A Thorough Theoretical Analysis

Our goal is to prove that ordering the voxels by their Morton Code value can achieve the shortest Octree update runtime. Updating a voxel in the Octree corresponds to a round-trip memory visit from the root node to a leaf node. Thus, we can use the number of common ancestor nodes shared by each 2 neighboring voxels in the sequence to estimate how much data locality a voxel sequence can utilize when inserting them into the Octree. A higher number of common ancestors shared always implies a slower Octree update speed and vice versa.

We first prove that the theorem holds true for a uniform depth octree. Afterward, we prove for any non-uniform depth octree. Note that in a uniform depth octree, each leaf node has equal leaf-to-root path length, thus giving them equal length Morton Code.

Notations: Denote N leaf nodes in a perfect l-level Octree[10]: $a_1, a_2, ..., a_N$. Denote $A(a_1, a_2, ...a_n)$ as the closest common ancestor of leaf node $a_1, a_2, ..., a_n$. Denote $D(a, b)$ as the distance between leaf node a and node $A(a, b)$[11]. Denote S as an ordering of the N leaf nodes. Note that an l-level Octree divides the entire space into 8^{l-1} voxels [12]. Let each tree node be a set, whose elements are the voxels this node contains in the 3D space. The root node belongs to the l^{th} level and the leaf node belongs to the 1^{st} level. There is only one set $A_{n,1}$ on the l^{th} level, containing 8^{l-1} voxels. The 1^{st} level contains 8^{l-1} sets, denoted as $A_{1,1}, A_{1,2}, ..., A_{1,8^{l-1}}$ and each set contains only 1 finest level voxel. Each set above the 1^{st} level has 8 child sets corresponding to their 8 children in the Octree and has size 8 times the size of their child set.

Theorem A.1. *Ordering the leaf nodes $a_1, a_2, ..., a_n$ by the Morton Code of their 3D coordinates is one of the optimal sequence S^* that minimizes the following formula:*

$$\mathcal{F}(S) = D(a_1, a_2) + D(a_2, a_3) + ... + D(a_{N-1}, a_N)$$

The formula $\mathcal{F}(S)$ is the sum of each two consecutive leaf node's tree-based distance in sequence $a_1a_2, ..., a_n$. The update of a voxel with the finest resolution corresponds to a round-trip memory visit from the root node to a leaf node in the Octree. Therefore, We can use $\mathcal{F}(S)$ to estimate how much data locality a voxel sequence can utilize when inserting them into the Octree. A voxel sequence with a larger $\mathcal{F}(S)$ value always implies a slower Octree update speed and vice versa.

Lemma A.2. *For any 3 leaf nodes randomly picked, $\{a, b, c\}$, then $A(a, b), A(a, c), A(b, c)$ can be at most 2 different inner nodes. And there are at least 2 members among $A(a, b), A(a, c), A(b, c)$ equals $A(A(a, b), A(a, c), A(b, c))$*

Proof. Let's prove by contradiction: if $A(a, b), A(a, c), A(b, c)$ are three different inner tree nodes, then let's look at the 3 sub-trees whose root nodes are $A(a, b), A(a, c), A(b, c)$. Leaf node a which belongs to sub-tree under $A(a, b)$ and sub-tree under $A(a, c)$. Therefore, these 2 sub-trees must have overlapping nodes. Note that any 2 sub-trees with common leaf nodes and different root nodes should be contained by one another. Therefore, we can suppose the sub-tree under $A(a, c)$ contains the sub-tree under $A(a, b)$. Then we can have $A(a, b) = A(b, c)$, which is a contradiction. □

Lemma A.3. *For any three leaf nodes randomly picked, $\{a, b, c\}$, $D(a, b), D(a, c), D(b, c)$ can be at most two different numbers. Also, at least two values among $D(a, b), D(a, c), D(b, c)$ equal $max\{D(a, b), D(a, c), D(b, c)\}$*

Proof. Note that $A(a, b), A(a, c), A(b, c)$ only contains two different nodes. The distance between any inner tree node and its descendent leaf node is the same. Therefore, $D(a, b), D(a, c), D(b, c)$ can only be two values. □

Lemma A.4. *For any level k ($1 \le k \le l$) and any two different sets on that level $A_{k,i}$ and $A_{k,j}$, leaf nodes $x_1, x_2 \in A_{k,i}$ and $y_1, y_2 \in A_{k,j}$, we have $D(x_1, y_1) > D(x_1, x_2)$ and $D(x_1, y_1) = D(x_2, y_2)$.*

Lemma A.4 is easy to get based on the definition of $A_{i,j}$ ($1 \le i \le l, 1 \le j \le 8^{l-i}$).

