2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 27 - October 1, 2021. Prague, Czech Republic

2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-1714-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IROS51168.2021.9636002

Fast Generation of Obstacle-Avoiding Motion Primitives for Quadrotors*

Saurabh Upadhyay[1], Thomas Richardson[2], and Arthur Richards[2]

***Abstract*— This work considers the problem of generating computationally efficient quadrotor motion primitives between a given pose (position, velocity, and acceleration) and a goal plane in the presence of obstacles. A new motion primitive tool based on the logistic curve is proposed and a closed-form analytic approach is developed to satisfy constraints on starting pose, goal plane, velocity, acceleration, and jerk. The geometric obstacle avoidance problem is represented as a combinatorial set problem and a heuristic approach is proposed to accelerate the solution search. Numerical examples are presented to highlight the fast motion primitive generation in multi-obstacle pose-to-plane scenarios.**

I. INTRODUCTION

Motion primitive based motion planning approaches are widely used in UGV and UAV applications [1]–[6]. The reasons behind their popularity are as follows: (i) they discretize the configuration space allowing the graph search methods to be used directly [4], and (ii) avoid the path smoothing step of the sampling-based motion planners for generating kinodynamically feasible trajectories [5]. Focusing on the kinodynamic motion planning, a prospective motion primitive tool should provide multiple trajectory segments originated from a prescribed starting pose (position, velocity, and acceleration) that have continuous velocity and acceleration variations with bounded values. It is also desirable that the motion primitives have obstacle avoidance properties.

One of the first works on motion primitives based motion planning was presented in [7] where an approximate algorithm using acceleration bang primitives was proposed. In [8], an exact algorithm for planar workspace was discussed where position and velocity were time parameterized to obtain the motion primitives. Note that one of the important reasons behind the computation complexity of [7] and [8] was the absence of simple geometric conditions for obstacle avoidance [6]. Using a state-time space representation approach, motion planning in presence of dynamic obstacles was solved in [9]. This approach mapped the planning problem to a simple curve finding problem in the state-time space where piece-wise constant acceleration motion primitives were used to discretize the space in a grid and applying the graph search algorithms directly to find the shortest path. Reducing the computational complexity of the state-space time approach, a sampling-based motion planner was proposed in [10] which randomly sampled the grid points rather than considering all points, for finding the path.

A state lattice based approach was proposed in [11] to ensure the smoothness of the trajectory where motion primitives were represented by polynomial curvature function of path length. The position expressions of these primitives were given by polynomial spirals (also known as Cornu spirals) [12] with few design parameters that reduce the search space dimension, however, require numerical computation or approximation of Fresnel integrals for computing the position. Polynomial time parameterized motion primitives were used with state lattice approach in [4] to provide inherent smoothness of the trajectory, and simple bounded velocity and acceleration conditions. Therein, a constant control was applied for a time period and obstacle avoidance was checked over an occupancy grid map with conservative conditions. In [13], quintic Bézier curve based motion primitives were proposed and integrated with the state lattice based approach to obtain the smooth trajectories with fewer design parameters and computation efficiency. Therein, a signed distance map was used to compute the distance from the obstacle, and avoidance was ensured by making the distance of a path segment less than the corresponding signed distance.

A polynomial time parameterized motion primitive approach with flexible end-point constraints and closed-form solutions was discussed in [14]. The free-end framework provided feasibility checking and trajectory generation with computational efficiency. Recently, a four parameter logistic (4PL) curve based path planning tool was proposed in [15] and applied to a UAV application with geometric obstacle avoidance conditions [16]. Explicit position coordinate expression of the 4PL curve with just two design parameters, shape flexibility, and asymptotic final point reachability makes it a computationally efficient path planning tool.

Inspired by [14], this work defines the problem of generating smooth motion primitive in a pose-to-plane planning framework where the starting pose (position, velocity, and acceleration) and goal plane position are given. As the key contribution, a new motion primitive tool is derived from the 4PL curve and investigated as a prospective solution. Closed-form conditions are derived to satisfy the starting pose, goal plane, and dynamic feasibility (smooth and bounded velocity, acceleration, and jerk variations) constraints. The obstacle avoidance problem is solved by selecting appropriate com-

*This work is funded by the Engineering and Physical Sciences Research Council CASCADE Programme (Ref. EP/R009953/1).

