Development of a B-Spline-based Path Planner for Semi-Autonomous Field Guidance

Patrick Fleischmann, Tobias Föhst and Karsten Berns
Robotics Research Lab, Department of Computer Science
University of Kaiserslautern
67663 Kaiserslautern, Germany
{fleischmann,foehst,berns}@cs.uni-kl.de)

***Abstract*—This paper presents a B-spline based path planning approach for agricultural guidance applications which is able to handle successively generated trajectories. Additionally, the lateral controller which was used to calculate the steering angle is described and the method by which the required parameters were determined is introduced. We explain stepwise how the output of a perception system is used to determine the control points for the planner and why B-splines and Bézier curves are used to model the trajectory. Moreover, it is shown how the inputs for the path tracker are derived from the B-splines. The approach was implemented on two different farm machines where the performance of the planning and control approach were evaluated. Furthermore, both algorithms were used to build a windrow guidance demonstrator.**

Keywords—B-splines, agricultural guidance, lateral control, Path planning, trajectory planning

I. INTRODUCTION

At present, nearly every existing automatic guidance system for agricultural vehicles like tractors, sprayers or harvesters is based on global navigation satellite systems (*GNSS*) like the *NAVSTAR-GPS*. The majority of manufacturers offer these guidance solutions like the *StarFire* system by *John Deere*, the *GPS PILOT* by *Claas* or the *EZ-GUidE* products of *New Holland*, to name only some examples. To achieve the accuracy required for agricultural applications, correction signals are used to improve the localization quality up to ±2 cm. Beside cost-free satellite based augmentation systems like *EGNOS* or *WAAS*, the manufacturers additionally offer correction signals (*StarFire SF1/SF2*, *OmniSTAR XP/HP*) or *RTK* based systems which are subject to licensing. Although very successful on the market, all these *GNSS* based systems have to deal with some main drawbacks. Weaknesses are that the localization robustness depends on structures like trees, building, bridges etc. in the environment which shadow the satellite connection, and that the tracks have to be pre-recorded or calculated by calibrated control points. Other disadvantages beside the hardware and licensing costs are that these systems are not capable of following already existing structures such as orchards, plant rows, windrows, or cut edges, as well as the inability to handle obstacles or changed environmental conditions.

These drawbacks lead us to focus on agricultural guidance solutions where *GNSS* sensors are not required, or are supported by other more cost effective or reliable sensors which map the environment. As a first proof of concept, a model driven windrow detection based on a laser scanner was developed during the last two years. To steer the tractor and implement in a smooth way and to keep both above the windrow we have identified that calculating the steering angle based on the immediate output of the sensor is not reliable, especially for higher velocities. This is because the output of the detection is not robust enough as the appearance and quantity of the straw varies too much. Additionally, a calculation of the required steering angle based only on the output of the sensor requires a specific position of the device to handle the delays of the steering actuator. Especially in curves it is very difficult to distinguish between outliers and correct values because of the very local view. In addition, our target was to reach velocities up to 20 km/h and in this case it was not possible to achieve a smooth path which is comfortable for the driver as well as suitable for the mechanics of the steering.

To solve these difficulties a trajectory planner was implemented which is able to handle the successive extension of the path, which creates an output that is smooth enough even for higher velocities and which is able to reduce the influence of outliers or detection failures. Due to the requirements a B-spline based approach was selected and implemented as Bézier curves because of the advantages of this representation. Furthermore, a lateral controller was designed to keep the tractor on the trajectory, which is also subject of this paper.

In computer graphics Bézier curves are widely used because of their numerous positive features e.g. the convex hull property which can be used for an efficient intersection calculation or a subdivision algorithm which can improve the drawing of a curve [1]. Another important attribute which is very interesting for robotic applications is that a Bézier curve – in contrast to a polynomial interpolation – will interpolate points which lie on a straight line as a line segment. The drawback that the modification of a single control point alters the whole curve can be solved by the usage of B-splines where the changes are limited to a given number of segments. Reference [2] shows an approach for a small electric car where a Bézier curve based path planner is used to alter a pre-defined path if it is blocked by an obstacle. A similar research issue is

Fig. 1. Application scenario: Tractor with a large round baler following a windrow while collecting the material.

