An orchestrator for networked control systems and its application to collision avoidance in multiple mobile robots.

AGRAWAL, S., JAIN, S.K. and IBEKE, E.

2021

An Orchestrator for networked control systems and its application to collision avoidance in multiple mobile robots

Abstract

Networked Control System (NCS) consists of controlled distributed nodes while an Orchestrator functions as a central coordinator for controlling the distributed tasks. The NCSs have challenges of coordination and right execution sequencing of operations. This paper proposes a framework named Controlled Orchestrator (COrch) for coordinating and sequencing the tasks of NCSs. An experiment was performed with three robotic vehicles that are considered as individual control system. Furthermore, the proposed orchestrator COrch decided the sequencing of operations of the robots while performing obstacle avoidance task for spatially distributed robots in parallel. COrch is used to control this task by utilizing the concept of Remote Method Invocation (RMI) and multithreading. RMI is used to prepare the software for controlling the robots at remote end while multithreading is used to perform parallel and synchronize execution of multiple robots. The remote end software generates signals for sequential ,parallel and hybrid mode execution.

Keywords

Orchestrator, coordination, NCS, collision detection. Multiple Robots

1 Introduction

A distributed system consists of a large number of computing nodes. The nodes interconnect and perform the desired operations. Examples of distributed systems are automation in industries, building automation, automation at home and offices, automation in vehicle systems , in aircraft and spacecraft(A. Selivanov and E. Fridman, 2016; Qinyi Wang and Hongjiu Yang, 2019;Z. Wuet al., 2018; S. Liu et al., 2013;Wang et al., 2008). NCS consists of sensors, actuators and controllers .The operations of these components are distributed over physically spread locations .The coordination of the operations done by sending the information through a communication network. NCSs have many advantages which include reduction in cost, simple system diagnosis, and flexibility, minimum use of wires, simple addition and replacement of individual elements.

NCSs have grown significantly in the recent years and operators face major challenges in maintaining reasonable throughput due to heavy network traffic, heterogeneous technologies and dynamic demands. It also leads to challenges like integration of data, coordination, sequencing of operations and security. These limitations have motivated the network and systems community to develop new paradigms and architectures which improve network infrastructure flexibility, A new control and management system is require which should be capable to *orchestrate* the different technologies and resource types available in modern network infrastructures.

An Orchestrator integrates, manages and coordinates the functioning of multiple components(Agrawal S. and Rajkmal, 2010; Lavnya Ramakrishnan et al., 2011; Escobedo et al., 2010; Peltz, 2003). The Orchestrator can decide the calling sequences of multiple services and can perform controlling action. The Orchestrator manages timeouts, priority and service failures also.(B. S. Heck et al., 2013) gave an overview of software orchestrator for reusable and distributed control systems.

Figure 1 shows an Orchestrator to coordinate the multiple NCSs. Assume that there are two sets of control systems. One set consistsof m control systems and another set consists ofn control systems. The set one is executed at time t_1 and set two is executed at time t_2. All the control systems available in set one and two are parallel initiated systems.

CS represents a control system in Figure 1.

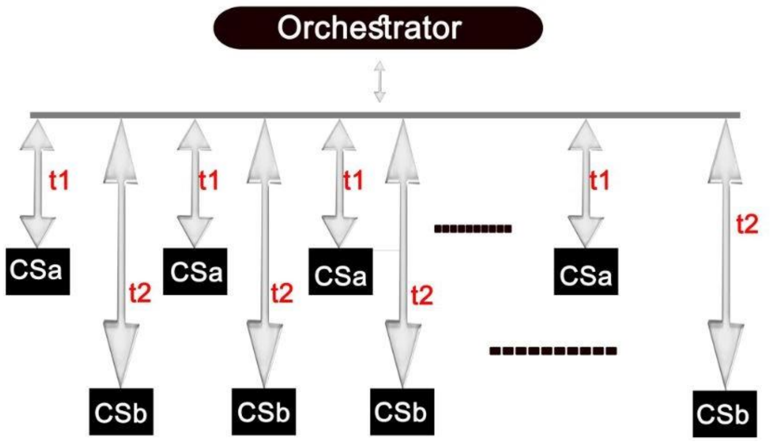


Fig.1. Orchestrator for NCSs

Various researches based on orchestrators in NCS have been done but the earlier proposed orchestrator are designed for the specific tasks and by adapting single execution strategy like sequential or parallel or hybrid execution of sequences. Our objective of this research is to propose, design and implement an orchestrator COrch that has a centralize coordinator for handling multiple robots and can perform sequential or parallel or hybrid execution of robotic tasks in achieving a specific goals.

The contributions of this paper include

1. Design and implementation of COrch a centralized coordinator that can handle multiple roots and perform many of the known robotic execution tasks (sequential or parallel or hybrid execution) with a specific goal.
2. Implementation of COrch has moving robotic vehicles that detect obstacles.
3. Implementation of the orchestrator to decide the calling sequence of robots and ability to send control signals for collision avoidance
4. The proposed orchestrator COrch is a centralized coordinator for handling multiple robots and can perform sequential or parallel or hybrid execution of robotic tasks to achieve a specific goal. The application of COrch shown here is for moving robotic vehicles which can detect obstacles. The orchestrator decides the calling sequence of robots and sends control signals for avoiding collision.

The paper is organized as follows: Section II describes the COrch model, section III describes the various modes of execution, Section IV describes the implementation of COrch and Section V gives the conclusion drawn from the work.

