Risk-Aware Planning and Assignment for Ground Vehicles using Uncertain Perception from Aerial Vehicles

Vishnu D. Sharma[†1], Maymoonah Toubeh[†1], Lifeng Zhou[†1], Pratap Tokekar[2]

***Abstract*— We propose a risk-aware framework for multi-robot, multi-demand assignment and planning in unknown environments. Our motivation is disaster response and search-and-rescue scenarios where ground vehicles must reach demand locations as soon as possible. We consider a setting where the terrain information is available only in the form of an aerial, georeferenced image. Deep learning techniques can be used for semantic segmentation of the aerial image to create a cost map for safe ground robot navigation. Such segmentation may still be noisy. Hence, we present a joint planning and perception framework that accounts for the risk introduced due to noisy perception. Our contributions are two-fold: (i) we show how to use Bayesian deep learning techniques to extract risk at the perception level; and (ii) use a risk-theoretical measure, CVaR, for risk-aware planning and assignment. The pipeline is theoretically established, then empirically analyzed through two datasets. We find that accounting for risk at both levels produces quantifiably safer paths and assignments.**

I. Introduction

Consider scenarios where an autonomous ground vehicle must navigate in an unknown environment. Examples include search and rescue, space exploration, and disaster response. For instance, consider a disaster response scenario where ground vehicles must supply resources to specific demand locations as soon as possible. In such settings, prior GPS or satellite maps of the environment may no longer be valid. Instead, an aerial robot may be employed to take live aerial images which can then be used to plan the paths of the ground vehicles towards the demand locations. However, due to the inherent uncertainty of aerial images, the paths that are found may not actually represent the situation on the ground. Therefore, there is a risk of the vehicles taking longer to reach the demand positions than planned. In safety-critical situations, one way to mitigate the risk is to assign multiple vehicles to the same demand, with the earliest arriving one actually responding to the demand.

Motivated by this scenario, we study the problem of how to find risk-aware paths for multiple vehicles to serve multiple demand locations. There are two problems to be solved — assigning the vehicles to the demand locations and finding risk-aware paths from start to demand locations. We present a risk-aware framework to solve both problems simultaneously.

The environment where the vehicles navigate is captured by an overhead image. We implement a deep learning technique for semantic segmentation of the overhead image. Due to the uncertainty from segmentation, the travel cost of the vehicle turns out to be a random variable. Build on our previous work [1], our first contribution is to show how to utilize Bayesian deep learning techniques to handle the risk from the planning and perception level. After risk-aware planning and perception, we generate a set of candidate paths corresponding to different risk-levels from each vehicle's start position to each demand location. Our second contribution is to assign each vehicle to a risk-aware path from its candidate path set to a demand. We utilize a risk measure, Conditional-Value-at-Risk (CVaR), to manage the risk from the uncertainty. Our assignment framework provides the flexibility to trade off between risk and reward, which builds on our previous work [2], with risk here being assessed at multiple levels of the algorithm. Our empirical results show that this risk-aware framework results in safer path planning and assignment.

Related Work. Deep learning has shown significant improvements in perception capabilities for many robotics applications. However, the potential of the positive impact deep learning may have on real-world scenarios is inevitably proportionate to their interpretability and applicability to imperfect environments. In these cases, deep neural networks can even misrepresent data outside the training distribution, giving predictions that are incorrect without providing a clear measure of certainty associated with the result [3]. The extraction of uncertainty information, as opposed to the reliance on point estimates, is crucial in safety-critical applications, such as autonomous navigation in an urban, unstructured setting. The methods like Natural-Parameter Networks [4] propose modeling the network parameters and inputs as Gaussian distributions. However, these modifications impose huge computation cost on the model due to the increment in the number of trainable parameters. Lightweight Probabilistic Deep Networks [5] alleviates this concern to some extent by making the weights deterministic. Large size networks are also unsuitable for real-time robotics applications which may have constraints on inference time and memory. [6] and [7] propose methods that allow uncertainty extraction from deep learning models, specifically those that do not interfere with the overall structure or training process. In this work, we leverage [6], which shows that dropouts, which are often used as a regularization enhancement in neural networks,

[1]The authors are with the Department of Electrical & Computer Engineering, Virginia Tech, Blacksburg, VA 24061, USA. {vishnusharma, may93, lfzhou}@vt.edu.

[2]The author is with the Department of Computer Science, University of Maryland, College Park, MD 20742, USA (email: tokekar@umd.edu).

†These authors contributed equally.

This work is supported by the Office of Naval Research under Grant No. N000141812829.

can provide approximate Bayesian inference and thus help in uncertainty estimation for the deep learning models.

In addition to considering uncertainty in the perception level, we also utilize some popular risk measures to handle uncertainty in the planning and assignment levels. A typical measure for optimizing under uncertainty is the expectation of the stochastic function. However, the expectation is a risk-neutral measure and may not be desirable, especially in critical tasks, like search-and-rescue operations [8]. For example, we may find a path with a lower expected cost but with high variance. It is quite likely that the vehicle may encounter a much larger cost (as compared to the expected one) when traveling on this path. Thus, instead of using expected cost, we utilize some other risk-aware measures, such as mean-variance [9], [10] and conditional-value-at-risk (CVaR) [8], [11].