[1]Saurabh Upadhyay is with the Department of Aerospace Engineering, University of Bristol, Bristol, United Kingdom.s.upadhyay@bristol.ac.uk

[2]Thomas Richardson and Arthur Richards are with the Department of Aerospace Engineering, University of Bristol, Bristol, United Kingdom and Bristol Robotics Laboratory, Bristol, United Kingdom. thomas.richardson@bristol.ac.uk, arthur.richards@bristol.ac.uk

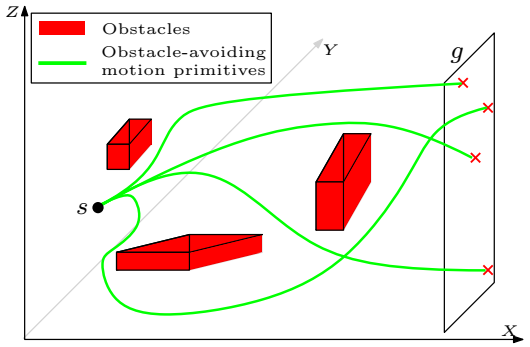

Fig. 1: Pose-to-plane motion primitives in the obstacle cluttered workspace

binations of 1−D trajectories such that the set of intersection times is empty. A heuristic approach for finding all suitable combinations is proposed. Computational performance of the proposed approach is tested in simple and complex multi-obstacle scenarios.

The remainder of this paper is organized as follows: The concerned problem is defined in Section II. A logistic curve based motion primitive tool satisfying pose-to-plane and dynamic feasibility constraints is proposed in Section III. Section IV proposes an obstacle avoidance approach. The proposed method is validated with numerical examples in Section V. Section VI provides the concluding remarks.

II. PROBLEM DEFINITION

A. Relationship between Quadrotor Motion and Time-Parameterized Curves

As discussed in [17], a quadrotor position $X=(x,y,z)\in\mathbb{R}^3$ can be defined as differential flat outputs and hence, each position component can be represented by a thrice differentiable time-parameterized curves. Using the quadrotor dynamics, expressions for the quadrotor control inputs in the inertial frame can be written as [18]

$$f=||\ddot{X}-G||,\quad \begin{bmatrix}\omega_y\\-\omega_x\end{bmatrix}=\frac{1}{f}\begin{bmatrix}1&0&0\\0&1&0\end{bmatrix}R^{-1}\dddot{X} \tag{1}$$

where f and (ω_x,ω_y) are the mass normalized force and component-wise angular velocities, respectively. The proper orthogonal matrix R converts a body frame vector to an inertial frame vector and $G=[0,0,-g]^T$ where g represents the gravitational acceleration. As the yaw motion is independent of the translation motion, $\omega_z=0$ (constant yaw ψ) is assumed in this work. Here, $\ddot{X}$ and $\dddot{X}$ are the second (acceleration) and third derivatives (jerk) of the quadrotor center of mass position X with respect to time t, respectively. With this, the derivatives of all three curves (velocity $V=\dot{X}$, acceleration $A=\ddot{X}$, and jerk $J=\dddot{X}$) can be used in (1) to obtain the quadrotor control inputs. Also, the bounds on the inputs (force and angular velocities) can be considered in terms of the trajectory acceleration and jerk bounds [6].

B. Problem Statement

Consider a motion planning workspace $W\subseteq\mathbb{R}^3$ with a given starting position $s=(x_s,y_s,z_s)$ and a goal plane g as shown in Fig. 1. Without loss of generality, the goal plane can be defined as $g:=\{(x_g,y_g,z_g)|y_g\in[\underline{y_g},\bar{y_g}],z_g\in[\underline{z_g},\bar{z_g}]\}\subseteq\mathbb{R}^2$ (here $Y-Z$ plane) where $x_g,\underline{y_g},\bar{y_g},\underline{z_g}$, and $\bar{z_g}$ are predefined constants. Velocity, acceleration and jerk of the 3-D motion primitives are represented by vectors $V=[v_x,v_y,v_z]$ and $A=[a_x,a_y,a_z]$, and $J=[j_x,j_y,j_z]$, respectively. Starting time t_s, velocity $V_s=[v_{sx},v_{sy},v_{sz}]$ and acceleration components $A_s=[a_{sx},a_{sy},a_{sz}]$ at starting position are predefined. Also, the upper and lower bounds on the velocity $(\underline{v},\bar{v})$, acceleration $(\underline{a},\bar{a})$, and jerk $(\underline{j},\bar{j})$ components are given. The workspace contains n known static rectangular shaped obstacles $\{R_k\}_{k=1}^n$ defined as $R_k:=[\underline{x_k},\bar{x_k}]\times[\underline{y_k},\bar{y_k}]\times[\underline{z_k},\bar{z_k}]$ where $(\underline{x_k},\underline{y_k},\underline{z_k})$ and $(\bar{x_k},\bar{y_k},\bar{z_k})$ represent the extreme corners of the k obstacle. With this, the problem can be formally stated as follows: Find a motion primitive $X(t):=\{t\rightarrow(x(t),y(t),z(t))\}\in\mathbb{R}^3$ that

1) connects the starting position and the goal plane, that is, $X(t_s)=(x_s,y_s,z_s)$ and $X(t_g)\in g$ where, t_g is the goal time, respectively.
2) has desired velocity and acceleration values, respectively, at the starting position, that is, $\dot{X}(t_s)=V_s$ and $\ddot{X}(t_s)=A_s$,
3) has continuous velocity and acceleration variations within the prescribed bounds, that is, $\dot{X}(t)\in[\underline{V},\bar{V}]$ and $\ddot{X}(t)\in[\underline{A},\bar{A}]$, respectively, for all $t\in[t_s,t_g]$, and
4) avoids all obstacles, that is, $X(t)\notin\bigcup_k R_k$ for all t.

