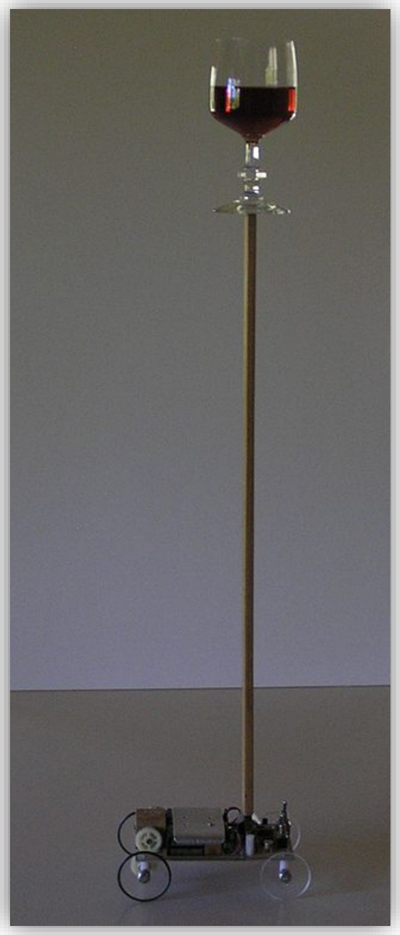
Реализация контроллера управления обратным маятником на C++

Константинов Даниил Николаевич

Физическая картина явления



Обратный маятник:
балансировка с бокалом вина,
установленным на робота

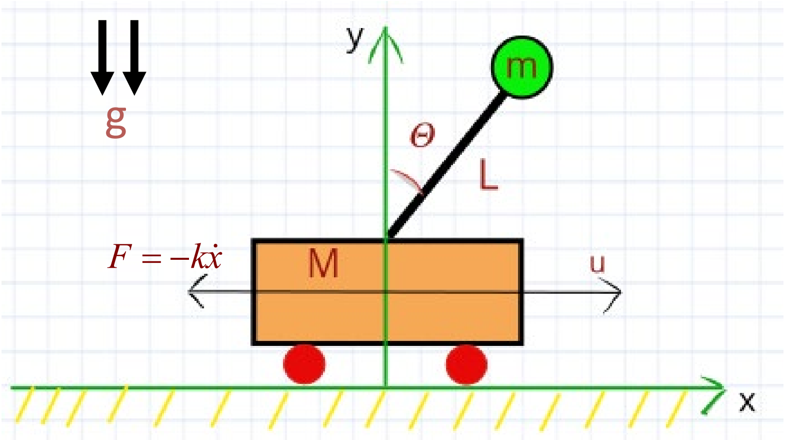
Математическая модель

Лагранжиан системы:

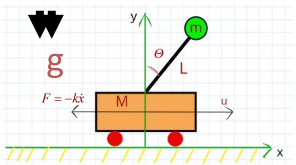
$$L = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m\left[\left(\frac{d}{dt}(x - L\sin\theta)\right)^2 + \left(\frac{d}{dt}(L\cos\theta)\right)^2\right] - mgL\cos\theta$$

Уравнения движения:

$$\begin{cases} \ddot{x} = \dfrac{mg\sin\theta\cos\theta - mL\dot{\theta}^2\sin\theta - k\dot{x} + u}{m\sin^2\theta + M} \\ \ddot{\theta} = \dfrac{(M+m)g\sin\theta - mL\dot{\theta}^2\sin\theta\cos\theta - k\dot{x}\cos\theta + u\cos\theta}{L(m\sin^2\theta + M)} \end{cases}$$



Постановка задачи



Итоговая система уравнений

$$
\begin{cases}
\ddot{x} = \dfrac{mg\sin\theta\cos\theta - mL\dot{\theta}^2\sin\theta - k\dot{x} + u}{m\sin^2\theta + M} \\[2ex]
\ddot{\theta} = \dfrac{(M+m)g\sin\theta - mL\dot{\theta}^2\sin\theta\cos\theta - k\dot{x}\cos\theta + u\cos\theta}{L(m\sin^2\theta + M)}
\end{cases}
$$

линеаризация
системы около т. θ ≈ 0

$$
\begin{cases}
x = \begin{pmatrix} x & \dot{x} & \theta & \dot{\theta} \end{pmatrix}^T \\[1ex]
\dot{x} = Ax + Bu
\end{cases}
$$

Как получить u?

Линейно-квадратичный регулятор (LQR)

$$\dot{x} = Ax + Bu$$

$$J = \int_0^\infty \left(x^T Q x + u^T R u \right) dt \to \min_u$$ ← функция потерь

Q, R – регуляторы

Уравнение Гамильтона-Якоби-Беллмана + теорема ГЯБ:

$$0 = \underbrace{\min}_{u} \left[x^T Q x + u^T R u + \frac{\partial J^*}{\partial x} \cdot \left(Ax + Bu \right) \right]$$

Известный факт:

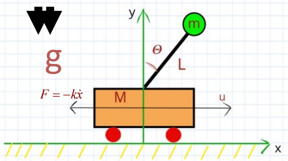
$$J^*\left(x\right) = x^T S x$$

J^* – оптимальные затраты механизма

$$\frac{\partial J^*}{\partial x} = 2x^T S$$

$$u^*\left(x\right) = -R^{-1} B^T S x = -Kx$$

Линейно-квадратичный регулятор (LQR)



Как получить S?

Честно подставляем u* в уравнение Гамильтона-Якоби-Беллмана:

$$0 = SA + A^T S - SBR^{-1}B^T S + Q$$

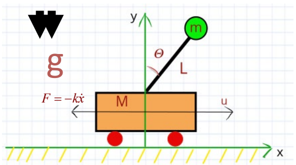
Реализация программы

```
ct::optcon::LQR<state_dim, control_dim> lqrSolver;
lqrSolver.compute(Q, R, A, B, K, RisDiagonal, solveRiccatiIteratively);
```

$$x^{n+1} = x^n + \Delta t\left[Ax + B\underbrace{\left(-K\left(x - x^{final}\right)\right)}_{u} \right]$$ – первый порядок точности по времени

$$\left\| x^{final} - x^{current} \right\|_2 \le \varepsilon$$ – критерий остановки

Случай конечного времени



Случай непрерывных систем

$$\dot{x} = Ax + Bu$$

$$J = \int_0^T \left(x^T Q x + u^T R u \right) dt \to \min_u$$ ⟵ функция потерь

Решение:

$$u = -Kx,$$

$K = R^{-1} B^T S,$ где S – решение уравнения:

$$A^T S + SA - SBR^{-1} B^T S + Q = -\dot{S}$$

Случай дискретных систем

$$x_{k+1} = Ax_k + Bu_k$$

$$J = \sum_{k=0}^{\infty} \left(x_k^T Q x_k + u_k^T R u_k \right) \to \min_u$$ ⟵ функция потерь

Решение:

$$u_k = -Fx_k,$$

$F = \tilde{R}^{-1} B^T P, \quad \tilde{R} = R + B^T PB,$ где P – решение уравнения:

$$A^T P + PA - PBR^{-1} B^T P + Q = -\dot{P}$$