LSwarm: Efficient Collision Avoidance for Large Swarms With Coverage Constraints in Complex Urban Scenes

Senthil Hariharan Arul, Adarsh Jagan Sathyamoorthy, Shivang Patel, Michael Otte, Huan Xu, Ming C. Lin, and Dinesh Manocha

***Abstract*—In this letter, we address the problem of collision avoidance for a swarm of UAVs used for continuous surveillance of an urban environment. Our method, LSwarm, efficiently avoids collisions with static obstacles, dynamic obstacles and other agents in three-dimensional urban environments while considering coverage constraints. LSwarm calculates collision avoiding velocities that maximize the conformity of an agent to an optimal path given by a global coverage strategy and ensure sufficient resolution of the coverage data collected by each agent. Our algorithm is formulated based on optimal reciprocal collision avoidance and is scalable with respect to the size of the swarm. We evaluate the coverage performance of LSwarm in realistic simulations of a swarm of quadrotors in complex urban models. In practice, our approach can compute collision avoiding velocities for a swarm composed of tens to hundreds of agents in a few milliseconds on dense urban scenes consisting of tens of buildings.**

***Index Terms*—Collision Avoidance, Path Planning for Multiple Mobile Robots or Agents, Simulation and Animation.**

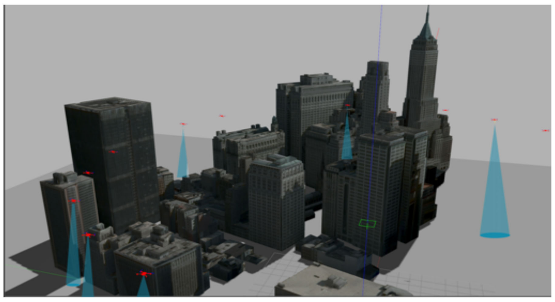


Fig. 1. Simulation of 30 quadrotors surveying our “High-Dense” urban environment using LSwarm for collision avoidance with agents, dynamic obstacles and static obstacles while considering constraints on coverage.

I. Introduction

RECENT advances in multi-rotor UAVs have created new areas of application, including surveillance, search and rescue, and monitoring. Continuous surveillance involves gathering sensory data (such as from a camera) from all regions in an area or volume of interest by traversing a path optimized for maximum coverage while accounting for static and dynamic obstacles [1]. For applications such as surveillance, using a single UAV/agent may not be effective when time, battery capacity, reliability, and coverage performance are considered.

A solution to this problem is to use a swarm of agents to perform surveillance, where each agent follows an optimal path provided by a global coverage strategy that maximizes the covered area or the information gathered from the environment. We use “optimal coverage path” or “global coverage path” to refer to this global path in this letter. Many techniques have been proposed that have formulated this optimal coverage path for multi-agent systems, while considering collision avoidance [2]–[6]. However, existing algorithms either ignore collisions between the swarm agents and dynamic obstacles in the environment [2], [4], or do not provide solutions that can be scaled to hundreds of agents.

Moreover, making an agent hover to avoid collision would fail when multiple dynamic obstacles approach head-on. When an agent encounters large numbers of dynamic obstacles, a haphazard maneuver to avoid collisions will again lead to loss of valuable data on the ground. In addition, the resolution of the gathered data must be considered to ensure its usefulness.

Main Contributions: We present LSwarm, a local collision avoidance method for quadrotor swarms performing continuous surveillance of large, complex 3-D urban environments (see Fig. 1). LSwarm builds on ORCA [7] (a Velocity Obstacle based collision avoidance method), and to the best of our knowledge, it is the first implementation that considers all of the following:

a) Collision avoidance with agents & dynamic obstacles,
b) Collision avoidance with static obstacles,

Manuscript received February 24, 2019; accepted June 26, 2019. Date of publication July 22, 2019; date of current version August 2, 2019. This letter was recommended for publication by Associate Editor M. Morales and Editor N. Amato upon evaluation of the reviewers’ comments. This work was supported in part by the Defense Advanced Research Projects Agency cooperative agreement HR00111820028 as part of DARPA OFFSET, in part by the Army Research Office under Grant W911NF-19-1-0069, and in part by the Intel. *(Corresponding author: Adarsh Jagan Sathyamoorthy.)*

S. H. Arul and A. J. Sathyamoorthy are with the Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: sarul1@umd.edu; asathyam@umd.edu).

S. Patel, M. Otte, and H. Xu are with the Department of Aerospace Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: spatel43@umd.edu; otte@umd.edu; mumu@umd.edu).

M. C. Lin is with the Department of Computer Science, University of Maryland, College Park, MD 20742 USA (e-mail: lin@cs.umd.edu).

D. Manocha is with the Department of Computer Science, and Electrical and Computer Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: dm@cs.umd.edu).

This letter has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/LRA.2019.2929981

2377-3766 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

Authorized licensed use limited to: University of Maryland College Park. Downloaded on May 05,2023 at 15:59:40 UTC from IEEE Xplore. Restrictions apply.

c) Scalability to large number of agents,
d) Acceleration Limits (Dynamics constraint),
e) Constraints on coverage,
f) Uncertainty in position and velocity.

Our coverage constraints consider the following:

(i) Each agent conforms maximally to an optimal path given by a global coverage strategy during collision avoidance. In other words, the calculated collision avoidance velocities for each agent also minimizes the coverage area loss (Section III.A)

(ii) The coverage data obtained by the agents' sensors have sufficient resolution (defined by Ground Sampling Distance (GSD)).

Our collision avoidance method is compatible with any global coverage strategy that provides waypoints. We use a simple lawnmower strategy to provide the global coverage path in this letter. We evaluate LSwarm's performance with respect to ORCA in urban environments which contain a plethora of static obstacles such as buildings of various heights, and dynamic obstacles in the form of birds, helicopters and other swarm agents. We perform accurate simulations of the quadrotor swarm in four different 3-D urban scenes. Depending on the model, the number of buildings vary between 10 to 80, and the height of buildings vary between 5 to 30 meters. Our algorithm takes milliseconds to compute collision-free velocities for 50 quadrotor agents on eight cores. We observe that LSwarm provides an improvement of up to a factor of two in coverage area over the use of ORCA, as the number of dynamic obstacles increases.