2 Literature Review

Several authors have proposed frameworks and applications of NCSs and Orchestrators. An application of NCS in fire control systems is proposed by (Chen et al., 2007). A NCS for mobilerobot navigation is proposed by (Lee et al., 2013). It is a hierarchical, decentralized NCS that consists of a high-level navigation controlmodule and a remote low-level closed-loop motion control module. An Orchestrator, Dynamic Mission Services Orchestrator (DMSO) is proposed by (H. Paul et al., 2014) . It provides architecture, to orchestrate the application of mission services in both near-real-time and real-time environments. A middleware architecture FTT-CORBA for synchronizing the taskof a distributed system is proposed by (Noguero, A. and Calvo, 2012) .Important examples of middleware currently in use are Java RemoteMethod Invocation (Java RMI), Microsoft Component Object Model (COM) and CommonObject Request Broker Architecture (CORBA) . A precise component definition and an appropriate composition framework to deal with synchronization issues between NCS and networking protocols is proposed by (H. Kopetz and G. Bauer, 2003). An Etherware which is a middleware for NCSs is proposed by (G. Baliga, 2004) . This middleware focuses on the ability to maintain communication channels during component restarts. A ControlWare which is again is a middleware is proposed by (R. Zhang et al., 2002). It utilizes feedback control for guaranteeing performance in software systems An Orchestrator OrchSec for enhancing network security is proposed by (Zaalouk, A.et al., 2014). OrchSec is using network monitoring and SDN control functions. The review on progress in distributed multi-agent coordination is described by (Yongcan C. et al., 2013). A Self-balancing robot by applying Proportional-Derivative Proportional-Integral control system is presented by (C. Iwendiet. al., 2019). Collision avoidance in multiple mobile robots through model predictive control is presented by (R. Mikumo et. al., 2017). Algorithms for avoiding collision in mobile robots which are equipped with IR sensors is presented by (A. M.Alajlanet. al., 2015; Javed, A.R et al., 2020;Mohit Mittal and Celestine Iwendi, 2019). Fuzzy controller based algorithm to avoid collision among mobile robots is presented by (Z. Liu and N. Kubota,2006). Multiple fuzzy rules are created to consider multiple strategies to avoid collision.

A moving robot controlled by computer or tablet for object transportation is presented by (J. H. Lee et al., 2019). A method for trajectory planning of multiple flying robots in decentralized manner is presented by (A. Kosari and M. M. Teshnizi, 2018) .A method for collision avoidance in line follower multiple robots by using IR sensors is presented by (M. M. Almasri et al., 2016; V. Singhal et al., 2020). A formation based methodology to control mobile robots and to avoid large obstacle is presented by (A. Fujimori et al., 2018). The formation is constructed using sonars. A method to dynamically changing the information regarding robot movement by calculating the difference between the best paths of the previous generation and the current iteration through ant colony algorithm is presented by (Z. Yi et al.,2019) . Online trajectory optimization with multiple mobile robots which are operating in close proximity is proposed by (F. G. Lopez et al., 2017). The use of PSO algorithms, to optimize multiple robot path with consideration of obstacle avoidance is presented by (F. Okumuş and A. F. Kocamaz, 2018). Use of neural network and critic neural network to make a guidance strategy for online learning and optimization is presented by (X. Lan et al., 2019). A self-balanced robot is presented by (C. Iwendi et al., 2019)

3 Method

3.1 COrch Modeling

The proposed orchestrator COrch is an Orchestration based software to reduce the software complexity of NCS in coordinating multiple robots. It hides network programming details from the control system designer. COrch treats every module, as a service. Services are running on different nodes. The services are well defined entities that can be replaced without affecting the rest of the systems. They can be developed and tested separately and integrated later. The COrch can interact with services without knowing much of their internal structure. COrch provides an abstraction to the control system user. It hides the complexity of network communication and multi robot control. Multiple robots are having sensors and actuators. The COrch controls the movement of robots to avoid the collision from obstacles. COrch also enables initiation of the movement of robots in sequential, parallel or in hybrid modes. The COrch is utilizing the concept of client server model. Client is playing the role of COrch. The robots are connected to the servers. Clients can communicate with all servers. But servers cannot communicate with each other. Servers have the implementation of called method. Servers also provide the method to connect the robots via serial port. The method calling would be from client side. For controlling the balancing and movement of robots synchronized methods of multithreading in Java have been used. The concept of Java RMI is also used to control the robots from remote end. COrch passes the desired control parameters to the calling method .After receiving the control parameters from COrch, servers execute the method based on those parameters Server is continuously reading the input stream from robot. As soon as server gets any input from sensor, related to obstacle detection, it passes that input to COrch. The COrch again sends the new values for

control parameters and thus avoids collision.

Parameters used in modelling of COrch are:

t_i – initiation time of i_{th} robot

Δt- working time span of any robot (It is considered same for all robots)

n- Number of Robots

Subsections describe each mode of execution in detail with mathematical modeling.

3.2 Sequential Mode of execution:

Figure 3 shows the arrangement of robots for sequential execution. A fixed time pan is allotted to each robot in sequential mode. A number is assigned to every robot. Robot one is invoked first. It does its job of movement and collision detection. Second robot is invoked after completion of time span of robot one. Second robot is controlled in the same manner as was robot one. A time relation is as follows:

$$t_1 < t_2 < t_3 \ldots\ldots\ldots\ldots. < t_n \qquad (1)$$

Time relation (1) shows that t1 is lesser than t2 because robot one would be invoked first and after the completion of time span of robot one, robot two would be invoked at t_2. The process continues till the last one.

$$t_2 = t_1 + \Delta t \qquad (2)$$

and

$$t_n = t_{n-1} + \Delta t \qquad (3)$$

Equation 3 shows the time relation for sequential mode of execution of robots.