This work assumes starting pose with strictly non-zero acceleration and velocity values to avoid degeneracy in the closed-form solution of the initial conditions. A variation on the provided equations can relax this requirement somewhat but is beyond the present scope.

III. PROPOSED MOTION PRIMITIVE TOOL

First, the modified 4PL curve is introduced and its basic properties are discussed. Closed-form conditions are derived for generating a pose-to-plane 4PL motion primitive. Dynamic feasibility (bounded velocity, acceleration, and jerk) analysis is presented subsequently.

A. The Modified 4PL Curve with Time Scaling and Shifting Parameters

For generating the motion primitive with the prescribed starting constraints, the generic 4PL curve is defined as

$$p(t;t_s,t_d,B,C,p_i,p_g)=p_g+\frac{p_i-p_g}{1+\left(\frac{t-t_s+t_d}{C}\right)^{2B}} \tag{2}$$

where $t\geq 0,\ t_d\in\mathbb{R},\ B>2,\ C\in\mathbb{R},p_i<p_g$, and $t_s\geq 0$ is fixed. The design variables p_i and p_g are the extrema of the curve which are to be sampled. Design parameters (B,C) and t_d are obtained from the sampled (p_i,p_g) values and starting pose which provide shape control and time scaling, respectively, as shown in Fig. 2. From (2), the 3-D 4PL motion primitive components are written as

$$x(t)=p(t;t_s,t_{dx},B_x,C_x,x_i,x_g) \tag{3}$$
$$y(t)=p(t;t_s,t_{dy},B_y,C_y,y_i,y_g) \tag{4}$$
$$z(t)=p(t;t_s,t_{dz},B_z,C_z,z_i,z_g) \tag{5}$$

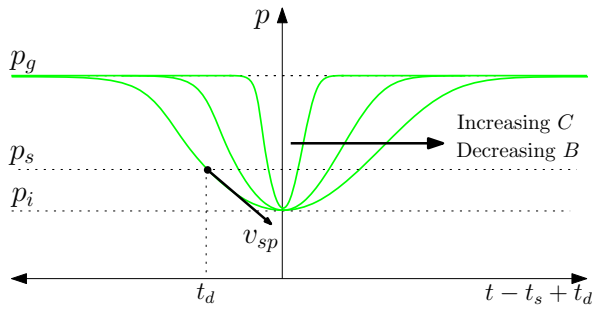

Fig. 2: Curve shape control by (B,C) and shifting by t_d

where, $t \in [t_s, t_g]$.

From (2) it can be seen that the 4PL curve attains the goal value p_g asymptotically which is not desirable for motion planning. The final point reachability property [15] of the 4PL curve can be used to obtain a final time t_g at which the 4PL primitive lies within an acceptable $\varepsilon > 0$ error band of the goal position. Imposing $|p_g - p(t_g)| \leq \varepsilon$, results in

$$t_g \geq t_s - t_d + |C| \left(\frac{|p_i - p_g| - \varepsilon}{\varepsilon} \right)^{\frac{1}{2B}} \tag{6}$$

Expressions for first, second, and third time derivatives (velocity, acceleration, and jerk, respectively) of the 4PL motion primitive components are obtained as

$$v_p = \frac{dp}{dt} = \frac{2BC^{2B}(p_g - p_i)(t - t_s + t_d)^{2B-1}}{\left((t - t_s + t_d)^{2B} + C^{2B}\right)^2} \tag{7}$$

$$a_p = \frac{d^2p}{dt^2} = \frac{2BC^{2B}(p_g - p_i)(t - t_s + t_d)^{2B-2}}{\left((t - t_s + t_d)^{2B} + C^{2B}\right)^2} \left[(2B-1) - \frac{4B(t - t_s + t_d)^{2B}}{(t - t_s + t_d)^{2B} + C^{2B}} \right] \tag{8}$$

$$j_p = \frac{d^3p}{dt^3} = \frac{2BC^{2B}(p_g - p_i)\,(t - t_s + t_d)^{2B-3}}{\left((t - t_s + t_d)^{2B} + C^{2B}\right)^2} \left[(2B-1)(2B-2) - \frac{12B(2B-1)(t - t_s + t_d)^{2B}}{(t - t_s + t_d)^{2B} + C^{2B}} \frac{24B^2(t - t_s + t_d)^{4B}}{\left((t - t_s + t_d)^{2B} + C^{2B}\right)^2} \right] \tag{9}$$

respectively, which have zero values at final time t_g, for all $B > 2$. Also, from (7), (8), and (9), it can be observed that the 4PL primitive has continuous velocity, acceleration, and jerk variation with respect to t for $B > 2$.

