Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 50-1 (2017) 3714–3719

Autonomous Drift Parking using a Switched Control Strategy with Onboard Sensors

E. Jelavic * **J. Gonzales** ** **F. Borrelli** **

* *ETH Zurich, Zurich, Switzerland, e-mail: jelavice@ethz.ch*
** *University of California Berkeley, USA,*
e-mail: {jon.gonzales, fborrelli }@berkeley.edu

Abstract: Drift parking represents an extreme maneuver that is beyond the skill set of the average driver, requiring adept use and timing of the handbrake, pedal brake, and steering wheel. The maneuver causes the vehicle to rotate rapidly and slide, nearly sideways (i.e. high side slip angle), into the desired parking spot. In this paper we propose a control strategy to quickly park a vehicle in a narrow space using a high side slip angle maneuver, or 'drift parking' maneuver. The proposed control scheme switches between nonlinear model predictive control and a linear feedforward-feedback policy. For experimental validation, we use an open source, low cost 1/10 scale RC vehicle called the Berkeley Autonomous Race Car developed at UC Berkeley.

Keywords: Nonlinear MPC, Drifting, Parking, Autonomous Vehicles

1. INTRODUCTION

Drift maneuvers are among the most challenging coordination tasks for an expert driver to perform, either for sport or for entertainment. During motor sport events, many of the worlds best drivers deliberately operate their vehicle with saturated tires in order to perform a quick turn or a slide. Several videos online show expert drivers performing advanced maneuvers, e.g. GuinnessWorldRecords (2012).

In this paper, we focus on the design of a control scheme to park a vehicle with a sliding maneuver using a switching control architecture. When the vehicle dynamics are well modeled, the controller uses an MPC strategy (Gallestey and Al-Hokayem (2014), Rosolia and Borrelli (2016), Camacho and Alba (2013)). When the model no longer accurately captures the vehicle dynamics, the controller switches to a linear feedforward-feedback control policy. Similar switching controllers can be found throughout the literature in Atkeson and Schaal (1997), Ding (2012), Hodgins and Raibert (1990), Kolter et al. (2010), Liberzon (2012).

The motivation for the switched control architecture stems from the lack of an accurate vehicle model during the drift. Despite this, for short time spans the controller can apply a fixed input sequence, and produce a predictable, repeatable response from the system. This insight motivates the use of a feedforward policy during the sliding part of the maneuver.

To the best of the author's knowledge, the only work related to drift parking comes from Kolter and coauthors Kolter et al. (2010). They use a mixed open-loop closed-loop strategy using a probabilistic method called multi-model LQR based on a demonstration of the desired maneuver. The control design proposed in this paper is simple and can also be generalized to extreme maneuvers other than drift parking. Additionally, we validate our approach on 1/10 RC vehicle using *only* onboard sensors. Using the onboard sensors gives the advantage of having a lower cost, increased system's autonomy and increased mobility. Furthermore, onboard sensors are suitable for both outdoor and indoor use.

The paper is outlined as follows. In section 2, we discuss the vehicle and tire models used for control design and state estimation. In section 3, we discuss the path planner and the switched control strategy. In section 4 we show the results of the proposed approach in simulation using a high fidelity vehicle dynamics software called CarSim. Then, in section 5 we discuss the results of the proposed control strategy on an open source experimental 1/10 scale RC platform, called the Berkeley Autonomous Race Car (BARC). Lastly, we conclude in section 6 with some remarks on future work.

2. SYSTEM MODEL

We adopt two separate vehicle models to capture the system dynamics, a kinematic model and a kinetic model. The kinematic model is used by the the nonlinear MPC tracking control for the first part of the drift-parking maneuver, and the dynamic bicycle model is used by the nonlinear observer for the entire experiment. We discuss the details of this control architecture in the next section. The kinematic model ignores the mass of the object and focuses on describing the motion of the vehicle only using the steering angle and acceleration. The kinematic model is shown in Figure 1 and the state space form is given below with equations (1a) - (1d).