We first survey the previous work in coverage strategies, collision avoidance and modeling uncertainties in Section II. We formally define our coverage constraints and description of the global lawnmower strategy in Section III and present our local collision avoidance algorithm in Section IV. Section V presents the experiments and results of our implementation. Section VI summarizes the letter and suggests possible directions for future work. For a detailed discussion of all the concepts in this letter, refer to [8].

II. Previous Work

A recent survey on control, planning, and design of aerial swarm robots can be found in [9].

A. Swarm Control Strategies for Coverage Optimization

In this section, we give a brief overview of prior work on coverage optimization, and collision avoidance strategies.

Coverage Optimization: Urban area surveillance with added localization constraints, and swarm distribution optimization to cover high priority surveillance regions is presented in [4]. Julian *et al.* present a scalable consensus-based approximation method to generate trajectories to maximize the information gathered by a swarm in [2]. Dames *et al.* [10], track an unknown number of mobile targets using a team of robots which employ the Probability Hypothesis Density (PHD) filter to simultaneously estimate the number of targets and their positions. In [11], a distributed formulation of the PHD filter presented, which is combined with Lloyd's algorithm to control the robots' motions. Velagapudi *et al.* [6] present a distributed planner to explore moderately constrained environments that also accounts for communication constraints in large swarms in 2-D. A receding horizon planner for multi-robot coverage in partially known environments is presented by Das *et al.* in [12].

Collision Avoidance through precomputed trajectories: Saha *et al.* [3] demonstrate an incremental motion planning method to find the order of dispatch for robots in constricted environments, which could be useful to plan paths through dense urban environments. Turpin *et al.* [5] present a centralized trajectory generation algorithm with optimality guarantees along with a scalable decentralized case, assuming the absence of static obstacles. A centralized feedback control law for uniform and ergodic domain coverage using multi-agent systems is shown in [13]. Trajectory generation for hundreds of quadrotors in dense continuous environments using an inter-robot conflict annotated roadmap is demonstrated in [14]. A parallelization formulation for centralized trajectory generation methods for swarms is shown in [15].

In contrast to previous work, our method combines a global coverage strategy (based on space filling curves) with a local collision avoidance scheme (based on Velocity Obstacle) which avoids collisions on the fly. The global coverage strategy in our case accounts only for static obstacles and not the inter-agent collisions in the swarm. The decentralized local collision avoidance formulation can scale up to a large number of swarm agents, avoids having a single point of failure, and also accounts for the dynamic nature of the environment such as moving obstacles not known a priori.

B. VO-based Collision Avoidance

Velocity Obstacle (VO) [16] transforms the collision avoidance problem in the workspace to an equivalent formulation in the velocity space. Reciprocal Velocity Obstacles (RVO) [17] improved on VO to tackle robot-robot collision avoidance, where each agent computes its velocity based on the velocity of other agents. Optimal Reciprocal Collision Avoidance (ORCA) [7] further improved RVO by formulating a sufficient condition for multi-robots collision avoidance. ORCA was extended to 3-D in [18] and to include a proximity based static obstacle avoidance in [19]. Breitenmoser and Martinoli [20] combine multi-robot coverage and reciprocal collision avoidance and evaluate the new method's benefits in several scenarios. Recently, learning based methods such as Proximal Policy Optimization [21] show promising results for scalable decentralized collision avoidance. Although ORCA has several advantages since it is decentralized, and scalable, it assumes perfect knowledge of the position and velocity of each agent and does not model uncertainty.

C. Modeling Uncertainties

Many papers have modeled localization and sensing uncertainties to make VO based collision avoidance more practical. Hennes *et al.* present an Adaptive Monte Carlo localization-based solution (AMCL) [22] for modeling the pose of a ground robot. However, AMCL's dependence on odometry and laser data makes it hard to adapt for UAVs. The authors of PIXHAWK [23] have shown the feasibility of vision-based localization for

Authorized licensed use limited to: University of Maryland College Park. Downloaded on May 05,2023 at 15:59:40 UTC from IEEE Xplore. Restrictions apply.

UAVs and the use of stereo cameras to detect nearby obstacles. Snape *et al.* [24] and Kamel *et al.* [25] present Kalman filtering based methods to vary the safe distance between the agents based on position and velocity uncertainties. We use a Kalman filtering method similar to [24], [25] to handle uncertainties in localization and sensing.

III. Swarm Coverage in Urban Scenes

In this section, we formally define our constraints on coverage loss and resolution of the collected data. We state our assumptions and then describe the global lawnmower strategy that provides the optimal coverage path. LSwarm is compatible with any global method that generates waypoints for each agent, and in this letter we use a lawnmower strategy due to its simplicity.

A. Coverage Constraint Definition

As stated earlier, LSwarm's coverage constraints ensure that collision avoidance velocities are calculated such that coverage area loss is minimized and the resolution of collected data is satisfactory.

1) Coverage Area Loss: Each swarm agent's primary objective is to follow the optimal coverage path as closely as possible. This path provides the maximum coverage of the environment, and any deviation from this path would result in loss of coverage area and information. The following definitions apply to each quadrotor in the swarm.

Let $area_{pref}$ be the *preferred* area that would have been covered if a quadrotor followed the optimal coverage path perfectly for a given period of time. Let $area_{actual}$ be the *actual* area covered by the quadrotor. When the quadrotor does not face any obstacle while on the optimal coverage path, $area_{actual} = area_{pref}$. In cases where the quadrotor faces an obstacle, $area_{actual} \bigcap area_{pref} \leq area_{pref}$. Let $area_{overlap}$ denote the overlap area between $area_{pref}$ and $area_{actual}$. Then, $area_{overlap} = area_{pref} \bigcap area_{actual}$ and the overlap ratio is given as, $area_{overlap}/area_{pref}$. The coverage loss is then defined as $(1 - overlap\ ratio)$. When $overlap\ ratio = 1$, (optimal path is perfectly followed), the coverage loss is 0.

