Obstacle Clustering and Path Optimization for Drone Routing

Ang Li, Mark Hansen
Department of Civil and Environmental Engineering
University of California, Berkeley
Berkeley, CA, USA
angli@berkeley.edu, mhansen@ce.berkeley.edu

***Abstract*—To enable safe and efficient Unmanned Aircraft Systems (UAS) operations at low altitudes, it is necessary to conduct airspace management and operations for UAS traffic. This study focuses on deterministic clustering-based drone routing, with specific emphasis on the trade-off between horizontal and vertical travel costs. The routing problem is simplified to a 2D problem that we solve at several altitude candidates. Altitude candidates were generated based on clustered static obstacles in low urban airspace. Fast-Marching algorithm is performed to generate the shortest path at each altitude candidate. The optimal altitude is determined by weighing the vertical cost for ascent and descent over the horizontal cruising cost at certain altitude. Experiments are conducted to choose proper number of clusters and weight given to building height in the clustering procedure, and different shortest path algorithms are compared. Larger scale of Unmanned Aerial Vehicles (UAV) missions are simulated, based on which we analyze the relationship between optimal travel altitude and shortest cruise path, and estimate the UAV cost function.**

Keywords-UAV path planning; UAV cost function; Fast-Marching; A star (JPS);

I. INTRODUCTION

UAV have received increasing attention over the last decade, because of their immense potential to benefit commercial and industrial activities [1]. With large potential demand, it becomes necessary to manage UAV traffic in urban airspace. Organizations including National Aeronautics and Space Administration (NASA) [2] and Netherlands Aerospace Centre (NLR) [3], have undertaken the task of developing traffic management methods for UAV operations. There are also emerging UAS Traffic Management projects from Europe, Singapore and Korea.

The aim of UAV path planning is to identify efficient, safe flight trajectories in a timely manner, so that UAV can accomplish their missions and avoid threats. Performance of multiple UAV path planning algorithms were compared in various environments. These include Dijkstra's algorithm, Bellman Ford's algorithm, Floyd-Warshall's algorithm and the A Star algorithm in [4], and A star is found to perform better than others. A Star (JPS) is an improved path planning algorithm based on A Star [5,6,7]. Reference [8] finds that A Star (JPS) has better performance than Rectangular Symmetry Reduction (RSR). Fast-Marching methods have been found to yield consistent, accurate and highly efficient algorithms in optimal path planning around obstacles [9]. Reference [10] uses Fast-Marching algorithms to navigate a small quadrotor on an optimal collision avoidance path with a helicopter. Reference [11] presents both an offline optimal path planning algorithm based on A Star without considering the computation cost, and real-time suboptimal path planning algorithm based on genetic algorithm and potential fields technology. A spline-based path planning scheme that generates feasible flight routes for an UAV is developed in [12], which allows quick computation using a decomposition strategy.

An understanding of the path-related costs of UAV operation is needed for path planning. Horizontal and vertical cost ratio was used in [13] for drone routing. Horizontal path length cost and height cost were specified in [14] to perform 3D UAV trajectory planning. Reference [15] used required number of turns to measure cost function. Dublin path length was used in [16] as travel cost.

The primary purpose of this paper is to investigate the combined cost of horizontal and vertical distance when routing drones in low-altitude airspace in the presence of tall buildings. Using San Francisco as a case study, we investigate the trade-offs between routing a single drone at lower altitude with the resulting need to avoid many obstacles and using a higher altitude, which allows more direct horizontal paths but entails more vertical flight. The "sweet spot" in this trade-off depends on the relative cost of horizontal and vertical flight, and we study this relationship parametrically.

As part of this investigation, we consider different approaches to represent tall building obstacles for purposes of path planning, and also compare the performance of Fast-Marching and A Star routing algorithms. We assume that airspace structures will incorporate “no fly zones” that keep UAVs from tall buildings, but that to avoid undue complexity, these zones will be defined based on a relatively small number of building groups rather than a multiplicity of zones, each corresponding to an individual building. We therefore propose a methodology for identifying these building groups based on clustering, and consider how the number of clusters and the weight attached to building altitude in the clustering algorithm affects route efficiency and the computation time. The choice of routing algorithm is also critical in our analysis. In this regard, we show that Fast-Marching dominates A Star, even when a faster variant of the latter is used.

The contribution of this paper is that we apply obstacle clustering to efficiently reduce the obstacle complexity for routing. We applied Fast-Marching to horizontal drone routing and combined with obstacle clustering, which quickly returns better routes than many other shortest path algorithms. Finally, this paper proposes a UAV path cost function that predicts the cost of the least-cost path as a function of direct-line distance and the relative cost of horizontal and vertical travel.

II. DATA

This paper uses the financial district in San Francisco (SF) as study area. Only buildings are considered as static obstacles in urban airspace. San Francisco building footprints data from DataSF was used. The data contains San Francisco building footprint features, including roof boundary and building height. The research was performed using the projected coordinate system of EPSG 32610, WGS 84 / UTM zone 10N.