Lemma A.5. *For any level k ($1 \le k \le l$) and any two different sets in that level $A_{k,i}$ and $A_{k,j}$ $1 \le i, j \le 8^{l-k}$, there are at most one pair of leaf nodes ($x_1 \in A_{k,i}, y_1 \in A_{k,j}$) satisfies that (x_1, y_1) is a neighboring pair in S^*.*

Proof. We use proof by contradiction. For any ordering S that we have at least two of such pairs, denoted as (x_1, y_1) and (x_2, y_2), where $x_1, x_2 \in A_{k,i}$ and $y_1, y_2 \in A_{k,j}$, we prove that we can change the leaf node order in S and construct S' so that $\mathcal{F}(S') < \mathcal{F}(S)$.

Condition 1: $S = ...x_1, y_1, [w_1, w_2, ..., w_m], x_2, y_2, ...,$ where $[w_1, w_2, ..., w_m]$ denotes the leaf nodes between y_1 and x_2 in S. We can construct $S' = ...x_1, x_2, [w_1, w_2, ..., w_m], y_1, y_2, ...$ We can easily prove that:

$$\mathcal{F}(S) - \mathcal{F}(S') = D(x_1, y_1) + D(x_2, y_2) - D(x_1, x_2) - D(y_1, y_2) > 0$$

Condition 2: $S = ..., p, x_1, y_1, [w_1, w_2, ..., w_m], y_2, x_2, q, ...$ Here p and q denote the leaf nodes former to x_1 and later to y_2. Without loss of generality, for any non-exist p (i.e., x_1 is already the first item in the sequence S), we give a complementary definition of $D(p, x_1) = 0$. The same complementary definition also fits q.

We construct $S' = ..., p, x_1, x_2, y_1, [w_1, w_2, ..., w_m], y_2, q, ...$ Based on Lemma A.3 and A.4 we can prove that:

$$\begin{aligned}\mathcal{F}(S) - \mathcal{F}(S') =& D(x_1, y_1) - D(x_2, y_1) - D(x_1, x_2) \\ &+ D(y2, x_2) + D(x_2, q) - D(y_2, q) > 0\end{aligned}$$

We find a contradiction. □

Lemma A.6. *In an optimal sequence S^*, the leaf nodes covered by the same set are always arranged together. In other words, for level k ($1 \le k \le l$), leaf nodes $x_1, x_2 \in A_{k,i}$, there is no leaf node $y \notin A_{k,j} (j \ne i)$ that lies between x_1 and x_2 in S^*.*

Proof. We still use proof by contradiction. For any sequence $S = ..., b, a_1, ..., a_2, c, ..., d, a_3, ..., a_4, e, ...,$ where $a_j \in A_{k,i}$ for $j = 1, 2, 3, 4$, and b, c, d, e don't. Our conclusion does not alter if there exists no b and e (i.e., , a_1 is the head of the sequences or a_4 is the tail of the sequence.). If S is an optimal sequence that minimizes $\mathcal{F}(S)$,

[10] All leaf nodes are on the same level in a perfect Octree.

[11] In a perfect Octree, the distance between node a and node $A(a, b)$ equals the distance between node b and node $A(a, b)$

[12] In the proof sketch, we refer to voxels as finest-level voxels.

we can know from Lemma A.5 that b, c, d, e belongs to different sets in level k. We switch the sub-sequence $c, ..., d$ and $a_3, ..., a_4$ and consider new sequence $S' = ..., b, a_1, ..., a_2, a_3, ..., a_4, c, ..., d, e, ...$

$$\begin{aligned}\mathcal{F}(S) - \mathcal{F}(S') =& D(a_2, c) + D(a_3, d) + D(a_4, e) \\ & - D(a_2, a_3) - D(a_4, c) - D(d, e)\end{aligned}$$

