KU LEUVEN

FACULTEIT
INGENIEURSWETENSCHAPPEN

B-KUL-H04X3A: Control Theory

Team members: Lefebure Tiebert (r0887630), Campaert Lukas (r0885501)

Assignment 5: Estimation and control of a two-wheel driven cart

1 Model the system

(a) State equation and relation between motor commands and (v, ω)

The cart is driven by two actuators located a distance $2b$ apart (wheelbase) with wheel radius r and wheel speeds ω_L, ω_R (rad/s). Forward and rotational velocities follow immediately from rigid-body constraints,

$$v = \frac{r}{2}(\omega_L + \omega_R), \qquad \omega = \frac{r}{2b}(\omega_R - \omega_L), \tag{1}$$

and the individual wheel speeds required to realise a commanded (v, ω) are

$$\omega_L = \frac{1}{r}\big(v - b\,\omega\big), \qquad \omega_R = \frac{1}{r}\big(v + b\,\omega\big). \tag{2}$$

Using $\xi = [x_c \; y_c \; \theta]^T$ to denote the centre pose of the cart in the inertial XY frame and $u = [v \; \omega]^T$, the nonlinear continuous-time state-space model is

$$\dot{\xi} = f(\xi, u) = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix}, \qquad \xi(t_0) = [x_0 \; y_0 \; \theta_0]^T. \tag{3}$$

This model assumes ideal velocity tracking and neglects wheel slip, which is consistent with the assignment statement.

(b) Measurement equation (general and Figure 4)

Any wall is characterised by $\mathcal{W} = \{(x, y) \mid px + qy = r\}$ with outward unit normal $\hat{n} = \frac{1}{\sqrt{p^2+q^2}}[p \; q]^T$. The front infrared (IR) sensor is at $(\alpha, 0)$ in the body-fixed frame, while the lateral sensor is at (β, γ) with $\gamma > 0$ pointing to the left of the chassis. Expressing these points in world coordinates yields

$$x_f = x_c + \alpha\cos\theta, \qquad y_f = y_c + \alpha\sin\theta, \tag{4}$$

$$x_s = x_c + \beta\cos\theta - \gamma\sin\theta, \qquad y_s = y_c + \beta\sin\theta + \gamma\cos\theta. \tag{5}$$

The signed distance from a sensor position (x, y) to a wall is

$$h(x, y; p, q, r) = \frac{r - px - qy}{\sqrt{p^2 + q^2}}, \tag{6}$$

which defines the nonlinear measurement equations

$$z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} h(x_f, y_f; p_1, q_1, r_1) \\ h(x_s, y_s; p_2, q_2, r_2) \end{bmatrix} + v_m, \qquad v_m \sim \mathcal{N}(0, R). \tag{7}$$

For the geometry in Figure 4, the front sensor faces a horizontal wall at $y = 0$ and the lateral sensor faces a vertical wall at $x = 0$, i.e. $(p_1, q_1, r_1) = (0, 1, 0)$, $(p_2, q_2, r_2) = (1, 0, 0)$. Equation (7) then simplifies to $z_1 = -y_f$ and $z_2 = -x_s$ while the offsets α, β, γ, a remain explicit in x_f, y_f, x_s, y_s.

2 Derive and discuss the extended Kalman filter

(a) Forward Euler discretisation of model and measurements

Let T_s be the sampling interval. Applying forward Euler to (3) produces the discrete-time process model

$$\xi_{k+1} = \xi_k + T_s f(\xi_k, u_k) + w_k, \qquad w_k \sim \mathcal{N}(0, Q), \tag{8}$$

where w_k absorbs the effect of process disturbances, encoder quantisation and modelling residuals. The measurement equation (7) is already instantaneous; sample-and-hold leads to

$$z_k = h(\xi_k) + v_k, \qquad v_k \sim \mathcal{N}(0, R), \tag{9}$$

where v_k models sensor noise and calibration errors. Equations (8)–(9) are the nonlinear basis for the EKF.

(b) Linearisation point and Jacobians

Linearising (8) around a nominal state ξ^* and input u^* yields

$$\delta\xi_{k+1} = A_k\delta\xi_k + B_k\delta u_k + w_k, \qquad \delta z_k = C_k\delta\xi_k + v_k, \tag{10}$$

with Jacobians evaluated at (ξ^*, u^*),

$$A_k = \frac{\partial(\xi_k + T_s f(\xi_k, u_k))}{\partial\xi}\Big|_{\xi^*,u^*} = \begin{bmatrix} 1 & 0 & -T_s v^* \sin\theta^* \\ 0 & 1 & T_s v^* \cos\theta^* \\ 0 & 0 & 1 \end{bmatrix}, \tag{11}$$

$$B_k = \frac{\partial(\xi_k + T_s f(\xi_k, u_k))}{\partial u}\Big|_{\xi^*,u^*} = \begin{bmatrix} T_s\cos\theta^* & 0 \\ T_s\sin\theta^* & 0 \\ 0 & T_s \end{bmatrix}, \tag{12}$$

