Modeling Supervisory Control of Autonomous Mobile Robots using Graph Theory, Automata and Z Notation

Javed Iqbal[1], Sher Afzal Khan[2], Nazir Ahmad Zafar[3] and Farooq Ahmad[1]

[1]Faculty of Information Technology, University of Central Punjab, Lahore, Pakistan
[2]Department of Computer sciences, Abdul Wali Khan University, Mardan, Pakistan
[3]Department of Computer Science, King Faisal University, Al Hassa, Saudi Arabia
javedsamon@ucp.edu.pk; sher.afzal@awkum.edu.pk;drfarooq@ucp.edu.pk;nazafar@kfu.edu.sa;gafzal@cae.nust.edu.pk

Abstract: Supervisory control of the mobile robot navigation system has critical importance. The supervisory control software development of mobile robot navigation can be performed in an unknown environment or for controlled robots in a known environment. Finite automata and graph theory are functional tools in modeling the robot navigation system through discrete environment. This research has emphasis on an integration of graph theory, automata and Z notation for modeling supervisory control of robot navigation system. The design of robot blocked, not blocked and its supervisory control is developed using automata, in which the states are represented by nodes (rooms) and transitions (stairs and doors) by directed edges. In this paper, the integration of approaches as an effective tool for modeling is investigated by using Z/Eves.
[Iqbal J, Khan SA, Zafar NA, Ahmad F and Khan GA. **Modeling Supervisory Control of Autonomous Mobile Robots using Graph Theory, Automata and Z Notation .** *J Am Sci* 2012;8(12):799-804]. (ISSN: 1545-1003). http://www.jofamericanscience.org. 111

Keywords: Autonomous Mobile Robot; Supervisory Control; Integration of Approaches; Graph Theory; Automata and Z-notation

1. Introduction

Mobile robots navigation has a broad range of applications and mobile robot navigation system can be modeled either for autonomous in an unknown environment or for controlled robots in a known environment. Making progress toward autonomous robots is of major practical interest in a wide variety of application areas.

Fundamental quality of any mobile robot is its capability to navigate from one location to another. Navigation is an ability to identify its own position and then navigate towards the destination in an environment. To achieve the quality of navigation, different approaches have been proposed in the literature.

The mobile robot navigation system has attracted a much attention of many researchers and different navigation methodologies have been proposed. Behavior based navigation is developed using dynamic systems theory to generate the behaviors in [1]. The control of the navigation of a robot has been modeled by using Petri nets in unstructured environments [2]. A model to guide and keep track of a robot such that it is able to complete several tasks is described in [3]. Now the existing navigation of single robot has extended to multiple robots system. Further, simultaneous navigation of multiple robots in a common environment have raised a new and challenging problem. The navigation of multiple robots addressing the issues of cooperation and formation control has been addressed in [4] and [5].

The robot navigation problem has attracted considerable interest in recent years for movement through both discrete and a continuous environments [6],[7]. Melo et al. analyzed the problem of driving a robot population moving in a discrete environment from some initial to a target configuration [7]. The authors further examined the problem of multi-robot navigation. Robot navigation modeling in discrete environment has been performed by using finite automata and graph theory [8][9]. Finite automata have been used to control the navigation of mobile robot along the possible paths in [10].

In this paper, specification-based approach is developed by integrating graph theory, finite automata and Z notation. This integrated approach is applied to model mobile robot navigation and its supervisory control. We examine the problem of assigning a heterogeneous population of rescue robots to reach a set of target rooms where a set of people may be trapped. The robots in the population are assumed to have the properties such that these allowed transitions are different and therefore the resultant automata and directed graph models for each robot are also different.

A number of modeling techniques for mobile robot navigation have been developed by researchers such as partially observable Markov and behavior based navigation but the existing approaches have lack of the formalization. In this

paper, the integration of graph theory and automata is used for modeling of mobile robot navigation system. Z notation is used for formal description of the system and Z/Eves tool is used for analyzing and validating the formal models. Rest of the paper is organized as follow. In section 2, robot navigation problem, graph theory and automata are described. In section 3, an introduction to formal method is provided. In section 4, Formal modeling of robot navigation system using integrating of approaches is described. Finally, conclusion and future work are discussed in section 5.