10.1016/j.ifacol.2017.08.568

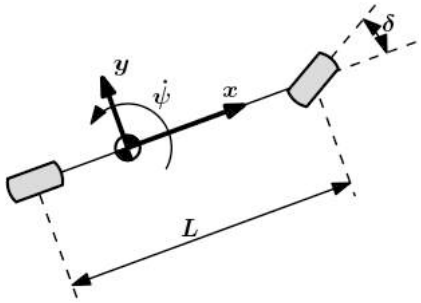

Fig. 1. Kinematic model

$$\dot{x} = v\cos\psi \tag{1a}$$
$$\dot{y} = v\sin\psi \tag{1b}$$
$$\dot{\psi} = \frac{v}{L}\tan\delta \tag{1c}$$
$$\dot{v} = a \tag{1d}$$

where (x, y) denote the coordinates of the vehicle, ψ is the yaw angle, v is the vehicle velocity, L is the length of the vehicle, δ is the steering angle, and a is the acceleration. The state and input vectors are $z = [x, y, \psi, v]^\top$ and $u = [\delta, a]^\top$, respectively. The bicycle model describes the vehicle as a single rigid body model with lumped front and rear wheels (Hindiyeh (2013)). The forces generated from these wheels move the vehicle according to Newtonian mechanics. The equations of motion for the vehicle dynamics are obtained from the balance of linear and angular momentum, yielding the following:

$$\dot{v}_y = \frac{1}{m}(F_{yF}cos\delta + F_{yR}) - rv_x \tag{2a}$$
$$\dot{r} = \frac{1}{I_z}(L_f F_{yF}cos\delta - L_r F_{yR}) \tag{2b}$$
$$\dot{v}_x = \frac{1}{m}(F_{xR} - F_{yF}\sin\delta) + v_y r \tag{2c}$$

where v_y and v_x are lateral and longitudinal velocity, respectively, $r = \dot{\psi}$ is the yaw rate, δ is the steering angle, and F_{xR} is the longitudinal rear force (assuming a rear wheel drive system). The lateral rear force is denoted with F_{yR}. Forces acting on the front tire have subscript F instead of R. The parameters m and I_z are the mass and moment of inertia about the z-axis, respectively, and L_f, L_r are geometric dimensions from the CoG to the front and rear axle. The state and input vectors are $z = [v_y, r, v_x]^\top$ and $u = [\delta, F_x]^\top$, respectively. For the tires, we use the Pacejka model (Pacejka (2005)) to estimate the lateral force as a function of tire slip angles.

$$F_y(\alpha) = \begin{cases} \mu F_z \sin(C\tan^{-1}(B\alpha)) & : |\alpha| \leq \alpha_{cr} \\ -F_y^{\max}\text{sign}(\alpha) & : |\alpha| > \alpha_{cr} \end{cases} \tag{3}$$

where the front and rear tire side slip angles are

$$\alpha_F = \tan^{-1}\left(\frac{v_y + L_f r}{v_x}\right) - \delta \tag{4a}$$
$$\alpha_R = \tan^{-1}\left(\frac{v_y - L_r r}{v_x}\right) \tag{4b}$$

In (3), μ is the coefficient of friction, F_z is the load force, B and C are fitting parameters, and α_{cr} is the critical slip angle, beyond which the tires cannot generate additional force. For lateral and longitudinal tire forces, the friction circle constraint holds:

$$F_{yi}^2 + F_{xi}^2 \leqslant (\mu F_z)^2, \quad i \in \{F, R\} \tag{5}$$

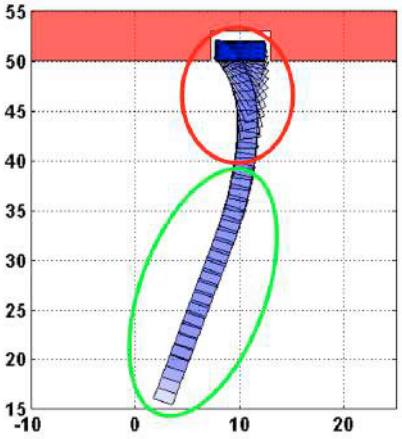

Fig. 2. Drift parking maneuver with highlighted sliding regime (red) and normal driving regime (green)

3. CONTROL SYSTEM DESIGN

Nearly all advanced driving maneuvers have some high side slip element, which manifests as sliding or drifting. Such maneuvers are beyond the skill set of the average driver, and therefore represent an interesting problem for control engineers. These high side slips maneuvers are difficult to control for two reasons: poor model fidelity in highly nonlinear regimes and actuator saturation. Additionally, the driver has to coordinate the timing and magnitude of multiple inputs.

In our control architecture, we employ a switched control approach. We control the car using MPC controller during the initial tracking segment, and then switch to feedforward-feedback controller for the last part of extreme maneuver, which is highly nonlinear. To illustrate this, we show an example parallel parking trajectory in Figure (2). The part of the trajectory encircled in green consists mainly of straight driving and slight turning. The part encircled in red involves mostly sliding, which is difficult to model accurately.

