Where Should We Place LiDARs on the Autonomous Vehicle? - An Optimal Design Approach

Preprint · September 2018

CITATIONS
0

READS
293

3 authors:

Zuxin Liu
Carnegie Mellon University
4 PUBLICATIONS **17** CITATIONS
SEE PROFILE

Mansur Arief
Carnegie Mellon University
14 PUBLICATIONS **11** CITATIONS
SEE PROFILE

Ding Zhao
University of Michigan
84 PUBLICATIONS **563** CITATIONS
SEE PROFILE

Where Should We Place LiDARs on the Autonomous Vehicle? - An Optimal Design Approach

Zuxin Liu[1] , Mansur Arief[2] and Ding Zhao[2,*]

***Abstract*— Considering its reliability to provide accurate 3D views along with precise distance measures under highly uncertain driving conditions, automakers deems LiDAR a critical sensor for autonomous vehicle (AV) applications. However, its monumental cost restricts the practical implementations and surges many AV designers to best calibrate the sensor configurations to fully utilize its capability. This paper investigates the optimal LiDAR configuration problem to achieve such utility maximization purposes. The perception area and non-detectable subspace are used to construct the min-max optimization problem and a bio-inspired measure – volume to surface area ratio (VSR) – is proposed as an easy-to-evaluate cost function representing the notion of the size of the non-detectable subspaces of a given configuration. A cuboid-based approach is then adopted to show that VSR-based measure is a well-suited proxy for object detection rate and to yield tractable cost function computation using artificial bee colony evolutionary algorithm. Our experiments highlight the effectiveness of this approach to prescribe the cost-effective configuration of the LiDARs and to provide insightful analyses for automakers and suppliers to better design AV systems.**

I. INTRODUCTION

Critical to establishing safe driving for an autonomous vehicle (AV) is ensuring AV's ability to 'see' the surrounding. In essence, this ability well-affects how the AV perceives the environment and correspondingly plans for maneuvers. Extensive efforts have been dedicated to ultra-reliably guaranteeing AVs to have accurate 'seeing' capability, including installing industry-grade cameras and mounts such as Grashopper2, Flea2, etc. [1], [2], [3]. While this effort is amenable to achieve the purpose, it is highly dependent on the device configurations and even more on the external environmental situations, such as lighting conditions. An alternative or complementary approach to alleviate such external dependencies, which is widely adopted by various AV companies nowadays, is equipping LiDARs [4]. The major advantage is not only about the LiDARs' capability to provide distance measures and thus reconstructing 3D views but also its 360-degree coverage area up to a reasonable distance for assuring safe driving. In fact, some AV designs use multiple expensive LiDARs to achieve sensor redundancy and thus establishing even higher degree of 'seeing' capability [5].

It is noted, however, that such an alternative approach is generally costly and will skyrocket the price of the AV design, which are highly undesirable by both AV automakers and markets. Although the auto-suppliers have been trying to cut down the costs to make LiDARs more affordable for AV applications, the market price is quite substantial [6]. For instance a 16-beam Velodyne LiDAR, which is a relatively affordable design in the market currently, each costs about $8,000 [7]. Therefore, we are generally interested in optimally placing the expensive sensors that can reliably achieve some notion of a required 'seeing' capability to establish safe driving.

To the best of our knowledge, researches about optimal LiDAR configuration for AVs are rarely found in the literature. Some related works in the literature investigate how to achieve optimal configuration for camera or 3D sensors. For example, [8] proposed an optimal camera configuration method to achieve the largest field of view. [9] optimize the camera's pose for motion capture systems. However, these methods cannot be directly applied for optimal LiDAR configuration, because the receptive field of LiDAR is discrete, which greatly differ from cameras. In our previous work [10], we investigate a LiDAR configuration problem viewed as a min-max optimization with cylindrical representation as a proxy to the cost function and a mixed integer programming to solve the model. This method, however, is suffering from the curse of dimensionality and thus not suitable for large scale situations, which is typical when the design involves multiple LiDARs.

