Optimal Control for Modern Robotics: Lecture 4

경희대학교
로봇 제어 및 지능 연구실
석박통합과정 2기 강민형

Optimal Control

Limitation

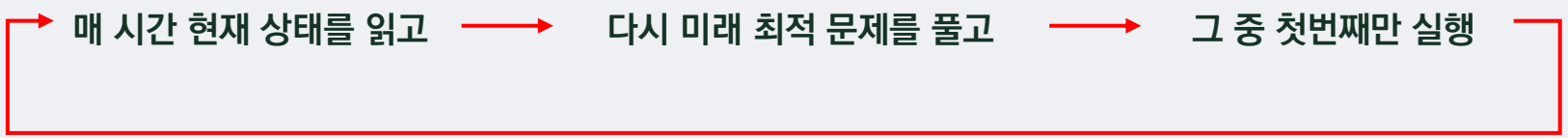
LQR

- 선형 시스템만 가능
- 제약 반영 불가 (Ricatti Equation)

iLQR / DDP

- 비선형 시스템에서의 trajectory optimization 가능
- 외란 발생 시 대처 x, -> 다시 최적화가 필요함

제약의 반영이 필요하고 외란이나 장애물로 인해 한번 최적화했던 궤적이 무의미해짐 → Feedback + 최적화가 지속적으로 이루어져야함

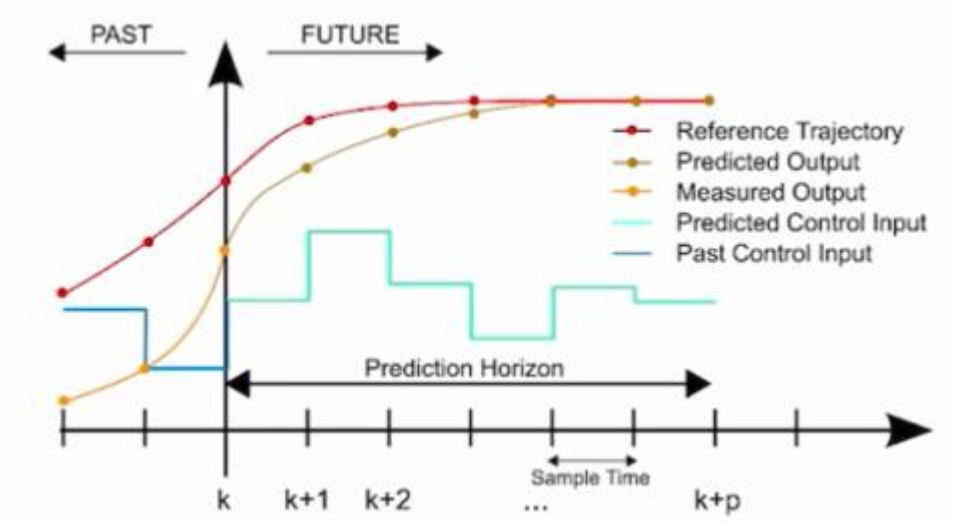


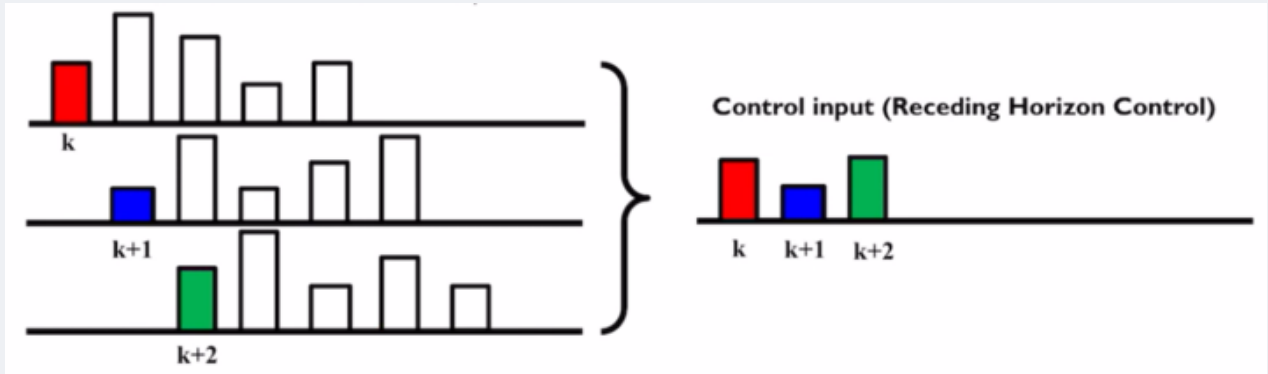
Model Predictive Control (MPC)

Optimal Control

Model Predictive Control

핵심 흐름





Receding Horizon –〉 각 시점마다 새로운 optimal input sequence가 생성됨

QP (Quadratic Programming) 문제를 풂으로써 optimal input sequence를 찾고 첫 step 만 채택

?

