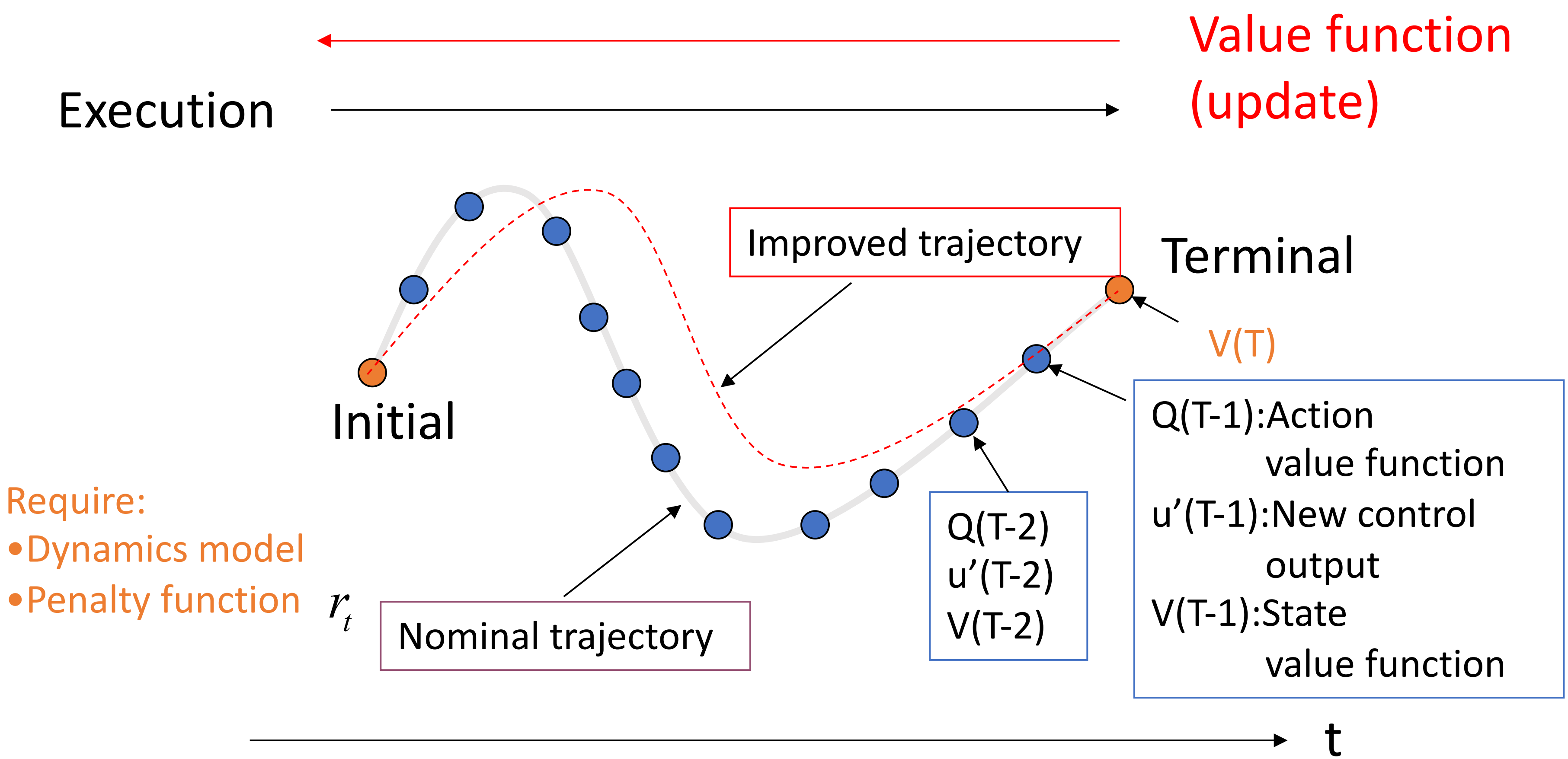
Implementing Trajectory Optimization on GPUs

Wei Chen

I. Background

- About Algorithm:
  - Trajectory Optimization[1]
  - Differential Dynamic Programming[1]
  - Iterative-Linear-Quadratic Regulator(iLQR)
- About Device:
  - Structure in GPU
  - CPU: 2.7GHz Intel Xeon E5 2686 v4
  - GPU: NVIDIA Tesla M60

1. Trajectory Optimization



2. Differential Dynamic Programming

$$\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i)$$

Discrete-time dynamics

$$J_0(\mathbf{x}, \mathbf{U}) = \sum_{i=0}^{N-1} \ell(\mathbf{x}_i, \mathbf{u}_i) + \ell_f(\mathbf{x}_N)$$

Total cost

$$\mathbf{U}^*(\mathbf{x}) \equiv \operatorname*{argmin}_{\mathbf{U}} J_0(\mathbf{x}, \mathbf{U}).$$

Goal: minimizing control sequence

$$J_i(\mathbf{x}, \mathbf{U}_i) = \sum_{j=i}^{N-1} \ell(\mathbf{x}_j, \mathbf{u}_j) + \ell_f(\mathbf{x}_N).$$

Cost-to-go

$$V(\mathbf{x}, i) = \min_{\mathbf{u}}[\ell(\mathbf{x}, \mathbf{u}) + V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i+1)]$$

Minimizations over a single control

2. Differential Dynamic Programming

$$V(\mathbf{x}, i) = \min_{\mathbf{u}}[\ell(\mathbf{x}, \mathbf{u}) + V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i{+}1)]$$

Perturbations around i-th (x, u) pair

$$\begin{aligned} Q(\delta\mathbf{x}, \delta\mathbf{u}) = {} & \ell(\mathbf{x} + \delta\mathbf{x}, \mathbf{u} + \delta\mathbf{u}, i) - \ell(\mathbf{x}, \mathbf{u}, i) \\ & + V(\mathbf{f}(\mathbf{x} + \delta\mathbf{x}, \mathbf{u} + \delta\mathbf{u}), i{+}1) - V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i{+}1) \end{aligned}$$

Expand to second order

$$\approx \frac{1}{2} \begin{bmatrix} 1 \\ \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} 0 & Q_{\mathbf{x}}^{\mathsf{T}} & Q_{\mathbf{u}}^{\mathsf{T}} \\ Q_{\mathbf{x}} & Q_{\mathbf{xx}} & Q_{\mathbf{xu}} \\ Q_{\mathbf{u}} & Q_{\mathbf{ux}} & Q_{\mathbf{uu}} \end{bmatrix} \begin{bmatrix} 1 \\ \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}$$

2. Differential Dynamic Programming

$$\approx \frac{1}{2}\begin{bmatrix}1\\ \delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}0 & Q_{\mathbf{x}}^{\mathsf{T}} & Q_{\mathbf{u}}^{\mathsf{T}}\\ Q_{\mathbf{x}} & Q_{\mathbf{xx}} & Q_{\mathbf{xu}}\\ Q_{\mathbf{u}} & Q_{\mathbf{ux}} & Q_{\mathbf{uu}}\end{bmatrix}\begin{bmatrix}1\\ \delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}$$

$$\begin{aligned}\Delta V(i) &= -\tfrac{1}{2}Q_{\mathbf{u}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{u}}\\ V_{\mathbf{x}}(i) &= Q_{\mathbf{x}} - Q_{\mathbf{u}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{ux}}\\ V_{\mathbf{xx}}(i) &= Q_{\mathbf{xx}} - Q_{\mathbf{xu}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{ux}}.\end{aligned}$$

$$\begin{aligned}Q_{\mathbf{x}} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}}V'_{\mathbf{x}}\\ Q_{\mathbf{u}} &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}V'_{\mathbf{x}}\\ Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}}V'_{\mathbf{xx}}\mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}}\cdot\mathbf{f}_{\mathbf{xx}}\\ Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}V'_{\mathbf{xx}}\mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}}\cdot\mathbf{f}_{\mathbf{uu}}\\ Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}V'_{\mathbf{xx}}\mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}}\cdot\mathbf{f}_{\mathbf{ux}}.\end{aligned}$$

$$\delta\mathbf{u}^* = \operatorname*{argmin}_{\delta\mathbf{u}} Q(\delta\mathbf{x},\delta\mathbf{u}) = -Q_{\mathbf{uu}}^{-1}(Q_{\mathbf{u}} + Q_{\mathbf{ux}}\delta\mathbf{x})$$

open-loop term $\mathbf{k} = -Q_{\mathbf{uu}}^{-1}Q_{\mathbf{u}}$

feedback gain term $\mathbf{K} = -Q_{\mathbf{uu}}^{-1}Q_{\mathbf{ux}}$

2. Differential Dynamic Programming

$$\boxed{\begin{aligned} Q_{\mathbf{x}} \\ Q_{\mathbf{u}} \\ Q_{\mathbf{xx}} \\ Q_{\mathbf{uu}} \\ Q_{\mathbf{ux}} \end{aligned}}\begin{aligned} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}} \\ &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}} \\ &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}} \\ &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\ &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}. \end{aligned}$$

Cost function, dynamics and next value

$\downarrow$

Q

2. Differential Dynamic Programming

$$Q_{\mathbf{x}} = \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{u}} = \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{xx}} = \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}}$$
$$Q_{\mathbf{uu}} = \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}$$
$$Q_{\mathbf{ux}} = \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.$$

$$\mathbf{k} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}}$$
$$\mathbf{K} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}$$

Cost function, dynamics and next value

Q

Control modifications

2. Differential Dynamic Programming

$$
\begin{aligned}
Q_{\mathbf{x}} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{u}} &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}} \\
Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\
Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.
\end{aligned}
$$

Cost function, dynamics and next value

↓

Q

↓

Control modifications

Local quadratic models of V(i)

$$
\begin{aligned}
\mathbf{k} &= -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}} \\
\mathbf{K} &= -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}
\end{aligned}
$$

$$
\begin{aligned}
\Delta V(i) &= -\tfrac{1}{2} Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}} \\
V_{\mathbf{x}}(i) &= Q_{\mathbf{x}} - Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}} \\
V_{\mathbf{xx}}(i) &= Q_{\mathbf{xx}} - Q_{\mathbf{xu}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}.
\end{aligned}
$$

2. Differential Dynamic Programming

$$Q_{\mathbf{x}} = \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{u}} = \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{xx}} = \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}}$$
$$Q_{\mathbf{uu}} = \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}$$
$$Q_{\mathbf{ux}} = \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.$$

Cost function, dynamics and next value

$\downarrow$

Q

$\downarrow$

Control modifications

Local quadratic models of V(i)

$$\mathbf{k} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}}$$
$$\mathbf{K} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}$$

$$\Delta V(i) = -\tfrac{1}{2} Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}}$$
$$V_{\mathbf{x}}(i) = Q_{\mathbf{x}} - Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}$$
$$V_{\mathbf{xx}}(i) = Q_{\mathbf{xx}} - Q_{\mathbf{xu}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}.$$

Backward Pass

2. Differential Dynamic Programming

$$\hat{\mathbf{x}}(1) = \mathbf{x}(1)$$
$$\hat{\mathbf{u}}(i) = \mathbf{u}(i) + \mathbf{k}(i) + \mathbf{K}(i)(\hat{\mathbf{x}}(i) - \mathbf{x}(i))$$
$$\hat{\mathbf{x}}(i+1) = \mathbf{f}(\hat{\mathbf{x}}(i), \hat{\mathbf{u}}(i))$$

Forward Pass

3. Iterative LQR

$$
\begin{aligned}
Q_{\mathbf{x}} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{u}} &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}} \\
Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\
Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.
\end{aligned}
$$