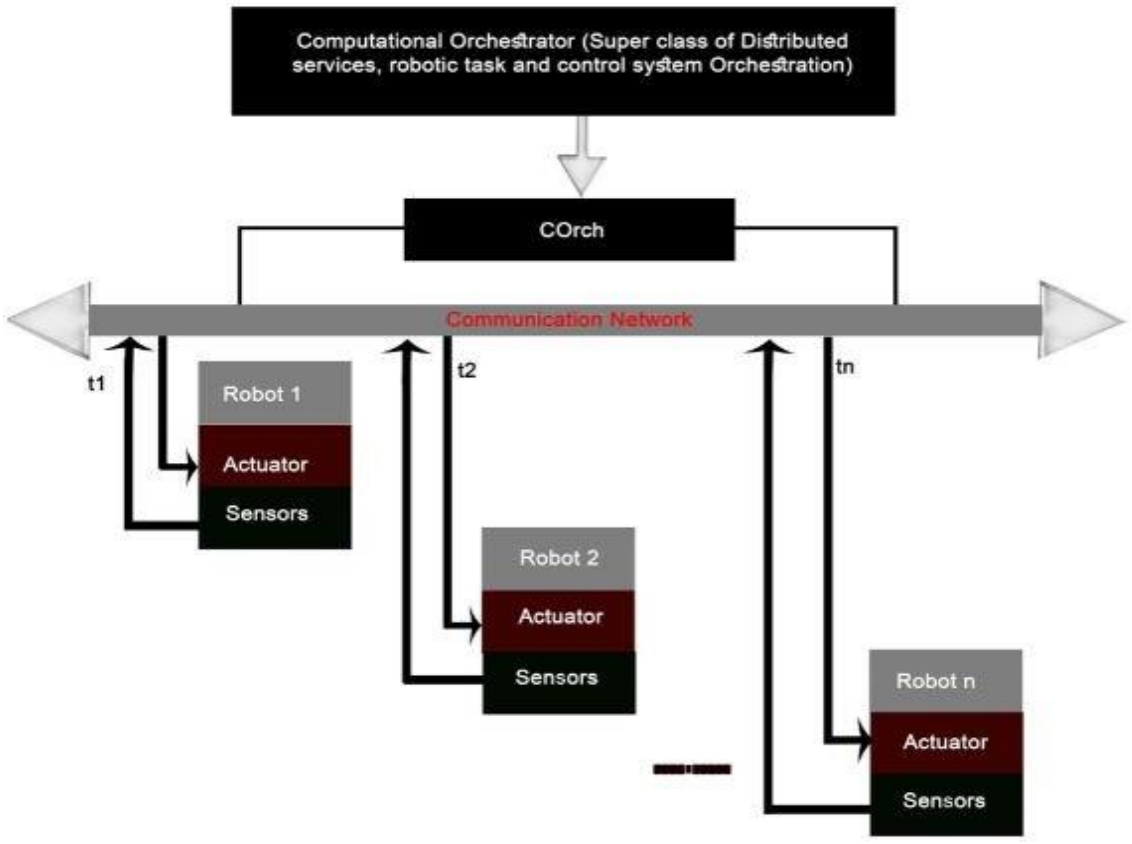

Computational Orchestrator (Super class of Distributed services, robotic task and control system Orchestration)
COrch
Communication Network
t1
t2
tn
Robot 1
Actuator
Sensors
Robot 2
Actuator
Sensors
Robot n
Actuator
Sensors

Fig.3. Sequential Execution in COrch

3.3 Parallel mode of execution

Multiple robots can also be invoked concurrently in parallel execution. Execution of multiple robots is controlled By COrch. All robots start moving concurrently, they avoid obstacles also.

Figure 4 shows the parallel mode execution.Time relation is

$$t_1 = t_2 = t_3 \ldots\ldots\ldots\ldots. = t_n \qquad (4)$$

Time relation 4 shows that movement of all robots is invoked concurrently.

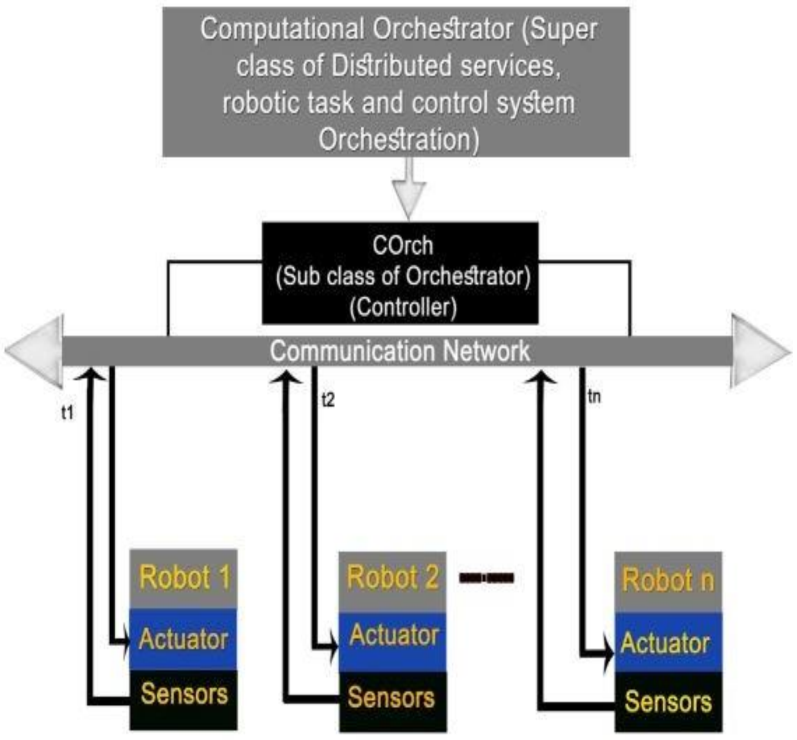

Computational Orchestrator (Super class of Distributed services, robotic task and control system Orchestration)
COrch
(Sub class of Orchestrator)
(Controller)
Communication Network
t1
t2
tn
Robot 1
Actuator
Sensors
Robot 2
Actuator
Sensors
Robot n
Actuator
Sensors

