FUEL: Fast UAV Exploration using Incremental Frontier Structure and Hierarchical Planning

Abstract

- 본 논문에서는 복잡한 unknown environments에서 빠른 UAV Exploration이 되도록 도움을 주는 FUEL을 제안하였다.

Introduction

Unmanned aerial vehicles은 많은 분야에서 인기를 얻어왔다. 많은 method가 제안되어 왔지만, 거의 대부분은 큰 공간에 적용하지 못할만큼 좋은 결과를 내지는 못하였다. 하지만 본 논문에서 제안한 방법은 structure이 효율적으로 업데이트 될 수 있도록 한다.

System Overview

본 논문의 framework는 voxel grid map에서 작동된다. 이 method는 FIS의 증가하는 업데이트와 hierarchical exploration planning approach로 구성되어 있다. 센서를 통해 map이 업데이트 될 때에 frontier cluster이 영향을 받았는지 확인한다.

Incremental Frontier Information Structure

Frontier은 unknown voxel에 인접한 known-free voxel로 정의된다. 본 논문에서는 frontier에서 많은 정보를 뽑아내어 정교한 planning이 가능하게 하였다.

Frontier Information Structure

Frontier information structure인 FI_i 는 새로운 frontier cluster인 F_i 가 생성될 때 계산된다. FI_i 은 cluster 안에 속하는 모든 cell인 C_i 를 저장하고 평균 위치인 $p_{avg,i}$ 를 뽑아낸다. Axis-aligned bounding box인 B_i 도 frontier 변화의 detection을 가속 하기위해 연산

된다. Exploration planning을 위해 candidate viewpoints인 VP_i 가 cluster 근처에서 생성된다. 그리고 F_i 와 모든 cluster 간의 connection cost를 포함하는 $L_{cost,i}$ 가 계산된다.

Data	Explanation
C_i	Frontier cells that belong to the cluster
$\mathbf{p}_{\text{avg},i}$	Average position of C_i
B_i	Axis-aligned bounding box of C_i
VP_i	Viewpoints covering the cluster
$L_{\text{cost},i}$	Doubly linked list of connection costs to all other clusters

Incremental Frontier Detection and Clustering

map은 센서를 통해 업데이트 된다. 또한 그에 따라 B_m 도 업데이트 되는데, 먼저 모든 cluster을 통과하고 새로운 B_i 와 겹치는 부분만이 반환된다. 그 후, precise check가 겹치는 부분에 대해 시행되어 더 이상 frontier이 아닌 cell을 제거한다. 그 후에 새로운 frontier이 region growing 알고리즘을 통해 탐색되고 그룹으로 cluster 된다. 그 그룹 중에서 cell 수가 적은 그룹은 무시한다. 남은 그룹이 너무 큰 cluster을 가지게 될 수도 있다. 이를 해결하기 위해 cluster에 PCA를 진행하여 여러 작은 cluster으로 나눈다.