In particular, we use mean-variance as risk/cost measure in the A* planner [12] for planning paths for the vehicles. The mean-variance measure allows us to balance the mean cost and uncertainty (variance) when planning paths. Also, for the path assignment, we use CVaR to deal with the uncertainty on the path level. CVaR_α explicitly takes into account the risk associated with bad scenarios [8], [11]. Specifically, CVaR_α measures the expectation of a random variable in the 100α–percentile worst scenarios. Here, the user-defined risk-level, $0 < \alpha \leq 1$, provides a user with the flexibility to choose a risk that they would like to take. Setting $\alpha = 1$ makes CVaR_α equal to the expectation whereas $\text{CVaR}_\alpha \approx 0$ is akin to worst-case optimization.

Contributions. In this paper, we have three main contributions.

- We present a framework that plans and assigns risk-aware paths for the robots when they navigate in unknown environments.
- We utilize the Bayesian deep learning technique to learn an unknown environment whose information is only available by an overhead, georeferenced image.
- We deal with the uncertainty in both path planning and assignment levels by optimizing the corresponding risk measures. In the end, we assign each vehicle a risk-aware path to a demand location and the path assignment is guaranteed to have a bounded approximation performance of the optimal.

This work builds on our prior work where we studied these two problems (uncertainty extraction from deep learning and CVaR optimization) individually. Here, we investigate the joint problem. We find that utilizing the uncertainty extraction from deep learning and managing the risk from uncertainty by CVaR optimization provide the vehicles with safer and risk-aware paths in unknown environments.

II. Preliminaries

We start by defining the notations used in the paper. We then give background on a risk measure, conditional-value-at-risk (CVaR).

A. Conditional Value at Risk

Let X be a random variable. $\text{CVaR}_\alpha(X)$ denotes the expectation on the α-worst scenarios of f with $\alpha \in (0, 1]$. More specifically, if X indicates reward or benefit, $\text{CVaR}_\alpha(X)$ denotes the expectation on the left α-tailed scenarios. While, if X represents loss or penalty, $\text{CVaR}_\alpha(X)$ is the expectation on the right α-tailed cases. Here, α is the confidence level or the risk level. If α is close to 0, CVaR_α is close to the worst-case. If α is equal to 1, CVaR_α is same as the expectation.

In this paper, the utility function $f(\mathcal{S}, y)$ defined on set $\mathcal{S}$ is a random variable with randomness induced by parameter y. Since $f(\mathcal{S}, y)$ is utility that indicates benefit, $\text{CVaR}_\alpha[f(\mathcal{S}, y)]$ denotes the expectation on the left α-tailed cases, as shown in Figure 1.

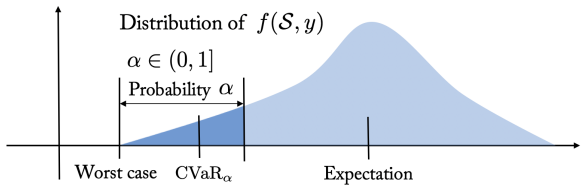


Fig. 1. CVaR_α of function $f(\mathcal{S}, y)$.

We generally maximize $\text{CVaR}_\alpha[f(\mathcal{S}, y)]$ by solving

$$\max_{\mathcal{S},\tau} \ \tau - \frac{1}{\alpha}\mathbb{E}[(\tau - f(\mathcal{S}, y))_+], \tag{1}$$

where $\mathcal{S}$ is a decision set (or solution set) and $\tau \in \mathbb{R}_+$ is an auxiliary parameter. For the ease of expression, we define

$$H(\mathcal{S}, \tau) = \tau - \frac{1}{\alpha}\mathbb{E}[(\tau - f(\mathcal{S}, y))_+] \tag{2}$$

B. Problem Formulation

We consider the problem of finding paths for multiple vehicles to serve multiple demand locations. In particular, we are given N vehicles' start positions, $\mathcal{V} = \{1, \cdots, N\}$ and M demand locations, $\mathcal{D} = \{1, \cdots, M\}$ in the environment. The environment is represented by an overhead, georeferenced, RGB image as shown in Figure 2. The goal is to find offline paths for each vehicle such that they collectively serve all the demands using navigation costs derived from the overhead images.

The cost of a path in the environment can be estimated by first performing a semantic segmentation of the overhead image. However, semantic segmentation is typically imperfect [3] and as such the estimated cost of a path may not be accurate. The problem we address in this paper is that of finding paths for vehicles to collectively serve all demands under travel-cost uncertainty.

We are motivated by tasks that are urgent and time-critical, such as fighting fires [13] and delivering medical supplies in emergencies [14]. When the number of vehicles is more than the demands, assigning multiple redundant vehicles to demands helps counter the effect of uncertainty [15]. The waiting time at a demand location is the time taken by the earliest vehicle to arrive at that location. If the travel times are deterministic, then it is known in advance which vehicle

will arrive first. When travel times are uncertain, as in this work, the arrival time of the earliest vehicle itself is a random variable. The goal is to assign vehicles to demand locations and find corresponding paths for the vehicles from the start to the assigned demand locations.