Fig.4: Parallel Execution in COrch

3.4 Hybrid mode of execution

This mode consists of combination of sequential and parallel mode of executions. Out of multiple robots some robots may be invoked in sequential manner and some may be in parallel manner. Consider an example where robot 1 and 2 are invoked to execute in parallel mode and robot 3 and robot 4 are invoked in parallel mode after completion of time span of robot one and two. Figure5 shows hybrid execution for four robots. The time relations for t are as follows:

$$\mathrm{t}_1 = \mathrm{t}_2 \qquad (5)$$

and

$$\mathrm{t}_3 = \mathrm{t}_4 = \mathrm{t}_1 + \Delta\mathrm{t} \qquad (6)$$

or

$$\mathrm{t}_3 = \mathrm{t}_4 = \mathrm{t}_2 + \Delta\mathrm{t} \qquad (7)$$

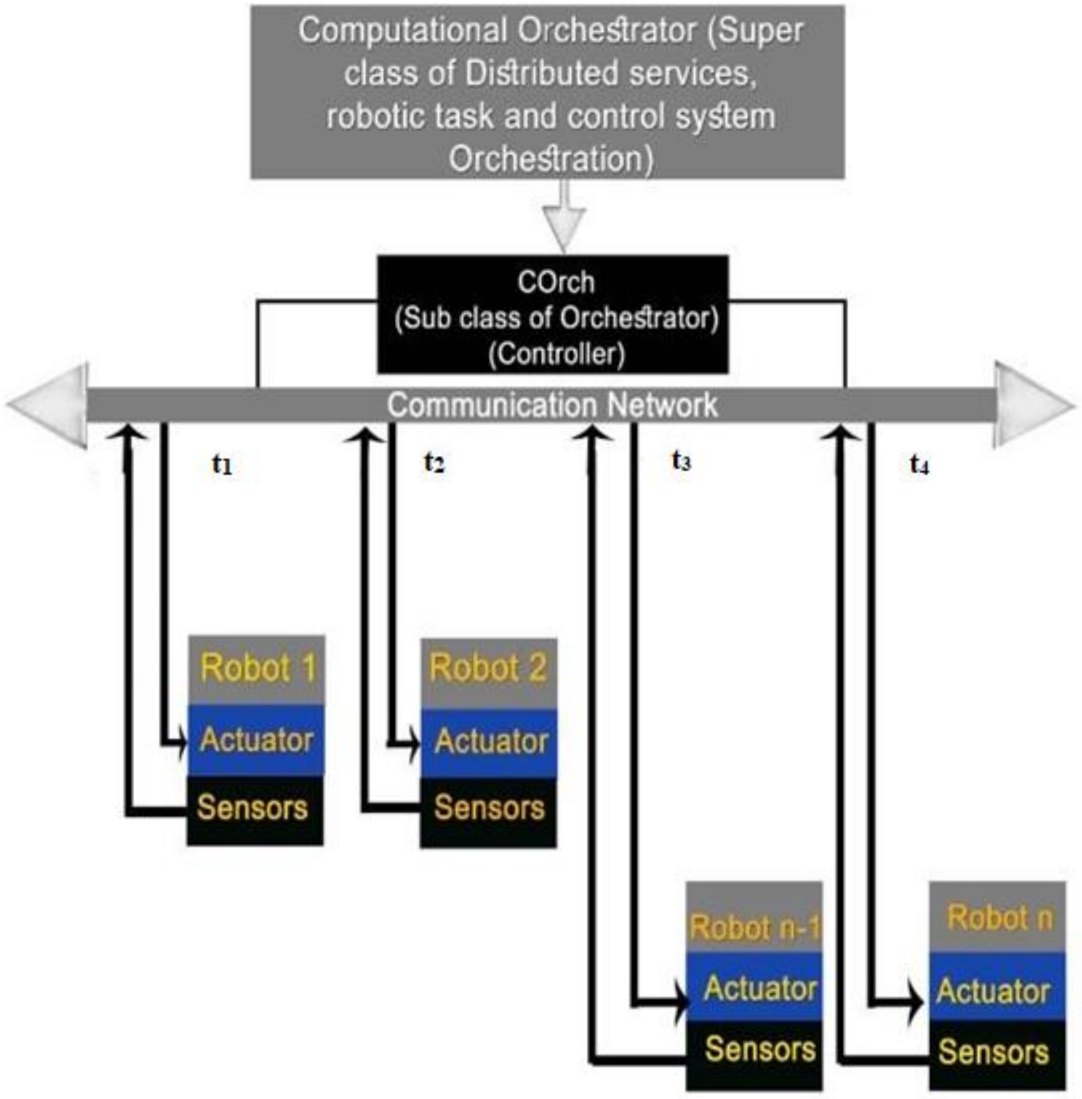

Computational Orchestrator (Super class of Distributed services, robotic task and control system Orchestration)
COrch
(Sub class of Orchestrator)
(Controller)
Communication Network
t1
t2
t3
t4
Robot 1
Actuator
Sensors
Robot 2
Actuator
Sensors
Robot n-1
Actuator
Sensors
Robot n
Actuator
Sensors

