MT Robot Reading Group

Real-Time Trajectory Replanning for MAVs using Uniform B-splines and a 3D Circular Buffer

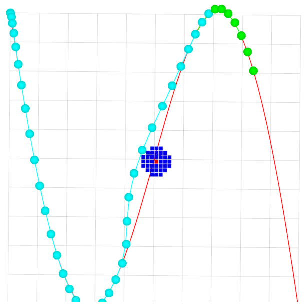
by Vladyslav Usenko, Lukas Von Stumberg, Andrej Pangercic and Daniel Cremers

IROS 2017, Best Paper Finalist

Agenda

Introduction

- Most system assume a static environment
- For an agent in a dynamic environment
  - Two layer approach:
    - Global planner: Plan trajectory with static obstacles
    - Reactive planner: Considers local information



Related Work

- Trajectory generation
- Map representation

Related Work

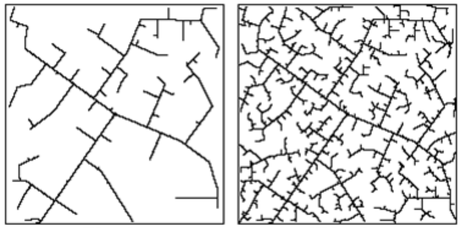
Trajectory generation:

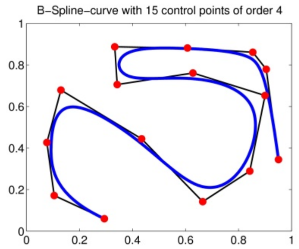
- Search-based
- Optimization-based

Related Work

Trajectory generation:

- Search-based:
  - Non-smooth path planned over a graph e.g. from RRT
  - Polynomial or B-spline computed (smooth)





Related Work

Trajectory generation:

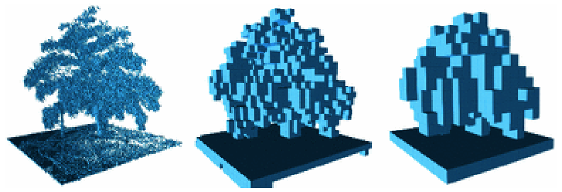
- Optimization-based:
    - Minimize a cost function (smoothness & collision)

      $\arg\min_x \lambda_s \cdot E_s(x) + \lambda_c \cdot E_c(x)$

Related Work

Map representation

- Needed to plan collision-free trajectories
    - Information about occupancy required
- Voxel grid: Simplest and most used solution


[Hornung AR 2013]

Approach

Uniform B-splines (of degree $k - 1$)

$$p(t) = \sum_{i=0}^{n} p_i B_{i,k}(t)$$

where $p_i \in \mathbb{R}^n$ are control points at times $t_i, i \in [0, \dots n]$

and $B_{i,k}(t)$ are basis functions

Uniform B-splines have a fixed time interval Δt

Approach

Uniform B-splines

Advantages (compared to polynomial-splines)

- Faster optimization (fewer variables and constraints)

Disadvantages (compared to polynomial-splines)

- Trajectory does not pass through the control points
  - For local replanning not very important

Approach

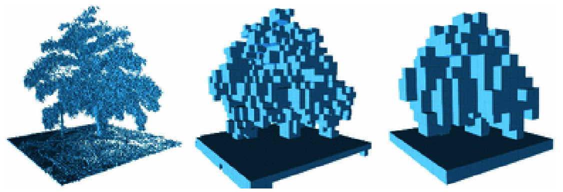
Local Environment Map

- For obstacle avoidance
- Maintain an occupancy model of the environment
- Most recent sensor measurements & some past measurements

Approach

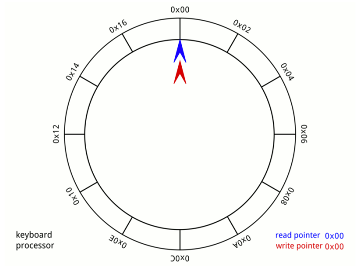
3D Circular Buffer

- Discretize the volume into voxels



Approach

3D Circular Buffer



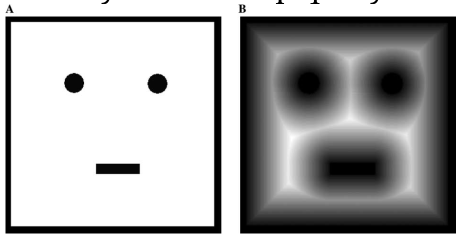
- Check if a voxel is in the buffer:
  $\text{insideVolume}(x) = 0 \leq x - o < N$, where o is the offset
- Find address: $\text{address}(x) = (x - o) \bmod N$

Approach

Local Environment Map

Distance map computation

- Compute the Euclidean distance transform (EDT)
- For fast collision checking
- Checking for collision by one look-up query



Approach

Trajectory Optimization

The local replanning problem represented as an optimization

$$E_{\text{total}} = E_{\text{ep}} + E_c + E_q + E_l$$

with

E_{ep} an endpoint cost function

E_c a collision cost function

E_q cost of the integral over the squared derivatives

E_l soft limit on the norm of time derivatives

Approach

Trajectory Optimization

E_{ep} an endpoint cost function:

- Keep the local trajectory close to the global one
  $E_{\text{ep}} = \lambda_p(p(t_{ep}) - p_{ep})^2 + \lambda_v(p^{'}(t_{ep}) - p^{'}_{ep})^2$

E_c a collision cost function:

- Penalizes the trajectory point that are close to obstacles

Approach

Trajectory Optimization

E_q cost of the integral over the squared derivatives:

- For smoothness

E_l soft limit on the norm of time derivatives:

- To ensure that velocity, acceleration and higher derivatives of position remain bounded.