For convenience, we convert the minimization problem into a maximization one by taking the reciprocal of the travel cost. Specifically, we use the travel *efficiency*, the reciprocal of travel cost, as the measure. Thus, the travel efficiency of a demand location is the maximum of the travel efficiencies of the vehicles that reach this demand location. The overall travel efficiency, denoted by f is the sum of the travel efficiencies of all demand locations. Notably, f is also a random variable.

Our goal is to find risk-aware paths from vehicles' start positions to demand locations given a user-defined risk level α. We formulate a risk-aware path finding problem by maximizing CVaR_α on the travel efficiency (Problem 1).

Problem 1 (Risk-Aware Path Finding)

$$\max_{\mathcal{S}\subseteq\mathcal{X}} \text{CVaR}_\alpha[f(\mathcal{S}, y)] \tag{3}$$

where $\mathcal{S}$ is a path set for vehicles with "per path per vehicle", $\mathcal{X}$ is a ground set of paths from which $\mathcal{S}$ is chosen, and $f(\mathcal{S}, y)$ is the travel efficiency on the path set $\mathcal{S}$, with randomness induced by y.

III. Framework and Analysis

The framework we propose consists of three main parts: semantic segmentation, candidate path generation, and risk-ware assignment. The overall pipeline of the framework and its parts are shown in Figure 2. The inputs to the framework are a single overhead aerial two-dimensional image, vehicles' start positions, and demand locations. The output is a risk-aware assignment of paths from vehicles' start positions to demand locations.

The input is first semantically segmented into per pixel labels. These labels are assigned a cost proportionate to the risk involved in traversing them. The cost map and the uncertainty associated with the segmentation are then used as input to a path planner which generates candidate paths for assignment. Finally, the candidate paths from each vehicle's start position to each demand location are computed by maximizing CVaR, for risk-aware path assignment.

A. Semantic Segmentation

To plan a path for a vehicle in the environment, we need to first recognize features in the image, such as road, person, car, and so on. Given an input aerial image provided by an unmanned aerial vehicle (UAV), a deep learning model is used to provide a semantically segmented map. We then utilize an approximation technique, using dropout, to learn the uncertainty in the semantic labels provided by the model [6]. The advantage of using uncertainty has been studied [3]. After semantic segmentation and uncertainty extraction, we assign different cost values for the robot traveling on different terrain features. For example, the cost of traveling on the road is less than that of traveling on vegetation. Traveling on a person and cars is impossible, and thus, the corresponding cost can be set to infinity. Notably, the cost can be the energy or time spent by a vehicle traveling on the terrains. Since there exists uncertainty in the terrain features provided by the deep learning model, the cost of traveling on them is also a random variable with some uncertainty.

B. Candidate Path Generation

Once a risk-aware cost map is generated based on the semantic segmentation of the terrain, augmented by the confidence in prediction, candidate paths are generated from each vehicle location to each demand location by A* planner. For the different combinations of vehicles and demands, a candidate path represents a potential feasible route given on the aerial map. Relying on deep learning segmentation alone can be risky. Given confidence information, we expect paths to avoid regions of high uncertainty which could involve out-of-distribution data or misclassifications, as shown in Figure 4.

For A* planner, a risk-aware cost function $C(x)$ is defined on each pixel x. It combines the cost associated with the classified terrain type and the variance.

$$C(x) := C(l_x) + \lambda \texttt{Var}(x), \tag{4}$$

where l_x refers to the most likely predicted label for a pixel x. This function assigns a cost to each pixel x, characterised by a user-defined cost map and a weight parameter λ to quantify emphasis on the uncertainty. We use variance in prediction as a measure of uncertainty.

C. Risk-Aware Path Assignment

Consider a set of K candidate paths, $\mathcal{P} = \{1, \cdots, K\}$ are generated from each vehicle's start position to each demand location in Section III-B. We then assign each vehicle a path to a demand location. As mentioned in Section II-B, we follow a redundant assignment setting [15] where each vehicle can be assigned to at most one demand location and multiple vehicles can be assigned to the same demand location. Only the vehicle through a path arrives with the highest efficiency is chosen at each demand location.