2) Sensor Resolution: We assume that each quadrotor in our swarm has a sensor (camera) to gather information from the environment. The sensor is modeled to have a conical view of the environment with a fixed apex angle. The radius r of the base circle of this view-cone would depend on the altitude h at which the quadrotor is flying as:

$$r = h \cdot \tan\theta \tag{1}$$

Where θ is half of the apex angle of the cone. We use a measure called Ground Sampling Distance (GSD) to impose resolution constraints on the gathered information. Since GSD is applied on rectangular images, we use the largest area square inside the base circle of the view-cone to calculate GSD as:

$$GSD_h = K_h \cdot h \text{ and } GSD_w = K_w \cdot h. \tag{2}$$

Where K_h and K_w are constants given as:

$$K_h = \frac{Sensor\ Height}{f \cdot Image\ Height} \text{ and } K_w = \frac{Sensor\ width}{f \cdot Image\ Width}. \tag{3}$$

The sensor height and sensor width here are the dimensions of the camera sensor and *f* is the focal length of camera. Since we are considering a square camera footprint, GSD_h is equal to GSD_w in our case. The higher the GSD value, the poorer the resolution.

B. Assumptions on Swarm Agents

We assume that the sensor attached to the agent always faces downwards irrespective of the orientation of the agent. This can be achieved by attaching the sensor to a gimbal. We also assume that each agent knows its own position and velocity, and can sense the positions and velocities of other obstacles around it, which could be noisy.

C. Global Coverage Strategy

Lawnmower sweep, a simple yet widely used strategy that guarantees 100% coverage of a search space [26], [27] is used to generate our optimal coverage path. Some implementations [26] segregate the obstacle from a non-obstacle region and perform a separate search sweep in each connected non-obstacle region, i.e., free space. Similar ideas have also been used in unknown terrain; for example, [27] uses a combination of boustrophedon and A* to, respectively, (a) perform a search sweep in unknown terrain, and (b) optimally traverse the known terrain.

Using lawnmower sweep in urban environments provides a unique set of challenges. Covering occluded areas where buildings are densely constructed is one challenge that is not typically addressed by the standard two-dimensional implementations of the approach. The occlusion problem can be mitigated by having agents adjust their height to fly over the buildings that create such occlusion. In other words, connections between different regions of ground-level search are possible by having agents perform a temporary change in altitude by flying over buildings.

1) Lawnmower Sweep: A mathematical formulation of the lawnmower sweep and sampling of free space are discussed in detail in [8].

2) Global Solution: Our global lawnmower solution experiments take the quadrotors' sensor model and a resolution measure for the sensor output into account and precompute the waypoints (as seen in Fig. 2) over the environment. We use a simple approach that extends a standard lawnmower sweep from 2-D to 3-D by determining the optimal flying altitude to obtain a good resolution in the output and the necessary altitude changes to avoid collisions with building along the sweep path, and then augmenting the height components of the path accordingly. It is possible to use more sophisticated methods, which calculate a sweep path that minimizes the number of altitude changes required over the entire search.

Authorized licensed use limited to: University of Maryland College Park. Downloaded on May 05,2023 at 15:59:40 UTC from IEEE Xplore. Restrictions apply.

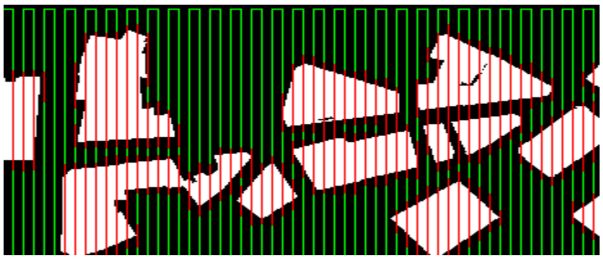


Fig. 2. Top-down view of Lawnmower waypoints over a city block. The white part represents the buildings (obstacles) and the black regions represent obstacle-free space. A part of lawnmower sweep is shown in green and red colors. Green color represents the optimal height of operation while the red color represents an elevated path to avoid buildings.

IV. Local Collision Avoidance With Coverage Constraints

In this section, we provide an introduction to ORCA and define our local collision avoidance method with coverage constraints. Refer to [8] for the definitions of the symbols used here.

A. *Optimal Reciprocal Collision Avoidance*

ORCA is a computationally fast algorithm that calculates collision avoiding velocities in real-time using linear programming. We provide ORCA's main result that is used to select collision avoiding velocities. The collision free ORCA velocity set for an agent A (given the position and velocity of an agent B) can be formulated as:

$$ORCA^{\tau}_{A|B} = \{v \,|\, (v - (v_A + \gamma u)) \cdot \hat{n} \geq 0\}, \tag{4}$$

where v_i is the current velocity of agent 'i', u is the vector from $v_A - v_B$ to the Velocity Obstacle boundary point and $\hat{n}$ is the outward normal at $(v_A - v_B) + u$ on the boundary of $VO^{\tau}_{A|B}$. γ is the responsibility factor for agent A in avoiding the collision. For collision avoidance between two agents (reactive obstacles), each take half of the responsibility for changing their velocities such that their relative velocities lie outside $VO^{\tau}_{A|B}$. Hence, $\gamma = \frac{1}{2}$ for reactive obstacles. In the case of non-reactive obstacles (e.g., buildings or birds), $\gamma = 1$ for the agent.

After computing the intersection of all pairwise ORCA sets to get $ORCA^{\tau}_{A}$, agent A selects its new velocity from it such that

$$v^{new}_{A} = \underset{v \in ORCA^{\tau}_{A}}{\text{argmin}} \,||\, v - v^{pref}_{A} ||. \tag{5}$$

For a detailed discussion of ORCA refer to [7].

