Fast Approximate Path Coordinate Motion Primitives for Autonomous Driving

Matthew Sheckells[1], Timothy M. Caldwell[2], and Marin Kobilarov[3]

***Abstract*— In autonomous driving, it is often useful to plan trajectories in a curvilinear coordinate frame with respect to a given reference curve, such as a path produced by a high-level route planner. In this domain, standard planning methods rely on expensive coordinate transformations or on solving computationally intensive boundary value problems for computing motion primitives between states. This work develops efficient, approximate path coordinate motion primitives appropriate for fast planning in autonomous driving scenarios. We gain a 1000x speed-up in primitive computation time relative to standard approaches at the loss of some precision with respect to the position along the reference line, which we statistically quantify. Motion primitive properties like path length, acceleration, and the reference line offset are exactly preserved.**

I. Introduction

Fast path planning is essential for navigating dynamic environments, since a system must quickly respond to the actions of other agents. A planner must also produce a dynamically feasible path, which can be accurately and safely tracked by the vehicle. Dynamic feasibility can be achieved during planning by restricting the set of permitted motions to a finite number of dynamic motion primitives. These primitives can be pre-computed to make the planning process efficient, with important characteristics such as energy expenditure and path length stored along with the primitive [1]. For example, a graph-based planner can sequence pre-computed primitives of known cost together instead of employing a costly boundary value optimizer or integrator to connect neighboring states at each step in the planning process.

In the realm of autonomous driving, it is useful to specify the state of the vehicle with respect to some curvilinear reference line (e.g. the center of the road) instead of with respect to a Cartesian coordinate system [2]. Numerous feedback control methods have been developed to track such reference lines [3], [4], [5], [6]. A significant benefit is that one can easily specify corridor constraints along the reference line that correspond to road boundaries or parked cars, oftentimes as simple piecewise box constraints on the state. The challenge in using motion primitives in a path coordinate representation is that the dynamics strongly depend on the curvature of the path, which is not known ahead of time. Even with knowledge of the path, it is impractical to pre-compute a full library of fully-defined motion primitives.

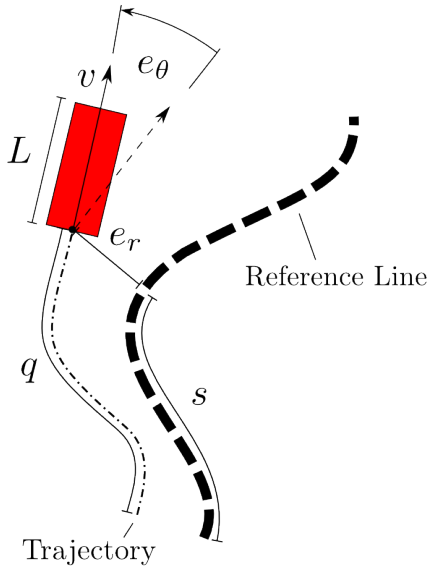

Fig. 1. Illustration of path coordinate system

To our knowledge, no pre-defined primitive for path coordinate motion planning has been developed previously. Several existing techniques plan in path coordinates using motion primitives (e.g. [7], [8], [9]), but they rely on expensive coordinate transformations or boundary value problem solvers to fully compute characteristics of the primitives at run-time.

Ziegler et al. develop a graph-based path planner which samples points on a spatiotemporal lattice in path coordinates [7]. The points are then converted to Cartesian coordinates and a boundary value problem is solved to obtain quintic spline motion primitives between states. Converting to Cartesian splines is useful since closed form expressions exist to describe the integral of squared jerk, the maximum speed, and the maximum acceleration along the splines. However, this method maps from path coordinate space to Cartesian space by evaluating or approximating Fresnel integrals, which is computationally expensive. Although the graph search itself is fast, construction of the graph takes on the order of seconds, and the construction is completely dependent on the road geometry, which changes as the car moves.

*This work was supported by and conducted at Zoox Inc.

