Constrained Iterative LQR

Teerthaa Parakh

1 Introduction

As today's technology is becoming more and more automated, autonomous driving presents a huge challenge both for industries and researchers, the reason being that the automation is easily capable of proving fatal in even little crowded environments. As emergency situations in these environments are inevitable, any solution proposed should consider the possible sources of uncertainties and respond to them in real time. This makes motion in such scenarios to be constrained in a variety of ways and their mathematical representation is often complex to deal with in real time. Trajectory generation for autonomous driving is a problem in the spatiotemporal domain and requires a motion planner which can efficiently process complex collision avoidance constraints along with the various vehicle dependent parameters of motion; generate results in real time to handle emergency situations like an impending collision; and can produce trajectory of considerable length in the spatiotemporal domain. [1] proposes a Constrained Iterative Linear Quadratic regulator to solve the problem of constrained motion efficiently. ILQR is shown ([2],[3]) to be efficient in generating trajectories in real time and the CILQR is an extension to solve a more general problem using barrier function.

2 Discrete-time Finite-horizon Motion Planning Problem

$$x^*, u^* = \underset{x,u}{\operatorname{argmin}}\, [\phi(x_N) + \sum_{k=0}^{N-1} L^k(x_k, u_k)] \tag{1a}$$

$$s.t.\ x_{k+1} = f^k(x_k, u_k),\ k = 0, 1, ..., N-1 \tag{1b}$$

$$x_0 = x_{start} \tag{1c}$$

$$g^k(x_k, u_k) < 0,\ k = 0, 1, ..., N-1 \tag{1d}$$

$$g^N(x_N) < 0 \tag{1e}$$

Equation (1a) is the objective function.
Equation (1b) is the state dynamic constraint.
Equation (1c) is start constraint.
Equation (1d) is inequality constraint.
Equation (1e) in inequality constraint at the last step.
Assumption made in this model-

- The system dynamics constraints are the only equality constraints. Rest Equality constraints are eliminated.
- All the functions defined in this model are assumed to have continuous first and second order derivatives.

3 Iterative LQR in Unconstrained Motion Planning Problem

The nonlinear model is given as:

$$x^*, u^* = \underset{x,u}{\operatorname{argmin}}\,[\phi(x_N) + \sum_{k=0}^{N-1} L^k(x_k, u_k)] \tag{2a}$$

$$s.t.\ x_{k+1} = f^k(x_k, u_k),\ k = 0, 1, ..., N-1 \tag{2b}$$

$$x_0 = x_{start} \tag{2c}$$

The Bellman Equation is:

$$V^k(x_k) = \min_{u_k}\,[L^k(x_k, u_k) + V^{k+1}(f^k(x_k, u_k))] \tag{3}$$

Where $V^k(x_k)$ is the minimum cost to go function from x_k.
The algorithm to solve Equation (2a):

Step 1: $(\bar{x}_k, \bar{u}_k)$=Initial feasible trajectory given as nominal trajectory.

Step 2: (Backward Pass) Starting from N^{th} step we have $V^N(x_N) = \phi(x_N)$. Then going from $(N-1)^{th}$ step backward in time we have the permutation term:

$$\begin{aligned} P^k(\delta x_k, \delta u_k) = L^k(\bar{x}_k + \delta x_k, \bar{u}_k + \delta u_k) - L^k(\bar{x}_k, \bar{u}_k) \\ + V^{k+1}(f^k(\bar{x}_k + \delta x_k, \bar{u}_k + \delta u_k)) - V^{k+1}(f^k(\bar{x}_k, \bar{u}_k)) \end{aligned} \tag{4}$$

$P^k(\delta x_k, \delta u_k)$ approximated by its seacond order Taylor expansion we have:

$$P^k(\delta x_k, \delta u_k) = \frac{1}{2}\begin{bmatrix} 1 \\ \delta x_k \\ \delta u_k \end{bmatrix}^T \begin{bmatrix} 0 & (P_x^k)^T & (P_u^k)^T \\ (P_x^k) & (P_{xx}^k)^T & (P_{xu}^k)^T \\ (P_u^k) & (P_{ux}^k)^T & (P_{uu}^k)^T \end{bmatrix} \begin{bmatrix} 1 \\ \delta x_k \\ \delta u_k \end{bmatrix} \tag{5}$$

where

$$P_x^k = L_x^k + (f_x^k)^T V_x^{k+1} \tag{6a}$$

$$P_u^k = L_u^k + (f_u^k)^T V_x^{k+1} \tag{6b}$$

$$P_{xx}^k = L_{xx}^k + (f_x^k)^T V_{xx}^{k+1} f_x^k + V_x^{k+1} f_{xx}^k \tag{6c}$$

$$P_{uu}^k = L_{uu}^k + (f_u^k)^T V_{xx}^{k+1} f_u^k + V_x^{k+1} f_{uu}^k \tag{6d}$$

$$P_{ux}^k = L_{ux}^k + (f_u^k)^T V_{xx}^{k+1} f_x^k + V_x^{k+1} f_{ux}^k \tag{6e}$$

