Conference Paper

A Spatial and Operations Modeling Method for Automatic Parking Systems

Ramiro C. Advincula and Aaron Vito M. Baygan

Polytechnic University of the Philippines, Parañaque City Campus

Abstract

Automatic or Automated Parking Systems (APSs) would have spatial configurations that depend on both or any of the following factors: 1) its operations and 2) the physical structure design. Without a modeling method, designing and configuring an APS could be relatively challenging and result to special instances of APSs. On the other hand, a modeling method for this purpose would facilitate representing the spaces and operations used by APSs thus aid their design and configuration, and possibly make them adaptive to physical space constraints. This study developed such method for modeling the spaces and operations of APSs allowing their design and configuration to be highly flexible. It involved defining an approach for spatial representations and establishing a model for representing the operations of autonomous parking devices of APSs. The implementation of the spatial representations and operations model into a data structure suitable for computer programming was also described. A number of configuration examples based on those offered by current APS service providers and a few hypothetical APS designs were used to test the applicability of the method. The test was facilitated by simulation software that allowed input of varied APS configurations, input of basic car parking and retrieval operations, and showing results of such operations. Results show that basic operations are correctly executed thus indicating that the model is applicable.

Keywords: modeling method, Automatic Parking Systems (APS), autonomous parking devices, spatial model, parking device operations model

Corresponding Author:
Ramiro C. Advincula
arcadvincula@pup.edu.ph

Received: 23 April 2018
Accepted: 8 May 2018
Published: 4 June 2018

Publishing services provided by Knowledge E

Selection and Peer-review under the responsibility of the IRCHE 2017 Conference Committee.

1. Introduction

Current Automatic or Automated Parking Systems (APSs) would have specific *spatial configurations* for the *spaces* or the parking slots and areas used for delivering the cars into and out of the parking slots [3, 9]. They also employ varied computerized or *autonomous devices* that would execute said delivery of cars [7, 15]. These spatial configurations either depend on both or any of the following factors: 1) its *operations*—how the APS delivers the cars into and out of the parking slots and 2) the *physical*

structure design— the shape of its parking structure as may be constrained by shape of the lot or other structural requirements [1][3, 9]. Without a modeling method, designing an APS, and even configuring it later, becomes very challenging and results to specialized instances of APSs [1][3, 9]. It's for this reason that existing APS designs would be categorized as either a vertical, horizontal, circular or puzzle type [2], [7]-[9], [12]-[14]. On the other hand, a modeling method for this purpose would facilitate representing the spaces used by the APS and provide high degree of abstraction about its operations and physical structure design thus allow a flexible way of designing or configuring APSs [3, 8, 9]. The availability of a modeling method could further facilitate the computer programming implementation of the spaces as an applicable *data structure* thereby allowing also direct usage of relevant algorithms that are readily available or those that could be derived from existing ones for operating the APS [4].

2. Objectives

The main objective of this work is to establish a method for creating spatial and operations models that would allow flexibility to designing and configuring APSs.

The following are its specific objectives:

1. Describe an APS Spatial Model. This will establish an approach for representing the spaces in an APS.

2. Describe an APS Operations Model. This is stablishing a model for representing robotic parking devices and their corresponding operations.

3. Identify an Applicable Data Structure. This will entail describing and demonstrating the use of a data structure suitable for computer programming implementation of both the spatial and operations models.

4. Test the Method. This will be focused on demonstrating the method's applicability and the flexibility it provides to APS design and configuration.

2.1. Scope and limitations

1. The study is focused on fully automatic or automated parking systems, i.e. those that use only autonomous or robotic parking devices that delivering a car from a receiving area into a parking slot, then to a re-claiming area from a parking slot.

2. The use of pallets and operations for managing them were assumed integrateable with the other functions involving them thus were not considered in this study.
3. The algorithm used for operating the APS would be only for demonstrating basic and instantaneous operations. It assumes that the spaces being accessed are vacant thus does not include tests for simultaneous operations, parking and retrieval strategies, nor optimality.

2.2. Significance of the study

The study offers to standardize the designing and configuring processes for APSs at least in the spatial and operations aspects. It also provides a generalized APS model where identifying a better algorithm or set of algorithms for executing parking activities of an APS could be facilitated which in turn could lead to the creation of a better and generic control or operating system for APSs.

Studies that focus on the computer network for APSs may find the work relevant for identifying approaches how data communications may be done or improved depending on the algorithm identified for manipulating the devices.

On a higher perspective, it may serve as basis for modelling any system dealing with material flow, or could be directly applied to facilities that handle material storage and retrieval through use of robotic devices,.e.g. Automated storage and retrieval systems (AS/RSs).

