ECE 726 Project Report – A Study on Finite-Horizon LQT, Extended Kalman Filter and Integral Reinforcement Learning

Mohammad Sareeb Hakak Sreechakra Vasudeva Raju Rachavelpula

Introduction- The aim of the project is to develop a Linear Quadratic Tracking (LQT) control for a quadcopter for optimal trajectory tracking through the finite horizon tracking LQR (FHLQT) and the Integral Reinforcement Learning. An extended Kalman Filter (EKF) and a Linear Quadratic Gaussian (LQG) were developed as the observers to track the states in real world noisy conditions.
Initially the dynamics of the quadcopter were formulated, and the set of equilibrium points were found around which the model was linearized. For the tracking problem, FHLQT was developed by solving the Differential Riccati Equation. It was observed that FHLQT can also be developed by augmenting the reference trajectory command generator dynamics to the original system dynamics and solving the DRE for the augmented system.
To estimate and observe the states, a Linear Quadratic Gaussian was developed by solving the Algebraic Riccati Equation. To improve on the performance of the observer, an extended Kalman Filter was developed for the nonlinear model to estimate the states in noisy conditions.
Treating our system as partially unknown, the Integral Reinforcement Learning is implemented by developing the LQT Algebraic Riccati equation and finally comparing the gains to that of the Infinite Horizon tracking LQR.

Modeling

For the model, we have 12 states: the positions, velocities and accelerations in x, y, z direction, the angular velocities in x, y, z direction and the Euler angles of yaw, pitch and roll. The equilibrium points of the system were found by solving the 12 state equations simultaneously and are constant values for x, y, z and yaw, the rest being zero. The system was linearized around these points. The system has 4 inputs relating to thrust, roll, pitch and yaw. These inputs control the coupled system. The input for thrust controls the parameters in z, roll inputs the y-direction, pitch control the x-direction and yaw as a separate input.

Finite Horizon Tracking LQR

The FHLQT problem is formulated as optimizing

$$J = 0.5\left(y_r(t_f) - Cx(t_f)\right)' F \left(y_r(t_f) - Cx(t_f)\right) + 0.5\int_0^{t_f} (y_r - Cx)'Q(y_r - Cx) + u'Ru\, dt$$

implemented by solving the Differential Riccati equation backward in time

$$\frac{dK}{dt} + KA + A'K - KBR^{-1}B'S + C'QC = 0$$

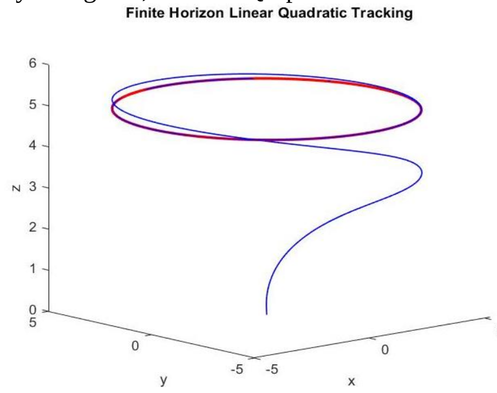
$$\frac{dS}{dt} + A'S - KBR^{-1}B'S - C'Qy_r = 0$$

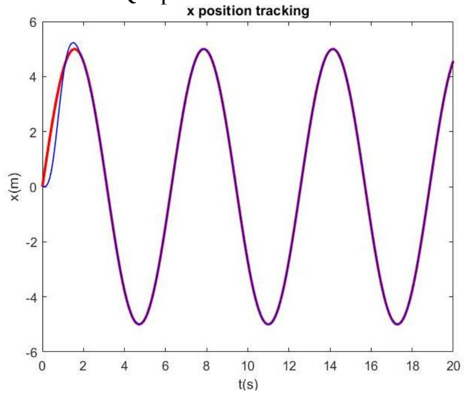
$$u = -BR^{-1}B'Kx - BR^{-1}B'S$$

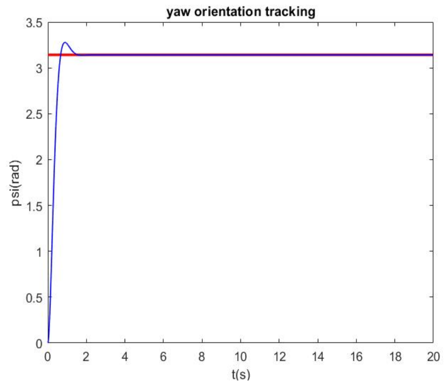
With the terminal boundary conditions as $K(t_f) = C'FC$ and $S(t_f) = -C'Fy_r(t_f)$. However, as shown in ref. [3] the FHLQT was also implemented by augmenting the trajectory command generator with the original system as $A_{aug} = \begin{bmatrix} A & 0 \\ 0 & D \end{bmatrix}$, $B_{aug} = \begin{bmatrix} B \\ 0 \end{bmatrix}$, $C_{aug} = \begin{bmatrix} C & -E \end{bmatrix}$ where the command the generator dynamics are given by:

$$\dot{x_r} = Dx_r \,;\; y_r = Ex_r$$

By doing this, the FHLQT problem can be solved as a FHLQR problem.







