Puzzle-based Parking for Large Lots

Parag J. Siddique*, John S. Usher, Kevin R. Gue

Department of Industrial Engineering, University of Louisville, Louisville, KY 40292, USA

Abstract

Intelligent capabilities of autonomous vehicles give us the opportunity to achieve very high-density parking. However, as the size of a dense parking lot increases, so does the time to retrieve a vehicle. To address this problem, we introduce the *sub-lot* design in which small modules of high-density lots are connected together to develop a layout for a large parking lot. We describe computational models of parking density and vehicle retrieval time in sub-lot designs and use our models to investigate various design alternatives. We also perform a case study to illustrate the proposed method.

Keywords: high density parking, autonomous vehicles, Rush Hour puzzle, heuristic search

1. Introduction

The prospect of autonomous vehicles presents many interesting opportunities to improve urban transportation. For example, the potential for self-parking would allow cars to be parked closer together because there would be no need for doors to open to allow passengers to exit. It is also possible to omit driving lanes, as these cars will be equipped with technology to communicate with other cars. Whenever a car needs to leave from a dense parking lot, it requests its neighbor car to make room for it, and then using this makeshift lane, that car leaves.

Some researchers have already looked into the high-density parking problem. Ferreira et al. (2014) propose a design for high density parking for autonomous vehicles which increases parking capacity by 50 percent. In their design, cars are parked by blocking each other in parallel lanes, similar to the deep lane storage systems used in warehouses and distribution centers. The same group of researchers has extended this work to consider various operational parameters such as parking space per car, number of maneuvers, travel distance, and retrieval time (Nunes et al., 2014; d'Orey et al., 2016; Azevedo et al., 2017, 2020). Timpner et al. (2015) modified the design of Ferreira et al. (2014) to reduce the number of interfering cars by reducing the lane depth and allowing cars to depart from either end of a lane. The authors estimate density improvement of 33%. Banzhaf et al. (2017) propose modifying existing layouts by allowing perpendicular lanes of blocking cars within existing aisles. They report an increase in density of 25%. Nourinejad et al. (2018) suggest a layout design in which the width of the aisle varies with the depth of the lane to help accommodate relocating

*Corresponding author
Email address: parag.siddique@louisville.edu (Parag J. Siddique)

cars. The authors develop a mixed-integer non-linear program to minimize the expected number of interfering cars. Recently, Siddique et al. (2021) proposed *puzzle-based parking* a high-density design for small parking lots without considering traditional "row and aisle" structure. They present a parking algorithm which can achieve maximum density in a parking lot.

All these studies keep an buffer space for vehicle relocation which is shared by all the cars in the parking lot. In the parking lot peak demand vehicle relocation in this shared buffer space will create long delays in vehicle storage and retrieval. The question becomes, how can we design parking lots in such a way that we get very high density, but cars can be retrieved in a reasonable time? In this paper, we solve this problem by connecting small modules of puzzle-based layouts to create large parking lot. We call these modules *sub-lots*.

2. Modeling the modular parking lot

A traditional parking lot follows a "row-and-aisle" geometry as shown in Figure 1a. A simple approach for integrating puzzle-based designs in a large parking lot would be arranging density-packed sub-lots in a "row-and-aisle" pattern (Figure 1b). This way a parking lot design could be done sequentially in two steps: i) design the sub-lot, and ii) integrate the sub-lots into the large lot.

(a) traditional lot. (b) Sub-lot.

Figure 1: A large parking lot.

2.1. Design of a sub-lot

Similar to Siddique et al. (2021), we consider a sub-lot as a rectangular grid. Cars can be parked horizontally or vertically. We assume cars occupy two cells in the grid. Cars can make three types of moves inside the lot: straight, right angle, and parallel, and they accomplish these maneuvers in forward or reverse. We assume that cars leave the lot only when called upon to exit; that is, a car may not leave the lot for the sole purpose of allowing another car to exit. We assume there is a central controller agent in the parking lot, that can communicate with the vehicles to execute movements. Constraints that to consider includes: i) multiple cars cannot share one location (no overlapping), ii) an entire row cannot be parked with horizontal

cars, or an entire column cannot be parked with vertical cars. Considering these design features, Siddique et al. (2021) developed a puzzle-based parking algorithm, which yields almost 100 percent density.

In a traditional parking lot, cars are not blocked. Each car is able to leave the lot at any time. However, in puzzle-based parking lots, makeshift aisles are created by moving blocking cars to retrieve a car. Designing efficient retrieval methods is the most challenging task for planners. Similar to Siddique et al. (2021), we model the parking lot as a state space graph, then find optimal retrieval time using A* search. Though Siddique et al. (2021) used only one entrance/exit point for the puzzle-based designs, in this larger case, we allow cars to enter/exit the sub-lot along an entire side. Figure 2 shows a 5×5 sub-lot with 10 cars. Cars exit/enter through entire right side of the grid. The number of moves required to retrieve each car is shown on that car. The mean number of moves required to retrieve a car is 10.6. We seek to arrange cars in such a way as to minimize the mean number of retrieval moves. Due to combinatorial nature of this puzzle-based design, as the size of the sub-lot increases, computing optimal retrieval time becomes intractable.

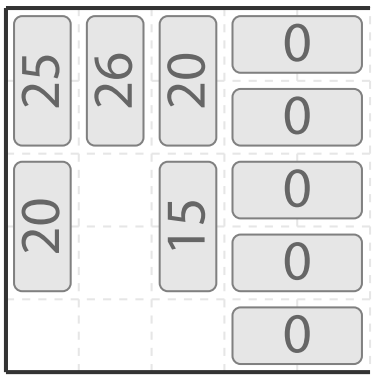


Figure 2: A 5×5 sub-lot with 10 cars. Cars are retrieved when it reaches on right two columns (gray colored cells). Number on a car indicates the number of moves it takes to retrieve a car.

2.2. Integrate the sub-lots into the large lot

To integrate sub-lots into a large lot, we follow the process of designing a traditional parking lot. Let's revisit this process:

1. Discretize the parking lot into grids. We assume a car takes two cells in this grid while parked horizontally or, vertically.
2. Identify a "column-block" which consists of two parking spots and driving lane segment, as shown in Figure 3a.
3. Stack multiple column-blocks to complete one column in the parking lot (Figure 3b).
4. Repeat this column in the entire length of the parking lot (Figure 3c).