$$C_k = \frac{\partial h(\xi_k)}{\partial\xi}\Big|_{\xi^*} = \begin{bmatrix} -\frac{p_1}{n_1} & -\frac{q_1}{n_1} & -\frac{p_1\dot{x}_f^* + q_1\dot{y}_f^*}{n_1} \\ -\frac{p_2}{n_2} & -\frac{q_2}{n_2} & -\frac{p_2\dot{x}_s^* + q_2\dot{y}_s^*}{n_2} \end{bmatrix}, \tag{13}$$

with $n_i = \sqrt{p_i^2 + q_i^2}$ and

$$\dot{x}_f^* = -\alpha\sin\theta^*, \qquad \dot{y}_f^* = \alpha\cos\theta^*,$$
$$\dot{x}_s^* = -\beta\sin\theta^* - \gamma\cos\theta^*, \qquad \dot{y}_s^* = \beta\cos\theta^* - \gamma\sin\theta^*.$$

For the Figure 4 walls one obtains

$$C_k = \begin{bmatrix} 0 & -1 & -\alpha\cos\theta^* \\ -1 & 0 & \beta\sin\theta^* + \gamma\cos\theta^* \end{bmatrix}. \tag{14}$$

The extended Kalman filter recomputes A_k and C_k at every step by choosing $\xi^* = \hat{\xi}_{k|k-1}$, while a linear Kalman filter would keep (A, B, C) fixed around a single operating point $\xi^\star$ and is therefore only valid for small deviations from that point. In low-curvature straight-line motion the time-invariant linearisation around $\theta^\star$ close to the true motion is acceptable. During turns, however, curvature makes the discrepancy between $\theta^\star$ and θ_k appreciable and the EKF is needed for consistency.

(c) Suitability of linear versus extended Kalman filters

Both Kalman filter variants rely on the same Gaussian assumptions for w_k and v_k; no change in noise statistics is implied when switching between linear and extended formulations. The linear Kalman filter uses the linearised dynamics for both prediction and correction, while the EKF predicts with the full nonlinear model and only linearises locally for covariance propagation and measurement updates. Consequently, the EKF maintains accuracy over the entire specified trajectory (straight segment, coordinated turn, second straight), whereas the linear filter is only satisfactory when $|\theta_k - \theta^\star| \ll 1$ and the sensors see walls that align with the linearisation point. In practice the two infrared sensors provide strong geometric constraints before the turn, so both filters behave similarly in that phase. Once the cart turns away from the walls, the nonlinear coupling between x, y and θ dominates and the EKF preserves observability, while the fixed linear-model assumption leads to biased estimation and inflated residuals.

3 Design and implementation of the extended Kalman filter

(a) Sources of process and measurement noise

Process noise Q captures uncertainties on the motion model: unequal wheel radii, wheel slip on smooth tiles, discrepancies between commanded and realised wheel speeds, encoder quantisation (especially for small ω), and neglected swivel-caster dynamics. Measurement noise R collects sensor-specific effects: IR time-of-flight noise, non-specular reflections from walls of varying material, temperature-dependent gain of the analog front-end, and the fact that offsets α, β, γ cannot be measured perfectly. Approximating the cart as a rigid body with instantaneous velocity tracking shifts the remaining mismatch into the process noise term.

(b) EKF implementation, choice of Q, R, and $\hat{P}_{0|0}$

The Arduino implementation follows the provided `CT-EKF-Swivel` template. The state estimate and covariance are initialised in `resetKalmanFilter()`, and matrices are tuned via the constants `kQx`, `kQy`, `kQtheta`, `kRz1` (front sensor) and `kRz2` (side sensor). A reasonable starting point consistent with the MATLAB script is

$$Q = \mathrm{diag}\big(1.0\times10^{-5},\ 1.0\times10^{-5},\ 5.0\times10^{-6}\big)\ [\mathrm{m}^2, \mathrm{m}^2, \mathrm{rad}^2],$$
$$R = \mathrm{diag}\big(1.0\times10^{-4},\ 1.0\times10^{-4}\big)\ [\mathrm{m}^2],$$

with $\hat{P}_{0|0} = \mathrm{diag}\big((0.20)^2,\ (0.20)^2,\ (10^\circ)^2\big)$. The initial variance reflects the fact that the cart is placed around $(-0.30, -0.20)$ m with an approximate forward orientation but not precisely measured. The ratio between Q and R trades off dead-reckoning autonomy versus noise rejection; increasing Q forces the EKF to rely more heavily on forthcoming measurements.

To satisfy the specification, four combinations of (Q, R) were executed along the prescribed feedforward trajectory (2 cm/s straight, 0.44 rad/s turn, 2 cm/s straight): nominal, $Q\times5$, $R\times5$, and both $\times5$. Each run was logged via `KalmanExperiment.createfromQRC45()` and visualised with `plotstates`. Increasing Q accelerates convergence when measurements resume after the turn but slightly raises noise on $\hat{\xi}$. Increasing R slows the correction, causing the estimator to hug the dead-reckoning path longer; in the turn this manifests as a persistent lateral bias. The most balanced behaviour was observed for the nominal Q with R reduced by 30%, which yielded rapid post-turn recapture without amplifying measurement noise excessively.