B. Proposed Motion Primitive Satisfying Given Starting Constraints

From the given starting position (p_s), velocity (v_{sp}), and acceleration (a_{sp}), the explicit expressions of the design parameters are obtained as follows:

1) Computation of t_d from the given starting position: Imposing $p = p_s$ at $t = t_s$ in (2), results in

$$t_d = \pm C \left(\frac{p_s - p_i}{p_g - p_s} \right)^{\frac{1}{2B}} \tag{10}$$

Figure 2 shows that any $p_s \in [p_i, p_g]$ can be achieved by shifting the curve with a t_d computed from (10).

2) Expression for C using v_s: Similarly, imposing starting velocity constraint in (7) and substituting t_d from (10), yields

$$C = \frac{2B(p_g - p_s)(p_s - p_i)}{v_{sp}\,(p_g - p_i)} \left(\frac{p_g - p_s}{p_s - p_i} \right)^{\frac{1}{2B}} \tag{11}$$

3) Parameter B from acceleration: Using the starting acceleration value in (8), and substituting t_d and C from (10) and (11), respectively, results in

$$B = \frac{1}{2} \frac{v_{sp}^2 (p_g - p_i)}{v_{sp}^2 (p_g - 2p_s + p_i) - a_{sp}(p_g - p_s)(p_s - p_i)} \tag{12}$$

For the given starting pose (p_s, v_{ps}, a_{ps}) and chosen (p_g, p_i), the parameters B, C, and t_d can be determined in sequence from (12), (11), and (10), respectively.

Remark 1: From (11) and (12), it can be seen that zero starting velocity or acceleration leads to a singularity in the 4PL curve where $p_i = p_g = p_s$. In this case, the parameters B and C become the design variables and all analysis need to be carried out in terms of (B,C) which is out of the scope of this paper.

Remark 2: For $p_i = p_s = p_g$, the 4PL curve has a constant position p_g and zero velocity, acceleration, jerk values for all $t \in [t_s, t_g]$.

C. Conditions Satisfying Velocity, Acceleration, and Jerk Bounds

By equating (8), (9), and the first derivative of (9) to zero and solving for t gives the time instants corresponding to velocity, acceleration, and jerk extrema $(v^\star, a^\star, j^\star)$, respectively. These extrema can be computed by substituting respective time instants in (7), (8), and (9), respectively which are trivial expressions and omitted here for the sake of brevity. With this, the conditions for bounded velocity, acceleration, and jerk are written as

$$\underline{v} \leq v^\star \leq \bar{v} \tag{13}$$

$$\underline{a} \leq a^\star \leq \bar{a} \tag{14}$$

$$\underline{j} \leq j^\star \leq \bar{j} \tag{15}$$

D. Selection of Dynamically feasible Sets

Steps for computing a dynamically feasible set (S_f) of (p_i, p_g) pairs that satisfy the starting pose, goal plane, and dynamic feasible constraints, are discussed in this subsection. First, (p_i, p_g) pairs are sampled from $p_i \in (-\infty, p_g)$ and $p_g \in [\underline{p}_g, \bar{p}_g]$. Using the sampled pairs, given starting pose, and goal plane position, design parameters B,C, and t_d are computed in order from (12), (11), and (10), respectively. Next, the dynamic feasibility conditions, given by (13), (14), and (15) are checked for computed parameters and all feasible pairs are stored in the set S_f. Feasible design variable sets (S_f^x, S_f^y, S_f^z) for the 3-D motion primitive are obtained in each axis independently.

Remark 3: Sampling step (grid) size of the design variable space affects the performance of the proposed approach in terms of number which is related to the resolution completeness property. Hence, there is a trade-off between the number of solutions and computations.

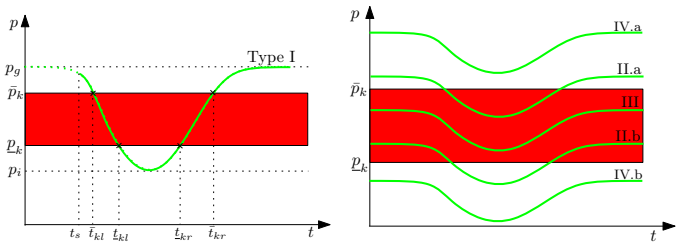

Fig. 3: Time intersection between an obstacle projection and 1-D 4PL primitive: Four possible intersections (cross markers) for a Type I 1-D 4PL primitive (a) and the remaining three possible primitive types: II) two intersection points (II.a and II.b), III) complete overlap, and IV) no intersection (b).

