Design and Simulation of a LQG Optimal Controller for a Mobile Cart

Zexi Liu

Department of Electrical & Computer Engineering, Temple University
1801 N. Broad Street, Philadelphia, PA 19122

Abstract: This paper presents the basic principle, design and simulation of a LQG optimal controller for a mobile cart. The optimal control is a significant branch of control theory, which combines the statistical process theory with the optimal control theory to solve many control problems in uncertain systems like astronavigation, satellite attitude and orbit, civil purpose and industrial processes. In this paper, based on descriptions of the optimal control theory and techniques, an application of optimal control for a mobile cart is discussed. The controller structure, the optimization method and the simulation results are also presented.

Keywords: Mobile cart, Random noise, LQG optimal control, Kalman optimal estimation, Minimized cost function

I INTRODUCTION

Mobile carts are broadly used in industry, ports, planet exploration, nuclear waste cleanup, agriculture and mining [1]. The control configuration of a mobile cart mainly comprises position estimation and path tracking. Position estimation plays an important role in mobile cart control, which produces an accurate estimation based on the measured data from sensors on the mobile cart. A Kalman filter is a typical application to fuse multi-sensor measurements to provide an accurate position estimation [1]. Path tracking actually is an optimal control problem which predicts output based on a control model according to the optimal estimation of the cart state variables. The model predictive control (MPC) is a most common application of path tracking based on linear models of dynamic systems and evaluated by minimizing a cost function [2]. However, many optimal control models have been proposed to solve different control problems over last several decades. Gu et al reported a neural predictive control scheme for a car-like mobile robot to solve a nonlinear system optimal control problem [1]. Conceicao et al presented a nonlinear model predictive control strategy for trajectory tracking of a four-wheeled omni-directional mobile robot [2]. Approaches of the control problems mainly focus on minimizing the cost function which penalizes errors and control effort.

Linear-Quadratic-Gaussian (LQG) control is commonly applied for these control purposes. The LQG controller is a combination of a Kalman filter i.e. a Linear-Quadratic Estimator (LQE) with a Linear-Quadratic Regulator (LQR). The separation principle guarantees that these can be designed and computed independently [3]. LQG optimal control techniques have been applied in many linear dynamic system cases since 1960s [4- 6]. An optimal LQG controller for linear stochastic systems with unknown parameters has been developed by Basin et al. “The original controller problem was split into the optimal filtering problem for incompletely measured bilinear states over linear observations and the optimal control (regulator) problem for the designed filter estimate” [7]. Petersen et al reported a practical robust controller design methodology based on a minimax LQG control for the control of vibrations in a flexible cantilever beam [8]. Kadali et al presented a controller performance analysis with LQG benchmark obtained under closed loop conditions. “The optimal LQG-benchmark variances are obtained directly from the subspace matrices corresponding to the deterministic inputs and the stochastic inputs, which are identified using closed-loop data with setpoint excitation. These variances are used for assessing the controller performance". Kadali et al claimed that the method proposed is applicable to both univariate and multivariate systems [9].

Furthermore, researchers proposed some novel optimal control methods. Won presented optimal control methods with parameter robust risk-sensitive control synthesis for satellite structure attitude control and using an algebraic method for control-affine nonlinear systems [10, 11]. Sain et al reported a cost mean and variance theory with application to seismic protection of structures [12].

In summary, the configuration and design of a LQG optimal controller for linear systems are principally involved in both determining an optimal process estimation by a linear-quadratic estimator and making an optimal control strategy by a linear-quadratic regulator. This paper describes the design and simulation of a LQG optimal controller for a mobile cart. The system description and the system model are first given. Then the structure, design and optimization method of LQG controller are discussed. Simulation results are finally presented.

II BASIC PRINCIPLE OF LQG OPTIMAL CONTROL

A linear dynamic system can be described by the following mathematical models:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{1}$$

where *A*, *B*, *C* and D are state space matrices of the linear system model and *x* is state vectors, and *y* is the system output. If D=0, it represents that this is a single input system. If C=1, it means that this is a single output system. The LQG control signal *u* is a state feedback described below:

$$u = -Kx \tag{2}$$

where, the vector *K* is obtained from the solution of a Riccati Algebraic Equation. u can be derived from the minimization of the quadratic cost function:

$$\begin{aligned} J = \sum_{k=0}^{\infty} & [x^T(k)q_c(k)x(k) \\ & + u^T(k)r_c(k)u(k)] \end{aligned} \tag{3}$$

where $q_c(k)$ and $r_c(k)$ weight matrices, i.e., design parameters chosen to meet the desired closed loop performance [13]. Eq.(3) is a general quadratic cost function which not only state excursions but control excursions and state-control products as well. This equation plays a significant role in designing linear optimal controller [14].

III APPLICATION OF OPTIMAL CONTROL TO A MOBILE CART

3.1 System description

In a navigation system using GPS, MEMS inertial sensors and knowledge-based data fusion, a cart was used as a mobile platform. The cart is assumed to be linear movement and the discrete time state equation of cart motion is given by：

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Cx_k + Du_k + z_k \end{aligned} \tag{4}$$

where y_k is an observation (or measurement) of the true state x_k at time *k*; w_k is the process noise which is assumed to be drawn from a zero mean multivariate normal distribution with covariance $\mathbf{Q}_k$; z_k is the measurement noise or observation noise which is assumed to be zero mean Gaussian white noise with covariance $\mathbf{R}_k$; w_k and z_k are assumed to be independent of each other. The status of the cart is determined by its position p and its velocity v. The control variable is acceleration u_k and the output y_k is measurement position. The acceleration may be changed with time or be constant and the position of the cart is measured in an interval T. The reference frame of the system is shown as in Figure 1.