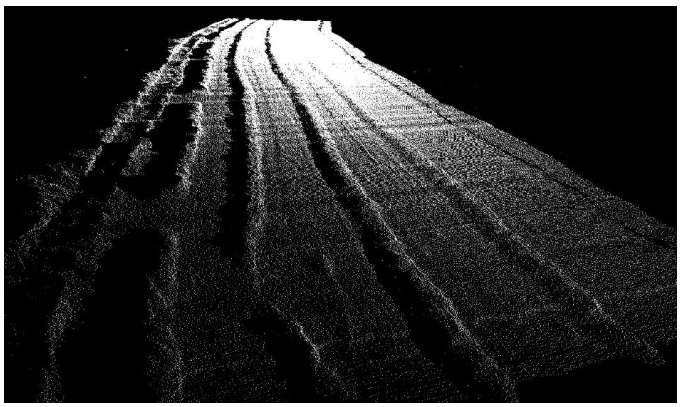

Fig. 2. 3D visualization of straw swindrows on a real field created from multiple scans.

described in [3] where a B-spline is used to create a trajectory inside an indoor environment which can be locally and thus efficiently modified if the pre-calculated path is blocked. Reference [4] described a spline based path planning algorithm for the outdoor robot *Overbot* which was a participant of the *DARPA Grand Challenge*. Three pre-defined *GPS* waypoints are interpolated each time by a cubic spline to generate a trajectory. If the path is obstructed, additional control points next to the obstacle are inserted until a collision free path is found.

For autonomous road vehicles the path tracking problem has also been a subject to numerous approaches. The *DARPA Grand* and *Urban Challenges* have pushed the development of various lateral controllers. Perhaps the most famous one was presented in [5], where a nonlinear feedback function of the cross-track and orientation error at the front axle was used to calculate the steering angle. The controller presented in [6] was also developed for an autonomous car with *Ackermann* kinematics and determines the output by comparing the vehicle's current position and orientation with a virtual reference vehicle that drives on the trajectory. A good overview of different types of lateral controllers, starting from simple geometric path trackers to controllers which have an underlying kinematic or dynamic model of the vehicle, as well as their capabilities and limitations is given in [7]. Path trackers for tractors are e.g. described in [8] where a simple *Pure Pursuit* controller is used and tested for velocities up to 8 km/h.

Unfortunately, this type of controller is limited to lower speeds as vehicle dynamics will influence the performance for higher velocities. Various experiments have shown that controllers which rely on a dynamic model easily outperform simple geometric controllers. But a main drawback of those systems is the complexity and the requirement of many parameters that are in some cases very difficult to identify. For a tractor, interesting approaches are described in [9] and [10] where the controller tries to compensate for the sliding of the farm vehicle. Here, e.g. the cornering stiffness of the tires is estimated by the controller itself, which is a very promising method as this parameter is usually very difficult to identify.

II. Application Scenario

The path planner and lateral controller which are contents of this paper are part of a showcase for a model based detection approach for typical structures in the agricultural environment. Since the described drawbacks of *GNSS*-based steering systems should be avoided, the developed assistance system exclusively relies on distance data from a laser scanner which is additionally robust against varying illumination and can be used 24/7. This guidance application for a tractor was designed to follow windrows (rows of cut or mowed crop like hay, straw, or silage) in order to collect the material using a baler. The most common type of baler, which was also used during the experiments, is the large round baler. This device collects the crop using a so-called pick-up, compresses the material in a round chamber, and wraps the bale with a mesh when the chamber is filled. Due to the fact that the collection mechanism has a limited width and that the crop is taken into the chamber as it is, the farm machine has to be driven almost centered over the swath to get a homogeneous shaped bale. Fig. 1 illustrates a tractor together with its implement during this task. Additionally, Fig. 2 shows real test scenario with straw windrows on a rye field. The image was created by combining multiple laser scans and shows the gaps and accumulations which have to be handled by the perception system as well as by the trajectory planner.

III. Trajectory Generation Using B-Splines

The input for the presented trajectory planner is the output of the detection module described in [11]. Based on the output of a 2D laser range finder the detection module uses a *RANSAC* algorithm to perceive the ground plane and a particle filter approach to find a windrow like shape in the distance data. After some filtering steps it delivers a so-called guidance point which describes the position of the swath in a tractor fixed reference frame (*RCS*) which is originated on the ground below the rear axis of the tractor. For the experiments a distance of around 11.5m in front of the rear axis was selected for the guidance point which varies with the surface of the ground, the shape of the windrow and which can be modified through the mounting position and angle of the sensor.