4. Regularization

$$\widetilde{Q}_{\mathbf{uu}} = \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}(V'_{\mathbf{xx}} + \mu \mathbf{I}_n)\mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}$$

$$\widetilde{Q}_{\mathbf{ux}} = \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}(V'_{\mathbf{xx}} + \mu \mathbf{I}_n)\mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}$$

$$\mathbf{k} = -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{u}}$$

$$\mathbf{K} = -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{ux}}$$

As role of Levenberg-Marquardt parameter

$$\Delta V(i) = \quad + \tfrac{1}{2}\mathbf{k}^{\mathsf{T}} Q_{\mathbf{uu}}\mathbf{k} + \mathbf{k}^{\mathsf{T}} Q_{\mathbf{u}}$$

$$V_{\mathbf{x}}(i) = Q_{\mathbf{x}} + \mathbf{K}^{\mathsf{T}} Q_{\mathbf{uu}}\mathbf{k} + \mathbf{K}^{\mathsf{T}} Q_{\mathbf{u}} + Q_{\mathbf{ux}}^{\mathsf{T}}\mathbf{k}$$

$$V_{\mathbf{xx}}(i) = Q_{\mathbf{xx}} + \mathbf{K}^{\mathsf{T}} Q_{\mathbf{uu}}\mathbf{K} + \mathbf{K}^{\mathsf{T}} Q_{\mathbf{ux}} + Q_{\mathbf{ux}}^{\mathsf{T}}\mathbf{K}.$$

5. Line search

$$\hat{\mathbf{u}}(i) = \mathbf{u}(i) + \alpha\mathbf{k}(i) + \mathbf{K}(i)(\hat{\mathbf{x}}(i) - \mathbf{x}(i))$$

$$\Delta V(i) = \qquad + \tfrac{1}{2}\mathbf{k}^\mathsf{T} Q_{\mathbf{uu}}\mathbf{k} + \mathbf{k}^\mathsf{T} Q_{\mathbf{u}}$$

$$\Delta J(\alpha) = \alpha \sum_{i=1}^{N-1} \mathbf{k}(i)^\mathsf{T} Q_{\mathbf{u}}(i) + \frac{\alpha^2}{2} \sum_{i=1}^{N-1} \mathbf{k}(i)^\mathsf{T} Q_{\mathbf{uu}}(i)\mathbf{k}(i).$$

$$z = [J(\mathbf{u}_{1..N-1}) - J(\hat{\mathbf{u}}_{1..N-1})]/\Delta J(\alpha)$$

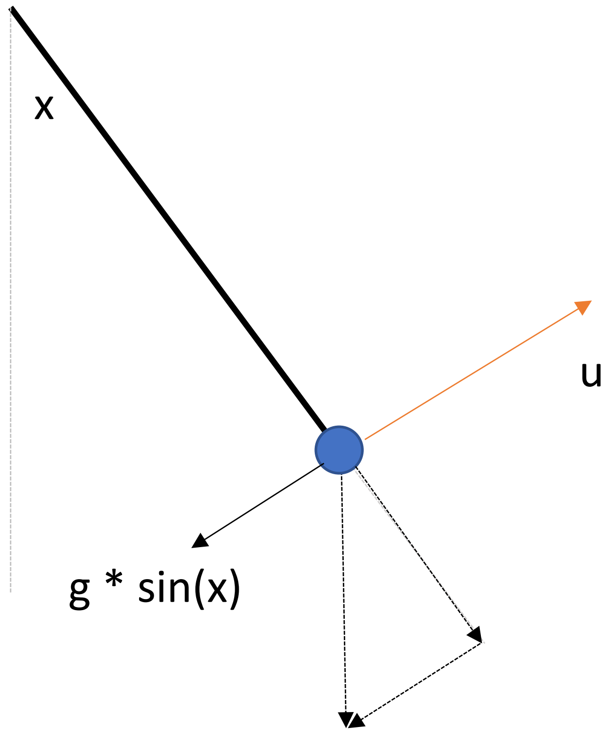
II. Implementation

- Sequential iLQR on CPU w/ linear algebra library
  - Library: Armadillo

- Sequential iLQR on CPU w/o linear algebra library

- Parallel version on GPU

Dynamics & Cost Function



$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} = \begin{bmatrix} x_2 \\ u - gsin(x_1)\end{bmatrix}$$

Cost function

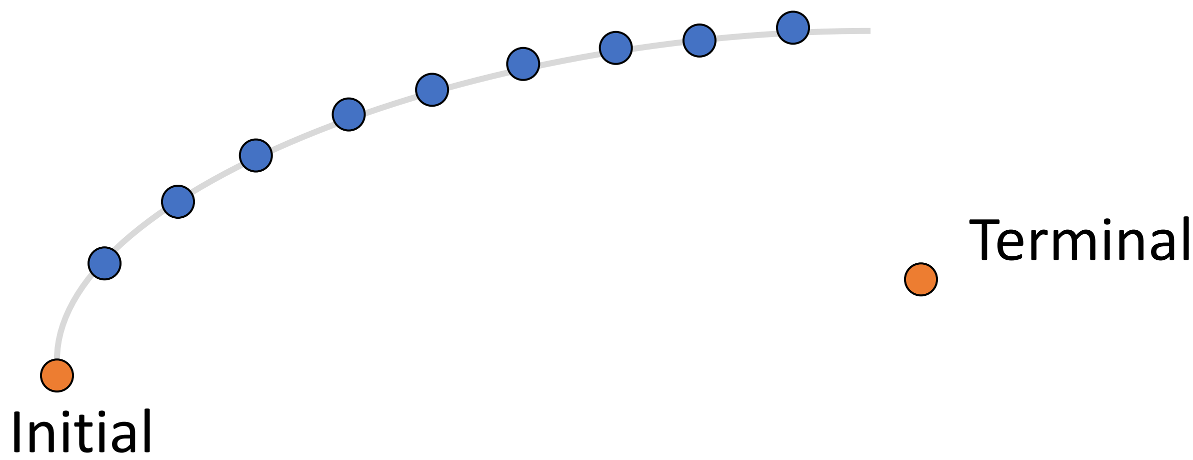
$$\frac{1}{2}(x_k - x_g)^T Q (x_k - x_g) + \frac{1}{2} u_k^T R u_k$$

Initial state

$$U = \{u_0, \ldots, u_{N-1}\}$$

Dynamics

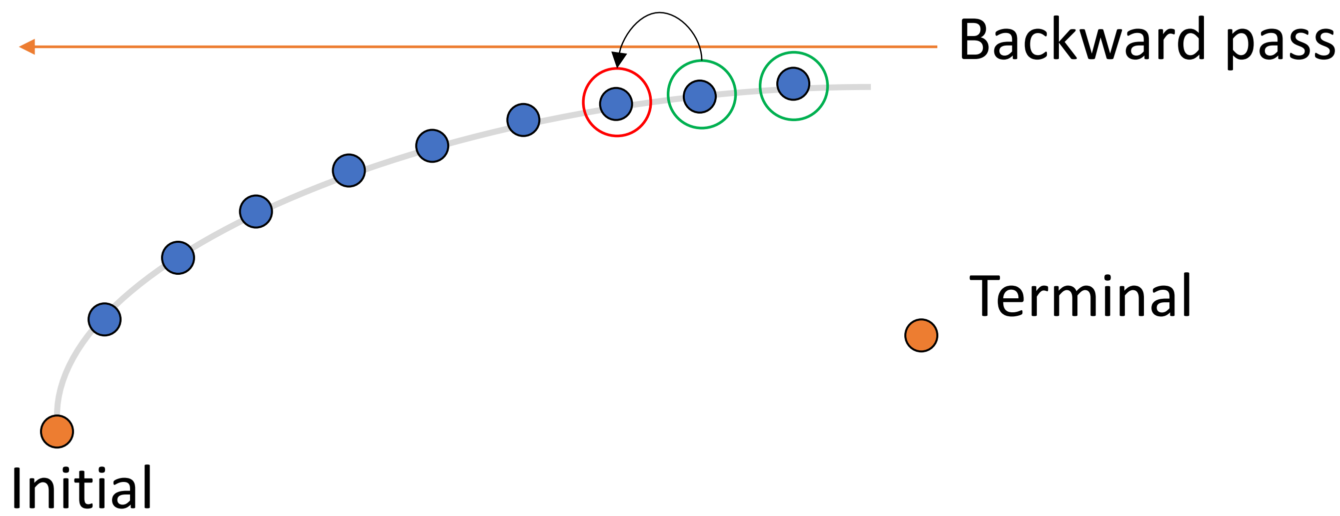
Cost function



1. Set the start point and goal
2. Set the number of steps
3. Initialize each states & controls
4. Load the dynamics & cost function

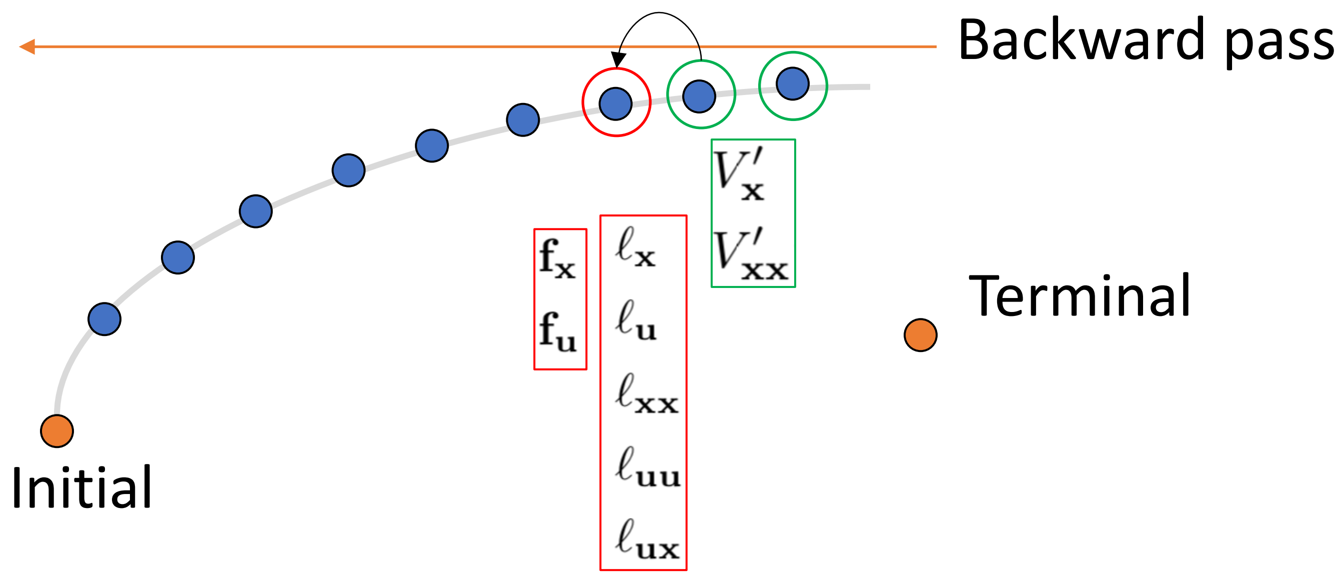
$$J = \frac{1}{2}(x_N - x_g)^T Q_N (x_N - x_g) + \sum_{k=0}^{N-1} \frac{1}{2}(x_k - x_g)^T Q (x_k - x_g) + \frac{1}{2} u_k^T R u_k$$