3.1 Path Planning

The objective of the path planner is to generate an input sequence that results in the vehicle sliding into the parking spot without any collisions (i.e. the vehicle remains within boundaries of the parking spot at all times). On a typical full-scale vehicle, the input signals are steering angle, throttle, hand brake, and pedal brake [1]. For the 1/10-scale RC car used for experimentation, we use steering angle and longitudinal force as inputs, $u = [\delta, F_{xR}]^\top$. The RC has a single motor that function as both an accelerator and a brake.

For the path planner, each input signal takes the form of a step function (or sum of multiple step functions) that is parameterized by two parameters: step time and magnitude. The rule-based path planner samples these parameters from a uniform distribution $\mathcal{U}(a, b)$, where a and b represent the lower and upper bounds of the sample space. The limits of the sample domain a, b are based on

[1] The handbrake acts only on the rear wheels, while the pedal brake acts on both the front and rear wheels. Further details are found in Chakraborty et al. (2011)

observing expert driver conducting extreme maneuvers.
During a drift parking maneuver, the driver executes a sequence of precisely timed actions. He first swerves the vehicle with the steering wheel, next pulls the handbrake (causing the vehicle's wheels to lock), and then applies the pedal brake. Based on this sequence of maneuvers (i.e. first steer, next use handbrake, then pedal brake), the authors select an appropriate range (a, b) for each parameter. This allows the path planner to sample from a smaller range and more quickly find a successful input sequence.
Each input signal is of the form below,

$$\delta(t) = \begin{cases} 0 & : t \leq t_1 \\ \delta_1 & : t_1 < t \end{cases} \tag{6}$$

$$F_{\text{hb}}(t) = \begin{cases} 0 & : t \leq t_2 \\ F_1 & : t_2 < t \end{cases} \tag{7}$$

$$F_{\text{pb}}(t) = \begin{cases} 0 & : t \leq t_3 \\ F_2 & : t_3 < t \end{cases} \tag{8}$$

where $0 < t_1 \leq t_2 \leq t_3$ and δ_1, F_1, F_2 are the sampled parameters.
Algorithm 1 outlines the steps of the rule-based path planner.

Algorithm 1 Path planner

```
1: Define obstacles
2: Define initial state z_0
3: while i ≤ i_max do
4:     u ← U(a, b)
5:     z ← f(z_0, u)
6:     if collisionFree(x) then
7:         u* ← u
8:         z* ← z
9:         return (z*, u*)
10:    end if
11:    i ← i + 1
12: end while
```

The input sequence $\mathbf{u}$ is generated from parameters that are sampled from the uniform distribution $\mathcal{U}(a, b)$. The initial state z_0 and input sequence $\mathbf{u}$ are then fed into the vehicle model to generate the vehicle's trajectory. Lastly, the function *collisionFree* checks if the vehicle remains within the parking spot boundaries. If successful, then the algorithm returns the reference trajectory $(\boldsymbol{z}^*, \boldsymbol{u}^*)$, otherwise it continues to sample and generate other candidate step functions. The vehicle model f from line 5 comes from CarSim, a high fidelity vehicle dynamics software. For the experiments, the vehicle's trajectory was obtained from a demonstration. This corresponds to the Algorithm (1) terminating in one iteration since the demonstration trajectory is collision free. Note, the reference trajectory is *discrete* with (z_t^*, u_t^*) corresponding to a single state/input pair at time $\tau = t$.

3.2 Path Following & Switched Control

The proposed controller tracks a drift parking trajectory by switching between a nonlinear MPC program and a linear feedforward-feedback control policy. The controller determines automatically when to switch based on the feasibility of the MPC optimization routine. At each time step, the controller first localizes itself to the nearest point on the reference trajectory, and then attempts to solve an MPC optimization routine to track the trajectory.
The controller first locates the nearest point on the reference trajectory z_i^* from the current state estimate $\hat{z}_t$ according to the optimization program in equation (9).

$$D_t = \min_i \|z_i^* - \hat{z}_t\|_2 \tag{9}$$

We use the optimal index i to construct a target tracking trajectory $(\bar{\boldsymbol{z}}, \bar{\boldsymbol{u}})$ for the MPC at each time step, and the minimum distance D_t as a constraint parameter in the MPC. The target tracking trajectory is a subset of the reference trajectory. For a horizon preview horizon N, the target tracking trajectory is constructed as

$$\bar{\boldsymbol{z}} = \begin{bmatrix} z_i^*, & z_{i+1}^* & \dots & z_{i+N}^* \end{bmatrix} \tag{10}$$

$$\bar{\boldsymbol{u}} = \begin{bmatrix} u_i^*, & u_{i+1}^* & \dots & u_{i+N}^* \end{bmatrix} \tag{11}$$