[1]Matthew Sheckells is with the Department of Computer Science, Johns Hopkins University, 3400 N Charles Str, Baltimore, MD 21218, USA msheckells@jhu.edu

[2]Timothy M. Caldwell is with Zoox Inc., Menlo Park, California, USA timbot@zoox.com

[3]Marin Kobilarov is with Zoox Inc., Menlo Park, California, USA and with the Johns Hopkins University, 3400 N Charles Str, Baltimore MD 21218, USA marin@zoox.com

Li et al. develop a planner which employs motion primitives connecting the system's current state to sampled lookahead points on a lattice in a curvilinear path coordinate system [8]. The control trajectory between states on the lattice is given as a cubic spline over the curvature of the primitive, where a boundary value problem must be solved using a nonlinear optimization involving forward simulation of the vehicle model to get the control spline coefficients. Although the authors note that this numerical method results in easily enforced control and actuator physical constraints, it is computationally expensive.

Hudecek et al. develop transformations between Cartesian and curvilinear reference path space and perform planning in path coordinate space, using the transformations to then convert back to Cartesian space [9]. However, the transformations are computationally intensive when the reference clothoid has non-zero curvature derivative, as Fresnel integrals must be evaluated. Additionally, transforming from Cartesian space back to path coordinates requires an expensive numerical cross point search. The authors do approximate some aspects of the primitives in order to make the process efficient, as they develop bounds on curvature and sharpness in regard to feasibility of the primitive maneuvers.

In this work, we propose an easily computed path coordinate motion primitive whose state transition and control cost are almost fully specified. That is, particular characteristics of the motion primitive, such as the acceleration, are precomputed and known regardless of the reference curvature, whereas other characteristics, such as the arc length travelled along the curvilinear reference coordinates, are unknown but efficiently bounded at run-time given the reference curvature. We review path coordinate dynamics in §II and use them to develop our motion primitives, including proofs for bounded quantities, in §III. Experimental results in §IV show that the primitives are fast to evaluate given a reference trajectory and that the approximated values are accurate.

II. Path Coordinate Dynamics

During motion planning, it is useful to express the state of an autonomous vehicle with respect to some path-centric coordinate system. We define the state of the vehicle as $x = (s, e_r, e_\theta, v) \in \mathbb{R}^4$, where s is the arc length along the reference path, e_r is the lateral offset from the path, e_θ is the angular offset from the path tangent at s, and v is the forward body-velocity. Figure 1 illustrates the coordinate system. The controls of the system consist of the acceleration $a \in \mathbb{R}$ and steering angle $\delta \in \mathbb{R}$. Typical bicycle dynamics expressed using path coordinates are given as

$$\dot{s} = \frac{v \cos e_\theta}{1 - \kappa(s) e_r}, \tag{1}$$

$$\dot{e_r} = v \sin e_\theta, \tag{2}$$

$$\dot{e_\theta} = v \left(\frac{\tan \delta}{L} - \kappa(s) \dot{s} \right) = v \left(\frac{\tan \delta}{L} - \frac{\kappa(s) \cos e_\theta}{1 - \kappa(s) e_r} \right), \tag{3}$$

$$\dot{v} = a,$$

where $\kappa(s)$ is the curvature of the path at s and L is the length of the vehicle [2].

We assume that the reference line is composed of piecewise geometry segments whose curvature is linear in s, i.e. each segment is a clothoid curve. This is a standard assumption, as railroad and highway engineers use clothoids as transition curves for roads between straight and circularly curved sections [10], [11]. We parameterize the curvature of the i-th segment along the reference line as $\kappa_i(s) = \kappa_{i0} + \kappa_{i1} s$ with $\kappa_{i0}, \kappa_{i1} \in \mathbb{R}$ and define the end point of the segment as s_{κ_i}. Then, N segments can be appended piecewise to form a full reference curve

$$\kappa(s) = \begin{cases} \kappa_0(s), & 0 \le s < s_{\kappa_0} \\ \kappa_1(s), & s_{\kappa_0} \le s < s_{\kappa_1} \\ \vdots & \vdots \\ \kappa_N(s), & s_{\kappa_{N-1}} \le s \le s_{\kappa_N}, \end{cases}$$

where $\kappa(s)$ must be continuous. In the next section, we use these dynamics to develop the proposed motion primitives.

III. Path Coordinate Motion Primitives

A. Primitive Parameterization

Ideally, one would like to generate a motion primitive for which the state and control trajectories are fully defined analytically, either regardless of curvature or as a simple function of the curvature. Additionally, the mapping from the motion primitive at some current state to its final state should be efficient. A naive approach defines the motion primitives in Cartesian space since the cost of a primitive can usually be computed analytically there. At run-time, the approach maps the primitives back to the path coordinate frame when given the curvature. However, the mapping from Cartesian space to the path coordinate frame involves either an expensive cross-point search [9] or Euler integration of the path dynamics (1). This transformation makes the naive approach inappropriate for use in a motion planning algorithm which must efficiently evaluate motion primitives to operate in real-time.

