is reset and its previous values are used to correct the local timer period, i.e., there is a counter that counts the ticks of a local timer (ticks of DSP clock) between every two consecutive synchronizing signals. When a synchronizing message enters the process, the current counter value is used to recalculate the local timer period. Consequently the counter is reset. Note that this process occurs in every single module.

$$T_{i+1} = \frac{\arctan C \times T_i + t}{N}. \quad (1)$$

Where:
C: Num. of control cycles; T_i: Current local timer period.
t: Current cycle timer ticks; N: Ctrl. cycles / synch. period.

The master module creates a synchronization signal. It counts the number of interrupts generated by its timer until 300 units, and a message is then delivered to the slave mod-

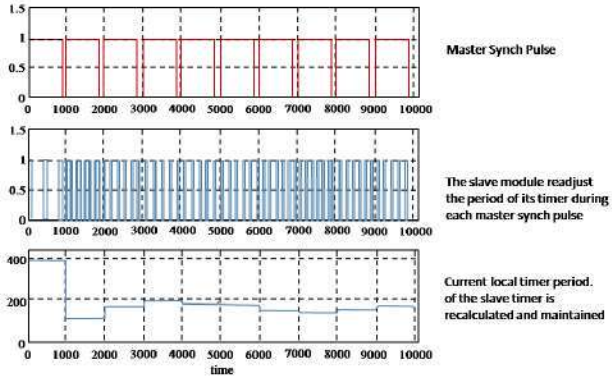

Figure 4. Synchronization using Intra-robot communication

ules. The slave modules receive the message and adjust the period of their timer. To pick up the synchronization signal, a low level interrupter message associated with the successful reception of a message in a defined mailbox is programmed. Once the message has been received, the period of every slave timer is recalculated following the equation (1) and all the modules clocks should have the same phase and period, as shown in Fig. 4. In this way actions can be executed in a coordinated manner. This algorithm for module synchronization achieves single robot behavior with the M-Robot. For example, using the above method it is possible to change from one modular robot configuration to another configuration with simultaneous joint movements, as shown in Fig. 5. The starting-stopping time for any motor is below one millisecond. Synchronization is required in order to properly execute trajectories or in other cases, simultaneous forces are required for manipulation tasks.

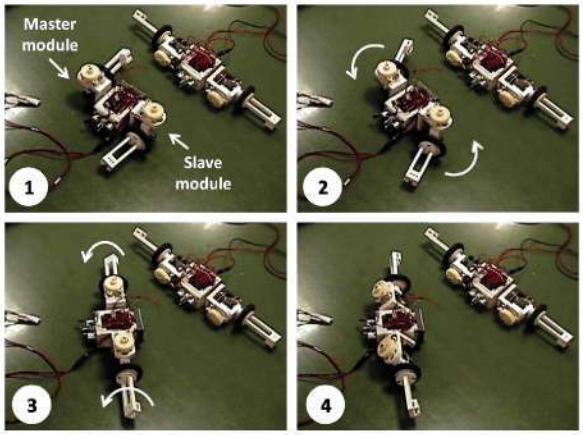

Figure 5. A synchronized 2M-Robot structure

4 Synchronizing a colony of M-Robots using the inter-communication

The main idea of synchronizing a colony of modular robots is coordination in the execution of its tasks. Simultaneous actions require synchronized joint movements that would otherwise not be correctly achieved. For instance, an object which has to be lifted between two or more robots, an object which has to be moved through a passage, etc. Coordinated movements are vital to accomplish such a task in an optimal way. In the M-Robot, the slave modules are synchronized by means of intra-robot communication, i.e., the master module periodically sends a synchronization signal by CAN bus to the slave modules to readjust their timer or control loop. Now, the idea is to synchronize master modules of each M-Robot. In order to achieve the desired synchronization, a control station (PC) is used as the main coordinator of tasks.While the master module synchronizes the slave modules, it periodically sends a signal (pulse A) to the control station via BT indicating its synchronization signal. In order to synchronize a second M-Robot from the colony, it is required to acquire a second signal from the second master module. The second M-Robot executes the same previous process, i.e., it periodically sends a signal (pulse B) to the computer. At this time, the control station has two signals originating from each M-Robot. To be more specific, each signal originates from the master module of each M-Robot. The control station calculates the clock skew (CS) between pulse B with respect to pulse A, represented by equation $CS = C_A(t) - C_B(t)$.

Once the difference between pulse A and pulse B is calculated, the difference of internal counts between master modules is then calculated according to the CS. As a consequence, it then proceeds to correct the internal count of one of the modules with the information sent by the control station via BT. The robot receiving the count modification command will adjust its internal count to that value of the internal count of the reference robot. Thus, synchronization is achieved for both robots. The motion control loops are executed at the same time, thereby enabling the execution of simultaneous movements in the robots (Fig. 6). The timing calculations do account for communication delays which are expected with BT message transmission. The operator can set the period of transmission of the sync signal for the robots as well as the minimum allowable gap in synchronization signals. When not allowing a minimum CS between the robots, the algorithm would constantly calculate a count correction of one of the robots due to the variability of the communication delays via BT and the computer calculation.