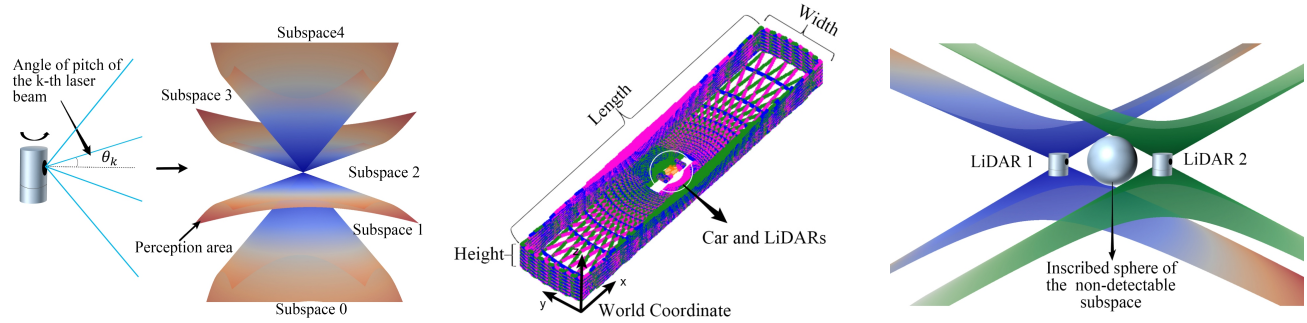
In this paper, we propose a novel approach to reformulate the cost function by utilizing an easy-to-solve proxy. That is, we use volume to surface area ratio (VSR), which has both geometric and bionic interpretations as we will elaborate in Sec. II-C as a measure to evaluate the performance of a particular configuration. The resulting solution allows us to further analyze trade-off between the 'seeing' capability and the design cost of an AV to balance the reliability and affordability of AV perception systems. It is worth noting that this approach even allows us to tractably evaluate more complex sensors designed specifically for autonomous systems applications such as the artificial compound eyes [11], [12].

The structure of the rest of this paper is as follows. Section II introduces the general framework to model the optimal LiDAR configuration problem. Section III describes the model and algorithm details and Section IV explains the experiment settings and results to show the effectiveness of the proposed approach. Finally, Section V provides a brief conclusion and possible future directions.

* Corresponding author: Ding Zhao. Email: (dingzhao@cmu.edu)

[1]Zuxin Liu is with the School of Instrumentation Science and Opto-electronics Engineering, Beihang University, China

[2]Mansur Arief and Ding Zhao are with the Department of Mechanical Engineering, Carnegie Mellon University, USA.



(a) Cone-like Perception area of LiDAR beams

(b) Region of interest, where the red, purple, green represent 3 different LiDARs' beams

(c) Non-detectable subspace

Fig. 1: Perception area, ROI and subspace demonstration

II. Methodological Framework

In this section, the structure and general idea of our approach is presented. Firstly, region of interest (ROI) and perception area of LiDAR beams are defined to simplify the problem. Secondly, non-detectable subspace is introduced and then the maximum VSR is proposed as the cost function. Finally, Artificial Bee Colony (ABC) algorithm is adopted to solve the resulting optimization problem.

A. ROI and perception area of LiDAR beams

We account for the limitation of LiDAR detection range by considering the region that are far away from the car not being influenced by the LiDAR placement. This consideration aligns with many real-world applications in where the perception algorithm choose only part of the LiDAR sensor data as its input. For example, VoxelNet [13] use the point cloud data within a certain bounding box around the LiDAR as the input to perform object detection task. With similar spirit, we focus our attention to the ROI around the AV in the LiDAR configuration problem, and the goal is to obtain as much information as possible from it, which can be achieved by well-covering the whole ROI with LiDAR beams.

To make it concrete, let ROI be defined by three-dimensional geometric properties – the length, width and height – of a cuboid in the world coordinate. Fig. 1(b) illustrate an example, where the car is located at the bottom-center of the x-y plane of the ROI. As such, the perception area can be defined as the whole trajectory of LiDAR beams after a 360-degree rotation, i.e. each beam forming a surface. It is noted that only the objects that are intersecting with the perception area has the chance to be detected.

To simplify the representation, we assume the beam will always provide a perception line along its trajectory. That is to say, when the LiDAR beams rotate 360-degree, the trajectory will form a cone. Therefore, the whole perception area of a multi-beams LiDAR can be modeled as the union of finitely many cones sharing the same vertex and vertical axis. Fig. 1(a) shows the cone-like perception area of the LiDAR beams, where θ_k represents the pitch angle of the k-th laser beam.

B. Non-detectable subspace segmentation

Given the ROI and the LiDAR configuration, the perception area of each LiDAR may intersect with each other, as well as the ROI boundaries. Therefore, the perception area will segment the ROI into many irregular shape subspaces. We define these segmented subspaces as non-detectable subspaces, because the object located within one of the subspaces could not be detected if the LiDAR and object are both static.