Backward pass



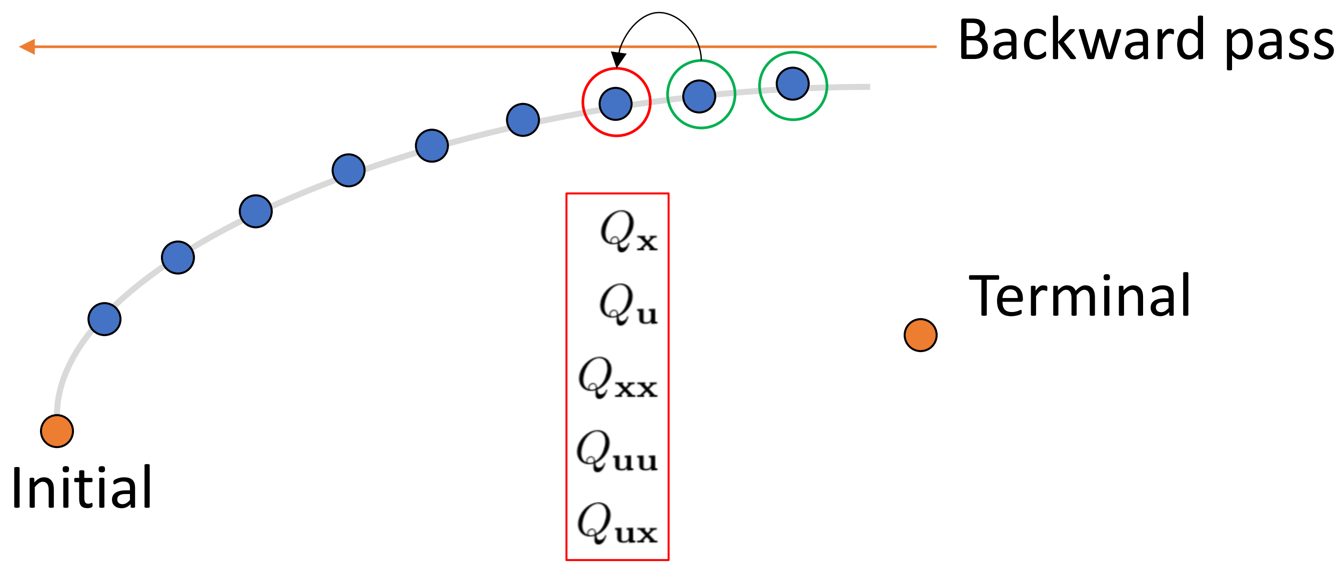
1. Calculate Q, inverse of Quu
2. Calculate control modification K and k
3. Update current value V

Backward pass



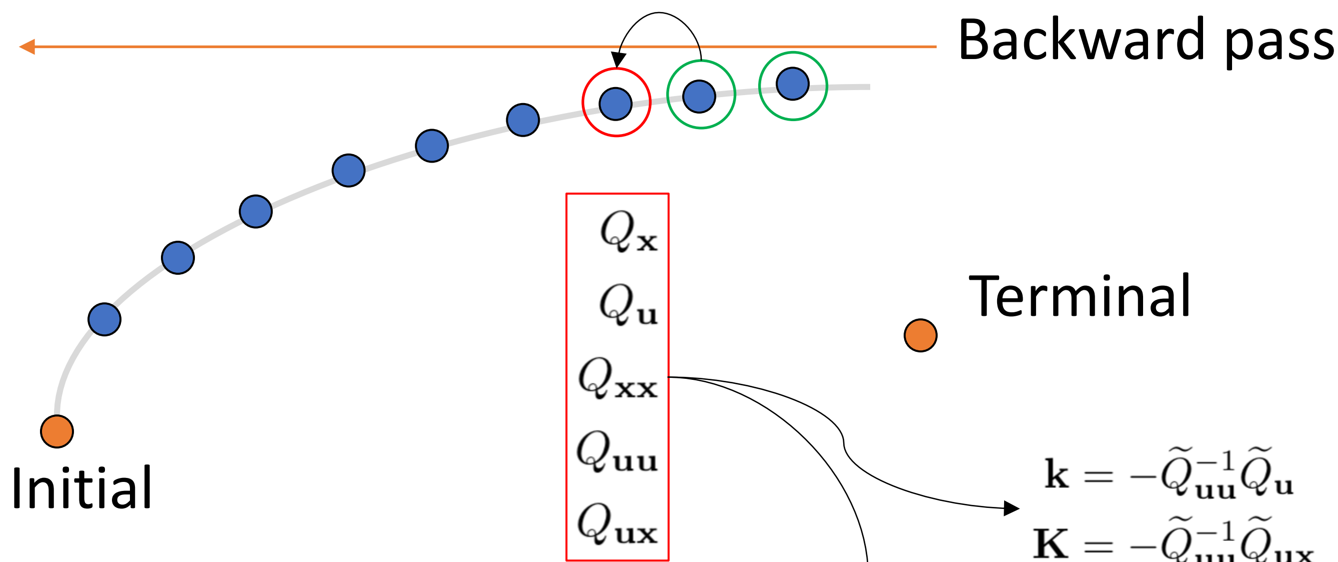
1. Calculate Q, inverse of Quu
2. Calculate control modification K and k
3. Update current value V

Backward pass



1. Calculate Q, inverse of Quu
2. Calculate control modification K and k
3. Update current value V

Backward pass

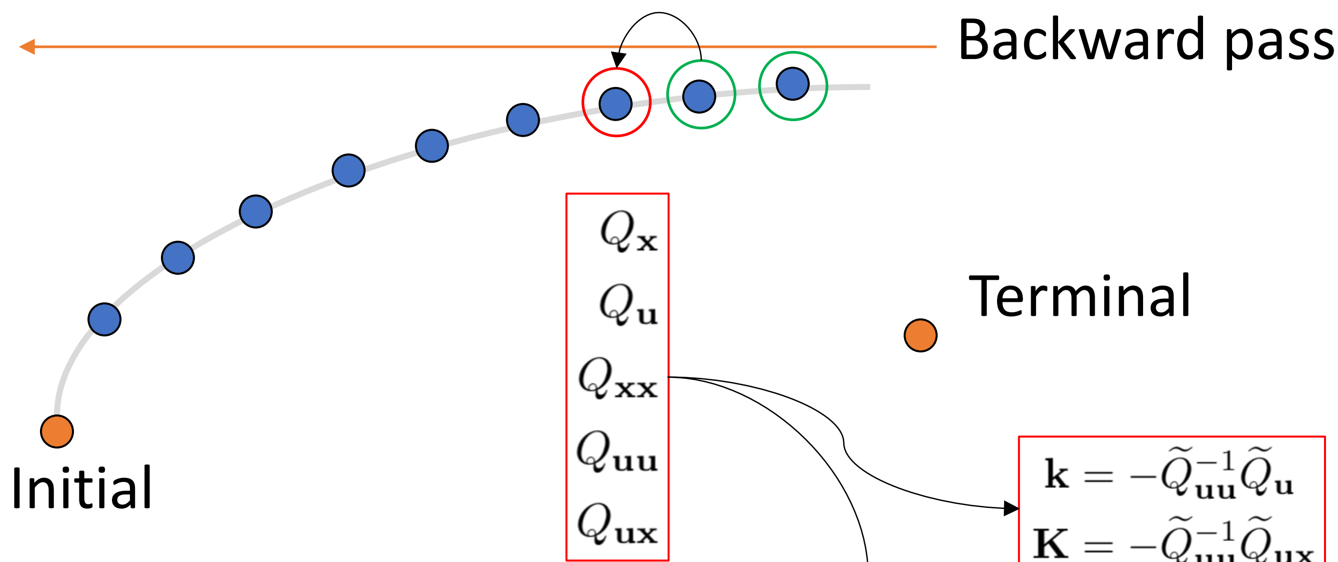


$$\mathbf{k} = -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{u}}$$
$$\mathbf{K} = -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{ux}}$$

$$\Delta V(i)$$
$$V_{\mathbf{x}}(i)$$
$$V_{\mathbf{xx}}(i)$$

1. Calculate Q, inverse of Quu
2. Calculate control modification K and k
3. Update current value V

Backward pass



$$\boxed{\begin{aligned}\mathbf{k} &= -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{u}} \\ \mathbf{K} &= -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{ux}}\end{aligned}}$$

For forward pass

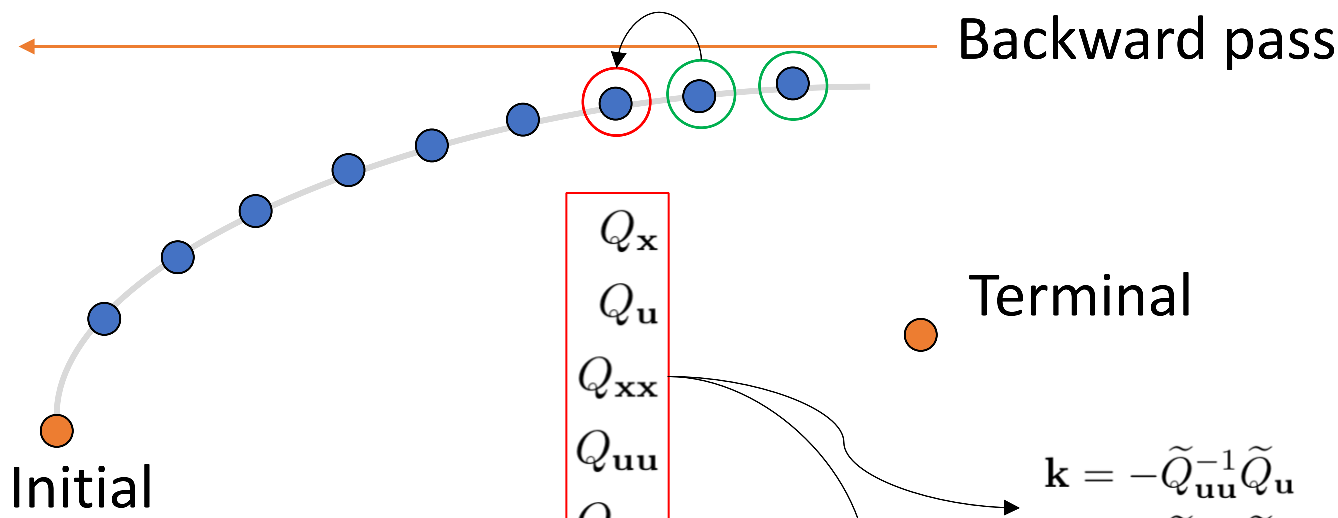
$$\boxed{\Delta V(i)}$$

$$V_{\mathbf{x}}(i)$$

$$V_{\mathbf{xx}}(i)$$

1. Calculate Q, inverse of Quu
2. Calculate control modification K and k
3. Update current value V

Backward pass



$$Q_{\mathbf{x}}$$
$$Q_{\mathbf{u}}$$
$$Q_{\mathbf{xx}}$$
$$Q_{\mathbf{uu}}$$
$$Q_{\mathbf{ux}}$$