From lemma A.4 we know that $D(a_2, c) = D(a_4, c)$, we have:

$$\begin{aligned}\mathcal{F}(S) - \mathcal{F}(S') &= D(a_3, d) + D(a_4, e) - D(a_2, a_3) - D(d, e) \\ &= D(a_2, d) + D(a_2, e) - D(a_2, a_3) - D(d, e) > 0\end{aligned}$$

We find a contradiction. □

Leaf nodes ordering that meet Lemma A.6 requirements all have the same $\mathcal{F}$ value. This is because for each non-leaf node a in the octree, we have minimized $D(a_1, a_2) + D(a_2, a_3) + ... + D(a_{n-1}, a_n)$, where $a_1, ..., a_n$ are leaf nodes that belongs to the descendent set of a. Ordering voxels by their Morton Code is one of these orderings that gives the minimal $\mathcal{F}$ and thus is one optimal sequence.

For a non-uniform octree, the Morton Code of the leaf nodes are with different length because the leaf to tree distance is different. We specify the order of any 2 Morton Codes (of leaf nodes) by their dictionary order. For example, a tree node with Morton Code 11_2 should be smaller than a tree node with Morton Code 111_2 and 110_2. This allows us to order leave nodes for a non-uniform octree.

For any non-uniform octree, we first project all leaf nodes to their ancestors on the same level where the highest leaf node lies. Consider a uniform sub-octree with these ancestor nodes as leaf nodes. Based on theorem A.1 for a uniform octree, these ancestors should follow the Morton Code order S'.

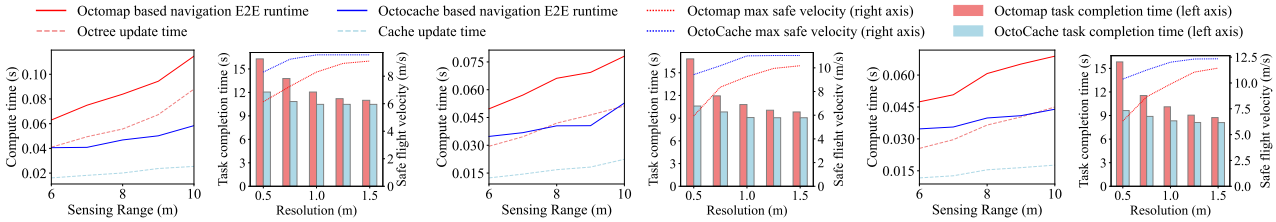
Any ancestor node should have its child nodes follow Morton Code order and arrange together in S. Suppose leaf nodes $w_1, w_2, \ldots, w_n$ are all children under one ancestor node w, and v is not a descendent node of w. Changing the order of $w_1, w_2, \ldots, w_n$ does not affect the distance $D(w_i, v)$ in S.

By "divide and conquer", we analyze the leaf nodes from bottom to top in the octree. Any leaf nodes with the same ancestor should follow Morton Code order and stay together in S, regardless of which level they are in the octree.

B More Evaluation Results

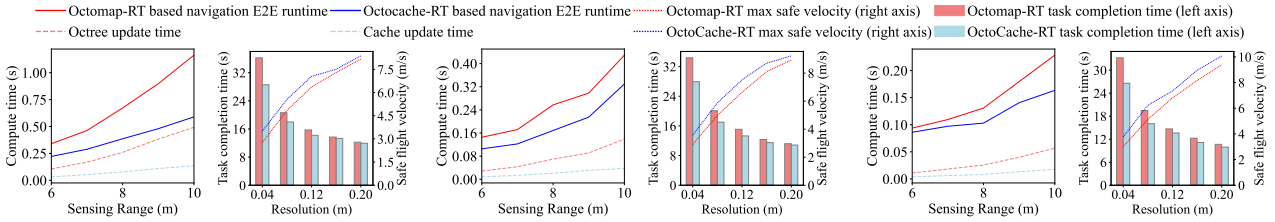
Here we show the thorough evaluation results covering four environments and various sensing range and mapping resolution choices. The max safe flight velocity and task completion time are calculated by using the AseTec UAV.