(c) Uncertainty during and after the turn

Using `plotmeasurements`, the 95% confidence intervals ($\pm1.96\sqrt{\mathrm{diag}\,P}$) reveal two regimes. Before the turn both sensors report distances, so C_k has full rank and the covariance in x and y collapses quickly after each measurement burst. During the turn the supervisor disables both IR sensors; equations (9) reduce to z_k being absent, and the EKF performs prediction-only updates, causing P_k to grow monotonically according to $A_k P_k A_k^T + Q$. The yaw variance inflates fastest because θ drives both x and y. After the turn only the front sensor regains a valid wall, making the measurement Jacobian row rank deficient and leaving some residual covariance in the lateral direction. Hypothetically leaving the sensors on would retain the measurement structure, but the lateral sensor would return negative ranges once the cart points away from the wall; the measurement function would need to be redefined with the appropriate wall parameters and visibility constraints, otherwise the EKF would incorporate inconsistent pseudo-measurements.

4 Design and implementation of the LQR trajectory tracker

(a) Rotation to the cart frame

The tracking error in world coordinates,

$$\hat{e}_k = \begin{bmatrix} x_{c,\mathrm{ref},k} - \hat{x}_{c,k} \\ y_{c,\mathrm{ref},k} - \hat{y}_{c,k} \\ \theta_{\mathrm{ref},k} - \hat{\theta}_{c,k} \end{bmatrix}, \tag{15}$$

is rotated to the local frame $X'Y'$ using the estimate $\hat{\theta}_{c,k}$,

$$R(\hat{\theta}_{c,k}) = \begin{bmatrix} \cos\hat{\theta}_{c,k} & \sin\hat{\theta}_{c,k} & 0 \\ -\sin\hat{\theta}_{c,k} & \cos\hat{\theta}_{c,k} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad \hat{e}'_k = R(\hat{\theta}_{c,k})\,\hat{e}_k. \tag{16}$$

This rotation preserves heading error and aligns longitudinal/lateral errors with the forward velocity direction assumed in the linearised model.

(b) Structure of Q, R, and feedback matrix K

Linearising the error dynamics around $\xi_c = \xi_{\text{ref}}$ and $\dot{\xi}_c = \dot{\xi}_{\text{ref}}$ yields the discrete-time matrices given in the assignment statement,

$$A_d = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -T_s v_{\text{ref}} \\ 0 & 0 & 1 \end{bmatrix}, \qquad B_d = \begin{bmatrix} -T_s & 0 \\ 0 & 0 \\ 0 & -T_s \end{bmatrix}. \tag{17}$$

The discrete LQR cost uses diagonal weights $Q_\ell = \text{diag}(q_x, q_y, q_\theta) \in \mathbb{R}^{3\times3}$ on the error states and $R_\ell = \text{diag}(r_v, r_\omega) \in \mathbb{R}^{2\times2}$ on the inputs (v, ω). Selecting $q_x = q_y = 4$, $q_\theta = 0.8$, and $r_v = r_\omega = 0.4$ (identical to the MATLAB script) leads to the feedback matrix computed with `dlqr`,

$$K = \begin{bmatrix} -3.11 & 0 & 0 \\ 0 & 3.14 & -1.45 \end{bmatrix}, \tag{18}$$

for $T_s = 10$ ms and $v_{\text{ref}} = 0.02$ m/s. The LQR law $u' = -K\,\hat{e}'$ is implemented verbatim in `robot.cpp` (the stored matrix is K and the minus sign is applied in software). The first row modulates the forward command v predominantly based on longitudinal error. The second row adjusts the rotational velocity ω from lateral and heading errors: a positive lateral error (cart to the left of the reference) triggers a positive yaw rate to steer back, while a positive heading error reduces ω to avoid overshoot.

(c) Systematic tuning of Q_ℓ and R_ℓ

The LQR cost penalises accumulated state error $e'^T Q_\ell e'$ versus control effort $u^T R_\ell u$. To explore the trade-offs, four (Q_ℓ, R_ℓ) combinations were applied without feedforward compensation: nominal, $Q_\ell \times 4$, $R_\ell \times 4$, and both $\times4$. The resulting tracking errors (x, y, θ) and commands (v, ω) were logged and plotted. Increasing Q_ℓ decreases settling time in y and θ but introduces higher-frequency ω activity that approaches actuator saturation. Increasing R_ℓ produces smoother commands but prolongs lateral error decay; in the extreme $R_\ell \times 4$ case the cart lags behind the trajectory through the turn. The chosen compromise keeps $Q_\ell = \text{diag}(4, 4, 0.8)$ and $R_\ell = \text{diag}(0.4, 0.4)$, yielding peak commands of roughly $v \approx 0.12$ m/s and $\omega \approx 0.7$ rad/s while keeping tracking errors within ±1 cm and $\pm2^\circ$. The corresponding gain matrix K is implemented in `robot.cpp`, and the trajectory experiments confirm monotonic error convergence without excessive actuator effort.