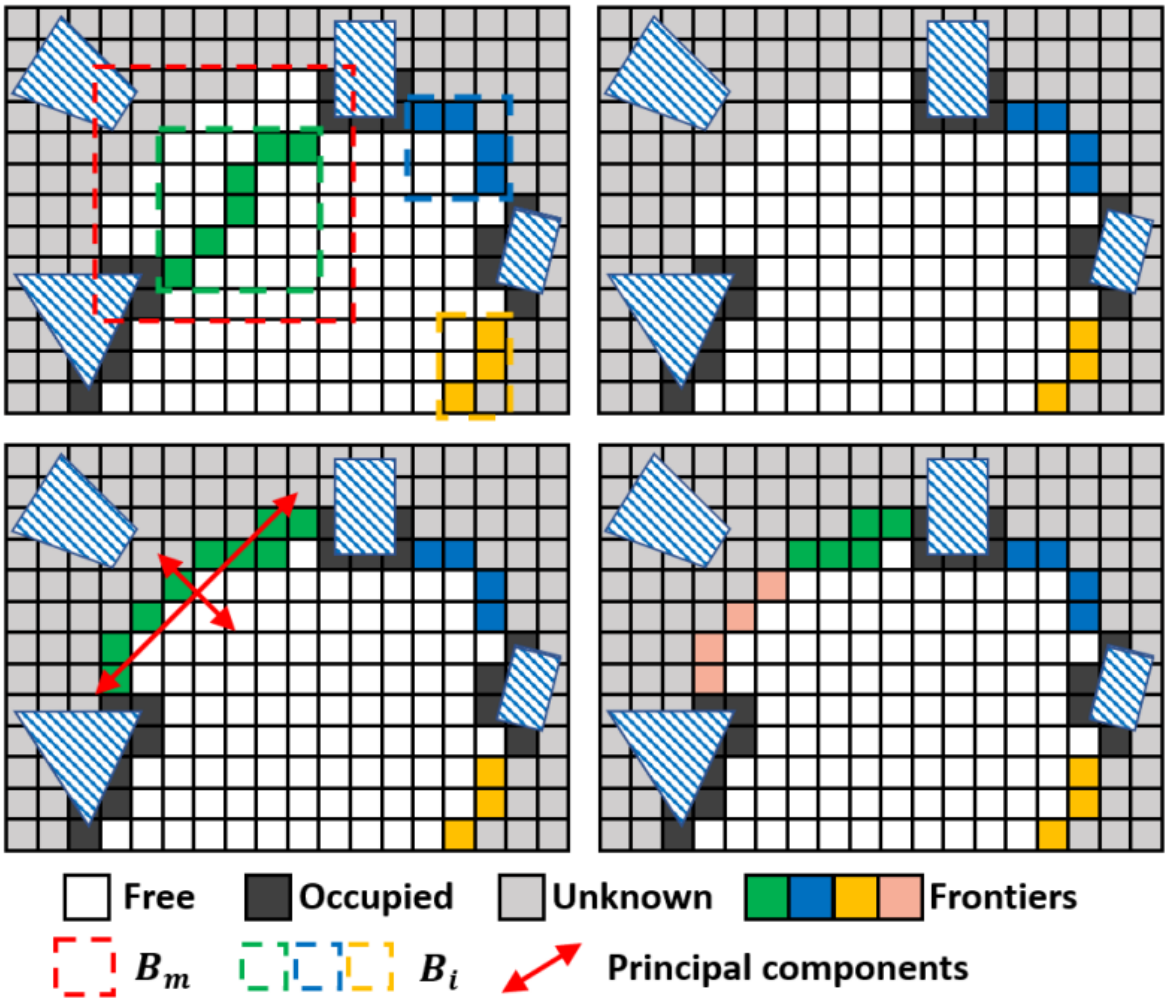

Fig. 3. Incremental frontier detection and clustering. Top: detecting and removing outdated frontiers. Bottom: new frontier is detected (left) and PCA is performed, the large cluster is split into two smaller ones (right).

Viewpoint Generation and Cost Update

본 논문의 frontier cluster이란 직관적으로 탐색할 잠재적 공간을 의미한다. cluster의 중심 부분으로 찾아가던 이전의 method와는 다르게, 본 논문의 저자들은 더욱 정교한 decision making을 추구하였다. 이를 위해 cluster인 F_i 가 생성될 때에 viewpoints의 집합인 $VP_i = \{x_{i,1}, x_{i,2}, ..., x_{i,n_i}\},\ where\ x_{i,j}(p_{i,j}, \xi_{i,j})$ 를 생성한다. 이는 cluster의 중심을 기반으로 원통형 좌표계에서 점을 균일하게 sampling하여 구한다.