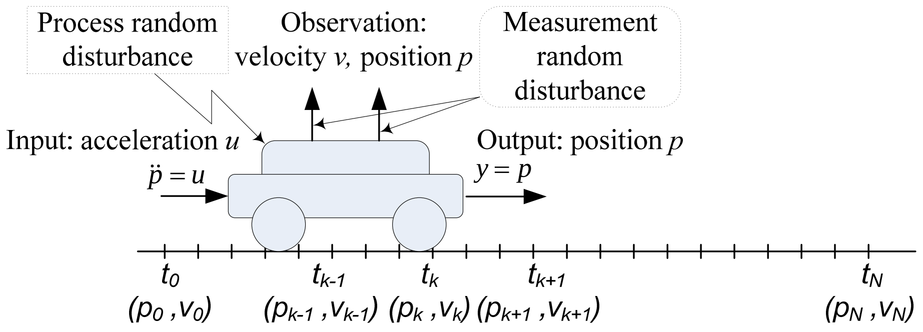


Figure 1 Sschematic of a cart motion model

3.2 System model

The motion model of the cart is built up in terms of Newton' s laws:

$$dv = udt \text{ or } \int_{v_0}^{v} dv = \int_{t_0}^{t} udt$$

$$ds = vdt = v_0 dt + utdt \text{ or } \tag{5}$$

$$\int_{s_0}^{s} ds = \int_{t_0}^{t} (v_0 dt + utdt)$$

where u is the acceleration of the mobile cart and v is the velocity of the mobile cart.
Their difference equations can be written below:

$$v_{k+1} = v_k + Tu_k \tag{6}$$

where u_k is the acceleration of the cart. The Eq. (6) demonstrates that the velocity in the next moment (after T) is the current velocity plus the product of acceleration and T. However, the real velocity of mobile cart will be disturbed with a gust or other unexpected interferences. When interference noise (or the process noise) $\tilde{v}_k$ is considered, Eq.(6) will be modified as follows:

$$v_{k+1} = v_k + Tu_k + \tilde{v}_k \tag{7}$$

Likewise, the equation of position p can be derived according to Eq.(5):

$$p_{k+1} = p_k + Tv_k + \frac{1}{2}T^2 u_k + \tilde{p}_k \tag{8}$$

where p_k the position of the cart, $\tilde{p}_k$ is interference noise (or the process noise). The state vector x_k of the system consists of as follows:

$$x_k = \begin{bmatrix} p_k \\ v_k \end{bmatrix} \tag{9}$$

Hence, the state equation of the linear uncertain system can be described by:

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} 1 & \mathrm{T} \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} u_k + w_k \\ y_k &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x_k + z_k \end{aligned} \tag{10}$$

where, y_k the output of the measured variable, w_k and z_k are the process noise and the measurement noise, respectively. It is assumed that the w_k and z_k are the random Gaussian white noise with zero mean, furthermore they are independent each other.

3.3 Design of a linear quadratic Gaussian (LQG) controller

Design of a linear quadratic Gaussian (LQG) controller for the mobile cart mostly focuses on three parts based on the basic principle of the LQG controller mentioned above: (1) design of an optimal linear quadratic estimator (LQE); (2) design of an optimal linear quadratic Gaussian regular (LQR); (3) system integration. Together with the linear quadratic estimator the linear quadratic Gaussian regular (LQR) solves this linear-quadratic-Gaussian control problem. Figure 2 shows the configuration of the LQG system.

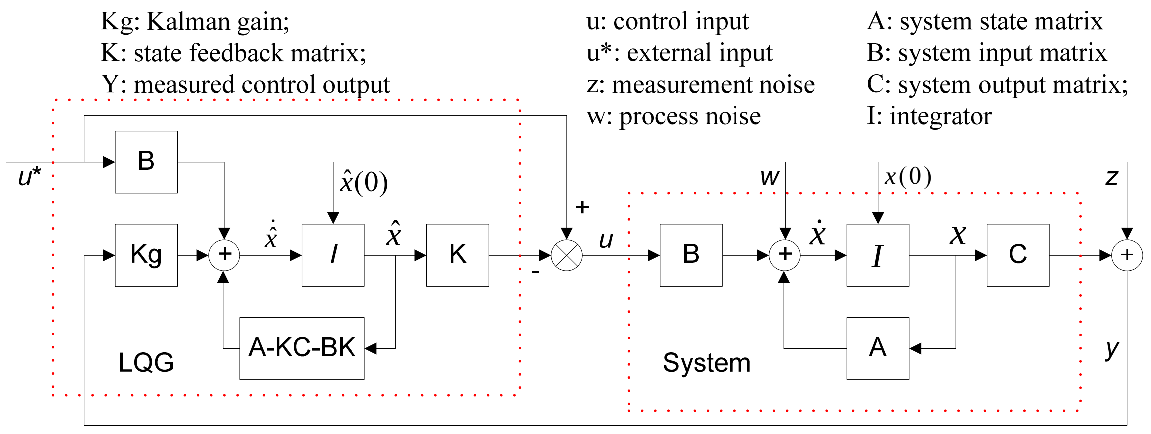


Figure 2 Configuration of the LQG system

3.3.1 Design of a linear quadratic estimator

The Kalman filter is a feasible estimation approach that can fuse multiple sensory measurements to provide relatively accurate results [1]. The Kalman filter, in a way, can minimize the mean of the squared error from a series of noisy measurements. The Kalman filter provides an optimal recursive data processing algorithm which works in the way that the present estimated value of the state is determined by the previous estimated value of the state and the present measured data instead of the entire measured data. The Kalman filter addresses the general problem of trying to estimate the state of a discrete-time controlled process that is governed by the linear stochastic difference equation. In practice, the Kalman filter can be presented as below:

$$
\begin{aligned}
\hat{x}_k &= A\hat{x}_{k-1} + Bu_{k-1} \\
P_k &= AP_{k-1}A^T + Q_{k-1} \\
\hat{y}_k &= y_k - c\hat{x}_k \\
S_k &= CP_kC^T + R_k \\
Kg_k &= AP_kC^T(CP_kC^T + R_k)^{-1} \\
&= AP_kC^TS_k^{-1}
\end{aligned}
\tag{11a}
$$

$$
\begin{aligned}
\hat{x}_{k+1} &= (A\hat{x}_k + Bu_k) \\
&\quad + Kg_k(y_{k+1} - C\hat{x}_k) \\
P_{k+1} &= AP_kA^T + Q_k \\
&\quad - AP_kC^TR_k^{-1}CP_kA^T
\end{aligned}
\tag{11b}
$$

where $\hat{x}_k$ is the predicted state; P_k is the predicted estimate covariance; $\hat{y}_k$ is the innovation or measurement residual; S_k is the innovation (or residual) covariance; Kg_k is the optimal Kalman gain; $\hat{x}_{k+1}$ is the innovation/updated state estimate; P_{k+1} is the updated estimate covariance. The covariance of the process noise Q_k and the covariance of the measurement noise R_k are given by, respectively:

$$
\begin{aligned}
&Q_k = E(w_kw_k^T) \text{ and} \\
&R_k = E(z_kz_k^T)
\end{aligned}
\tag{12}
$$

where w_k^T and z_k^T present the transpose of random noise matrices w_k and z_k, respectively; the mathematical expectation E(ε) is the expected value of ε.

The Matlab codes (see Appendix) of the Kalman estimator have been programmed for the system to obtain the optimal estimations of the velocity and the position for the mobile cart in the simulation. Assuming that the acceleration is the variable of time t: $u=a+bt$ (a and b are coefficients); the acceleration disturbed noise (process noise) is u_e = 2 m/s^2; the sampling rate

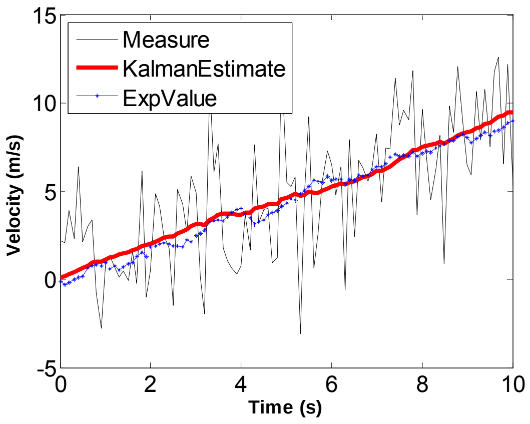


Figure 3 Velocity estimate by a Kalman estimator

(measurement cycle) T is 0.1 second. Due to $p \propto (T^2u^2/2)$ and $v \propto Tu$ from Eq.(7) and Eq.(8), the covariance of the process noise Q_k can be derived:

$$Q_k = E(w_k w_k^T) = E\left(\begin{bmatrix} p_{pe} \\ v_{pe} \end{bmatrix} \begin{bmatrix} p_{pe} & v_{pe} \end{bmatrix}\right)$$

$$= E(\begin{bmatrix} p_{pe}^2 & p_{pe}v_{pe} \\ v_{pe}p_{pe} & v_{pe}^2 \end{bmatrix})$$

$$= E\left(u_e^2 \begin{bmatrix} T^4/4 & T^3/2 \\ T^3/2 & T^2 \end{bmatrix}\right)$$

where $v_{pe}=Tu_e$ is the standard deviation of velocity and $v_e^2 =(Tu_e)^2$ is the variance of the velocity; $p_e=T^2u_e/2$ is the standard deviation of the position and $p_e^2 = (T^2u_e/2)^2$ the variance of the position. Likewise, the measurement noise R_k can be derived from Eq.(10) in a similar form:

$$R_k = E(z_k z_k^T) = E\left(\begin{bmatrix} p_{me} \\ v_{me} \end{bmatrix} \begin{bmatrix} p_{me} & v_{me} \end{bmatrix}\right)$$

$$= E(\begin{bmatrix} p_{me}^2 & p_{me}v_{me} \\ v_{me}p_{me} & v_{me}^2 \end{bmatrix})$$

Assuming that the position measurement error p_{me} is 1 m (standard deviation) and the velocity measurement noise v_{me} is 3 m/s (standard deviation), R_k can hence be presented by

$$R_k = E(\begin{bmatrix} 1\times1 & 1\times3 \\ 3\times1 & 3\times3 \end{bmatrix}) = \begin{bmatrix} 1 & 3 \\ 3 & 9 \end{bmatrix}$$