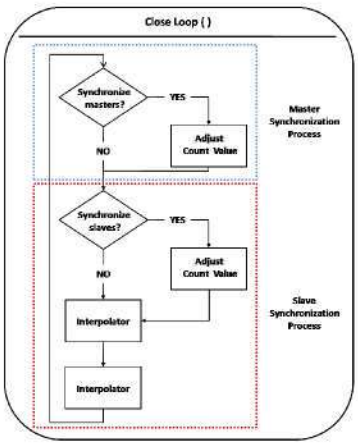

Figure 6. Synchronization flow chart

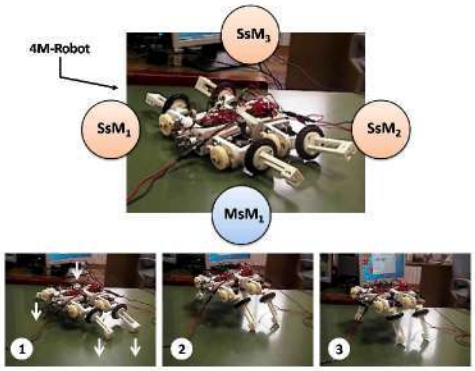

Figure 7. Intra-robot communication allows synchronization within the 4M-Robot modules for changing configuration with simultaneous joint movements.

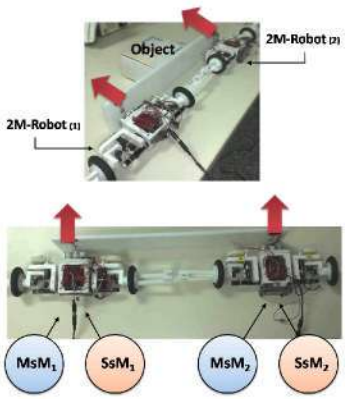

Figure 8. Synchronizing a modular robot colony for cooperative tasks

5 Experiments and results

The experiments that are described below show the behavior of the colony and the way the robots act separately when synchronized. The first experiment consists of testing the synchronization via the Intra-robot communication method. As presented earlier in section 3, Fig. 5, the synchronization within a 2M-Robot is accomplished, i.e., the master module synchronizes the slave module. Considering the scalability of the system, a new experiment is performed with a 4M-Robot, i.e., the master module should synchronize three slave modules. The test consists of changing from the mobile configuration to the four-legs configuration. The movement coordination is crucial for this specific task. Fig. 7 shows the 4M-Robot movements during the execution of the task.

The second experiment consists of testing the synchronization via Inter-robot communication to push an object. The scenario consists of a colony composed of two 2M-Robots, as shown in Fig. 8. Each 2M-Robot has a MsM which constantly transmits the synchronization signal to the control station (via BT) and to its SsM (via Intra-robot communication). At the control station two signals from both MsM are received, i.e., a blue pulse from the master module originating from one of the two 2M-Robot (1) and a red pulse from the second master module as shown in Fig. 9. A CS exists between them and if the robots receive a command to move during that period of time, the tasks executed by both 2M-Robots would not be synchronized.

The encoder value variation situated at each actuator shows the instance of time when the joint movement is executed. One of the actuators from the 2M-Robot (1) begins the movement before the actuator from the second 2M-Robot (2). This implies that a simultaneous task cannot be achieved, as shown in Fig. 10 (left). Each robot will execute commands, according to their own internal control loop. Applying the synchronization method via inter-robot communication to the colony the synchronization results are detailed in Fig. 9. One can see that the second master module receives a synchronization signal to correct its own count. The following pulse signal of the second master module is in synchronization with the first master module. The third graph displays the CS between both master software modules. It is clear how the CS decreases after the robots are synchronized. Figure 10 (right) displays a synchronized modular robot colony utilizing inter-robot communication. The encoder values from two actuators, each one from each 2M-robot are illustrated to demonstrate the coordinated joint movement. The joint movement of each M-Robot begins at the same time and it shares equal behavior during the test.

