Quintic G^2-Splines for the Iterative Steering of Vision-Based Autonomous Vehicles

Aurelio Piazzi, *Member, IEEE*, Corrado Guarino Lo Bianco, Massimo Bertozzi, *Member, IEEE*, Alessandra Fascioli, *Member, IEEE*, and Alberto Broggi, *Associate Member, IEEE*

***Abstract*—This paper presents a new motion planning primitive to be used for the iterative steering of vision-based autonomous vehicles. This primitive is a parameterized quintic spline, denoted as η-spline, that allows interpolating an arbitrary sequence of points with overall second-order geometric (G^2-) continuity. Issues such as completeness, minimality, regularity, symmetry, and flexibility of these G^2-splines are addressed in the exposition. The development of the new primitive is tightly connected to the inversion control of nonholonomic car-like vehicles. The paper also exposes a supervisory strategy for iterative steering that integrates feedback vision data processing with the feedforward inversion control.**

***Index Terms*—Automatic steering, autonomous vehicles, dynamic inversion, iterative steering, lateral control, machine vision, path generation, quintic G^2-splines.**

I. Introduction

The MOTION planning problem for car-like vehicles or, in general, nonholonomic systems, has stimulated a significant variety of research contributions [1]–[3]. To cite just a few approaches we have Lie-algebraic techniques, geometric phases methods, control input parametrizations, and optimal planning (a survey reviewing motion planning methodologies with extensive bibliography is reported in [1]). Another planning technique is the differential flatness approach of [4], [5]. In particular, Rouchon *et al.* [6], [7] showed, for a car with multiple trailers, how to obtain an open-loop control input starting with an appropriate trajectory planning on the so-called flat output. A distinguished feature of this approach that requires the flatness property of the relevant system is the obtaining of the input function through finite order derivation of the desired flat output without involving any integration. With regard to this input synthesis, the flatness approach implies a dynamic inversion procedure performed on a system with trivial zero dynamics [8].

The necessity to provide continuous-curvature paths for the navigation of autonomous vehicles was stated by Nelson [9], [10] who devised two distinct types of paths, Cartesian quintics for lane changing maneuvers and polar splines for symmetric turns. These primitives were conceived to smoothly connect line segments so that they can be used to interpolate points with fixed zero curvature. Conversely, pursuing a (flatness) inversion control for the iterative steering of an autonomous front-wheel driving vehicle, we focus on Cartesian continuous-curvature path planning and derive a new motion planning primitive called η-spline. It is a completely parameterized quintic spline that allows the interpolation of an arbitrary sequence of points while guaranteeing an overall second-order geometric (G^2-) continuity. This new primitive permits assigning arbitrary tangent angle and curvature at each point.

The acquisition of the required data by the path planning strategy is performed by machine vision. Some errors may occur using this technique, and, thus, specific recovery strategies have been considered. These image processing techniques derive from the work on autonomous driving systems developed in the framework of the ARGO project [11]. The ARGO prototype vehicle is equipped with a vision system and autonomous steering capabilities. The control strategy discussed in this paper will replace the current gain scheduled proportional controller.

Overall, the proposed iterative steering (cf. Section IV) is another addition to the rich research field of vision-based lateral control strategies for autonomous navigation. The work of Dickmanns and co-researchers [12]–[14] has been especially relevant in this field. They developed a temporal data-fusion approach based on extended Kalman filtering for the state estimation of the vehicle with respect to its near environment and road visual scene; the vehicle autonomous guidance was then achieved through a full-state feedback control. Another successfully research effort has been the NavLab project [15], [16] and, in particular, the Rapidly Adapting Lateral Position Handler (RALPH) [17] was used for the steering guidance based on the on-line and robust visual estimation of the vehicle lateral offset, relative to the lane center, and of the upcoming road curvature. A survey with further references and some comparisons of steering strategies for vision-based autonomous navigation is reported in [18].

The paper's organization is described next. Section II describes the inversion based guidance of a car-like vehicle and motivates the posed polynomial G^2-interpolation problem. The quintic G^2-splines are derived in Section III reporting the main result (Proposition 2), an evidence for minimality (Property 1), a regularity result (Property 2), line segment generation (Property 3), and the symmetry of the η-parameterization (Property 4). Section IV exposes a supervisory strategy for iterative steering that integrates feedback vision data processing with feedforward inversion guidance. A description of the acquisition of the required data by means of visual techniques is included in Section V. The paper ends with the final remarks of Section VI.

Manuscript received April 2001; revised January 4, 2002. This work was supported in part by the Italian National Research Council (CNR) through the MADESS2 Project under Grant 99-00055-PF48 and in part by the MURST scientific research funds under the COFIN 2000 Projects. The Guest Editor for this paper was Petros A. Ioannou.

The authors are with the Dipartimento di Ingegneria dell'Informazione, Università di Parma, Parco Area delle Scienze, 43100 Parma, Italy (e-mail: aurelio@ce.unipr.it; guarino@ce.unipr.it; bertozzi@ce.unipr.it; fascal@ce.unipr.it; broggi@ce.unipr.it).

Publisher Item Identifier S 1524-9050(02)02699-6.

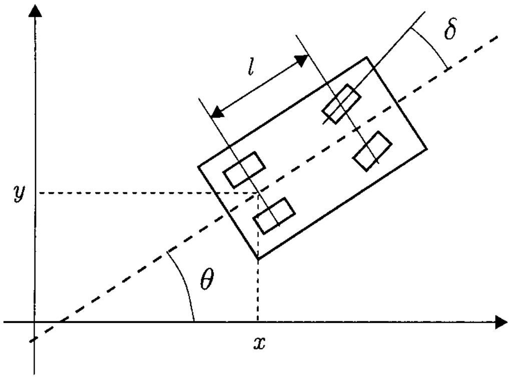

Fig. 1. The car-like vehicle and its variables.

II. Inversion Based Guidance of a Car-Like Vehicle via G^2-Path Planning

Notation: A curve on the $\{x, y\}$-plane can be described by a parameterization $\mathbf{p}(u) = [x(u)\ y(u)]^T$ with real parameter $u \in [u_0, u_1]$. The associated "path" is the image of $[u_0, u_1]$ under the vectorial function $\mathbf{p}(u)$. We say that the curve $\mathbf{p}(u)$ is regular if there exists $\dot{\mathbf{p}}(u)$ over $[u_0, u_1]$ and $\dot{\mathbf{p}}(u) \neq 0 \,\forall\, u \in [u_0, u_1]$. Associated with every point of a regular curve $\mathbf{p}(u)$ is the orthonormal moving reference system $\{\mathbf{e}_1(u), \mathbf{e}_2(u)\}$ that is congruent with the axes of the $\{x, y\}$-plane and where $\mathbf{e}_1(u)$ denotes the unit tangent vector of $\mathbf{p}(u)$. From Frenet formulae, the curvature vector is $\kappa(u)\mathbf{e}_2(u)$ where $\kappa(u)$ is the scalar curvature with well defined sign. The Euclidean norm of a vector $\mathbf{p}$ is denoted with $||\mathbf{p}||$.

We consider as a motion model of a vehicle the following simplified nonholonomic system:

$$\begin{cases} \dot{x}(t) = v\cos\theta(t) \\ \dot{y}(t) = v\sin\theta(t) \\ \dot{\theta}(t) = \dfrac{v}{l}\tan\delta(t) \end{cases} \tag{1}$$

where (see Fig. 1) x and y are the Cartesian coordinates of the rear axle midpoint, v is the velocity of this midpoint (to be considered constant), θ is the vehicle's heading angle, l is the inter-axle distance, and δ, the front wheel angle, is the control variable to steer the vehicle.

A system is defined (differentially) flat if there exists a set of outputs, equal in number to the number of inputs, and usually called flat outputs such that all states and inputs can be expressed in terms of these flat outputs and their finite-order derivatives [4], [19]. As known, system (1) is a flat system with respect to inputs v and δ and the flat outputs are simply x and y [6]. For the case at hand, with a slight modification of the standard flatness approach we consider v as a constant parameter and, on the basis of a suitable Cartesian path planning, we derive the control input $\delta(t)$ by means of a dynamic inversion procedure that does not still require any integration, cf. Proposition 1 and expression (3). Now introduce the following definitions.

Definition 1 (G^1- and G^2-Curves [20]): A parametric curve $\mathbf{p}(u)$ has first-order geometric continuity and we say $\mathbf{p}(u)$ is a G^1-curve if $\mathbf{p}(u)$ is regular and its unit tangent vector is a continuous function along the curve. The curve $\mathbf{p}(u)$ has second-order geometric continuity and we say $\mathbf{p}(u)$ is a G^2-curve if $\mathbf{p}(u)$ is a G^1-curve and its curvature vector is continuous along the curve.