3. Methods

3.1. The APS spatial model

To create a spatial model of the APS, the spaces used should be viewed as rectangular prism units, identified per unit according to their usages, and appropriate labels are used to aid their identification.

The following spaces are identified as the typical spaces used by an APS and corresponding labels were suggested:

1. Slot – (slot) a space where a car is parked
2. Horizontal Path – (hPath) a space used by devices to convey cars horizontally
3. Vertical Path – (vPath) a space used by devices to convey cars vertically

4. Slot-Horizontal Path – (slotHPath) a space where a car is parked but also used by devices to convey cars horizontally
5. Slot-Vertical Path – (slotVPath) a space where car is parked but also used by devices to convey cars vertically
6. Entry (or Input) Area – (iArea) a space for admitting a car from outside of the parking structure
7. Exit (or Output) Area – (oArea) a space for releasing a car out of the parking structure
8. Rotation Area – rArea a virtual space created when rotation movements is required

Table 1: Classification of spaces in the APS.

ID	Long Label	Short Label
sl	Slot	Slot
hp	Horizontal Path	hPath
vp	Vertical Path	vPath
sh	Slot-Horizontal Path	slotHPath
sv	Slot-Vertical Path	slotVPath
ia	Entry (/Input) Area	iArea
oa	Exit (/Output) Area	oArea
ra	Rotation Area	rArea

3.2. The devices and their operations

To create an operations model for the APS, a model for each robotic device should also be established. To model each device, the following steps are required.

1. Consider that devices would have "heads" or a reference point for its position orientation, that it is possible for it to have directions or positions and ability to measure horizontal, vertical, or angular displacements
2. Consider basic functions of a device
    a. Take a load – i.e. by its designed manner, engage a material to carry, the material being a car or other devices of expected standard size
    b. Release a load – i.e. disengage or release the load by its intended detailed process

c. Move – i.e. make movements about or through the spaces in the parking structure in the detailed manner it is designed, these movements may be while the device has a load or without. Movements of devices are further modelled in Section 3.

d. Request for another device to get or receive load – to request for presence of another device that is needed at the other adjacent space (destination) for receiving the load of the current device – for or simplicity, this is assumed fulfilled once requested, or 2) releasing the current load and allowing such other device to execute its needed function for getting the load—as there is movement required from the other device, the fulfilment of the function is then delegated to the requested device.

e. Ensure acquisition of requested load – to check if there was a function call from other devices for function d, i.e. "Request for another device to get or receive load", if there is, positioning as requested, executing the needed function to acquire the load, and completing the taking a load function

3. Consider a device movement as a unit motion consistent with dimensions of the spaces. Movement functions such as move, push, pull and rotate, can optionally indicate needed displacement. For rotation, displacement may be signed values to indicate direction, such that e.g. -90 means 90 degrees anti-clockwise, and also that rotation movements (15. Rotate) may be appended with basic movements of taking a load (16. getLoad) or releasing a load (0. releaseLoad)

4. Consider the order by which the device takes a load

   a. Anteloading – the device must first be present in a designated space before a load is assigned to it

   b. Postloading – the load may already be in a designated space before the device can take it as a load

5. Finally, using information established from earlier steps, consider modelling the devices.

   a. Base Model. A base model of device representing all generic movements and having the variable characteristics described can be used for modeling highly abstract APSs or when specific devices are not yet clearly established when designing the APS.

b. More Specific Device Descriptions. The base device model can later on be refined or configured basing from current implementation of devices in most APSs available in the market, such as the following:

   i. Elevator – a device that moves upward or downward at certain vertical displacement. It may also be able to rotate clockwise or counter clockwise at certain angular displacement. Elevators are pre-loading devices that may carry other devices.

   ii. Transfer Elevator – an elevator capable of pushing or pulling its load along its major axes

   iii. Wagon – a pre-loading device that moves horizontally along one of its axes and may run along continues rails and may be thought of as analogous to the elevator but for horizontal movements.

   iv. Transfer Wagon – a wagon capable of pushing or pulling its load along one of its axes.

   v. Carrier – an autonomous pre-loading device capable of all horizontal axial movements and rotation. It may be thought of as a robotic pallet thus it can ride onto other devices.

   vi. Rotator or Rotating Table – a device for rotating clockwise or anticlockwise from 0 to 360 degrees a car or another device it is currently carrying

Table 2: Generic description of available APS devices.