Then the optimal control strategy is:

$$\delta u_k^* = \underset{\delta u_k}{\operatorname{argmin}}\, P^k(\delta x_k, \delta u_k) = -(P_{uu}^k)^{-1}(P_u^k + P_{ux}^k \delta x_k) = q^k + Q^k \delta x_k \tag{7}$$

Where $q^k = -(P_{uu}^k)^{-1}(P_u^k)$ and $Q^k = -(P_{uu}^k)^{-1}(P_{ux}^k)$
Putting this into expansion of P we get

$$V_x^k = P_x^k - P_u^k (P_{uu}^k)^{-1} P_u^k \tag{8}$$

$$V_{xx}^k = P_{xx}^k - P_{xu}^k (P_{uu}^k)^{-1} P_{ux}^k \tag{9}$$

We store values of q^k and Q^k for each time step.

Step 3: Forward simulate the dynamic equations

$$x_0 = x_{start} \tag{10a}$$

$$u_k = \bar{u}_k + q^k + Q^k(x_k - \bar{x}_k) \tag{10b}$$

$$x_{k+1} = f^k(x_k, u_k) \tag{10c}$$

Step 4: Now $(\bar{x}_k, \bar{u}_k) =$ states, control input obtained from Step 3.

Iteratively applying Step 2, Step 3 and Step 4 the solutions converge to optimum trajectory.

4 Constrained Iterative LQR

The inequalities are introduced as penalties in the objective function. The penalty function is:

$$b(g(x,u)) = -\frac{1}{t} log(-g(x,u)) \tag{11}$$

This function has the following useful properties:

- It ensures hard constraints.
- As we increase parameter 't' , it converges to indicator function.

Hence, the algorithm for Constrained ILQR is:

Step 1: Set the parameter 't' of the logarithmic barrier function and the nominal trajectory.

Step 2: Transform the constraints Equation (1d) and Equation (1e) and add it to Equation (2a).

Step 3: Perform ILQR.

Step 4: Set the nominal trajectory equal to trajectory obtained in Step 3 and update parameter 't' to αt, where $\alpha > 1$. Go to Step2 and repeat until convergence.

5 CILQR for Autonomous Driving Motion Planning

$$x^*, u^* = \underset{x,u}{\operatorname{argmin}} \, [\phi(x_N) + \sum_{k=0}^{N-1} L^k(x_k, u_k)] \tag{12a}$$

$$s.t. \; x_{k+1} = f^k(x_k, u_k), \; k = 0, 1, ..., N-1 \tag{12b}$$

$$x_0 = x_{start} \tag{12c}$$

$$d(x_k, O_j^k) > 0, \; k = 0, 1, ..., N-1 \; j = 1, 2, ..., m \tag{12d}$$

$$u < u_k < \bar{u}, \; k = 1, 2, ..., N-1 \tag{12e}$$

Here Equation (12d) is obstacle avoidance constraint. And Equation (12e) is a control constraint. Vehicle Model used here is:

$$l = v_0 T_r + \frac{1}{2} a T_r^2 \tag{13a}$$

$$v_1 = v_0 + a T_r \tag{13b}$$

$$\theta_1 = \theta_0 + \int_0^l \frac{\tan \delta}{L} \, ds \tag{13c}$$

$$x_1 = x_0 + \int_0^l \cos(\theta_0 + \frac{\tan \delta}{L} s) \, ds \tag{13d}$$

$$y_1 = y_0 + \int_0^l \sin(\theta_0 + \frac{\tan \delta}{L} s) \, ds \tag{13e}$$

$$\tag{13f}$$

Objective Function:

(a) *Acceleration* : To minimize fuel consumption, we try to minimize acceleration.

$$c_k^{acc} = w_{acc} u_k^T \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} u_k \tag{14}$$

(b) *Steering Angle* : To increase passenger comfort , we try to minimize the angle.