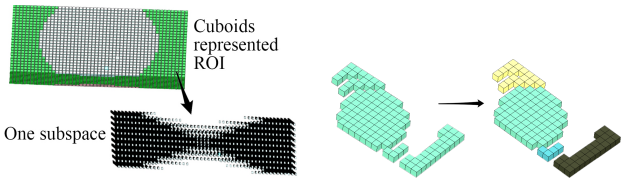
Fig. 1(c) shows an example of the non-detectable subspaces. Let N_l denote the number of LiDARs to be placed. Each LiDAR has N_b beams. Fig. 1(a) shows that one LiDAR could segment the ROI into N_b+1 subspaces. From bottom to top, each subspace could be denoted by a number in ascending order from 0 to N_b. Therefore, for N_l LiDARs, the maximum number of subspace N_s would be

$$N_s = (N_b + 1)^{N_l}. \tag{1}$$

Since the shape of these non-detectable subspace is different and irregular, it is very hard to represent them in an analytic way. Instead, we proposed a cuboid-based segmentation method, which will be elaborated in detail in Section III.

C. Cost function

To attain the most informative perception from the LiDAR, all of the non-detectable subspaces should be *as small as possible*. However the notions of how big or small a subspace is are somewhat vague because the shape is different with each other. Therefore, we use the radius of the inscribed sphere to define the size of the subspace, just as Fig. 1(c) shows. In that sense, the subspace of the largest size, i.e. the one with largest 'blindspot', is the one with the largest inscribed sphere. Thus, if one were interested to fix the worst case possible, then the optimization goal is to minimize the largest inscribed spheres from all subspaces. This means the optimal LiDAR configuration can be expressed as a min-max optimization problem. See [10] for more details of this particular approach. We note, however, it is very difficult to directly solve the inscribed sphere. Instead, we propose another easy-to-solve indicator, the volume to surface area



(a) Segment ROI into cuboids (b) BFS-based second-level segmentation result

Fig. 2: Subspace segmentation

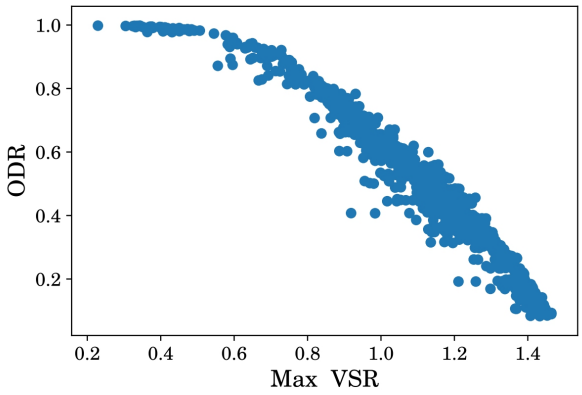


Fig. 3: General relationship between VSR and ODR

ratio (VSR), which is inspired by bionics, to define the size of the subspace.

From bionics perspective, VSR is a dominating factor in the cells shape and size [14], [15]. In high level, the cell need nutrition from outside environment through the cell membrane and the surface area represents its efficiency of absorbing nutrition while its volume represents the amount of nutrition it needs. If the cell is growing bigger, its volume is also growing bigger and so is its membrane surface area. Unfortunately, the rate of increase of the surface area is less than volume, and thus the absorption efficiently decreases in size growth, which explains why cell size is generally very small [14].

For our context, each non-detectable subspace can be viewed as a cell, and its surface area as the membrane. Loosely speaking, the object to be detected could be regarded as the nutrition. Thus, we want to achieve the smallest VSR possible to better detect an object. With this view, we can use VSR as a proxy for our cost function in representing the ability to detect small objects. More specifically, we are interested in mitigating the worst case scenario, i.e. we want to minimize the maximum VSR from the constructed non-detectable subspaces of the ROI.

We can also see VSR from the formula of radius of inscribed sphere of polyhedron [16]. If we know the volume, denoted as V, and surface area of the polyhedron, denoted as S, the radius of inscribed sphere R can be expressed as $R = 3 \times \frac{V}{S}$. Therefore, the VSR can represent the 'size' of any shape of our interest.

In real-world application, engineers care about the LiDAR configurations ability to detect object. Here object detected rate (ODR) is defined to represent the probability for the vehicle to detect a certain object.

Suppose some object of interest, which can also be represented as cuboids-based object, randomly appear within the ROI. Denote the number of subspace these cuboids occupied as N_{ds}. If $N_{ds} > thres$, where $thres$ is a predefined threshold, then the object is assumed to be detected. To evaluate a particular configuration in the following experiment, we simulate the object to randomly appear within the ROI M times and tally the occurrences T in which the object is detected. With this setting, the ODR can be calculated as $ODR = \frac{T}{M}$.