Fig.5: Hybrid Execution in COrch

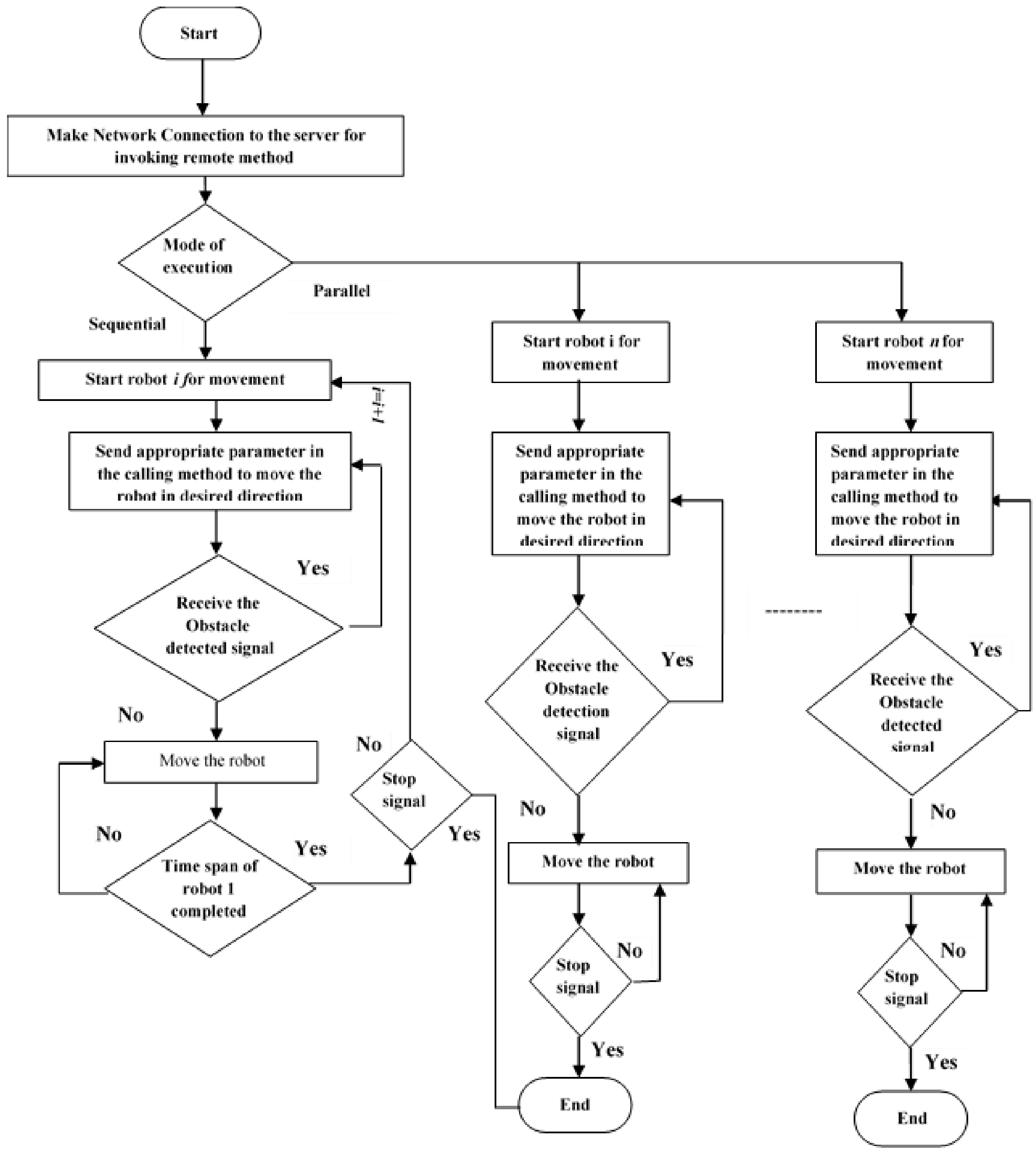
3.5 COrchAlgorithm

```
Assumptions: Number of robots n, time span of each robot $\Delta t$
Initiate n robots in listening mode
Enter choice of execution
1 for sequential mode execution
2 for parallel mode execution
3 for hybrid mode execution

If choice=1
        Send signal to robot 1 to start its movement
        Wait till completion of time span of robot 1 and detect obstacle also
        If "obstacle detected"
                Change the direction of movement
        Else
                Continue movement
        After completion of time span of robot 1 repeat for n robots
```

```
If choice=2
    Send signal to n robots to start their movement
    Wait till completion of time span of n robots and detect obstacle also
    If "obstacle detected"
        Change the direction of movement
    Else
        Continue movement
    Wait till $\Delta t$ time
If choice=3
    Calculate integer value of n/2
    Send signals to n/2 robots to start their movement
    Detect obstacle
    If "obstacle detected"
        Change the direction of movement of robots
    Else
        Continue movement
    Wait for $\Delta t$ time
    Send signal to remaining n/2 robots to start their movement
    Detect obstacle
    If "obstacle detected"
        Change the direction of movement of robots
    Else
        Continue movement
    Wait for $\Delta t$ time
```

The algorithm is explained with the help of flowchart also. Flow chart in Figure 6 shows algorithmic steps of COrch.It shows Corch works on client server model. It works on three modes of execution sequential, parallel and hybrid. Figure 6 is showing two modes sequential and parallel because hybrid mode is the combination of sequential and parallel mode of execution. It also shows the concept of obstacle avoidance that is being implemented in COrch. In flow chart i ranges from 1 to n-1 , where n is the number of robots used


Start
Make Network Connection to the server for invoking remote method
Mode of execution
Parallel
Sequential
Start robot i for movement
i=i+1
Send appropriate parameter in the calling method to move the robot in desired direction
Yes
Receive the Obstacle detected signal
No
Move the robot
No
Time span of robot 1 completed
Yes
No
Stop signal
Yes
Start robot i for movement
Send appropriate parameter in the calling method to move the robot in desired direction
Receive the Obstacle detection signal
Yes
No
Move the robot
Stop signal
No
Yes
End

Start robot n for movement
Send appropriate parameter in the calling method to move the robot in desired direction
Receive the Obstacle detected signal
Yes
No
Move the robot
Stop signal
No
Yes
End

Fig.6: Flow chart for COrch

4 Implementation and Results

The COrchis implemented using Remote Method Invocation (RMI) and serial port communication in Java. RMI is used to invoke the services from remote machines. Multiple threads execute the services in parallel or sequential. Three servers ControlServer1, ControlServer2 and ControlServer3 are developed. Control Client class plays the role of COrch. Threads are developed at ControlClient to call three servers in parallel or in sequential manner. Synchronized methods are used to manage the sequence of operations.