Received 24 June 2024; revised 12 November 2024; accepted 27 January 2025



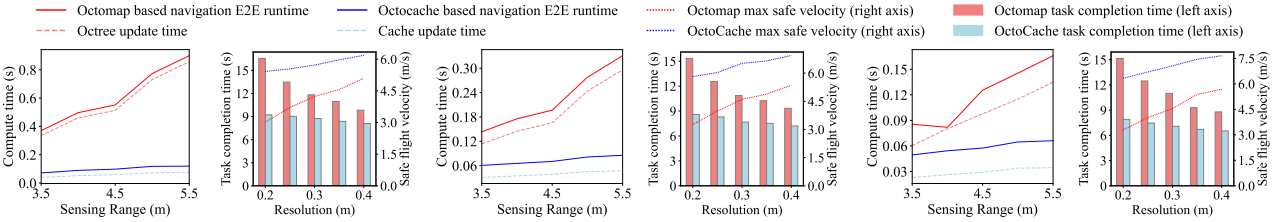
(a) Resolution 1.0 m (b) Sensing range 6.0 m (c) Resolution 1.25 m (d) Sensing range 8.0 m (e) Resolution 1.5 m (f) Sensing range 10.0 m

Figure 25: Runtime, flight speed, and task completion time comparison between Octomap-based navigation system vs OctoCache-based navigation system on openland scenario.



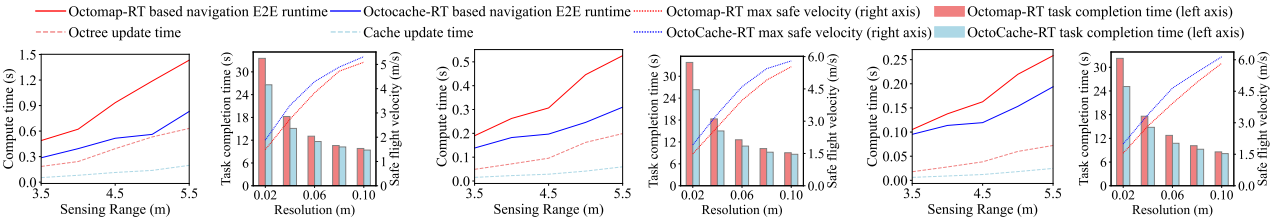
(a) Resolution 0.08 m (b) Sensing range 6.0 m (c) Resolution 0.12 m (d) Sensing range 8.0 m (e) Resolution 0.16 m (f) Sensing range 10.0 m

Figure 26: Runtime, flight speed, and task completion time comparison between Octomap-RT-based navigation system vs OctoCache-RT-based navigation system on openland scenario.



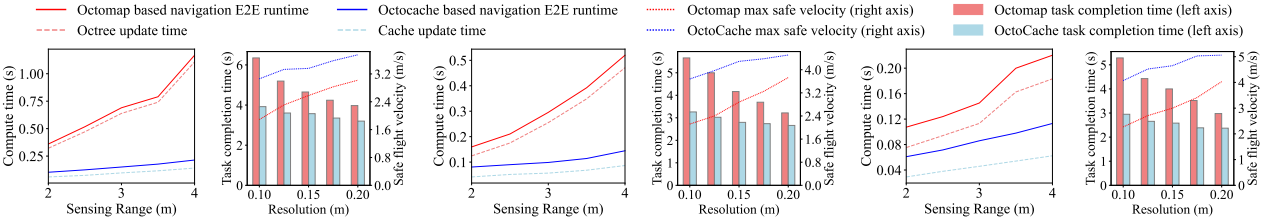
(a) Resolution 0.2 m (b) Sensing range 3.5 m (c) Resolution 0.3 m (d) Sensing range 4.5 m (e) Resolution 0.4 m (f) Sensing range 5.5 m

Figure 27: Runtime, flight speed, and task completion time comparison between Octomap-based navigation system vs OctoCache-based navigation system on farm scenario.