Denote the travel efficiency for vehicle starting at $i \in \mathcal{V}$ arriving at demand location $j \in \mathcal{D}$ through path $k \in \mathcal{P}$ as e_{ijk}. Correspondingly, we take the tuple (i, j, k) as an assignment where vehicle-path pair (i, k) is assigned to demand location j. Denote the total efficiency at all M demand locations as,

$$f(\mathcal{S}, y) = \sum_{j\in M} \max_{(i,j,k)\in\mathcal{S}_j} \; e_{ijk} \tag{5}$$

where $\mathcal{S}_j$ denotes the assignment set $\{(i, j, k)\}$ to demand location j. The total assignment set $\mathcal{S} := \bigcup_{j=1}^{M} \mathcal{S}_j$ is a collection of assignment sets at all demand locations. Notably, since each vehicle-path pair (i, k) can be assigned

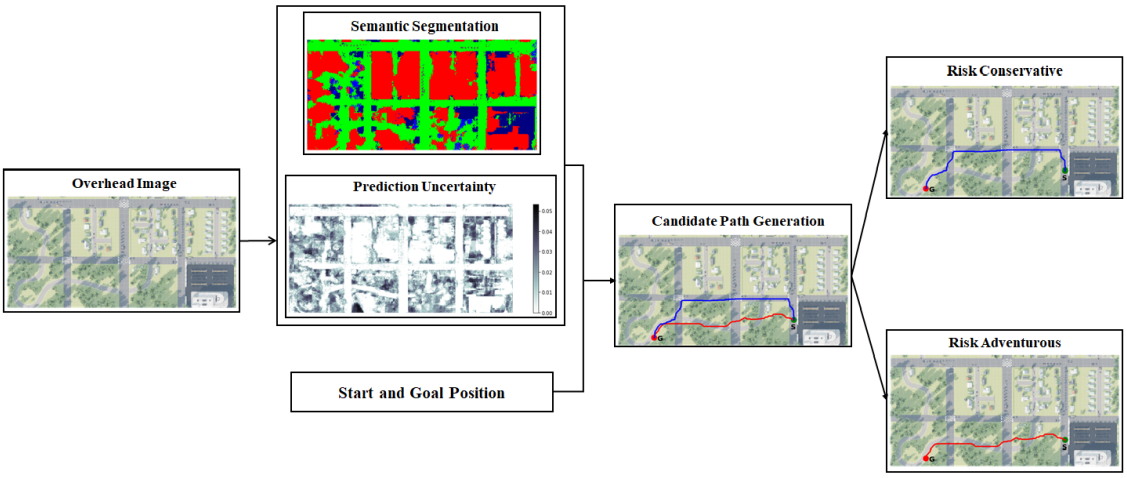


Fig. 2. The breakdown of the framework's parts. Given an overhead image input, the algorithm provides a semantic segmentation and uncertainty map, then generates candidate paths, and finally performs the risk-aware path assignment of vehicles to demands.

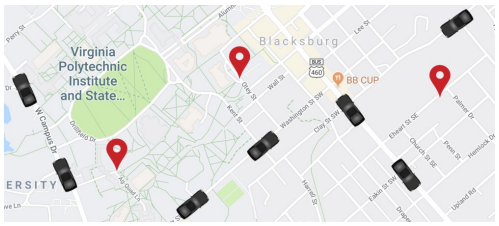


Fig. 3. Mobility-on-demand with multiple demands (red anchor) and multiple self-driving vehicles (black car).

to at most one demand location, all $\mathcal{S}_j$(s) are disjoint, i.e., $\mathcal{S}_j \cap \mathcal{S}_{j'} = \emptyset,\ \forall j \neq j', j, j' \in \mathcal{D}$. This is called a partition matroid constraint in the literature [16], which we denote by $\mathcal{I}$.

We use the "max" operator to capture the selection rule that only the vehicle-pair (i, k) with the maximum efficiency is chosen at each demand location. Due to the "max" operator, the total efficiency $f(\mathcal{S}, y)$ turns out to be monotone submodular in $\mathcal{S}$. If there is no assignment, we set $f(\emptyset, y) = 0$ to make f normalized. Here, y indicates the randomness of $f(\mathcal{S}, y)$ due to the uncertainty in travel efficiency.

Then, by utilizing Equation 1 and the candidate paths generated in Section III-B, we transform Problem 1 to a risk-aware path assignment problem below.

Problem 2 (Risk-Aware Path Assignment)

$$\max_{\mathcal{S}\subseteq\mathcal{X}} \ \tau - \frac{1}{\alpha}\mathbb{E}[(\tau - \sum_{j\in M} \max_{(i,j,k)\in\mathcal{S}_j} e_{ijk})_+]$$
$$s.t.\ \mathcal{S} = \bigcup_{j=1}^{M} \mathcal{S}_j,\ \mathcal{S} \in \mathcal{I},$$
$$\tau \in [0, \Gamma], \qquad (6)$$

with $\mathcal{S}$ the path assignment set ("per path per vehicle"), $\mathcal{I}$ the partition matroid constraint [16], $\mathcal{X}$ the ground set of all possible assignments, $\{(i, j, k)\},\ k \in \mathcal{P},\ i \in \mathcal{V},\ j \in \mathcal{D}$, and $\Gamma \in \mathbb{R}^+$ the upper bound of the parameter τ. Γ can be set as an upper bound on $f(\mathcal{S}, y)$ (Eq. 5).