Figure 3 demonstrates the velocity estimate of the cart. The expected velocity was generated by Eq.(7) with a random signal. The measured velocity signal is mixed with the process noise and a white Gaussian noise as measurement noise. The Kalman estimator offers an optimal estimate using the measured signal together with the Kalman filtering algorithm. It is clear that the estimate of velocity is very close to the expected data regardless of noises. Figure 4 illustrates the position estimates of the cart. Again, the expected position was generated by Eq.(8) with a random signal. The measured position signal is mixed with the process noise and a white Gaussian noise as position measurement noise. Apparently, the estimate of position is also close to the expected position regardless of noises. Figure 5 shows the position and velocity estimate errors. It needs to be pointed out that the results will be slightly different each time due to the built in random signal generators.

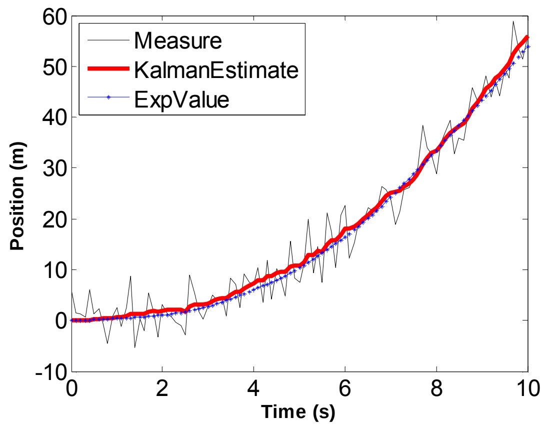


Figure 4 Position estimate by a Kalman estimator

3.3.2 Design of linear quadratic regulator

➢ Optimization method

The control strategy is characterized by the control configuration illustrated in Figure 2. The realization of the optimal control strategy depends on the state estimation and the control law. Considering the real mobile cart control problem, the control law is optimized by minimizing the cost function to guarantee control objectives which are a minimal position error and using as little control effort as possible [14]. The cost function is shown as follows:

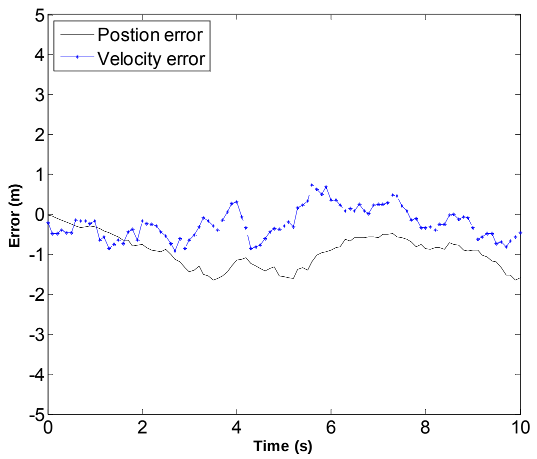


Figure 5 Position and velocity estimate error

$$J = \phi(x_f) + \int_{t_0}^{t_f} \zeta(t)dt \qquad (13)$$

with

$$\phi = q(x_{1_f} - s)^2, \zeta(t) = ru^2(t) \qquad (14)$$

where, *q* and *r* reflect the relative importance of satisfying each objective, s is related to the desired final value. A factor is introduced to adjoin the dynamic constraint to the cost function integrand, which represents the cost function's sensitivity to dynamic effects in the control problem [14]:

$$\lambda_1(t) = 2q(x_{1_f} - s), \qquad (15)$$
$$\lambda_2(t) = 2q(x_{1_f} - s)(t_f - t)$$

➢ Control law

The control values can be obtained by the optimization method mentioned above. In order to calculate this, the steepest descent method was applied, which is an iterative method, being simple implementation, but with slow convergence [2]. The algorithm steps of the steepest descent method are given in Table 1.

Table 1 Algorithm of steepest descent

Algorithm of steepest descent
(1) Given x_0 and set $k=0$;
(2) Calculate the lambda and the cost function's sensitivity;
(3) Calculate Hamiltonian $H(i)$ and $\frac{\partial H(i)}{\partial x}$;
(4) If $norm\frac{\partial H(i)}{\partial x} < \varepsilon$, stop; else, calculate the new point: $u = u - eps \times \frac{\partial H(i)}{\partial x}$, where ε the stop criterion, eps the step size of steepest descent;
(5) Set $k = k+1$, go to step 2.

➢ Calculation and simulation results

The Matlab codes (see Appendix) were made and simulation has been conducted to (1) select the controller's parameters, like the cost function's sensitivity to dynamic effects λ and the stop criterion ε; (2) obtain the cost function and the control law. Figures 6 and 7 present the

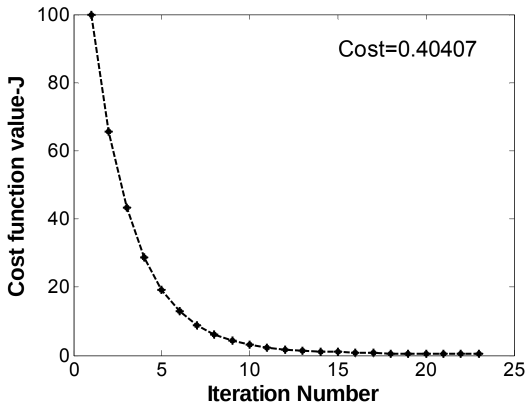


Figure 6 Cost function values

obtained cost function and the control law. It can be noted that the cost function has been close to zero after 5 iterations. Furthermore, a larger contribution for the solution takes always place in the first iterations, a small contribution at the end. It is noticed that the control law decreases linearly with time.