The MPC routine attempts to track the target trajectory by solving the the following optimization problem at each time step t:

$$\begin{aligned} \min_{\boldsymbol{z},\boldsymbol{u}} \quad & \sum_{k=0}^{N-1} (z_k - \bar{z}_k)^T Q (z_k - \bar{z}_k) \\ & + (u_k - \bar{u}_k)^T R (u_k - \bar{u}_k) \\ & + \Delta {u_k}^T P \Delta u_k \\ & + (z_N - \bar{x}_N)^T Q (x_N - \bar{z}_N) \end{aligned} \tag{12a}$$

$$s.t \quad z_{k+1} = f(z_k, u_k), \qquad \forall k \tag{12b}$$

$$z_0 = \hat{z}_t \tag{12c}$$

$$u_{\min} \leqslant u_k \leqslant u_{\max}, \qquad \forall k \tag{12d}$$

$$\Delta u_{\min} \leqslant \Delta u_k \leqslant \Delta u_{\max}, \qquad \forall k \tag{12e}$$

$$\Delta u_k = \frac{u_k - u_{k-1}}{T_s}, \qquad \forall k \tag{12f}$$

$$z_{\min} \leqslant z_k \leqslant z_{\max}, \qquad \forall k \tag{12g}$$

$$\Delta z_{\min} \leqslant \Delta z \leqslant \Delta z_{\max}, \qquad \forall k \tag{12h}$$

$$\Delta z_k = \frac{z_k - z_{k-1}}{T_s}, \qquad \forall k \tag{12i}$$

$$\|z_k - \bar{z}_k\| \leqslant D_t, \quad \forall k \tag{12j}$$

$$-\epsilon \leqslant z_k - \bar{z}_k \leqslant \epsilon, \quad \forall k \tag{12k}$$

In (12a), $\boldsymbol{z}$ and $\boldsymbol{u}$ represent the optimized state and input vectors, and $\bar{\boldsymbol{z}}$ and $\bar{\boldsymbol{u}}$ denote the target state and input vectors, with k indexing the time step. For the objective function, $Q \in \mathbb{S}_{++}^n$ [2] and $R, P \in \mathbb{S}_{++}^n$ represent state and input penalties.

Constraint (12b) imposes system dynamics (dynamics given with equations (1a) - (1d)), with the initial state given by constraint (12c), which is estimated from sensor measurements. Constraints (12d) and (12e) limit the input magnitude and rate of change, respectively. T_s denotes sampling time in the system.

Constraint (12g) restricts the magnitude of the state, and constraint (12e) bounds the rate of change of the state. These constraints prevent the car from driving too fast and also limit the yaw rate, which prevents oscillations. Constraint (12j) places a hard limit on the maximum position deviation from the trajectory, where D_t is the last estimated deviation from the trajectory. Constraint (12k) places hard limit on the deviation of the heading angle.

[2] $\mathbb{S}_{++}^n$ denotes the set of positive definite symmetric matrices of size n

The MPC controller tracks the trajectory closely when the dynamics are well modeled. Once the maneuver becomes aggressive and starts to slide, the model becomes unreliable, and the MPC program will become infeasible, and will fail to track the trajectory. At that point, the control system switches to an feedforward-feedback control policy with a linear gain K. The feedforward term, however, dominates the behavior of the control policy; this is discussed next. The overall switched control architecture is given by Algorithm 2.

Algorithm 2 Switched control algorithm

Inputs
  $(\boldsymbol{z}^*, \boldsymbol{u}^*)$, $\hat{z}_t$
while while $t < t_f$ **do**
  $y \leftarrow$ measurement, $\hat{z}_t \leftarrow \text{ekf}(\hat{x}, y, u)$
  $D_t \leftarrow \min_i \|z_i^* - \hat{z}_t\|$, $k \leftarrow \arg\min_i \|z_i^* - \hat{z}_t\|$
  $\bar{\boldsymbol{z}} \leftarrow \begin{bmatrix} z_k^*, & z_{k+1}^* & \dots & z_{k+N}^* \end{bmatrix}$
  $\bar{\boldsymbol{u}} \leftarrow \begin{bmatrix} u_k^*, & u_{k+1}^* & \dots & u_{k+N}^* \end{bmatrix}$
  $[\text{status}, \boldsymbol{u}^{\text{MPC}}] \leftarrow \text{solveMPC}(\bar{\boldsymbol{z}}, \bar{\boldsymbol{u}}, \bar{z}_t, D_t)$,
  if status = feasible **then**
    $u \leftarrow u_0^{\text{MPC}}$
  else
    $u \leftarrow \bar{u}_0 + K_P(\bar{z}_0 - \hat{z}_t)$
  end if
end while