III. ASSUMPTIONS

The deterministic clustering-based single drone routing focuses on the trade-off between horizonal and vertical costs. For the purposes of our analysis make several simplifying assumptions. Since wind or any other features that cause uncertainties are not considered in this stage, cost is insensitive to where along its path the drone ascends and descends. We assume the drone flies at a single altitude, after vertically ascending at the origin and prior to descending at the destination, and correspondingly that the cost of the route depends on the vertical distance and the horizontal distance of the route. Cost of turning is not considered in this research. Though we didn’t explicitly consider random deviations between the actual path of the drone and its nominal path, in order to ensure safety, a keep-out geofence, the safety distance that drones are required to keep away from buildings, is considered when we generate aggregated obstacles by clustering. We simply add the keep-out geofence distance by expanding the actual building boundary outward a certain distance. All the following research is using 10-meter keep-out geofence distance. In addition, we don’t consider any geographical ground level in the current stage. Above Ground Level (AGL) or Median Sea Level (MSL) can be added by performing this research in corresponding projected coordinate system.

IV. METHODOLOGY

The routing approach can be simplified from 3D path planning to 2D by routing at several attitudes with assumptions above. The optimal travel altitude will be determined by weighing the horizontal travel cost over vertical cost for ascent and descent. A set of altitude candidates is needed to compare the vertical and horizontal cost since exhaustive search over all altitudes is computational expensive. Obstacle complexity will influence the computation time of shortest path algorithm. In order to generate the most appropriate altitude candidates and reduce complexity of the obstacle field, employ a clustering approach to summarize the height and location of the numerous static obstacles. Based on the generated altitude candidates set, horizontal shortest paths that avoid obstacles will then be generated for each altitude candidate. We compare the vertical and horizontal costs to decide the optimal travel altitude and 2D cruise path at optimal altitude.

A. Static Obstacles Clustering for Altitude Candidates

A set of altitude candidates was generated by clustering obstacles. The K-means clustering algorithm is applied to perform clustering over all buildings in SF financial district. We first generate the minimum bounding rectangle containing the footprint of each building, since the K-means algorithm requires the same feature dimensions for every building observation. Each building is summarized by nine features: X and Y coordinates of four minimum bounding rectangle building vertices plus building height. While all variables are in units of meters, height is unique because it varies far less than the X and Y coordinates. For this reason, building heights are rescaled by different factors. We presented the results of 20, 30, and 40 clusters, with scale of 10, 20, 30, 40, and 50 times for building height in this paper.

After clustering, the convex hull of all minimum bounding rectangle vertices of buildings in the same cluster forms an aggregated obstacle whose altitude is the maximum height of buildings in the cluster. The path planning method in later section is performed with the aggregated obstacles. Two trade-offs of obstacle clustering, number of clusters and building height rescaling factor, are shown in Fig. 1. On the one hand, when the number of clusters is small, more airspace will be

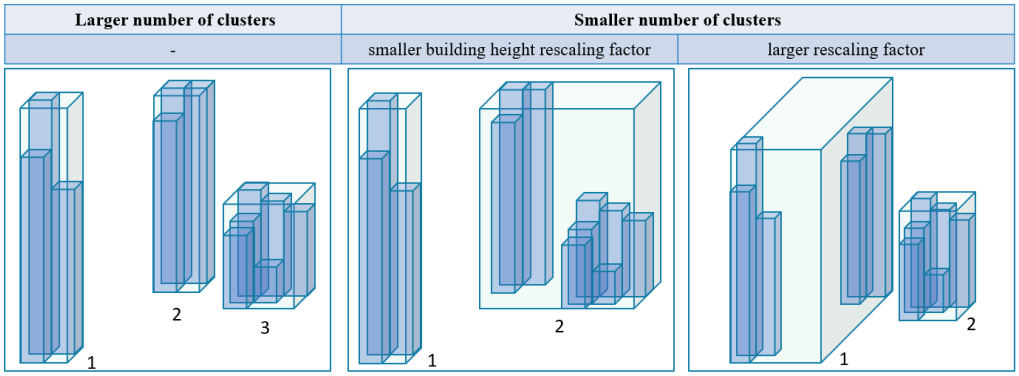

Figure 1. Trade-offs of different number of clusters and building height rescaling factors

made unavailable as a result of being included in the polygons of aggregated obstacles. On the other hand, when the number of clusters increases the complexity of the obstacle field increases and it will be more computationally expensive to generate shortest paths. With larger rescaling factor, buildings with similar height will be more likely to be clustered in the same aggregated obstacle, instead of buildings that are located closer. Aggregated obstacles are most dispersed but of more uniform height in the case with more scaling. We perform sensitivity analysis for these two factors in Section V.

Fig. 2 shows an example of aggregated obstacles at altitude 146.07m in 20-cluster case. Red dots are the boundary points of aggregated obstacles after clustering, and the aggregated obstacles area is filled with green. Dots with the same color within the boundary, as well as the red boundary points, are the minimum bounding rectangles vertices of buildings belong to the same cluster. Grey points represent all the other buildings lower than the current altitude in SF financial district.