Figure 8 illustrates the cost function's sensitivity λ and Hamiltonian H. This sensitivity is specified by $\frac{\partial \phi}{\partial x}$ at the end time, providing a boundary condition for the solution of $\lambda(t)$. $\lambda(t_f)$ hence scales the effects of final state variables on the cost function [14]. In the mobile cart case, λ_1 is a constant with time and λ_2 increases linearly over time. This seems to suggest that the effects of final state variables on the cost function expressed by λ_1 are unvaried all the time whilst effects of final state variables on the cost function expressed by λ_2 enhance gradually with time.

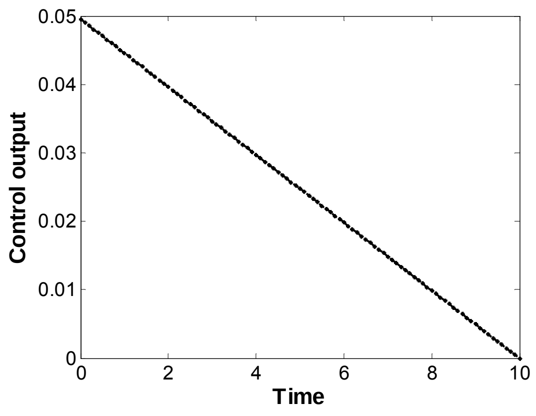


Figure 7 Optimal control

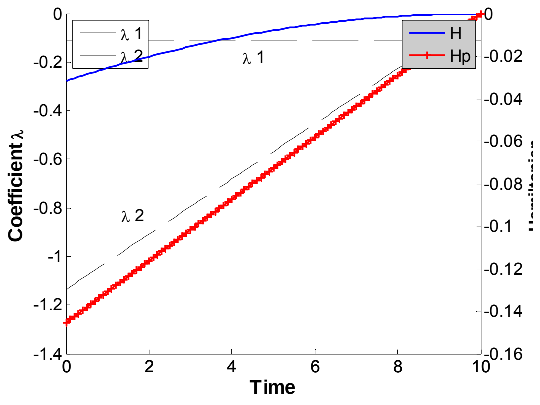
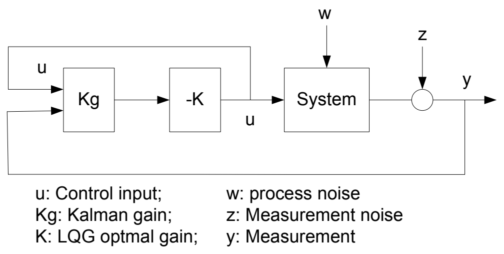


Figure 8 Coefficient λ and Hamiltonian



u: Control input; w: process noise
Kg: Kalman gain; z: Measurement noise
K: LQG optmal gain; y: Measurement

Figure 9 System closed-loop configuration

➢ System integration

System integration is a significant step in designing LQG controller after designing the optimal estimate and configurating the optimal controller [15]. Figure 9 shows the system closed-loop configuration. Figure 10 presents a comparison of open- and closed-loop impulse responses of the system. Figure 11 displays

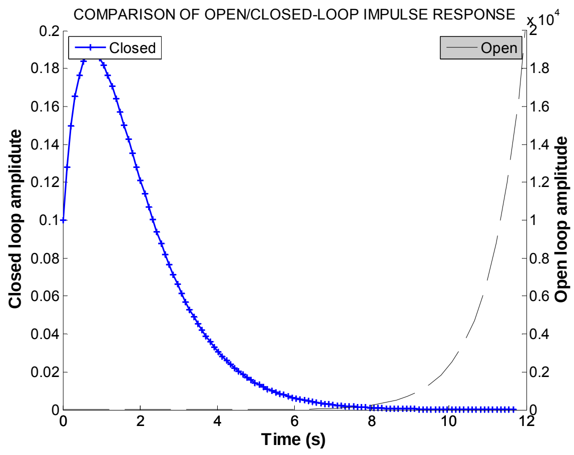


Figure 10 Comparison of the impulse response of the open/closed loop system

a comparison of open- and closed-loop step responses of the system. Apparently, the system is unstable when it is running in open loop status. When the control variance is introduced (in closed-loop status), the system is controllable and stable.

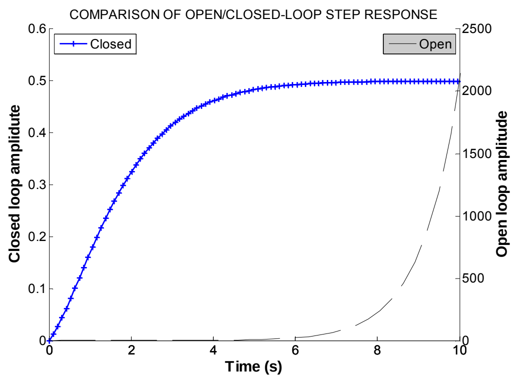


Figure 11 Comparison of the step response of the open/closed loop system

IV CONCLUSION

An optimal controller for a mobile cart has been designed in Matlab simulation to implement a predictive control to the mobile cart. The Kalman estimate was applied to determine the system output and the numeric optimization method was used to optimize the cost function, which enable a tradeoff between the regulation performance and the control effort. Process disturbance and measurement noises were taken into account in controller design. The velocity and position estimates, the optimal control law and the impulse and step responses of the closed-loop system are presented.