The control architecture ultimately attempts to track the drift. At each time step, the algorithm gets a new measurement y and estimates its state using an extended kalman filter (EKF), next localizes itself with respect to the reference trajectory $(\boldsymbol{z}^*, \boldsymbol{u}^*)$, then extracts $N+1$ reference states and inputs to form the target trajectory $(\bar{\boldsymbol{z}}, \bar{\boldsymbol{u}})$ for the MPC. Lastly, the controller attempts to solve the MPC, and if it fails, it switches to a linear feedforward-feedback control policy. The vehicle's initial position is as close as possible to the first state on the reference trajectory (z_0^*, u_0^*).

As noted earlier, the rear wheels lock before the front wheels do during the maneuver. When this happens, actuation is limited to only steering. Even with one less degree of actuation, the yaw angle ψ can still be controlled through the steering angle δ. We use a proportional controller with a constant gain matrix $K \in \mathbb{R}^{p \times n}$, where p is number of control inputs and n is the dimension of the state vector. We set K to be almost entirely zero, except for one diagonal entry $k_{\Delta\psi}$, which multiplies the yaw angle error. We tune the value of $k_{\Delta\psi}$ to improve tracking of the target yaw angle, but limit its magnitude so that the feedforward term $\bar{u}_0$ dominates the control policy. For a simple proportional controller, setting $k_{\Delta\psi}$ too large would result in oscillatory motion about the target trajectory. Other control techniques are possible, but a proportional controller is a simple, model-free method to enhance tracking during the drift maneuver.

3.3 State Estimation

For state estimation, we apply an Extended Kalman Filter (EKF, see Fujii (2013)) that uses the vehicle model in (2), with the tire model in (3). We model process noise $w[k]$ and measurement noise $v[k]$ at time step k as Gaussian with zero mean and covariance Q_k and R_k , respectively.

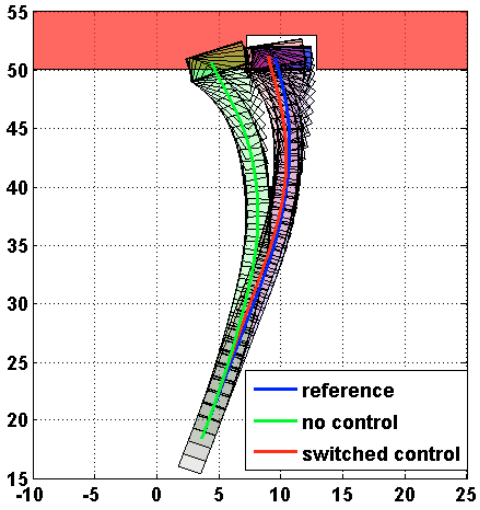

Fig. 3. Drift parking maneuver tracking

4. SIMULATION

The authors simulated the performance of the proposed control algorithm for three different aggressive maneuvers: a handbrake parking slide, a handbrake turn and a gymkhana turn. Reference trajectories were obtained using our path planner. To test the robustness of the approach we add some model uncertainty.

Maneuver	Description
(a) Handbrake parking slide	parallel park inside a tight parking spot (see GuinnessWorldRecords (2012))
(b) Handbrake turn	180 degree turn in a narrow enclosed space (see Gear (2006), Wikipedia (2004))
(c) Gymkhana turn	270 degree turn inside a tight maneuvering space (see DCshoesFILM (2010))

The authors validated using MATLAB/Simulink and Carsim, a high fidelity vehicle simulation software. For Figures, 3, 4 and 5, blue indicates the reference trajectory, red shows the trajectory using the proposed control algorithm, and green shows the path followed by the car without using any control, i.e. merely executing the control inputs in the open loop fashion. Obstacles in all figures are solid red. For maneuver (a) we use steering angle, acceleration and handbrake as control inputs. Maneuvers (b) and (c) also require gear shifting in order to be realized in CarSim.

From each of the plots, we observe that simply following open-loop commands from a single successful trial is not sufficient to execute a maneuver successful. It can be observed that the trajectories without feedback may collide into solid objects.