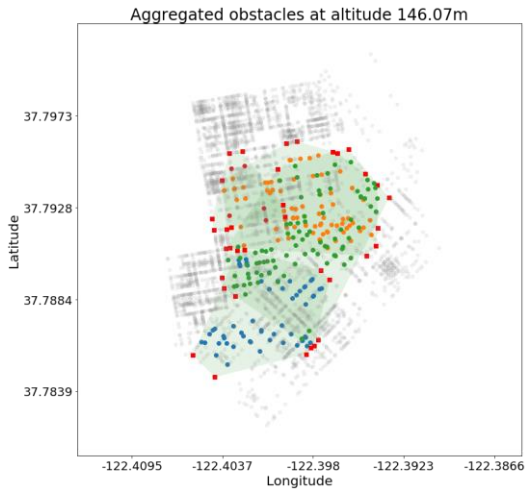

Figure 2. Sky view of aggregated obstacles in SF financial district (20 clusters)

B. *Optimal Horizontal Travel Route*

The height of each aggregated obstacle, which is the maximum actual building height within each cluster, forms the altitude candidates set. The optimal 2D cruise path is generated at each altitude candidate in this subsection.

The Fast-Marching (FM) algorithm is used to generate the shortest cruise path. Compared to the traditional Dijkstra algorithm or A star algorithm, FM replaces the graph update by a local resolution of gradient descent, instead of only considering standard 8 directions of neighbors, which significantly reduces the grid bias. The computation complexity of FM is $O(NlogN)$, where N is the total number of grid points, which is the number of visited points during the computation in practice. FM method has less computation complexity compared to A star whose complexity depends on heuristic, and FM yields a better approximation of the true shortest path.

In our research, the grid size is set to be 1m and the step size is set to be 5m. Fig. 3 shows an example of shortest path results for the case with 20 clusters and 50x building height rescaling. The dark blue areas in the plots represent all aggregated obstacles at given altitude. The red and green dots represent the origin and destination. As the altitude increases, some obstacles disappear, leaving more available airspace for drone to travel. The shortest cruise path decreases accordingly.

C. *Determine Optimal Travel Altitude*

Shortest travel paths at different candidate altitudes are generated as described above. To decide the optimal travel altitude, we are interested in how the length of the shortest path changes at different altitudes. Fig. 4 plots the pattern of shortest path length at different altitude candidates of the same OD as in Fig. 3 with 20 clusters and 50x rescaling of building height. The red dots represent the data at each altitude candidate, and