Algorithm 1: Risk-Aware Path Assignment

Input:
- Vehicles' initial positions $\mathcal{V}$, demand locations $\mathcal{D}$, and path set $\mathcal{P}$
- User-defined risk level $\alpha \in [0, 1]$
- Range of the parameter $\tau \in [0, \Gamma]$ and discretization stage $\Delta \in (0, \Gamma]$
- An oracle $\mathcal{O}$ that approximates $H(\mathcal{S}, \tau)$ as $\hat{H}(\mathcal{S}, \tau)$

Output:
- Path assignment $\mathcal{S}$

1: $\mathcal{M} \leftarrow \emptyset$
2: **for** $i \in \{0, 1, \cdots, \frac{\Gamma}{\Delta}\}$ **do**
3: $\quad \tau^i = i\Delta$
4: $\quad \mathcal{S}^i \leftarrow \emptyset$
5: $\quad$ **for** $l = 1 : |\mathcal{D}|$ **do**
6: $\quad\quad (i^\star, j^\star, k^\star) = \underset{i\in\mathcal{V}, j\in\mathcal{D}, k\in\mathcal{P}}{\operatorname{argmax}} \ \hat{H}(\mathcal{S}^i \cup (i, j, k), \tau^i) - \hat{H}(\mathcal{S}^i, \tau^i)$
7: $\quad\quad \mathcal{S}^i \leftarrow \mathcal{S}^i \cup (i^\star, j^\star, k^\star)$
8: $\quad\quad \mathcal{V} = \mathcal{V} \setminus i^\star$
9: $\quad$ **end for**
10: $\quad \mathcal{M} = \mathcal{M} \cup \{(\mathcal{S}^i, \tau^i)\}$
11: **end for**
12: $(\mathcal{S}, \tau^\star) = \underset{(\mathcal{S}^i, \tau^i)\in\mathcal{M}}{\operatorname{argmax}} \ \hat{H}(\mathcal{S}^i, \tau^i)$

Building on the sequential greedy algorithm (SGA) from our previous work [2], we present Algorithm 1 for solving Problem 2.

These are four stages in Algorithm 1:

a) Initialization (line 1): We initialize a storage set $\mathcal{M}$ to be empty. We use $\mathcal{M}$ to store the assignment $\mathcal{S}$ with the corresponding τ when searching all possible values of τ.

*b) Searching for τ (**for** loop in lines 2–11):* We sequentially search for all possible values of τ within $[0, \Gamma]$ by

a user-defined separation Δ (line 3). Γ is an upper bound on τ and can be set by the user based on the specific problem at hand. We show how to compute Γ in specific scenarios in Section IV.

c) Greedy algorithm (lines 4–9): For a specific τ, e.g., τ_i, we use the greedy approach [16] to choose the corresponding assignment set $\mathcal{S}^i$. We first initialize set $\mathcal{S}^i$ to be the empty set (line 4). Then we execute the greedy algorithm in $|\mathcal{D}|$ rounds (line 5), since the total number of demand locations to be served is $\mathcal{D}$. At each round, the assignment $(i^\star, j^\star, k^\star)$ which gives the maximum marginal gain of $\hat{H}(\mathcal{S}^i, \tau^i)$ is selected (line 6) and added into set $\mathcal{S}^i$ (line 7). Then, we remove the vehicle position $i^\star$ from $\mathcal{V}$ (line 8) to make sure vehicle position $i^\star$ and the corresponding paths will never be selected in the following rounds.

d) Find the best assignment (line 12): From the collection of $(\mathcal{S}, \tau)$ pairs, $\mathcal{M}$ (line 10), we pick the one that maximizes $\hat{H}(\mathcal{S}^i, \tau^i)$ as the output $\mathcal{S}$ with the corresponding τ denoted by $\tau^\star$ (line 12).

Oracle Design: We use an oracle $\mathcal{O}$ to calculate the value of $H(\mathcal{S}, \tau)$. The oracle uses a sampling based method to approximate $H(\mathcal{S}, \tau)$ [2].

It has been shown in [2] that Algorithm 1 generates an assignment that has the bounded approximation performance of the optimal assignment.

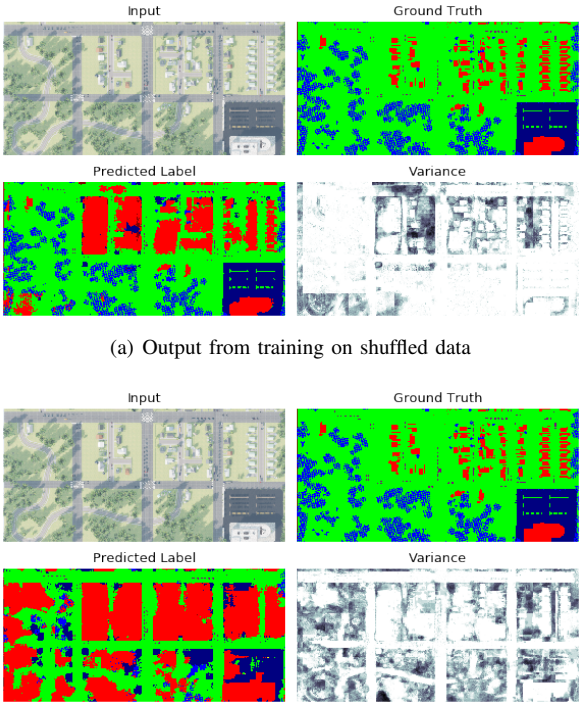
IV. Evaluation

In this section, we report the results from empirical studies evaluating the proposed risk-aware perception and planning framework. We start by describing the experimental setup and then describe the results.