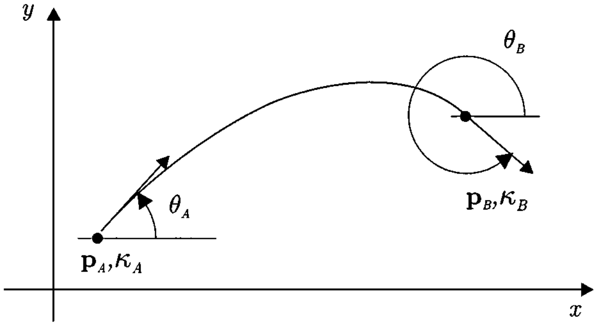

Fig. 2. The G^2-interpolation problem on the $\{x, y\}$-plane.

Definition 2 (G^1- and G^2-Paths): A path of a Cartesian space, i.e., a set of points of this space, is a G^i-path ($i = 1, 2$) if there exists a parametric G^i-curve whose image is the given path.

A basic result on the vehicle's motion of model (1) is stated below, originally presented in [21].

Proposition 1: A path on plane $\{x, y\}$ is generated by vehicle model (1) via a continuous control input $\delta(t)$ if and only if the $\{x, y\}$-path is a G^2-path.

From a control viewpoint the main consequence of the above proposition can be described as follows. Given any G^2-curve $\mathbf{p}(u)$ with $u \in [u_0, u_1]$ introduce the arc length function

$$s(u) := \int_{u_0}^{u} \sqrt{\dot{x}(v)^2 + \dot{y}(v)^2}\, dv \tag{2}$$

and denote by s^{-1}: $[0, s(u_1)] \to [u_0, u_1]$ its inverse function that is evidently a continuous function. Moreover, the scalar curvature $\kappa(u)$ is as well continuous over $[u_0, u_1]$ because $\mathbf{p}(u)$ is a G^2-curve. At the initial time t_0, consider the state of model (1) be given by $[x(u_0)\ y(u_0)\ \arg(\mathbf{e}_1(u_0))]^T$. Then, applying the continuous input

$$\delta(t) = \arctan\left[l\kappa\left(s^{-1}(v(t - t_0))\right)\right] \tag{3}$$

the vehicle's motion from t_0 to $t_0 + s(u_1)/v$ exactly matches the path of the given $\mathbf{p}(u)$ curve.

Consider a sequence of Cartesian points $\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \ldots$ on the $\{x, y\}$ plane. If a piecewise curve interpolating these points can be generated with the requirement of being an overall G^2-curve, then, using the flat dynamic inversion control given by (3), the vehicle can be steered to exactly intersects the points of the sequence. Hence, it appears naturally to pose the following problem.

Polynomial G^2-Interpolation Problem: Determine the minimal-order polynomial curve that interpolates between given points $\mathbf{p}_A = [x_A\ y_A]^T$ and $\mathbf{p}_B = [x_B\ y_B]^T$ with associated unit tangent vectors defined by angles θ_A and θ_B and scalar curvatures κ_A and κ_B (see Fig. 2). All the interpolating data $\mathbf{p}_A, \mathbf{p}_B \in \mathbb{R}^2$, $\theta_A, \theta_B \in [0, 2\pi)$, and $\kappa_A, \kappa_B \in \mathbb{R}$ can be arbitrarily assigned.

The data $\mathbf{p}_A$, θ_A, and κ_A represents the vehicle's current status at time t_A, i.e., the coordinates x_A and y_A of the rear axle midpoint, the heading angle, and the curvature given by

$$\kappa_A = (1/l)\tan\delta(t_A) \tag{4}$$

where $\delta(t_A)$ is the steering angle at the instant t_A. The data $\mathbf{p}_B$, θ_B, and κ_B are the desired future status of the vehicle, iteratively assigned by the autonomous guidance supervisor (cf. Section IV).

III. Quintic G^2-Splines

To solve the posed interpolation problem, consider a quintic polynomial curve $\mathbf{p}(u) = [x(u)\ y(u)]^T$, $u \in [0,\,1]$ where

$$x(u) := x_0 + x_1 u + x_2 u^2 + x_3 u^3 + x_4 u^4 + x_5 u^5 \quad (5)$$

$$y(u) := y_0 + y_1 u + y_2 u^2 + y_3 u^3 + y_4 u^4 + y_5 u^5. \quad (6)$$

The interpolating conditions are the following:

$$\mathbf{p}(0) = \mathbf{p}_A, \quad \mathbf{p}(1) = \mathbf{p}_B \quad (7)$$

$$\mathbf{e}_1(0) = \begin{bmatrix} \cos\theta_A \\ \sin\theta_A \end{bmatrix}, \quad \mathbf{e}_1(1) = \begin{bmatrix} \cos\theta_B \\ \sin\theta_B \end{bmatrix} \quad (8)$$

$$\kappa(0) = \kappa_A, \quad \kappa(1) = \kappa_B \quad (9)$$

where the unit tangent vector $\mathbf{e}_1(u)$ is given by $\dot{\mathbf{p}}(u)/\|\dot{\mathbf{p}}(u)\|$. The quintic polynomial curve satisfying all the above conditions can be expressed as follows:

$$x_0 = x_A \quad (10)$$

$$x_1 = \eta_1 \cos\theta_A \quad (11)$$

$$x_2 = \tfrac{1}{2}\left(\eta_3 \cos\theta_A - \eta_1^2 \kappa_A \sin\theta_A\right) \quad (12)$$

$$\begin{aligned} x_3 = {} & 10(x_B - x_A) - \left(6\eta_1 + \tfrac{3}{2}\eta_3\right)\cos\theta_A \\ & - \left(4\eta_2 - \tfrac{1}{2}\,\eta_4\right)\cos\theta_B + \tfrac{3}{2}\,\eta_1^2\kappa_A \sin\theta_A \\ & - \tfrac{1}{2}\,\eta_2^2\kappa_B \sin\theta_B \end{aligned} \quad (13)$$

$$\begin{aligned} x_4 = {} & -15(x_B - x_A) + \left(8\eta_1 + \tfrac{3}{2}\,\eta_3\right)\cos\theta_A \\ & + (7\eta_2 - \eta_4)\cos\theta_B - \tfrac{3}{2}\,\eta_1^2\kappa_A \sin\theta_A \\ & + \eta_2^2\kappa_B \sin\theta_B \end{aligned} \quad (14)$$

$$\begin{aligned} x_5 = {} & 6(x_B - x_A) - \left(3\eta_1 + \tfrac{1}{2}\eta_3\right)\cos\theta_A \\ & - \left(3\eta_2 - \tfrac{1}{2}\,\eta_4\right)\cos\theta_B + \tfrac{1}{2}\,\eta_1^2\kappa_A \sin\theta_A \\ & - \tfrac{1}{2}\,\eta_2^2\kappa_B \sin\theta_B \end{aligned} \quad (15)$$

$$y_0 = y_A \quad (16)$$

$$y_1 = \eta_1 \sin\theta_A \quad (17)$$

$$y_2 = \tfrac{1}{2}\,\left(\eta_3 \sin\theta_A + \eta_1^2\kappa_A \cos\theta_A\right) \quad (18)$$

$$\begin{aligned} y_3 = {} & 10(y_B - y_A) - \left(6\eta_1 + \tfrac{3}{2}\,\eta_3\right)\sin\theta_A \\ & - \left(4\eta_2 - \tfrac{1}{2}\,\eta_4\right)\sin\theta_B - \tfrac{3}{2}\,\eta_1^2\kappa_A \cos\theta_A \\ & + \tfrac{1}{2}\,\eta_2^2\kappa_B \cos\theta_B \end{aligned} \quad (19)$$

$$\begin{aligned} y_4 = {} & -15(y_B - y_A) + \left(8\eta_1 + \tfrac{3}{2}\,\eta_3\right)\sin\theta_A \\ & + (7\eta_2 - \eta_4)\sin\theta_B + \tfrac{3}{2}\,\eta_1^2\kappa_A \cos\theta_A \\ & - \eta_2^2\kappa_B \cos\theta_B \end{aligned} \quad (20)$$

$$\begin{aligned} y_5 = {} & 6(y_B - y_A) - \left(3\eta_1 + \tfrac{1}{2}\,\eta_3\right)\sin\theta_A \\ & - \left(3\eta_2 - \tfrac{1}{2}\,\eta_4\right)\sin\theta_B - \tfrac{1}{2}\,\eta_1^2\kappa_A \cos\theta_A \\ & + \tfrac{1}{2}\,\eta_2^2\kappa_B \cos\theta_B. \end{aligned} \quad (21)$$

The real parameters η_i, $i = 1, \ldots, 4$, appearing in expressions (10)–(21) can be packed together to form the four-dimensional vector $\boldsymbol{\eta} := [\eta_1\eta_2\eta_3\eta_4]^T$ so that the resulting parametric curve be concisely denoted as $\mathbf{p}(u;\,\boldsymbol{\eta})$ or, informally, $\boldsymbol{\eta}$-spline . Moreover, denote with $\mathcal{H}$ the set given by the Cartesian product $\mathbb{R}^+ \times \mathbb{R}^+ \times \mathbb{R} \times \mathbb{R}$. It follows the main finding of the section.