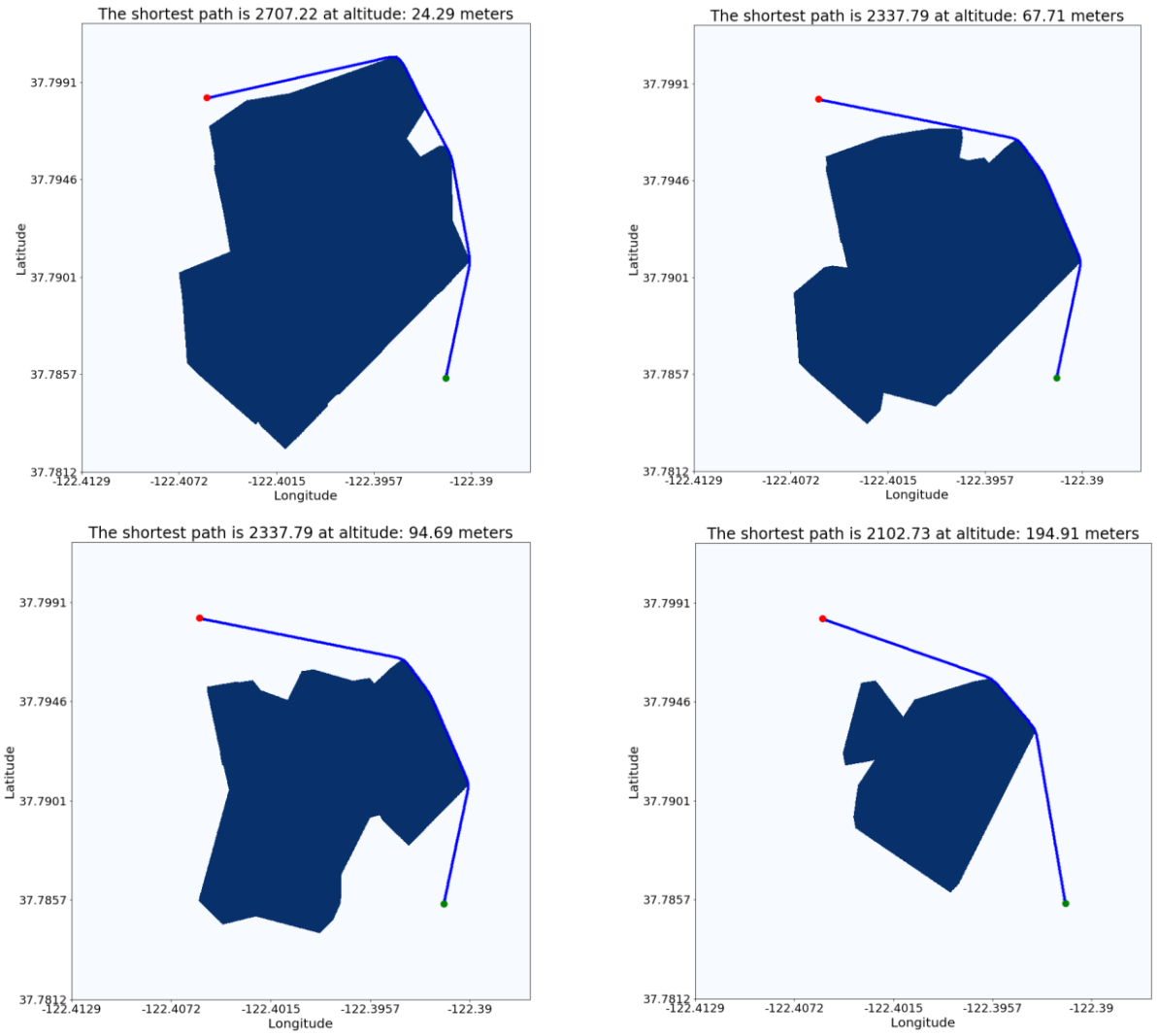

Figure 3. An example of path planning at different altitudes for the same OD

we superimpose step lines. Based on the shortest path length profile in this case, we can determine the optimal route altitude for a given ratio of vertical cost to horizontal cost. (In this paper we represent vertical unit cost as the average of climbing and descending unit cost.) This ratio determines the slope of the black iso-cost line in Fig. 4. The point where the lowest iso-cost line touches the red plot will give the optimal travel altitude. A given altitude will be optimal for a range of cost ratios.

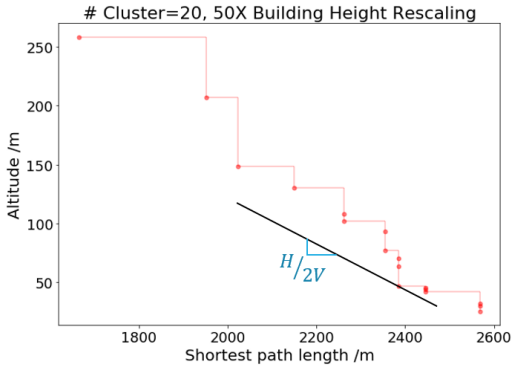

Figure 4. Shortest horizontal path lengths at different altitudes

V. EXPERIMENTS

This section performs sensitivity analysis for the number of clusters and the building height rescaling factor, and compares different shortest path algorithms. 10 random OD pairs were generated for analysis in this section.

A. *OD Sampling*

A random sample of OD's is generated within the red circle containing all obstacles in San Francisco area shown in Fig. 5. The center of the red circle is the middle point of maximum and minimum X and Y coordinates of all obstacles. The radius is 1100 meters, which just contains all obstacles in study area. Points are randomly sampled within the red circle. We assume that only the points not located in the minimum bounding rectangle of buildings can be used as O's and D's. Red and green dots are the sampled origins and destinations. The obstacles area (filled with blue) here uses the minimum bounding rectangle of buildings without clustering. As mentioned in the last section, there exists O's or D's located in the clustered aggregated obstacle area but not in the actual obstacles, because of wasted airspace by clustering. These points located in the wasted airspace can be used as O's and D's, but the shortest path is only feasible at altitude higher than both altitudes of aggregated obstacles at origin and destination. We only consider OD's with Euclidean Distance longer than 1000 meters to reduce the possibility that the travel paths of OD samples are obstacle free.

B. *Sensitivity Analysis of the Number of Clusters*

Sensitivity analysis of 20, 30 and 40 clusters, assuming 30x height rescaling, for 10 OD's is performed in this subsection in order to determine the proper number of clusters. An example of altitude vs. horizontal path length plot is shown in Fig. 6. The shortest cruise path length is always shorter with more clusters, since less available airspace is wasted. More clusters require more computation time (see Table I). The largest distance gap (refer to Fig. 6) between 20 and 40 clusters cases is calculated for all 10 OD samples. The maximum percentage savings of