We now propose a more efficient motion primitive which trades some precision with respect to the position along the reference line for a faster computation time. The motion primitives are time-invariant, so we define the primitive itself over the time interval $t \in [0, t_f]$, where t_f is a parameter of the primitives controlled by the user. We propose a motion primitive for which some of the state and control profiles, specifically $a(t)$, $v(t)$, $e_r(t)$, and $e_\theta(t)$, are fully defined and for which the arc length $s(t)$ can be bounded and approximated when given the reference curvature. The quality of the approximation depends on the variability of $e_r(t)$ and we experimentally verify in §IV that the error does not exceed 4% of the true value on average for reasonable boundary conditions and road curvatures. This formulation allows for fast, approximate planning in addition to slower, more precise planning by numerical integration of $\dot{s}$ to retrieve $s(t)$.

To create such a primitive, we parameterize the acceleration as a constant value across time, i.e. $a(t) = a$,

and the lateral path offset as cubic in time, i.e. $e_r(t) = e_{r_0} + e_{r_1}t + e_{r_2}t^2 + e_{r_3}t^3$. We use a cubic track offset so that the constants $\{e_{r_0}, e_{r_1}, e_{r_2}, e_{r_3}\}$ satisfy the boundary conditions $e_r(0), \dot{e_r}(0), e_r(t_f), \dot{e_r}(t_f)$ imposed by the state transition, i.e. continuity in e_r and e_θ. Since t_f is chosen by the user, $v(t)$ is known.

Given $a(t)$ and $e_r(t)$, the angular offset trajectory $e_\theta(t)$ can be recovered. Rearranging (2) to solve for e_θ, we get

$$e_\theta(t) = \arcsin\left(\frac{\dot{e_r}(t)}{v(t)}\right). \tag{4}$$

Using (4) for e_θ in (1), the dynamics for s can then be written as

$$\dot{s} = \frac{\sqrt{v(t)^2 - \dot{e_r}(t)^2}}{1 - \kappa(s)e_r(t)}. \tag{5}$$

Note that, even with knowledge of $\kappa(s)$, (5) cannot be solved analytically for $s(t)$.

However, if the track offset trajectory is constant, i.e. $e_r(t) = e_{r_0}$, then, by noting that $e_\theta(t) = 0$ or π in this case, we can solve for $s(t)$ analytically. We have the following result:

Proposition 1: Let $q(t) \triangleq \int_0^t v(t')\, dt'$ be the total distance travelled (see Figure 1). Let the track offset trajectory be constant, i.e. $e_r(t) = e_{r_0}$. Let the reference line consist of a single clothoid segment, so $\kappa(s) = \kappa_0 + \kappa_1 s$. Assume that $e_{r_0} < 1/\kappa(s)$, $v(t) \geq 0$, and $e_\theta(t) = 0$ (as opposed to π) along the interval considered. The arc length trajectory $s(t)$ is related to $q(t)$ by

$$s(t) = \begin{cases} \frac{(1-e_{r_0}\kappa_0) - \sqrt{(1-e_{r_0}\kappa_0)^2 - 2e_{r_0}\kappa_1 q(t)}}{e_{r_0}\kappa_1}, & e_{r_0}\kappa_1 \neq 0 \\ \frac{q(t)}{1-e_{r_0}\kappa_0}, & e_{r_0}\kappa_1 = 0. \end{cases}$$

Proof: Starting from (1), we have

$$\frac{ds}{dt} = \frac{v\cos e_\theta}{1 - e_{r_0}\kappa(s(t))} = \frac{1}{1 - e_{r_0}\kappa(s(t))}\frac{dq}{dt}.$$

Bringing s terms to the left side and integrating gives

$$\int 1 - e_{r_0}\kappa(s)\; ds = \int dq$$

$$s - e_{r_0}(\kappa_0 s + \frac{1}{2}\kappa_1 s^2) = q$$

$$-\frac{1}{2}e_{r_0}\kappa_1 s^2 + (1 - e_{r_0}\kappa_0)s - q = 0. \tag{6}$$

When $e_{r_0}\kappa_1 = 0$, then the equation can be immediately solved for $s = q/(1 - e_{r_0}\kappa_0)$. When $e_{r_0}\kappa_1 \neq 0$, we solve the quadratic in (6) to yield

$$s(t) = \frac{(1 - e_{r_0}\kappa_0) \pm \sqrt{(1 - e_{r_0}\kappa_0)^2 - 2e_{r_0}\kappa_1 q(t)}}{e_{r_0}\kappa_1}, \tag{7}$$

where the assumption $e_{r_0} < 1/\kappa(s)$ guarantees a real solution (and is not restrictive in practice).