$$c_k^{steer} = w_{steer} u_k^T \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} u_k \tag{15}$$

(c) *Velocity Tracking* : $w(v - v_{ref})^T (v - v_{ref})$

(d) *Acceleration constraint* : Introduced as barrier function, $a_{low} = -9.8, a_{high} = 9.8$

$$a_{low} \leq u_k^T \begin{bmatrix} 1 \\ 0 \end{bmatrix} \leq a_{high} \tag{16}$$

(e) *Steering constraint* : Introduced as barrier function, $\bar{s} = .5$

$$-\bar{s} \leq u_k^T \begin{bmatrix} 0 \\ 1 \end{bmatrix} \leq \bar{s} \tag{17}$$

(f) *Obstacle avoidance constraint* : Introduced as barrier function and obstacles are assumed to be of rectangular shape.

6 Results

There are 2 cases are considered here - Static obstacle avoidance and Dynamic obstacle avoidance. In both cases the sampling time $T_s = .2s$ and the horizon is $N = 45$, so the planning is for 9s ahead.

6.1 Static Obstacle Avoidance

6.1.1 CILQR

There are three obstacles (red coloured in figure 2) : one obstacle is at $(20, 0)$, second at $(30, 0)$, and the last one at $(40, 0)$. Initial velocity of ego vehicle is $0m/s$ and desired velocity is $8m/s$. The desired reference line is $y = 0$.
In this case average number of iteration is 17, time taken per iteration is $.02s$. So total time taken is $0.26s$.

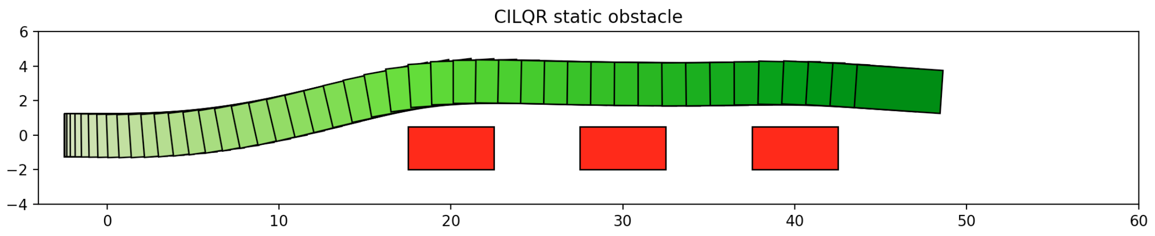


Figure 1: trajectory of the ego vehicle

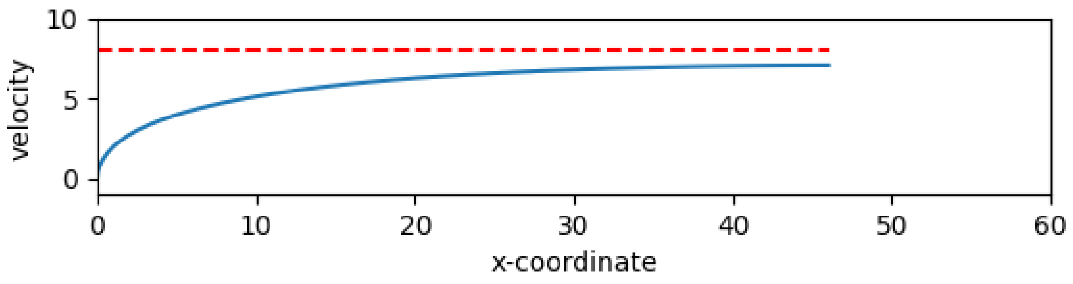


Figure 2: location in x-coordinate vs velocity

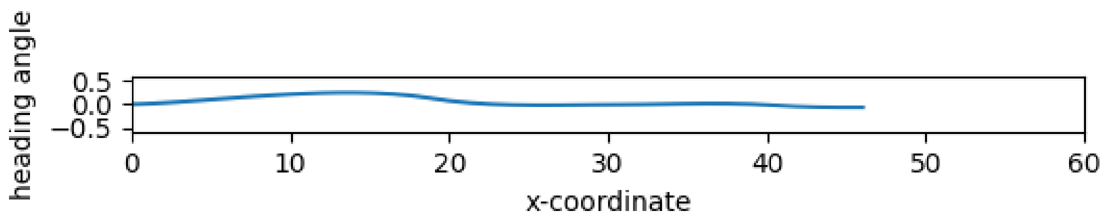


Figure 3: location in x-coordinate vs angle

6.1.2 SQP

In this case average number of iteration is 81, time taken per iteration is $.28s$. So total time taken is $22.94s$.