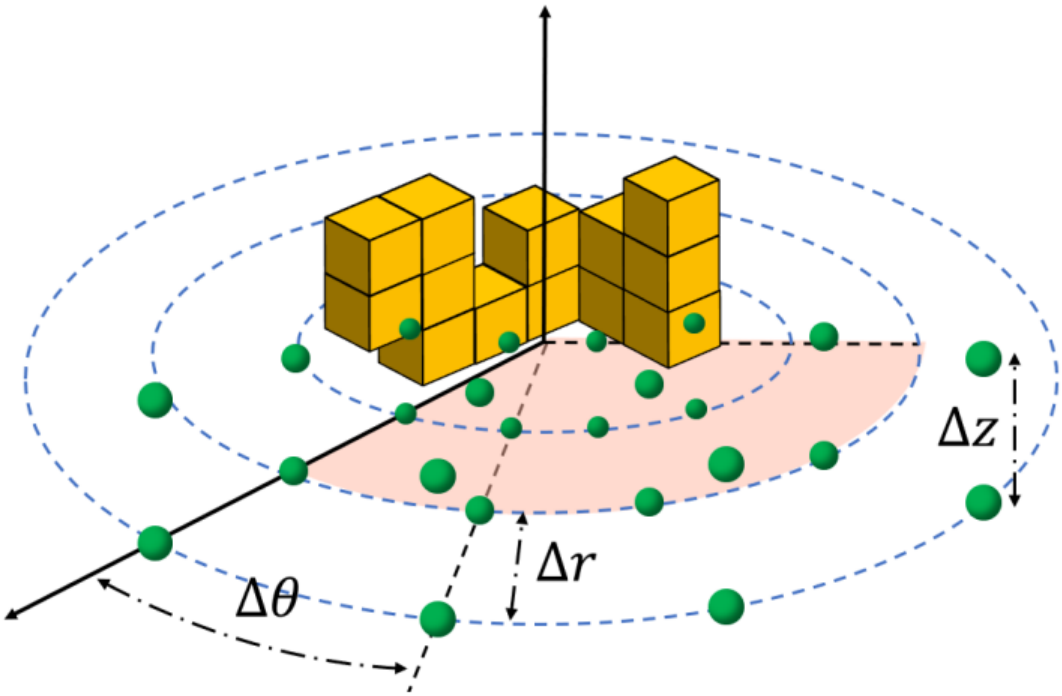

Fig. 4. Generating candidate viewpoints for a frontier cluster. Within the cylindrical coordinate system centered at the average position of the cluster, points are sampled uniformly.

각각의 sampled point인 p 에 대해 yaw angle인 ξ 는 cluster에 대한 센서 coverage를 yaw optimization을 통해 최대화 하는 것으로 결정된다. Coverage는 occupied voxel에 가려지지 않은 frontier 수와 sensor model을 준수하는 voxel 수로 평가된다. 그 후 threshold 보다 높은 viewpoint가 내림차순으로 정렬된다.

Global planning of exploration tour을 수행하기 위해, 각 cluster의 pair 사이의 connection cost가 필요하다.

$$t_{\mathrm{lb}}(\mathbf{x}_{k_1,j_1},\mathbf{x}_{k_2,j_1}) = \max\left\{\frac{\mathrm{length}\left(P\left(\mathbf{p}_{k_1,j_1},\mathbf{p}_{k_2,j_2}\right)\right)}{v_{\max}}, \frac{\min\left(\left|\xi_{k_1,j_1}-\xi_{k_2,j_2}\right|, 2\pi-\left|\xi_{k_1,j_1}-\xi_{k_2,j_2}\right|\right)}{\dot{\xi}_{\max}}\right\} \quad (1)$$

- $t_{lb}(x_{k_1,j_1}, x_{k_2,j_1})$: 두 viewpoint인 x_{k_1,j_1}, x_{k_2,j_1} 를 이동할 때의 time lower bound
- $P(p_{k_1,j_1}, p_{k_2,j_1})$: p_{k_1,j_1}, p_{k_2,j_1} 사이의 충돌이 없는 경로, A 스타 알고리즘을 통해 계산된다.

각 pair (F_{k_1}, F_{k_2}) 마다 cost인 $t_{lb}(x_{k_1,1}, x_{k_2,1})$ 을 계산한다.

Hierarchical Exploration Planning

본 논문에서는 그리디 알고리즘 등이 아닌 다른 효과적인 비행 경로를 생성하는 방법을 제안하였다.

Global Exploration Planning

본 논문에서는 TSP의 변형 버전을 공식화 하였다.

N_{cls} cluster이 완전히 있다고 가정할 때에, M_{tsp} 가 $N_{cls}+1$ 차원의 square matrix와 대응된다.

$N_{cls} \times N_{cls}$ block이 아래와 같은 식을 통해 계산된다.

$$\begin{aligned}&\mathbf{M}_{\text{tsp}}(k_1,k_2)=\mathbf{M}_{\text{tsp}}(k_2,k_1)\\&=t_{\text{lb}}(\mathbf{x}_{k_1,1},\mathbf{x}_{k_2,1}),\ \ k_1,k_2\in\{1,2,\cdots,N_{\text{cls}}\}\end{aligned}$$

위 정보는 frontier이 감지될 때에는 유지된다.