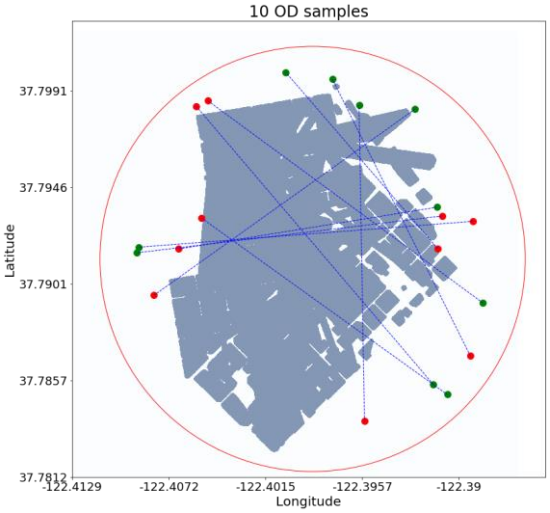

Figure 5. OD Sampling

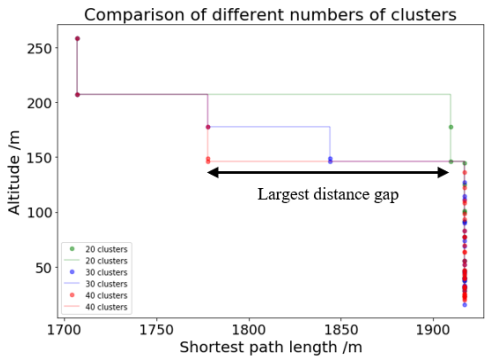

Figure 6. Horizontal shortest path lengths at different altitudes with 30X scaling of building height

shortest cruise path length are calculated as the largest distance gap divided by corresponding 20-cluster shortest path length. The average maximum path length savings is only about 10% using 40 clusters compared to 20 clusters. Therefore, we pick 20 clusters for later study considering both path length savings and computational convenience.

TABLE I. COMPUTATION TIME WITH DIFFERENT CLUSTER NUMBERS

# clusters	***20***	***30***	***40***
Computation time for 10 OD's /s	386.2	549.7	950.2

The influence of different numbers of clusters on path cost is also analyzed. We assume constant unit costs for vertical and horizontal travel. Given vertical and horizontal cost ratio (V/H), the adjusted cost, in horizontal distance units can be calculated by the following equations:

$$C = l(h) + {}^{2V}/_{H} \cdot h \tag{1}$$

$$C^*({}^{V}/_{H}) = l(h^*({}^{V}/_{H})) + {}^{2V}/_{H} \cdot h^*({}^{V}/_{H}) \tag{2}$$

Where $l(h)$ is the shortest horizontal path length at altitude h, and $h^*(\cdot)$ is the optimal altitude that minimizes total cost. We use simple enumeration among all the altitude candidates to find $h^*(\cdot)$. Cost ratios of 1, 2, 5, and 10 are used to compare cost results since vertical cost is higher than horizontal travel cost in most cases [17]. As shown in Fig. 7, total travel cost has lower mean with more clusters for all four cost ratios. The optimal travel altitude is lower with more clusters, since less airspace is wasted and horizontal path is shorter. The cost difference with different numbers of clusters increases as the V/H cost ratio increases. When V/H cost ratio is small (e.g. V/H=1 in Fig. 7), cost is quite insensitive to the number of clusters, since the optimal paths are higher and therefore avoid most obstacles. The sensitivity to the number of clusters becomes greater when high

costs of vertical movement push optimal paths toward lower altitudes, where obstacles matter more.

C. Sensitivity Analysis of Building Height Rescaling Factor

While all features in clustering are in units of meters, height is unique because it varies far less than the X and Y coordinates. The height of a cluster is the height of the tallest building in the cluster. For this reason, building heights are rescaled by different factors. The impact of changing the building height rescaling factor from 10 to 50 times is analyzed in this subsection. In Fig. 8, the total travel cost is larger with higher V/H cost ratio. The cost is not monotonically increasing or decreasing as building height rescaling factor changes when V/H cost ratio is small (V/H=1 or 2). This can be explained by the trade-off of rescaling factor described in Fig. 1. The more we scale, the more likely available airspace between buildings with similar height is regarded as obstacles. However, if we don't scale enough, more airspace is wasted because of building height difference. Drones will be able to travel at lower altitude with higher rescaling factor, since a higher rescaling factor saves more airspace associated with the height difference of buildings, which results in lower cost with higher rescaling factor if the cost ratio is large (V/H=5 or 10).

In order to determine the proper rescaling factor, we plot the total volume of obstacles in study airspace under different cases in Fig. 9. The total volume of obstacles decreases with more clusters, since less available airspace will be counted as obstacles. The total volume of obstacles in the airspace decreases at first as rescaling factor increases, then stays almost stable, after the 30x rescaling factor. Fig. 8 suggests that a higher rescaling factor (e.g. 50) yields a lower cost when V/H is high, without any significant cost penalty when V/H is small. The computation time does not change significantly with different building height rescaling factors.

D. Comparison with A star (JPS)

The A Star Jump Point Search (JPS) algorithm makes pathfinding on a rectangular grid more efficient, especially in open spaces. It performs very well on quickly generating a path. This algorithm is compared with Fast-Marching method.