B. *Collision Avoidance With Static Obstacles*

In dense urban scenarios, the quadrotors might be required to maneuver close to buildings. Even though the global coverage path accounts for static obstacles, any deviation from the global path during collision avoidance may cause the quadrotor to collide with the buildings (Fig. 3). To prevent such collisions, our method accounts for the static obstacles using proximity queries [19]. Each quadrotor continuously computes its proximity to static obstacles, like the buildings in an urban scene. The closest

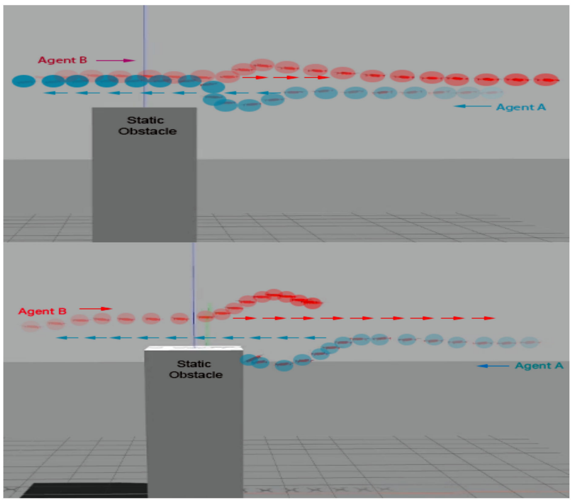


Fig. 3. In this scenario, both agents have a collision free global path but reactive collision avoidance deviates agent towards the building. **[Top]** LSwarm avoids static obstacle while **[Bottom]** ORCA suffers Collision. Arrows indicate the preferred path and more recent time steps are indicated with spheres of higher color intensity.

point on the static obstacle is considered as a point obstacle and the Minkowski sum is calculated considering a small positive value for the radius (ϵ) of this closest point and the quadrotor sphere's radius (r).

$$R = r \oplus \epsilon. \tag{6}$$

We use the ORCA formulation for non-reactive obstacles as described in Section IV-A to select a new collision free velocity for the agent. Since this method uses only proximity queries, it can be easily incorporated in a physical quadrotor system using depth sensors.

Avoiding Deadlock: In ORCA, an agent A's preferred velocity v^{pref}_{A} at each time step is directed towards the next waypoint. In certain cases, with the deviation caused by collision avoidance, the quadrotor might be directed towards a building. Although the static obstacle avoidance would prevent any collision, v^{pref}_{A} might still be directed through the building. Since we consider only the closest point as obstacles, the quadrotor might reach a deadlock trying to follow this v^{pref}_{A}. To solve this issue, we update the v^{pref}_{A} as being directed to the closest point along the line segment connecting the previous and the next waypoints. This helps overcome potential deadlock scenarios.

C. *Dynamics Constraints*

ORCA assumes that agents can modify their velocity instantaneously. Quadrotors in the swarm may destabilize when there is a large change in their velocities, which results in large pitch or roll angles, that might topple the quadrotors. We prevent such scenarios by adding constraint on each quadrotor's maximum acceleration. The feasible velocity space for a quadrotor is given by:

$$v_{new} < v_{current} + a_{\max} \cdot t. \tag{7}$$

This formulation keeps the collision-free velocity space convex, thus ensuring fast computation time. The search for appropriate

Authorized licensed use limited to: University of Maryland College Park. Downloaded on May 05,2023 at 15:59:40 UTC from IEEE Xplore. Restrictions apply.

velocities can be performed using acceleration velocity obstacles [28].

D. Modeling Uncertainty

We use Kalman filtering to handle noise in the positions and velocities of all moving entities. We first estimate the means and the covariance matrices of the positions and velocities of all moving entities. The square root of the largest eigenvalue of an entity's position covariance matrix is used as a measure to increase the radius of the bounding sphere of that entity. This ensures that the agent remains within this augmented bounding sphere as long as the real position is within one standard deviation of estimated position. The VO is constructed using this bounding sphere and the resultant VO is augmented by the covariance of the velocity uncertainty. Thus, as the uncertainty in sensing increases, each agent takes a more conservative approach to avoid collisions.

E. Dynamic Collision Avoidance With Coverage Constraints

Our goal is to compute a velocity that avoids collision and detours the quadrotor in a path which minimizes the coverage loss. In this section, we provide details on the coverage loss formulation.

In the absence of dynamic obstacles, all quadrotors would fly between one waypoint to the next with their preferred velocities, at the optimal altitude computed by the lawnmower strategy. As mentioned previously, we assume a square footprint for the camera's coverage cone at any instant of time. Therefore, the area covered by a quadrotor at any time instant $t \in [0, \tau]$ can be given as:

$$\begin{aligned} area_{pref} &= Area(x, y, s, t) \\ &= \text{Square area of side s centered at } (x,\ y)\,. \end{aligned} \tag{8}$$

Where τ is the time horizon over which we calculate the area, $x = v_{pref_x} \cdot t,\ y = v_{pref_y} \cdot t,$ and $s = (h - v_{pref_z} \cdot t) \cdot \frac{\tan\theta}{\sqrt{2}}$.