As a first step, these guidance points are converted from the *RCS* which moves with the tractor to a world fixed 2D coordinate system (*WCS*) which is set up during the start of the system. To convert the point to the *WCS* the position and orientation of the tractor is required which is determined by using an Inertial Measurement Unit (I*MU*) and the wheel speed signal of the tractor. It is obvious that this localization is highly error prone after a while due to the error accumulation of the *IMU* and the inaccuracies of the wheel velocities. But as the tractor follows an existing structure, only a roughly valid localization is required for the next couple of meters to create a trajectory. In addition to this conversion, every incoming guidance point is evaluated according to its drivability which is limited to a speed dependent maximal desired steering angle. This mechanism allows the tractor to drive sharp curves for low velocities, reduces unwanted amplitudes for higher velocities, and also removes outliers. Furthermore, not every point is taken as a waypoint for the trajectory since many close control points for the spline would lead to a highly varying and oscillating path, as the output of the detection is influenced by vibrations of the tractor, the nature of the surface, the noise of the sensor and the uncertainty of the probabilistic processing algorithm. To handle this, a parameterized (Euclidean) distance between two waypoints must be exceeded until the point is added as a control point (see also Fig. 5).

Driving along these predefined waypoints can be easily implemented by moving to the first point in the list and using straight lines from each waypoint to its successor. However, driving with an implement and especially at higher speeds needs a smooth parametric curved trajectory that allows error estimation for the current position and orientation even while approaching the curve and transitioning from one point to the other. Therefore, polynomial curves can be fit into the set of waypoints to define a smooth trajectory.

The typical stability issues with polynomials of higher degree can be avoided by looking only at a reduced local set of waypoints, ignoring already passed points and possibly limiting the foresight of the controller. Alternatively, the trajectory can be defined as a complete spline composed of local curves of lower degree. Additionally, the proposed method uses Bézier polygons to represent the polynomials due to their good numerical stability and fast algorithms for evaluation, derivation and drawing [1].

A. *Bézier Curves*

Every polynomial curve can be represented using its so called Bézier polygon. The curve and the polygon share their end points and end tangents and the curve lies within the convex hull of the Bézier polygon.

Applying the Bernstein polynomials

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}, \qquad i = 0, \dots, n$$

a Bézier polygon consisting of $n+1$ *control points* $\boldsymbol{c}_i$ defines a polynomial parametric curve $\boldsymbol{s}(t)$ of degree $\leq n$ with $0 \leq t \leq 1$ (see Fig. 3).

B. *The de Casteljau Algorithm*

A curve

$$\boldsymbol{s}(t) = \sum_{i=0}^{n} \boldsymbol{c}_i B_i^n(t)$$

can be easily evaluated using the de Casteljau algorithm [12].

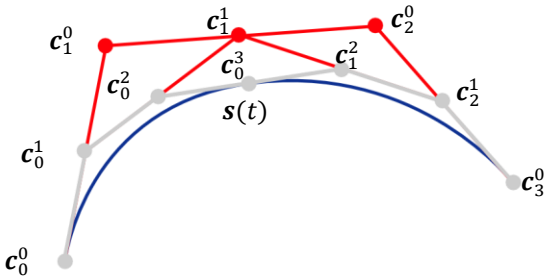

Fig. 3. The de Casteljau construction of s(t) and the resulting subdivision of a cubic Bézier polygon.

From the identity

$$\binom{n+1}{i} = \binom{n}{i-1} + \binom{n}{i}$$

follows the recursion formula for Bernstein polynomials:

$$B_i^{n+1}(t) = tB_{i-1}^n(t) + (1-t)B_i^n(t)$$

with $B_{-1}^n = B_{n+1}^n = 0$ and $B_0^0 = 1$. Hence, the curve also has a recursive formula:

$$s(t) = \sum_{i=0}^{n} \boldsymbol{c}_i^0 B_i^n(t)$$

$$= \sum_{i=0}^{n-1} \boldsymbol{c}_i^1 B_i^{n-1}(t) = \dots = \sum_{i=0}^{0} \boldsymbol{c}_i^n B_i^0(t) = \boldsymbol{c}_0^n$$

with $\boldsymbol{c}_i^{k+1} = (1-t)\boldsymbol{c}_i^k + t\boldsymbol{c}_{i+1}^k$, $\boldsymbol{c}_i^0 = \boldsymbol{c}_i$.

C. *Subdivision, Intersections, Closest Points and Drawing*

Applying the de Casteljau algorithm with an arbitrary parameter $0 \leq t \leq 1$ not only evaluates $\boldsymbol{s}(t)$ but also subdivides $\boldsymbol{s}$ at t into two new Bézier curves with control points $\{\boldsymbol{c}_0^0, \boldsymbol{c}_0^1, \dots, \boldsymbol{c}_0^n\}$ and $\{\boldsymbol{c}_0^n, \boldsymbol{c}_1^{n-1}, \dots, \boldsymbol{c}_n^0\}$ (Fig. 3). The twist of the curve (maximum of second derivative) measures the flatness of s and can therefore be used as criterion when the curve can be approximated by its baseline between the first and last control point. Hence, geometric operations like intersecting with other curves or lines, finding the closest point on the curve to a reference point, etc. can be reduced to line operations after iteratively subdividing $\boldsymbol{s}$. Even visualizing the curve can be implemented by drawing lines after subdividing depending on the current scaling (resolution) of the image. The implementation of these algorithms benefits from the convex hull property of Bézier curves and can therefore avoid unnecessary subdivisions of unaffected regions of $\boldsymbol{s}$.