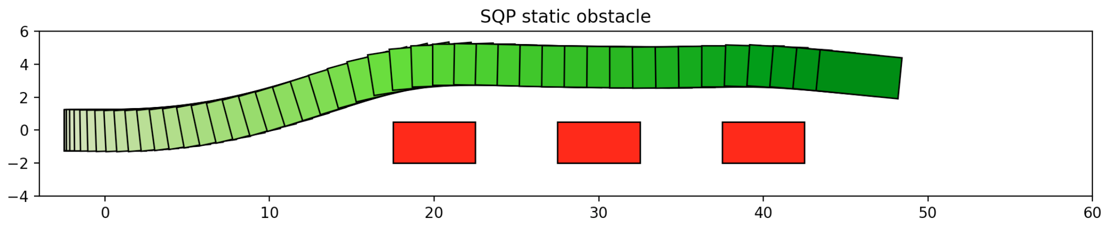


Figure 4: trajectory of the ego vehicle

6.2 Dynamic Obstacle Avoidance

6.2.1 CILQR

Depending upon the weights in objective function, we can either observe a lane changing behavior or an overtaking behavior.
Case - 1: Ego vehicle starts at $(0, 0)$ with velocity 0 m/s, the desired trajectory is $y = 0$ the target velocity= $8m/s$, and $N = 48$.
In this case there are three obstacles - first obstacle starts at position $(0, 4.5)$ with constant velocity $3m/s$, second obstacle starts at $(22.5, 0)$ with velocity $2m/s$ and third obstacle starts at $(52.5, 4)$ with velocity $3m/s$.
In this case average number of iteration is 18, time taken per iteration is $.01s$. So total time taken is $0.23s$.

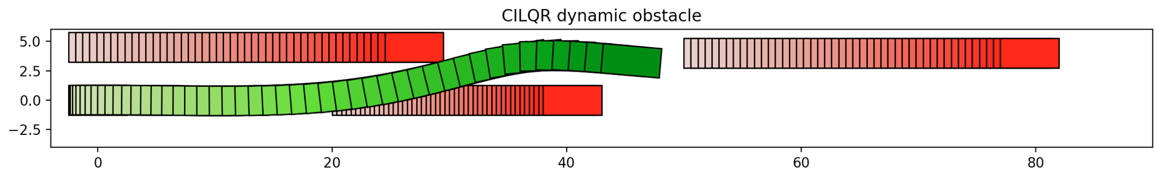


Figure 5: trajectory of the ego vehicle

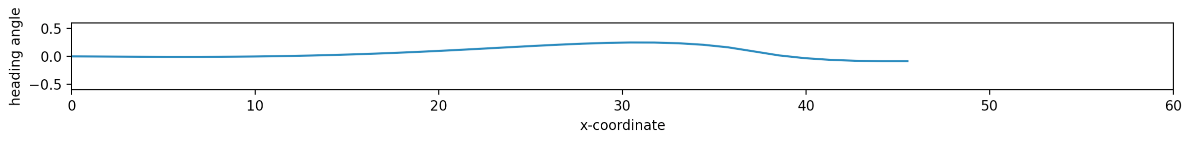


Figure 6: location in x-coordinate vs angle

Case - 2: If we change the horizon length that is $N = 48$ and increase the weight for steering angle, we get the following trajectory:

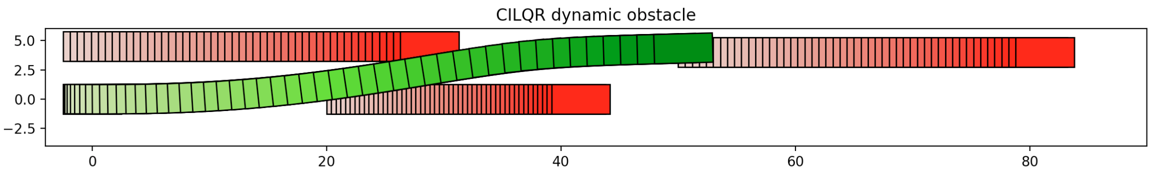


Figure 7: trajectory of the ego vehicle

Case - 3: In this case overtaking behavior is observed. Ego vehicle starts at $(0, 0)$ with velocity $0m/s$, desired velocity is $8m/s$, the desired trajectory is $y = 0$ and $N = 55$.
In this case there are two obstacles, one starting at $(70, 4.5)$ and moving in left direction with velocity $5m/s$ and another starting at $(20, 0)$ moving in right direction with velocity $3m/s$.