Thus minimizing the maximum VSR will increase ODR (see Fig. 3 for a general relationship between these measures). Therefore, we set the objective of the optimization model as minimization, which is then solved by the artificial bee colony algorithm.

D. Artificial Bee Colony optimization algorithm

In this paper, artificial bee colony (ABC) optimization algorithm [17] is used to minimize the cost function of the LiDAR configuration problem. ABC algorithm is a metaheuristics-type approach inspired by the behaviour of bee swarm in foraging. In such, the solutions correspond to food sources and the cost function corresponds to the nectar amount (i.e. the fitness) of a particular food source. A greedy selection policy is employed to ensure the algorithm converges quickly, while a roulette wheel selection method is adopted to explore around the food sources for nectar of the highest amount.

Technically, the swarm of bees are divided into three groups: employed bees, onlookers and scouts. The employed bees are used to measure the nectar amount of the food sources and each employed bee is corresponding with a food source. The onlookers adopt roulette wheel selection method to select the existing food sources to be further explored. If the quality of an existing food source has not been improved within a certain number of iterations, then the corresponding employ bee will abandon this food source and becomes a scout to search for a new food source randomly. See [18] for a comprehensive review on ABC algorithms.

III. Model Formulation

Since it is very hard to represent the subspaces in a compact analytical form, we adopt a cuboid-based method for approximation. The general idea is to segment (discretize) the ROI into many small cuboids. Then, all we need is only to determine which subspace these cuboids belong to. Fig. 2 summarizes the method pipeline in high level. First, segment the ROI into many cuboids and transform each cuboid from world global coordinate to LiDAR local coordinate. Then, conduct first-level segmentation and use base-n notation to represent each subspace. Finally, for each base-n notation represented subspace, use breadth-first search method to perform second-level segmentation.

A. Cuboid coordinate transformation

Denote θ_k as the k-th laser beam of a LiDAR, then the cone-like perception area could be formulated as

$$z_p = \tan\theta_k \sqrt{x_p^2 + y_p^2}, \tag{2}$$

where x_p, y_p, z_p are the coordinate points on the perception area relative to LiDAR coordinate system. However, it is difficult to formulate the cone in the world coordinate if the LiDAR rotates. Therefore, we transform the cuboid from world coordinate into LiDAR coordinate to facilitate the following segmentation.

Let $[x_w, y_w, z, \alpha_w, \beta_w, \gamma_w]$ denote the pose of a LiDAR in the world coordinate, where $[x_w, y_w, z]$ represent the position and $[\alpha_w, \beta_w, \gamma_w]$ represent the orientation. Then we get the rotation matrix $\boldsymbol{R_{\alpha\beta\gamma}}$ as

$$\boldsymbol{R_{\alpha\beta\gamma}} = \begin{bmatrix} c_\alpha c_\beta & c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma \\ s_\alpha c_\beta & s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma & s_\alpha s_\beta c_\gamma - c_\alpha s_\gamma \\ -s_\beta & c_\beta s_\gamma & c_\beta c_\gamma \end{bmatrix}, \tag{3}$$

where c represents $\cos$ and s represents $\sin$ in short. For example, c_α represents $\cos(\alpha)$. Denote by $\boldsymbol{T_{xyz}} = [x, y, z]^T$ the translation vector. Then transformation matrix from LiDARs coordinate to world coordinate $\boldsymbol{T_{wl}}$ can be expressed as

$$\boldsymbol{T_{wl}} = \begin{bmatrix} \boldsymbol{R_{\alpha\beta\gamma}} & \boldsymbol{T_{xyz}} \\ \boldsymbol{0}^T & 1 \end{bmatrix}. \tag{4}$$

Finally, suppose $\boldsymbol{X_w} = [x_w, y_w, z_w]$ and $\boldsymbol{X_l} = [x_l, y_l, z_l]$ as the coordinates of cuboid in world coordinate system and LiDAR coordinate system, respectively. Then, the transformation between the coordinate system satisfies

$$\boldsymbol{X_l} = \boldsymbol{T_{wl}^{-1}} \boldsymbol{X_w}. \tag{5}$$

Together, these systems can represent the cuboids in both LiDAR and world to set the stage for the segmentation processes, which we decompose into two levels: base-n first-level segmentation and breadth-first search (BFS) second-level segmentation.