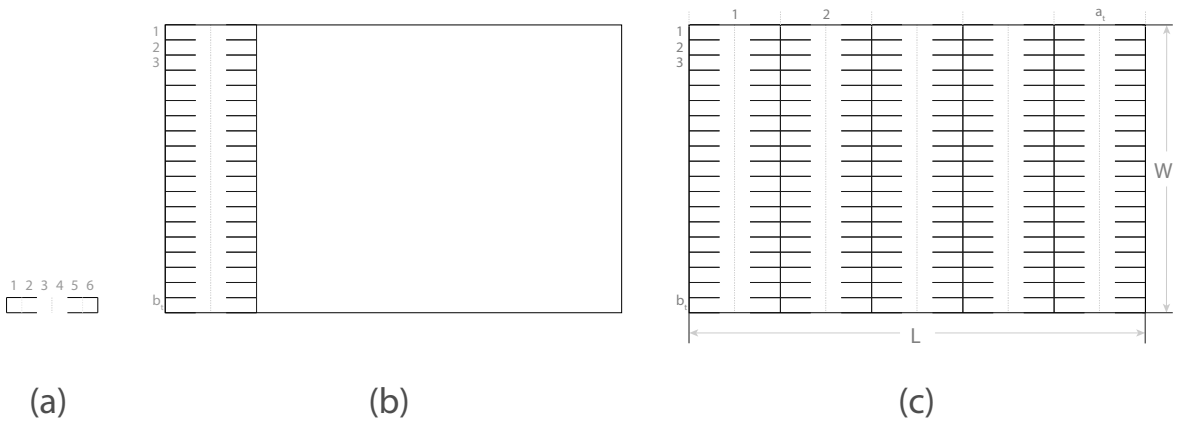


Figure 3: (a) column-block, (b) A Stack of column-blocks, (c) a traditional parking lot.

According to this routine, we can compute the number of cars, length, and width of a traditional parking lot using the following expressions:

Number of cars: $N_t = 2a_t b_t$

Length of the parking lot: $L = 6a_t c$

Width of the parking lot: $W = b_t c$

where a_t is the number of column-blocks in a traditional lot, b_t is the number of rows in a traditional lot, and c is cell width in the grid.

We are also interested in measuring parking density, which can be expressed as:

$$\text{Density} = 1 - \frac{\text{empty area}}{\text{total area}}$$

According to Figure 3a, density of a traditional parking lot can be computed as:

$$D_t = 1 - \frac{2}{6} = \frac{1}{3} = 0.67$$

To simplify our design and computation method, we assume one side of the parking lot is closed-end and the exit/entrance is on the bottom side, located centrally. To compute vehicle retrieval time in the large lot, we use the same A^* search.

Similar to this traditional design we can design a sub-lot parking lot, steps are as follows:

1. Discretize the parking lot into grids.
2. Identify a sub-lot column-block (Figure 4a).
3. Stack multiple column-blocks to fit the entire width of the parking lot (Figure 4b).
4. If there any leftover rows, make a sub-lot with those rows.
5. Repeat this column-block in the entire length of the parking lot (Figure 4c).

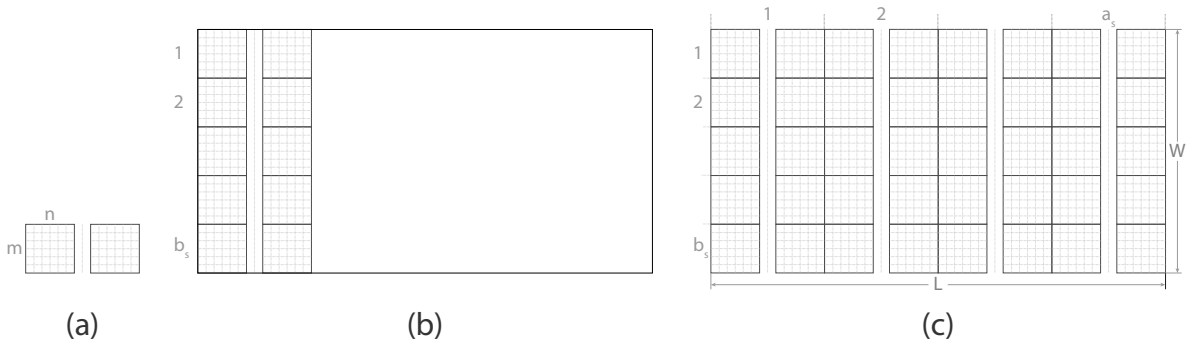


Figure 4: (a) Sub-lot column-block, (b) a stack of sub-lot column-blocks, and (c) a sub-lot parking lot

6. If there any leftover columns, fit traditional design into it.

According to this routine, we can compute design features of a sub-lot parking lot using following expressions:

Number of cars: $N_s = 2a_s b_s(\frac{mn-e}{2}) + 2a_t b_t$

Length of the parking lot: $L = (2n+2)a_s c + 6a_t c$

Width of the parking lot: $W = b_s c + b_t c$

where a_s is the number of sub-lot column-blocks, b_s is the number of sub-lot rows in a parking lot, m is number of rows in a sub-lot and n number of columns in a sub-lot and e is the number of empty cells in a sub-lot.

The density of a uniform sub-lot can be computed as (Figure 4a):

$$D_s = 1 - \frac{2m + 2e}{2m + 2(m \times n)} = 1 - \frac{m+e}{m(1+n)} \tag{1}$$

Similar to traditional design, we assume one side of the parking lot is closed-end and the exit/entrance is on the bottom side, located centrally. In a sub-lot design, a vehicle is retrieved in two steps: i) retrieval from the sub-lot and ii) retrieval through the driving lanes (Figure 5). For both of these steps, we use the same A^* search.

3. Experimental Design

To determine when sub-lot designs could be useful and which size of sub-lot provides improvements in terms of density and retrieval time, we conducted computational experiments with several candidate sub-lots, parking lot capacity and parking lot shape.