According to user input the COrch decides the sequences of signals which are then transferred to the robots. The experimental setup was on three machines, one client, one local host and two servers were made for implementing the COrch. Robots are equipped with IR sensors. The server method is continuously reading the input stream for any change in sensor data, if there is no obstacle it returns 0 as soon as an obstacle arrives in front of it, it returns a nonzero value, the server passed this value to COrch and COrch changes the direction of movement of robots. Figure 7 is showing the flow of tasks of server which include registration of lookup name in RMI registry, communication with RS-232 port for interacting with robots, and execution of appropriate method based on received signal. Figure 8 shows the class diagram for COrch which describes all the classes and methods used in implementation of COrch Figure 9 (a, b) shows robot, and experimental setup.

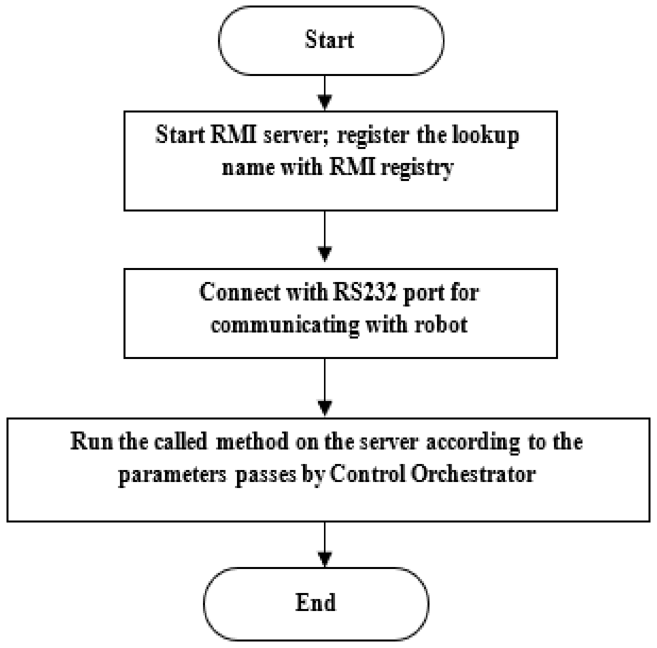

Start
Start RMI server; register the lookup name with RMI registry
Connect with RS232 port for communicating with robot
Run the called method on the server according to the parameters passes by Control Orchestrator
End

Fig. 7: Flowchart for server side

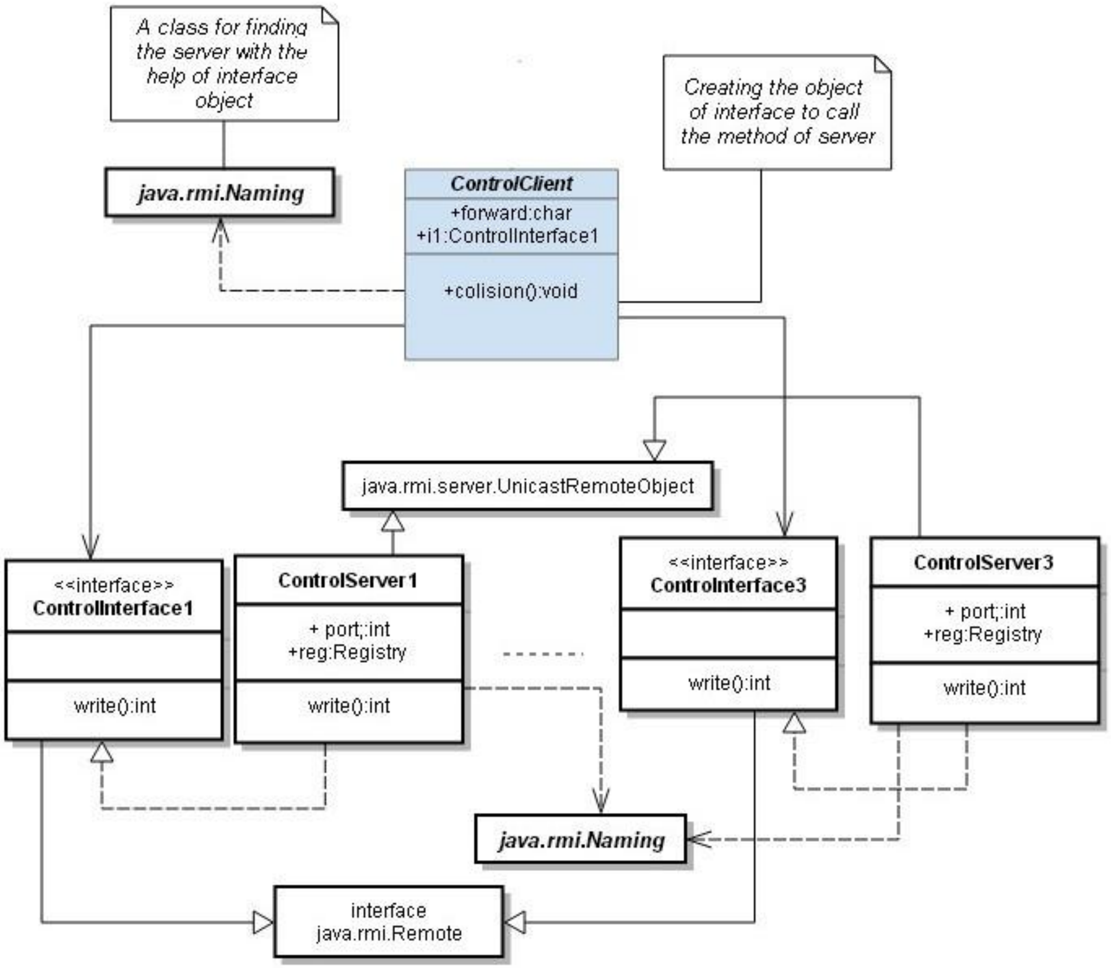
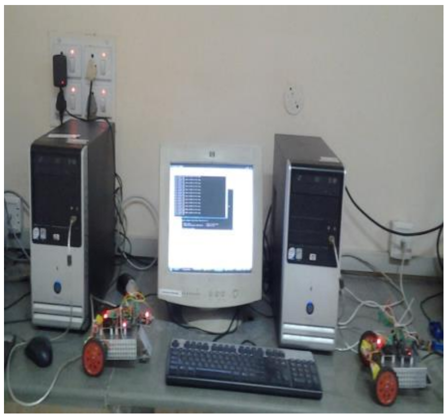