B. Base-n notation first-level segmentation

After obtaining the cuboid coordinates in LiDAR coordinate, the next step is to determine which subspace the cuboid belongs to. Here base-n notation is used to identify the subspace.

Assume there are N_l N_b-beam LiDARs to be placed. According to the equation (1), each LiDAR could segment the ROI into N_b+1 subspace. The maximum number of the subspace is thus $(N_b+1)^{N_l}$. Therefore, each subspace could be represented by a base-n notation $d_1 d_2 ... d_i ... d_n$, where n equals LiDARs' number N_l, and d_i indicates the number of subspace of the i-th LiDAR, ranging from 0 to N_b.

Denote θ_{ik} as the pitch angle of the k-th beam of the i-th LiDAR, where $k \in 0, 1, ..., N_b$. $[x_{il}, y_{il}, z_{il}]$ is the cuboid coordinates in the i-th LiDAR coordinate system. To determine the digit d_i, we need to find the unique beam k that satisfies

$$\begin{cases} z_{il} \geq \tan\theta_{i(k-1)} \sqrt{x_{il}^2 + y_{il}^2}, \\ z_{il} < \tan\theta_{ik} \sqrt{x_{il}^2 + y_{il}^2} \end{cases} \quad 1 \leq k \leq N_b - 1 \tag{6}$$

or

$$z_{il} < \tan\theta_{ik} \sqrt{x_{il}^2 + y_{il}^2}, \qquad k = 0 \tag{7}$$

or

$$z_{il} \geq \tan\theta_{ik} \sqrt{x_{il}^2 + y_{il}^2}, \qquad k = N_b \tag{8}$$

Then, the digit $d_i = k$ and thus, iterating over all the LiDARs allow us to determine the cuboids' subspace in the ROI.

C. Breadth-first search (BFS) based second-level segmentation

Each cuboid in the ROI can be represented by a base-n notation of the subspace. The cuboids of the same base-n notation are thus the non-detectable subspace segmented by the laser beams. However, the segmentation is not thorough. For example, the right side of Fig 2(b) shows an example of the cuboids of the same base-n notation. Therefore, a BFS based method is used to perform the second-level segmentation.

The cuboid that has an overlap face with the root node is classified into the same group of the root cuboid. After search all the non-visited candidate cuboids, the root node cuboid would be marked as visited. Then, implement the same process for the left non-visited candidate cuboids until all the cuboids are marked as visited. The left side of Fig 2(b) shows the result of the second-level segmentation.

D. Optimization model

To construct the optimization problem, first denote by N_l the number of LiDAR and N_b the number of laser beams. Let i-th LiDAR configuration be denoted as $C_i = [x_i, y_i, z_i, \alpha_i, \beta_i, \gamma_i]$, where $\alpha_i, \beta_i, \gamma_i$ each represents the yaw, pitch and roll of the LiDAR in world coordinate, respectively.

Since the LiDAR configuration C is limited by the installation position of the car, define $C_{\min} = [x_{\min}, y_{\min}, z_{\min}, \alpha_{\min}, \beta_{\min}, \gamma_{\min}]$ and $C_{\max} = [x_{\max}, y_{\max}, z_{\max}, \alpha_{\max}, \beta_{\max}, \gamma_{\max}]$ which represent some lower- and upper-bound of the LiDAR configurations, respectively. As such, the i-th configuration C_i is valid if it satisfies the physical constrain:

$$C_{\min} \leq C_i \leq C_{\max}. \tag{9}$$

Let $CN = \{C_1, C_2, ..., C_{N_l}\}$ be the configuration set. Further, if we let the VSR of the j-th subspace be denoted as $Loss_j(CN)$ for any given configuration set CN and the set of cuboids coordinate in j-th subspace be denoted as S_j, then

$$Loss_j(CN) = \frac{V(S_j)}{SA(S_j)}, \tag{10}$$

where $V(S_j)$ and $SA(S_j)$ is the volume and surface area of the S_j cuboid set, respectively. Solving the $V(S_j)$, just needs

to calculate the number of the cuboids. Denote the cuboid's edges' length along x,y,z axis are $[e_x, e_y, e_z]$, the number of cuboids in S_j is n_j. The $V(S_j)$ could be solved by:

$$V(S_j) = e_x e_y e_z n_j \tag{11}$$

The surface area $SA(S_j)$ is made up of 3 parts: the surface area along $x-y$ plane SA_{xy}, along $x-z$ plane SA_{xz} and along $y-z$ plane SA_{yz}. We just need to count the overlap faces along the different plane and know the size of the cuboid set. The algorithms to solve the 3 parts are the same, so here we take SA_{xy} as an example. Denote S as the cuboid set, S_{ix}, S_{iy}, S_{iz} as the $i-th$ cuboid's index along x, y, z axes. The steps are shown in Algorithm 1.