TABLE I: Intersection intervals between the 4PL primitive and the obstacle projection and respective flag

Type	(p_i, p_g) position	Starting time	Time intervals t_{ovl}	Flag (f)
I	$p_i < \underline{p}_k < \bar{p}_k < p_g$	$t_s < \bar{t}_{kl}$	$[\bar{t}_{kl}, \underline{t}_{kl}]$, $[\underline{t}_{kr}, \bar{t}_{kr}]$	2
		$\bar{t}_{kl} < t_s \leq \underline{t}_{kl}$	$[t_s, \underline{t}_{kl}]$, $[\underline{t}_{kr}, \bar{t}_{kr}]$	2
		$\underline{t}_{kl} < t_s \leq \underline{t}_{kr}$	$[\underline{t}_{kr}, \bar{t}_{kr}]$	2
		$\underline{t}_{kr} < t_s \leq \underline{t}_{kr}$	$[t_s, t_{kr}]$	2
		$\bar{t}_{kr} < t_s$	$\emptyset$	0
II.a	$\underline{p}_k \leq p_i \leq \bar{p}_k < p_g$	$t_s \leq \bar{t}_{kl}$	$[\bar{t}_{kl}, \bar{t}_{kr}]$	2
		$\bar{t}_{kl} < t_s \leq \bar{t}_{kr}$	$[t_s, \bar{t}_{kr}]$	2
		$\bar{t}_{kr} < t_s$	$\emptyset$	0
II.b	$p_i < \underline{p}_k \leq p_g \leq \bar{p}_k$	$t_s \leq \underline{t}_{kl}$	$[t_s, \underline{t}_{kl}]$, $[\underline{t}_{kr}, \infty)$	2
		$\underline{t}_{kl} < t_s \leq \underline{t}_{kr}$	$[\underline{t}_{kr}, \infty)$	2
		$\underline{t}_{kr} < t_s$	$[t_s, \infty)$	1
III	$\underline{p}_k \leq p_i < p_g \leq \bar{p}_k$	all t_s	$[t_s, \infty)$	1
IV.a	$p_i < p_g < \underline{p}_k < \bar{p}_k$	all t_s	$\emptyset$	0
IV.b	$\underline{p}_k < \bar{p}_k < p_i < p_g$	all t_s	$\emptyset$	0

IV. Obstacle Avoiding Motion Primitives

This section discusses the generation of 4PL motion primitives with obstacle avoidance. Avoidance of multiple obstacles can be guaranteed by avoiding their projections in at least one axis for all time. This is achieved in the following two steps:

1) Find the time intervals during which a 4PL motion primitive in one axis overlaps with the projection of an obstacle into the same axis. These time intervals are computed for all dynamic feasible pairs of each axis.
2) Find all combinations of $1-$D motion primitives such that the time intervals of overlap with each obstacle do not all intersect, i.e. there is no time when the $3-$D trajectory is inside the obstacle. Brute force evaluation of all combinations of $1-$D primitives is the baseline approach. Here, we present heuristics to accelerate that search by exploiting special overlap cases.

Details of the steps are discussed in the sequel.

A. Overlap Between an Obstacle Projection and a 4PL Motion Primitive Component

Consider a static obstacle projection in the $t-p$ (where p could be x, y, or z) plane, as shown in Fig. 3 by the infinite red rectangular strip with width $\bar{p}_k - \underline{p}_k$, where $\bar{p}_k > \underline{p}_k$. For simplicity of notation, we assumed $p_i < p_g$ for obtaining the time intervals. The p_i and p_g are swapped in the obtained time intervals for $p_g < p_i$. The overlap between the curve and obstacle is formally defined as $T_{ovl} := \{t | p(t) \in [\underline{p}_k, \bar{p}_k]\}$. Based on the (p_i, p_g) position, four possible types of the 4PL curve exist that are defined in Table I and discussed as follows:

1) Type I (Four intersection points): In the first case, the curve intersects the edges of the obstacle projection at four points as shown in Fig. 3a by the cross markers. By equating (2) to $\bar{p}_k$, the time instants for the left and right upper intersection points can be obtained as

$$\bar{t}_{kl} = t_s - t_d - C\left(\frac{\bar{p}_k - p_i}{p_g - \bar{p}_k}\right)^{\frac{1}{2B}}; \bar{t}_{kl} = t_s - t_d + C\left(\frac{\bar{p}_k - p_i}{p_g - \bar{p}_k}\right)^{\frac{1}{2B}} \tag{16}$$

respectively. Similarly, the time instants corresponding to two lower intersection points are given as

$$\underline{t}_{kl} = t_s - t_d - C\left(\frac{\underline{p}_k - p_i}{p_g - \underline{p}_k}\right)^{\frac{1}{2B}}, \underline{t}_{kr} = t_s - t_d + C\left(\frac{\underline{x}_k - p_i}{p_g - \underline{x}_k}\right)^{\frac{1}{2B}} \tag{17}$$

From (16) and (17), the time intervals $[\bar{t}_{kl}, \underline{t}_{kl}]$ and $[\underline{t}_{kr}, \bar{t}_{kr}]$ correspond to the left and right curve-obstacle intersections.