Fig.8: Class diagram for COrch



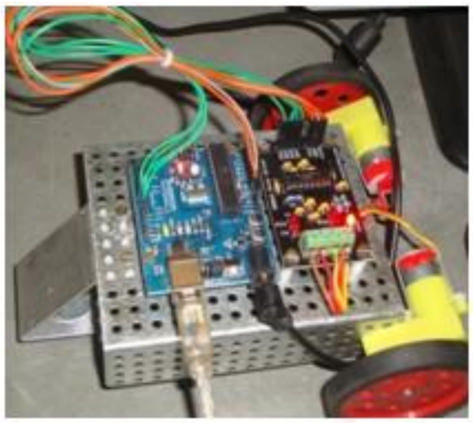


Fig.9: Robot and the experimental setup

Result: Software COrch is developed to demonstrate the coordination in networked control system. COrch is specifically coordinating the sequence of robots in three modes sequential, parallel and hybrid and avoiding obstacles also.

5 Conclusion

A software based orchestration model of COrch for NCSs has been designed and developed. COrch is a centralized coordinator which is managing the execution sequence of multiple vehicle robots and avoiding obstacle by sensing it and sending appropriate signal for change in direction. The COrch is functioning by hiding software complexity from the NCS operator with a unique platform which is providing execution of services in three modes that includes: sequential, parallel and hybrid modes. This paper explores the concepts of RMI, multithreading and the serial port communication of Java. They are implemented for networking, synchronization and communication with robots. In the future, the COrch can be generalized for other publications like process flow of any industry, automation at offices, and travel management only by changing method definitions at server nodes. The coordination done by COrch would be same for other applications also.

References

A. Fujimori, A. Hosono, K. Takahashi and S. Oh-hara (2018) ‘Formation Control of Multiple Mobile Robots with Large Obstacle Avoidance’,*15th International Conference on Control, Automation, Robotics and Vision (ICARCV),* Singapore, pp. 1374-1379.

A. Kosari and M. M. Teshnizi (2018) ‘Optimal Trajectory Design for Conflict Resolution and Collision Avoidance of Flying Robots using Radau-Pseudo Spectral Approach’,*6th RSI International Conference on Robotics and Mechatronics (IcRoM)*, Tehran, Iran, pp. 82-87.

A. M. Alajlan, M. M. Almasri and K. M. Elleithy(2015) ‘Multi-sensor based collision avoidance algorithm for mobile robot’, *Long Island Systems, Applications and Technology,* Farmingdale, NY, pp. 1-6.

A. Selivanov and E. Fridman (2016) ‘Distributed event-triggered control of diffusion semilinear PDEs’,Automatica, vol. 68, pp. 344–351.

Agrawal S., Raj Kamal (2010) ‘A design framework of Orchestrator for computing systems’, IEEE International Conference on Computer Information Systems and Industrial Management Applications (CISIM), 2010, Gwalior, pp. 410- 413.

B. S. Heck, L. M. Wills, and G. J. Vachtsevanos (2003) ‘Software technology for implementing reusable, distributed control systems’, IEEE Control Systems Magazine, Vol. 23, No. 1, pp. 21–35.

C. Iwendi, M. A. Alqarni, J. H. Anajemba, A. S. Alfakeeh, Z. Zhang and A. K. Bashir (2019) ‘Robust Navigational Control of a Two-Wheeled Self-Balancing Robot in a Sensed Environment’, in *IEEE Access,* vol. 7, pp. 82337-82348.

C. Iwendi, M. A. Alqarni, J. H. Anajemba, A. S. Alfakeeh, Z. Zhang and A. K. Bashir, "Robust Navigational Control of a Two-Wheeled Self-Balancing Robot in a Sensed Environment," in *IEEE Access*, vol. 7, pp. 82337-82348, 2019.

Chen, Chen Jie, Zhang Juan (2007) ‘Networked Control System and its Application in Fire Control System’, Control Conference, pp.561, 564.

Escobedo, J. P., Gaston, C., Le Gall, P., &Cavalli, A. (2010) ‘Testing web service orchestrators in context: A symbolic approach’ In 2010 8th IEEE International Conference on Software Engineering and Formal Methods , pp. 257-267.

F. G. Lopez, J. Abbenseth, C. Henkel and S. Dorr (2017), ‘A predictive online path planning and optimization approach for cooperative mobile service robot navigation in industrial applications’, *European Conference on Mobile Robots (ECMR),* Paris, pp. 1-6.