5. EXPERIMENTAL RESULTS

The authors use a low-cost 1/10 scale RC vehicle platform, called Berkeley Autonomous Race Car (BARC)(see Gonzales (2015)), to validate the control architecture. The RC

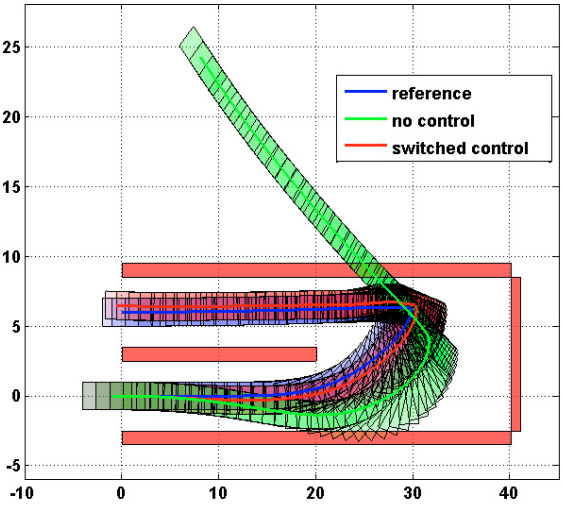

Fig. 4. Handbrake turn maneuver tracking

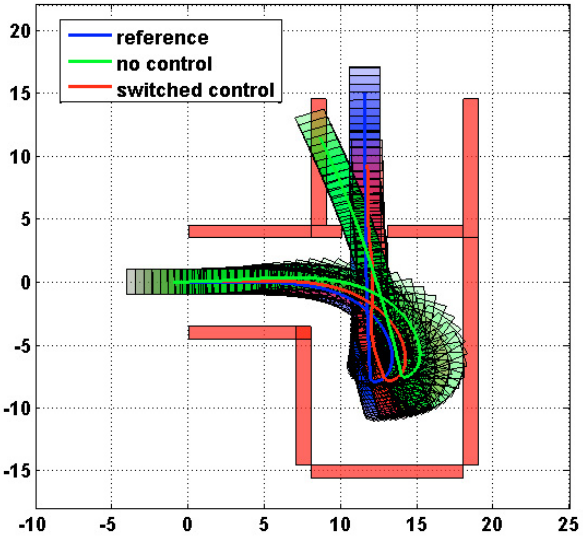

Fig. 5. Gymkhana maneuver tracking

vehicle is shown in Figure 6. The authors focused only on testing the handbrake parking slide maneuver due to space constraints and localization accuracy from the available on-board sensors, namely an inertial measurement unit and encoders. The other two maneuvers, the handbrake turn and gymkhana, require very accurate vehicle localization after the sliding phase. The handbrake slide park, on the other hand, comes to a full stop at the end of the maneuver.

All experiments were conducted in an empty indoor room with the reference trajectory obtained from a demonstration. The tracking performance with and without the proposed architecture is shown in the Figure 7 and Figure 8. Using just a pre-recorded open loop sequence of commands, the vehicle not only fails to track the reference trajectory, but follows different paths with each trial, even under static environmental conditions. Also, a strong bias in the steering system can be observed from figure 8. The proposed algorithm had an average tracking error of around (0.2 ± 0.1) m, while the pure open loop one has an

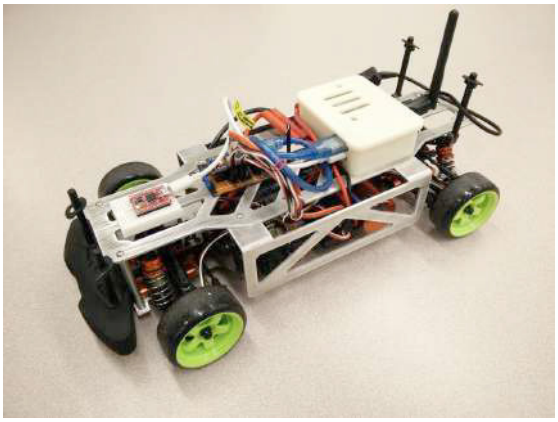

Fig. 6. Berkeley Autonomous Race Car (BARC)

average error of around (1.2 ± 0.6) m. Since the authors used only on-board sensors for state estimation (e.g. imu, encoders), the plotted trajectories deviate from the ground truth. The parking spot, constructed as a narrow space between two cardboard boxes was fixed.