Proposition 2: Given any interpolating data $\mathbf{p}_A$, θ_A, κ_A and $\mathbf{p}_B$, θ_B, κ_B, the parametric curve $\mathbf{p}(u;\,\boldsymbol{\eta})$ satisfies conditions (7)–(9) for all $\boldsymbol{\eta} \in \mathcal{H}$. Conversely, given any quintic polynomial curve $\mathbf{p}(u)$ with $\dot{\mathbf{p}}(0) \neq 0$, $\dot{\mathbf{p}}(1) \neq 0$ satisfying (7)–(9) there exists a parameter vector $\boldsymbol{\eta} \in \mathcal{H}$ such that the quintic curve can be expressed as $\mathbf{p}(u;\,\boldsymbol{\eta})$.

The following lemma whose proof is omitted for brevity is useful in proving the above proposition.

Lemma 1: Assigned any planar curve $\mathbf{p}(u) = [x(u)\ y(u)]^T$, its scalar curvature $\kappa(u)$ is given by

$$\kappa(u) = \frac{\dot{x}(u)\ddot{y}(u) - \ddot{x}(u)\dot{y}(u)}{\left(\dot{x}(u)^2 + \dot{y}(u)^2\right)^{3/2}}. \quad (22)$$

Proof of Proposition 2—Sufficiency: It can be proved by direct computations. Indeed note that $\mathbf{p}(0;\,\boldsymbol{\eta}) = \mathbf{p}_A$ and $\mathbf{p}(1;\,\boldsymbol{\eta}) = \mathbf{p}_B$ for all $\boldsymbol{\eta} \in \mathcal{H}$. Moreover, the following relations hold for all $\boldsymbol{\eta} \in \mathcal{H}$:

$$\dot{\mathbf{p}}(0;\,\boldsymbol{\eta}) = \eta_1[\cos\theta_A\ \ \sin\theta_A]^T \quad (23)$$

$$\dot{\mathbf{p}}(1;\,\boldsymbol{\eta}) = \eta_2[\cos\theta_B\ \ \sin\theta_B]^T. \quad (24)$$

Taking into account that η_1 and η_2 are strictly positive we have $\eta_1 = \|\dot{\mathbf{p}}(0;\,\boldsymbol{\eta})\|$ and $\eta_2 = \|\dot{\mathbf{p}}(1;\,\boldsymbol{\eta})\|$ so that we derive for all $\boldsymbol{\eta} \in \mathcal{H}$ [conditions (8)]

$$\dot{\mathbf{p}}(0;\,\boldsymbol{\eta})/\|\dot{\mathbf{p}}(0;\,\boldsymbol{\eta})\| = [\cos\theta_A\ \ \sin\theta_A]^T \quad (25)$$

$$\dot{\mathbf{p}}(1;\,\boldsymbol{\eta})/\|\dot{\mathbf{p}}(1;\,\boldsymbol{\eta})\| = [\cos\theta_B\ \ \sin\theta_B]^T. \quad (26)$$

Finally, using the Cartesian expression (22) for the curvature we arrive at $\kappa(0,\,\boldsymbol{\eta}) = \kappa_A$ and $\kappa(1,\,\boldsymbol{\eta}) = \kappa_B$ for all $\boldsymbol{\eta} \in \mathcal{H}$.

Necessity: From conditions (7) we derive $x_0 = x_A$, $y_0 = y_A$, i.e., relations (10) and (16), and the equations

$$x_1 + x_2 + x_3 + x_4 + x_5 = x_B - x_A \quad (27)$$

$$y_1 + y_2 + y_3 + y_4 + y_5 = y_B - y_A. \quad (28)$$

Define

$$\eta_1 := \|\dot{\mathbf{p}}(0)\|, \quad \eta_2 := \|\dot{\mathbf{p}}(1)\| \quad (29)$$

and from the first of conditions (8) we deduce $x_1 = \eta_1 \cos\theta_A$ and $y_1 = \eta_1 \sin\theta_A$, i.e., relations (11) and (17), respectively. Analogously, the equations

$$2x_2 + 3x_3 + 4x_4 + 5x_5 = \eta_2 \cos\theta_B - \eta_1 \cos\theta_A \quad (30)$$

$$2y_2 + 3y_3 + 4y_4 + 5y_5 = \eta_2 \sin\theta_B - \eta_1 \sin\theta_A \quad (31)$$

follow from the second of conditions (8).

By virtue of Lemma 1, the condition $\kappa(0) = \kappa_A$ is equivalent to

$$\frac{1}{\eta_1^3}\,[2x_1y_2 - 2x_2y_1] = \kappa_A \quad (32)$$

so that we infer

$$-2x_2 \sin\theta_A + 2y_2 \cos\theta_A = \eta_1^2\kappa_A. \quad (33)$$

Analogously, condition $\kappa(1) = \kappa_B$ can be written as

$$\begin{aligned} \frac{1}{\eta_2^3}\,[\dot{x}(1)&(2y_2 + 6y_3 + 12y_4 + 20y_5) \\ & -(2x_2 + 6x_3 + 12x_4 + 20x_5)\dot{y}(1)] = \kappa_B \end{aligned}$$

and transformed to

$$\begin{aligned}&-(2x_2+6x_3+12x_4+20x_5)\sin\theta_B\\&\quad+(2y_2+6y_3+12y_4+20y_5)\cos\theta_B=\eta_2^2\kappa_B.\end{aligned}\tag{34}$$

Now introduce the real parameters η_3 and η_4 defined as

$$\eta_3 := 2x_2\cos\theta_A + 2y_2\sin\theta_A \tag{35}$$

$$\begin{aligned}\eta_4 :=& (2x_2+6x_3+12x_4+20x_5)\cos\theta_B\\&+(2y_2+6y_3+12y_4+20y_5)\sin\theta_B.\end{aligned}\tag{36}$$

Equation (33) and definition (35) permit finding relations (12) and (18)

$$x_2 = \tfrac{1}{2}\left(\eta_3\cos\theta_A - \eta_1^2\kappa_A\sin\theta_A\right)$$

$$y_2 = \tfrac{1}{2}\left(\eta_3\sin\theta_A + \eta_1^2\kappa_A\cos\theta_A\right).$$

In a similar way, equation (34) and the definition of η_4 can be used to determine

$$2x_2+6x_3+12x_4+20x_5 = \eta_4\cos\theta_B - \eta_2^2\kappa_B\sin\theta_B \tag{37}$$

$$2y_2+6y_3+12y_4+20y_5 = \eta_4\sin\theta_A + \eta_2^2\kappa_B\cos\theta_B. \tag{38}$$

From (27), (30), and (37) we obtain the linear equation system

$$\begin{cases}x_3+x_4+x_5 = x_B - x_A - x_1 - x_2\\ 3x_3+4x_4+5x_5 = \eta_2\cos\theta_B - \eta_1\cos\theta_A - 2x_2\\ 6x_3+12x_4+20x_5 = \eta_4\cos\theta_B - \eta_2^2\kappa_B\sin\theta_B - 2x_2.\end{cases}$$

The above system admits an unique solution in the unknowns x_3, x_4, and x_5. By using the previously found expressions for x_1 and x_2, after some computations we establish relations (13)–(15). Analogously, considering (28), (31), and (38) we arrive at the following system:

$$\begin{cases}y_3+y_4+y_5 = y_B - y_A - y_1 - y_2\\ 3y_3+4y_4+5y_5 = \eta_2\sin\theta_B - \eta_1\sin\theta_A - 2y_2\\ 6y_3+12y_4+20y_5 = \eta_4\sin\theta_A + \eta_2^2\kappa_B\cos\theta_B - 2y_2.\end{cases}$$

Again we have an unique solution in the unknowns y_3, y_4, and y_5, and using (17) and (18), relations (19)–(21) can be deduced. □

Note that, cf. (29), η_1 and η_2 are the "velocity" parameters at the beginning and end of the curve whereas η_3 and η_4 can be described as the "twist" parameters.

Property 1: The curve $\mathbf{p}(u;\,\boldsymbol{\eta})$ is the minimal-order polynomial curve interpolating any arbitrarily given data $\mathbf{p}_A,\,\mathbf{p}_B \in \mathbb{R}^2$, $\theta_A,\,\theta_B \in [0,\,2\pi)$, and $\kappa_A,\,\kappa_B \in \mathbb{R}$.

Proof: The argument is based on the already proven Proposition 2. This result points out that the curve $\mathbf{p}(u;\,\boldsymbol{\eta})$ characterizes all the polynomial curves of order at most five, that interpolate the given endpoint data. Hence, if a lower-order polynomial curve exists this must coincide with $\mathbf{p}(u;\,\boldsymbol{\eta})$ for some appropriate $\boldsymbol{\eta} \in \mathcal{H}$.