3.1. Sub-lot size, $m \times n$

We varied $m = 3, 4, \ldots, 10$, and $n = 3, 4, 5, 6$, a total of 32 sizes. We assume each of these sub-lots are utilized with maximum parking density. A few example sub-lots are illustrated in Figure 6. The number on each car indicates the number of moves it takes to retrieve that car

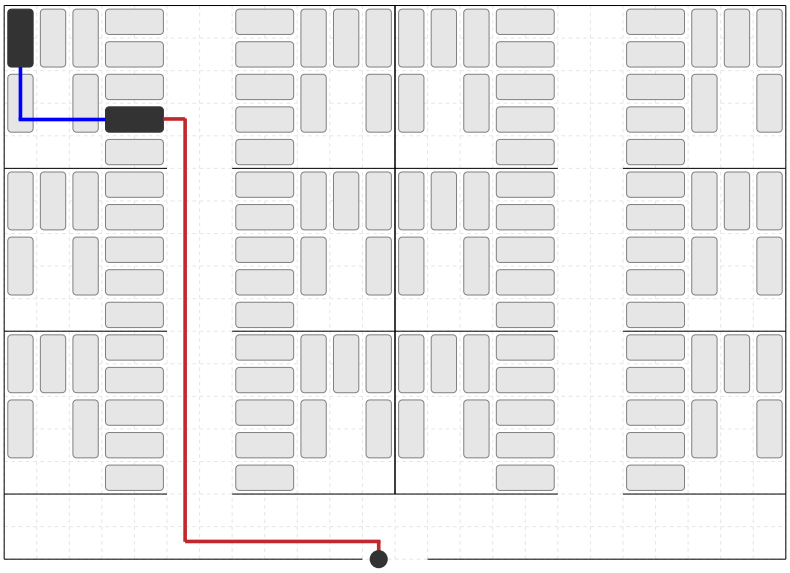


Figure 5: Vehicle retrieval in two steps, i) retrieval from the sub-lot and ii) retrieval through the driving lanes

3.2. Parking capacity

We studied both small and large parking capacities. For small lots, we consider parking lots with capacity 50, 100, 150, and 200 cars. Similarly, for large lots we consider parking lots with capacity 500, 1000 1500, and 2000.

3.3. Parking lot shape

Parking lots vary in width and length, so we varied parking lot shape, $r = \frac{W}{L}$, from $0.1, 0.2, \ldots, 2.0$, a total of 20 shapes.

3.4. The experiment

Steps for the experiment are as follows:

1. Compute width and length of a traditional parking lot with parking capacity N_t, and shape r, where $r = \frac{W}{L}$.

   Solving following equations,

   $$\begin{aligned} 2a_t b_t - N_t &= 0 \\ \frac{b_t}{6a_t} - r &= 0 \end{aligned} \tag{2}$$

   we compute a_t and b_t. Then, we can compute width and length of a parking lot.

2. Compute width and length of various shapes (0.0 to 2.0) of parking lot keeping the area ($L \times W$) same as in step 1.

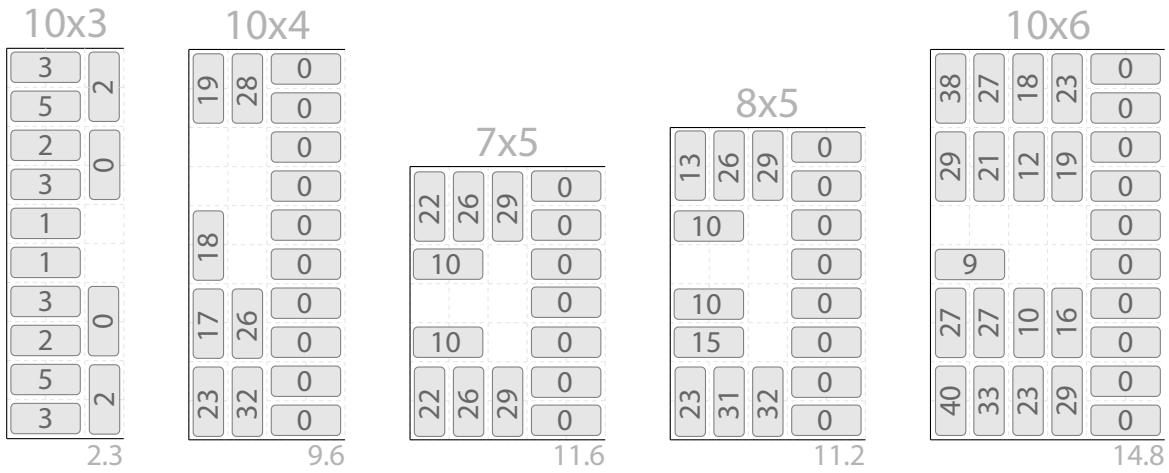


Figure 6: Various sub-lots.

3. For all shapes, compute number of cars and retrieval time for both traditional and sub-lot design (varying m and n).

In all, we evaluated 5,120 different combinations of parking lot sizes, shapes and sub-lots. We compared parking density and retrieval time of traditional design and sub-lot design.

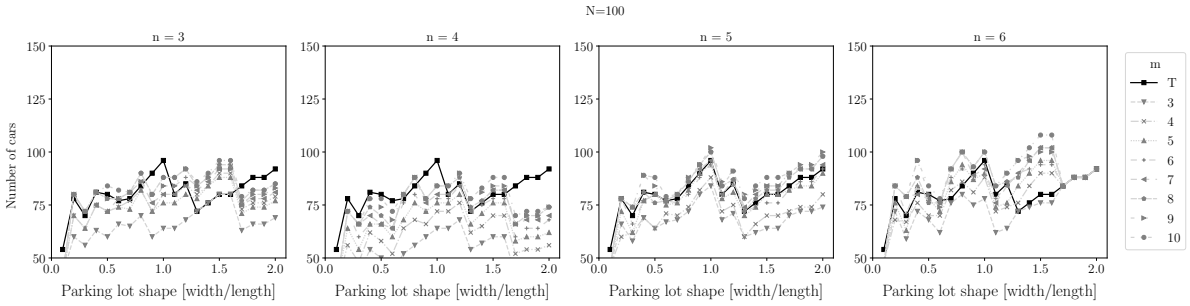
4. Results

4.1. Density analysis

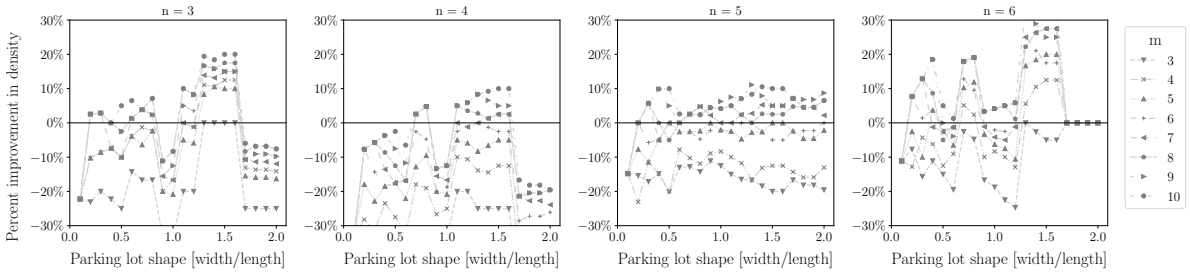
Figure 7 shows the results of a parking lot with approximate capacity of 100 cars. Similarly, Figure 8 shows the results of a parking lot with capacity of 1000 cars. The dark line in Figure 7a and Figure 8a is the capacity of a traditional parking lot for various shapes. We observe that shape does not have any significant impact on the parking capacity in a traditional parking lot. The same is true for sub-lot designs. However, as the size of sub-lot increases, so does the percent improvement in density. This improvement is significant in the large parking lot. We also observe that the sub-lot with column $n = 4$ always yields less density than the traditional design.