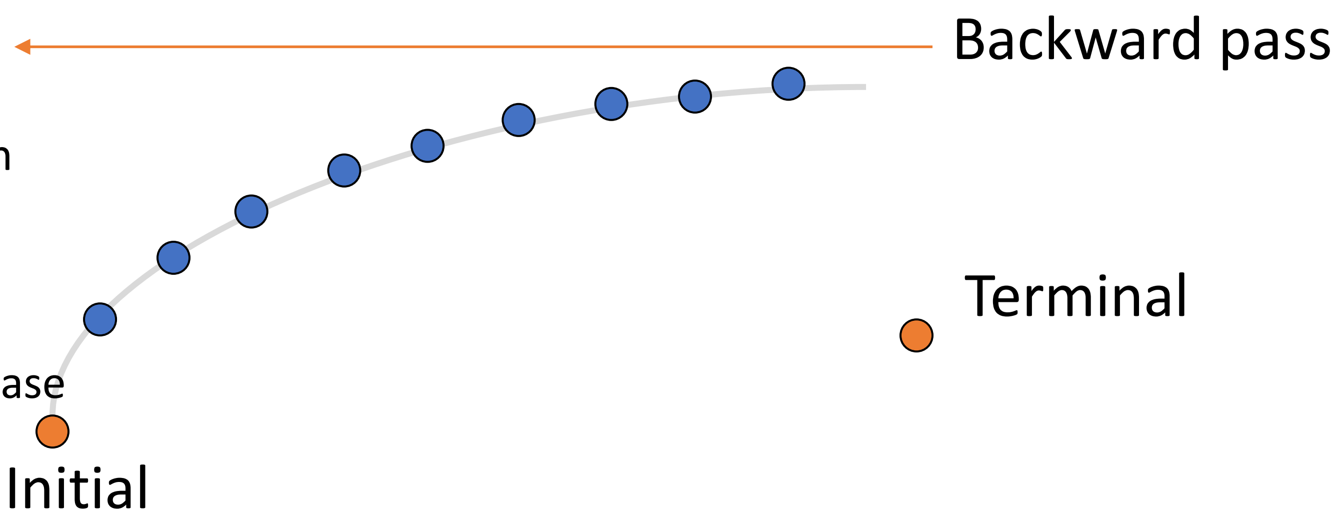
$$\mathbf{k} = -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{u}}$$
$$\mathbf{K} = -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{ux}}$$

$$\Delta V(i)$$
$$V_{\mathbf{x}}(i)$$
$$V_{\mathbf{xx}}(i)$$

For next step

1. Calculate Q, inverse of Quu
2. Calculate control modification K and k
3. Update current value V

Backward pass



Control modification

$$\mathbf{k} = -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{u}}$$

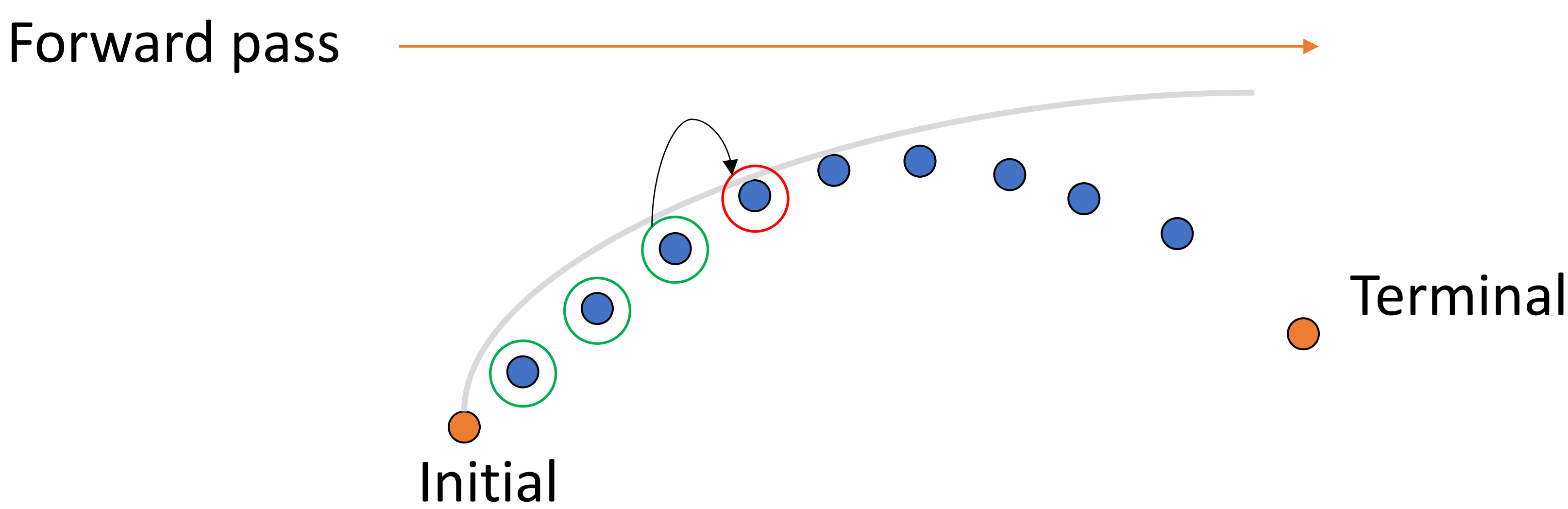
$$\mathbf{K} = -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{ux}}$$

Expected cost decrease

$$\Delta V$$

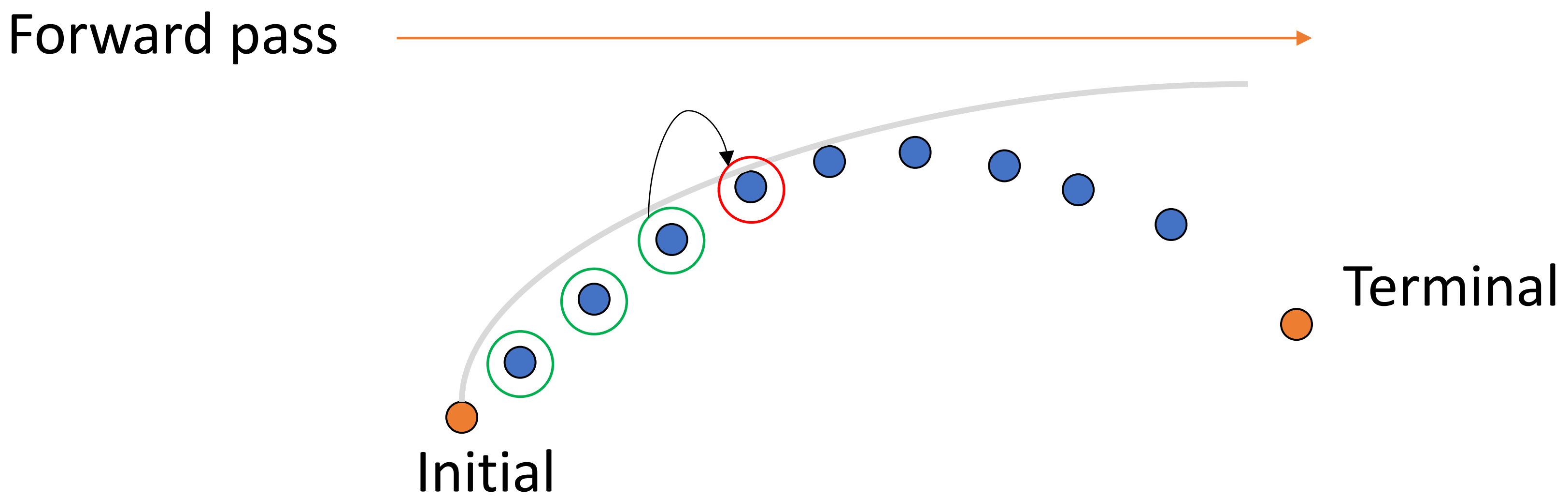
1. Calculate Q, inverse of Quu
2. Calculate control modification K and k
3. Update current value V

Forward pass



$$\hat{\mathbf{x}}(1) = \mathbf{x}(1)$$
$$\hat{\mathbf{u}}(i) = \mathbf{u}(i) + \mathbf{k}(i) + \mathbf{K}(i)(\hat{\mathbf{x}}(i) - \mathbf{x}(i))$$
$$\hat{\mathbf{x}}(i+1) = \mathbf{f}(\hat{\mathbf{x}}(i), \hat{\mathbf{u}}(i))$$

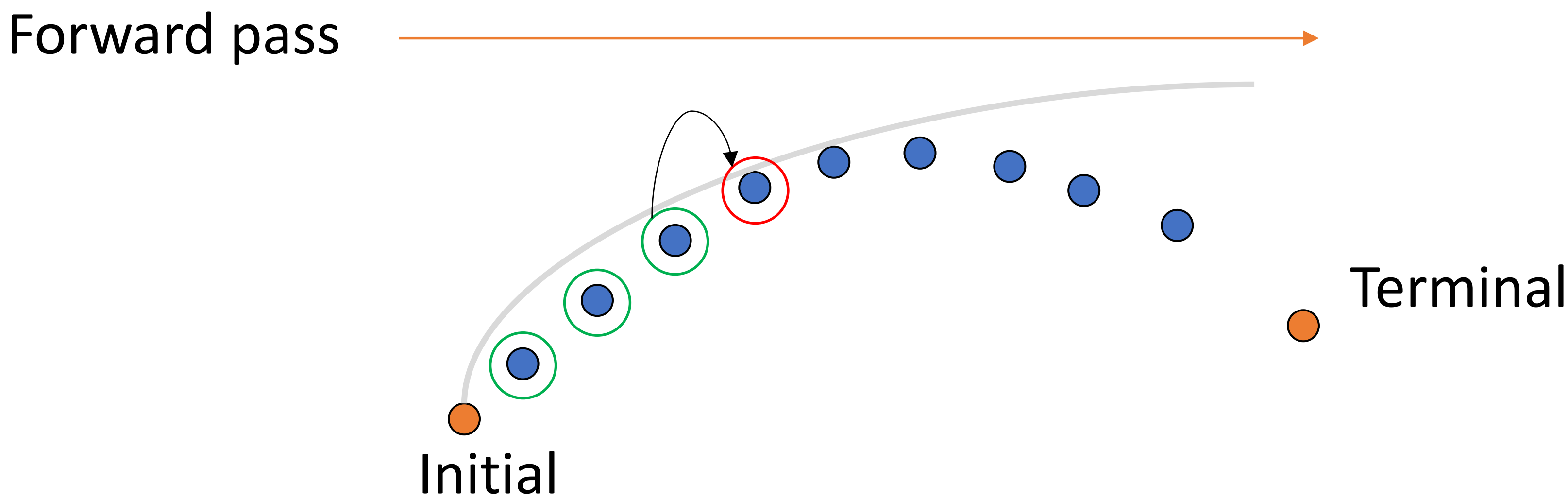
Forward pass



Do the forward pass for each alpha from 1 to 0

$$\hat{\mathbf{x}}(1) = \mathbf{x}(1)$$
$$\hat{\mathbf{u}}(i) = \mathbf{u}(i) + \alpha \mathbf{k}(i) + \mathbf{K}(i)(\hat{\mathbf{x}}(i) - \mathbf{x}(i))$$
$$\hat{\mathbf{x}}(i+1) = \mathbf{f}(\hat{\mathbf{x}}(i), \hat{\mathbf{u}}(i))$$

Forward pass



If success, decrease regularization term
If not, increase regularization term

$$\hat{\mathbf{x}}(1) = \mathbf{x}(1)$$

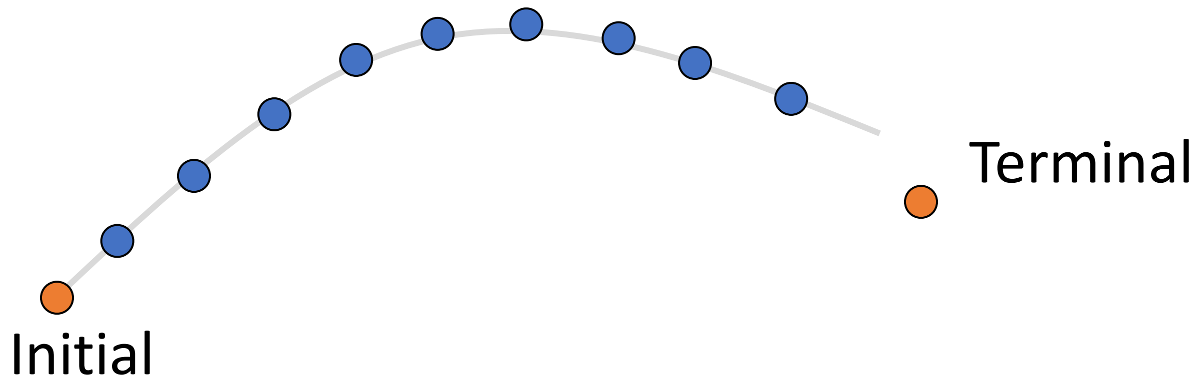
$$\hat{\mathbf{u}}(i) = \mathbf{u}(i) + \alpha \mathbf{k}(i) + \mathbf{K}(i)(\hat{\mathbf{x}}(i) - \mathbf{x}(i))$$