2. Robot Navigation problem, Graph Theory And Automata

A mobile robot is developed and programmed to accomplish a mission which has to move to the target position. The problem implies that a robot has to localize the target. It requires a world representation commonly defined as a map where the robot identifies a target and has to estimate its current position by localizing the problem. The map could be complete, which addresses the map building problem. Finally, to accomplish the mission, the robot has to move to reach the target goal, selecting the best trajectory.

Graph Theory

Graph theory is an important field of mathematics because of its various applications in different areas: biochemistry, electrical engineering, operations research and computer science. Graph theory can be applied to any problem that has configurations of nodes and connections, such as electrical circuits, roadways or organic molecules.

A graph G = (V, E) consists of a finite nonempty set of objects called vertices V and a set of order pairs called edges E. The sets V and E are the vertex and edge sets of G respectively. A graph is often represented as a diagram with the vertices represented by dots and the edges represented by lines joining those vertices. An example of a graph G = (V, E) is shown in Figure 1 which consists of the vertex-set V = {1, 2, …, 9} and edges E = {(1, 2), (1, 3), (2, 5), (2, 9), (3, 4), (4, 5), (5, 6), (6, 7), (7, 9)}.

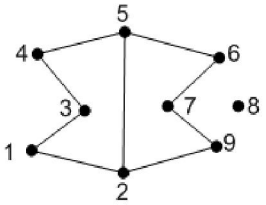

Figure 1: An example of a graph

Automata Theory

Automata theory deals with the theoretical models of computer known as abstract machines. These machines are also known as automata which can be deterministic or deterministic and are excellent way of modeling finite-state systems. These automata can be used to change the state according to an input is given.

Finite Automata

A finite state automaton represents a problem as a series of states with transitions between the states caused by an input to it. The automaton starts from an initial state and an input causes a transition from the current to the next state which may be an accepting state in which the automaton can terminate successfully [12].

A deterministic finite automaton consists of a finite set of states denoted Q, finite set of input symbols denoted Σ, transition function that takes in its argument a state and an input symbol and returns the same or a new state. The transition function is commonly denoted by δ. The start state is denoted by q0 which is one of the states in Q. The set of final or accepting states is denoted by F which is a subset of Q. A deterministic finite automaton will be denoted by DFA. The shortest representation of a DFA is a list of the five components as defined above, DFA = (Q, Σ, δ, q0, F).

- Q is a finite set of states.
- Σ is a finite set of legal input symbols which is also called the alphabet of the automaton.
- δ: Q × Σ → Q is the transition function which takes a state from Q and symbol from Σ as input and returns a state from Q as an output.
- F is a set of final states also called the accepting states and subset of Q, i.e., F ⊆ Q.

Navigation Automata

The navigation automaton is a generalization of the deterministic finite automaton in which the alphabet is replaced by an event set and an extra active event set is introduced. The navigation automaton can be defined by a six-tuple, G = (Y, E, f, Γ, Y0, Ym) where

- Y is the state space.
- E is the set of possible events.
- f : D → Y, D ζ Y × E are the transition functions.
- Γ: Y → 2E is the active event function. Y0 is the initial state.
- Ym is the set of marked or accepting states or also called the target states.

3. Formal Methods

Formal methods can provide a complete, consistent and correct model of a system. Formal methods can be used at any stage of systems development, for removing ambiguities, inconsistencies and incompleteness of a system [17][18][19][20][21][22][23][24]. Formal methods provide verification of the modeled system and can remove the flaws at early stages of system development. If such errors are discovered at later stages, debugging and testing, of a system it will result costly treatment [13]. Further, correctness of a system can also be verified by using formal method at the verification phase [14]. Huge vocabulary of natural language has multi-meanings which make the specifications highly ambiguous [15]. Syntax and semantics are well defined in the formal specification. Formal specification is consistent, concise and it can prove the properties of a system without implementation. The use of formal method can be seen from its application [25][26][27][28][29].

The requirements engineering is most important step in high quality software development process. The architectural structure of a complex software system and their components require careful management. Every software system implicitly uses a theorem that if some condition are satisfied or not to accomplish the requirement of a software system thus there is need of code verification. Code verification is the attempt to prove these theorems or finding the reasons why this theorem failed to achieve the desired properties of a system. Therefore, formal method are useful and required for capturing the requirements, architectural structure, model checking, theorem proving and code verification [16].