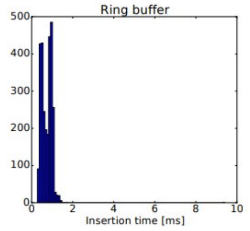
Approach

Implementation

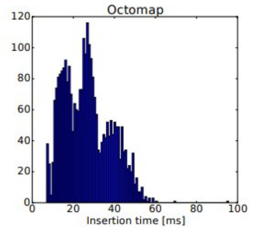
- First a global trajectory is computed
- In every iteration the endpoint constraints are set to the position and velocity of the global trajectory at t_{ep}
- The collision cost is evaluated using the circular buffer

Results

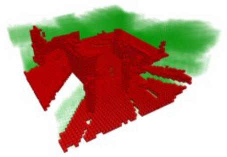
Circular Buffer Performance

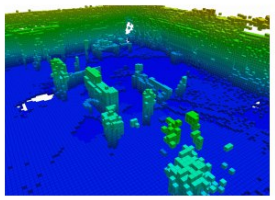


(a)



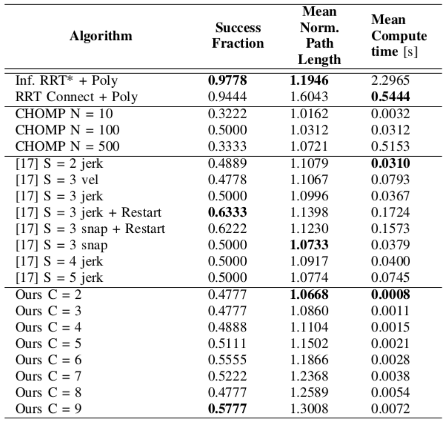
(b)


(c)

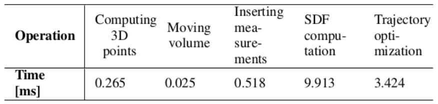

(d)

Results

Optimization performance & System simulation



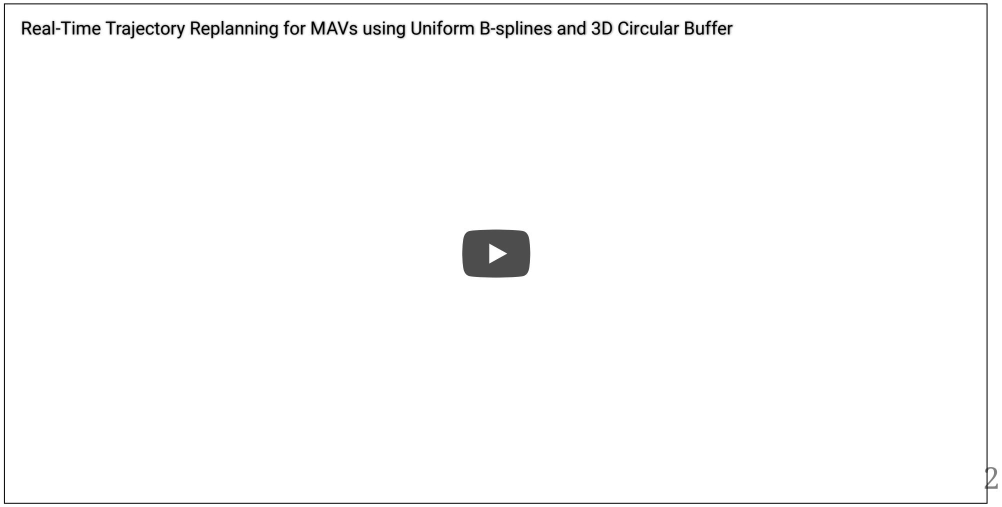
Algorithm	Success Fraction	Mean Norm. Path Length	Mean Compute time [s]
Inf. RRT* + Poly	**0.9778**	**1.1946**	2.2965
RRT Connect + Poly	0.9444	1.6043	**0.5444**
CHOMP N = 10	0.3222	1.0162	0.0032
CHOMP N = 100	0.5000	1.0312	0.0312
CHOMP N = 500	0.3333	1.0721	0.5153
[17] S = 2 jerk	0.4889	1.1079	**0.0310**
[17] S = 3 vel	0.4778	1.1067	0.0793
[17] S = 3 jerk	0.5000	1.0996	0.0367
[17] S = 3 jerk + Restart	**0.6333**	1.1398	0.1724
[17] S = 3 snap + Restart	0.6222	1.1230	0.1573
[17] S = 3 snap	0.5000	**1.0733**	0.0379
[17] S = 4 jerk	0.5000	1.0917	0.0400
[17] S = 5 jerk	0.5000	1.0774	0.0745
Ours C = 2	0.4777	**1.0668**	**0.0008**
Ours C = 3	0.4777	1.0860	0.0011
Ours C = 4	0.4888	1.1104	0.0015
Ours C = 5	0.5111	1.1502	0.0021
Ours C = 6	0.5555	1.1866	0.0028
Ours C = 7	0.5222	1.2368	0.0038
Ours C = 8	0.4777	1.2589	0.0054
Ours C = 9	**0.5777**	1.3008	0.0072



Operation	Computing 3D points	Moving volume	Inserting mea-sure-ments	SDF compu-tation	Trajectory opti-mization
Time [ms]	0.265	0.025	0.518	9.913	3.424

TABLE II: Mean computation time for operations involved in trajectory replanning in the simulation experiment with depth map measurements sub-sampled to 160×120 and seven control points optimized.

Results


Real-Time Trajectory Replanning for MAVs using Uniform B-splines and 3D Circular Buffer

Something for us?

Differences:

- 2D sensor & map
- Car like robot (2D)

Similarities:

- Need for replanning (dynamic environment)
- Sort of local map already in use

Discussion