Extended Kalman Filter

To observe the states of the system, we designed the optimal observer, the Linear Quadratic Gaussian (LQG) for the system. The LQG was designed by solving the Observer Algebraic Riccati equation which is the dual of the Controller Riccati equation. The LQG is implemented on top of the finite horizon tracking LQR.
Real-time systems like a quadcopter needs state estimation based on the system model and environment conditions like noise to operate as dead reckoning can lead to failures. For the estimation of the states, we developed a Kalman filter which gives the estimate of the states by calculating a joint probability distribution based on the model and the measurements of the system. For the

implementation, we are using the discrete version of our model. In brief, the algorithm gives an optimal weighting factor for each consequent state based on the previous known state of the model and the measurements as given: $\widehat{x_K} = (I - KC)\widehat{x_{K-1}} + Kz_K$. The algorithm is divided into two parts: The time update and the measurement update.

Time Update:

$$x_{kk1}[k] = Ax_{kk}[k-1] + Bu[k]$$

$$P_{kk1}[k] = APA^T + Q$$

where x_{kk1} is the prior estimate, x_{kk} is the current estimate and time steps are k and k-1 as in brackets. P_{kk1} is the prior covariance and P is the current one. For k=0 we take the initial estimates.

Measurement Update:

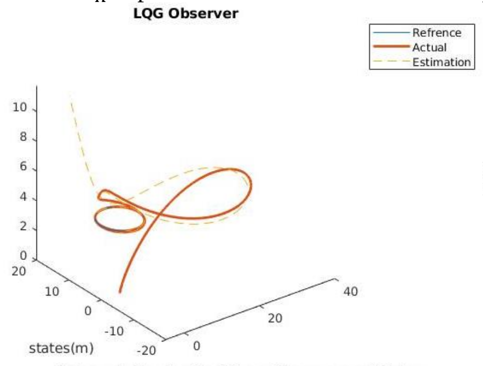
$$K = P_{kk1}C^T[CP_{kk1}C^T + R]^{-1}$$

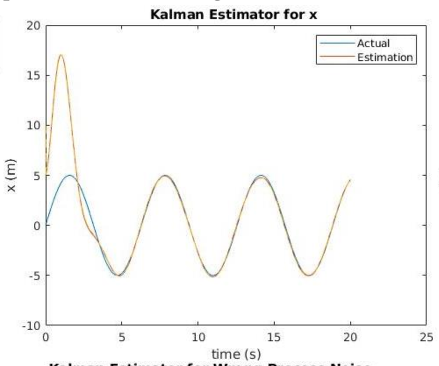
$$x_{kk} = x_{kk1} + K[z - Cx_{kk1}]$$

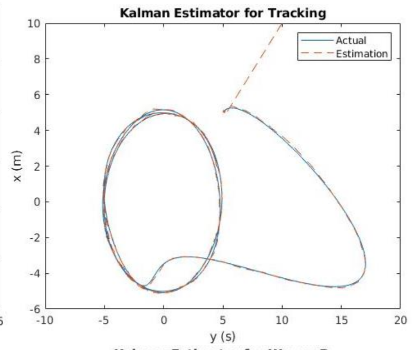
$$P = [I - KC]P_{kk1}$$

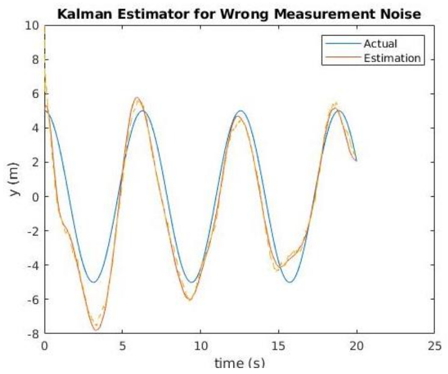
where $z = Cx + w_2$. This gives the estimates which are then further fed into the time update.

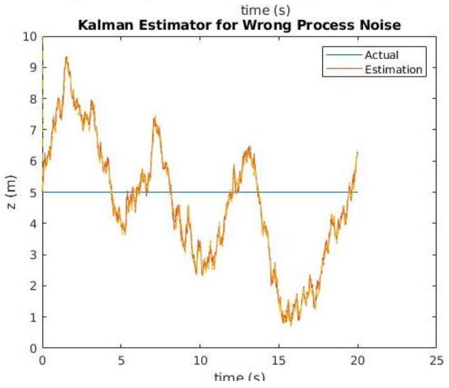
Where $\widehat{x_K}$ represents the state at Kth time step, K is the Kalman gain, and z_K is the measurement.