Optimal Control

Quadratic Programming

목적함수가 2차, 제약이 linear인 최적화 문제

Ricatti LQR → 즉 LQR은 제약이 없는 QP 문제

$$\min_u \left[x^\top Q x \; + \; u^\top R u \; + \; (2x^\top S)(Ax + Bu) \right] = 0.$$

$$\frac{\partial}{\partial u} \left(u^\top R u + 2x^\top S B u + \sim \right) = 2Ru + 2B^\top S x = 0.$$

미분해서 0이 나오는 곳이 최솟값

But, 제약이 걸리면

여기가 최솟값

Optimal Control

Why QP ?

목적함수가 2차, 제약이 linear인 최적화 문제

$$\begin{aligned} \min_{z} \quad & \frac{1}{2} z^{\top} H z + f^{\top} z \\ \text{s.t.} \quad & Az \leq b, \\ & Ez \leq d. \end{aligned}$$

Quadratic Programming의 장점

- 목적 함수가 convex
- 제약이 linear -> convex set

Convex Optimization

해 유일, 빠르게 찾을 수 있다

Local 적으로 minimu인 곳이 -> Global로도 minimum 보장

Optimal Control

Constrained LQR

Ricatti based LQR

- 선형 시스템만 가능
- 제약 반영 불가 (Ricatti Equation)

But, 제약이 걸리면

여기가 최솟값

QP based LQR

$$\min_{\{x_k,\,u_k\}_{k=0}^{N-1}} \quad \frac{1}{2}x_N^{\mathsf{T}} S\, x_N \;+\; \sum_{k=0}^{N-1} \frac{1}{2}\left(x_k^{\mathsf{T}} Q\, x_k + u_k^{\mathsf{T}} R\, u_k\right)$$

$$\text{s.t.} \qquad x_{k+1} = Ax_k + Bu_k, \qquad k = 0, \ldots, N-1,$$

$$x_0 \text{ given.}$$

LQR을 QP based 로 접근하면 제약이 있는 상황도 다룰 수 있게 됨

최적화 결과가 Model Predictive Control 의 한 step이 됨

Optimal Control

Constrained LQR

제약을 걸지 않은 QP 문제 = LQR

$$\min_{\{x_k,\,u_k\}_{k=0}^{N-1}} \quad \frac{1}{2}x_N^{\mathsf{T}} S\, x_N \;+\; \sum_{k=0}^{N-1} \frac{1}{2}\left(x_k^{\mathsf{T}} Q\, x_k + u_k^{\mathsf{T}} R\, u_k\right)$$

$$\text{s.t.} \qquad x_{k+1} = Ax_k + Bu_k, \qquad k = 0, \ldots, N-1,$$

$$x_0 \text{ given.}$$

$$x_1 = Ax_0 + Bu_0$$

$$x_2 = A^2x_0 + ABu_0 + Bu_1$$

$$x_3 = A^3x_0 + A^2Bu_0 + ABu_1 + Bu_2$$

$$x_k = A^kx_0 + \sum_{i=0}^{k-1} A^{k-1-i}Bu_i$$

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \underbrace{\begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ A^2B & AB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix}}_{\bar{C}} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} + \underbrace{\begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}}_{\hat{A}} x_0$$

$$X = \bar{C}U + \hat{A}x_0$$

Optimal Control

Constrained LQR

제약을 걸지 않은 QP 문제 = LQR

$$J = \frac{1}{2}x_N^\top S x_N + \sum_{k=0}^{N-1}\frac{1}{2}(x_k^\top Q x_k + u_k^\top R u_k).$$

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \qquad U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}.$$

모든 상태를 하나의 X, U로 모아서 정리

$$J = \frac{1}{2}X^\top \bar{Q} X + \frac{1}{2}U^\top \bar{R} U + \frac{1}{2}x_0^\top Q x_0.$$

$$\text{s.t.} \qquad X = \bar{C}U + \hat{A}x_0$$

$$\bar{Q} = \begin{bmatrix} Q & & \\ & \ddots & \\ & & S \end{bmatrix}, \qquad \bar{R} = \begin{bmatrix} R & & \\ & \ddots & \\ & & R \end{bmatrix}.$$

하나의 문자로 정리

$$\min_w \ \frac{1}{2}w^\top H w + f^\top w \quad \text{s.t.} \quad \tilde{A}w = \tilde{b}.$$