2) Type II (Two intersection points): The second type of the 4PL curve intersects the edges of obstacle projection at two points in such a way that either p_i (Type II.a) or p_g (Type II.b) lies within the obstacle projection. These two subtypes are labeled as II.a and II.b in Fig. 3b, respectively. As the $p_i \in [\underline{p}_k, \bar{p}_k]$ for Type II.a curve, it has only one overlapping time interval $[\bar{t}_{kl}, \bar{t}_{kr}]$. In contrast, the Type II.b curve has two time intervals $[\infty, \underline{t}_{kl}]$, $[\underline{t}_{kr}, \infty]$.

3) Type III (Complete overlap): This type of curve lies within the obstacle projection for all time as shown in Fig. 3b by the curve labeled *III*. Hence, the time interval span over the whole time period $(-\infty, \infty)$.

4) Type IV (No intersection): Any curve either above or below the obstacle projection (IV.a and IV.b label curves, respectively, in Fig. 3b), falls under this category. For this type of curve, the overlapping interval is empty.

Effect of t_s on the time intervals: The 1-D 4PL motion primitive is a part of the 4PL curve that starts from a specific t_s. Hence, any time intersection point less than the t_s should be replaced with t_s in the overlapping time intervals. With this, the intersecting time intervals for all types are summarized in Table I which will be used in the next section for obstacle avoidance. To support the subsequent search, we define an overlap type flag as an output of this process: 0 for no overlap, 1 for continuous overlap, and 2 for partial overlap, i.e. for a subset of $[t_s, \infty)$. The time intervals and intersection information for each dynamically feasible pairs $(p_i, p_g) \in S_f$ are stored in matrices $T_{(p_i,p_g)}$ and $F_{(p_i,p_g)}$ defined as $T_{(p_i,p_g)} := \{\{t_{int}\}\}_{k=1}^{n}$ and $F_{(p_i,p_g)} := \{f_k\}_{k=1}^{n}$ respectively, which are further stored in $T := \{(p_i,p_g)_j\}_{j=1}^{|S_f|}$ and $F := \{F_{(p_i,p_g)_j}\}_{j=1}^{|S_f|}$ for all feasible pairs. Using this process, we

can calculate T_x, T_y, and T_z arrays, where $T_x(j,k)$ denotes the overlap interval T_ovl between obstacle k and $1-$D $x-$axis primitive j from S_x^f, and so on. Similarly, define matrices F_x, F_y and F_z such that $F_x(j,k)$ is the overlap type flag for $x-$primitive j from S_f^x with respect to obstacle k.

B. Heuristic-Based Obstacle Avoidance Approach

The steps of the proposed obstacle avoidance approach are discussed as follows:

1) A motion primitive component not intersecting any obstacle edge avoids all obstacles. Hence, feasible design pairs not intersecting all obstacles in one axis and their combinations with all feasible pairs of other two axis components are stored as solutions. These pairs are removed from the feasible sets subsequently.
2) Based on the number of design pair, dynamic feasible sets (S_f^x, S_f^y, and S_f^z) are selected in the ascending order which reduces the number of obstacle avoidance checks. Using the intersection type information sets (F), each pair of the first selected feasible set is checked against all feasible pairs of the second feasible set. Three possible pair combinations are as follows:
   a) A feasible pair combination avoiding every obstacle in both or either axis (flag combinations: (0,1 or 2),(1 or 2,0), (0,0)) is combined with the all feasible pairs of third feasible set and stored as solutions.
   b) If both axis components intersect one or more obstacles then the intersection of respective time intervals is checked. All pair combinations with no intersections for all obstacles are combined with all feasible pairs of the third component and stored as solutions.
   c) Remaining pair combinations are checked one by one against all pairs of third trajectory components. All pair combinations avoiding each obstacle in at least one axis are stored as solutions. Time intersection is checked for a pair combination with two or more components intersecting one or more obstacles. All pair combinations with no intersection for all obstacles are stored as solutions.

Using (3)-(5), (10)-(12), and all design variable sets of s, multiple dynamically feasible obstacle-avoiding motion primitives can be generated in one shot. The final time t_g for a motion primitive is given as $t_g = \max\{t_g^x, t_g^y, t_g^z\}$ where t_g^x, t_g^y, t_g^z are computed from (6).