D. Derivatives

To compute the orientation and curvature at a specific point on the curve $\boldsymbol{s}(t)$ the first and second derivatives must be calculated. The derivative of a Bernstein polynomial of degree n is

$$\frac{\partial}{\partial t} B_i^n(t) = n\Big(B_{i-1}^{n-1}(t) - B_i^{n-1}(t)\Big), \quad B_{-1}^{n-1} = B_n^{n-1} = 0.$$

Thus, the derivative $\boldsymbol{s}'(t)$ is obtained by

$$\frac{\partial}{\partial t}\boldsymbol{s}(t) = n\sum_{i=0}^{n-1} \Delta\boldsymbol{c}_i B_i^{n-1}(t) \;\text{ with }\; \Delta\boldsymbol{c}_i = \boldsymbol{c}_{i+1} - \boldsymbol{c}_i.$$

This makes the first derivative of $\boldsymbol{s}$ again a Bézier curve of degree $n-1$ with control points $n\Delta\boldsymbol{c}_0, \dots, n\Delta\boldsymbol{c}_n$. The second and further derivatives can be obtained recursively.

E. B-splines

Depending on the density and noise of the generated waypoints, the generated spline curve can interpolate or approximate the waypoints. Interpolation means each point is hit by the trajectory whereas approximation means that the generated curves are within a wider path spanned by the waypoints. The latter has an additional filtering effect that smooth out smaller instabilities of waypoint locations.

To create a spline curve, basis functions can be used as weights for a linear combination of the control points like in Bézier curves. While the basis functions of Bézier curves (Bernstein polynomials) are non-zero over the whole domain, a spline needs more *local* functions. This is achieved by introducing a knot vector with elements u_i and basis functions that are non-zero for adjacent subintervals of that knot vector, using the Cox-de Boor recursion formula:

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_i \le u < u_{i+1}, \\ 0 & \text{otherwise} \end{cases}$$

$$N_i^n(u) = \frac{u - u_i}{u_{i+n} - u_i} N_i^{n-1}(u) + \frac{u_{i+n+1} - u}{u_{i+n+1} - u_{i+1}} N_{i+1}^{n-1}(u).$$

Hence, a B-spline basis function $N_i^n(u)$ is non-zero on $[u_i, u_{i+n+1})$, defining the locality of the control points' influence on the curve.

From that definition, a B-spline of degree n with m control points needs a knot vector of length $n+m+1$. Satisfying this requirement means to add start and end conditions that fix the first and last segments that do not have enough neighbors to one side, e.g. by a uniform knot vector in the domain $[0,1]$ with multiple knots at 0 and 1:

$$(u_0 = \dots = u_n, u_{n+1}, \dots, u_m, u_{m+1} = \dots = u_{m+n})$$
$$= \left(0, \dots, 0, \frac{1}{m-1}, \dots, \frac{m-2}{m-1}, 1, \dots, 1\right).$$

F. Knot Insertion

One important algorithm for B-splines is knot insertion. That is inserting new knots into the knot vector together with according control points and transforming the existing control points so that the shape of the spline curve does not change.

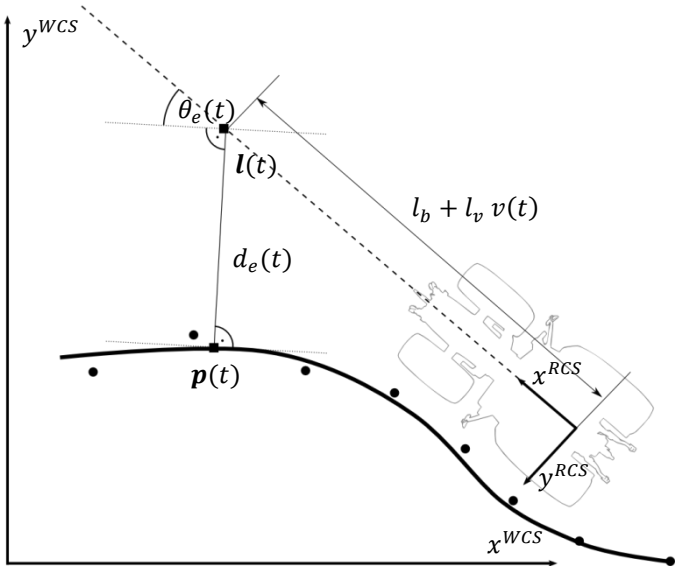

Fig. 4. Parameters of the lateral controller.