Of the two real solutions to the quadratic, the physically meaningful solution must be chosen. Since we have $v(t) \geq 0$ and $e_\theta(t) = 0$, we must have $s(t) \geq 0$ and $q(t) \geq 0$. First, consider the case where $e_{r_0}\kappa_1 < 0$. The denominator of (7) is negative in this case. Thus, numerator must be negative to give $s(t) \geq 0$. Since $e_{r_0}\kappa_1 < 0$ and $q(t) \geq 0$, then $\sqrt{(1 - e_{r_0}\kappa_0)^2 - 2e_{r_0}\kappa_1 q(t)} \geq (1 - e_{r_0}\kappa_0)$. So, choosing the solution with numerator as $(1 - e_{r_0}\kappa_0) - \sqrt{(1 - e_{r_0}\kappa_0)^2 - 2e_{r_0}\kappa_1 q(t)}$ gives the only positive solution to $s(t)$ in this case. So,

$$s(t) = \frac{(1 - e_{r_0}\kappa_0) - \sqrt{(1 - e_{r_0}\kappa_0)^2 - 2e_{r_0}\kappa_1 q(t)}}{e_{r_0}\kappa_1}, \quad e_{r_0}\kappa_1 < 0.$$

Now, let $e_{r_0}\kappa_1 > 0$. In this case, $\sqrt{(1 - e_{r_0}\kappa_0)^2 - 2e_{r_0}\kappa_1 q(t)} \leq (1 - e_{r_0}\kappa_0)$, so both solutions of (7) are always positive. The physically meaningful solution is still the same as the previous case, since it is the only solution that has $s(t)$ increase as a function of $q(t)$. This condition is imposed by the assumption that $v(t) \geq 0$ and $e_\theta(t) = 0$, i.e. the system is always moving forward along the reference line. Thus, we now have

$$\frac{(1 - e_{r_0}\kappa_0) - \sqrt{(1 - e_{r_0}\kappa_0)^2 - 2e_{r_0}\kappa_1 q(t)}}{e_{r_0}\kappa_1}, \; e_{r_0}\kappa_1 > 0,$$

as the only feasible solution, which completes the proof. ∎

Note that the total distance travelled $q(t)$ can be precomputed based on the acceleration primitive, so solving for $s(t)$ has the complexity of taking a square root. Next, we use Proposition 1 to bound $s(t)$ in the case that $e_r(t)$ is not constant. We start by bounding $s(t)$ when the reference line consists of a single geometry segment and later extend it to the general case.

B. Bounds on Arc Length for a Single Segment

We now use the developments from §III-A to bound $s(t)$ when $e_r(t)$ is not constant and when $\kappa(s)$ consists of a single geometry segment and has constant sign over the interval considered.

Proposition 2 (Lower Bound on $s(t)$ for Single Segment): Let $e_{r_{min}} \leq e_r(t) \leq e_{r_{max}}$. Assume $\kappa(s)$ consists of a single geometry segment, has constant sign over the interval considered, and $e_{r_{min}}\kappa(s) < 1, e_{r_{max}}\kappa(s) < 1$. Define the lower bound on the distance travelled parallel to the reference curve as $q_{||}^\ell(t) \triangleq q(t) - \int_0^t |\dot{e_r}(t')|dt'$ and define $s_c(e_r, q, \kappa) \triangleq \frac{(1-e_r\kappa_0) - \sqrt{(1-e_r\kappa_0)^2 - 2e_r\kappa_1 q}}{e_r\kappa_1}$. Then $s(t)$ is bounded below by

$$s(t) \geq s_{lb}(t) \triangleq \begin{cases} s_c(e_{r_{min}}, q_{||}^\ell(t), \kappa), & e_{r_{min}}\kappa_1 \neq 0, \kappa(s) > 0 \\ \frac{q_{||}^\ell(t)}{1-e_{r_{min}}\kappa_0}, & e_{r_{min}}\kappa_1 = 0, \kappa(s) > 0 \\ s_c(e_{r_{max}}, q_{||}^\ell(t), \kappa), & e_{r_{max}}\kappa_1 \neq 0, \kappa(s) \leq 0 \\ \frac{q_{||}^\ell(t)}{1-e_{r_{max}}\kappa_0}, & e_{r_{max}}\kappa_1 = 0, \kappa(s) \leq 0. \end{cases}$$