ID	Label	Functions
BM	Base Model	All functions
EL	Elevator	holdLoad, releaseLoad, moveUp, moveDown, rotate
TE	Transfer Elevator	holdLoad, releaseLoad, moveUp, moveDown, rotate, pushFore, pushBack, pushRight, pushLeft, pullFore, pullback, pullRight, pullLeft,
WA	Wagon	holdLoad, releaseLoad, moveFore, moveBack, moveRight, moveLeft, rotate
TW	Transfer Wagon	holdLoad, releaseLoad, moveFore, moveBack, moveRight, moveLeft, pushFore, pushBack, pushRight, pushLeft, pullFore, pullback, pullRight, pullLeft
CA	Carrier	holdLoad, releaseLoad, moveFore, moveBack, moveRight, moveLeft, rotate
RO	Rotator	rotate

3.3. Representing the models as data structure

By careful analysis, this spatial and operational configuration can be sufficiently represented as a graph. The spaces are the vertices while the devices with respective set of device functions needed to execute movement from an origin to a destination space or vertex may be assigned to the edges. Thus this follows the representation of a non-self-cyclic bidirectional graph

$$G' = (V, \cup_{\{u,v\}} \in E\{(u, v), (v, u)\}) \quad (1)$$

where V is the set of vertices with n ≥ 2 number of elements, and E is the set of edges where (u, v) ∈ V and u ≠ v.

As data structure for computer programming purposes, the graph may be represented as an array of vertices and an adjacency matrix of edges. With this, the identification of a suitable algorithm to be used for the APS control or operating system may be facilitated as algorithms, such as breadth-first-search (BFS), depth-first-search (DFS), or shortest-path [4], may be applied to locate a space for parking or retrieval operations [1].

3.4. Applying the method to popular current and hypothetical APS configurations

The method was used for modelling the following existing APS types. The detailed documentation of the application of the method is shown for Vertical or Tower Type APS to illustrate the method's application process.

1. A Current Vertical or Tower Type APS [12, 13]
    a. Physical Structure
    b. Spaces Identification and Device Assignments
    c. Graph Representation
    d. Configuration.
2. Vertices List
3. Edges List
4. A Current Horizontal Type APS
5. A Current Circular Type APS.

Figure 1: Pictorial view of the Tower Type APS.

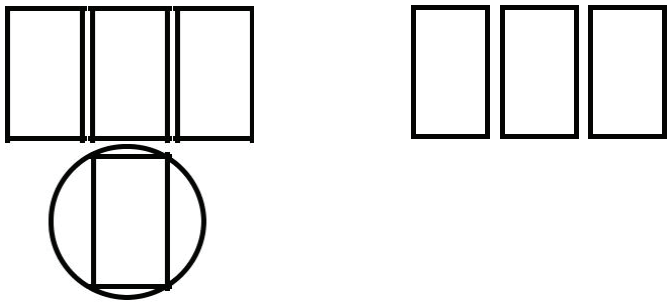

Figure 2: Plan of Entry/Exit Level (1st Level) of the Tower Type APS; and Typical Plan of 2nd, 3rd and 4th Level of the Tower Type APS.

6. A Current Vertical Type Puzzle
7. A Hypothetical Two-Level Compound Circular Type APS
    a. Physical Structure
    b. Spaces Identification and Device Assignments
8. Hypothetical Combination of Vertical, Horizontal and Circular Type APS
9. A Hypothetical Multilevel Horizontal Puzzle Type APS

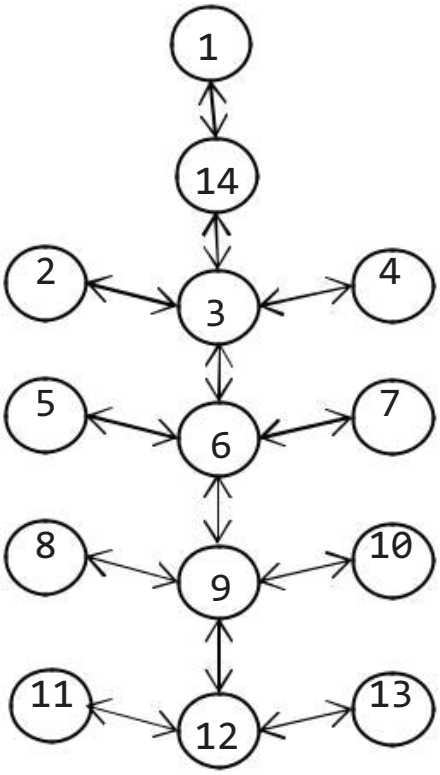

Figure 3: Graph representation of the Tower Type APS.

TABLE 3: Assignment of Spaces for the Tower Type APS.

Index	Type	Initial Device
1	xi/xo	1:CA
2	sl	
3	vp	2:TE
4	sl	
5	sl	
6	vp	
7	sl	
8	sl	
9	vp	
10	sl	
11	sl	
12	vp	
13	sl	
14	ra	2:TE, 180^o

Table 4: Assignment of devices and respective functions for the Vertical Type APS.