Inserting $\bar{u}$ into $u = (u_0, \dots, u_l, u_{l+1}, \dots)$ gives $v = (u_0, \dots, u_l, \bar{u}, u_{l+1}, \dots)$ and $\sum_{i=0}^{m-1} \boldsymbol{c}_i N_i^n(u)$ becomes $\sum_{i=0}^{m} \bar{\boldsymbol{c}}_i N_i^n(v)$ with the new control points

$$\bar{\boldsymbol{c}}_i = (1-\alpha_i)\boldsymbol{c}_{i-1} + \alpha_i \boldsymbol{c}_i,$$

$$\alpha_i = \begin{cases} 1 & \text{if } i \le l-n \\ 0 & \text{if } i > l \\ \dfrac{\bar{u} - u_i}{u_{l+n} - u_i} & \text{else.} \end{cases}$$

Doing that, the multiplicity of each inner knot can be increased until the affected basis functions for each segment equal to the Bernstein polynomials in the according local domain. Then, the new control points that are involved in shaping a segment become Bézier control points for that particular local curve.

IV. Lateral Controller

In order to steer the tractor along the calculated trajectory, a so-called lateral controller can be used to calculate the required steering angles. The controller which was implemented for that task uses up to 3 factors which are derived according the actual position of the robot and the current trajectory. As shown in Fig. 4, a velocity dependent look-ahead point $\boldsymbol{l}(t)$ is determined which lies on the y-axis of the tractor fixed coordinate system (*RCS*). The point can be found by evaluating the following equation:

$$\boldsymbol{l}(t)^{(RCS)} = (l_b + l_v\, v(t), 0)^T \qquad (1)$$

where l_b is a vehicle dependent, constant addend which should not be set smaller then the wheelbase of the vehicle. To consider the delay of the communication with the steering system, which has a certain reaction time, the value could be additionally increased. Additionally, the steering actuator needs some time to realize the given steering angle. This results in the requirement that the look-ahead point needs to be farther away from the vehicle for higher speeds to be able to realize the

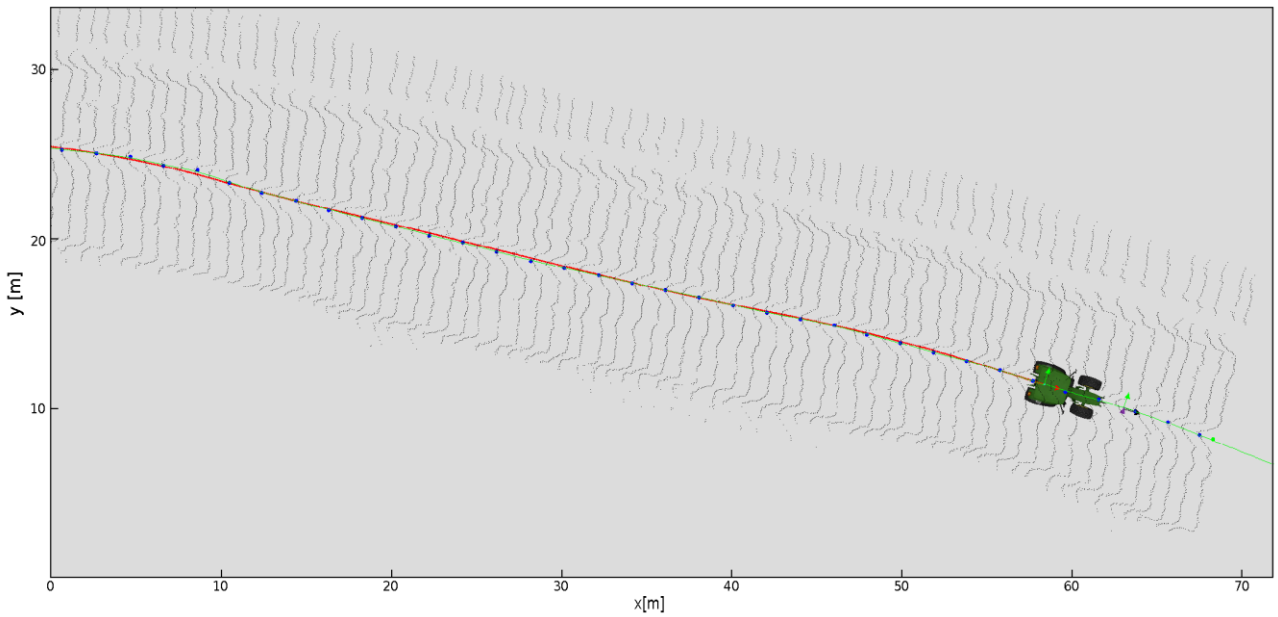

Fig. 5. A B-spline of 6th degree (green line) was used to create a trajectory which approximates control points determined by the detection module (solid blue points).

angle in time. In (1) this is modeled by a velocity-dependent part which consists of a factor l_v and the current velocity $v(t)$ of the vehicle.