F. Okumuş and A. F. Kocamaz (2018) ‘Comparing Path Planning Algorithms for Multiple Mobile Robots’, *International Conference on Artificial Intelligence and Data Processing (IDAP),* Malatya, Turkey, pp. 1-4.

G. Baliga, S. Graham, L. Sha, and P. R. Kumar (2004) ‘Service continuity in networked control using Etherware’, IEEE Distributed Systems Online, Vol. 5, No. 9,pp. 2-2.

H. Kopetz and G. Bauer(2003) ‘The time-triggered architecture’,Proc. of the IEEE, 91:pp. 112–126.

H. Paul, Clark, Dan,Wisniewski, David(2014) ‘System architecture for Dynamic Mission Services Orchestrator (DMSO)’, 8th Annual IEEESystems Conference (SysCon), pp.259-265.

J. H. Lee, K. Tanaka and S. Okamoto (2019) ‘Collision-Free Navigation Using Laser Scanner and Tablet Computer for an Omni-Directional Mobile Robot System with Active Casters’,*2019 International Conference on Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO)*, Athens, Greece, pp. 31-36.

A. R. Javed, M. U. Sarwar, , S. Khan, C. Iwendi, M. Mittal, N. Kumar (2020) ‘Analysing the Effectiveness and Contribution of Each Axis of Tri-Axial Accelerometer Sensor for Accurate Activity Recognition’, Sensors 2020, Issue 20, no. 8,pp. 2216

L. Ramakrishnan, S. Jeffrey, Chase, Dennis Gannon, Daniel Nurmi, Rich Wolski (2011) ‘Deadline-sensitive workflow orchestration without explicit resource control’, Elsevier J. Parallel and Distributed Computing Vol. 71, No. 3, pp. 343-353.

Lee, M.F.R., Chiu, F.H.S. (2013) ‘A networked intelligent control system for the mobile robot navigation’, IEEE/SICE International Symposium on System Integration (SII), pp.42-47.

M. M. Almasri, A. M. Alajlan and K. M. Elleithy (2016) ‘Trajectory Planning and Collision Avoidance Algorithm for Mobile Robotics System’, IEEE Sensors Journal, Vol. 16, No. 12, pp. 5021-5028.

M. Mittal and C.Iwendi(2019) ‘A Survey on Energy-Aware Wireless Sensor Routing Protocols’, EAI Endorsed Transactions on Energy Web Vol. 6 No. 24

Noguero, A.,Calvo, I. (2012) ‘A time-triggered data distribution service for FTT-CORBA’, IEEE 17th Conference on Emerging Technologies & Factory Automation (ETFA) , pp.1-8.

Peltz, C.(2003) ‘Web services orchestration and choreography’, Computer, Vol.36, No.10, pp.46- 52.

W. Qinyi and Hongjiu Yang (2019) ‘A survey on the recent development of securing the networked control systems, Systems Science & Control Engineering’, Vol. 7, No. 1, pp. 54-64.

R. Mikumo and H. Ichihara (2017) ‘Dynamic collision avoidance among multiple mobile robots: A model predictive control approach’, *56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE),* Kanazawa, pp. 1136-1137

S. Liu, P. Liu and A. E. Saddik (2013) ‘A stochastic security game for Kalman filtering in networked control systems under denial of service attacks', 3rd IFAC International Conference on Intelligent Control and Automation Science, Vol. 46, No. 20, pp. 106–111.

V. Singhal, S. Jain, D. Anand, A. Singh, S.Verma, J. JPC Rodrigues, N.ZamanJhanjhi, U.Ghosh, O. Jo, and C. Iwendi (2020) 'Artificial Intelligence Enabled Road Vehicle-Train Collision Risk Assessment Framework for Unmanned Railway Level Crossings' IEEE Access Vol. 8, pp. 113790-113806.

Wang, Fei-Yue, Liu, Derong (2008), 'Networked Control Systems Theory and Applications' Springer publication.

X. Lan, C. Qin, Y. Liu, H. Ouyang and G. Liu (2019) 'Intelligent guidance of autonomous mobile robots based on adaptive dynamic programming'*34rd Youth Academic Annual Conference of Chinese Association of Automation (YAC),* Jinzhou, China, pp. 695-699.

Yongcan Cao, Wenwu Yu, Wei Ren, Guanrong Chen (2013) 'An Overview of Recent Progress in the Study of Distributed Multi-Agent Coordination' IEEE Transactions on Industrial Informatics, , Vol.9, Issue 1, pp.427-438.

Z. Liu and N. Kubota (2006) 'Hybrid Learning Approach for the Collision Avoidance Behavior of a Mobile Robot' *International Conference on Mechatronics and Automation,* Luoyang, Henan, pp. 19-24.

Z. Wu, Y. Xu, Y. Pan, H. Su and Y. Tang (2018) 'Event-triggered control for consensus problem in multi-agent systems with quantized relative state measurement and external disturbance',IEEE Transactions on Circuit and Systems, Vol. 65, No. 7, pp. 2232-2242

Z. Yi, Z. Yanan and L. Xiangde (2019) 'Path Planning of Multiple Industrial Mobile Robots Based on Ant Colony Algorithm', 16th International Computer Conference on Wavelet Active Media Technology and Information Processing, Chengdu, China, pp. 406-409.

A. Zaalouk , R. Khondoker , R. Marx , K. Bayarou(2014) 'An orchestrator-based architecture for enhancing network-security using Network Monitoring and SDN Control functions', IEEE Network Operations and Management Symposium (NOMS), 2014 , pp.1-9, 5-9 May 2014

Zhang, R., Lu, C., Abdelzaher, T. F., &Stankovic, J. A. (2002) 'Controlware: A middleware architecture for feedback control of software performance' In Proceedings 22nd International Conference on Distributed Computing Systems , pp. 301-310.