Edge Index	Origin	Destination	Assigned Device	Function/s
1	1	14	1:CA	getLoad and rotate
2	14	1	1:CA	releaseLoad
3	14	3	1:CA	moveFore
4	3	14	2:TE	moveFore
5	3	6	2:TE	moveUp
6	6	3	2:TE	moveDown
7	6	9	2:TE	moveUp
8	9	6	2:TE	moveDown
9	9	12	2:TE	moveUp
10	12	9	2:TE	moveDown
11	2	3	1:CA	holdLoad, moveRight
12	3	2	1:CA	moveLeft, releaseLoad
13	3	4	1:CA	moveRight, releaseLoad
14	4	3	1:CA	holdLoad, moveLeft
15	5	6	1:CA	holdLoad, moveRight
16	6	5	1:ca	moveLeft, releaseLoad
17	6	7	1:ca	moveRight, releaseLoad
18	7	6	1:ca	holdLoad, moveLeft
19	8	9	1:ca	holdLoad, moveRight
20	9	8	1:ca	moveLeft, releaseLoad
21	9	10	1:ca	moveRight, releaseLoad
22	10	9	1:ca	holdLoad, moveLeft
23	11	12	1:ca	holdLoad, moveRight
24	12	11	1:ca	moveLeft, releaseLoad
25	12	13	1:ca	moveRight, releaseLoad
26	13	12	1:ca	holdLoad, moveLeft

3.5. Testing for flexibility

1. Software design

To test the applicability and flexibility offered by the method, an application was created to implement the spatial and operational configuration and allow as well the simulation of the basic operations in the APS. Object-oriented Application Development (OOAD) method was used for creating simulation software with Java as the programming language.

Figure 4: A Pictorial of the Horizontal Type APS [2].

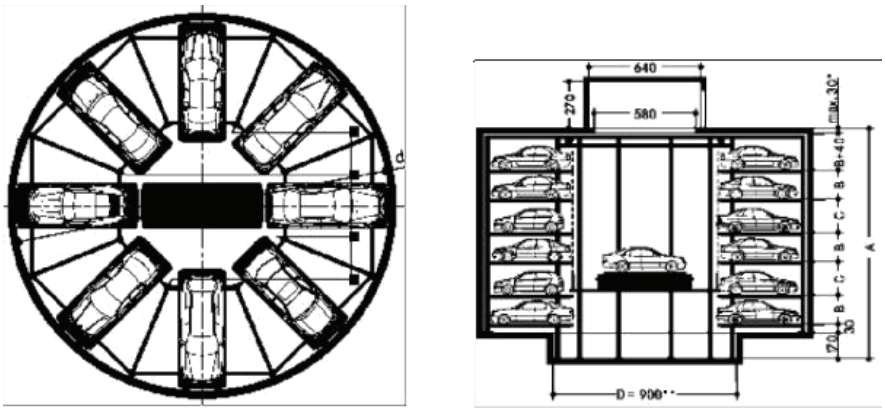

Figure 5: floor Plan and Side View of the Circular Type APS [13].

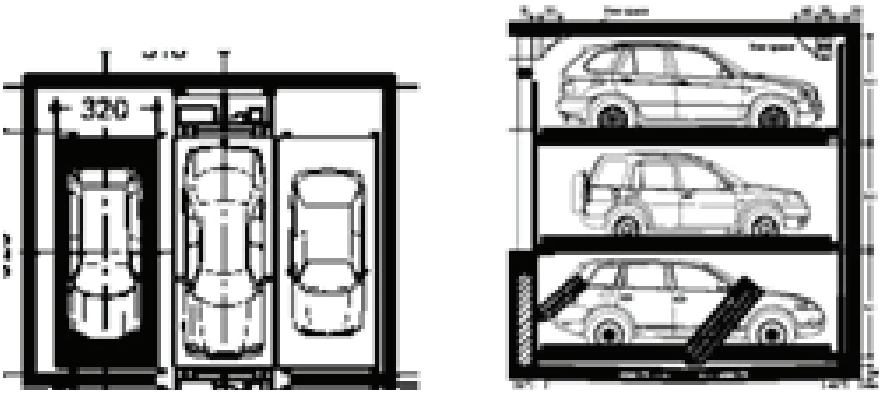

Figure 6: Vertical Puzzle Type System [14].