Algorithm 1 Solving the Surface Area of the Cuboid Set

Require: Cuboid set, S; Cuboid resolution, $[e_x, e_y, e_z]$;
Ensure: The surface area along $x-y$ plane, SA_{xy};
1: Sort the coordinates of cuboids in S according to $x-y-z$ axes in an increasing order
2: $sz = Size(S)$, $cnt = 0$, $i = 1$
3: **repeat**
4: **if** $S_{ix} = S_{(i-1)x}$ and $S_{iy} = S_{(i-1)y}$ and $S_{iz} = S_{(i-1)z} + 1$ **then**
5: $cnt = cnt + 1$
6: **end if**
7: $i = i + 1$
8: **until** $i = sz$
9: $SA_{xy} = 2(sz - cnt)e_x e_y$

Thus, the optimization problem can be written as

$$\begin{aligned} \overline{CN} = \arg\min_{CN} \max_{j} \quad & Loss_j(CN) \\ \text{subject to} \quad & C_{\min} \le CN \le C_{\max} \ , \end{aligned} \tag{12}$$

where $\overline{CN}$ represents the optimized configuration solution.

To solve (12), we deploy ABC algorithm. We generate a swarm of artificial bees to explore the solution space. Let $x_i = (x_{i1}, ..., x_{i2}, ..., x_{id})$ $(i = 1, 2, , \tau)$ denote the solutions, where τ represents the number of bees and d represents the solutions dimension. Denote by $fit(x_i)$ the quality of the solution x_i, which is negatively related with our cost function. With probability P_i defined as

$$P_i = \frac{fit(x_i)}{\sum_{i=1}^{\tau} fit(x_i)}, \tag{13}$$

the onlookers select the solution x_i and examine its quality $fit(x_i)$. Next, the algorithm prescribe a new solution v_i near the currently selected solution x_i by computing

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}), \tag{14}$$

where $k \in \{1, 2, ..., \tau\}$ and $j \in \{1, 2, ..., d\}$ are randomly chosen indices. Note that ϕ_{ij} is a random number in the range $[-1, 1]$. If the $fit(v_i) > fit(x_i)$, then the solution x_i is replaced with, i.e. the bees move to, v_i. Adopting the analogy described in Section II-D, the iterative steps of the ABC algorithm are summarized in Algorithm 2.

Algorithm 2 Artificial Bee Colony (ABC) Algorithm

Require: Number of bees, τ; Iteration number, $IterMax$; Threshold to abandon a food source, $thres$;
Ensure: The optimal solution, x_i, $i = 1, ..., \tau$;
1: Randomly initialize the food source x_i, $i = 1, ..., \tau$
2: Get the quality of each food source $fit(x_i)$
3: Set $cycle = 0$ and $cnt_i = 0$, $i = 1, ..., \tau$
4: **repeat**
5: **for** each food source x_i **do**
6: Generate new food source v_i using equation (14)
7: **if** $fit(v_i) > fit(x_i)$ **then**
8: Replace x_i with v_i
9: **else**
10: $cnt_i = cnt_i + 1$
11: **end if**
12: **end for**
13: **for** each onlooker **do**
14: Select a food source using a roulette wheel selection method based on equation (13)
15: Generate new food source v_i around x_i using equation (14)
16: **if** $fit(v_i) > fit(x_i)$ **then**
17: Replace x_i with v_i
18: **else**
19: $cnt_i = cnt_i + 1$
20: **end if**
21: **end for**
22: **for** each food source x_i **do**
23: **if** $cnt_i = thres$ **then**
24: Replace x_i with a randomly generated solution
25: **end if**
26: **end for**
27: $cycle = cycle + 1$
28: **until** $cycle$ equals $IterMax$

IV. Experiment and Discussion

Consider an ROI of size $[60, 20, 4]$ in meter, excluding the rectangular region $x \in [27, 33], y \in [8, 12], z \in [0, 4]$, where such region is very close to the car and will not be considered in the perception algorithm at this point. The lower- and upper-bound of the LiDARs' pose are $[28, 9, 2.2, 0, 0, 0]$ and $[31, 11, 3, 3.1415, 3.1415, 0]$, where the first three components are in meter and the last three are in radian measure. The yaw angle will not be optimized because the LiDAR will rotate $360°$.