To clearly understand relationship between percent improvement in density, sub-lot size and parking capacity, we reproduce earlier plots, without the parking lot shape, in Figure 9. It can clearly observed that percent improvement in density increases with the increase in sub-lot size. We also observe a similar trend for parking capacity. However, percent improvement in density is almost same in the large parking lots ($N = 1000, \ldots, 2000$).

Note that percent improvement in density levels off at some point in each case. So, in Figure 10, we increase value of m, and find that improvement does not exceed 25%. If we increase m more than 10, there is no significant improvement in density. Therefore, $m \leq 10$ is a good limit for a sub-lot. It is also interesting to note that a sub-lot with $n \geq 4$ shows gain in density when $m \geq 10$. Afterwards, we also relax the value

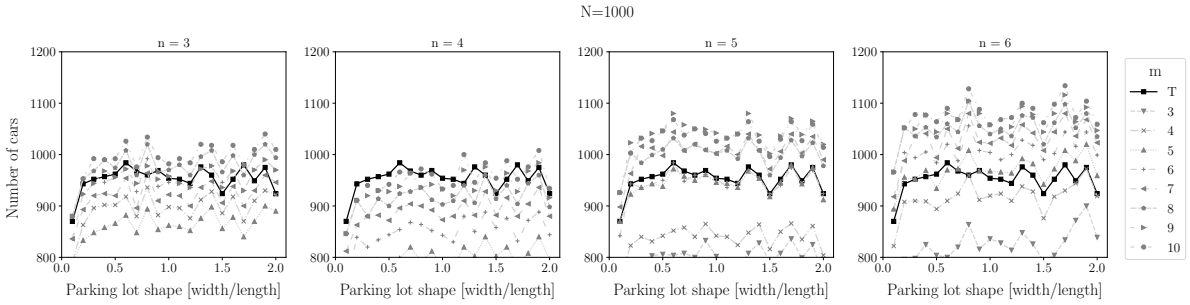


(a) Number of cars

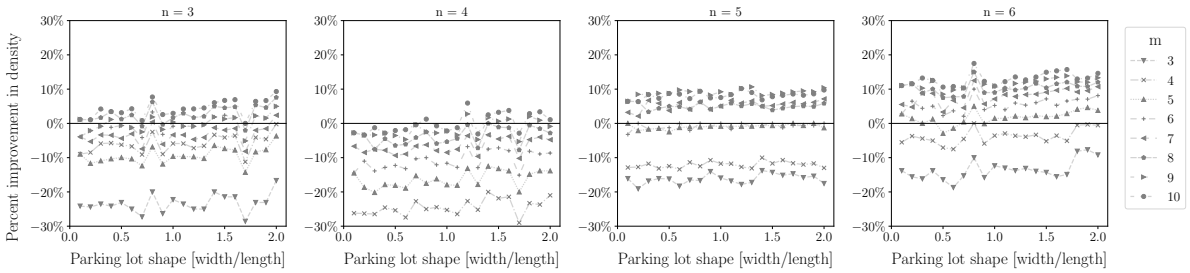


(b) Percent improvement in density

Figure 7: A small parking lot with approximate of capacity 100 cars.



(a) Number of cars



(b) Percent improvement in density

Figure 8: A Large parking lot with approximate of capacity 1000 cars.

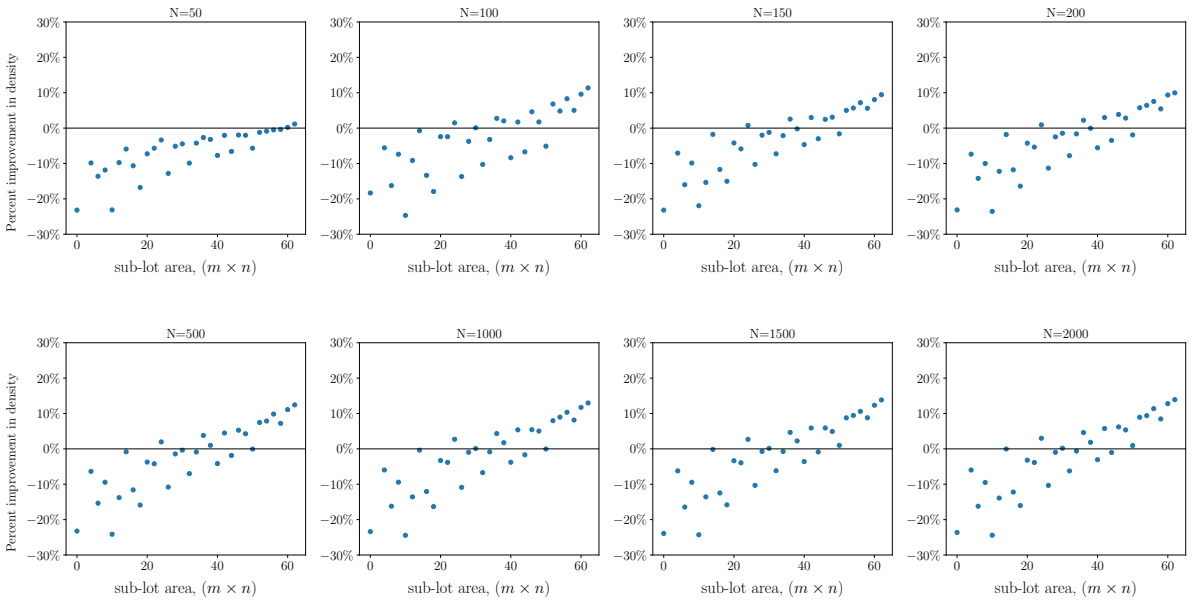


Figure 9: Percent improvement in density with respect to sub-lot area varying parking capacity

of n, and observe that improvement in density reaches as high as 50 % which is the maximum possible improvement.