The cumulative frequency diagram of cost difference between A star (JPS) and Fast-Marching algorithms with different cost ratios is presented in Fig. 10. Cost difference of two algorithms is distributed in a larger range with higher cost ratio. Given different vertical and horizontal cost ratios, cost using A star (JPS) is always larger since Fast-Marching gives a near-optimal shortest path and A star (JPS) does not necessarily do so. The travel cost difference between two algorithms amplifies with larger vertical and horizontal cost ratio. In addition, by comparing computation time in Table III, we see Fast-Marching has better performance. For this reason, we subsequently used Fast Marching.

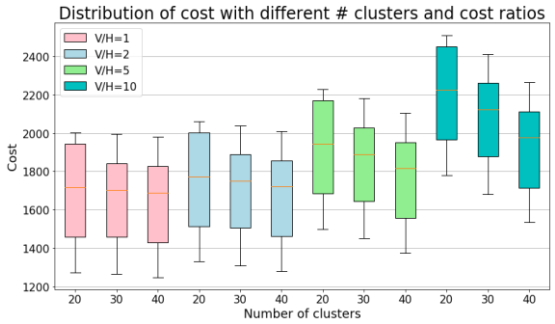

Figure 7. Sensitivity analysis of different numbers of clusters

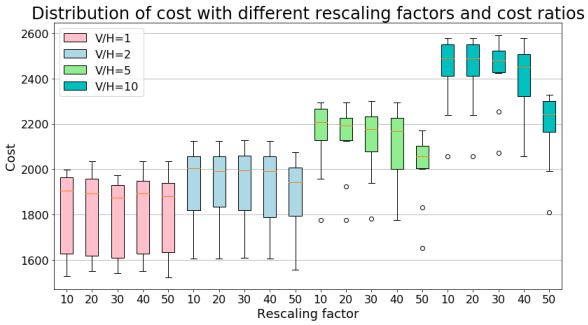

Figure 8. Building height rescaling factors sensitivity analysis

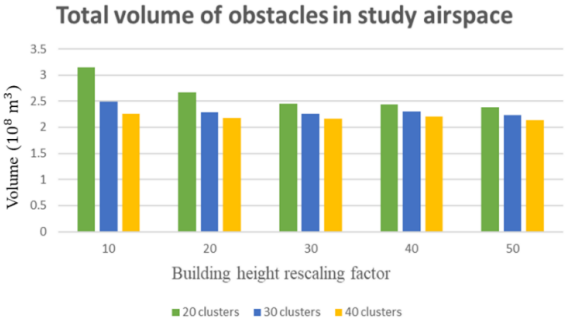

Figure 9. Total volume of obstacles in different scenarios

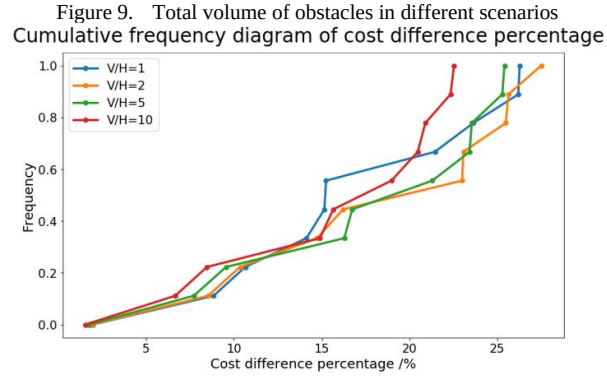

Figure 10. Cumulative frequency diagram of cost difference percentage

TABLE II. COMPUTATION TIME OF DIFFERENT ALGORITHMS

Shortest path algorithm	***A star (JPS)***	***Fast-Marching***
Computation time for 10 OD's /s	585.07	386.2

VI. LARGER SAMPLE ANALYSIS

More OD pairs are simulated in this section to analyze the relationship between shortest cruise path length and travel altitude, and trade-off between horizontal and vertical cost. 200 OD pairs are generated using the same sampling strategy in Section V. Results in this section are based on 20 clusters and 30x rescaling factor case.

A. Analysis of Path Length and Altitude

The ratio of shortest cruise path length and horizontal Euclidean Distance at different altitudes is presented in Fig. 11. The median of shortest path length and Euclidean Distance ratio decreases with altitude, since less obstacles must be avoided. From altitude 258.49m to 23.26m, the median distance ratio increases from 1 to around 1.25, and the 75th percentile increases from 1.05 to around 1.35. At the highest altitude of 20 clusters, 258.49m, more than 75% of the paths are of Euclidean Distance, while for the balance the path must be adjusted to avoid the single obstacle cluster that has this maximum altitude.

The relationship between additional shortest cruise path length compared to Euclidean Distance and Euclidean Distance, at altitudes 258.49m, 109.71m, and 23.26m are presented in Fig. 12. At very high altitude, 258.49m, most of shortest cruise paths equal to Euclidean Distance, as drones fly direct Euclidean Distance for most OD's. The shortest cruise path length varies much more at the lower bound altitude 23.26m. At the intermediate altitude of 109.71m, the shortest cruise path length varies more with longer distance between OD, because at shorter distances it is more likely that the shortest cruise path is obstacle free.