After transforming the look-ahead point into the WCS by employing the current location of the tractor, the first step is to determine the closest point $\boldsymbol{p}(t)$ on the B-spline (see section III-C). This point has the shortest distance from the trajectory to the look-ahead point $\boldsymbol{l}(t)^{(WCS)}$. Once this point is found – this is done by subdividing the B-spline into small segments which can be linearized afterwards without introducing a large error – the lateral error $d_e(t)$ can be easily computed as the distance between $\boldsymbol{p}(t)$ and $\boldsymbol{l}(t)^{(WCS)}$. This error, which is also known as cross track error, indicates how far the vehicle would differ from the trajectory at the look-ahead point if the current orientation of the vehicle would be maintained (the vehicle would go straight).

To get the so-called heading error $\theta_e(t)$ which describes the difference between the orientation the vehicle should have based on the trajectory, and the projection of the current orientation to the look-ahead point, the tangent of the trajectory at $\boldsymbol{p}(t)$ is required. Instead of calculating the deviation at the closest point the tangent can be computed using the vector which points from $\boldsymbol{p}(t)$ to the look-ahead point as the tangent is always perpendicular to this vector by definition. As shown in Fig. 4 the heading error is the enclosed angle between a vector originated at $\boldsymbol{l}(t)^{(WCS)}$ while lying on the x-axis of the *RCS* and the tangent shifted to the same point.

Another factor that characterizes the trajectory, which can be used to improve the robustness of the lateral controller, can be derived from the spline be using the 2nd order derivative of the spline at $\boldsymbol{p}(t)$. It can be determined with the method presented in section III-D. In combination with the 1st derivative the curvature $\kappa(t)$ of the spline at $\boldsymbol{p}(t)$ can be calculated employing (2). It is used to predict the trend of the path to avoid oscillations of the vehicle (e.g. if you are on the right side of a trajectory and a lateral error would tend you to steer to the left although you are in a right curve).

$$\kappa(t) = \frac{\|\boldsymbol{p}'(t) \times \boldsymbol{p}''(t)\|}{\|\boldsymbol{p}'(t)\|^3} \tag{2}$$

Based on these three variables ($d_e(t)$, $\theta_e(t)$, and $\kappa(t)$) and the way they are determined, at least a cubic spline is required to be able to compute the derivatives and to get continuous values. Nevertheless, a spline of 6th degree was most suitable for the described scenario as it was a good trade-off between smoothness of the curve and computational requirements. A comparison of different degrees for an artificial trajectory is shown in Fig. 6.

Since the B-spline is linearized to calculate both errors $d_e(t)$ and $\theta_e(t)$ the output over time is not always a smooth curve. To smoothen these values, a moving average filter is attached to each calculated error which provides the mean of a certain number of past values.

There exists a wide family of lateral controllers each having different underlying models. Here, a controller based on the *Stanley Controller* [5] is used which was extended according the application's demands. This controller has several advantages. On one hand it is a simple and intuitive controller since it can be deduced from geometric relationships between the path and the vehicle.

Although controllers relying on a dynamic model are known to outperform such a controller, they are very difficult to realize for agricultural machines. For those controllers significant knowledge of machine parameters is required and a reasonable modeling of the unknown, dynamic ground surface is very challenging. On the other hand the *Stanley Controller* is very computationally efficient and produces a very suitable output under varying normal driving scenarios [7].

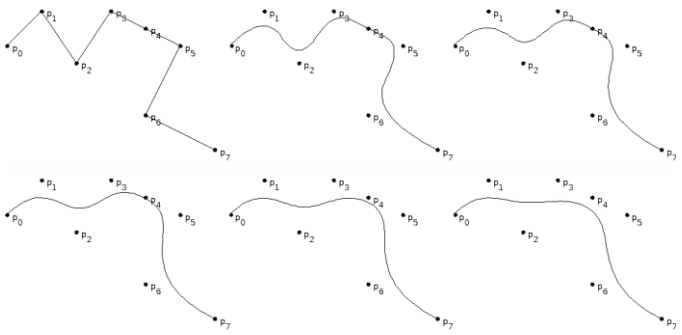

Fig. 6. Comparison of different B-splines for equal control points – from 1st degree (upper left image) which results in a linear interpolation of the points to a smooth approximating curve of 6th degree (lower right).