M_{tsp} 의 첫 row와 column은 viewpoint인 $x_0=(p_0,\xi_0)$ 와 N_{cls} cluster과 연관된다. x_0 에서 시작해서 k-th cluster은 아래와 같이 evaluate 된다.

$$\begin{aligned}\mathbf{M}_{\text{tsp}}(0,k)=t_{\text{lb}}(\mathbf{x}_0,\mathbf{x}_{k,1})+w_{\text{c}}\cdot c_{\text{c}}(\mathbf{x}_{k,1}),\\k\in\{1,2,\cdots,N_{\text{cls}}\}\end{aligned}$$

위 식에서 motion consistency $c_c(x_k,1)$ 은 다음과 같다.

$$c_{\text{c}}(\mathbf{x}_{k,j})=\cos^{-1}\frac{(\mathbf{p}_{k,j}-\mathbf{p}_0)\cdot\mathbf{v}_0}{\|\mathbf{p}_{k,j}-\mathbf{p}_0\|\,\|\mathbf{v}_0\|}$$

- v_0 : 현재 속도

마지막으로 $M_{tsp}(k,0)$ 을 0으로 할당하여 ATSP로 줄일 수 있다.

결국 이러한 방식은 추가 비용이 발생하지 않아 더욱 개선된 형태로 볼 수 있으며, 최적의 open-loop tour이 가능하게 한다.

Local Viewpoint Refinement

Global tour planning은 모든 cluster을 방분하는 promising order을 찾는다. 하지만 이는 단일 viewpoint에서의 정보만 포함하기에 완전하지 못하다.

따라서 그래프 탐색을 이용해 더욱 다양한 viewpoints에서의 정보를 고려한다. 그림으로 보자면 다음과 같다.

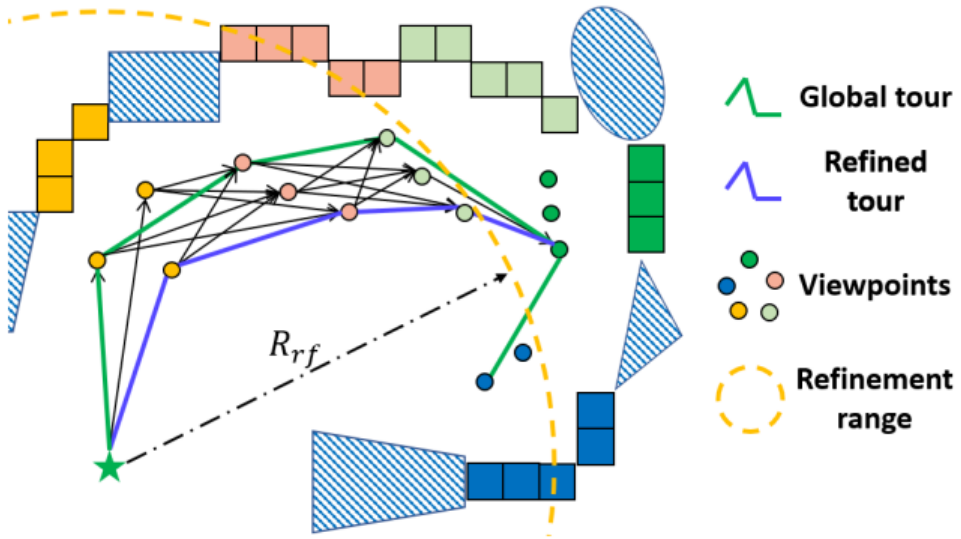

Fig. 6. Refining viewpoints locally using the graph search approach. Along a truncated segment of the global tour, multiple viewpoints of each visited cluster are considered to select the optimal set of viewpoints.

본 논문의 저자들은 Dijkstra(다익스트라) 알고리즘을 통해 최적을 루트를 탐색하였다.

$$c_{\mathrm{rf}}(\Xi) = t_{\mathrm{lb}}(\mathbf{x}_0, \mathbf{x}_{1,j_1}) + w_{\mathrm{c}} \cdot c_{\mathrm{c}}(\mathbf{x}_{1,j_1}) + t_{\mathrm{lb}}(\mathbf{x}_{N_{\mathrm{rf}},j_{N_{\mathrm{rf}}}}, \mathbf{x}_{N_{\mathrm{rf}}+1,1}) + \sum_{k=1}^{N_{\mathrm{rf}}-1} t_{\mathrm{lb}}(\mathbf{x}_{k,j_k}, \mathbf{x}_{k+1,j_{k+1}}) \tag{6}$$

(위 식을 최소화하도록 노력하며 최적의 루트를 탐색한다.)