QP의 표준형

Optimal Control

Constrained LQR

하나의 문자로 정리

$$J = \frac{1}{2}X^\top \bar{Q}X + \frac{1}{2}U^\top \bar{R}U + \frac{1}{2}x_0^\top Q x_0.$$

$$\text{s.t.} \qquad X = \bar{C}U + \hat{A}x_0$$

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \qquad U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}.$$

$$\bar{Q} = \begin{bmatrix} Q & & \\ & \ddots & \\ & & S \end{bmatrix}, \qquad \bar{R} = \begin{bmatrix} R & & \\ & \ddots & \\ & & R \end{bmatrix}.$$

$$J = \frac{1}{2}X^\top \bar{Q}X + \frac{1}{2}U^\top \bar{R}U + \frac{1}{2}x_0^\top Q x_0.$$

$$X = \hat{A}x_0 + \bar{C}U.$$

$$J = \frac{1}{2}(\hat{A}x_0 + \bar{C}U)^\top \bar{Q}(\hat{A}x_0 + \bar{C}U) + \frac{1}{2}U^\top \bar{R}U + \frac{1}{2}x_0^\top Q x_0$$

$$= \frac{1}{2}U^\top(\bar{C}^\top \bar{Q}\bar{C} + \bar{R})U + x_0^\top \hat{A}^\top \bar{Q}\bar{C}U + \text{const.}$$

$$H = \bar{C}^\top \bar{Q}\bar{C} + \bar{R}, \qquad F = \hat{A}^\top \bar{Q}\bar{C}.$$

$$\nabla_U J = HU + Fx_0 = 0 \quad \Rightarrow \quad U^\star = -H^{-1}Fx_0.$$

Optimal Control

Constrained LQR

Adding Input Constraint

$$\min_U \ \frac{1}{2}U^\top HU + x_0^\top F^\top U$$

$$u_k \le u_{\max}, \qquad u_k \ge u_{\min}.$$

$$U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}. \quad U \le u_{\max}\mathbf{1}, \qquad -U \le -u_{\min}\mathbf{1}.$$

$$\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ -1 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} \le \begin{bmatrix} u_{\max} \\ \vdots \\ u_{\max} \\ -u_{\min} \\ \vdots \\ -u_{\min} \end{bmatrix}. \Rightarrow \begin{bmatrix} I \\ -I \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} \le \begin{bmatrix} u_{\max}\mathbf{1} \\ -u_{\min}\mathbf{1} \end{bmatrix}. \Rightarrow \boxed{\bar{A}_u U \le \bar{b}_u}$$

$$\bar{A}_u = \begin{bmatrix} I \\ -I \end{bmatrix}, \qquad \bar{b}_u = \begin{bmatrix} u_{\max}\mathbf{1} \\ -u_{\min}\mathbf{1} \end{bmatrix}.$$

Optimal Control

Constrained LQR

Adding State Constraint

$$x_k \le x_{\max}$$

$$\bar{C}U + \bar{A}x_0 \le x_{\max}$$

$$\bar{C}U \le x_{\max} - \bar{A}x_0$$

$$\bar{A}_x U \le \bar{b}_x - \bar{A}x_0$$

Constrained LQR

$$\min_U \quad \frac{1}{2}U^\top HU + x_0^\top F^\top U + (\text{constant})$$

$$\text{s.t.} \quad \begin{cases} \bar{C}\,U \le \bar{b}_x - \hat{A}x_0, \\ \bar{A}_u\,U \le \bar{b}_u. \end{cases}$$