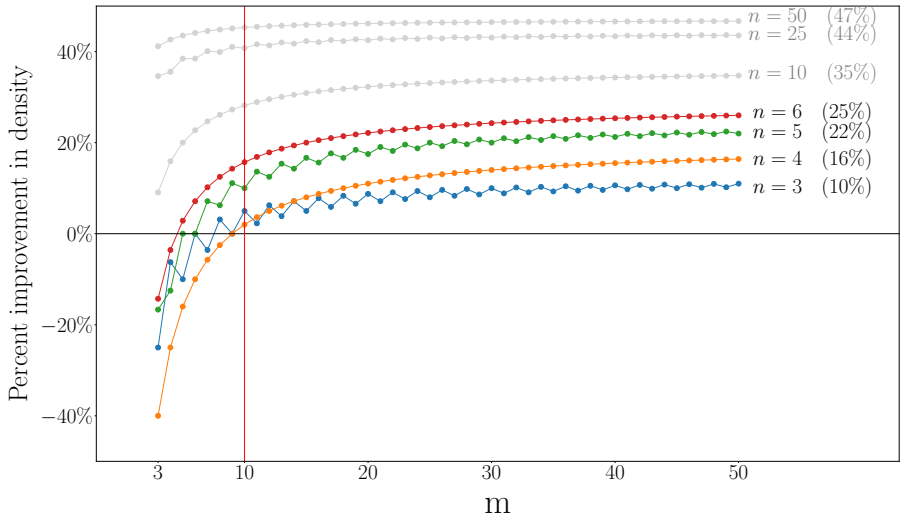


Figure 10: Bound on percent improvement in density.

4.2. Retrieval analysis

As in a sub-lot design, a vehicle is retrieved in two steps: i) retrieval from the sub-lot and ii) retrieval through the driving lanes. We analyze these two steps separately. Later, we also present an overall retrieval study. To generalize our retrieval results, we use a number of moves require to retrieve a car as the performance

metric in this experiment. Note that retrieval moves can be converted to retrieval time by considering cell dimension and speed of vehicle.

4.2.1. sub-lot retrieval

Figure 11 shows box plots of number of moves require to retrieve cars from various sub-lot design. As a general trend, we observe that as the size of the sub-lot increases, the number of moves required to retrieve vehicles increases. Additionally, computational time increases exponentially. The time required to retrieve all the vehicles from a 6×10 sub-lot takes more that 100 hours (Figure 12). It should be noted that this retrieval computation is not performed in real-time, but instead is computed in a preprocessing step and then stored in the parking controller's memory. We ran the computations on a desktop computer running macOS Catalina with an Intel(R) Core(TM) i7 3.5 GHz with 8 cores and 16GB RAM. Algorithms were implemented in Python 3.7.

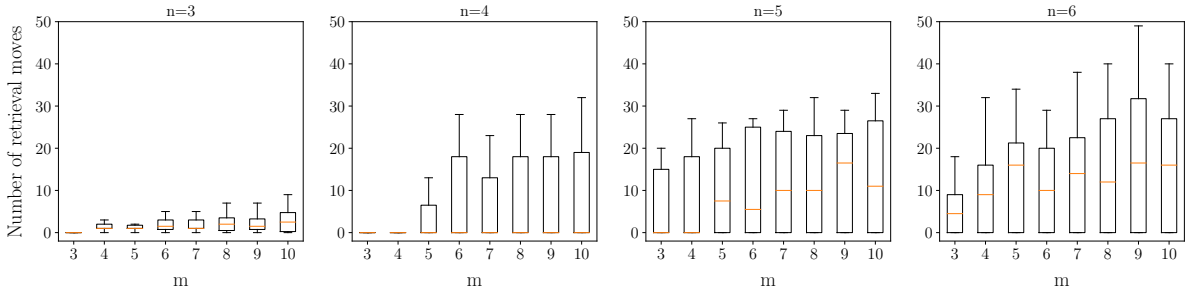


Figure 11: Sub-lots retrieval moves

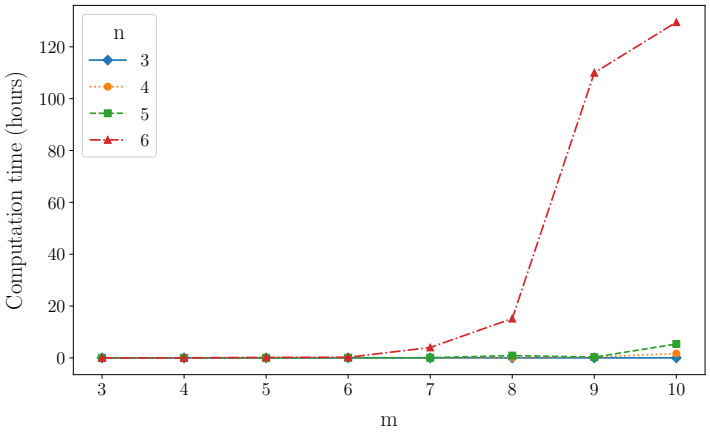


Figure 12: Sub-lots retrieval computation time

4.3. Driving lane retrieval

Figure 13 shows mean retrieval moves in sub-lot parking lots. As the capacity of parking lot increases (50 to 2000) it, of course, takes longer to retrieve a car. Additionally, the shape of the lot has a significant impact

on retrieval time. A parking lot shape with $r = 0.5$ yields minimum mean retrieval moves in a parking lot. Pohl et al. (2009) also reported similar results for warehouse designs. We observe that for larger parking lots, the retrieval plot takes on a convex shape. Following the law of large numbers, retrieval time becomes close to the expected value.

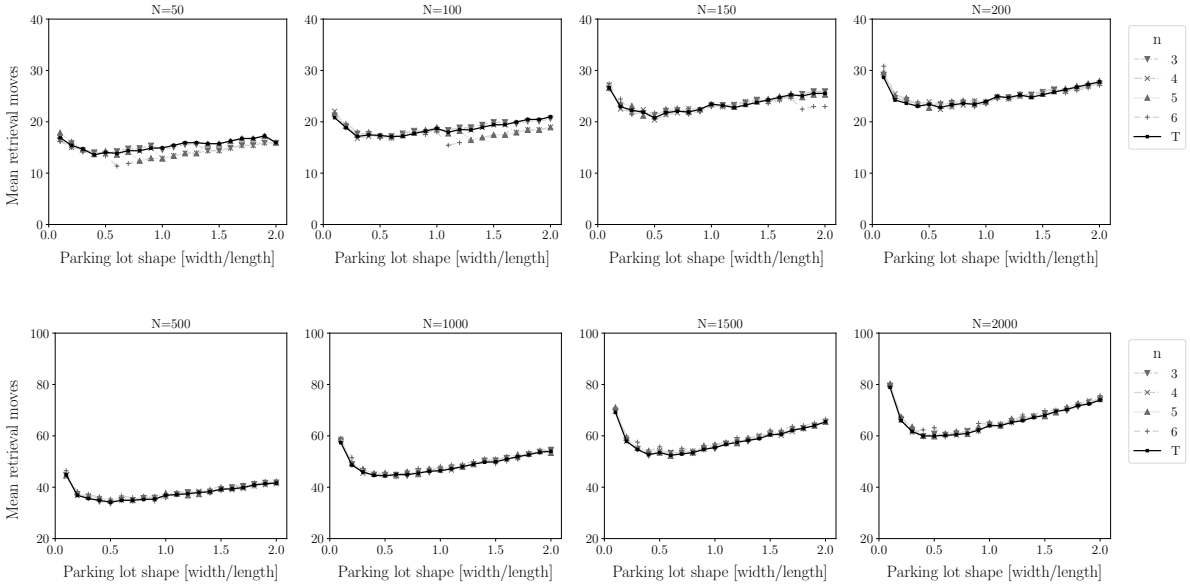


Figure 13: Mean retrieval moves in driving lanes.