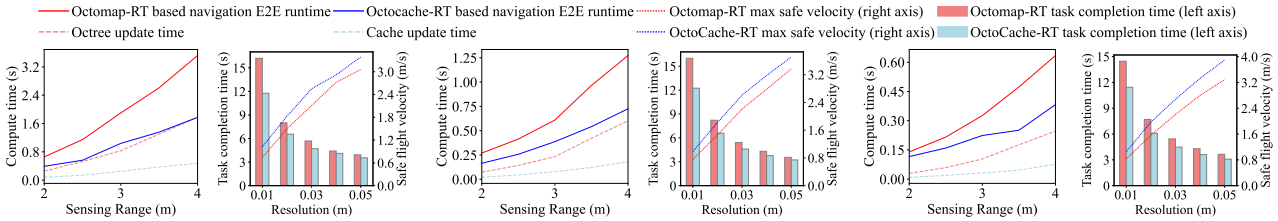
(a) Resolution 0.04 m (b) Sensing range 3.5 m (c) Resolution 0.06 m (d) Sensing range 4.5 m (e) Resolution 0.08 m (f) Sensing range 5.5 m

Figure 28: Runtime, flight speed, and task completion time comparison between Octomap-RT-based navigation system vs OctoCache-RT-based navigation system on farm scenario.



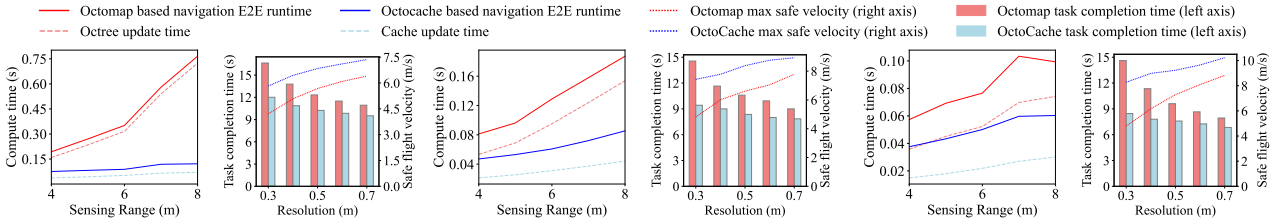
(a) Resolution 0.1 m (b) Sensing range 2.0 m (c) Resolution 0.15 m (d) Sensing range 3.0 m (e) Resolution 0.2 m (f) Sensing range 4.0 m

Figure 29: Runtime, flight speed, and task completion time comparison between Octomap-based navigation system vs OctoCache-based navigation system on room scenario.



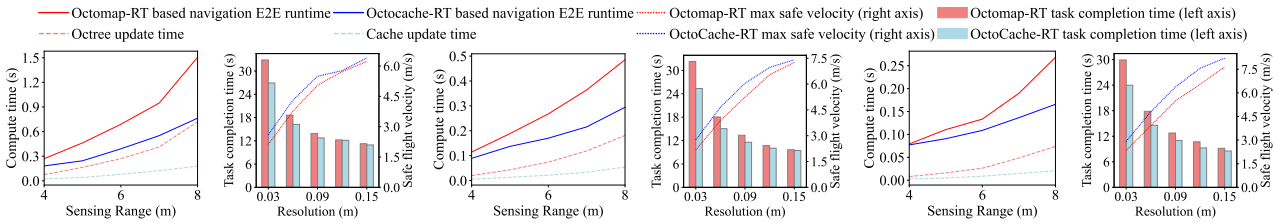
(a) Resolution 0.02 m (b) Sensing range 2.0 m (c) Resolution 0.03 m (d) Sensing range 3.0 m (e) Resolution 0.04 m (f) Sensing range 4.0 m

Figure 30: Runtime, flight speed, and task completion time comparison between Octomap-RT-based navigation system vs OctoCache-RT-based navigation system on room scenario.



(a) Resolution 0.3 m (b) Sensing range 4.0 m (c) Resolution 0.5 m (d) Sensing range 6.0 m (e) Resolution 0.7 m (f) Sensing range 8.0 m

Figure 31: Runtime, flight speed, and task completion time comparison between Octomap-based navigation system vs OctoCache-based navigation system on factory scenario.



(a) Resolution 0.06 m (b) Sensing range 4.0 m (c) Resolution 0.09 m (d) Sensing range 6.0 m (e) Resolution 0.12 m (f) Sensing range 8.0 m

Figure 32: Runtime, flight speed, and task completion time comparison between Octomap-RT-based navigation system vs OctoCache-RT-based navigation system on factory scenario.