Consider a particular interpolated data set $\mathbf{p}_A = [0\ \ 0]^T$, $\mathbf{p}_B = [100\ \ 5]^T$, $\theta_A = \theta_B = 0$, and $\kappa_A = \kappa_B = 0$. From formulae (10)–(21) it is easy to derive

$$\begin{aligned}x(u;\,\boldsymbol{\eta}) =& \eta_1 u + (1/2)\eta_3 u^2\\&+[1000-6\eta_1-4\eta_2+(3/2)\eta_3+(1/2)\eta_4]u^3\\&+[-1500+8\eta_1+7\eta_2+(3/2)\eta_3-\eta_4]u^4\\&+[600-3\eta_1-3\eta_2-(1/2)\eta_3+(1/2)\eta_4]u^5\end{aligned}\tag{39}$$

$$y(u;\,\boldsymbol{\eta}) = 50u^3 - 75u^4 + 30u^5. \tag{40}$$

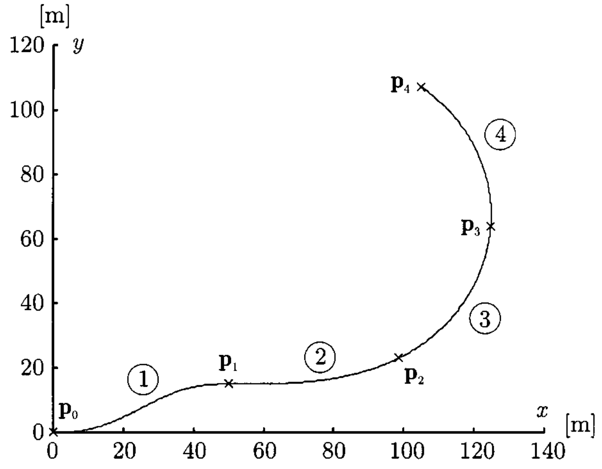

Fig. 3. $\boldsymbol{\eta}$-splines interpolating given points.

Evidently, from (40), it is not possible to interpolate the given data with a fourth- or lower-order polynomial curve for any arbitrary choice of $\boldsymbol{\eta} \in \mathcal{H}$. □

Property 2: Denote with $\mathcal{D}$ the interpolated data set (cf. the definition of the G^2-interpolation problem). The curve $\mathbf{p}(u;\,\boldsymbol{\eta})$ is generically a regular curve over $\mathcal{H}\times\mathcal{D}$.

Proof: A sketch of proof is offered. Denote with $\mathbf{d} = [x_A\ y_A\ x_B\ y_B\ \theta_A\ \theta_B\ \kappa_A\ \kappa_B]^T \in \mathcal{D}$ the vector of the interpolated data. Hence, we can more precisely redenote the $\mathbf{p}(u;\,\boldsymbol{\eta})$ curve as

$$[x(u;\,\boldsymbol{\eta},\,\mathbf{d})\ y(u;\boldsymbol{\eta},\,\mathbf{d})]^T.$$

With this notation $\mathbf{p}(u;\,\boldsymbol{\eta})$ is not a regular curve if and only if there exists $u \in [0,\,1]$ satisfying the following system of fourth-order polynomial equations:

$$\begin{cases}\dot{x}(u;\,\boldsymbol{\eta},\,\mathbf{d}) = 0\\ \dot{y}(u;\,\boldsymbol{\eta},\,\mathbf{d}) = 0.\end{cases}\tag{41}$$

Hence, necessarily, the satisfaction of system (41) implies that the associated resultant is zero

$$R(\boldsymbol{\eta},\,\mathbf{d}) = 0. \tag{42}$$

Relation (42) defines a one-dimensional manifold on $\mathcal{H}\times\mathcal{D}$. This manifold has zero Lebesgue measure on $\mathcal{H}\times\mathcal{D}$, i.e., $R(\boldsymbol{\eta},\,\mathbf{d})$ is generically not zero over $\mathcal{H}\times\mathcal{D}$. On the other hand, if $R(\boldsymbol{\eta},\,\mathbf{d}) \neq 0$ then the curve $\mathbf{p}(u;\,\boldsymbol{\eta})$ is regular so that the statement of Property 2 follows. □

Proposition 2 and Property 1 make evident that the devised $\boldsymbol{\eta}$-spline is the solution to the posed polynomial G^2-interpolation problem. In particular, Proposition 2 indicates that the $\boldsymbol{\eta}$-spline is a complete parameterization of all fifth-order polynomial curves interpolating the given endpoint data. This fact alongside the closed-form expressions (10)–(21) is very useful in performing optimal path planning [22]. On the other hand, Property 2 shows that the $\boldsymbol{\eta}$-spline per se is generically a G^2-curve and interpolating a sequence of Cartesian points with regular $\boldsymbol{\eta}$-splines results in an overall G^2-curve composed with quintic G^2-splines. Fig. 3 exemplifies $\boldsymbol{\eta}$-splines interpolating five points with assigned tangent angles and curvatures (see Table I). It is apparent from the curvature plots of Fig. 4 that the second spline is a good approximation of a clothoid whereas

TABLE I
AN INSTANCE OF G^2-INTERPOLATED DATA

	x [m]	y [m]	θ [rad]	k [m^{-1}]
$\mathbf{p}_0$	0	0	0	0
$\mathbf{p}_1$	50.00	15.00	0	0
$\mathbf{p}_2$	98.76	23.19	0.50	1/50
$\mathbf{p}_3$	124.67	63.53	1.50	1/50
$\mathbf{p}_4$	104.72	107.12	2.50	1/50

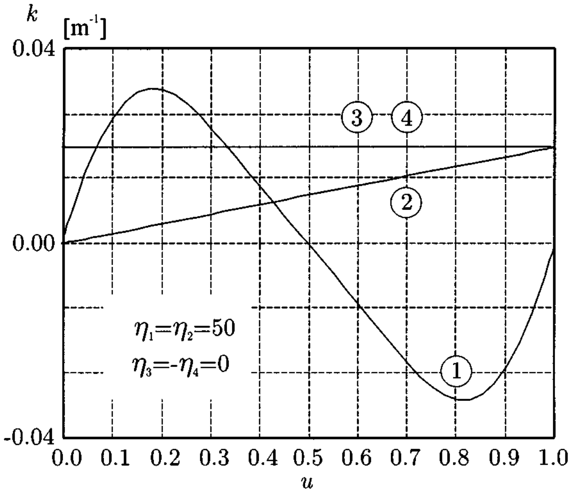

Fig. 4. Plots of curvatures of the $\boldsymbol{\eta}$-splines appearing in Fig. 3.

the third and fourth ones approximate circle arcs. This is somewhat remarkable in spite of having chosen just one set of $\boldsymbol{\eta}$-parameters: $\eta_1 = \eta_2 = 50$ and $\eta_3 = -\eta_4 = 0$ for all the splines. Better clothoid and circle approximations as well as better lane-change curves can be obtained by choosing for each spline an optimal set of $\boldsymbol{\eta}$-parameters [22].

The next result investigates the possibility to generate line segments by means of the $\boldsymbol{\eta}$-splines.

Property 3: Define $d := \|\mathbf{p}_B - \mathbf{p}_A\|$ and assume $x_B = x_A + d\cos\theta$, $y_B = y_A + d\sin\theta$, $\theta_A = \theta_B = \theta \in [0, 2\pi)$, and $\kappa_A = \kappa_B = 0$. Then the path generated by $\mathbf{p}(u; \boldsymbol{\eta})$ is a line segment for any $\boldsymbol{\eta} \in \mathcal{H}$.

Proof: The property proof is reported in [23]. □

A symmetry property with reference to the so-called "lane-change" curve is exposed below (see Fig. 5).

Property 4: Assume $\eta_1 = \eta_2 = v \in \mathbb{R}^+$ and $\eta_3 = -\eta_4 = w \in \mathbb{R}$ and define $\overline{\boldsymbol{\eta}} := [v \ v \ w \ -w]^T$. Moreover, consider $\theta_A = \theta_B = \theta \in [0, 2\pi)$, $\kappa_A = \kappa_B = 0$, and

$$\begin{cases} x_B = x_A + d_1\cos\theta - d_2\sin\theta \\ y_B = y_A + d_1\sin\theta + d_2\cos\theta \end{cases} \tag{43}$$

where $d_1 \in \mathbb{R}^+$ and $d_2 \in \mathbb{R}$. Then it follows

$$\mathbf{p}(1-u; \overline{\boldsymbol{\eta}}) = \mathbf{p}_A + \mathbf{p}_B - \mathbf{p}(u; \overline{\boldsymbol{\eta}}) \tag{44}$$

$\forall u \in [0, 1], \forall v \in \mathbb{R}^+, \forall w \in \mathbb{R}$.