It is interesting to observe that the number of retrieval moves in a traditional design is the same as driving lane retrieval moves in a sub-lot design. Assuming parking spots as continuous random variables and the I/O point in the center of the parking lot width, the expected distance travelled to retrieve a car with random storage is the sum of the expected travel along the length and the expected travel along the width.

As the the number of retrieval moves in a traditional design is the same as driving lane retrieval moves in a sub-lot design, retrieval moves from a sub-lot is the most important consideration in designing a sub-lot parking lot. Additionally, due road traffic flow, the I/O point of a parking cannot be always placed in the center of a parking lot width. Even for this case, driving lane retrieval moves will be same as the traditional lot retrieval moves.

4.4. Total retrieval moves in sub-lot designs

Figure 14 shows overall retrieval moves, combining sub-lot retrieval and driving lane retrieval, in sub-lot parking lots. As expected, we observe that total retrieval moves are offset from traditional retrieval moves by sub-lot retrieval moves.

4.5. Comparison with the designs in literature

At this point it is important to compare sub-lot design with existing literature. The designs of Ferreira et al. (2014); Timpner et al. (2015); Nourinejad et al. (2018) produce almost 100 percent density in a large

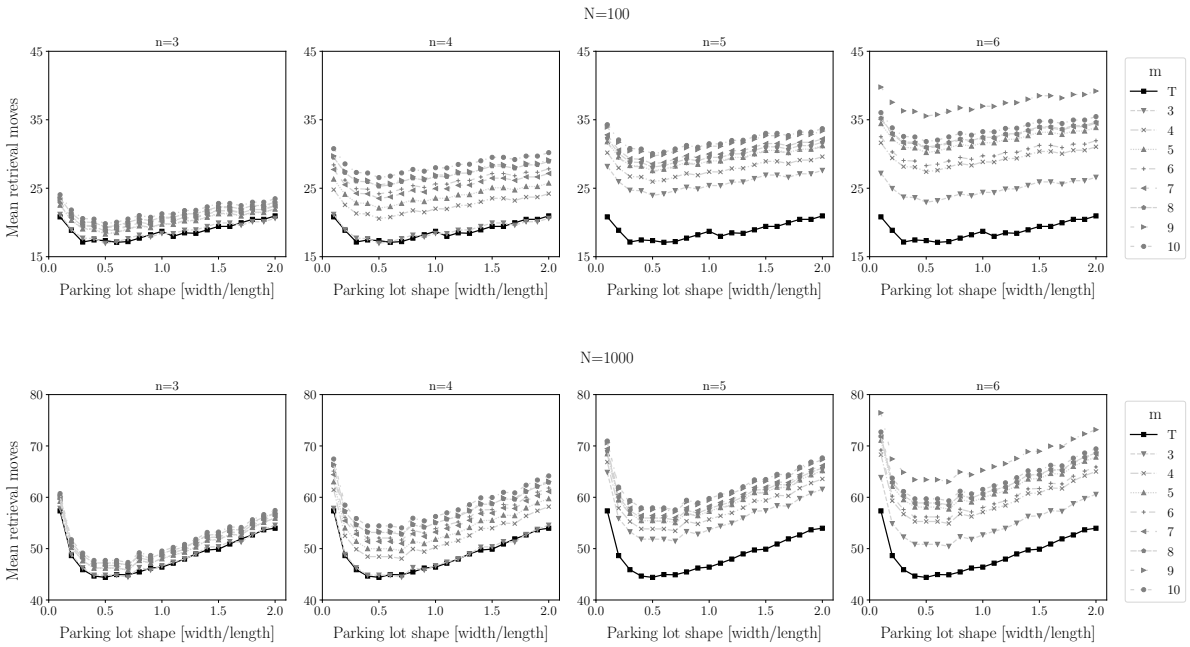


Figure 14: Mean retrieval moves, combining sub-lot retrieval and driving lane retrieval, in sub-lot parking lots.

parking lot which is same as our sub-lot design. The design proposed by Banzhaf et al. (2017) produces approximately 80% density, which is much less than our design.

The main benefit of sub-lot designs is reduced retrieval time. In all these designs, buffer space for vehicle relocation is shared by all the cars in the parking lot. Therefore, during the parking lots peak demand (morning and afternoon), vehicle relocation in shared space will create unavoidable delays. As the problem space is very large, it is very difficult to solve this planning problem. However, in our design this delay is limited inside the sub-lots only—relocations in one sub-lot will not have any impact on other parts of the parking lot.

5. Case Study

In this section, we design a real parking lot using our proposed sub-lot methodology. We have chosen the Cardinal Stadium parking lot B at the University of Louisville (Figure 15). This lot can accommodate approximately 1300 cars. We have chosen this lot because it is large and rectangular in shape.

To verify our model, we developed a traditional parking lot to fit this parking lot. We discretize this parking lot using a grid of $9' \times 9'$ squares. We consider a parking stall length 18 feet and width 9 feet. The width of the driving lane is 18 feet. We found that our model computes 1323 cars in this parking lot, which means the model is very accurate. To exit this parking lot, a car travels 460 feet on average. Assuming an average vehicle speed 11 ft/s, mean time to retrieve a car is 42 seconds.