Proof: Define $\dot{q_{||}}(t) \triangleq v(t)\cos e_\theta(t)$ as the velocity parallel to the reference line and $q_{||}(t)$ as the distance travelled parallel to the reference line. Note that $v^2 = \dot{e_r}^2 + \dot{q_{||}}^2$

which gives

$$|v| = \sqrt{\dot{e_r}^2 + \dot{q_{||}}^2} \leq \sqrt{\dot{e_r}^2} + \sqrt{\dot{q_{||}}^2}$$
$$\Rightarrow |\dot{q_{||}}| \geq |v| - |\dot{e_r}|. \tag{8}$$

Assume $v(t) \geq 0$, $\kappa(s) \geq 0$, and $-\pi/2 \leq e_\theta(t) \leq \pi/2$ over the path interval considered. From the dynamics in (1), we have

$$\begin{aligned}\dot{s} &= \frac{v \cos e_\theta}{1 - \kappa(s) e_r(t)} \\ &\geq \frac{v \cos e_\theta}{1 - \kappa(s) e_{r_{min}}} \\ &= \frac{\dot{q_{||}}}{1 - \kappa(s) e_{r_{min}}}.\end{aligned}$$

Rearranging and integrating gives

$$\begin{aligned}\int (1 - \kappa(s) e_{r_{min}}) \frac{ds}{dt}\, dt &\geq \int \frac{dq_{||}}{dt}\, dt \\ &\geq \int v - |\dot{e_r}|\, dt \\ &= q - \int |\dot{e_r}|\, dt\end{aligned} \tag{9}$$

$$s - e_{r_{min}}(\kappa_0 s + \frac{1}{2}\kappa_1 s^2) \geq q^\ell_{||}. \tag{10}$$

We solve the quadratic in (10) for the lower bound of $s(t)$. Note that the solution has the same form as Proposition 1, where s_c defines that solution for a particular constant offset and a particular path length.

The case for $\kappa(s) \leq 0$ can be proven analogously by replacing $e_{r_{min}}$ with $e_{r_{max}}$. ■

Note that $e_{r_{min}}$, $e_{r_{max}}$, and $\int |\dot{e_r}|\, dt$ can be easily pre-computed for cubic track offset trajectories, so computing the lower bound on $s(t)$ has the complexity of taking a square root.

Proposition 3 (Upper Bound on $s(t)$ for Single Segment): Assume the same conditions as Proposition 2. Then, $s(t)$ is bounded above by

$$s(t) \leq s_{ub}(t) \triangleq \begin{cases} s_c(e_{r_{max}}, q(t), \kappa), & e_{r_{max}}\kappa_1 \neq 0, \kappa(s) > 0 \\ \frac{q(t)}{1 - e_{r_{max}}\kappa_0}, & e_{r_{max}}\kappa_1 = 0, \kappa(s) > 0 \\ s_c(e_{r_{min}}, q(t), \kappa), & e_{r_{min}}\kappa_1 \neq 0, \kappa(s) \leq 0 \\ \frac{q(t)}{1 - e_{r_{min}}\kappa_0}, & e_{r_{min}}\kappa_1 = 0, \kappa(s) \leq 0. \end{cases}$$

Proof: Assume $\kappa(s) \geq 0$ over the path interval considered. From the dynamics in (1), we have

$$\begin{aligned}\dot{s} &= \frac{v \cos e_\theta}{1 - \kappa(s) e_r(t)} \\ &\leq \frac{v}{1 - \kappa(s) e_r(t)} \\ &\leq \frac{v}{1 - \kappa(s) e_{r_{max}}}.\end{aligned}$$