The controller uses the lateral and heading error information to generate a steering angle. For velocities above a certain threshold v_t the angle could be calculated using the following equation.

$$\delta(t) = k_{p_1}\left[\bar{\theta}_e(t) + \tan^{-1}\left(\frac{k_{p_2}\bar{d}_e(t) + k_i\int_0^t \bar{d}_e(\tau)\mathrm{d}\tau}{v(t)}\right)\right] \quad (3)$$

In this formula k_{p_1}, k_{p_2} and k_i are gains which have to be identified for the vehicle, $\bar{d}_e(t)$ and $\bar{\theta}_e(t)$ describe the averaged cross-track and heading errors. Our proposed method to determine these parameters is shown in section IV-A. The integral part of the arctangent function helps to remove the residual steady-state error which is important to steer the vehicle precisely along a straight line.

It can be seen that the velocity $v(t)$ of the vehicle should not be zero since it is used in the denominator of (3). In addition to that special case, velocities smaller then 1.0 let the argument grow which leads to an unwanted behavior when the vehicle is slowed down to stand. Here, the steering is sharply turned to the side even for small lateral errors. To handle this problem, the velocity $v(t)$ can be substituted in (3) by a fixed value if it is below v_t.

To improve the "quality" of the steering output according to oscillations and fast changes the steering angle is additionally processed by a moving average filter which considers the last n_δ values. Although all averaging modules introduce a delay to the whole control process, the experiments showed that the steering "experience" could be significantly improved.

A. Parameter Estimation

A crucial part is the estimation of the gains k_{p_1}, k_{p_2} and k_i used in the equation of the lateral controller. Setting them by trial and error is an endless project and success is not guaranteed. The method which is described here uses the assumption that a good human driver steers the vehicle in a very smooth way. To learn from this driving behavior, a path defined by control points, collected with a *StarFire 3000 RTK-GPS* receiver, as well as the wheel angles were recorded while manually driving the vehicle. Using this data, those parameters were calculated which lead to a set of steering angles which are as similar as possible to the angles set by the human driver.

V. Experiments And Results

To prove the performance of the proposed trajectory generation and lateral controller, a demonstrator was implemented in the robotics framework *FINROC* [13]. The path tracker together with the trajectory planner were then tested on a modified *John Deere ProGator 2030a* as well as a *John Deere 7530 Premium Series tractor*. Furthermore, the system was used together with a detection system to successfully create over 100 round bales of straw.

Fig. 5 shows a 50m excerpt of a real trajectory which was created by the windrow guidance system at a speed of 12 km/h. It was recorded on a rye field with windrows which had a mean width of 1.48 m, an average height of 0.38m and a distance of about 6 m. Based on the output of the laser scanner (small black dots) a guidance point is detected at the center of the windrow which is used as a new control point (solid blue points) at intervals of 2 m. As can be seen, the B-spline based trajectory (red line) approximates this points and leads to a smooth curve for the tractor.

The result of the described method to obtain the controller gains is shown in Fig. 8. Here, the steering angles set by a human driver on a parking lane test course are shown in blue and the output of the lateral controller is shown in red. Since the angles set by the human driver were measured by a sensor and the output of the path tracker was saved after it was determined by the algorithm, the curve of the lateral controller is shifted by about 1s. The comparison shows that the controller produces a smooth output in the majority of cases which is very similar to the wheel steering angles during the manual driving.

To analyze the performance of the path tracker a test course of about 300m (oval with an S-shaped curve inside) was built on a field. Fig. 9 shows the lateral deviation between a path which was recorded using a *RTK-GPS* receiver and the path driven by the path tracker using the same trajectory. The graph shows that the deviation lies within ±25 cm at any time. To get an impression of the test course some still images out of a video, which was captured at the same field, are shown in Fig. 7, where the tractor's velocity was set to 18 km/h.

VI. Conclusion

The evaluation of these experiments has shown that the presented approach allows robust path planning on a field and the smooth tracking of a path in a high dynamic field scenario. It was successfully tested with velocities up to 12 km/h with a round baler and up to 22 km/h without an implement. Modeling the trajectory as B-splines, represented as Bézier curves, creates smooth curves out of the guidance points with many advantages (including calculation, drawing and stability). The lateral controller used is easy to understand although it delivers reasonable results and the proposed method to determine the parameters of the controller out of a manually driven path reduces the time to set up the controller on a new vehicle.