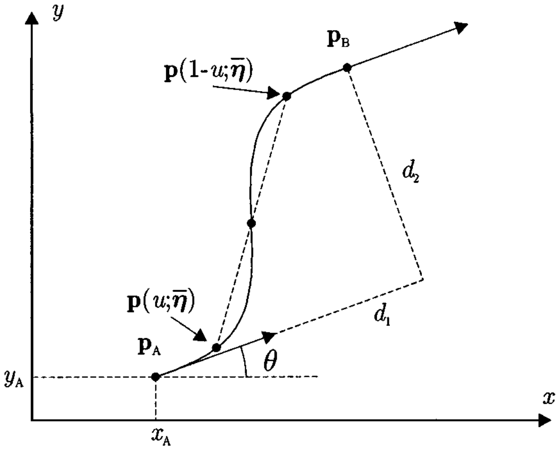
y
$\mathbf{p}_B$
$\mathbf{p}(1\text{-}u;\overline{\boldsymbol{\eta}})$
d_2
$\mathbf{p}(u;\overline{\boldsymbol{\eta}})$
d_1
$\mathbf{p}_A$
θ
y_A
x
x_A

Fig. 5. Symmetry in shaping a lane-change curve.

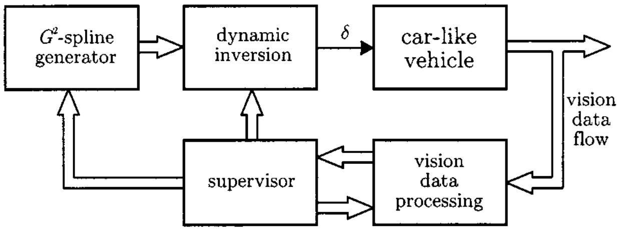

Fig. 6. The supervisory control scheme for vision-based autonomous navigation.

Proof: By direct substitution into (10)–(21) of all the posed assumptions and after some computation we have

$$\begin{aligned} \mathbf{p}(u; \overline{\boldsymbol{\eta}}) = {} & \begin{bmatrix} x_A \\ y_A \end{bmatrix} + v \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} u + \tfrac{1}{2} w \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} u^2 \\ & + \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} 10d_1 - 10v - 2w \\ 10d_2 \end{bmatrix} u^3 \\ & + \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} -15d_1 + 15v + (5/2)w \\ -15d_2 \end{bmatrix} u^4 \\ & + \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} 6d_1 - 6v - w \\ 6d_2 \end{bmatrix} u^5. \end{aligned} \tag{45}$$

Using the above expression (45) for $\mathbf{p}(u; \overline{\boldsymbol{\eta}})$ we verify that relation (44) holds for all $u \in [0, 1]$, $v \in \mathbb{R}^+$, and $w \in \mathbb{R}$. □

Besides symmetry, Property 4 points out the possibility of shaping the lane-change by varying the "velocity" parameter $\eta_1 = \eta_2 = v \in \mathbb{R}^+$ and the "twist" parameter $\eta_3 = -\eta_4 = w \in \mathbb{R}$. This can be done in an optimal fashion as explained in [22].

IV. SUPERVISORY STRATEGY FOR ITERATIVE STEERING OF VISION-BASED AUTONOMOUS VEHICLES

The overall control scheme for the iterative steering of a vision-based autonomous car-like vehicle is shown in Fig. 6. The controller kernel is given by the supervisor. It collects data coming from the vision data system and, by interacting with all the other functional blocks, imposes the steering strategy. This is based on iterative steering and on a feedforward inversion guidance exploiting the devised quintic G^2-splines. The

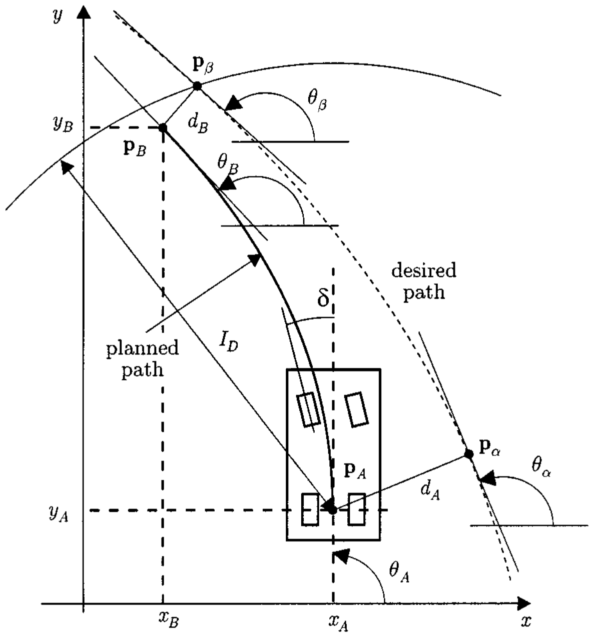

Fig. 7. Vision-based planning of the G^2-spline.

general concept of iterative-state steering was first introduced by Lucibello and Oriolo [24] for general controllable systems. In our context, the iterative steering is applied to pursue a lane following or lane inserting task (cf. Section V) but it can be used for more general navigation aims as well [21].

Fig. 7 aids in explaining the supervisory strategy. At time t_A the vehicle's state is described, with respect to a chosen Cartesian reference system $\{x, y\}$, by the position $\mathbf{p}_A$ of the middle point of the rear axle, the heading angle θ_A, and the curvature κ_A computed using relation (4) (for simplicity, the reference system $\{x, y\}$ could be chosen so that $x_A = y_A = 0$ and $\theta_A = \pi/2$). The supervisor assigns a quintic G^2-spline $\mathbf{p}(u; \boldsymbol{\eta})$ as the planned path in order that the vehicle approaches the desired path. According to (2) and (3) the corresponding open-loop steering control is given by

$$\delta(t) = \arctan[l\kappa(u; \boldsymbol{\eta})]|_{u=s^{-1}(v(t-t_A))} \tag{46}$$

where $\kappa(u; \boldsymbol{\eta})$ is the curvature of the $\boldsymbol{\eta}$-spline, s^{-1}: $[0, s(1; \boldsymbol{\eta})] \to [0, 1]$ is the inverse function of the arc length $s(u; \boldsymbol{\eta})$, and $t \in [t_A, t_A + T_p]$ with $T_p := s(1; \boldsymbol{\eta})/v$ being the planned traversal time interval.

The supervisor can iteratively update the path planning and the corresponding inversion guidance law (46) before the vehicle has covered the whole $\boldsymbol{\eta}$-spline, i.e., control (46) can be applied for $t \in [t_A, t_A + T]$ with T being a fraction of T_p ($T \le T_p$). The first part of the $\boldsymbol{\eta}$-spline is less affected by the interpolating data at $\mathbf{p}_B$, so the supervisor can help reducing the sensitivity of the overall steering to errors in the vision data processing. Moreover, a relatively fast replanning (i.e., a relatively high frequency $1/T$) can improve the robustness of the steering to modeling errors [21].

With regard to the path following convergence analysis, denote with $d(t)$ the distance of the rear axle midpoint from the desired path at time t. The delineated iterative steering provides an overall continuous steering control and guarantees convergence to the desired path provided that $d(t_A + T) < d(t_A)$ for each planning stage.

The actual assignment of the $\boldsymbol{\eta}$-spline $\mathbf{p}(u; \boldsymbol{\eta})$ also requires the choosing of the interpolated end-path data $\mathbf{p}_B$, θ_B, κ_B, and of the shaping vector parameter $\boldsymbol{\eta}$ (see Sections II and III). With some detail, the supervisor can act as follows. First, the interpolation distance, I_D, (see Fig. 7) is evaluated by the supervisor on the basis of the vehicle velocity v

$$I_D(v) := \begin{cases} I_{D_{\min}}, & \text{if } v < v^- \\ t_l\, v, & \text{if } v^- \le v \le v^+ \\ I_{D_{\max}}, & \text{if } v > v^+. \end{cases} \tag{47}$$

where $I_{D_{\min}} = v^- t_l$ and $I_{D_{\max}} = v^+ t_l$ indicate the minimum and the maximum interpolation distance, respectively, and t_l is the interpolation time constant. Parameters v^-, v^+, and t_l are determined according to the look-ahead range of the vision system.

The vision data processor receives I_D from the supervisor and provides the following data (see Fig. 7 and Section V) where:

d_A estimated vehicle distance from the desired path. It is positive or negative depending on the relative position of $\mathbf{p}_A$ to the desired path;

θ_α estimated tangent angle to the desired path at $\mathbf{p}_\alpha$; $\mathbf{p}_\alpha$ is the minimum distance point on the desired path from the vehicle rear axle midpoint;

$\mathbf{p}_\beta$ point on the desired path at the interpolating distance I_D from the vehicle rear axle midpoint;

θ_β estimated tangent angle to the desired path at $\mathbf{p}_\beta$;

κ_β estimated curvature of the desired path at $\mathbf{p}_\beta$.

The interpolated end-path data $\mathbf{p}_B$, θ_B, and κ_B are then determined through the following procedure. The parameter $K_p \in (1, +\infty]$ sets the rate of convergence to the desired path. It can be interpreted as the "position error constant" of the hybrid closed-loop system determined by the iterative steering strategy.