Rearranging and integrating gives

$$\begin{aligned}\int (1 - \kappa(s) e_{r_{max}}) \frac{ds}{dt}\, dt &\leq \int v\, dt \\ &= q\end{aligned}$$

$$s - e_{r_{max}}(\kappa_0 s + \frac{1}{2}\kappa_1 s^2) \leq q. \tag{11}$$

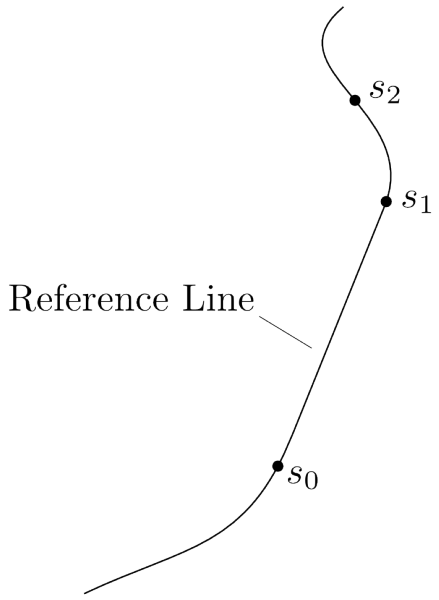

Fig. 2. A reference line composed of multiple geometry segments. Transition points along the reference line, where the curvature changes sign or switches to a new geometry segment, are marked by $\{s_0, s_1, s_2\}$.

The upper bound can now be obtained by solving the quadratic in (11). This is equivalent to solving for the change in arc length when travelling at a constant offset $e_{r_{max}}$ for a distance of length q, i.e. $s_c(e_{r_{max}}, q, \kappa)$. The case for $\kappa(s) \leq 0$ can be proven analogously by replacing $e_{r_{max}}$ with $e_{r_{min}}$. ■

For the case when $v(t) \leq 0$, reverse the sign of κ_1 and apply the same bounds given in Propositions 2 and 3, which flips the direction of the reference line and the direction of the vehicle so that $v(t) \geq 0$. Then, transform the bounds back to the original problem by taking the negative of the upper bound as the lower bound and the negative of the lower bound as the upper bound.

C. Bounds on Arc Length for Multiple Segments

Using the results developed in §III-B, we provide an algorithm for computing bounds on the change in arc length of a motion primitive that traverses a piecewise geometry whose curvature possibly changes sign. First, transition points along the reference line are identified. That is, an ordered list $(s_0, s_1, \ldots, s_n)$ is collected, where s_i corresponds to a point along the reference line where the curvature changes sign or where a new geometry segment begins (see Figure 2). Between each transition point, the curvature of the reference line is linear in s and has constant sign, and we denote the curvature function between s_{i-1} and s_i as $\kappa_i(s)$. Thus, the bounds developed earlier can be applied to each section of the reference line individually to come up with bounds for the whole trajectory. We define the function $q_c(s, e_r, \kappa) = s - e_r(\kappa_0 s + \frac{1}{2}\kappa_1 s^2)$ as the path length travelled for a constant offset e_r and a distance s along the reference line. Algorithm

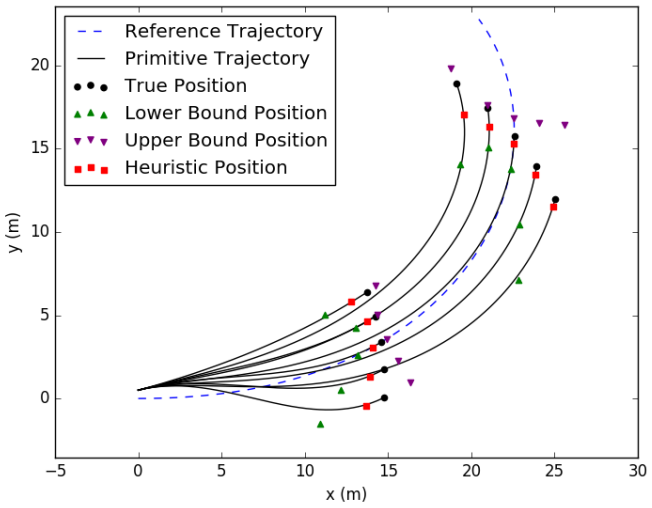

Fig. 3. Visualization of the proposed motion primitives. Note that the quickly computed heuristic position is close to the true position given by Euler integration.