V. NUMERICAL EXAMPLES

Two cases are considered to test the performance of the proposed approach in terms of the number of solutions and computation time. For both cases, start and goal positions are selected at $s(x_s, y_s, y_g) = (0, 1.5, 1)$ m, $x_g = 4$ m, $(\underline{y}_g, \bar{y}_g) = (2.8, 3.2)$ m, and $(\underline{z}_g, \bar{z}_g) = (1.7, 2.3)$ m. The velocity, acceleration, and jerk bounds of $(\underline{v}, \bar{v}) = (-5, 5)$ m/s, $(\underline{a}, \bar{a}) = (-10, 10)$ m/s^2 , and $(\underline{j}, \bar{j}) = (-50, 50)$ m/s^3,

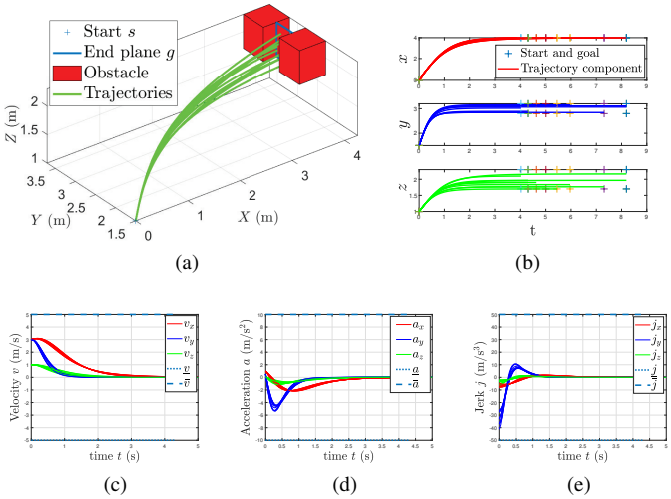

Fig. 4: Case 1: Proposed pose-to-plane motion primitive generation (a), primitive components (b), and continuous bounded derivative profiles (c)-(e).

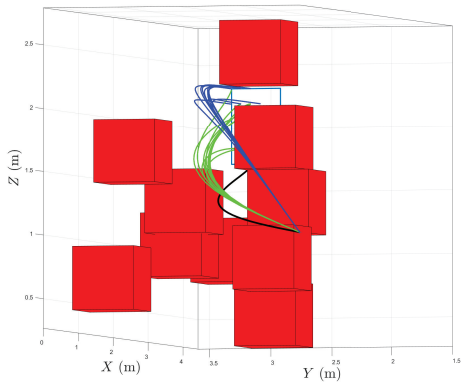

Fig. 5: Multiple obstacle avoidance with three starting poses: Case 2a (green), Case 2b (blue), Case 2c (black).

are considered for each axis component, respectively. The proposed approach is implemented in MATLAB R2019b on a 3.20 GHz Intel® Core™ i7-8770 CPU with 16 GB RAM.

1) Case 1: Trivial Obstacle Avoidance Scenario: Two obstacles are placed at both sides of the goal plane (red rectangular boxes in Fig. 4a) in such a way that all dynamically feasible pairs avoid both obstacles. At the starting point, velocity $V_s = [3,3,1]$ and acceleration $A_s = [1, 0.5, 0.1]$ are considered. The (p_i, p_g) values are sampled over 6×6 grid. With these inputs, the proposed approach has computed 5, 25, and 32 dynamically feasible sets for x, y, and $z-$axis components, respectively, in 101 ms. As given in Table II, a brute-force obstacle avoidance approach takes 0.312 s to generate 2720 solutions whereas the proposed approach computes all solutions in only 172 ms. Feasible motion primitives are generated for ten solutions which are plotted in Fig. 4a by the solid green lines. Figure 4b shows that all motion primitive components satisfy the starting and goal position constraints. Velocity and acceleration profiles of the motion primitive components are plotted in Figs.

TABLE II: Computation data for the numerical examples

Case	Starting velocity and acceleration	Dynamically feasible pairs (x,y,z) and computation time	First set of solutions and time	Total solutions	Planning time	
					Brute-force	Proposed
1	$V_s=[3,3,1]$, $A_s=[1,0.5,0.1]$	(5,25,32), 0.101 s.	32 Solutions, 0.116 s	2720	0.312 s	0.172 s
2a	$V_s=[3,3,1]$, $A_s=[1,0.5,0.1]$	(5,25,32), 0.111 s	1 solution, 0.152 s	1636	0.657 s	0.470 s
2b	$V_s=[3,3,-2]$, $A_s=[1,0.5,0.1]$	(5,25,5), 0.101 s	1 solution, 0.117 s	493	0.343 s	0.195 s
2c	$V_s=[3,2.8,0.5]$, $A_s=[0.5,0.03,0.1]$	(5,29,32), 0.116 s	1 solution, 0.210 s	1	0.704 s	0.444 s

4c and 4d, respectively, that starts from the given velocity and acceleration values and have continuous and bounded variations. Figure 4e shows that the jerk variations for all motion primitives lie within the prescribed bounds $(\underline{j},\bar{j})$.