4. Formal Modeling of Supervisor Control

The design of mobile robot navigation system requires modeling its environment, supervisory control behavior and complete functionality. Environment of mobile robots is modeled using graphs. Automata theory is used for modeling the control behavior. Functions of a system are decomposed into the operations and constraints which are properly captured in the Z notation. In this way, an integration of graph theory, automata and Z notation is used for modeling of mobile robot navigation system. To specify the supervisory control, we first describe the declaration part for connectivity, environment, and directed path and navigation automata which are using in our further formal specification.

$[Node]$

$Edge == Node \times Node$

Connectivity

$nodes: \mathbb{P}\ Node$

$edges: \mathbb{P}\ Edge$

DirectedGraph

$\Xi\ Connectivity$

Environment

$\Xi DirectedGraph$

$EdgeTypes: \mathbb{P}\ (\mathbb{P}\ Edge)$

AMovementofEachRobot

$\Xi Environment$

$properties: \mathbb{P}\ P,\ robots: \mathbb{P}\ Robot$

$probot: Robot \rightarrow \mathbb{P}\ P,\ amovement: Robot \rightarrow \mathbb{P}\ Edge$

Automata

$astates: \mathbb{P}\ Q$

$alphabets: \mathbb{P}\ X$

$atrans: Q \times X \rightarrow Q$

$q0: Q,\ \ qf: \mathbb{P}\ Q$

Specification of Navigation Automata

NavigationAutomata

$states: \mathbb{P}\ Y$

$events: \mathbb{P}\ E$

$f: Y \times E \rightarrow Y$

$possibleaction: Y \rightarrow \mathbb{P}\ E$

$y0: Y,\ Ym: \mathbb{P}\ Y$

GraphNAutomata

$\Xi Environment$

$states: \mathbb{P}\ Node,\ events: \mathbb{P}\ Edge$

$f: Node \times Edge \rightarrow Node$

$possibleaction: Node \rightarrow \mathbb{P}\ Edge$

$y0: Node$

$Ym: \mathbb{P}\ Node$

GraphNAutomataForAllRobot

$\Xi AMovementofEachRobot$

$\Delta GraphNAutomata,\ robot?: Robot$

Description of Path

A walk in a graph is a finite sequence of alternating nodes and edges or it can be defined by only finite sequence of edges. A path is a finite

sequence of edges from starting node to the final node with distinct nodes. The variable walk is introduced to define the sequence of nodes from n1? to n2?, where n2? is the final node. Moreover, n1? and n2? are input variables of type Node which is used to check that either a walk exits in between n1? and n2?.

Path

Ξ *DirectedGraph*
walk: seq *Node*
n1?: Node
n2?: Node

SuccessfulPathofRobot

Ξ*GraphNAutomataForAllRobot*
Ξ*Path*
navigate?: seq *Edge*

Specification of All Successful Path of Robot

AllSuccessfulPathofRobot

Ξ*GraphNAutomataForAllRobot*
Ξ*Path*
navigates?: $\mathbb{P}$ (seq *Edge*)

Design of Robot Blocked

The AllSuccessfulPahtofRobot defined by navigation automata can be used to define the robot blocked. Again the declaration AllSuccessfulPahtofRobot indicates that the schema is describing a state change. The input variable start? and target? Are introduced of the type Node and power set of Node respectively.

RobotBlocked

Δ *AllSuccessfulPahtofRobot*
start?: Node
target?: $\mathbb{P}$ *Node*

$start? \in nodes$
$target? \subseteq nodes$
$y0' = start?$
$Ym' = target?$
$navigates? = \emptyset$

Invariants:

- The variable start? is in the set nodes.
- The target? is the subset of a set nodes.
- The y0' is equal to the start?.
- The Ym' is equal to the target?.
- The navigates? is empty.
- Design of Robot Blocked and Controllable

- The AllSuccessfulPahtofRobot defined by navigation automata can be used to define the robot blocked and controlable. Again in this schema the declaration of óAllSuccessfulPahtofRobot indicates that the schema is describing a state change. The input variable start?, target? and block are introduced of the type Node, power set of Node and Node respectively.