Then we place various sizes of the sub-lots in the large lot using our sub-lot allocation method to identify which sub-lot gives the best space utilization. Figure 16a shows the sub-lots that give improved space

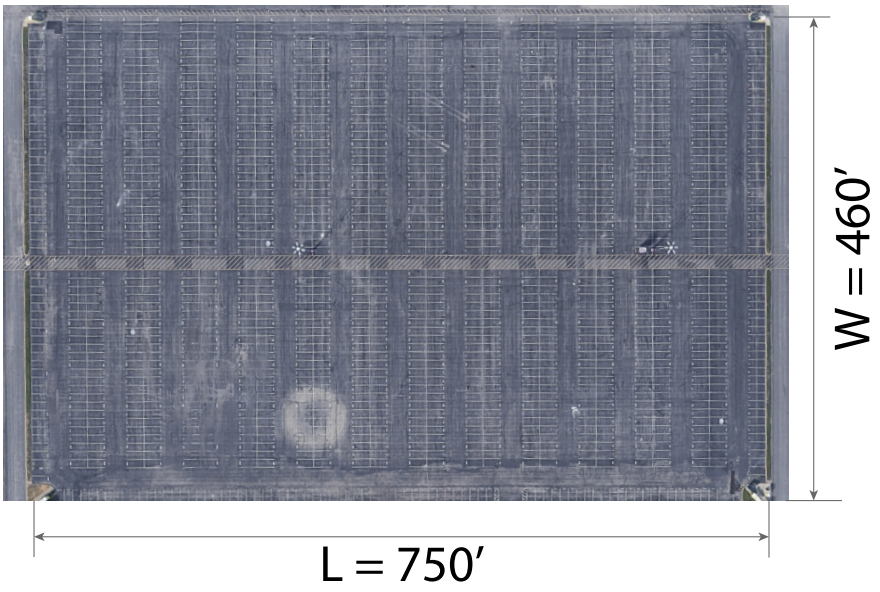
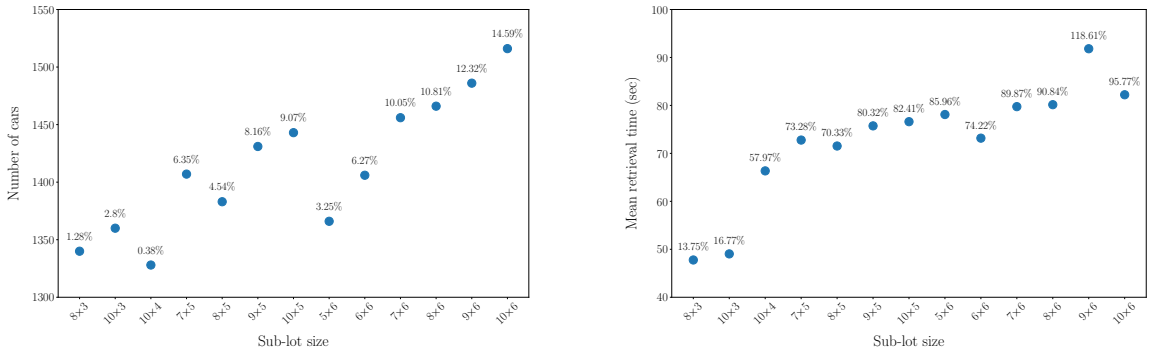


Figure 15: Cardinal stadium lot B at University of Louisville, parking capacity 1300 cars.

utilization than a traditional design. We have also annotated percent improvement for each sub-lot. To compute vehicle retrieval time, we assume a conservative speed inside a sub-lot: 3.5 ft/s. We have also plotted mean retrieval time from a modular parking lot using these sub-lots (Figure 16b). We observe that as we gain more parking capacity it takes longer to retrieve a car.



(a) Number of cars.

(b) Mean retrieval time.

Figure 16: Space utilization using various sub-lot size, (a) number of cars and (b) mean retrieval time.

The best result is obtained using a 10×6 sub-lots. In Figure 17, we show a layout using this sub-lot. In this layout, we have integrated five 10×6 sub-lot column-blocks and two traditional column-blocks. This design can accommodate 1516 cars. A car takes 82 seconds, on average, to exit this parking lot.

We develop an interesting layout design in Figure 18, using 10×6, 10×3 and traditional column-blocks. In this design we can park 1456 cars with an average retrieval time of 75 seconds. Retrieval from traditional column-blocks is very quick, then from 10×3 sub-lot columns it takes a little longer, and finally, from 10×6

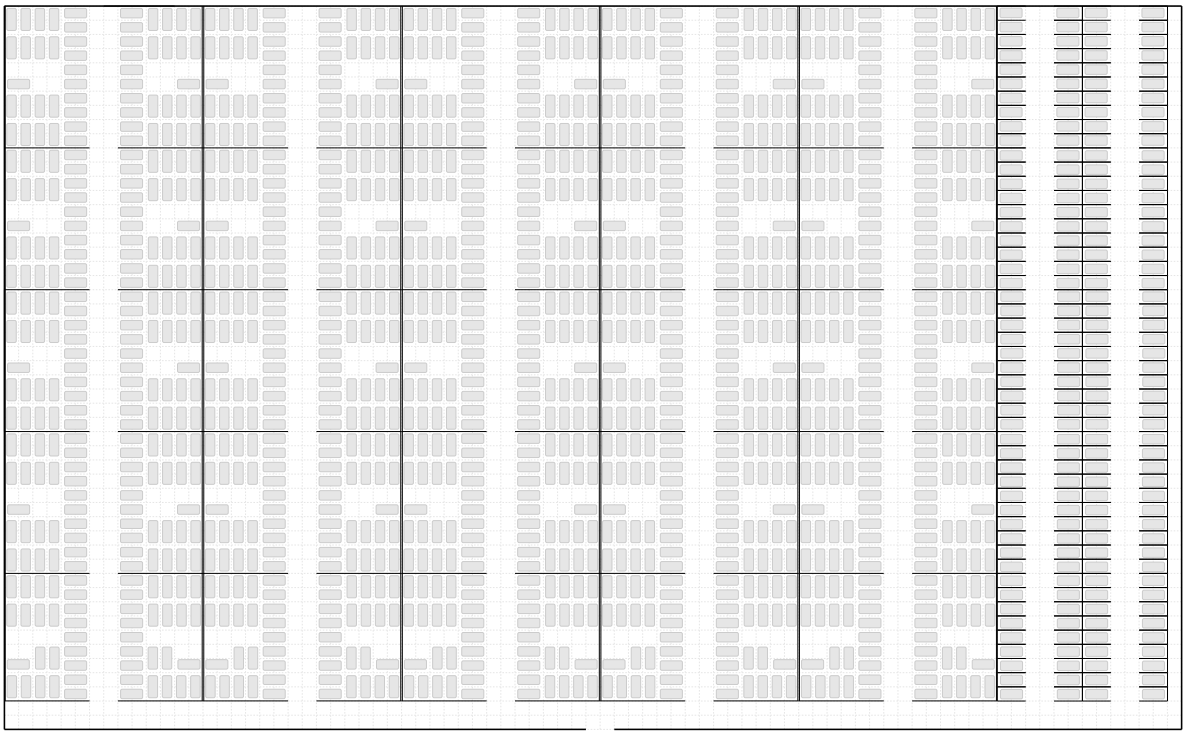


Figure 17: Modular design of the Cardinal Stadium lot B using 10×6 sub-lots. This design can accommodate 1516 cars. A car takes 82 seconds to exit the parking lot, on average.

it takes very long time. Based on priority, cars can be allocated these zones, depending on prioritization. In warehousing literature such a design is known as "class-based storage".