REFERENCES

[1] Gu Dongbing and Hu Huosheng (2002), Neural predictive control for a car-like mobile robot, *Robotics and Autonomous Systems*, 39, pp: 73–86.

[2] Conceicao A. S. ¸ Moreira A. P. and Costa P.J. (2007), A nonlinear model predictive control strategy for trajectory tracking of a four-wheeled omnidirectional mobile robot, *Optimal Control Applications and Methods*; 29, pp: 335–352.

[3] Wikipedia, Linear-quadratic-Gaussian control, http://en.wikipedia.org/wiki/Linear-quadratic-Gaussian_control.

[4] Kalman R.E., Bucy R.S. (1961), New results in linear filtering and prediction theory, *ASME Trans. D-J. Basic Eng*. pp. 83, 95–108.

[5] Kwakernaak H., Sivan R., Linear Optimal Control Systems, Wiley-Interscience, New York, 1972.

[6] Fleming F.H., Rishel R.W., Deterministic and Stochastic Optimal Control, Springer, New York, 1975.

[7] Basin M. and Alvarez D. C. (2008), Optimal LQG controller for linear stochastic systems with unknown parameters, *Journal of the Franklin Institute* 345, pp: 293–302.

[8] Petersen I. R. and Pota H. R.(2003), Minimax LQG optimal control of a flexible beam, *Control Engineering Practice*, 11, pp: 1273–1287.

[9] Kadali R. and Huang B. (2002), Controller performance analysis with LQG benchmark obtained under closed loop conditions, *ISA Transactions* 41, pp: 521–537.

[10] Won C.H. (2004), Satellite structure attitude control with parameter robust risk-sensitive control synthesis, *Proceedings of the 2004 American control conference*, Boston, June 30-July 2, 2004.

[11] Won C. H. and Biswas S. (2007), Optimal Control Using Algebraic Method for Control- Affine Nonlinear Systems, *International Journal of Control*, 80 (9), pp. 1491-1502.

[12] Sain M. K., Won C.H., Spencer B. F., and Liberty S. R. (2000), Cumulants and Risk-Sensitive Control: A Cost Mean and Variance Theory with Application to Seismic Protection of Structures, *Advances in Dynamic Games and Applications, Annals of the International Society of Dynamic Games*, 5, pp. 427-459.

[13] Ferreira A.M.D., Barreiros J.A.L., Jr. W.B. and Brito-de-Souza J.R. (2007), A robust adaptive LQG/LTR TCSC controller applied to damp power system oscillations, Electric Power Systems Research, 77, pp: 956–964.

[14] Stengel R.F. (1994), Optimal Control and Estimation, Dover Publications, INC. New York.

[15] The Mathworks, Control system toolbox, Functions for compensator design, http://www.mathworks.com/access/helpdesk/help/toolbox/control