Assume we have several Velodyne VLP-16 LiDARs [7] to be configured. For tractability, we set the resolution of the cuboids to be $[1, 0.5, 0.2]$. To solve the problem, we set 200 bees and use ABC algorithm to iterate 800 times. Fig. 4 shows how the ABC algorithm converges in the 4 LiDARs case. The red line represents the average value of all the bees' max VSR, and the blue line represents the best solution's max VSR in each iteration. It can be seen that the search area of the bees will converge to the regions associated with lower cost values. The optimized LiDAR configuration solutions of

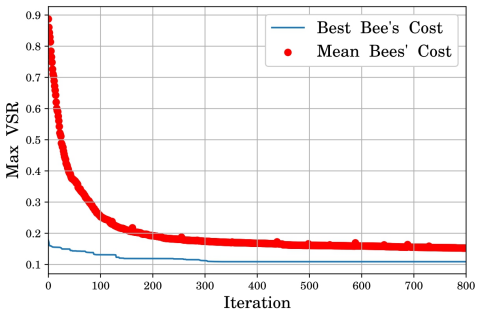


Fig. 4: The convergence trend of ABC solutions for minimizing the cost function - the max VSR

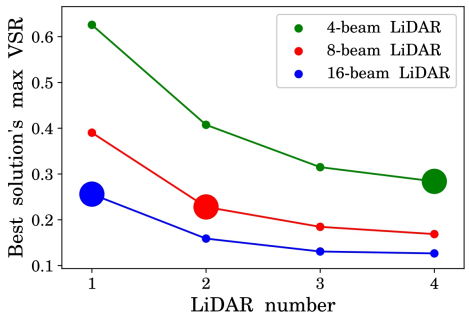
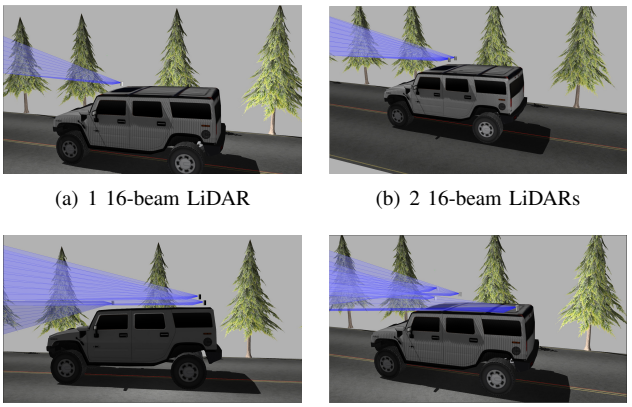


Fig. 6: Best solution's maximum VSR of different LiDAR configuration after 800 iterations



(a) 1 16-beam LiDAR (b) 2 16-beam LiDARs

(c) 3 16-beam LiDARs (d) 4 16-beam LiDARs

Fig. 5: LiDAR configuration visulization

1, 2, 3, 4 LiDARs are depicted in Fig. 5.

A. Maximum VSR trend in the number of LiDARs

Given the type of LiDAR, AV designers may care about the suitable number of LiDARs to place. Fewer LiDARs may influence the detection performance while too many LiDARs may cause information redundancy and increase the burden of computing resources. Here we showcase the application of our approach in AV design, especially in determining the number and the type of LiDARs to place. Suppose there are three types of LiDARs to choose: 4-beam, 8-beam, 16-beam LiDAR, whose angles of pitch are all evenly distributed from $-15°$ to $15°$. Running the proposed method, one shall yield the optimal configuration and compare the detection uncertainty performance for different number of LiDARs. In Fig. 6, we plot the results after 800 iterations.

It can be seen that the marginal improvement rate will decay as the number of LiDAR increases. This implies, the LiDAR beams' space configuration, which can also be interpreted as the ability to detect small object, may not increase when the number of LiDAR reaches a certain value. Besides, the optimal configuration plays a significant role when an AV design employs a few number of sensors, which is indeed the case when one considers using LiDARs or other expensive sensors, such as the artificial compound-eye [12]. Furthermore, we note that when the configuration is not optimized, the maximum VSR of the AV design would be larger, or the same at best, implying dominated performance compared to the solution prescribed by the proposed method.