Setup. We use AirSim [17] simulator as it offers photo-realistic images along with the semantically segmented ground truth in multiple pre-defined environments. We use *CityEnviron* environment which contains city-like and suburban landscapes. We collect the down-looking aerial images at an altitude value of 200m in the simulator. The dataset thus generated has 480 images, which is divided in the ratio of 10:3:3 as training, validation, and testing data for the Bayesian SegNet [18]. We use the PyTorch implementation of a basic version of this network [19] and add dropout layers in accordance with the original architecture. In our ground truth, we reduce the ground truth to 12 classes consistent with the original model [18].

Due to limitations in the simulator, roads and grass patches are indistinguishable in the ground truth as depicted in Figure 4. We keep such images limited to the test dataset. Hence, the grass patches act as unknown objects (similar to out-of-distribution) to the model and provide interesting observations on the uncertainty in prediction for such objects.

We produce 20 outputs (or *stochastic samples*) for each images using the trained Bayesian SegNet. For each pixel, the predicted label found as the most frequent label among all the most likely labels for each pixel, i.e., $l_{p,ml} = \texttt{Mode}_{\text{samples}}(\arg\max_c P(c|x))$ for a pixel p, where x is the input to the model, c is the class/label, and `mode` stands for the statistical mode. Uncertainty for each pixel is defined as the average variance in the probabilities of labels i.e. $\texttt{uncertainty}(p) = \frac{1}{c}\sum_c \texttt{Var}(P(c|x))$. The cost for each pixel is calculated using these two quantities, which is further fed to the path planning algorithm (Sec. III-B). To provide an orthographic view in the results, we make predictions over partially overlapping images with perspective projections. Then we combine subsets around the center of the images.



(a) Output from training on shuffled data

(b) Output from training on unshuffled data

Fig. 4. Difference in variance due to data distribution.

Results. *(a), Out-of-Distribution Data.* The Bayesian SegNet model used in our path planning algorithm has dissimilar data distribution for training and testing. The effect of this is shown in Figure 4. When the data distributions are the same across training and test dataset (shuffled data), the model is able to make predictions with very low variance (implied by brighter shades). However, in the case of dissimilar distribution, some part of the image acts as unseen data and thus the model prediction has high variance. This highlights the importance of uncertainty. For example, the prediction is precise for already observed data and thus decision involving such input would be less risky.

(b), Average Cross-Entropy on the Test Dataset. In order to understand the correlation between the quality of prediction and the training data, we look at the cross-entropy of the model prediction (over unshuffled data) averaged over the 20 samples, and the number of pixels in the training data for each class in Figure 5. The model performs comparatively well in identifying obstacles like buildings. Objects which are rarely observed in the ground truth have very high cross-entropy. The lack of difference between the ground and the road in the ground truth also affects the performance of relevant classes. As expected, high cross-entropy is observed

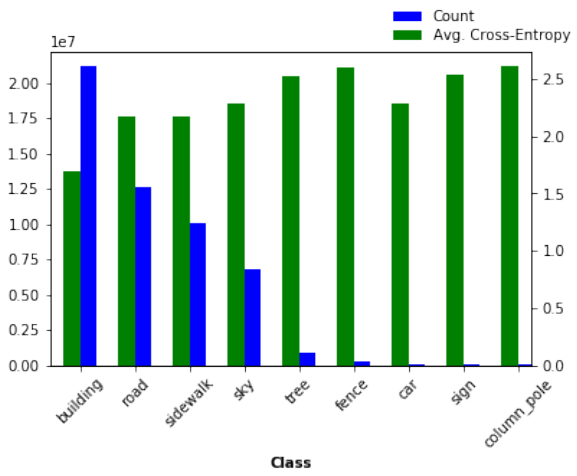
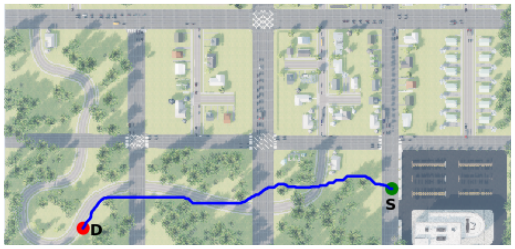


Fig. 5. Average Cross-Entropy for each class/label.



(a) λ = 10

(b) λ = 50

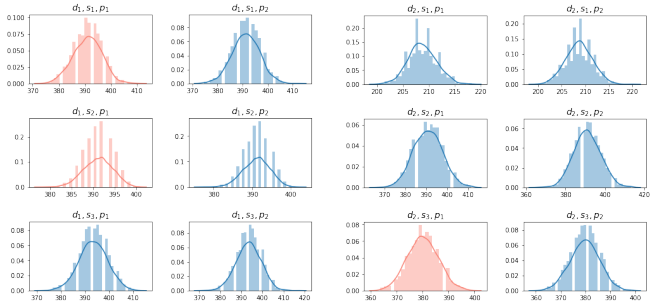
Fig. 6. Effect of λ on path planning. (a), the smaller λ gives a shorter path with high uncertainty; (b) the larger λ gives a longer path with low uncertainty.

for classes where the pixel count is low.