1 details a procedure for computing the lower arc length bounds along a reference line with curvature that is piecewise linear in s and when $v(t) \geq 0$. The algorithm for computing the upper bound is analogous, where the remaining path length q_r is instead initialized with $q(t_f)$ and e_{r_b} is instead set to $e_{r_{max}}$ when $\text{sign}(\kappa_i) > 0$ and set to $e_{r_{min}}$ otherwise. One can see that the algorithms' computational complexities are linear in the number of transition points.

The previously developed algorithms assume a positive velocity trajectory. Since the motion primitives are parameterized by a constant acceleration, the velocity of the trajectory can switch signs at most once. To extend the algorithm to the full case, we divide the trajectory into two segments: one with positive velocity and one with negative velocity. The time horizon for each segment is easily computed based on $v(0)$ and a. The change in arc length for each time segment can be bounded separately using Algorithm 1 and the upper bound algorithm and combined together to get a bound over the full time horizon, using the strategy outlined in §III-B for getting the bounds on the negative velocity segment.

The bounds themselves can approximate the arc length $s(t_f)$ of a primitive, but, based on experiments in §IV, we propose using the heuristic $s_h(t_f) \triangleq \frac{s_{lb}(t_f)+s_{ub}(t_f)}{2}$ instead.

IV. Experimental Validation

To validate that the bounds hold and that the heuristic $s_h(t_f)$ is a good approximation of the true arc length $s(t_f)$, we randomly sample primitives and compare the approximated arc length with the true arc length which is computed using Euler integration with a small time step. Furthermore, we evaluate the computation time for approximating a path coordinate primitive and compare it to the time taken to forward integrate the primitive.

Algorithm 1 Lower bound $s(t)$ for a reference line with piecewise linear curvature.

Given $q(t), e_r(t), t_f$
Compute $e_{r_{max}}, e_{r_{min}}$ along interval $[0, t_f]$
Compute transition points $s_{trans} \leftarrow (s_0, s_1, \ldots, s_n)$
$q_r \leftarrow q(t_f) - \int_0^{t_f} |\dot{e_r}(t')|\, dt'$ // Remaining path length
$i \leftarrow 0$
$s_{lb_{total}} \leftarrow 0$
$s_{start} \leftarrow 0$
while $q_r > 0$ **do**
  if $\text{sign}(\kappa_i) > 0$ **then**
    $e_{r_b} \leftarrow e_{r_{min}}$
  else
    $e_{r_b} \leftarrow e_{r_{max}}$
  end if
  if $e_{r_b}\kappa_{i1} \neq 0$ **then**
    $s_{lb_{segment}} \leftarrow s_c(q_r, e_{r_b}, \kappa_i)$
  else
    $s_{lb_{segment}} \leftarrow \frac{q_r}{1-e_{r_b}\kappa_{i0}}$
  end if
  if $s_{start} + s_{lb_{segment}} \geq s_{trans}[i]$ **then**
    $s_{lb_{total}} \leftarrow s_{lb_{total}} + s_{trans}[i] - s_{start}$
    $q_r \leftarrow q_r - q_c(s_{trans}[i] - s_{start}, e_{r_b}, \kappa_i)$
    $s_{start} \leftarrow s_{trans}[i]$
    $i \leftarrow i + 1$
  else
    $s_{lb_{total}} \leftarrow s_{lb_{total}} + s_{lb_{segment}}$
    $q_r \leftarrow 0$
  end if
end while
return $s_{lb_{total}}$

We randomly sample 1000 primitives with track offsets $e_r(0), e_r(t_f) \in [-3, 3]$m, initial and final velocities $v(0), v(t_f) \in [1, 15]$m/s, initial and final angles $e_\theta(0), e_\theta(t_f) \in [-\frac{\pi}{12}, \frac{\pi}{12}]$, and $t_f = 5$ seconds. The curvature profiles are chosen randomly but satisfy maximum curvature guidelines based on the maximum velocity and the coefficient of friction for car tires on a dry road. Table I gives average and maximum percent errors of the bounds and the heuristic $s_h(t_f)$ relative to the true arc length. The proposed heuristic has a small percent error relative to the true arc length and has a much smaller maximum percent error than either of the bounds, motivating its use as the approximate arc length in the motion primitive. Figure 3 shows a set of primitives connecting a particular initial state to different final states with different velocities and track offsets. The position given by $s_h(t_f)$ closely approximates the true position for each of the primitives.