RobotBlockedandControlable

Δ *AllSuccessfulPahtofRobot*
start?: Node
target?: $\mathbb{P}$*Node*
block: Node

$start? \in nodes$
$target? \subseteq nodes$
$block \in nodes$
$y0' = start?$
$\wedge\ Ym' = target?$
$\wedge\ navigates? \neq \emptyset$
$\wedge\ (\exists b: Node \mid b = block$
$\bullet\ y0' = b$
$\wedge\ Ym' = target?$
$\wedge\ navigates? = \emptyset$
$\wedge\ y0' = start? \wedge\ Ym' = \{b\} \wedge navigates? \neq \emptyset)))$

Invariants:

- The variable start? is in the set nodes.
- The target? is the subset of a set nodes.
- The block is in the set nodes.
- The y0' is equal to the start?, Ym' is equal to the target?, The navigates? is not equal to empty. Then there exists b of type Node equal to block, y0' is equal to b, Ym' is equal to the target? and navigates? is empty and y0' is equal to start?, Ym' is equal to {b} and navigates? is not equal to empty.
-

Design of Robot Not Blocked

In RobotNotBlocked schema the declaration of AllSuccessfulPahtofRobot indicates that the schema is describing a state change. The input variable start?, target? and block are introduced of the type Node, power set of Node and Node respectively.

Invariants:

- The variable start? is in the set nodes.
- The target? is the subset of a set nodes.
- The block is in the set nodes.
- The y0' is equal to the start?, Ym' is equal to the target?, The navigates? is not equal to empty. Then there does not exist b of type Node equal to block, y0' is equal to b, Ym' is

equal to the target? and navigates? is empty and y0' is equal to start?, Ym' is equal to the {b} and navigates? is not equal to empty.

RobotNotBlocked

Δ *AllSuccessfulPahtofRobot*
start?: Node
target?: ℙNode
block: Node

$start? \in nodes$
$target? \subseteq nodes$
$block \in nodes$
$y0' = start?$
$\land\ Ym' = target?$
$\land\ navigates? \neq \varnothing$
$\land \exists b: Node \mid b = block \bullet y0' = b$
$\land\ Ym' = target?$
$\land\ navigates? = \varnothing$
$\land y0' = start? \land Ym' = \{b\} \land navigates? \neq \varnothing)))$

Design of Supervisory Control

It is not required that a system reaches a blocked state where robot unable to reach the final state, this situation can be prevented by supervisory control. In the schema SupervisoryControl the specification of RobotBlockedandControlable is reused.

SupervisoryControl

Δ*RobotBlockedandControlable*

$navigates? \neq \varnothing$
$\land \exists b: Node \mid b = block \bullet y0' = b$
$\land\ Ym' = target?$
$\land\ navigates? = \varnothing$
$\land y0' = start? \land Ym' = \{b\} \land navigates? \neq \varnothing)$
$\land \forall n: Node \mid n \in nodes \bullet events' = events \mid \{(n, b)\})))$

5. Conclusion

In this paper, we have presented an integrated approach for modeling of supervisory control of mobile robot navigation system using graph theory, automata and Z notation. Automata and graph theory have proved to be useful tools in modeling the robot navigation system. Using automata, an automatic movement of robot is described in which the rooms are assumed as states and passageways, stairs and doors are taken as transitions. It is further assumed that a robot travels from a start state to a specified final state by a sequence of connected edges. Graph being a special case of automata is used to model and analyze the paths from start state to the final states. The Z/Eves tool is used to investigate and validate the formal models produced by describing mobile robot navigation system using above integration. The population of mobile robots have different properties to perform action in the environment, the movements of each robot is modeled accordingly. The graph navigation automata have been applied for all robots to model the allowed movement of each robot. Moreover, successful path to navigate from a specified start state to a set of target states was described. The integrated approach supported us to model the robot environment, its control behavior and functionality by capturing both its static and dynamics parts from an abstract to a detailed level of specification.

References

[1] R. Simmons and S. Koenig, "Probabilistic robot navigation in partially observable environments," Proceedings of the International Joint Conference on Artificial Intelligence, pp. 1080–1087, 1995.

[2] R. D. Hale, M. Rokonuzzaman and R. G. Gosine, "Control of mobile robots in unstructured environments using discrete event modeling," SPIE International Symposium on Intelligent Systems and Advanced Manufacturing, 1999.