APPENDICES

```
————————————————————————————————
%Kalman Estimator: CartKalman.m
close all;%deletes the current figure
clear all;%clear workspace and frees up system memory
clc;%clean screen
N=10;%end time;
T=0.1;%interval;
A=[1 T;0 1];%define the dynamic system
B=[T^2/2;T];%for acceleration
C=[1 0;0 1];%C(1,:)= position output; C(2,:)= velocity output
Cv=C(2,:);
D=[0;0];%for multi-output
u=[1];%corresponding to position=acceleration control variable
vmn=3;%velocity measurement noise
un=2.0;%acceleration noise/process noise
pmn=1;%position measurement noise
%x(1)=p;%x=state space; x(1)=position
%x(2)=v;%x=state space; x(2)=velocity
Q=un^2*[T^4/4 T^3/2;T^3/2 T^2];%process noise covairance of position
x=[0;0];%initial state of p and v;
xest=x;%initial state of estimate p and v;
%%%%%%%%%%%%%%Kalman filter%%%%%%%%%%%%%%%%%%%%%%
P=Q;%initial value of estimate covariance position
R=[pmn^2;vmn^2]; %position/velocity measurement error covariance;the
covariance with observing noise
Cn=0;%counter
for t=0:T:N;%loop
    Cn=Cn+1;%accumulator of iteration number
    eXpP=5+3*Cn/2;%expectation of position/velocity
    %u=[0.5-0.05*t/1];%acceleration change with time
    Pn=un*B*randn;%process noise wk=[w1;w2];
    x=A*x+B*u+Pn;%[position;velocity]: x(k+1)=Ax(k)+Bu(k)+w(k)
    PMNoise=pmn*randn;%position measurement noise random
    VMNoise=vmn*randn;%velocity measurement noise random
    MNoise=[PMNoise;VMNoise];%measurement noise zk=[z1;z2];
    y=C(1,:)*x+MNoise;%position output: y(k)=Cx(k)+z(k)
    yv=C(2,:)*x+MNoise(2,1);
    Upd=y-C(1,:)*xest; %update: y_est(k)=y(k)-
Cxest(k);Upd(1)=position;Upd(2)=velocity
    P1=C(1,:)*P*C(1,:)'+R;%covariance corresponding to the estimate
(Position/velocity)
    Kg=A*P*C(1,:)'/P1;%Kalman gain
    xest=A*xest+B*u+Kg*Upd;%update estimate of system state:
    P=A*P*A'+Q-A*P*C(1,:)'*inv(P1(1))*C(1,:)*P*A;%update estimate of system
state
    pt(Cn)=x(1);%Expected position
    vt(Cn)=x(2);%Expected velocity
    pe(Cn)=xest(1);%estimate position
    ve(Cn)=xest(2);%estimate velocity
    pm(Cn)=y(1);%measurement position
    vm(Cn)=yv(1);%measurement velocity
end;
pr=pt-pe;%expected value - estimated value
vr=vt-ve;%expected value - estimated value
close all;%deletes the current figure
t=(0:T:N)';%get the time axis
```

```
%velocity estimate
figure(1);%velocity
set(gca,'FontSize',16);
get(gca,'default');
plot(t,vm,'k');%Black=Velocity measurement;
hold on;
plot(t,ve,'r','LineWidth',3);%red=kalman estimate
plot(t,vt,'--*b','MarkerSize',4);%Blue=Velocity true;
legend('Measure','KalmanEstimate','ExpValue',2);
xlabel('Time (s)','FontWeight','bold','FontSize',14);
ylabel('Velocity (m/s)','FontWeight','bold','FontSize',14);
%axis([0 N 30 60]);
%title('Kalman predictor of velocity');
%position estimate
figure(2);%position
set(gca,'FontSize',16);
get(gca,'default');
plot(t,pm,'k');%Black=position measurement;
hold on;
plot(t,pe,'r','LineWidth',3);%red=Kalman estimate
plot(t,pt,'--*b','MarkerSize',4);%Blue=Velocity true;
legend('Measure','KalmanEstimate','ExpValue',2);
xlabel('Time (s)','FontWeight','bold','FontSize',14);
ylabel('Position (m)','FontWeight','bold','FontSize',14);
%axis([0 N 30 60]);
%title('Kalman predictor of position');
%error between expected and estimated values
figure(3);%error
set(gca,'FontSize',16);
get(gca,'default');
plot(t,pr,'k');%Black=position measurement;
hold on;
plot(t,vr,'--*b','MarkerSize',4);%Blue=Velocity true;
legend('Postion error','Velocity error',2);
xlabel('Time (s)','FontWeight','bold','FontSize',14);
ylabel('Error (m)','FontWeight','bold','FontSize',14);
%axis([0 N 30 60]);
%title('Kalman predictor of position');
```

————————————————————————————-

————————————————————————————-

```
%Gradient descent optimization of the optimal cart control: CostFunction.m
close all;%deletes the current figure
clear all;%clear workspace and frees up system memory
clc;%clean screen
q=0.01;%coefficient
r=10;%coefficient
eps=0.0005; % gradient descent step size
te=10; %end time
T=te/100; %time step size or interval
t=0:T:te;
N=size(t,2);
w=99;
%Configure the dynamic system.
A=[1 T; 0 1];%system state matrix
B=[T^2/2; T];%system input matrix
```

```
C=[1 0; 0 0];%system output matrix
D=[0;0];%
sys=ss(A,B,C,D);%system structure

%===
u=zeros(1,N);%Guess an initial optimal control.
SizeHp=inf;
for Cn=1:N;%iteration number
    [x,t]=lsim(sys,u,t);%Simulate the system with the given control.
    Lambda=zeros(N,2);%Compute the co-state.
    Lambda(N,1)=2*q*(x(N,1)-w);
    Lambda(N,2)=0;
    for i=N-1:-1:1;%
        LambdaDot=[0 -Lambda(i+1,1)];
        Lambda(i,:)=Lambda(i+1,:)-T*LambdaDot;
    end;
    for i=1:N;%Compute the Hamiltonian.
        H(i)=r*u(i)*u(i)+Lambda(i,:)*[x(i,2);u(i)];
    end;
    for i=1:N;%Compute the partial of the Hamiltonian with respect to the
control.
        Hp(i)=2*r*u(i)+Lambda(i,2);
    end;
    J=q*(x(N,1)-w)^2+r*T*(u(1)^2+u(N)^2)/2;%Compute the cost function.
    J=J+r*T*norm(u(2:N-1))^2;
    JArr(Cn)=J;%update each JArr in vectors
    SizeHpOld=SizeHp;%Compute the size of Hp.
    SizeHp=norm(Hp);
    if (SizeHp>SizeHpOld) break; %Quit when Hp gets close to zero.
    end;
    SizeHpArr(Cn)=SizeHp;%update each SizeHp in vectors
    if (SizeHp/N<0.01) break; %Quit when Hp gets close to zero.
    end;
    u=u-eps*Hp;%Update the control using gradient descent.
end;
close all;%deletes the current figure
%===figure 1: system response===
figure(1);
set(gca,'FontSize',14);
get(gca,'default');
plot(t,x,'k',...
         'LineWidth',1.5,...
         'MarkerEdgeColor','k',...
         'MarkerFaceColor','g',...
         'MarkerSize',4);
xlabel('Time','FontWeight','bold','FontSize',16);
ylabel('Position response','FontWeight','bold','FontSize',16);
%===figure 2: cost function===
figure(2);
set(gca,'FontSize',14);
get(gca,'default');
plot(JArr,'--*k',...
          'LineWidth',1.5,...
          'MarkerEdgeColor','k',...
          'MarkerFaceColor','g',...
          'MarkerSize',6);
%title('Cost Function');
text(15,90,['Cost=',num2str(J)],'FontSize',16);
xlabel('Iteration Number','FontWeight','bold','FontSize',16);
ylabel('Cost function value-J','FontWeight','bold','FontSize',16);
%===figure 3: control output===
figure(3);
set(gca,'FontSize',14);
get(gca,'default');
plot(t,u,'--*k',...
         'LineWidth',1.5,...
         'MarkerEdgeColor','k',...
         'MarkerFaceColor','g',...
         'MarkerSize',4);
%title('Optimal Control');
xlabel('Time','FontWeight','bold','FontSize',16);
ylabel('Control output','FontWeight','bold','FontSize',16);
%===figure 4: lambda, Hamiltonian, Hamilton partial differential===
figure(4);
ax1=gca;
%grid on;
set(ax1,'XColor','k','YColor','k','FontSize',14);
h1=line(t,Lambda,'Color','k','LineStyle','--');%
xlabel('Time','FontWeight','bold','FontSize',16);
ylabel('Coefficient \lambda','FontWeight','bold','FontSize',16);
legend('\lambda 1','\lambda 2',2);
%axis([0 t -1.4 1]);
ax2=axes('Position',get(ax1,'Position'),...
           'YAxisLocation','right',...
           'Color','none',...
           'FontSize',14,...
           'XColor','k','YColor','k');
h2=line(t,H,'Color','b','Parent',ax2,'LineWidth',1.5);%Hamiltonian
h3=line(t,Hp,'Color','r','Parent',ax2,'LineWidth',1.5,'Marker','+');%H'
ylabel('Hamiltonian','FontWeight','bold','FontSize',14);
legend('H','Hp',1);
```