The graphs were implemented using an ArrayList of nodes (vertices) while another ArrayList (edgesList) and an adjacency matrix (edgesMatrix) were used

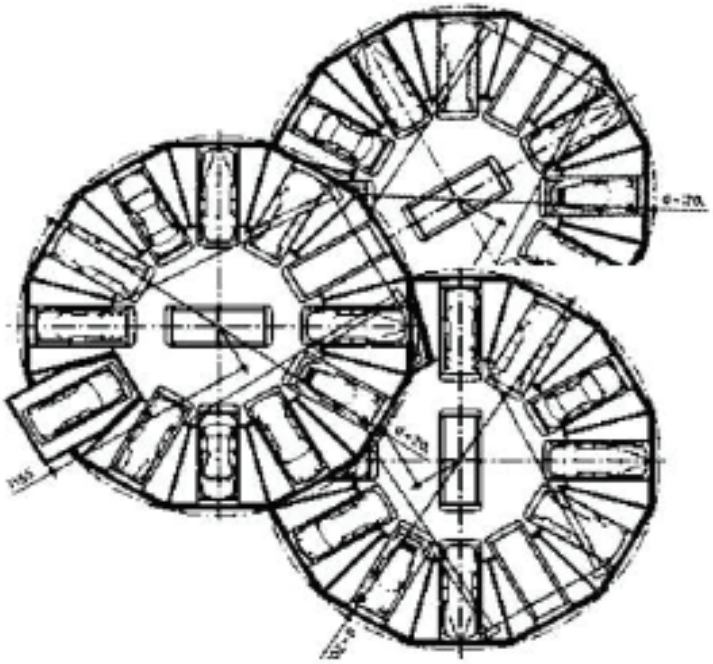

Figure 7: floor Plan of the 2-Level Compound Circular Type APS.

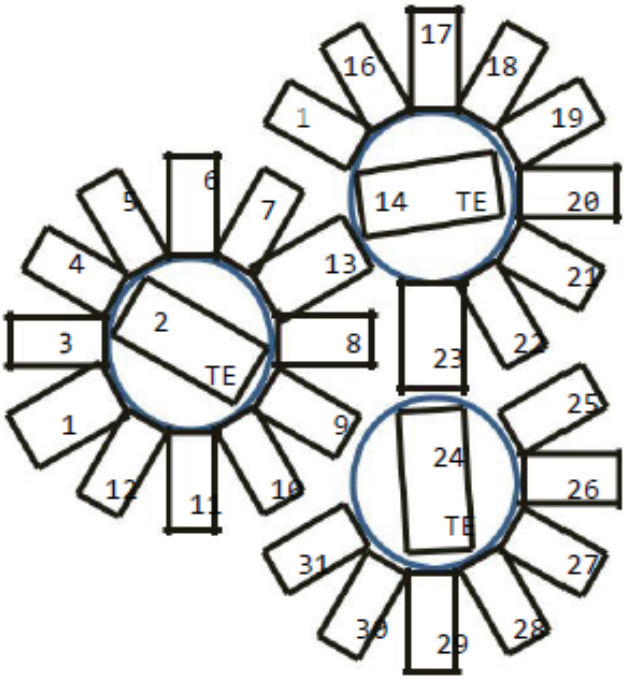

Figure 8: 1st Level of the Compound Circular Type APS.

to represent the set of edges for a particular APS graph. The input data were simply hardcoded for each test and results are displayed as simple text outputs on the console.

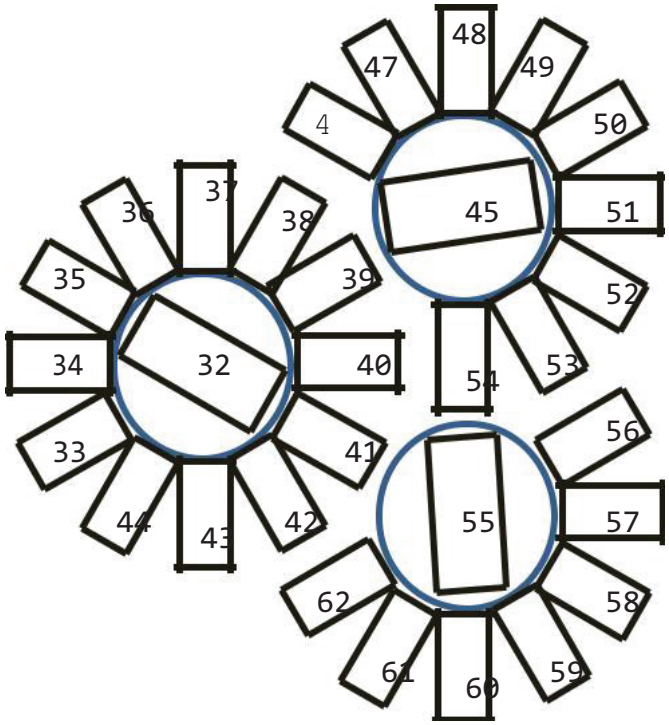

Figure 9: 2nd Level of the Compound Circular Type APS.