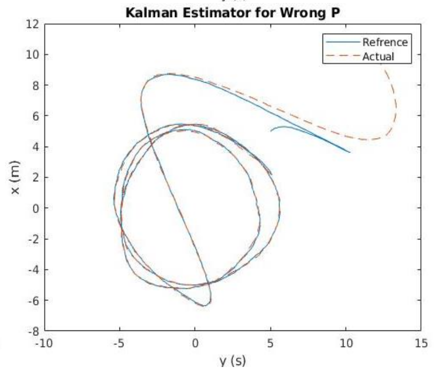












Integral Reinforcement Learning

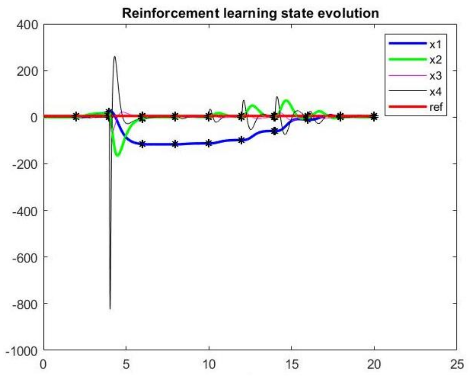
One of the problems in controlling dynamical systems is the non-availability of enough system information. This leads to modeling errors and hence a control law developed on such a model may not be robust to disturbances. However, an optimal control law can be established without the knowledge of the system dynamics or the trajectory command generator dynamics using reinforcement learning techniques. This section is an implementation of the Integral reinforcement learning algorithm proposed in ref. [1]. The algorithm assumes that open-loop system dynamics (A) and the command generator dynamics (D) unknown, however the system response to the control inputs(B) are known.

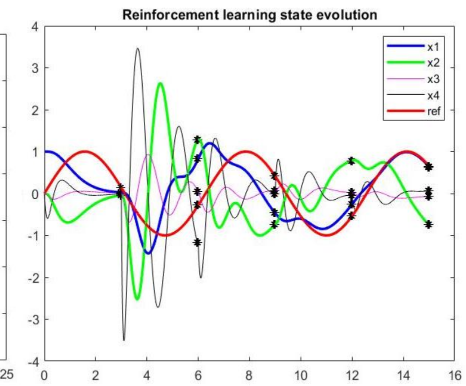
A recurring problem in the RL is the curse of dimensionality, where the transition matrix grows exponentially with the cardinality of the state space, thereby making the system infeasible to solve. This especially affects dynamical systems, which typically have 6 or more degrees of freedom that must be tracked through the states. To counter this, we decoupled our system into 4 parts- 4 states affecting x, 4 states affecting y, 2 states for z, and 2 states for yaw. All these 4 parts are decoupled from one another and each are controlled by one control input. So, we were able to solve them separately and track states from 4 parts separately.
The algorithm's policy iteration scheme is as follows:
Initialization: Start with an admissible control input $u^0 = F_1^0 X$
Policy evaluation: Given a control policy u^i, solve for the elements of K by reducing the mean-squared error from the equation

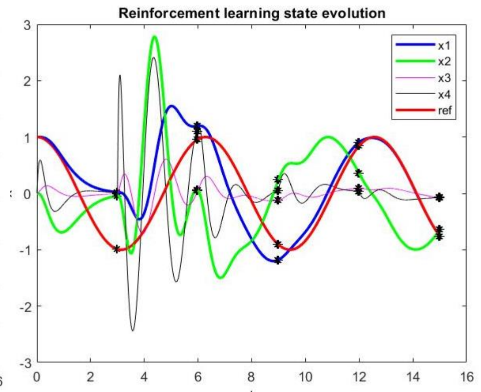
$$X(t)'\,P^i X(t) \;-\; e^{-\gamma \Delta t}\, X(t+\Delta t)'\,P^i X\,(t+\Delta t) \;=\; \int_t^{t+\Delta t} e^{-\gamma(\tau - t)}\,(X(t)'\,C_{aug}'\,Q\,C_{aug}X(t) \;+\; u^{i\prime}Ru^i)\,d\tau$$

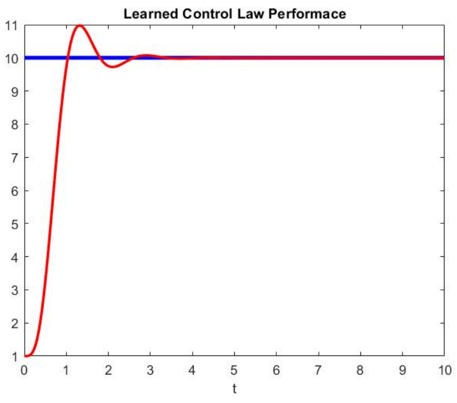
Policy update: $u^{i+1} = -R^{-1}B'_{aug}P^iX$