Here, h is the initial altitude of the quadrotor at t = 0. The area covered over τ for which the velocity will be constant can be computed as:

$$\bigcup_{\tau} Area(x, y, s, t). \tag{9}$$

We sample the time horizon τ into smaller time steps and forward simulate the velocity v_{new} to calculate the area that would be covered if v_{new} is chosen as the new velocity. When dynamic obstacles are present, the quadrotors chooses a new velocity v_{new} according to (5), and x, y and s in (8) would be with respect to v_{new}.

$$area_{new} = \bigcup_{\tau} Area\left(v_{new_x} \cdot t,\quad v_{new_y} \cdot t, (h - v_{new_z} \cdot t) \cdot \frac{\tan\theta}{\sqrt{2}},\quad t\right) \tag{10}$$

The conformity to the global optimal coverage path automatically results in minimizing the loss of the coverage area. Conformity increases with an increase in overlap between the areas in (9) and (10), while also limiting the GSD below an upper threshold. Therefore, our objective function becomes:

$$\begin{aligned} &\max_{GSD \leq GSD_{\max}} (area_{overlap}) \\ &= \max_{GSD \leq GSD_{\max}} \left(area_{pref} \bigcap area_{new}\right) \end{aligned} \tag{11}$$

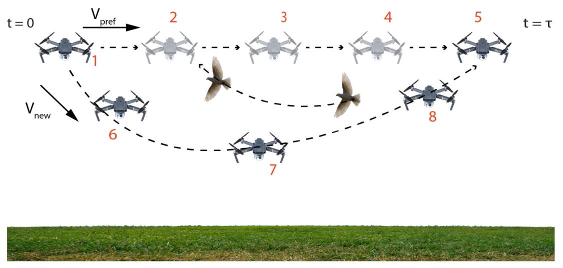


Fig. 4. Quadrotor deviating downward when avoiding a dynamic obstacle (a bird). The translucent quadrotors represent the path that would have been taken in the absence of the dynamic obstacle, when v = v_{pref}. The curve with the opaque quadrotors represents the path taken to avoid the obstacle when v = v_{new} and is computed by ORCA. The different states of the quadrotor are numbered. We highlight the coverage for these positions in Fig. 5.

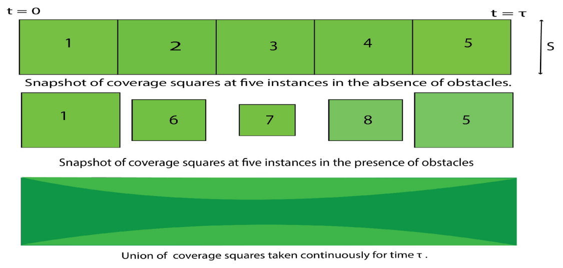

t = 0
t = τ
1
2
3
4
5
s
Snapshot of coverage squares at five instances in the absence of obstacles.
1
6
7
8
5
Snapshot of coverage squares at five instances in the presence of obstacles
Union of coverage squares taken continuously for time τ .

Fig. 5. The coverage areas for the numbers stated in Fig. 4 for different locations of the agent. We observe lowest coverage at location 7 and maximum coverage at locations 1 and 5. **[Top]** The preferred coverage $area_{pref}$, when v_{pref} is the velocity of the quadrotor. **[Middle]** The new coverage $area_{new}$, when v_{new} is chosen as the velocity of the quadrotor. **[Bottom]** The overlap area (shown in dark green) between $area_{pref}$ and $area_{new}$ when computed continuously for time τ. This overlap area needs to be maximized.

A collision avoidance scenario is shown in Fig. 4, and the corresponding coverage area for different states of the quadrotor are shown in Fig. 5. Overlap area can be computed by methods such as clipping, but performing clipping for each possible velocity, in each time-step for each agent would be computationally expensive.

F. LSwarm Acceleration

In order to reduce the computation burden and ensure scalability, LSwarm uses a Look Up Table (LUT) to select a collision avoiding velocity that maximizes (11). The LUT contains precomputed values for the overlap areas for various new velocities. At runtime, we select a new velocity based on its corresponding overlap area from this LUT and on certain conditions. The construction of the LUT and velocity selection are discussed in the following sections.

1) Constructing the Look Up Table: Let us consider the unit vector (1, 0, 0) and call it v_{unit}. This unit vector corresponds to the forward direction for the quadrotor in the quadrotor's frame

Authorized licensed use limited to: University of Maryland College Park. Downloaded on May 05,2023 at 15:59:40 UTC from IEEE Xplore. Restrictions apply.

of reference and can be transformed to any preferred velocity vector by using the standard 3 × 3 rotational transformation matrix **(R)** as,

$$v_{pref} = R_{trans} * v_{unit}. \tag{12}$$

For a time horizon τ, we calculate $area_{unit}$ corresponding to v_{unit} as given in (9) assuming a flying altitude of 5 m. We rotate v_{unit} (about Y and Z axes) in all possible forward directions to evaluate how such rotations affect the overlap area with $area_{unit}$. We first apply a rotation to v_{unit} about the Z-axis by an angle α as,

$$v_{\alpha} = R_{Z\alpha} * v_{unit}. \tag{13}$$

where $\alpha \in [-90^\circ, 90^\circ]$. For each angle of rotation α about the Z-axis, we again apply a rotation β about the Y-axis as,

$$v_{\alpha\beta} = R_{Y\beta} \cdot v_{\alpha}, \tag{14}$$

where $\beta \in [-90^\circ, 90^\circ]$. β is the angle which governs whether the quadrotor will move upward or downward.

Let us denote the coverage area for $v_{\alpha\beta}$ over time τ as $area_{\alpha\beta}$. The rotation of v_{unit} is continued until all values of α, and β are used by increments of 1°, and $v_{\alpha\beta}$ and the overlap between $area_{\alpha\beta}$ and $area_{unit}$ are recorded into the LUT. The fully constructed LUT has 32,761 × 5 entries.