We tested both accuracy and repeatability of our algorithm by trying to track the same trajectory multiple times. Experimental results show that the proposed algorithm is applicable in practice with using only low-cost on-board sensors. Video of the maneuver can be found in Gonzales (2015) and the frame-by-frame breakdown is shown in Figure 9.

6. CONCLUSION

In this paper, the authors propose a switched control architecture for executing extreme maneuvers for road vehicles. To the best of our knowledge, this is the first work that uses only on-board sensors for tracking extreme trajectories. The effectiveness of the approach is shown in a simulation with high fidelity model and experimentally. Future work includes enhancing the tracking performance using sensors that provide global coordinates, as well as conducting experiments on a full size vehicle.

REFERENCES

Atkeson, C.G. and Schaal, S. (1997). Learning tasks from a single demonstration. In *International Conference on Robotics and Automation*.

Camacho, E.F. and Alba, C.B. (2013). *Model predictive control*. Springer Science & Business Media.

Chakraborty, I., Tsiotras, P., and Lu, J. (2011). Vehicle posture control through aggressive maneuvering for mitigation of t-bone collisions. In *IEEE Conference on Decision and Control*.

DCshoesFILM (2010). Dc shoes: Ken block's gymkhana three. URL `http://www.youtube.com/watch?v=4TshFWSsrn8&t=4m3s`.

Ding, J. (2012). Reachability-based controller design for switched nonlinear systems. URL `https://inst.eecs.berkeley.edu/ ee291e/sp12/handouts/SwitchedSystemControl_04182012.ppt`.

Fujii, K. (2013). Extended kalman filter. *Refernce Manual*.

Gallestey, E. and Al-Hokayem, P. (2014). Nonlinear model predictive control. URL `http://control.ee.ethz.ch/ãpnoco/Lectures`

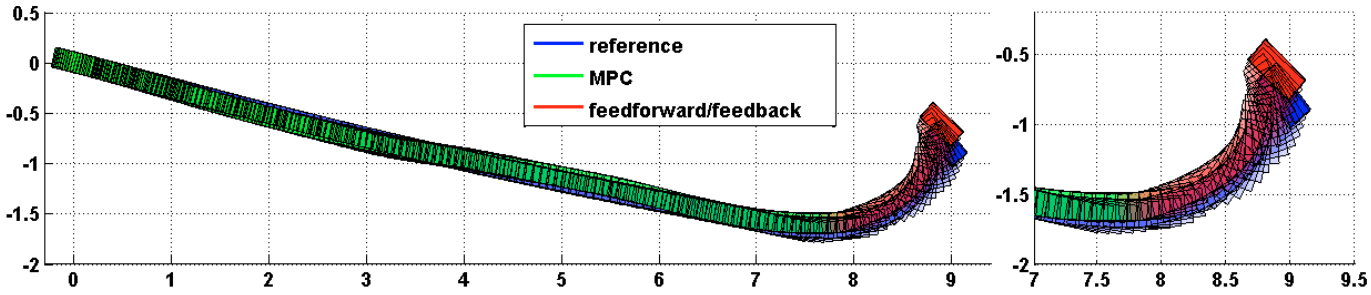

Fig. 7. Desired and actual trajectory for parking slide with switched control strategy. Parts of the trajectory are colored according to the controller that was active at the time

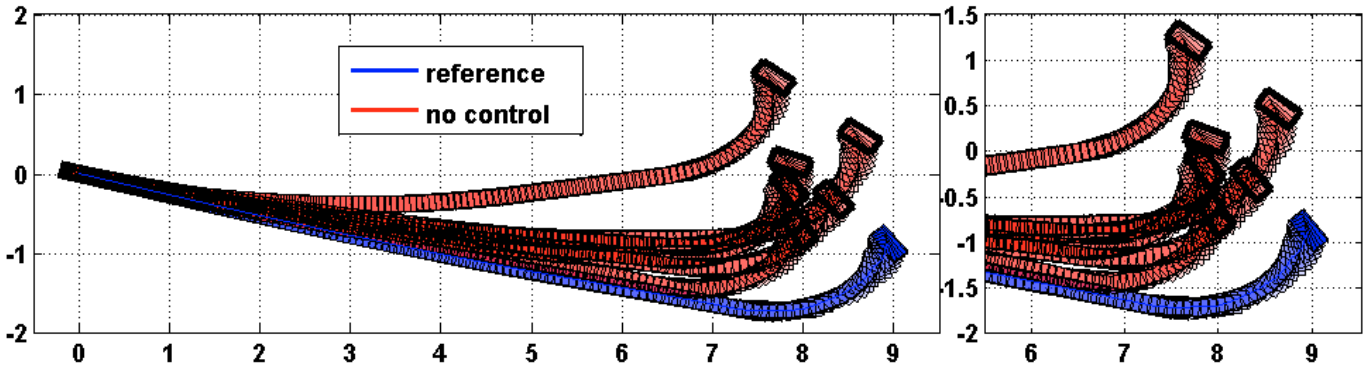

Fig. 8. Desired and actual trajectory for parking slide without any control. We can observe strong bias in the yaw angle that our controller was able to suppress