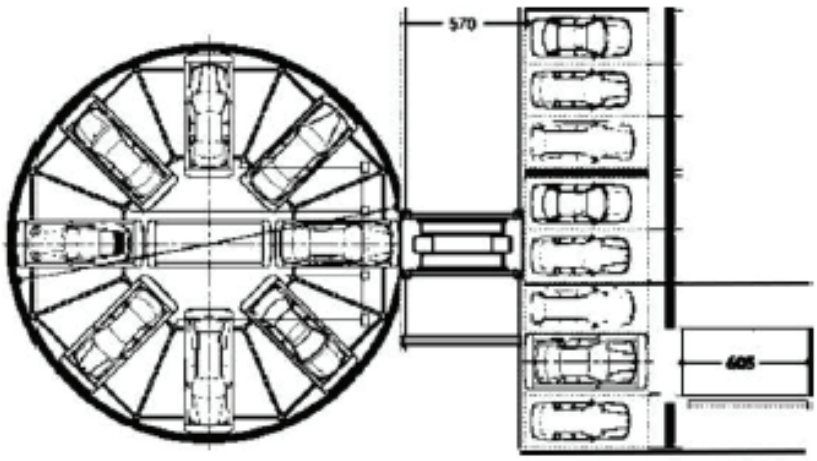

Figure 10: floor Plan of a Hypothetical Combination of Circular Vertical and Horizontal Type APS.

2. Algorithms

Although the execution of the basic operations can be directly done by manually identifying the series of functions that would be executed for a particular basic operation, a suitable algorithm instead was used to demonstrate that it is possible to employ one and automate the operations through its use. For this purpose, the algorithm in getPathDFS() method, which was based on DFS on graphs, was created and used for identifying the path from one node to another. Once the path is identified, the executions of the device functions were simulated using the

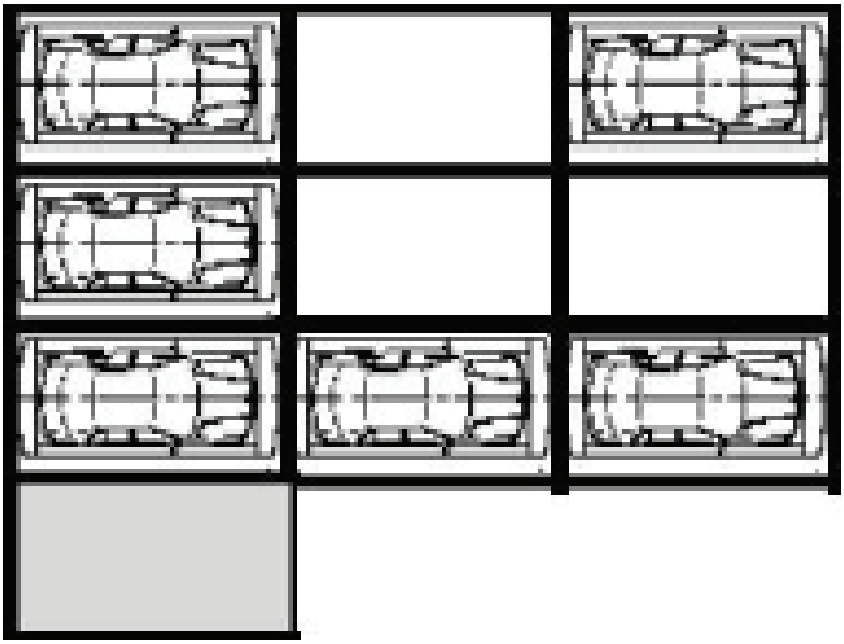

Figure 11: floor Plan of the Multilevel Horizontal Puzzle Type APS.

algorithm in traverse() method that simply displayed the name of each function as they are being done.

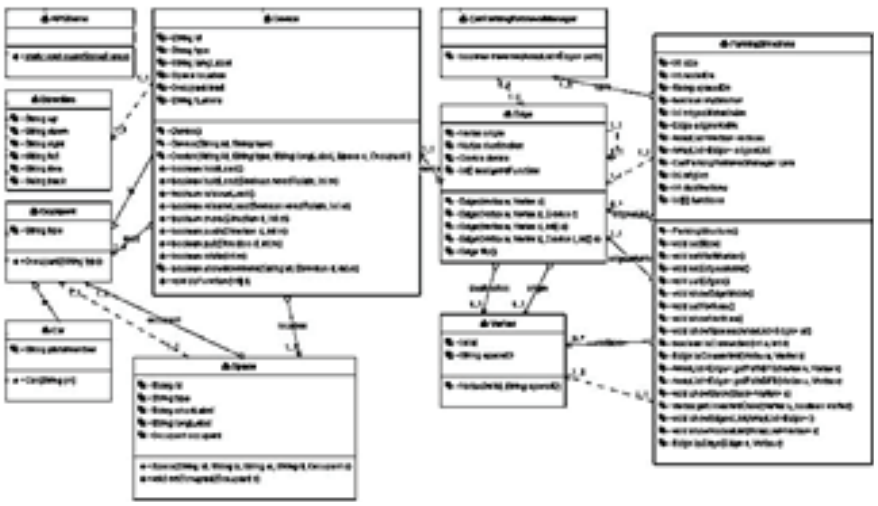

Figure 12: Class Diagram of the Simulator named APSDemo.