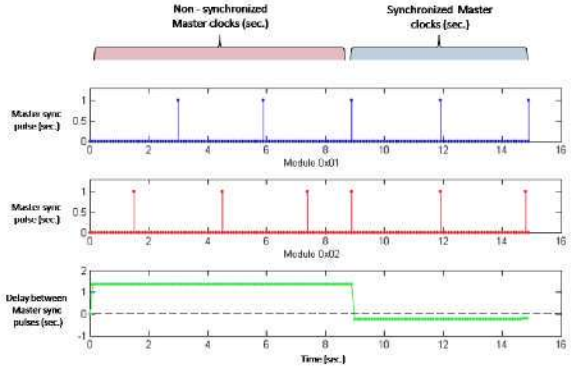

Figure 9. Synchronizing pulses generated by the colony

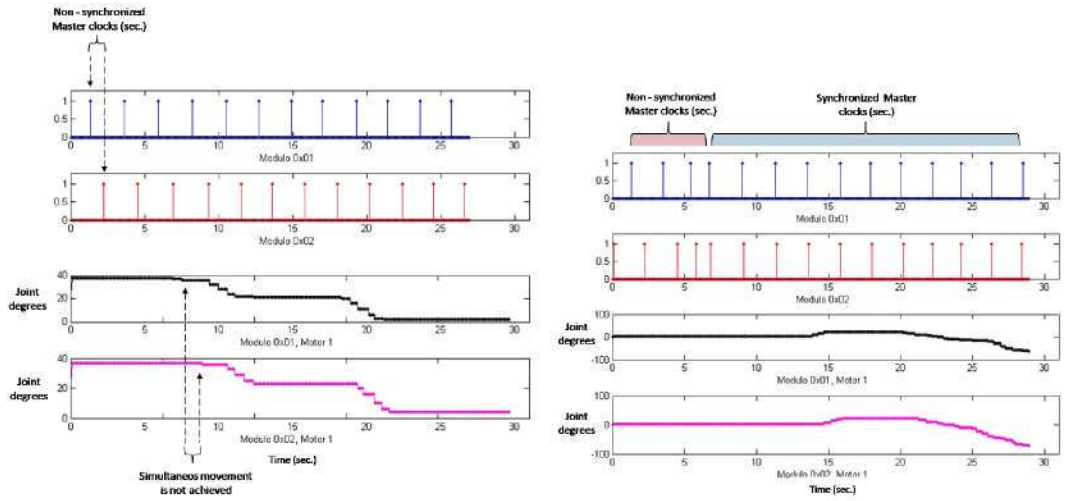

Figure 10. Non-sync joint movements (left) and Sync joint movements into the colony (right)

6 Conclusion

Intra-robot communication is used to synchronize a set of two or more physically connected robot modules. Inter-robot communication is used to synchronize modular robots that are not physically connected between them. By combining both type of communications a colony of reconfigurable modular robots may be synchronized to perform simultaneous actions regardless of robot configuration. It has been demonstrated that the synchronization is achieved between the modules of the M-Robot, as well as between the M-Robots within the colony. Test results of the synchronization method implemented in the SMART system demonstrate optimized performance during cooperative tasks.

7 Acknowledgment

The authors would like to thank CICYT and CONACYT for financing this research project. The SMART project has been financed within the Industrial Design and Production Program (DPI2003-00759 and DPI-2006-06493).

References

[1] M. Yim, Ying Zhang, K. Roufas, D. Duff, C. Eldershaw, "Connecting and disconnecting for chain self-reconfiguration with PolyBot," IEEE/ASME Transactions on Mechatronics, vol.7, no.4, pp.442-451, 2002.

[2] S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, S. Kokaji, "M-TRAN: self-reconfigurable modular robotic system," IEEE/ASME Transactions on Mechatronics, vol.7, no.4, pp.431-441, 2002.

[3] G. McKee, P. Schenker, "Networked robotics," in Proc. Sensor Fusion and Decentralized Control in Robotic Systems III, SPIE, 2000.

[4] Navarro-Serment, L.E., Grabowski, R., Paredis, C.J.J., Khosla, P.K., "Millibots," IEEE Robotics & Automation Magazine, vol.9, no.4, pp.31-40, 2002.

[5] R. Gusella, S. Zatti, "The accuracy of the clock synchronization achieved by TEMPO in Berkeley UNIX 4.3BSD," IEEE Transactions on Software Engineering, vol.15, no.7, pp.847-853, 1989.

[6] I. I. Blekhman, P. S. Landa, M. G. Rosenhlum, "Synchronization and chaotization in interacting dynamical systems," ASME Applied Mechanical Review, vol. 48, pp.733-752, 1995.

[7] H. Nijmeijer, A. Rodriguez-Angeles, "Synchronization of mechanical systems," World Scientific Publishing, Singapore, 2003.

[8] Y. H. Liu, Y. Xu, and M. Bergeman, "Cooperation control of multiple manipulators with passive joints," IEEE Trans. Robotics and Automation., vol. 15, pp.258-267, 1999.

[9] A. Rodriguez-Angeles and H. Nijmeijer, "Coordination of two robot manipulators based on position measurements only," International Joumal of Control, vol. 14, pp.1311-1323, 2001.

[10] Chow, Randy, and T. Johnson., "Distributed Operating Systems and Algorithms," Addison-Wesley, 1997.

[11] J. Xu, K. Hwang, "Heuristic methods for dynamic load balancing in a message-passing supercomputer," Proc. Supercomputing '90, pp.888-897, 1990.