$$\hat{\mathbf{x}}(i+1) = \mathbf{f}(\hat{\mathbf{x}}(i), \hat{\mathbf{u}}(i))$$

Prepare for next iteration

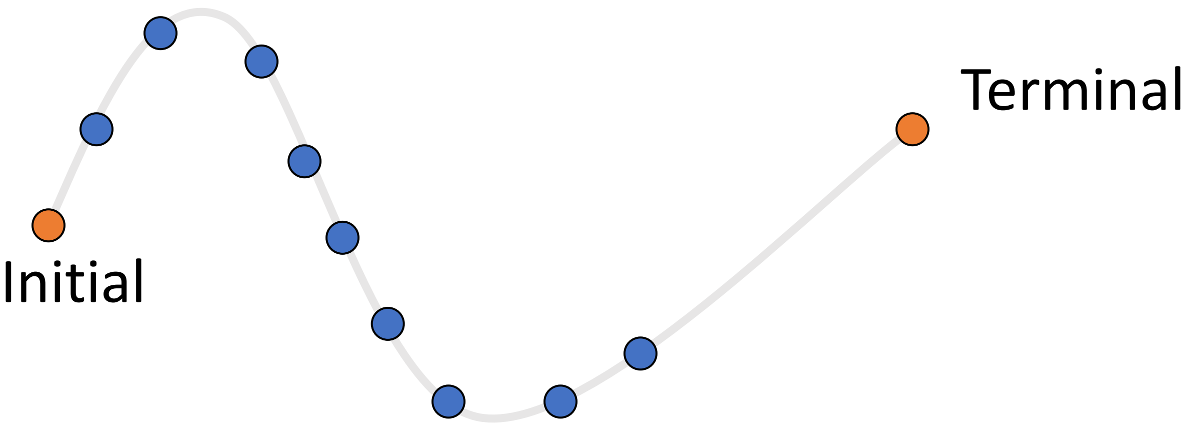
Dynamics

Cost function

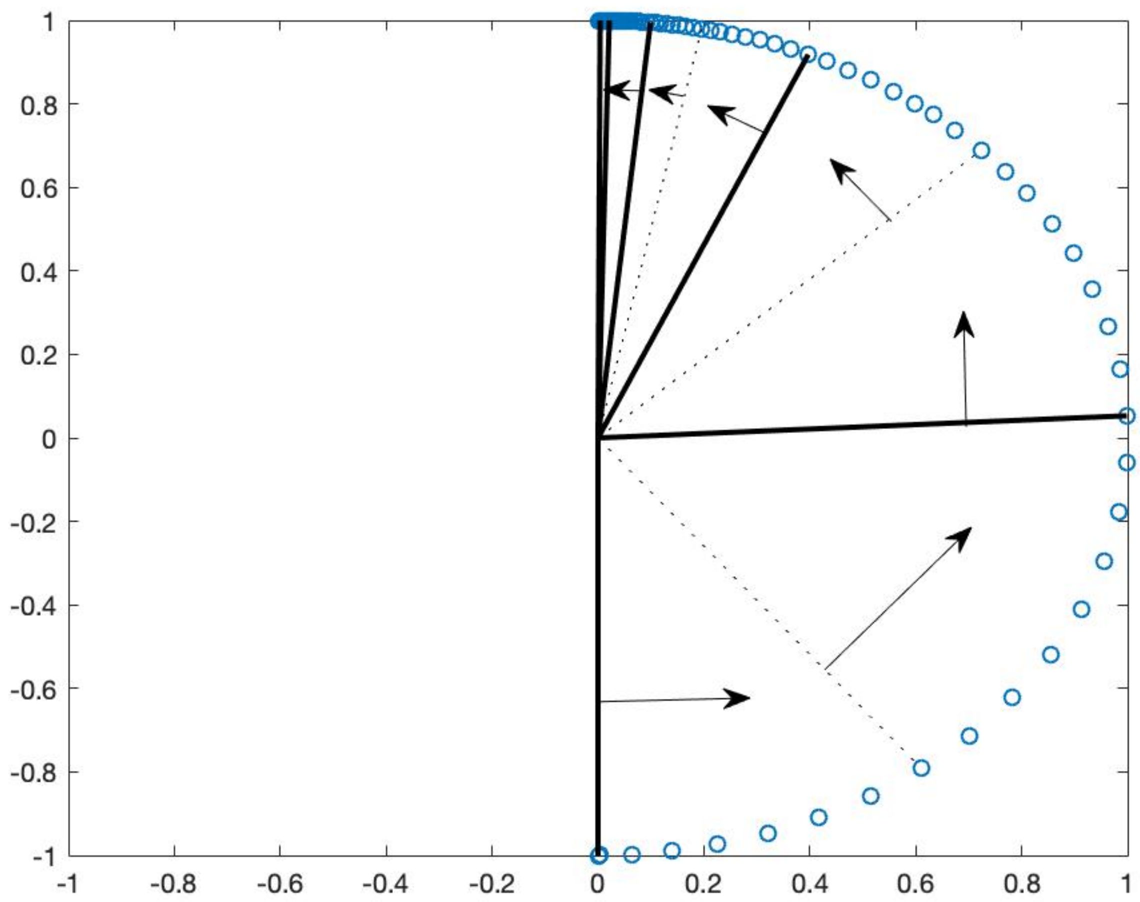


1. Load the dynamics & cost function

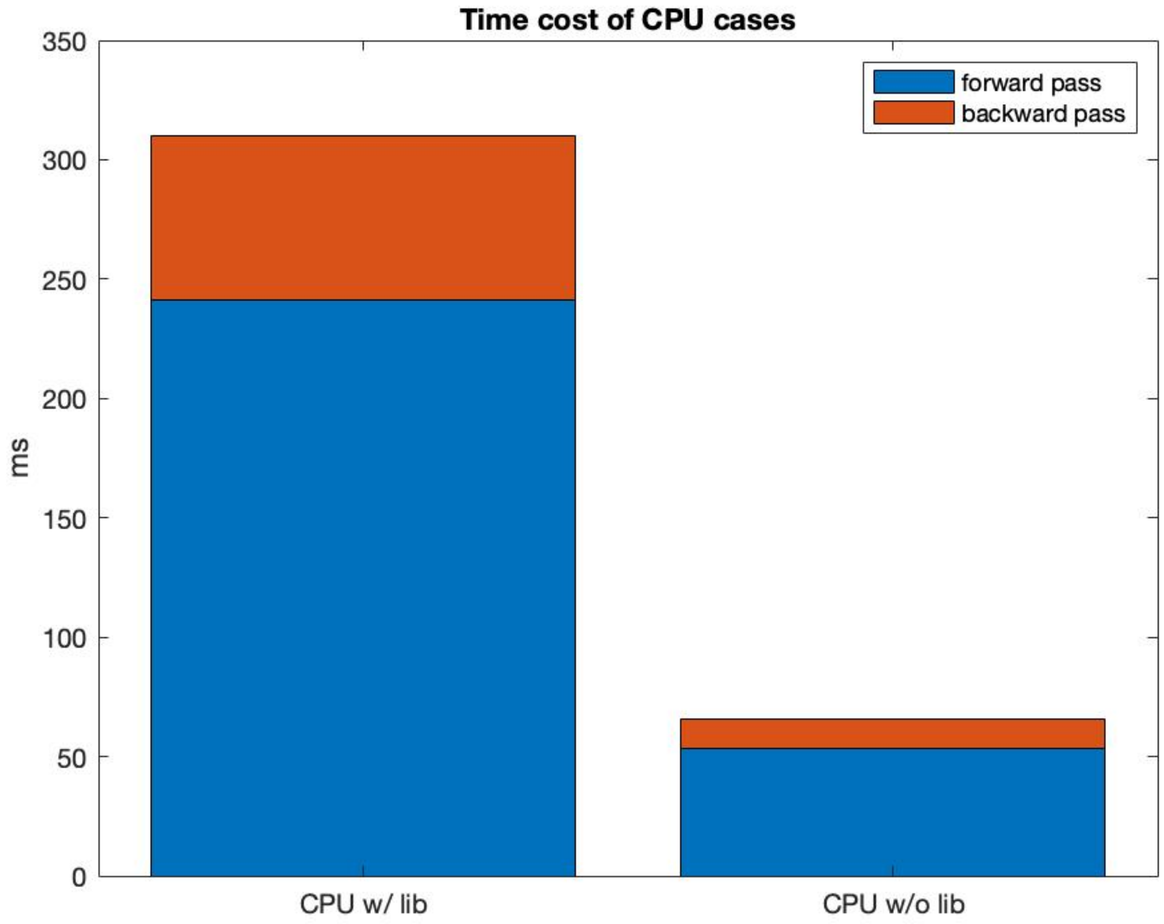
Final state



Illustrate the result



Comparison of time cost



Some parts can be parallel

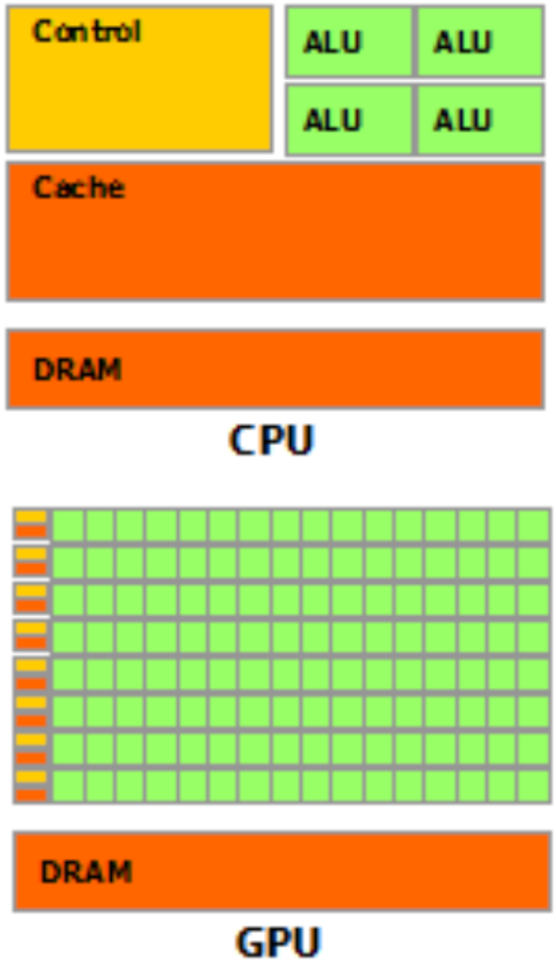
- Calculate forward pass with different alpha