6. Conclusions

In this paper, we have developed a computational model to design a large size high density parking lot. This design process consists of two steps: i) design of a high density modular lot, which we call *sub-lot*, and ii) integration of these sub-lots into a large parking lot. We have conducted a set of experiments to determine which sub-lot size provide the best performance in terms of density and retrieval time. Key findings from this experiments are summarised as below:

- Sub-lots with larger area provides better benefit in terms of improvement in density.
- A sub-lot with $n = 4$ should be used when $m \geq 10$.
- The number of retrieval moves in a traditional design is same as driving lane retrieval moves in a sub-lot design.
- The size of the sub-lots is the most important design criterion to minimize vehicle relocation.

We have applied our method using a case study to the parking lot of the University of Louisville football stadium. We found that we can achieve 15% higher parking density using high-density modular design, but

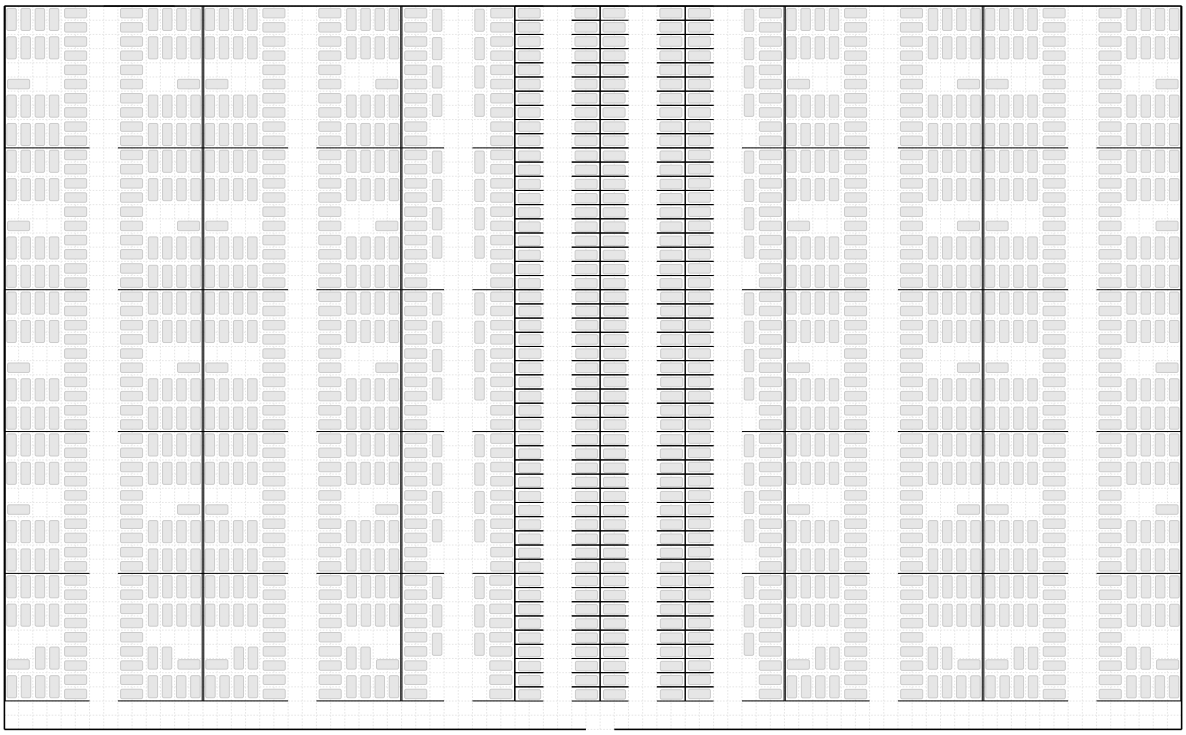


Figure 18: Class-based design of the cardinal lot B using 10×6, 10×3 sub-lots, and traditional column-blocks. This design can accommodate 1456 cars. A car takes 75 seconds, on average, to exit the parking lot.

with increased retrieval time. However, we want to admit that these results are "aspirational", as having thousands of connected and autonomous vehicles at one event is far into the future. Nonetheless, the high-density design on small lots could be realized in practice relatively soon.

References

M. Ferreira, L. Damas, H. Conceicao, P. M. d'Orey, R. Fernandes, P. Steenkiste, P. Gomes, Self-automated parking lots for autonomous vehicles based on vehicular ad hoc networking, in: Intelligent Vehicles Symposium Proceedings, 2014 IEEE, IEEE, 2014, pp. 472–479.

R. Nunes, L. Moreira-Matias, M. Ferreira, Using exit time predictions to optimize self automated parking lots, in: Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on, IEEE, 2014, pp. 302–307.

P. M. d'Orey, J. Azevedo, M. Ferreira, Exploring the solution space of self-automated parking lots: An empirical evaluation of vehicle control strategies, in: Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on, IEEE, 2016, pp. 1134–1140.

J. Azevedo, M. Macedo, P. M. d'Orey, M. Ferreira, High-density parking system enabled by vehicular networks, in: Vehicular Networking Conference (VNC), 2017 IEEE, IEEE, 2017, pp. 115–116.

J. Azevedo, P. M. D'orey, M. Ferreira, High-density parking for automated vehicles: A complete evaluation of coordination mechanisms, IEEE Access 8 (2020) 43944–43955.

J. Timpner, S. Friedrichs, J. van Balen, L. Wolf, k-stacks: High-density valet parking for automated vehicles, in: Intelligent Vehicles Symposium (IV), 2015 IEEE, IEEE, 2015, pp. 895–900.

H. Banzhaf, F. Quedenfeld, D. Nienhüser, S. Knoop, J. M. Zöllner, High density valet parking using k-deques in driveways, in: Intelligent Vehicles Symposium (IV), 2017 IEEE, IEEE, 2017, pp. 1413–1420.

M. Nourinejad, S. Bahrami, M. J. Roorda, Designing parking facilities for autonomous vehicles, Transportation Research Part B: Methodological 109 (2018) 110–127.

P. J. Siddique, K. R. Gue, J. S. Usher, Puzzle-based parking, Transportation Research Part C: Emerging Technologies 127 (2021) 103112.

L. M. Pohl, R. D. Meller, K. R. Gue, An analysis of dual-command operations in common warehouse designs, Transportation Research Part E: Logistics and Transportation Review 45 (2009) 367–379.