2) Selecting New Velocity from LUT: When the quadrotor has to change its velocity to avoid an obstacle, we first use ORCA to compute a new velocity v_{new} as given in (5). The difference δ between v_{new} and v_{pref} is given as,

$$\delta = |v_{pref} - v_{new}| \tag{15}$$

Next, we search the LUT for $v_{\alpha\beta}$ such that

$$|v_{unit} - v_{\alpha\beta}| \geq \delta + \varepsilon \tag{16}$$

where ε is a small positive value. All the velocities which satisfy (16) are ranked based on their corresponding overlap area with $area_{unit}$. The Z-component of $v_{\alpha\beta}$ is used to compute the altitude change in the time horizon, and the velocities whose altitude change fail the GSD constraint are neglected. The rotation R_{trans} is applied to each shortlisted $v_{\alpha\beta}$ to transform it correspond to the preferred velocity orientation. Let us denote the transformed velocity as $v_{\alpha\beta}^{trans}$. It is computed as:

$$v_{\alpha\beta}^{trans} = R_{trans} \cdot v_{\alpha\beta}. \tag{17}$$

We check if each $v_{\alpha\beta}^{trans} \in ORCA_{quadrotor}^{\tau}$ which is the ORCA set for the quadrotor for time τ. The transformed velocity $v_{\alpha\beta}^{trans}$ with the highest rank that belongs to $ORCA_{quadrotor}^{\tau}$ is guaranteed to avoid collisions and such a $v_{\alpha\beta}^{trans}$ may or may not be equal to the new velocity given by ORCA. Thus, the coverage area resulting from choosing this velocity will always be greater than or equal to the coverage area resulting from the velocity given by ORCA.

Computed Behavior: ORCA computes a set of collision free velocity for an agent, from which the velocity that satisfies (5) is chosen. In contrast, LSwarm replaces (5) by selecting the velocity in (16) that satisfies GSD constraint and has maximum coverage overlap.

V. Results and Performance

In this section, we describe our implementation and highlight LSwarm's performance.

A. Implementation

LSwarm was implemented on an Intel Xeon octacore processor at 3.6 GHz with 32 GB memory and GeForce GTX 1080 GPU. For simulating the swarm of quadrotors, we used ROS Kinetic, Gazebo 7.14.0, along with the PX4 Software-In-The-Loop firmware.

1) LawnMower Strategy: LSwarm can handle arbitrary 3-D models. X-Y values of the global waypoints are computed using an occupancy grid constructed from the 2D footprint of the environment, while the Z values vary based on the building heights (Section III.C).

2) Local Collision Avoidance: ORCA is implemented using the RVO2-3D [18] library. In LSwarm, we compute the Euclidean separation distance for static collision avoidance using the Proximity Query Package (PQP) library [29]. We do not include a static obstacle avoidance as a part of ORCA in our comparisons. ORCA models static obstacles as line segments or planes, which is unsuitable for real world implementation. Hence, we test ORCA with the lawnmower strategy (which accounts for static obstacles) to test if it would be adequate to avoid collision with buildings.

B. Benchmarks

Table I shows the dimensions and complexity of different urban models used to evaluate our method, and the time taken to compute the lawnmower waypoints. For a model of area $\sim$6000 m^2, the lawnmower strategy took 2.5 seconds to generate the waypoints. The models themselves are shown in Figs. 1 and 6. The first word in the model name corresponds to the average height (high-rise/low-rise), and the second word signifies the density of buildings in the model. The 4 benchmarks with varying building heights and building densities (buildings/sq.m) capture the real world scenarios that an aerial surveillance system may face from metropolitan cities (High Dense) to suburbs (Low Sparse).

C. Performance Evaluation

In this section, we compare the static collision avoidance and coverage performance of LSwarm and ORCA. LSwarm provides the same agent-agent collision avoidance guarantees as ORCA and hence results pertaining to this case are not included in this letter.

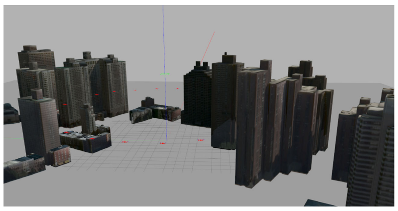
We evaluated the combined strategy (LSwarm or ORCA with lawnmower coverage) in the 4 benchmark models, using 20 quadrotors and 20 dynamics obstacles. From Table I, we observe that with higher building density and taller buildings, a few quadrotors collided with the buildings due to dynamic obstacle avoidance when ORCA was used. In contrast, LSwarm was effective in avoiding collisions with the buildings.

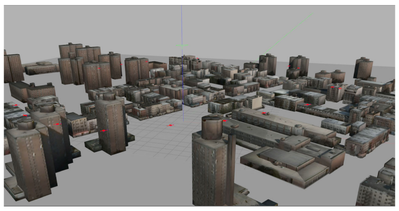
We compared LSwarm's coverage performance with ORCA's by making a quadrotor cover a 20 m straight line with dynamic

Authorized licensed use limited to: University of Maryland College Park. Downloaded on May 05,2023 at 15:59:40 UTC from IEEE Xplore. Restrictions apply.

TABLE I
WE PRESENT THE COMPLEXITY OF DIFFERENT BENCHMARK MODELS AND THE TIME TAKEN TO COMPUTE THE LAWNMOWER WAYPOINTS. WE TABULATE THE NUMBER OF COLLISIONS (WITH BUILDINGS) WHEN 20 QUADROTOR MANEUVER THE LAWNMOWER WAYPOINTS USING ORCA AND LSWARM, IN THE PRESENCE OF 20 DYNAMIC OBSTACLES

Environment Model	Dimension (in meters)	Number of Buildings	Global Path Calculation Time (ms)	Collisions with buildings	
				ORCA	LSwarm
High Dense	50.96 x 39.33 x 29.50	27	753	4	nil
High Sparse	56.25 x 53.03 x 14.25	16	1027	2	nil
Low Dense	64.26 x 53.80 x 12.5	79	1186	nil	nil
Low Sparse	96.67 x 62.92 x 7.2	23	2585	nil	nil


(a) **High Sparse:** Sparsely populated urban model with 16 skyscrapers


(b) **Low Sparse:** Sparsely populated model with 23 low-rise buildings

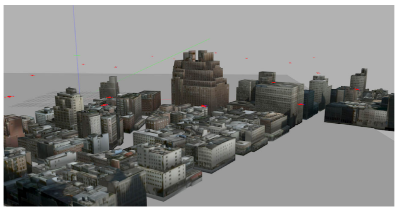

(c) **Low Dense:** Densely packed model with 79 low-rise buildings

Fig. 6. Benchmark urban environments. (a) **High Sparse:** Sparsely populated urban model with 16 skyscrapers. (b) **Low Sparse:** Sparsely populated model with 23 low-rise buildings. (c) **Low Dense:** Densely packed model with 79 low-rise buildings.