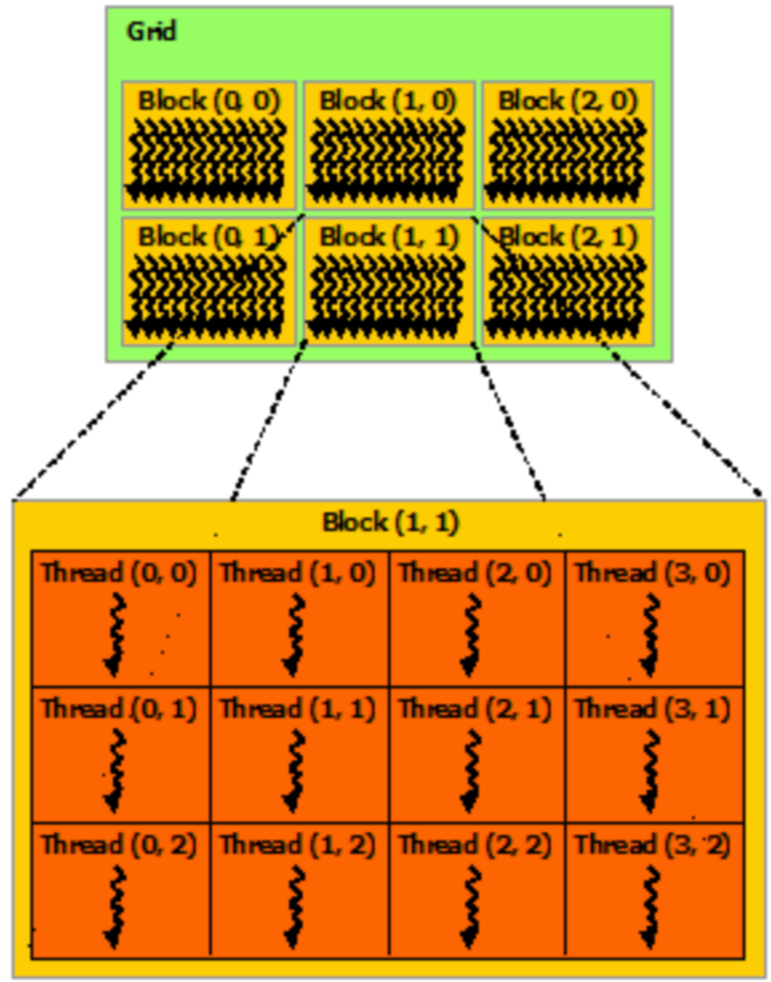
- Load dynamics and cost function

Basic ideas of GPU programming[2]

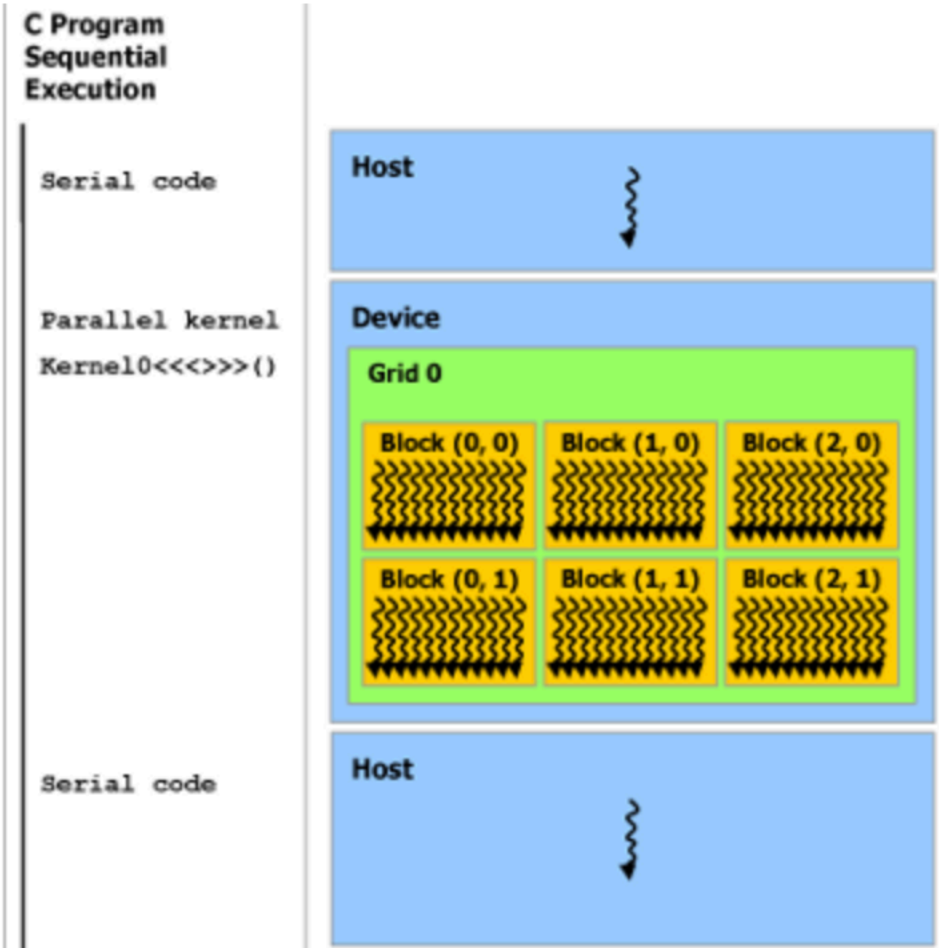
Host

Device





Basic ideas of GPU programming



1. Allocate the space on GPU
2. Initial the data on CPU and transfer it from host to device
3. Program on GPU is wrapped as kernel with decorator "__global__", and can be called from host
4. After finish on GPU, transfer data back to CPU

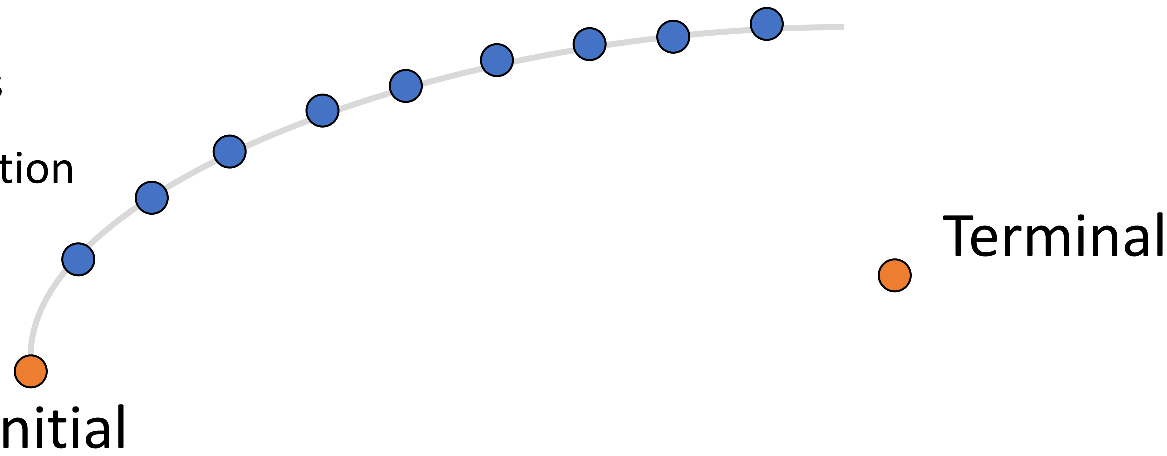
Initial state

Copy states and control from host to device

On device :Dynamics

On device: Cost function

$$U = \{u_0, \ldots, u_{N-1}\}$$



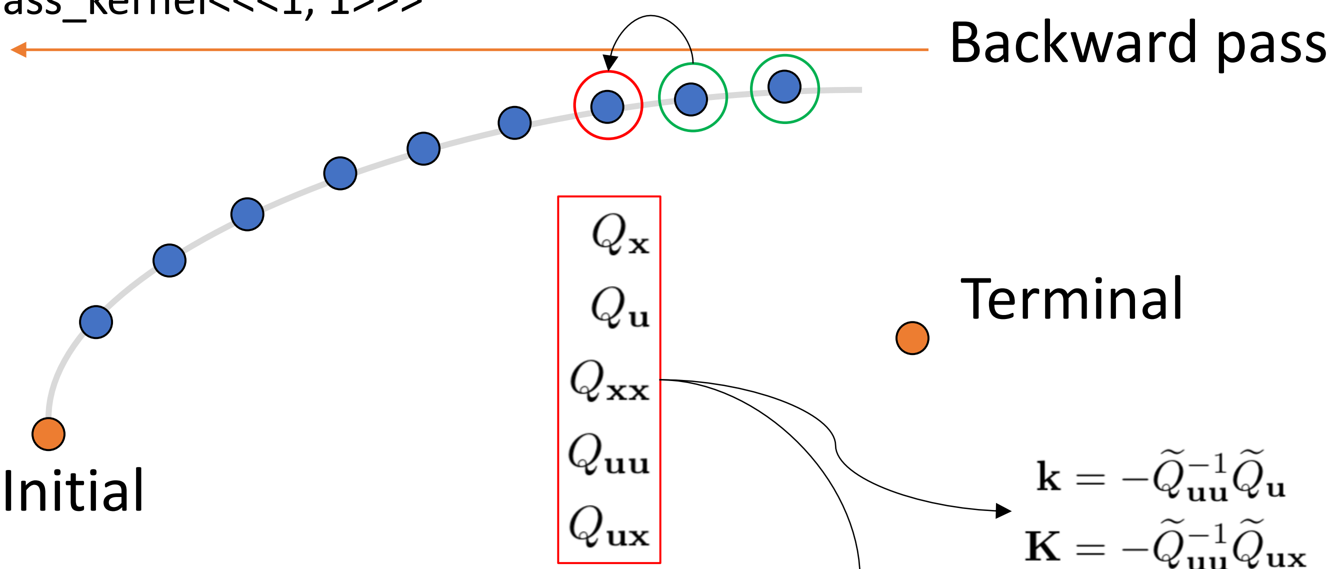
Launch load_variables_kern<<<1, NUM_STEPS>>>

1. Set the start point and goal
2. Set the number of steps
3. Initialize each states & controls
4. Load the dynamics & cost function

$$J = \frac{1}{2}(x_N - x_g)^T Q_N (x_N - x_g) \; + \sum_{k=0}^{N-1} \frac{1}{2}(x_k - x_g)^T Q (x_k - x_g) + \frac{1}{2} u_k^T R u_k$$