4. Results Analysis and Discussion

4.1. APS types or designs tested

The developed method was applied to design and configure popular and hypothetical APSs as described in section D, such as the Tower/Vertical Horizontal, Circular, Vertical Puzzle, Compound Circular, Combination of Vertical, Horizontal and Circular, and Multilevel Horizontal Puzzle types of APS

4.2. Basic operations tested

The following basic operations were used to check if the created models or configurations would allow correct operations:

1. Parking car from Entry/Exit area to a Slot in the same level
2. Parking car from Entry/Exit area to a Slot in another level
3. Retrieving a car from a Slot and delivering it to the Entry/Exit area
4. Moving a car from a Slot into another Slot located in the same level
5. Moving a car from a Slot into another Slot located in a different level

4.3. Test outputs

From the rigorous tests done, a simple representative output is shown to demonstrate the output of the simulator. This shows a basic operation of parking a car from the entry area into the last parking slot in the 4^{th} level of the Vertical Type APS shown in Figure 1.

4.4. Summary of tests and results

The following lists the types of APSs and the results of tests. A check mark indicates that the operation was successfully executed while occurrence of a cross mark would indicate the otherwise.

The verification of the applicability of the method involved currently available APS designs or configurations such the Tower Type System, Horizontal Type System, Circular, and Vertical Puzzle Type System. To further test the method, more rigorous experimentation were done using hypothetical configurations, such as a Compound

```
run:
vertices created: 15
main()| size of path: 6
--------
     origin: 1
destination: 14
doFunction()| Held Load
doFunction()| Rotated
--------
     origin: 14
destination: 3
doFunction()| Moved forward
--------
     origin: 3
destination: 6
doFunction()| Moved upward
--------
     origin: 6
destination: 9
doFunction()| Moved upward
--------
     origin: 9
destination: 12
doFunction()| Moved upward
--------
     origin: 12
destination: 13
doFunction()| Moved rightward
doFunction()| Released load
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 13: Sample Test Output of the Simulator.

Circular System, a Combination of Vertical, Horizontal and Circular Systems, and a Multilevel Horizontal Puzzle Type System were also used and simulated. Results show

Table 5: Summary of tests and results.

APS Type		Basic Operations			
Tower/Vertical	1	2	3	4	5
Horizontal	✓	✓	✓	✓	✓
Circular	✓	✓	✓	✓	✓
Vertical Puzzle	✓	✓	✓	✓	✓
Compound Circular	✓	✓	✓	✓	✓
Combination of Vertical,	✓	✓	✓	✓	✓
Horizontal and Circular					
Multilevel Horizontal Puzzle	✓	✓	✓	✓	✓

that all basic operations applicable to both current and hypothetical APS designs and configurations are correctly executed.

5. Conclusion and Recommendation

Basing from the results of the simulations, the developed method therefore is applicable for modelling current APS designs and configurations. It can also be used to design and configure any APS as shown by its applicability to model even hypothetical APS samples.

During the testing process, it could be also observed that the representations of spaces and devices are mechanically done thus a candidate for automation.

Further studies can be done focusing on identifying optimal algorithms, creating a complete operating or control system for the devices, or building a generic APS Car Parking and Retrieval Management System (CP/RS) while any of these future works may be more appreciable if the simulation could use graphical user interface (GUI) or, at best, some physical miniature models.

Appendix

Author's Note

Ramiro C. Advincula holds a Master of Science in Information Technology degree and has 6 units in Ph.D. in Computer Science major in AI. Both graduate studies are from De La Salle University, Manila. He is currently with Polytechnic University of the Philippines (PUP), Parañaque City Campus, appointed as the Information Technology Program Coordinator, and Chair of the Committee for Extensions of the campus.

Table 6: Basic functions of the devices.

ID	Function Description	Name
0	Release Load	releaseLoad
1	Move upward	moveUp
2	Move downward	moveDown
3	Move leftward	moveLeft