Fig. 7. Path tracking experiments on the test course at 18 km/h (frames taken from a recorded video)

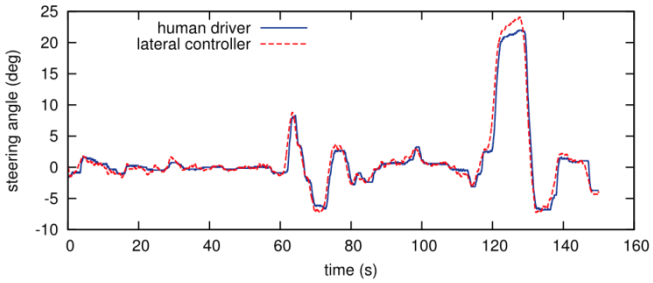

Fig. 8. Steering angles: Comparison between a human driver and the lateral controller

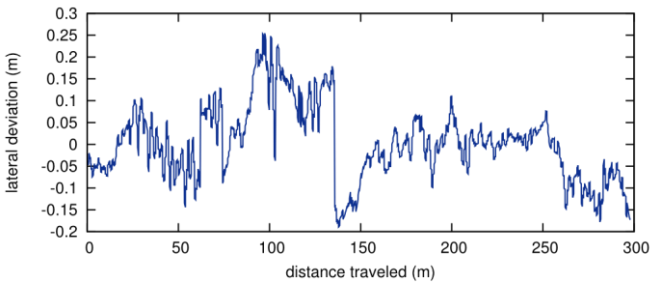

Fig. 9. Lateral deviation between the pre-recorded GPS trajectory and the path driven with proposed lateral controller

Future work will include the integration of the curvature value into the controller as well as testing the replacement of the *IMU* with an odometry-based localization to reduce the costs of the overall system.

REFERENCES

[1] H. Prautzsch, W. Boehm, and M. Paluszny, Bézier and B-Spline techniques, Springer, 2002.

[2] L. Han, H. Yashiro, H. Nejad, Q. H. Do, and S. Mita, "Bezier curve based path planning for autonomous vehicle in urban environment," in Intelligent Vehicles Symposium (IV), 2010 IEEE, University of California, San Diego, CA, USA, June 21-24 2010, pp. 1036–1042.

[3] T. Arney, "Dynamic path planning and execution using b-splines," in Proceedings, Third International Conference on Information and Automation for Sustainability (ICIAFS), December 4-6 2007, pp. 1–6.

[4] J. Connors and G. Elkaim, "Analysis of a spline based, obstacle avoiding path planning algorithm," in Proceedings, Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th, Dublin, Ireland, April 22-25 2007, pp. 2565–2569.

[5] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, and Pasc, "Stanley: The robot that won the DARPA Grand Challenge," Journal of Field Robotics, vol. 23, no. 9, September 2006, pp. 661–692.

[6] M. Linderoth, K. Soltesz, and R. Murray, "Nonlinear lateral control strategy for nonholonomic vehicles," in Proceedings, American Control Conference, 2008, June 11-13 2008, pp. 3219–3224.

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

[8] A. Stentz, C. Dima, C.Wellington, H. Herman, and D. Stager, "A system for semi-autonomous tractor operations," Autonomous Robots, vol. 13, no. 1, July 2002, pp. 87–103.

[9] H. Fang, R. Fan, B. Thuilot, and P. Martinet, "Trajectory tracking control of farm vehicles in presence of sliding." Robotics and Autonomous Systems, vol. 54, no. 10, 2006, pp. 828–839.

[10] C. Cariou, R. Lenain, B. Thuilot, and M. Berducat, "Automatic guidance of a four-wheel-steering mobile robot for accurate field operations," Journal of Field Robotics - Special Issue: Agricultural Robotics, vol. 26, no. 6–7, June – July 2009, pp. 504–518.

[11] P. Fleischmann, T. Föhst, and K. Berns, "Detection of Field Structures for Agricultural Vehicle Guidance", KI - Künstliche Intelligenz, http://dx.doi.org/10.1007/s13218-013-0264-1, 2013

[12] P. de Casteljau, "Outillages méthodes calcul," A. Citroen, Paris, Tech. Rep., 1959.

[13] M. Reichardt, T. Föhst, and K. Berns, "On software quality-motivated design of a real-time framework for complex robot control systems," in Proceedings of the 7th International Workshop on Software Quality and Maintainability (SQM), in conjunction with the 17th European Conference on Software Maintenance and Reengineering (CSMR), Genoa, Italy, March 5 2013.