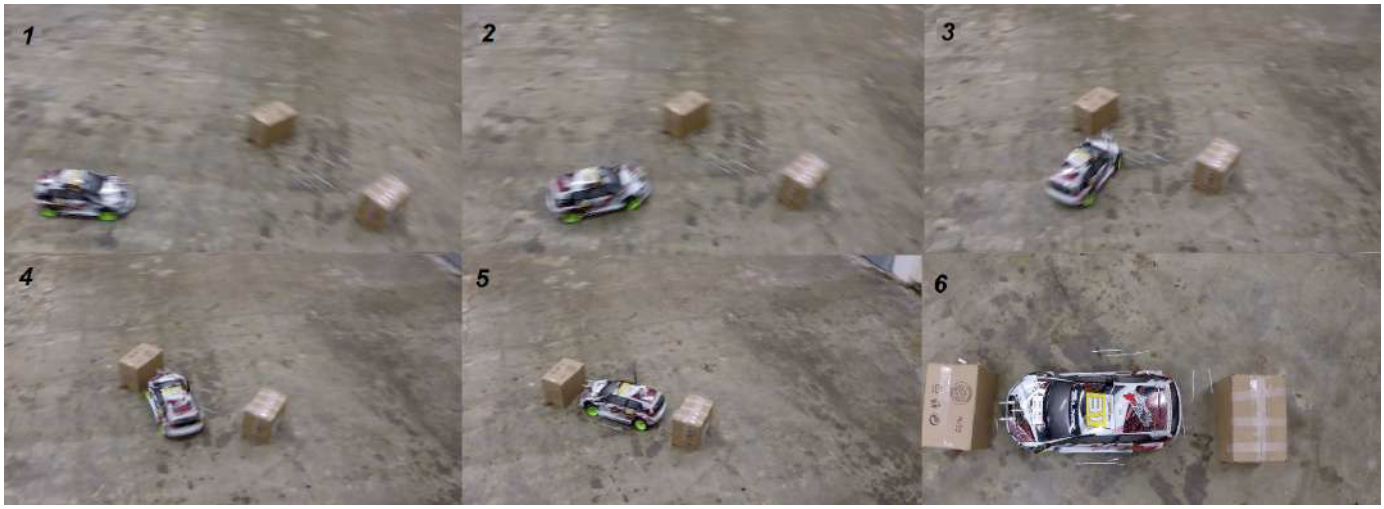

Fig. 9. Frame breakdown of drifting park maneuver

2014/08Nonlinear%20Model%20Predictive%20Control.pdf.

Gear, F. (2006). How to handbrake turn. URL https://www.youtube.com/watch?v=NlaQGCr0dfI.

Gonzales, J.M. (2015). Berkeley autonomus race car. URL http://www.barcproject.com/.

GuinnessWorldRecords (2012). Tightest parallel parking record. URL https://www.youtube.com/watch?v=q3BGkOKVMUU.

Hindiyeh, R. (2013). *Dynamics and control of drifting in automobiles*. Ph.D. thesis, Stanford University.

Hodgins, J.K. and Raibert, M.H. (1990). Biped gymnastics. *The International Journal of Robotics Research*.

Kolter, J.Z., Plagemann, C., Jackson, D.T., Ng, A.Y., and Thrun, S. (2010). A probabilistic approach to mixed open-loop and closed-loop control, with application to extreme autonomous driving. In *International Conference on Robotics and Automation*.

Liberzon, D. (2012). *Switching in systems and control*. Springer Science & Business Media.

Pacejka, H. (2005). *Tire and vehicle dynamics*. Elsevier.

Rosolia, U. and Borrelli, F. (2016). Learning model predictive control for iterative tasks. *arXiv preprint arXiv:1609.01387*.

Wikipedia (2004). Handbrake turn. URL https://en.wikipedia.org/wiki/Handbrake_turn.