4	Move rightward	moveRight
5	Move forward	moveFore
6	Move backward	moveBack
7	Push load forward	pushFore
8	Push load backward	pushback
9	Push load rightward	pushRight
10	Push load leftward	pushLeft
11	Pull load from front	pullFore
12	Pull load from back	pullback
13	Pull load from right	pullRight
14	Pull load from left	pullLeft
15	Rotate	Rotate
16	Hold Load	getLoad
17	Request for next device	callDevice
18	If requested, be in position for getting a load, execute function to get load, then hold load	ensureLoad

Table 7: Loading Order and Type of Loads of the Devices.

Long Label	**Loading Order**	**Load Type**
Base Model	Anteloading, Postloading	Device or Car
Elevator	Anteloading	Device or Car
Transfer Elevator	Anteloading, Postloading	Device or Car
Wagon	Anteloading	Device or Car
Transfer Wagon	Anteloading, Postloading	Device or Car
Carrier	Postloading	Car
Rotator	Anteloading	Device or Car

Aaron Vito M. Baygan is a graduate of Liberal Arts Major in Philosophy and a holder of a Licentiate in Theology (CHED certified MAT equivalent) attained at the Gregorian University (Rome, Italy). He joined the Polytechnic University of the Philippines to teach in the Department of the Humanities in 1996 until his designation as campus director of PUP Paranaque in 2011.

References

[1] Ramiro C. Advincula. (2000) A Class of Space Adaptive Car Parking and Retrieval System for Automatic Parking Systems, De La Salle University, Manila, an unpublished MSIT thesis

[2] Bortome Parking System (2013) © Qingdao Bortome Import & Export Co., Ltd., Qingdao, China, http://www.bortome.com/product_show.php?id=452, accessed 28 December 2016

[3] Aad Mathijssen, A. Johannes Pretorius. (2007) Verified Design of an Automated Parking Garage. In: Brim L., Haverkort B., Leucker M., van de Pol J. (eds) Formal Methods: Applications and Technology. FMICS 2006. Lecture Notes in Computer Science, vol 4346. Springer, Berlin, Heidelberg

[4] Kurt Mehlhorn, Peter Sanders. (2008) Algorithms and Data Structures: The Basic Toolbox, © Springer Science & Business Media, pp. 49 and 170– 171, ISBN 978-3-540-77978-0, Algorithms and Data Structures: The Basic Toolbox, *accessed* 04 January 2017

[5] Jean-Claude Kaufmann, Christopher Dale Langhart, John Chester. (2009) System and Method for Parking Vehicles, Patent Application Publication, No. US 2009/0078172 A1

[6] M.Y.I. Idris, Y.Y. Leng, E.M. Tamil, N.M. Noor and Z. Razak. (2009). Car Park System: A Review of Smart Parking System and its Technology. Information Technology Journal, 8: 101-113. DOI: 10.3923/itj.2009.101.113

[7] Hitendra G.Wasnik, R.D.Askhedkar, S.K. Choudhary. (2011) Optimal Automatic Car Parking System for Indian Environment, © Indian Streams Research Journal (ISRJ), ISSN:-2230-7850, Vol.1,Issue.X/Nov; 11pp.1-4

[8] Mala Aggarwal, Simmi Aggarwal, R.S.Uppal. (2012) Comparative Implementation of Automatic Car Parking System with least distance parking space in Wireless Sensor Networks, © *International Journal of Scientific and Research Publications, Volume 2, Issue 10, October 2012* 1, ISSN 2250-3153, www.ijsrp.org

[9] Christopher Alan. (2014). U.S. Patent No. 8,632,290. Washington, DC: U.S. Patent and Trademark Office.

[10] Yatin Jog, Anuja Sajeev, Shreyas Vidwans, and Chandradeep Mallick. (2015) Understanding Smart and Automated Parking Technology, SERSC, ISSN: 2005-4246 International Journal of u- and e-Service, Science and Technology (IJUNESST) Vol.8, No.2 (2015), pp.251-262, http://dx.doi.org/10.14257/ijunesst.2015.8.2.25

[11] Patil Aishwarya, Patil Rupali, Patil Aboli,Vijaykumar Kamble. (2016) Multi floor Automatic Car Parking Using PLC, Vol-2 Issue-3 2016 IJARIIE-ISSN(O)-2395-4396

[12] Tower Parking System, Simark Infrastructure Private Limited | Kolkata,West Bengal, https://dir.indiamart.com/impcat/tower-parking-system.html?biz=10, accessed 28 December 2016

[13] Otto Wöhr GmbH, Ölgrabenstr. 14, 71292 Friolzheim, Germany, Vehicle drawings © creativ collection Verlag GmbH/www.ccvision.de, http://www.woehr.de/en/product/parksafe-585.html, accessed 28 December 2016

[14] Wohr Parking System Pvt. Ltd © 2017. All Rights Reserved., http://www.wohrparking.in/combilift543.htm, accessed 29 December 2016

[15] MHE Automated Guided Vehicle Parking Systems, © MHE Demag (S) Pte Ltd., http://www.mhe-demag.com/products/car_parking_systems, accessed 29 December 2016