(c), Risk-Aware Path Planning and Assignment. Value of λ decides the risk-awareness of path planning. This effect is shown in Figure 6 where the cost of each navigable class is 1 and for non-navigable classes, the cost is 3 (except for the tree class, where it is 2). For a small value of λ, the algorithm plans a short path passing through the vegetation. For a high value of λ, this area is avoided due to high uncertainty in the prediction for this part of the scene.

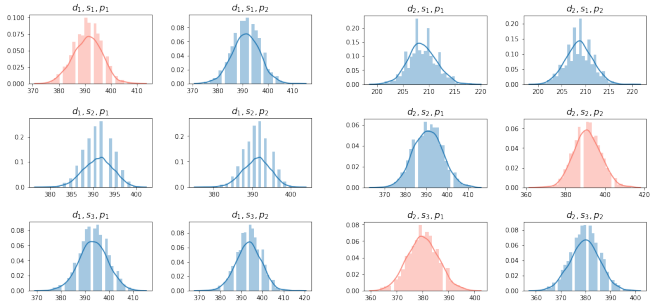
By varying the values of λ, we generate $K = 2$ candidate paths from each vehicle's start position to each demand location. We consider assigning $N = 3$ supply vehicles to $M = 2$ demand locations in this unknown environment (Fig. 6).

Due to the imperfectness of semantic segmentation, the



(a) Demand location d_1 (b) Demand location d_2

Fig. 7. Efficiency distributions of paths and the path assignment when $\alpha = 0.01$. The assigned path for each robot is marked in red.



(a) Demand location d_1 (b) Demand location d_2

Fig. 8. Efficiency distributions of paths and the path assignment when $\alpha = 1$. The assigned path for each robot is marked in red.

cost/efficiency of the path is a random variable. We show the efficiency distributions of the paths from vehicles' start positions to demand locations in Figure 7 and Figure 8.

We use Algorithm 1 to assign each vehicle a risk-aware path to a demand location. For example, in Figure 7-(a), vehicle v_2 is assigned path p_1 for demand location d_1 when the risk level is small, e.g., $\alpha = 0.01$. In contrast, when the risk level is high, e.g., $\alpha = 1$, the assignment is more adventurous, and thus the path with a larger mean efficiency and a larger variance is selected. As shown in Figure 8-(b), vehicle v_2 switches to path p_2 for demand location d_2. Because the efficiency of this path has a larger mean. The path assignment changes follows the comparison of CVaR values, i.e., $\text{CVaR}_{0.01}[e(p_1)] > \text{CVaR}_{0.01}[e(p_2)]$ and $\text{CVaR}_1[e(p_1)] < \text{CVaR}_1[e(p_2)]$ with $e(p_1)$ and $e(p_2)$ denoting the efficiency of path p_1 and p_2, respectively. Thus, the risk level, α, provides a user with the flexibility to trade off between risk and total efficiency (reward).

(d). Quantitative results. In order to quantify the effect of λ, we define a metric called *surprise* for a path as the difference between the cost of the ground truth labels and predicted labels. We take 6 combinations of start and demand positions given in Figure 9, and find the average value of *surprise* as shown in Figure 10. Ideally, we expect the *surprise* to reduce with emphasis on variance. However, for large values of λ, even a few pixels with high variance may greatly increase the cost. In general, a higher value of λ may cause the robot to choose a longer path, which may

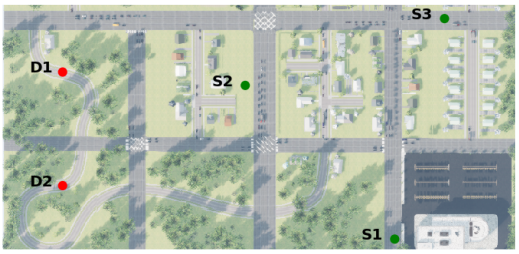


Fig. 9. Start and demand positions for *surprise* calculations.

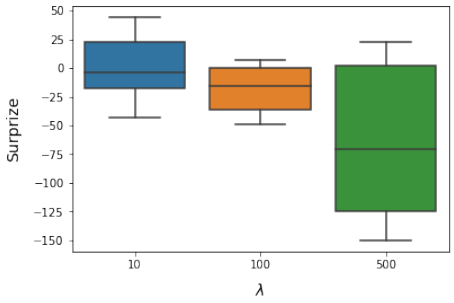


Fig. 10. *surprise* vs λ.

increase the cost of traversal. However, sometimes the path may pass though only navigable regions, resulting in a small value of *surprise*. This causes the *surprise* to have a larger variance. Thus, the value of λ should be chosen after careful consideration of the variance in model prediction and the range of the cost mapping.