TABLE II
WE TABULATE THE OVERLAP RATIO (SATISFYING GSD CONSTRAINTS) FOR DIFFERENT COVERAGE RADII WHEN THE QUADROTOR USES ORCA AND LSWARM TO COVER A 20M STRAIGHT LINE WITH THE OBSTACLES APPROACHING FROM DIFFERENT DIRECTIONS. BOTH LSWARM AND ORCA AVOID ALL OBSTACLES

Number of Dynamic Obstacles	ORCA		LSwarm	
	Overlap Ratio (with required Resolution) 1m Coverage Radius	Overlap Ratio (with required Resolution) 3m Coverage Radius	Overlap Ratio (with required Resolution) 1m Coverage Radius	Overlap Ratio (with required Resolution) 3m Coverage Radius
Approaching from all directions				
10	0.7210	0.9235	0.8977	0.9355
25	0.6594	0.7040	0.8715	0.9197
40	0.5916	0.6331	0.6084	0.8875
Approaching from Left to Right				
10	0.7762	0.8879	0.9308	0.9448
20	0.3201	0.5374	0.8923	0.9379

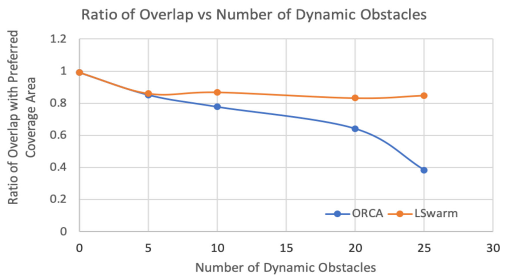


Fig. 7. The graph presents the ratio of overlap area vs. the number of dynamic obstacles while considering GSD constraints. For a small number of obstacles, both methods provide the same overlap. LSwarm however provides a better overlap for a large number of dynamic obstacles over ORCA.

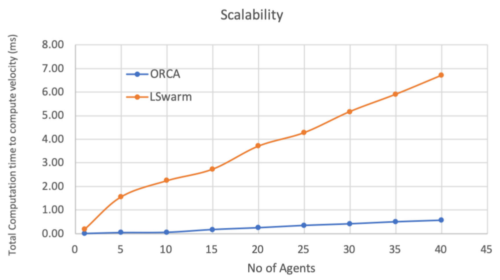


Fig. 8. Running time of LSwarm algorithm and ORCA vs. number of agents on a multi CPU-core. We observe nearly linear growth for LSwarm indicating scalability with the number of agents. As expected, ORCA takes less time to compute collision avoidance velocities.

obstacles moving (a) from all directions towards the quadrotor and (b) perpendicular to the quadrotors movement at the same altitude as the quadrotor. For satisfying the GSD constraint, we try to avoid increasing the quadrotor altitude by over 1.5 meters. From Table II, we observe that LSwarm offers better effective coverage overlap (satisfying GSD constraint) compared to ORCA, and this improvement is prominent when the camera's coverage radius is small. Furthermore, we tested the coverage performance when the dynamic obstacles move towards the quadrotor from all directions at various heights. From Fig. 7, we observe that with increasing number of dynamic obstacles, ORCA may compute collision free velocities that result in coverage data with unusable resolutions. LSwarm consistently produces velocities that lead to nearly optimal coverage with useful resolutions.

D. Benefits Over Prior Methods

Centralized trajectory generation can guarantee collision avoidance in known environments. But in time varying urban

Authorized licensed use limited to: University of Maryland College Park. Downloaded on May 05,2023 at 15:59:40 UTC from IEEE Xplore. Restrictions apply.

scenes, dynamic obstacles (such as birds) may enter the environment and collide with the quadrotors since they were not considered during centralized computation. LSwarm and ORCA, being local methods can effectively deal with such situations. LSwarm, in contrast to ORCA uses a more practical static collision avoidance and gives better coverage performance while maintaining scalability with respect to swarm agents (see Fig. 8).

VI. Conclusion, and Future Work

We present an efficient method for local collision avoidance under coverage constraints for large aerial swarms. We use an LUT to compute a velocity that provides collision-free trajectories and better coverage. LSwarm scales linearly with the number of swarm agents and takes few milliseconds for velocity computation for tens of swarm agents.

Our approach has some limitations. Currently, we use a simple Kalman filter to address the uncertainties in localization of agents/obstacles, whereas the uncertainty in actuation is not considered. Furthermore, the lawnmower algorithm assumes an exact representation of the urban environment. Our solution to the optimization formulation is approximate and does not provide guarantees with respect to global optimality. In the future, in addition to addressing these limitations, we would like to further evaluate our approach on complex urban models by coupling with other coverage strategies. It would be useful to develop more robust methods that can handle noise in the sensor measurements and account for uncertainty. Finally, we would like to accurately model the dynamics constraints of agents.

References

[1] E. Galceran and M. Carreras, “A survey on coverage path planning for robotics,” *Robot. Auton. Syst.*, vol. 61, no. 12, pp. 1258–1276, 2013.

[2] B. J. Julian, M. Angermann, M. Schwager, and D. Rus, “Distributed robotic sensor networks: An information-theoretic approach,” *Int. J. Robot. Res.*, vol. 31, no. 10, pp. 1134–1154, 2012.

[3] I. Saha, R. Ramaithitima, V. Kumar, G. J. Pappas, and S. A. Seshia, “Implan: Scalable incremental motion planning for multi-robot systems,” in *Proc. ACM/IEEE 7th Int. Conf. Cyber-Physical Syst.*, Vienna, 2016, pp. 1–10.

[4] M. Saska, V. Vonásek, J. Chudoba, J. Thomas, G. Loianno, and V. Kumar, “Swarm distribution and deployment for cooperative surveillance by micro-aerial vehicles,” *J. Intell. Robot. Syst., Theory Appl.*, vol. 84, no. 1–4, pp. 469–492, 2016.