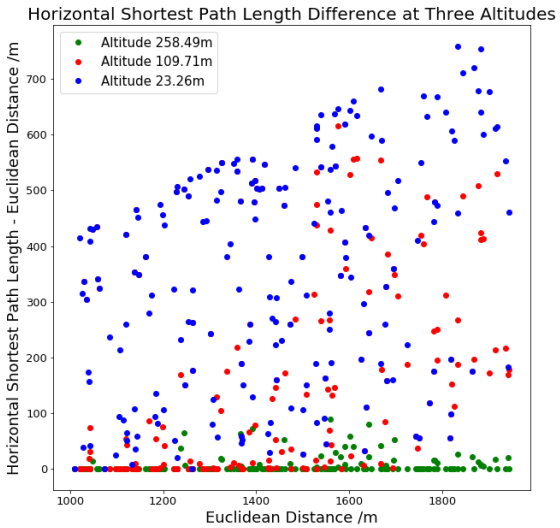

Figure 12. Horizontal shortest path length difference at three altitudes

B. Cost Function Estimation

Based on the analysis of the relationship between cost and other features, we generate the UAV path cost function. This function predicts the cost of the least cost path in horizontal distance units, taking into account both the vertical and horizontal cost. We propose the following cost function specification:

$$C^*\left(V/H\right) = ED \cdot \left(\alpha\left(V/H\right)^{\beta} + 1\right) \quad (3)$$

where ED is the Euclidean Distance between OD, α and β are the coefficients to be estimated. The intuition for this functional form is that if V/H=0, the optimal solution is to climb to an obstacle-free altitude and fly the Euclidean Distance. However, as this ratio increases, the optimal altitude will decrease, resulting in more circuitous paths as well as a larger vertical cost component.

Assigning cost ratios from 1 to 20 with increment 0.5, cost function is estimated based on 200 OD samples using linear regression. As before the minimum cost for a given OD is found by simple enumeration of all the altitude candidates. The estimated result is:

$$C^*\left(V/H\right) = ED \cdot \left(0.154 \cdot \left(V/H\right)^{0.617} + 1\right) \quad (4)$$

The R-square is 0.7, indicating the cost function is a good fit. The beta coefficient in the cost function is 0.617 from estimation. It is intuitive that this coefficient should be less than 1, since a higher cost ratio reduces the optimal altitude.

Predicted cost and actual cost are compared in Fig. 13. The blue scatter points show obvious quasi-linear patterns, which correspond to the results for different OD pairs, and the cost function captures the overall linear trend very well. Systematic differences between the OD pairs are also evident. The different curvatures of the OD-specific quasi-linear patterns show that least-cost paths for different OD's have different sensitivities to the V/H value; the results in equation (4) thus reflect the average of this sensitivity across the 200 OD's. Further analysis is expected to yield a cost model that is more sensitive to differences between OD pairs.

VII. CONCLUSIONS

This paper finds that a clustering-based method can efficiently summarize the trade-off between low altitude routes that must avoid many buildings and high-altitude routes that involve larger vertical cost. In the case of San Francisco, we

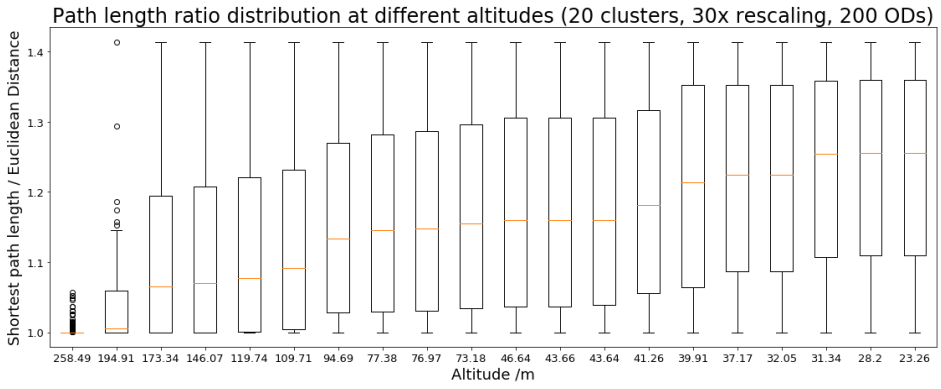

Figure 11. Horizontal path length ratio distribution at different altitudes of 200 samples