Minimum-time B-spline Trajectory

불연속적인 viewpoint에서는 연속적인 궤적이 안정적인 비행을 위해 필요하다. Quadrotor dynamics이 differentially flat하기 때문에, 본 논문의 저자들은 flat한 출력 $x \in (x, y, z, \xi)$ 에 대한 궤적을 plan했다. $X_{cb} = \{x_{c,0}, x_{c,1}, ..., x_{c,N_b}\},\ where\ x_{c,i} \in (p_{c,i}, \xi_{c,i})$ 으로 가정하자(p_b : degree uniform B-spline, $\triangle t_b$: knot span). 본 논문의 저자들은 몇가지 조건(smoothness and

total trajectory time, and satisfies safety, dynamic feasibility and boundary state constraints)을 충족하는 B-spline 찾는다. 이를 수식으로 표현하자면 다음과 같다.

$$\underset{\mathbf{x}_{\mathrm{c},b},\Delta t_b}{\arg\min}\ f_{\mathrm{s}} + w_{\mathrm{t}}T + \lambda_{\mathrm{c}} f_c + \lambda_{\mathrm{d}}\left(f_{\mathrm{v}} + f_{\mathrm{a}}\right) + \lambda_{\mathrm{bs}} f_{\mathrm{bs}}$$

f_s : elastic band smoothness cost

$$f_{\mathrm{s}} = \sum_{i=0}^{N_b-2} \mathbf{s}_i^{\mathrm{T}} \mathbf{R}_{\mathrm{s}} \mathbf{s}_i,\ \ \mathbf{s}_i = \mathbf{x}_{\mathrm{c},i+2} - 2\mathbf{x}_{\mathrm{c},i+1} + \mathbf{x}_{\mathrm{c},i}$$

$$\mathbf{R}_{\mathrm{s}} = \begin{bmatrix} w_{\mathrm{s,p}}\mathbf{I}_3 & \mathbf{0} \\ \mathbf{0}^{\mathrm{T}} & w_{\mathrm{s},\xi} \end{bmatrix}$$

- R_s : penalty matrix
- T: $\triangle t_b$ 에 따른 총 trajectory time

$$T = (N_b + 1 - p_b) \cdot \Delta t_b$$

- f_c, f_v, f_a : 안전한 비행을 위한 penalty

$$f_{\mathrm{c}} = \sum_{i=0}^{N_b} \mathcal{P}\left(d\left(\mathbf{p}_{\mathrm{c},i}\right), d_{\min}\right)$$

$$\mathcal{P}(\tau_1, \tau_2) = \begin{cases} (\tau_1 - \tau_2)^2 & \tau_1 \leq \tau_2 \\ 0 & \text{else} \end{cases}$$

$$f_{\mathrm{v}} = \sum_{i=0}^{N_b-1} \left\{ \sum_{\mu \in \{x,y,z\}} \mathcal{P}(v_{\max}, |\dot{p}_{\mathrm{c},i,\mu}|) + \mathcal{P}(\dot{\xi}_{\max}, |\dot{\xi}_{\mathrm{c},i}|) \right\} \tag{13}$$

$$f_{\mathrm{a}} = \sum_{i=0}^{N_b-2} \left\{ \sum_{\mu \in \{x,y,z\}} \mathcal{P}(a_{\max}, |\ddot{p}_{\mathrm{c},i,\mu}|) + \mathcal{P}(\ddot{\xi}_{\max}, |\ddot{\xi}_{\mathrm{c},i}|) \right\} \tag{14}$$

따라서 control point는 utilize 된다.

$$\dot{\mathbf{x}}_{\mathrm{c},i} = \left[\dot{p}_{\mathrm{c},i,x}, \dot{p}_{\mathrm{c},i,y}, \dot{p}_{\mathrm{c},i,z}, \dot{\xi}_{\mathrm{c},i} \right]^{\mathrm{T}} = \frac{\mathbf{x}_{\mathrm{c},i+1} - \mathbf{x}_{\mathrm{c},i}}{\Delta t_b} \tag{15}$$

$$\ddot{\mathbf{x}}_{\mathrm{c},i} = \left[\ddot{p}_{\mathrm{c},i,x}, \ddot{p}_{\mathrm{c},i,y}, \ddot{p}_{\mathrm{c},i,z}, \ddot{\xi}_{\mathrm{c},i} \right]^{\mathrm{T}} = \frac{\mathbf{x}_{\mathrm{c},i+2} - 2\mathbf{x}_{\mathrm{c},i+1} + \mathbf{x}_{\mathrm{c},i}}{\Delta t_b^2} \tag{16}$$

그리고 미분을 통해 목표하는 값을 구한다.

이를 통해 안정적인 비행이 가능하며, 탐색이 가능하게 됨을 실험을 통해 확인할 수 있었다.