[5] M. Turpin, N. Michael, and V. Kumar, “CAPT: Concurrent assignment and planning of trajectories for multiple robots,” *Int. J. Robot. Res.*, vol. 33, no. 1, pp. 98–112, 2014.

[6] P. Velagapudi, K. Sycara, and P. Scerri, “Decentralized prioritized planning in large multirobot teams,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, 2010, pp. 4603–4609.

[7] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, “Reciprocal n-body collision avoidance,” in *Proc. Robot. Res*. Springer, 2011, pp. 3–19.

[8] S. Arul *et al.*, “LSwarm: Efficient Collision Avoidance for Large Swarms with Coverage Constraints in Complex Urban Scenes,” 2019, *arXiv:1902.08379*.

[9] S. Chung, A. A. Paranjape, P. Dames, S. Shen, and V. Kumar, “A survey on aerial swarm robotics,” *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 837–855, Aug. 2018.

[10] P. Dames, P. Tokekar, and V. Kumar, “Detecting, localizing, and tracking an unknown number of moving targets using a team of mobile robots,” in *Proc. Springer Proc. Adv. Robot. Robot. Res.*, 2017, pp. 513–529.

[11] P. Dames, “Distributed multi-target search and tracking using the PHD filter,” in *Proc. Int. Symp. Multi-Robot Multi-Agent Syst.*, Los Angeles, CA, USA, 2017, pp. 1–8.

[12] S. N. Das and I. Saha, “Rhocop: Receding horizon multi-robot coverage,” in *Proc. ACM/IEEE 9th Int. Conf. Cyber-Physical Syst.*, Porto, 2018, pp. 174–185.

[13] G. Mathew and I. Mezić, “Metrics for ergodicity and design of ergodic dynamics for multi-agent systems,” *Physica D, Nonlinear Phenomena*, vol. 240, no. 4/5, pp. 432–442, 2011.

[14] W. Hönig, J. A. Preiss, T. K. S. Kumar, G. S. Sukhatme, and N. Ayanian, “Trajectory planning for quadrotor swarms,” *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 856–869, Aug. 2018.

[15] M. Hamer, L. Widmer, and R. D’andrea, “Fast generation of collision-free trajectories for robot swarms using GPU acceleration,” *IEEE Access*, vol. 7, pp. 6679–6690, 2019.

[16] P. Fiorini and Z. Shiller, “Motion planning in dynamic environments using velocity obstacles,” *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 760–772, 1998.

[17] J. Van Den Berg, M. Lin, and D. Manocha, “Reciprocal velocity obstacles for real-time multi-agent navigation,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Pasadena, CA, USA, 2008, pp. 1928–1935.

[18] J. Van Der Berg, S. J. Guy, J. Snape, M. C. Lin, and D. Manocha, “RVO2 Library: Reciprocal Collision Avoidance for Real-Time Multi-Agent Simulation,” May 2008. [Online]. Available: http://gamma.cs.unc.edu/RVO2/

[19] D. Alejo, J. A. Cobano, G. Heredia, and A. Ollero, “Optimal reciprocal collision avoidance with mobile and static obstacles for multi-UAV systems,” in *Proc. Int. Conf. Unmanned Aircr. Syst.*, Orlando, FL, USA, 2014, pp. 1259–1266.

[20] A. Breitenmoser and A. Martinoli, “On combining multi-robot coverage and reciprocal collision avoidance,” in *Proc. Springer Tracts Adv. Robot. Distrib. Auton. Robot. Syst.*, pp. 49–64, 2016.

[21] P. Long, T. Fanl, X. Liao, W. Liu, H. Zhang, and J. Pan, “Towards Optimally decentralized multi-robot collision avoidance via deep reinforcement learning,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Brisbane, QLD, Australia, 2018, pp. 6252–6259.

[22] D. Hennes, D. Claes, W. Meeussen, and K. Tuyls, “Multi-robot collision avoidance with localization uncertainty,” in *Proc. 11th Int. Conf. Auton. Agents Multiagent Syst.-Volume 1*, 2012, pp. 147–154.

[23] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, “PIXHAWK: A system for autonomous flight using onboard computer vision,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, 2011, pp. 2992–2997.

[24] J. Snape, J. V. D. Berg, S. J. Guy, and D. Manocha, “The hybrid reciprocal velocity obstacle,” *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 696–706, Aug. 2011.

[25] M. Kamel, J. Alonso-Mora, R. Siegwart, and J. Nieto, “Robust collision avoidance for multiple micro aerial vehicles using nonlinear model predictive control,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vancouver, BC, Canada, 2017, pp. 236–243.

[26] H. Choset and P. Pignon, “Coverage path planning: The boustrophedon cellular decomposition,” *Field Service Robot.*, pp. 203–209, 1998. [Online]. Available: https://link.springer.com/chapter/10.1007%2F978-1-4471-1273-0_32#citeas

[27] A. Ntawumenyikizaba, Hoang Huu Viet, and TaeChoong Chung, “An online complete coverage algorithm for cleaning robots based on boustrophedon motions and A* search,” in *Proc. 8th Int. Conf. Inf. Sci. Digit. Content Technol.*, Jeju, 2012, pp. 401–405.

[28] J. Van Den Berg, J. Snape, S. J. Guy, and D. Manocha, “Reciprocal collision avoidance with acceleration-velocity obstacles,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, May 2011, pp. 3475–3482.

[29] E. Larsen, S. Gottschalk, M. C. Lin, and D. Manocha, “Fast distance queries with rectangular swept sphere volumes,” in *Proc. IEEE Int. Conf. Robot. Autom. Symp. Proc.*, San Francisco, CA, USA, 2000, vol. 4, pp. 3719–3726.

Authorized licensed use limited to: University of Maryland College Park. Downloaded on May 05,2023 at 15:59:40 UTC from IEEE Xplore. Restrictions apply.