Backward pass

Launch backward_pass_kernel<<<1, 1>>>

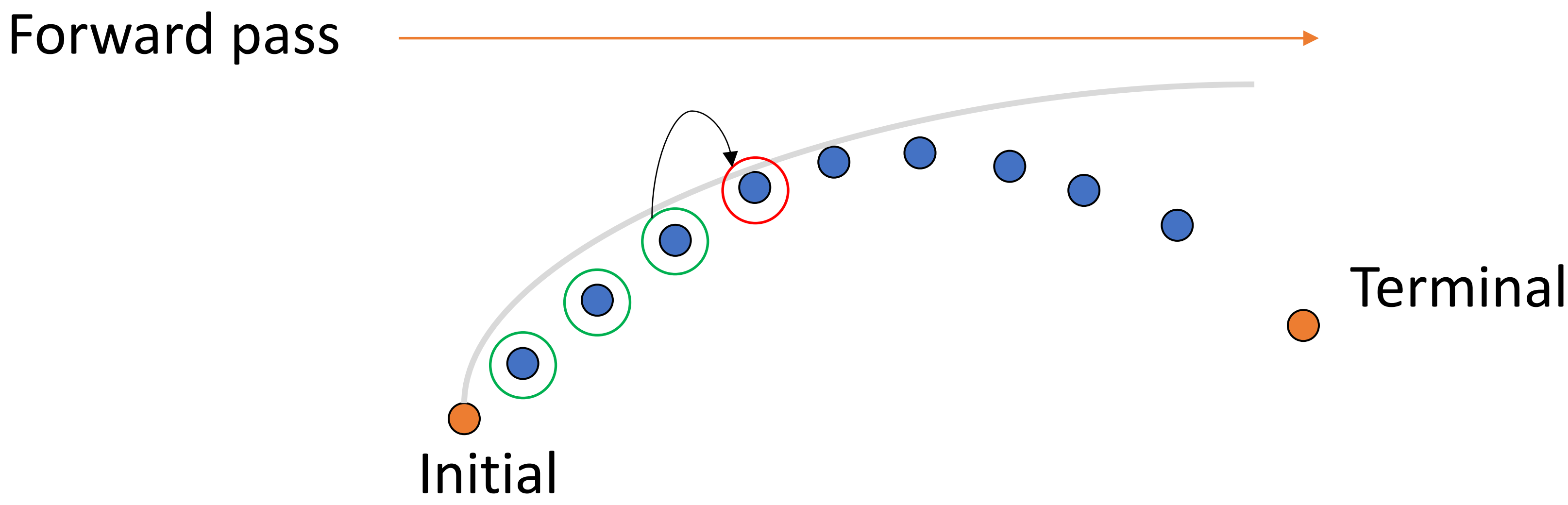


1. Calculate Q, inverse of Quu
2. Calculate control modification K and k
3. Update current value V

$$\Delta V(i)$$
$$V_{\mathbf{x}}(i)$$
$$V_{\mathbf{xx}}(i)$$

Forward pass

Launch `forward_pass_kernel<<<1, NUM_ALPHA>>>`

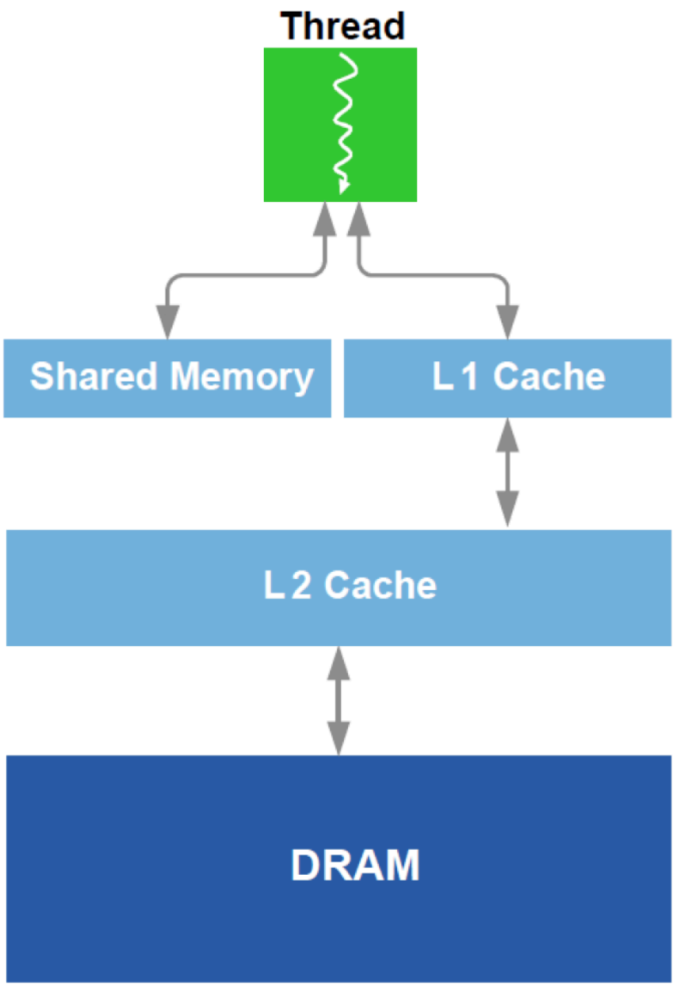


$$\hat{\mathbf{x}}(1) = \mathbf{x}(1)$$

$$\hat{\mathbf{u}}(i) = \mathbf{u}(i) + \alpha\mathbf{k}(i) + \mathbf{K}(i)(\hat{\mathbf{x}}(i) - \mathbf{x}(i))$$

$$\hat{\mathbf{x}}(i{+}1) = \mathbf{f}(\hat{\mathbf{x}}(i), \hat{\mathbf{u}}(i))$$

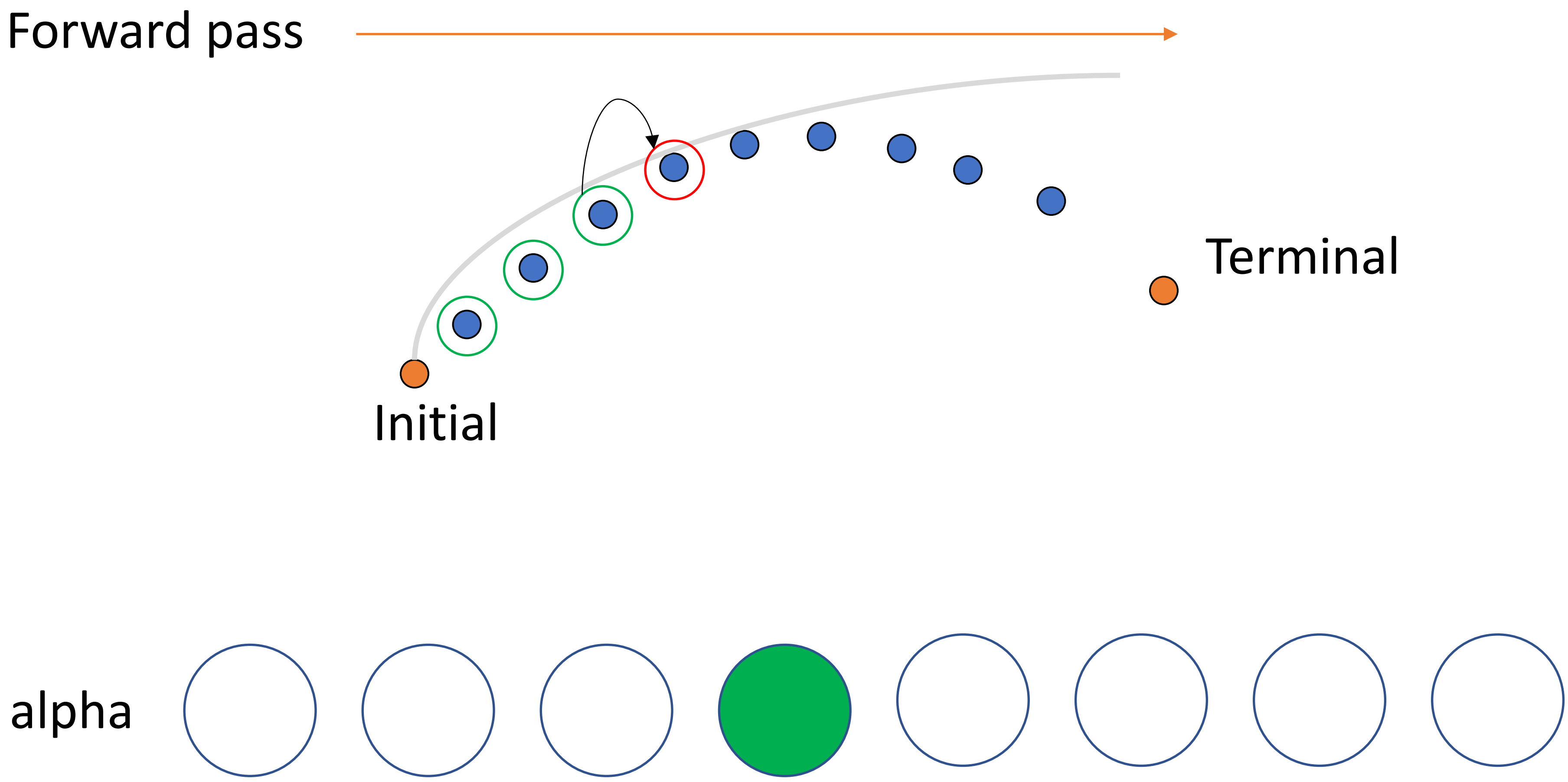
Forward pass



1. Load the previous states and control into shared memory
2. Run 32 threads at the same time
3. Synchronize in the end

$$\hat{\mathbf{u}}(i) = \mathbf{u}(i) + \alpha\mathbf{k}(i) + \mathbf{K}(i)(\hat{\mathbf{x}}(i) - \mathbf{x}(i))$$