————————————————————————————

————————————————————————————

```
%system response: SystemResponse.m
clear;
clc;
close all;
T=0.1;
A=[1 T;0 1];%system state matrix
B=[T^2/2;T];%system input matrix
C=[1 0];%system output matrix
D=[0];%

sys=ss(A,B,C,D);%Specify state-space models or convert LTI model to state
space
K=lqry(sys,1,10);%design LQ-optimal gain K:u=-Kx minimizes J(u);linear-
quadratic (LQ) state-feedback regulator with output weighting
P=sys(:,[1 1]);%Seperate control input u and disturbance input w(process
noise);
Kest=kalman(P,1,0.1);%Design discrete-time Kalman state estimator
Kest;measurement noise covariance E(n^2)=0.1
F=lqgreg(Kest,K);%Form LQG regulator=LQ gain (given state-feedback gain)+
Kalman filter(estimator);
clsys=feedback(sys,F,1);%Feedback connection of two LTI models
s=tf('s');%specify a TF model using a rational function in the Laplace
variable, s.
lpf=200/(0*s+10);%color filter,(actually =1;disable the filter)
clsys_fin=lpf*clsys;%closed loop after filter
```

```
%impulse response
[h1,to,tosd]=impulse(sys);
[h2,tc,tcsd]=impulse(clsys_fin);

figure(1);%impulse response
ax1=gca;
%grid on;
set(ax1,'XColor','k','YColor','k','FontSize',14);
lc=line(tc,h2,'Color','b','LineWidth',1.5,'Marker','+');%Hamiltonian
ylabel('Closed loop amplidute','FontWeight','bold','FontSize',16);
legend('Closed',2);
axis([0 12 0 0.20]);
ax2=axes('Position',get(ax1,'Position'),...
           'YAxisLocation','right',...
           'Color','none',...
           'FontSize',14,...
           'XColor','k','YColor','k');
lo=line(to,h1,'Color','k','Parent',ax2,'LineStyle','--');%
xlabel('Time (s)','FontWeight','bold','FontSize',16);
ylabel('Open loop amplitude','FontWeight','bold','FontSize',16);
legend('Open',1);
axis([0 12 0 20000]);
title('COMPARISON OF OPEN/CLOSED-LOOP IMPULSE RESPONSE');

%step response
[h4,to,tosd]=step(sys);%the larger
[h5,tc,tcsd]=step(clsys_fin);%the shorter

figure(2);%step response
ax1=gca;
%grid on;
set(ax1,'XColor','k','YColor','k','FontSize',14);
lc=line(tc,h5,'Color','b','LineWidth',1.5,'Marker','+');%Hamiltonian
ylabel('Closed loop amplidute','FontWeight','bold','FontSize',16);
legend('Closed',2);
axis([0 10 0 0.6]);
ax2=axes('Position',get(ax1,'Position'),...
           'YAxisLocation','right',...
           'Color','none',...
           'FontSize',14,...
           'XColor','k','YColor','k');
lo=line(to,h4,'Color','k','Parent',ax2,'LineStyle','--');%
xlabel('Time (s)','FontWeight','bold','FontSize',16);
ylabel('Open loop amplitude','FontWeight','bold','FontSize',16);
legend('Open',1);
axis([0 10 0 2500]);
title('COMPARISON OF OPEN/CLOSED-LOOP STEP RESPONSE');
```