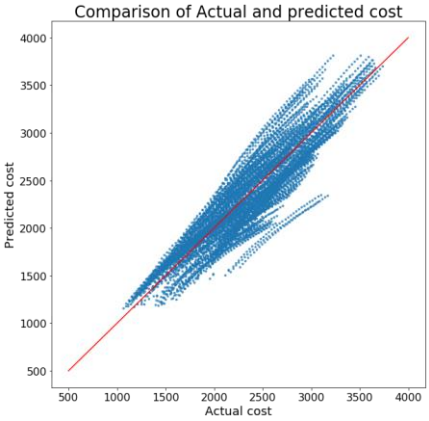

Figure 13. Comparison between actual and predicted cost

represented 931 buildings with 20-40 clusters. Each of these clusters has an altitude defined by the tallest building it contains and thus defines a candidate altitude for drone routing. Thus we can capture the essential trade-off with a small number of altitude candidates. For example, we find that the median ratio of horizontal path length to Euclidean Distance decreases from 1.25 to 1.0 if the drone climbs from about 30 meters to 250 meters. For a given ratio of vertical to horizontal cost, one of these candidates yields the lowest total path cost. The trade-off can be succinctly summarized with a cost function that gives the lowest total cost (vertical plus horizontal) for a route as a function of the Euclidean Distance between the origin and destination and the value V/H, which, despite having a very simple form, has very good predictive performance.

Future work should move along several lines. First, the cost function should be improved by considering other features of the OD pair aside from Euclidean Distance. Second, the analysis should be extended to other cities. Third, topography should be taken into account by performing the routing in a projected AGL coordinate system. Third, path costs should capture additional factors as such turning, operator-drone connectivity, population density, drone type, payload, and stochastic factors such as wind. Finally, once the single-drone problem have been satisfactorily solved, we must move on to the multiple drone routing problem, which requires de-conflicting drone paths in space and time.

REFERENCES

[1] Mordor Intelligence Industry Reports, "Global UAVs Market - Growth, Trends and Forecasts (2016 - 2021)," Mordor Intelligence, September 2016.

[2] P. Kopardekar, J. Rios, T. Prevot, M. Johnson, J. Jung and J. E. Robinson III, "Unmanned Aircraft System Traffic Management (UTM) Concept of Operations," in 16th AIAA Aviation Technology, Integration, and Operations Conference, AIAA Aviation, Washington, D.C., 2016.

[3] Netherlands Aerospace Centre (NLR), "Annual Report 2011," Netherlands Aerospace Centre (NLR), Netherlands, 2011.

[4] B. Moses Sathyaraj, L. C. Jain, A. Finn, S. Drake, "Multiple UAVs path planning algorithms: a comparative study, " Fuzzy Optimization and Decision Making 7, 257–267, 2008.

[5] A. Botea, M. Müller, and J. Schaeffer, "Near Optimal Hierarchical Path-finding, " in Journal of Game Development (Issue 1, Volume 1), 2004.

[6] D. Harabor, A. Botea, and P. Kilby, "Path Symmetries in Uniform-cost Grid Maps, " in Symposium on Abstraction Reformulation and Approximation (SARA), 2011.

[7] D. Harabor and A. Grastien, "Online Graph Pruning for Pathfinding on Grid Maps, " in National Conference on Artificial Intelligence (AAAI), 2011.

[8] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, and L. Jurišica, “Path Planning with Modified a Star Algorithm for a Mobile Robot”, Procedia Eng., vol. 96, pp. 59–69, 2014.

[9] J.A. Sethian, "Fast-Marching methods, " SIAM Rev., 41, p. 199, 1999.

[10] Z. Liu, and A. G. Foina, "An Autonomous Quadrotor Avoiding a Helicopter in Low-Altitude Flights, " IEEE Aerospace and Electronic Systems Magazine, Vol. 31, No. 9, pp. 30–39. doi:10.1109/MAES. 2016. 150131, 2016.

[11] Y. Qu, Q. Pan, J. Yan, "Flight path planning of UAV based on heuristically search and genetic algorithms", Proc. 31st Annu. Conf. IEEE Ind. Electron. Soc., Nov. 2005.

[12] K.B. Judd, T.W. McLain, "Spline based path planning for unmanned air vehicles, " in: AIAA Guidance, Navigation, and Control Conference and Exhibit, Montreal, Canada, 2000.

[13] S. Scherer, D. Ferguson, S. Singh, "Efficient C-space and cost function updates in 3D for unmanned aerial vehicles, " in IEEE international conference on robotics and automation, ICRA’09 (pp. 2049–2054). New York: IEEE Press, 2009.

[14] M. Bagherian, and A. Alos, "3D UAV trajectory planning using evolutionary algorithms: A comparison study, " Aeronautical Journal, 119, (1220), pp 1271-1285, October 2015.

[15] A. Otto, N. Agatz, J. Campbell, B. Golden and E. Pesch, Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey, Networks 72, 411–458, 2018.

[16] H. Oh, S. Kim, A. Tsourdos, B.A. White, "Coordinated Road-Network Search Route Planning by a Team of UAVs, " International Journal of Systems Science, doi: 10.1080/00207721. 2012. 737116, 2012.

[17] Z. Liu, R. Sengupta and A. Kurzhanskiy, "A power consumption model for multi-rotor small unmanned aircraft systems," 2017 International Conference on Unmanned Aircraft Systems (ICUAS), Miami, FL, USA, 2017, pp. 310-315, doi: 10.1109/ICUAS.2017.7991310.