1) Compute d_B as

$$d_B := d_A/K_p.$$

2) Compute $\mathbf{p}_B$ according to

$$\mathbf{p}_B := \begin{bmatrix} x_B \\ y_B \end{bmatrix} = \begin{bmatrix} x_\beta - d_B \sin\theta_b \\ y_\beta + d_B \cos\theta_b \end{bmatrix}.$$

3) Compute θ_B as

$$\theta_B := \theta_\beta - \psi$$

where

$$\psi := \frac{d_A}{I_D}\left(1 - \frac{1}{K_p}\right) C_\psi(|d_B|)$$

and

$$C_\psi := \begin{cases} 0, & \text{if } |d_B| < d_B^- \\ \dfrac{|d_B| - d_B^-}{d_B^+ - d_B^-}, & \text{if } d_B^- \le |d_B| \le d_B^+ \\ 1, & \text{if } |d_B| > d_B^+. \end{cases}$$

4) Compute κ_B as

$$\kappa_B := \begin{cases} \kappa_\beta, & \text{if } |d_B| < d_B^- \\ \kappa_\beta - \dfrac{\kappa_\beta}{d_B^+ - d_B^-}(|d_B| - d_B^-), & \text{if } d_B^- \le |d_B| \le d_B^+ \\ 0, & \text{if } |d_B| > d_B^+. \end{cases}$$

The choice of the threshold positive parameters d_B^- and d_B^+ depends on the vehicle type, the camera vision system, the road conditions, etc. For the ARGO vehicle [11] on an highway, typical values are $d_B^- = 0.3$ m and $d_B^+ = 1$ m. The position error constant, K_p, should be affected by the vehicle velocity, v. Indeed, when v is low, higher values of K_p are allowed, whereas smaller values of K_p should be assigned when v is high (in any case $K_p > 1$ in order to ensure convergence to the desired path). Moreover, when d_A is small, for example, $|d_A| < d_B^-$ (the vehicle is close to the desired path), it is appropriate to set $K_p = +\infty$ that implies $d_B = 0, \mathbf{p}_B = \mathbf{p}_\beta, \theta_B = \theta_\beta, \kappa_B = \kappa_\beta$, i.e., the interpolation end-path data is given by the desired path at the point $\mathbf{p}_\beta$. When the vehicle is far from the desired path, for example, when it is engaged in a lane inserting task, the intent of the above procedure is to obtain a planned path that directly points toward the desired path. This is attained at steps 3) and 4) by the artful modification of the values of θ_B and κ_B with respect to θ_β and κ_β.

The final task of the supervisor before passing the path data to the G^2-spline generator and to the dynamic inversion block that implements the guidance law (46) is to set the shaping vector parameter $\boldsymbol{\eta}$. Choosing $\boldsymbol{\eta}$ has a strong impact on the shape of the planned path. A suitable criterion is described in [22] where $\boldsymbol{\eta}$ is selected in order to minimize the maximal variation of the curvature along the path. A suboptimal choice that is effective in most cases is to determine $\boldsymbol{\eta}$ according to the straightforward relations: $\eta_1 = \eta_2 = \|\mathbf{p}_B - \mathbf{p}_A\|$ and $\eta_3 = \eta_4 = 0$.

The inversion guidance law (46) could be easily generalized to deal with a time-varying velocity $v(t)$, and the presented supervisory steering strategy should be basically retained with only small changes. As an alternative when the vehicle velocity is slowly varying, v can be considered approximately constant within the time interval $[t_A, t_A + T]$ and is updated with the iterative replanning of the $\boldsymbol{\eta}$-spline.

V. Data Acquisition by Means of Vision

This section presents the acquisition of the required data by the path planning strategy discussed throughout this paper. In order to generate the correct trajectory the position of lane markings has to be perceived. As in many other projects [25]–[28], the sensing is robustly performed using artificial vision.

The visual-based detection of lane markings (lane detection functionality) is based on the removal of the perspective effect through inverse perspective mapping [11]. Such a transform exploits a knowledge about the acquisition parameters (camera orientation, position, optics, …) and the assumption of a flat road. The *remapped image* represents a bird's eye view of the road region in front of the vehicle. Fig. 8(a) and (b) show an image acquired by the ARGO vision system and the corresponding remapped image, respectively.

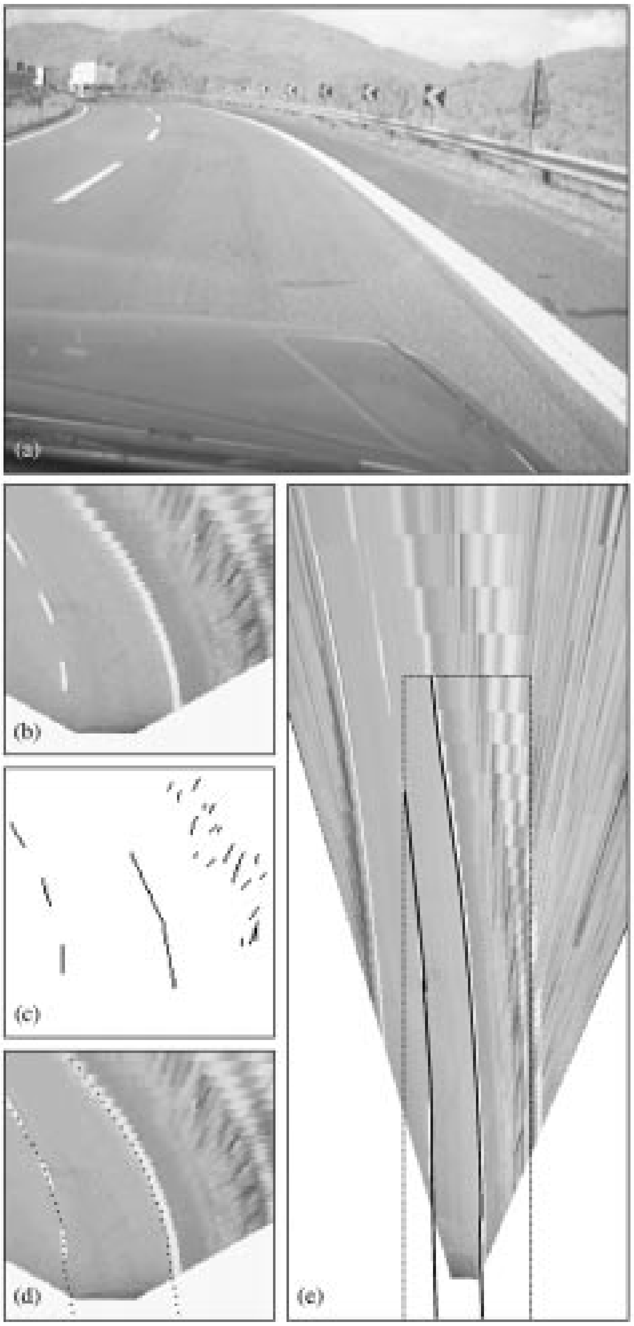

Fig. 8. Steps of lane detection. (a) Acquired image. (b) Remapped image with a 5 : 1 aspect ratio. (c) Result of the low level portion of lane detection. (d) Final result superimposed onto (b), each point represents a polyline vertex. (e) Final result represented by a concatenation of polylines, superimposed onto a remapped image of the road with a 1 : 1 aspect-ratio and a larger field of view. Area within the rectangle is the region actually processed as represented in (d).

The following subsections briefly describe the low-level and high-level processing steps used to extrapolate the data required by the supervisor to activate the proper control action.

A. Low Level Processing for Lane Detection

In the remapped image, road markings appear as quasivertical lines of constant width brighter than the road background. Therefore, the first phase of the detection is based on the extraction of pixels featuring a higher brightness value than their horizontal neighbors.

A new image, whose values encode the presence of a road marking, is computed and subsequently enhanced exploiting its vertical correlation. Finally, the image is binarized by means of an adaptive threshold [11].

The binary image is scanned in order to build chains of connected pixels. Each chain is then approximated with a *polyline* composed by one or more segments, by means of an iterative process. The result is shown in Fig. 8(c).

B. High Level Processing for Lane Detection

The resulting data structure (a list of polylines) is processed in order to semantically group homologous features and to produce a high level description of the scene. This process in divided into: filtering of noisy features and selection of the features that most likely belong to lane markings; joining of different segments in order to fill gaps caused by occlusions, dashed lines, or even worn lines; selection of the best representative and reconstruction of the information that may have been lost, on the basis of continuity constraints.

Each polyline is matched against the result of the previous frame, since continuity constraints provide a strong and robust selection procedure. The distance between the previous result and the considered polyline is computed: if it lays within a stripe centered onto the previous result then the polyline is marked as useful for the following process.

Once the polylines have been selected, all the possibilities are checked for their joining. In order to be joined, two polylines must have similar direction and must be not too far away. Their projection on the vertical axis must not overlap and, in case the gap is large, the direction of the connecting segment is checked for uniform behavior.

All the new polylines formed by concatenations of the original ones are then evaluated. Starting from a maximum score, each of the following rules provides a penalty. First each polyline is segmented. In case the polyline does not cover the whole image, a penalty is given: the polyline length is computed and a proportional penalty is given to short ones, as well as to polylines with extremely varying angular coefficients. Finally, the polyline with the highest score is selected as the best representative of the line marking.