We also plot the distribution of total travel efficiency (Eq. 5) in Fig. 11. $\mathcal{S}$ is the path assignment selected by Algorithm 1. With small risk levels α, paths with low efficiencies (equivalently, low uncertainty) are mostly selected. This is because a small risk level indicates the assignment is conservative and only willing to take a little risk. Thus, the path with lower efficiency and lower uncertainty is assigned to avoid the risk induced by the uncertainty. In contrast, when α is large, the assignment is riskier. In such a case, the paths with high efficiencies (equivalently, high uncertainty) are selected.

V. Conclusion

In this paper, we propose a risk-aware path planning and assignment framework for vehicles navigating in an unknown environment. We consider a scenario that the information of the unknown environment can only be available by an overhead, georeferenced image that is taken by an aerial robot. To deal with this challenge, we utilize Bayesian deep learning to learn the environment though the semantic segmentation. Since the output of this segmentation is noisy, the cost of the vehicle traversing in the environment is uncertain. To deal with the cost uncertainty, we optimize some popular risk measures to generate and assign risk-aware paths for the vehicles. We use extensive simulation results to show the effectiveness of our risk-aware strategy.

Future work will focus on the online coordination of aerial and ground vehicles to achieve long-term, real-time risk-aware navigation in unknown environments.

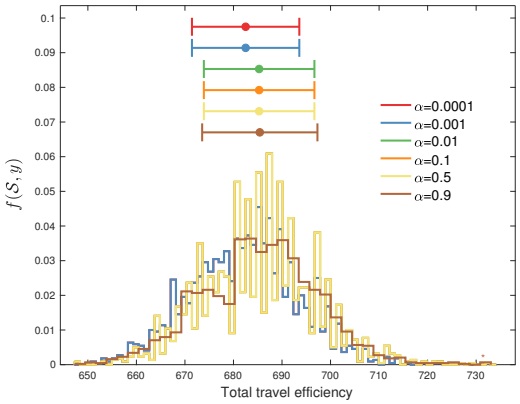


Fig. 11. Distribution of the total travel efficiency $f(\mathcal{S}, y)$ by Algorithm 1.

References

[1] M. Toubeh and P. Tokekar, "Risk-aware planning by extracting uncertainty from deep learning-based perception," in *AAAI 2018 Fall Symposium on Reasoning and Learning in Real-World Systems for Long-Term Autonomy*, 2018, pp. 82–87.

[2] L. Zhou and P. Tokekar, "An approximation algorithm for risk-averse submodular optimization," in *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2018, in press.

[3] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, University of Cambridge, January 2017.

[4] H. Wang, X. Shi, and D.-Y. Yeung, "Natural-parameter networks: A class of probabilistic neural networks," 2016.

[5] J. Gast and S. Roth, "Lightweight probabilistic deep networks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2018.00355

[6] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *International Conference on Machine Learning*, 2016, pp. 1050–1059.

[7] A. Loquercio, M. Segu, and D. Scaramuzza, "A general framework for uncertainty estimation in deep learning," *IEEE Robotics and Automation Letters*, p. 11, 2020. [Online]. Available: http://dx.doi.org/10.1109/LRA.2020.2974682

[8] A. Majumdar and M. Pavone, "How should a robot assess risk? towards an axiomatic theory of risk in robotics," in *Robotics Research*. Springer, 2020, pp. 75–84.

[9] S. I. Marcus, E. Fernández-Gaucherand, D. Hernández-Hernandez, S. Coraluppi, and P. Fard, "Risk sensitive markov decision processes," in *Systems and control in the twenty-first century*. Springer, 1997, pp. 263–279.

[10] J. J. Chung, A. J. Smith, R. Skeele, and G. A. Hollinger, "Risk-aware graph search with dynamic edge cost discovery," *The International Journal of Robotics Research*, vol. 38, no. 2-3, pp. 182–195, 2019.

[11] R. T. Rockafellar, S. Uryasev *et al.*, "Optimization of conditional value-at-risk," *Journal of risk*, vol. 2, pp. 21–42, 2000.

[12] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[13] K. Harikumar, J. Senthilnath, and S. Sundaram, "Multi-uav oxyrrhis marina-inspired search and dynamic formation control for forest fire-fighting," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 2, pp. 863–873, 2018.

[14] E. Ackerman and E. Strickland, "Medical delivery drones take flight in east africa," *IEEE Spectrum*, vol. 55, no. 1, pp. 34–35, 2018.

[15] A. Prorok, "Redundant robot assignment on graphs with uncertain edge costs," in *Distributed Autonomous Robotic Systems*. Springer, 2019, pp. 313–327.

[16] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey, "An analysis of approximations for maximizing submodular set functions–ii," in *Polyhedral combinatorics*. Springer, 1978, pp. 73–87.

[17] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and service robotics*. Springer, 2018, pp. 621–635.
[18] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," *arXiv preprint arXiv:1511.02680*, 2015.
[19] M. P. Shah, "Semantic segmentation architectures implemented in pytorch." *https://github.com/meetshah1995/pytorch-semseg*, 2017.