Figure 4 shows the ratio of the heuristic $s_h(t_f)$ computation time to the integrated $s(t_f)$ computation time for different numbers of transition points and different integration step sizes. As expected, the bound computation time varies linearly with the number of transition points in the reference path. Furthermore, the bound computation time achieves a

	Avg. Perc. Error	Max Perc. Error
s_{lb}	$10.0\% \pm 5.20\%$	46.2%
s_{ub}	$2.44\% \pm 3.14\%$	55.0%
s_h	$3.82\% \pm 1.61\%$	11.0%

TABLE I

ERROR STATISTICS OF ARC LENGTH BOUNDS AND HEURISTIC

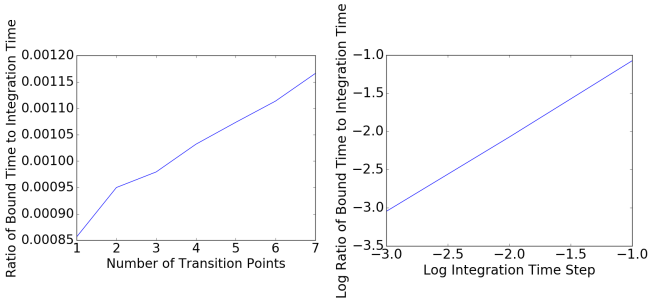

Fig. 4. Ratio of arc length bound computation time to arc length integration time versus number of transition points in the reference curve (left) and versus integration time step size (right). The bound computation is orders of magnitude faster than Euler integration. Note that the bound computation time increases about linearly with the number of transition points as expected.

1000x speed-up over the arc length integration time when using an Euler time step of 1ms. This improvement drops off linearly with the integration step size. However, making the integration step size too large actually makes the integrated arc length $s(t_f)$ less accurate than the heuristic $s_h(t_f)$. Even with a large step size of 0.1s, the heuristic arc length is still an order of magnitude faster to compute.

V. CONCLUSION

We presented a technique for the efficient computation of approximate path coordinate motion primitives, suitable for motion planning in autonomous driving scenarios. We showed that important characteristics of the motion primitives, such as path length, acceleration, and min/max velocity, can be pre-computed. We discussed how the change in arc length induced by the motion primitive is expensive to compute at run-time and proposed an efficient algorithm for bounding its value in terms of the reference curvature and minimum and maximum track offsets. Furthermore, we proposed an accurate heuristic approximation of the arc length supported by experimental validation. Future work will investigate bounds on the primitive's lateral acceleration given the reference curvature.

REFERENCES

[1] M. Likhachev and D. Ferguson, "Planning long dynamically feasible maneuvers for autonomous vehicles," *The International Journal of Robotics Research*, vol. 28, no. 8, pp. 933–945, 2009.

[2] A. De Luca, G. Oriolo, and C. Samson, "Feedback control of a nonholonomic car-like robot," in *Robot motion planning and control*, pp. 171–253, Springer, 1998.

[3] C. Samson, "Control of chained systems application to path following and time-varying point-stabilization of mobile robots," *IEEE transactions on Automatic Control*, vol. 40, no. 1, pp. 64–77, 1995.

[4] J. M. Snider *et al.*, "Automatic steering methods for autonomous automobile path tracking," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08*, 2009.

[5] P. F. Lima, M. Trincavelli, J. Mårtensson, and B. Wahlberg, "Clothoid-based speed profiler and control for autonomous driving," in *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2194–2199, IEEE, 2015.

[6] P. F. Lima, M. Trincavelli, J. Mårtensson, and B. Wahlberg, "Clothoid-based model predictive control for autonomous driving," in *European Control Conference (ECC)*, pp. 2983–2990, IEEE, 2015.

[7] J. Ziegler and C. Stiller, "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1879–1884, IEEE, 2009.

[8] X. Li, Z. Sun, Q. Zhu, and D. Liu, "A unified approach to local trajectory planning and control for autonomous driving along a reference path," in *IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 1716–1721, IEEE, 2014.

[9] J. Hudecek and L. Eckstein, "Improving and simplifying the generation of reference trajectories by usage of road-aligned coordinate systems," in *IEEE Intelligent Vehicles Symposium Proceedings*, pp. 504–509, IEEE, 2014.

[10] A. L. Higgins, *The Transition Spiral and Its Introduction to Railway Curves with Field Exercises in Construction and Alignment*. Van Nostrand Company, 1922.

[11] K. Baass, "Use of clothoid templates in highway design," tech. rep., 1982.