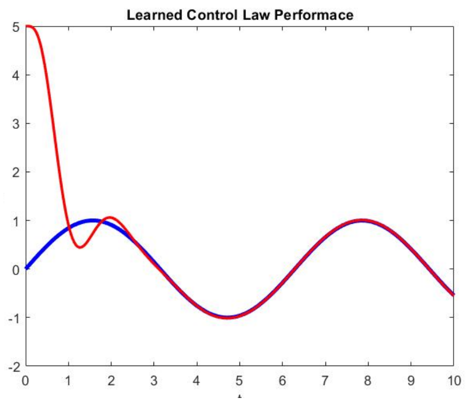
The $\gamma > 0$ is a discount factor introduced to make the augmented state ARE solvable. This is because the augmented state space with the command generator dynamics is uncontrollable.

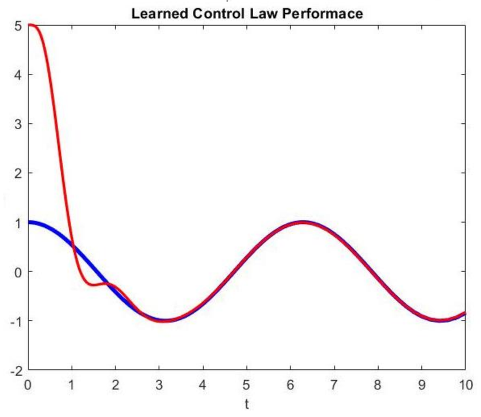












Conclusions

It was observed that an FHLQT can be implemented by augmenting the trajectory command generator dynamics to the original system dynamics. However, the same cannot be done for an infinite horizon LQT because the ARE for the augmented state space model becomes unsolvable due to the uncontrollability of the command generator states. A discount factor γ had to be defined to make the ARE solvable. This does not happen in the case of the FHLQT because the terminal conditions on the elements of augmented K matrix have been defined and hence a unique solution exists for all the elements of the K matrix.

The integral reinforcement learning was implemented with a discounted cost function which allowed the algorithm to converge. An important observation for the case of the RL algorithm was that efficient training and convergence of the algorithm was not possible for a large order A matrix due to the curse of dimensionality. This observation compelled us to decouple the system dynamics and tune the control laws individually for the decoupled system. Also, it was observed that the convergence of the RL algorithm was highly susceptible to changes in the discount factor and the command generator dynamics. This warranted tuning of the discounted factor according to the command generator dynamics.

For estimation, it was observed that both the LQG and Kalman filter can estimate and track the states. However, the LQG becomes computationally expensive when exposed to noise. As the noise increases steadily, it takes exponentially more time to solve and this is never a solution for a real system. The Kalman filter does an excellent job with noisy systems as it simulates the effects and solves in real time.

The Kalman filter essentially does an optimal weight balancing between the system and the measurements. If our system is partially unknown, having more process noise, then it be optimized to give more factor to the measurements and vice-versa. While simulating with low level of noise, the Kalman filter does an excellent job of estimating the states. While simulating with the wrong process and measurement noise, the filter does a bad job with the wrong process noise and decent with the wrong measurement. It thus relies more on the system dynamics rather than the sensor input. The algorithm also relies on the initial uncertainty, the initial covariance matrix, which keeps updating through the process. It works fine for a wrong guess, as the Kalman gain updates this covariance until its effect dies out. The Kalman gain itself saturates after some time estimating the system.

At the end, the effect of optimal controllers is visible as these are able to handle and control real life noisy systems (Kalman Filter) and uncertain systems (RL) which simple controllers can’t handle.

References

[1] H. Modares and F. L. Lewis, "Linear Quadratic Tracking Control of Partially-Unknown Continuous-Time Systems Using Reinforcement Learning," in *IEEE Transactions on Automatic Control*, vol. 59, no. 11, pp. 3051-3056, Nov. 2014.

[2] P. Wang, Z. Man, Z. Cao, J. Zheng and Y. Zhao, "Dynamics modelling and linear control of quadcopter," *2016 International Conference on Advanced Mechatronic Systems (ICAMechS)*, Melbourne, VIC, 2016, pp. 498-503.

[3] C. Li, D. Liu and H. Li, "Finite horizon optimal tracking control of partially unknown linear continuous-time systems using policy iteration," in *IET Control Theory & Applications*, vol. 9, no. 12, pp. 1791-1801, 6 8 2015.

[4] Greg Welch, Gary Bishop, "An Introduction to the Kalman Filter", University of North Carolina at Chapel Hill Department of Computer Science, 2001