B. Comparison between different LiDAR types

The proposed method could also help the AV engineer select the best LiDAR types, or help the LiDAR manufacturer design the LiDAR. For example, it is natural to wonder how would one 16-beam LiDAR perform compared to two 8-beam LiDARs and four 4-beam LiDARs (notice that the total number of beams are equal in all scenarios). From the marked points in Fig. 6, one shall immediately see, two 8-beam LiDARs would outperform the other two choices, *if configured optimally*. Intuitively, four 4-beam LiDARs should be the best because the configuration is more flexible. However, the physical properties of the 4-beam LiDAR (e.g. the angles of pitch of the beams) may limit its performance. This sort of analysis will be highly advantageous for AV designer to better achieve affordable and safe AV design, and for LiDAR manufacturer better design the LiDAR type.

V. CONCLUSIONS

We propose a novel approach to solve the optimal LiDAR configuration problem to help AV designers best utilize the expensive LiDAR sensors. By exploiting the overall relationship between the size of non-detectable subspaces (i.e. the blindspot of a given LiDAR configuration), the volume to surface area ratio (SVR), and the object detection rate (ODR), we show that the VSR-based measure is well-suited to solving optimal LiDAR configuration problem. Particularly, we deploy artificial bee colony (ABC) algorithm after segmenting the subspaces of region of interest (ROI) into a cuboid-based representation, that yields efficient computational performance. The experiment results indicate the effectiveness of the proposed approach in prescribing the configuration that attains appealing detection performance. The code and more extensive results will be made available through github repository. To improve the applicability we will consider the occlusion problems in the future work to

allow us moving one step closer toward a safe, reliable, and affordable AV mass deployment.

REFERENCES

[1] W. Maddern, A. Stewart, C. McManus, B. Upcroft, W. Churchill, and P. Newman, "Illumination invariant imaging: Applications in robust vision-based localisation, mapping and classification for autonomous vehicles," in *Proceedings of the Visual Place Recognition in Changing Environments Workshop, IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China*, vol. 2, 2014, p. 3.

[2] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[3] C. Häne, L. Heng, G. H. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys, "3d visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection," *Image and Vision Computing*, vol. 68, pp. 14–27, 2017.

[4] R. W. Wolcott and R. M. Eustice, "Visual localization within lidar maps for automated urban driving," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 176–183.

[5] B. Schwarz, "Lidar: Mapping the world in 3d," *Nature Photonics*, vol. 4, no. 7, p. 429, 2010.

[6] N. Jayaweera, N. Rajatheva, and M. Latva-aho, "Autonomous driving without a burden: View from outside with elevated lidar," *arXiv preprint arXiv:1808.08617*, 2018.

[7] Velodyne. [Online]. Available: https://velodynelidar.com/vlp-16.html

[8] J. Dybedal and G. Hovland, "Optimal placement of 3d sensors considering range and field of view," in *Advanced Intelligent Mechatronics (AIM), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1588–1593.

[9] J. E. Banta and M. A. Abidi, "Autonomous placement of a range sensor for acquisition of optimal 3-d models," in *Industrial Electronics, Control, and Instrumentation, 1996., Proceedings of the 1996 IEEE IECON 22nd International Conference on*, vol. 3. IEEE, 1996, pp. 1583–1588.

[10] S. Mou, Y. Chang, W. Wang, and D. Zhao, "An optimal lidar configuration approach for self-driving cars," *arXiv preprint arXiv:1805.07843*, 2018.

[11] D. Floreano, R. Pericet-Camara, S. Viollet, F. Ruffier, A. Brückner, R. Leitel, W. Buss, M. Menouni, F. Expert, R. Juston, *et al.*, "Miniature curved artificial compound eyes," *Proceedings of the National Academy of Sciences*, vol. 110, no. 23, pp. 9267–9272, 2013.

[12] D. Floreano and R. J. Wood, "Science, technology and the future of small autonomous drones," *Nature*, vol. 521, no. 7553, p. 460, 2015.

[13] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," *arXiv preprint arXiv:1711.06396*, 2017.

[14] W. M. Lewis, "Surface/volume ratio: implications for phytoplankton morphology," *Science*, vol. 192, no. 4242, pp. 885–887, 1976.

[15] L. K. Harris and J. A. Theriot, "Surface area to volume ratio: A natural variable for bacterial morphogenesis," *Trends in microbiology*, 2018.

[16] E. Weisstein. Insphere, from mathworld–a wolfram web resource. [Online]. Available: http://mathworld.wolfram.com/Insphere.html

[17] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm," *Journal of global optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[18] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, "A comprehensive survey: artificial bee colony (abc) algorithm and applications," *Artificial Intelligence Review*, vol. 42, no. 1, pp. 21–57, 2014.