The polyline that has been selected at the previous step may not be long enough to cover the whole image; therefore, a further step is necessary to extend the polyline. In order to take into account road curves, a parabolic model has been selected to be used in the prolongation of the polyline in the area far from the vehicle. In the nearby area, a linear approximation suffices.

This process is repeated for both left and right lane markings.

The two reconstructed polylines (one representing the left and one the right lane markings) are now matched against a model that encodes the knowledge about the absolute and relative positions of both lane markings on a standard road. A stronger model—a pair of parallel lines at a given distance (the lane width) and in a specific position—is initialized at the beginning of the whole process; a specific learning phase allows to adapt the model to errors in camera calibration (lines do not have to be perfectly parallel) and to the current lane width. Furthermore, this model can be slowly changed during the processing to adapt to new road conditions (lane width and lane position), thanks to a learning process running in the background.

The model is kept for reference. The two resulting polylines are fitted to this model, and the final result is obtained as follows. First, the two polylines are checked for nonparallel behavior. A small deviation is allowed, since it may derive from vehicle movements or deviations from the flat road assumption, that cause the calibration to be temporarily incorrect (diverging or converging lane markings). Then the quality of the two polylines, as computed in the previous steps, is matched. The final result will be attracted with a higher strength toward the polyline with the highest quality. In this way, polylines with equal or similar quality will equally contribute to the final result. On the other hand, in the case where one polyline has been heavily reconstructed, or is far from the original model, or is even missing, the other polyline will have a stronger influence in the generation of the final result.

To take into account possible small errors due to incorrect localization and/or drifts in the camera calibration, a low-pass filter is applied. The final result is computed as an average between the actual result and the result of the previous frame.

Fig. 8(d) and (e) show the result of the high level part of lane detection.

C. Control Data Estimation

The last step of visual data processing is the computation of parameters needed for path planning.

The reference system is located at the starting point of the vehicle trajectory ($\mathbf{p}_A = [0\ 0]^T$) and it is oriented such that the vehicle heading is initially directed along the y axis (i.e., $\theta_A = \pi/2$). In order to steer the vehicle along the desired path, the following parameters must be computed (see Fig. 9 and Section IV):

- coordinates of the point $\mathbf{p}_\alpha$ of the desired path closest to the reference system origin $\mathbf{p}_A$ (i.e., $d_A = ||\mathbf{p}_\alpha||$);
- coordinates of the point $\mathbf{p}_\beta$ of the desired path at distance I_D from $\mathbf{p}_A$;
- tangent angles in $\mathbf{p}_\alpha$ and $\mathbf{p}_\beta$ (i.e., θ_α and θ_β);
- path curvature in $\mathbf{p}_\beta$ (i.e., κ_β).

The left and right polylines corresponding to the left and right lane markings are used to compute the desired path. When the lane following task is pursued the desired path gets shaped as a new polyline placed onto the center of the lane. Other choices are possible as well, depending on the road shape and on particular driving needs (e.g., lane inserting, lane change, overtaking maneuvers, obstacle avoidance, etc.). Since the vehicle control module relies on values that cannot always be recovered directly from visual data, some extrapolations can be required. When the visible portion of the path does not contain all required information, its prolongations must be computed. For example, $\mathbf{p}_\alpha$ is generally close to the rear wheels and, therefore, it is not visible. In this case a linear approximation is used: the first segment of the polyline is simply prolongated assuming the path being a straight line. Conversely, the position of $\mathbf{p}_\beta$ depends on I_D which is a function of the vehicle speed. In case $\mathbf{p}_\beta$ lies outside the computed polyline, a parabolic approximation is used to recover the missing portion of the desired path far ahead the

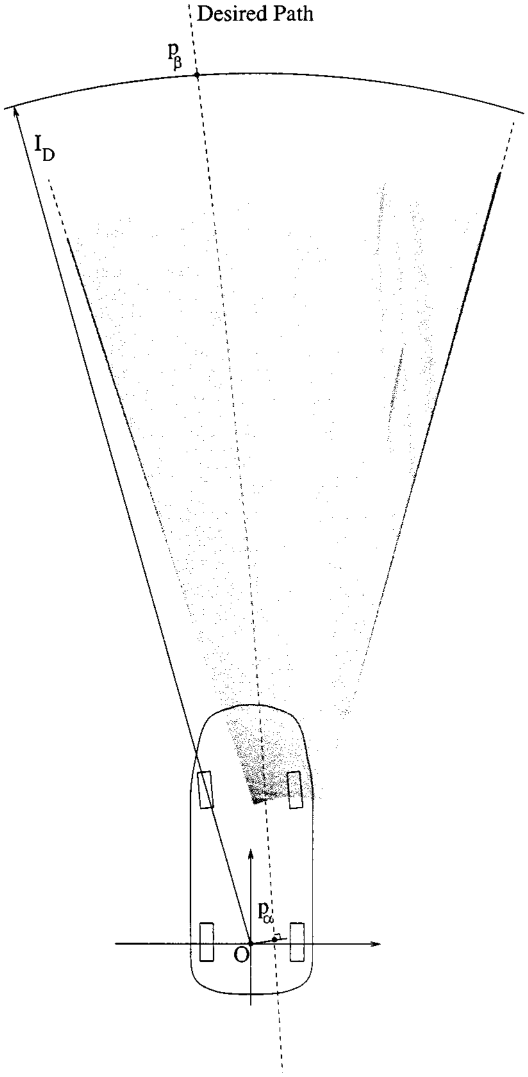

Fig. 9. Relationship between the geometrical data of the desired path and visual data. The road texture has been cut for displaying purposes.

vehicle. In such a case the last two segments of the polyline are used to determine a fitting parabola. Such a parabola is used to evaluate point $\mathbf{p}_\beta$, tangent angle θ_β, and curvature κ_β. When $\mathbf{p}_\beta$ is located inside the computed polyline, θ_β is approximated by the segment containing $\mathbf{p}_\beta$; the curvature κ_β is determined through a local circle-based best fitting.

VI. Conclusions

In this paper a new motion planning primitive, the quintic G^2-spline, has been introduced. Using the associated shaping parameter vector $\boldsymbol{\eta}$, it is easy to obtain, or approximate, line segments, circular arcs, clothoids, and lane-change curves in a unified fashion. It has been shown how this primitive can be exploited for the iterative steering of vision-based autonomous vehicles using a straightforward inversion guidance law.

A robust data acquisition is critical for this system. In fact, errors in the localization of distant lane markings may affect the planned path. Such errors may be generated by the following two causes.

- Partial visibility of lane markings: This happens in correspondence to occlusions or when the look ahead distance is far beyond the field of view, thus, an extrapolation is required.
- Errors in the processing: This happens when the calibration is incorrect or when the localization yields incorrect results.

To cope with these problems an averaging filter is applied to smooth possible sudden changes in the localization of lane markings in subsequent frames. Even if this filter cannot completely recover from errors, iterative replanning is performed before the vehicle reaches the end of the planned trajectory.

Another critical issue of the proposed iterative steering is the possibility of modeling mismatch (1) with the actual lateral dynamics of the vehicle that may depend on various uncertain parameters (road–tire interactions, vehicle handling characteristics, etc.). The supervisor can effectively face this problem ensuring the required convergence to the desired path by an appropriately fast replanning of the quintic G^2-spline. This has been evidenced in [21] for the case of $K_p = +\infty$ (i.e., the vehicle starting point is not too far from the desired path and the planned end-point $\mathbf{p}_B$ is always chosen on it) with a robustness analysis of the iterative steering based on simulations against the Wong vehicle model [29].

This new steering methodology is being implemented on the experimental ARGO vehicle and will substitute the current gain scheduled proportional controller. Moreover this technique, that can be extended to a time-varying velocity (cf. end of Section IV), can also be used for a variety of autonomous navigation tasks of car-like vehicles and mobile robots. Just to give a partial list of possible tasks that can be dealt with we can cite lane following, lane inserting, obstacle avoidance, platooning, and parking maneuvers.

References

[1] I. Kolmanovsky and N. McClamroch, “Developments in nonholonomic control problems,” *IEEE Contr. Syst. Mag.*, vol. 15, pp. 21–36, Dec. 1995.

[2] J.-C. Latombe, *Robot Motion Planning*. Norwell, MA: Kluwer, 1991.

[3] R. Murray, Z. Li, and S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, FL: CRC, 1994.

[4] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, “Flatness and defect of nonlinear systems: Introductory theory and examples,” *Int. J. Contr.*, vol. 61, no. 6, pp. 1327–1361, 1995.

[5] M. Fliess, J. Lévin, P. Martin, F. Ollivier, and P. Rouchon, “Controlling nonlinear systems by flatness,” in *Systems and Control in the Twenty-First Century*, C. Byrnes, B. Datta, D. Gillian, and C. Martin, Eds. Cambridge, MA: Birkäuser, 1997, pp. 137–154.