2) Case 2: Multi-Obstacle Scenario: This case considers 10 randomly generated obstacles with 0.5 m width in each axis as shown in Fig. 5. Three subcases with different starting poses are considered for which the starting pose values are provided in Table II. The first subcase (Case 2a) considers the starting pose of Case 1 for which the proposed approach computes 1636 solutions in 470 ms. Ten solutions are selected from the solution set and corresponding obstacle-avoiding motion primitives are plotted in Fig. 5 by the solid green lines. For Case 2b and 2c, the starting velocity and acceleration in $z-$axis are changed, respectively, and as given in Table II. The proposed approach has computed 493 and 1 solution in 195 and 444 ms, respectively. Motion primitives for Case 2b and 2c are plotted in Fig. 5 by the blue and purple lines, that avoid all obstacles and satisfy the pose-to-plane constraints.

Time to compute the first set of solutions is provided in Table II for all cases. It shows that the proposed approach can compute the first set of solutions in approximately half time of the total planning time. Also, for all cases, the proposed approach finds solutions faster than the brute-force approach.

VI. CONCLUSIONS

A new logistic curve based motion primitive tool is introduced to connect a given starting pose and a goal plane. The starting pose, goal plane, and dynamic feasibility constraints are imposed and closed-form conditions are derived to obtain a set of feasible design variable pairs in each axis separately. Potential obstacle overlap intervals are also found analytically, axis by axis. The final stage is to identify combinations of axis primitives whose overlap intervals do not intersect and hence ensure obstacle avoidance. A heuristic approach is proposed to find all solutions with a fewer number of intersection checks. Numerical examples with multiple obstacle scenarios demonstrate that the proposed approach is capable of generating a large number of obstacle-avoiding motion primitives in a fraction of a second.

REFERENCES

[1] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A Review of Motion Planning Techniques for Automated Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, Apr. 2016.

[2] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[3] D. Ferguson, T. M. Howard, and M. Likhachev, "Motion planning in urban environments," *Journal of Field Robotics*, vol. 25, no. 11-12, pp. 939–960, 2008.

[4] S. Liu, N. Atanasov, K. Mohta, and V. Kumar, "Search-based motion planning for quadrotors using linear quadratic minimum time control," in *IEEE International Conference on Intelligent Robots and Systems*, Vancouver, BC, Canada, Sep. 2017, pp. 2872–2879.

[5] B. Macallister, J. Butzke, A. Kushleyev, H. Pandey, and M. Likhachev, "Path planning for non-circular micro aerial vehicles in constrained environments," in *IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, May 2013, pp. 3933–3940.

[6] S. Liu, K. Mohta, N. Atanasov, and V. Kumar, "Search-Based Motion Planning for Aggressive Flight in SE(3)," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2439–2446, 2018.

[7] J. Canny, J. Reif, B. Donald, and P. Xavier, "On the complexity of kinodynamic planning," in *Annual Symposium on Foundations of Computer Science*, White Plains, NY, USA, Oct. 1988, pp. 306–316.

[8] J. Canny, A. Rege, and J. Reif, "An exact algorithm for kinodynamic planning in the plane," *Discrete & Computational Geometry*, vol. 6, no. 1, pp. 461–484, Dec. 1991.

[9] T. Fraichard, "Trajectory planning in a dynamic workspace: A 'state-time space' approach," *Advanced Robotics*, vol. 13, no. 1, pp. 75–94, Jan 1998.

[10] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Randomized Kinodynamic Motion Planning with Moving Obstacles," *The International Journal of Robotics Research*, vol. 21, no. 3, pp. 233–255, 2002.

[11] M. Pivtoraiko, R. A. Knepper, and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *Journal of Field Robotics*, vol. 26, no. 3, pp. 308–333, 2009.

[12] A. Kelly, B. Nagy, and A. Kelly Bryan Nagy, "Reactive Nonholonomic Trajectory Generation via Parametric Optimal Control," *The International Journal of Robotics Research*, vol. 22 (7-8), p. 583, 2003.

[13] J. Choi and K. Huhtala, "Constrained path optimization with bézier curve primitives," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Chicago, IL, USA, 2014, pp. 246–251.

[14] M. W. Mueller, M. Hehn, and R. Dandrea, "A Computationally Efficient Motion Primitive for Quadrocopter Trajectory Generation," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, Dec. 2015.

[15] S. Upadhyay and A. Ratnoo, "Continuous-curvature path planning with obstacle avoidance using four parameter logistic curves," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 609–616, July 2016.

[16] S. Upadhyay and A. Ratnoo, "Smooth Path Planning for Unmanned Aerial Vehicles with Airspace Restrictions," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 7, pp. 1596–1612, Jul. 2017.

[17] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE International Conference on Robotics and Automation*, Shanghai, China, 2011, pp. 2520–2525.

[18] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control," in *18th IFAC World Congress*, vol. 44, no. 1, Milano, Italy, Aug.-Sep. 2011, pp. 1485–1491.