KKT 및 Duality 를 활용해 풀기

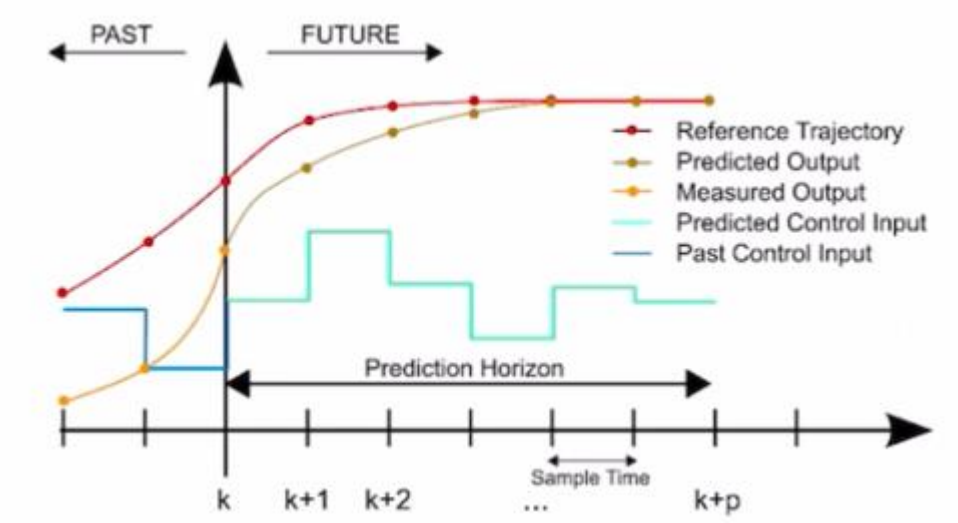
우리는 QP Solver 를 활용해서 품

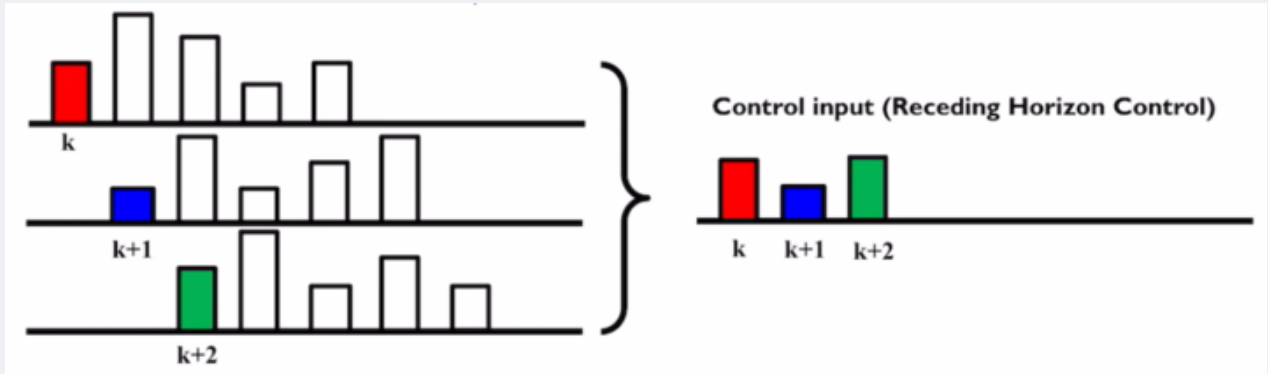
Linear System + 제약이 있는 상황에서 최적화 문제를 풀 수 있다

Optimal Control

• Model Predictive Control

QP를 풀어서 $U^\star = [u_0^\star,\ u_1^\star,\ \ldots,\ u_{N-1}^\star]$ 구함





$$x_1 = Ax_0 + Bu_0$$

$$x_0^{\text{new}} = x_1$$

Optimal Control

- Model Predictive Control

```python
# ------------------------------------------------------------------ #
# A_k, B_k, C_k, D_k, e_k 가져오기 (time-varying / invariant 대응)
# ------------------------------------------------------------------ #
def get_transition_state_matrix(self, k: int) -> np.ndarray:
    """k 시점의 상태 전이 행렬 A_k 반환."""
    return (
        self.transition_state_matrix[k]
        if isinstance(self.transition_state_matrix, list)
        else self.transition_state_matrix
    )

def get_transition_input_matrix(self, k: int) -> np.ndarray:
    """k 시점의 입력 전이 행렬 B_k 반환."""
    return (
        self.transition_input_matrix[k]
        if isinstance(self.transition_input_matrix, list)
        else self.transition_input_matrix
    )

def get_ineq_state_matrix(
    self,
    k: int,
) -> Union[None, np.ndarray, List[np.ndarray]]:
    """k 시점의 상태 제약 행렬 C_k 반환."""
    return (
        self.ineq_state_matrix[k]
        if isinstance(self.ineq_state_matrix, list)
        else self.ineq_state_matrix
    )

def get_ineq_input_matrix(
    self,
```

```python
# Quadratic term P, linear term q 초기값
# cost = ½ U' P U + q' U
# ---------------------------
P = (
    mpc_problem.stage_input_cost_weight
    * np.eye(stacked_input_dim, dtype=float)
)

# terminal cost: || x_N - x_goal ||² = || phi_last x0 + psi_last U - x_goal ||²
# → U' (psi_last' psi_last) U + ...
if mpc_problem.terminal_cost_weight is not None:
    P += (
        mpc_problem.terminal_cost_weight
        * psi.T.dot(psi)
    )

# stage state cost: Σ_k || x_k - x_ref_k ||²
# x_k = Phi_k x0 + Psi_k U → 전체를 스택하면 Psi 사용
if mpc_problem.stage_state_cost_weight is not None:
    P += (
        mpc_problem.stage_state_cost_weight
        * Psi.T.dot(Psi)
    )
```

```python
@property
def first_input(self) -> Optional[np.ndarray]:

    # 첫 번째 제어 입력 u_0 반환
    if self.__inputs is None:
        return None
    return self.__inputs[0]
```

감사합니다.