[3] A.Steinhage, "Dynamical systems for the generation of navigation behaviour," PhD dissertation, Bochum Germany, 1997.

[4] T. Balch and M. Hybinette, "Social potentials for scalable multi robot formations," IEEE International Conference on Robotics and Automation (ICRA-2000), pp. 73–80, 2000.

[5] O. Hachour and N. Mastorakism, "Intelligent Control and planning of IAR," 3rd International Multiconfrence on System Science and Engineering, 2004.

[6] J. Canny, "The complexity of robot motion planning," Cambridge, MIT Press, 1988.

[7] F. A. Melo, M. R. Isabel and P. Lima, "Event-driven modeling and control of a mobile robot population," Proceedings of the 8th Conference on Intelligent Autonomous Systems, pp. 237-244, 2004.

[8] A. Biran, and M. Breiner, "MATLAB for engineers," Addison-Wesley Longman Publishing, 1997.

[9] J. L Gross and J. Yellen, "Handbook of graph theory," 2nd Edition, CRC Press, 2005.

[10] N. A. Zafar, N. Sabir and A. Ali, "Construction of intersection of nondeterministic finite
[11] World Academy of Science, Engineering and Technology, 2008, vol. 30, pp. 96-101.
[12] J. E., Hopcropft, R. Motwani and J. D. Ullman, "Introduction to automata theory, language and computation," Addison-Wesley, 2001.
[13] J. M. Wing, "A specifier's introduction to formal methods," IEEE Computer, vol. 23(9), pp. 8-24, 1990.
[14] J. Seung-Ju, R. Jungwoo and L.Y. Chang, "Design of software security verification with formal method tools," International Journal of Computer and Network Security, vol. 6, 2006.
[15] R. S. Pressman, "Software engineering a practitioner's approach," 5th Edition, McGraw-Hill, 2001.
[16] V. George and R. Vaughn, "Application of lightweight formal methods in requirement engineering," The Journal of Defense Software Engineering, 2003.
[17] Ahmad, F. and S. A. Khan (2012). "Module-based architecture for a periodic job-shop scheduling problem." Computers & Mathematics with Applications.
[18] Ali, G., S. A. Khan, et al. (2012). "Formal modeling towards a dynamic organization of multi-agent systems using communicating X-machine and Z-notation." Indian Journal of Science and Technology 5(7).
[19] Khan, S. A., A. A. Hashmi, et al. (2012). "Semantic Web Specification using Z-Notation." Life Science Journal 9(4).
[20] Khan, S. A. and N. A. Zafar (2007). "Promotion of local to global operation in train control system." Journal of Digital Information Management 5(4): 231.

8/12/2012

automata using Z notation," Proceedings of
[21] Khan, S. A. and N. A. Zafar (2009). Towards the formalization of railway interlocking system using Z-notations, IEEE.
[22] Khan, S. A. and N. A. Zafar (2011). "Improving moving block railway system using fuzzy multi-agent specification language." Int. J. Innov. Computing, Inform. Control 7(7).
[23] Khan, S. A., N. A. Zafar, et al. (2011). "Extending promotion to operate controller based on train's operation." International J. Phy. Sci 6(31): 7262 - 7270.
[24] Khan, S. A., N. A. Zafar, et al. (2011). "Petri net modeling of railway crossing system using fuzzy brakes." International J. Phy. Sci 6(14): 3389-3397.
[25] M, F. and S. A. Khan (2012). "Specification and Verification of Safety Properties along a Crossing Region in a Railway Network Control." Applied Mathematical Modelling, 10.1016/j.apm.2012.10.047.
[26] Raza, M. I., Q. J. Zaib, et al. (2012). "Meticulous analysis of Semantic Data Model -An optimal approach for ERD." J. Basic. Appl. Sci. Res. 8(2): 8344-8354.
[27] Yousaf, S., N. A. Zafar, et al. (2010). Formal analysis of departure procedure of air traffic control system, IEEE.
[28] Zafar, N. A., S. A. Khan, et al. (2012). "Formal Modeling towards the Context Free Grammar." Life Science Journal 9(4).
[29] Zafar, N. A., S. A. Khan, et al. (2012). "Towards the safety properties of moving block railway interlocking system." Int. J. Innovative Comput., Info & Control.