[6] P. Rouchon, M. Fliess, J. Lévin, and P. Martin, “Flatness, motion planning and trailer systems,” in *Proc. 32nd IEEE Conf. Decision and Control, CDC*, 1993, pp. 2700–2705.

[7] ——, “Flatness and motion planning: The car with n trailers,” in *Proc. European Control Conf. ECC’93*, Groninger, The Netherlands, 1993, pp. 1518–1522.

[8] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, “A Lie–Bäcklund approach to equivalence and flatness of nonlinear systems,” *IEEE Trans. Automat. Contr.*, vol. 44, pp. 922–937, May 1999.

[9] W. Nelson, “Continuous-curvature paths for autonomous vehicles,” in *Proc. IEEE Conf. Robotics and Automation*, vol. 3, May 1989, pp. 1260–1264.

[10] ——, “Continuous steering-function control of robot carts,” *IEEE Trans. Ind. Electron.*, vol. 36, pp. 330–337, Aug. 1989.

[11] A. Broggi, M. Bertozzi, A. Fascioli, and G. Conte, *Automatic Vehicle Guidance: The Experience of the ARGO Autonomous Vehicle*, Singapore: World Scientific, 1999.

[12] E. D. Dickmanns and A. Zapp, “Autonomous high speed road vehicle guidance by computer vision,” in *Proc. 10th IFAC World Congr.*, Munchen, Germany, 1987.

[13] E. Dickmanns and B. Mysliwetz, “Recursive 3-D road and relative ego-state recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, pp. 199–213, Feb. 1992.

[14] E. Dickmanns, “The 4D-approach to dynamic machine vision,” in *Proc. 33rd Conf. Decision and Control*, Lake Buena Vista, FL, Dec. 1994, pp. 3770–3775.

[15] C. Thorpe, M. Hebert, T. Kanade, and S. Shafer, “Vision and navigation for the Carnegie-Mellon navlab,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 10, pp. 362–373, May 1988.

[16] ——, “The new generation system for the CMU Navlab,” in *Vision-Based Vehicle Guidance*, I. Masaki, Ed. New York: Springer-Verlag, 1991, pp. 83–110.

[17] D. Pomerleau and T. Jochem, “Rapidly adapting machine vision for automated vehicle steering,” *IEEE Expert*, vol. 11, pp. 19–27, Apr. 1996.

[18] C. J. Taylor, J. Košecká, R. Blasi, and J. Malik, “A comparative study of vision-based lateral control strategies for autonomous highway driving,” *Int. J. Robot. Res.*, vol. 18, no. 5, pp. 442–453, 1999.

[19] M. V. Nieuwstadt and R. Murray, “Real-time trajectory generation for differentially flat systems,” *Int. J. Robust Nonlinear Contr.*, vol. 8, pp. 995–1020, 1998.

[20] B. A. Barsky and J. C. Beatty, “Local control of bias and tension in beta-spline,” *Comput. Graph.*, vol. 17, no. 3, pp. 193–218, 1983.

[21] A. Broggi, M. Bertozzi, A. Fascioli, C. Guarino Lo Bianco, and A. Piazzi, “The ARGO autonomous vehicle’s vision and control systems,” *Int. J. Intell. Contr. Syst.*, vol. 3, no. 4, pp. 409–441, 1999.

[22] C. Guarino Lo Bianco and A. Piazzi, “Optimal trajectory planning with quintic G^2-splines,” in *Proc. IEEE Intelligent Vehicles Symp.*, Dearborn, MI, Oct. 2000, pp. 620–625.

[23] A. Piazzi and C. Guarino Lo Bianco, “Quintic G^2-splines for trajectory planning of autonomous vehicles,” in *Proc. IEEE Intelligent Vehicles Symp.*, Dearborn, MI, Oct. 2000, pp. 198–203.

[24] P. Lucibello and G. Oriolo, “Stabilization via iterative state steering with application to chained-form systems,” in *Proc. 35th IEEE Conf. Decision and Control*, vol. 3, Kobe, Japan, Dec. 1996, pp. 2614–2619.

[25] R. Gregor, M. Lützeler, M. Pellkofer, K. Siedersberger, and E. Dickmanns, “EMS-vision: A perceptual system for autonomous vehicles,” in *Proc. IEEE Intelligent Vehicles Symp.*, Detroit, MI, Oct. 2000, pp. 52–57.

[26] F. Paetzold, U. Franke, and W. von Seelen, “Lane recognition in urban environment using optimal control theory,” in *Proc. IEEE Intelligent Vehicles Symp.*, Detroit, Oct. 2000, pp. 221–226.

[27] R. Aufrère, R. Chapuis, and F. Chausse, “A fast and robust vision based road following algorithm,” in *Proc. IEEE Intelligent Vehicles Symp.*, Detroit, MI, Oct. 2000, pp. 192–197.

[28] R. Risack, N. Möhler, and W. Enkelmann, “A video-based lane keeping assistant,” in *Proc. IEEE Intelligent Vehicles Symp.*, Detroit, MI, Oct. 2000, pp. 356–361.

[29] J. Wong, *Theory of Ground Vehicles*, 2nd ed. New York: Wiley, 1993.

Aurelio Piazzi (M’93) received the Laurea degree in nuclear engineering, in 1982 and the Ph.D. degree in system engineering, in 1987 both from the University of Bologna, Bologna, Italy.

From 1990 to 1992, he was a Research Associate in system theory, Department of Engineering and Information Systems, University of Bologna, Bologna, Italy. From 1992 to 2001, he was an Associate Professor of automatic control at the Dipartimento di Ingegneria dell’Informazione, University of Parma, Parma, Italy, since 2001, he has been a Full Professor of system theory at the same university. His main research interests are in system and control theory and related engineering applications. His recent research activities have focused on methods of global optimization applied to robotics and control problems and on dynamic inversion techniques for vision-based automatic steering and for the design of robust control systems.

Dr. Piazzi is a member of the International Federation of Automatic Control and the Society for Industrial and Applied Mathematics.

Corrado Guarino Lo Bianco received the Dr.Eng. degree in electronic engineering and the Ph.D. degree in control system engineering from the University of Bologna, Bologna, Italy, in 1989 and 1994, respectively.

Currently, he is a Research Associate with the Computer Engineering Department, University of Parma, Parma, Italy. He is involved in research on variable reluctance motors, power devices thermal analysis, smooth profile generation for motion control, robust control design via semi-infinite optimization, genetic algorithms, and interval analysis.

Massimo Bertozzi (M’95) received the Dr.Eng. degree in electronic engineering and the Ph.D. degree in information technology, both from the Università di Parma, Parma, Italy, in 1994 and 1997, respectively. His thesis was on the implementation of simulation of Petri nets on the CM-2 massive parallel architecture.

He is currently a Research Associate in the Dipartimento di Ingegnerial dell’Informazione, Università di Parma, Parma, Italy. His research interests have included the application of image processing to real-time systems and to vehicle guidance, the optimization of machine code at assembly level, and parallel and distributed computing.

Dr. Bertozzi is a member of International Association for Pattern Recognition. From 1994 to 1997, he chaired the local IEEE student branch.

Alessandra Fascioli (S’97–M’99) received the Dr.Eng. degree in electronic engineering from the Università di Parma, Parma, Italy, in 1996 and the Ph.D. degree in information technology, in 1999 from the same university.

She is currently a Temporary Researcher at the Università di Parma, Parma, Italy. Her research interests focus on real-time computer vision and computer architectures for automatic vehicle guidance. She is also interested in image processing techniques based on the mathematical morphology computational model.

Dr. Fascioli is member of the Institute of Electrical and Electronics Engineers Computer Society, Italian Association for Artificial Intelligence, and the International Association for Pattern Recognition. From 1996 to 1999, she chaired the local IEEE student branch.

Alberto Broggi (S’90–A’96) received the Dr.Eng. degree in electronic engineering in 1990 and the Ph.D. degree in information technology in 1994 both from the Università di Parma, Parma, Italy.

From 1994 to 1998, he was a Research Associate at the Dipartimento di Ingegneria dell’Informazione, Università di Parma, Parma, Italy. From 1998 to 2001, he was an Associate Professor of artificial intelligence at the Dipartimento di Informatica e Sistemistica, Università di Pavia, Pavia, Italy, and since 2001, he has been a Professor of computer science at the same university. He is the Coordinator of the ARGO project, with the aim of designing, developing, and testing the ARGO autonomous prototype vehicle, equipped with special active safety features and enhanced driving capabilities. He is the author of more than 120 refereed publications in international journals, book chapters, and conference proceedings. He is actively involved in the organization of scientific events, and is on the editorial board and program committee of many international journals and conferences, and has been invited to act as Guest Editor of journals’ and magazines’ theme issues on topics related to intelligent vehicles, computer vision application, and computer architectures for real-time image processing. His research interests include real-time computer vision approaches for the navigation of unmanned vehicles, and the development of low-cost computer systems to be used on autonomous agents.