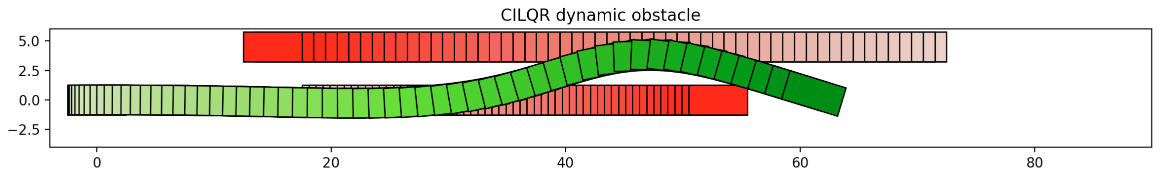


Figure 8: trajectory of the ego vehicle

6.2.2 SQP

For **case - 1**, the average number of iterations SQP took to solve this optimization problem is 115 and time taken per iteration is $.49s$. So total time taken is $56.51s$.

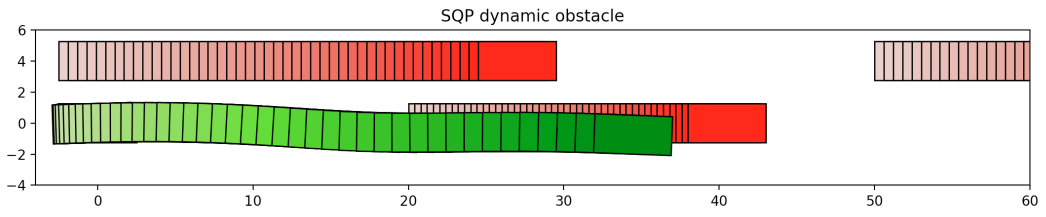
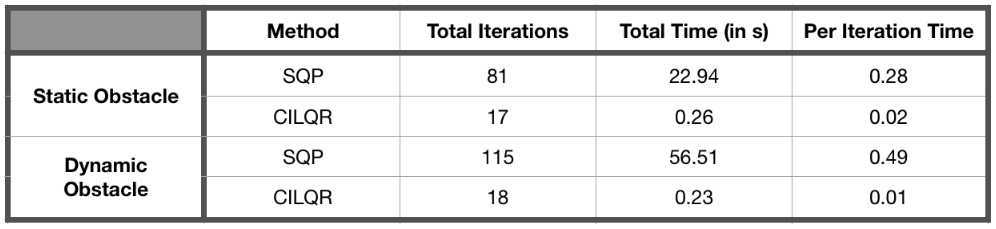


Figure 9: trajectory of the ego vehicle

The table below concludes the performance of SQP and CILQR.



	Method	Total Iterations	Total Time (in s)	Per Iteration Time
Static Obstacle	SQP	81	22.94	0.28
	CILQR	17	0.26	0.02
Dynamic Obstacle	SQP	115	56.51	0.49
	CILQR	18	0.23	0.01

Figure 10: Comparison of CILQR and SQP method

7 Conclusion and Future Work

In the [1], a Constrained ILQR was proposed to solve nonlinear dynamics with non-convex constraints. An autonomous driving motion planning problem was solved using CILQR . It was validated that CILQR is approximately 100 times faster than SQP solver. The author of paper [1] didn't take into account the sudden unexpected changes in the environment; it is assumed that other vehicles are moving with constant velocity in a straight line. However, in real world scenarios there can be sudden changes in the path of other vehicles. Modeling errors were not taken into account in [1]. To make this method more robust, estimation and feedback term for those errors can be added to state, which is left for future work.

References

[1] Chen,J. , Zhan,W. & Tomizuka,M. (2019) Autonomous Driving Motion Planning With Constrained Iterative LQR *IEEE Transactions on Intelligent Vehicles, Intelligent Vehicles, vol. 4, no. 2, pp. 244–254*.

[2] Li,W. & Torodov,E. (2004) Iterative linear quadratic regulator design for nonlinear biological movement systems *International Conference on Informatics in Control, Automation and Robotics (ICINCO), 2004, pp. 222–229.*.

[3] Li,W. & Torodov,E. (2005) A generalized iterative LQG method for locally optimal feedback control of constrained nonlinear stochastic systems *American Control Conference (ACC)*.