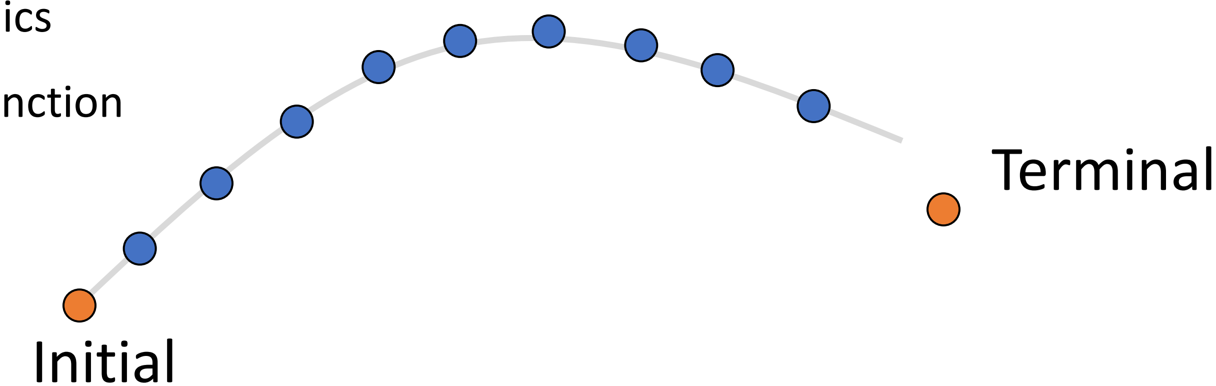
Forward pass



Prepare for next iteration

On device :Dynamics

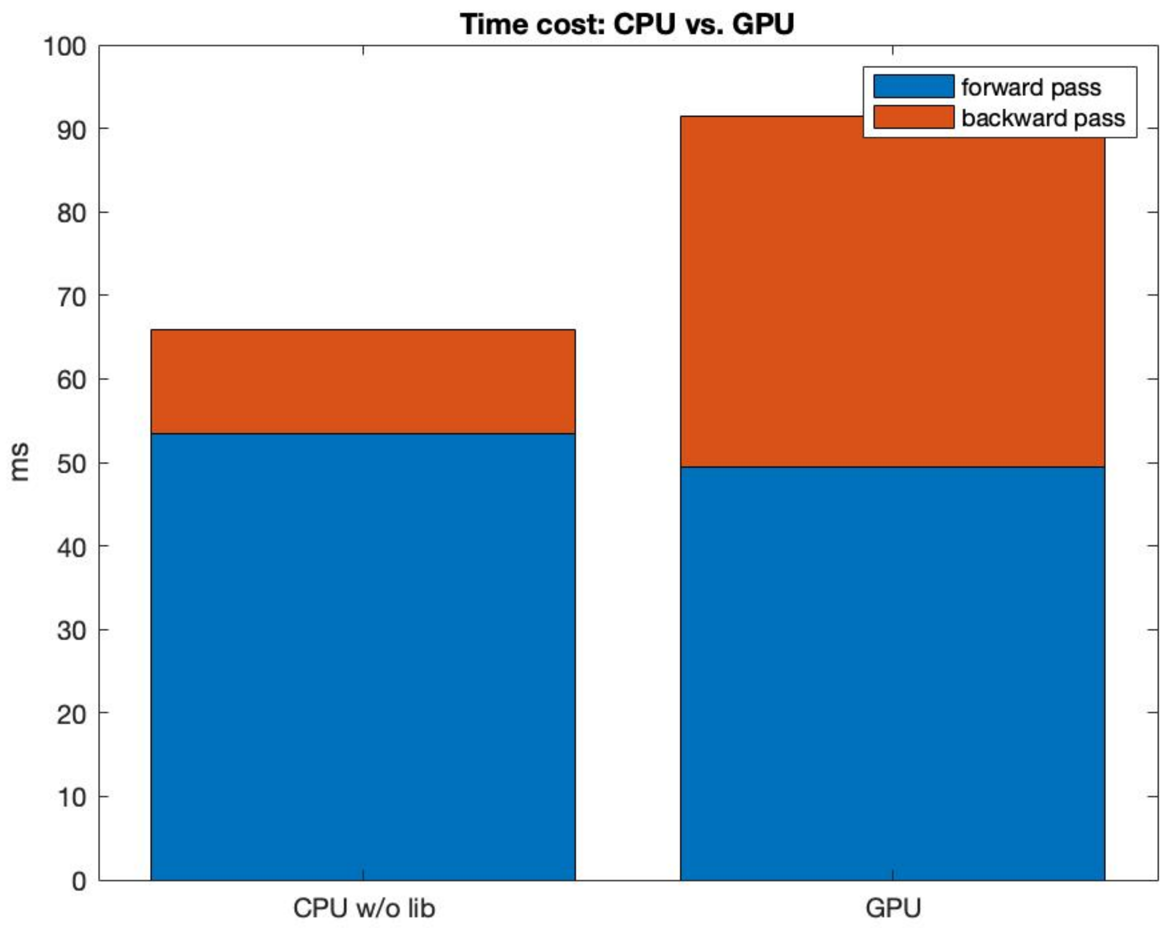
On device: Cost function



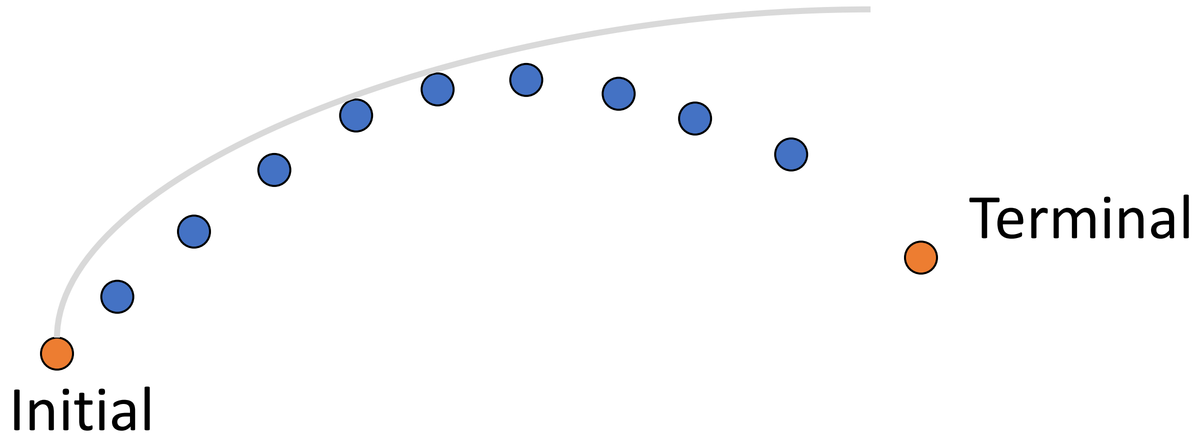
Launch load_variables_kern<<<1, NUM_STEPS>>>

1. Load the dynamics & cost function

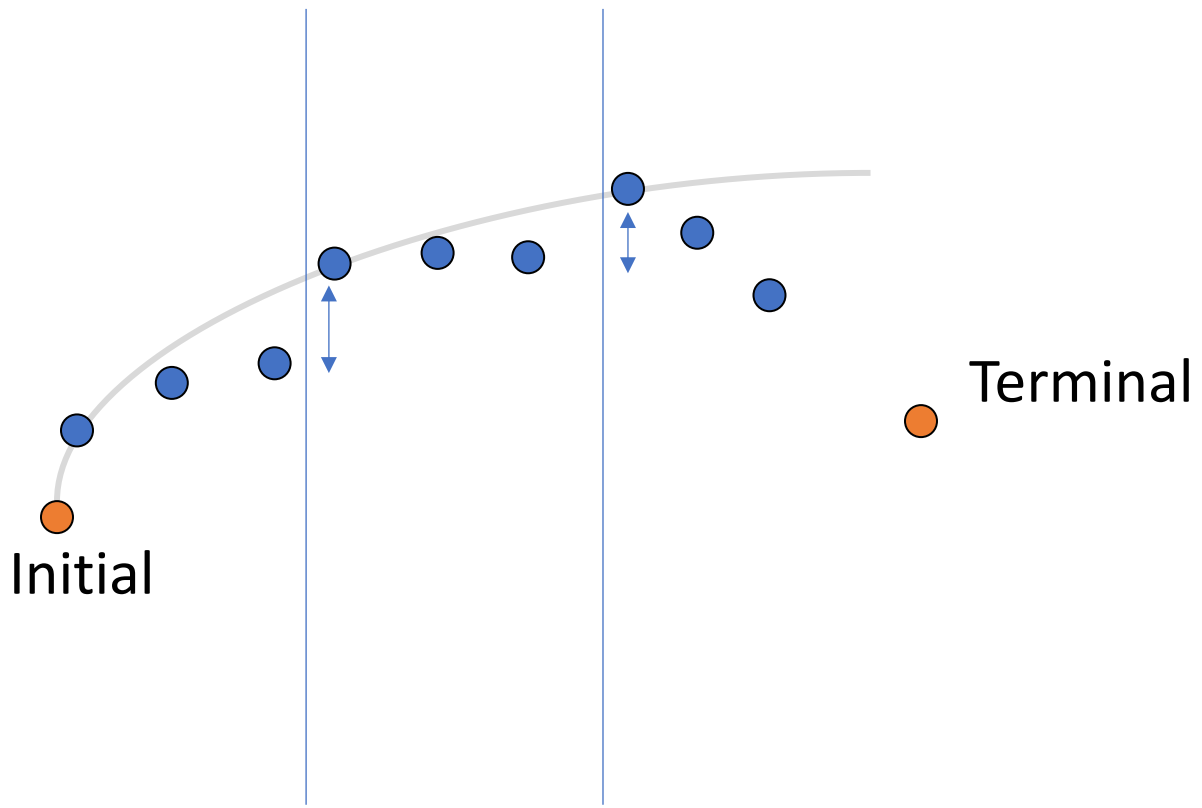
Comparison of time cost



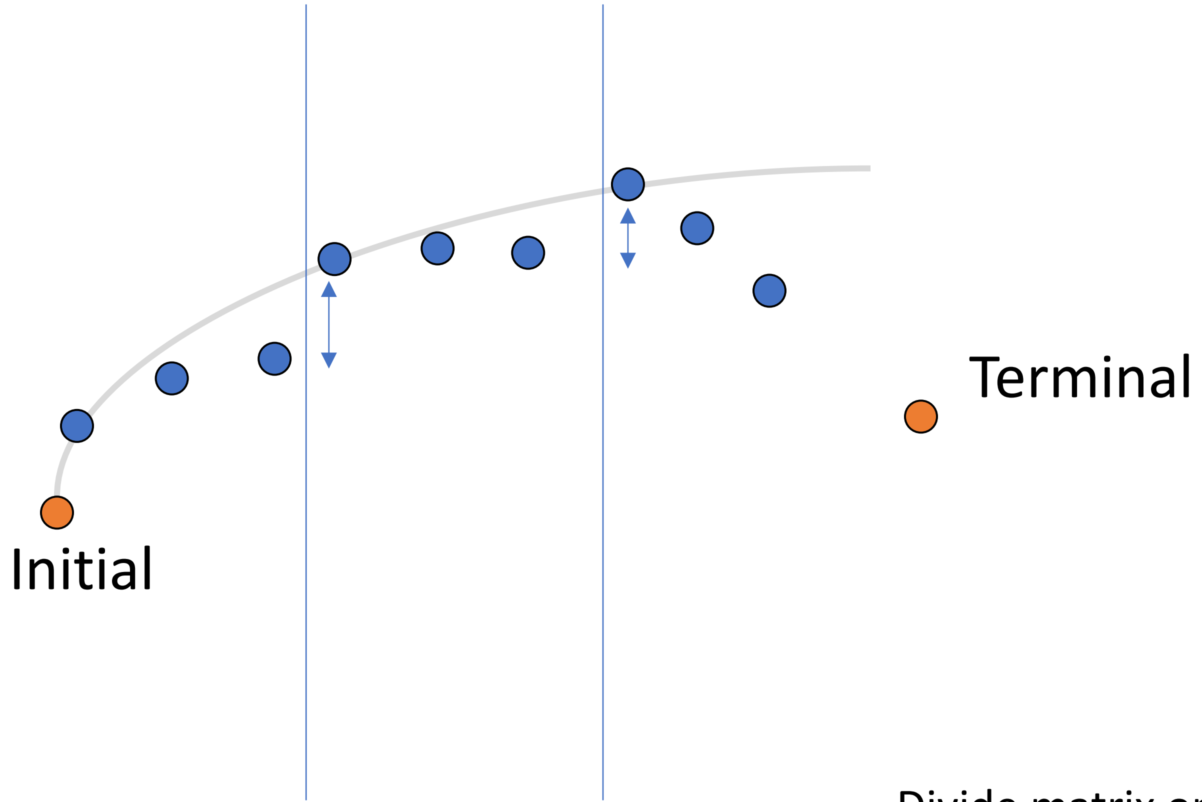
Further parallelization



Further parallelization



Further parallelization



backward_pass_kernel<<<NUM_BLOCKS, NUM_THREADS>>>

forward_pass_kernel<<<NUM_BLOCKS, NUM_ALPHA>>>

Divide matrix operations into different threads

Things need to be done before May 13th

- Divide trajectory into different blocks
- Implementing algorithms on quadrotors
- #Make the code into independent package for other program

III. References

1. 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems
   October 7-12, 2012. Vilamoura, Algarve, Portugal
2. https://docs.nvidia.com/cuda/cuda-c-programming-guide/
3. A Performance Analysis of Parallel Differential Dynamic Programming on a GPU, Brian Plancher and Scott Kuindersma, Harvard University, Cambridge MA 02138, USA

IV. Appendix

2. Differential Dynamic Programming

2. Differential Dynamic Programming

$$\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i)$$

Discrete-time dynamics

$$J_0(\mathbf{x}, \mathbf{U}) = \sum_{i=0}^{N-1} \ell(\mathbf{x}_i, \mathbf{u}_i) + \ell_f(\mathbf{x}_N)$$

Total cost

2. Differential Dynamic Programming

$$\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i)$$

$$J_0(\mathbf{x}, \mathbf{U}) = \sum_{i=0}^{N-1} \ell(\mathbf{x}_i, \mathbf{u}_i) + \ell_f(\mathbf{x}_N)$$

$$\mathbf{U}^*(\mathbf{x}) \equiv \operatorname*{argmin}_{\mathbf{U}} J_0(\mathbf{x}, \mathbf{U}).$$

Goal: minimizing control sequence

2. Differential Dynamic Programming

$$\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i)$$

$$J_0(\mathbf{x}, \mathbf{U}) = \sum_{i=0}^{N-1} \ell(\mathbf{x}_i, \mathbf{u}_i) + \ell_f(\mathbf{x}_N)$$

$$\mathbf{U}^*(\mathbf{x}) \equiv \underset{\mathbf{U}}{\operatorname{argmin}}\, J_0(\mathbf{x}, \mathbf{U}).$$

$$J_i(\mathbf{x}, \mathbf{U}_i) = \sum_{j=i}^{N-1} \ell(\mathbf{x}_j, \mathbf{u}_j) + \ell_f(\mathbf{x}_N).$$ Cost-to-go

2. Differential Dynamic Programming

$$\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i)$$

$$J_0(\mathbf{x}, \mathbf{U}) = \sum_{i=0}^{N-1} \ell(\mathbf{x}_i, \mathbf{u}_i) + \ell_f(\mathbf{x}_N)$$

$$\mathbf{U}^*(\mathbf{x}) \equiv \operatorname*{argmin}_{\mathbf{U}} J_0(\mathbf{x}, \mathbf{U}).$$

$$J_i(\mathbf{x}, \mathbf{U}_i) = \sum_{j=i}^{N-1} \ell(\mathbf{x}_j, \mathbf{u}_j) + \ell_f(\mathbf{x}_N).$$

$$V(\mathbf{x}, i) = \min_{\mathbf{u}}[\ell(\mathbf{x}, \mathbf{u}) + V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i+1)]$$ Minimizations over a single control

2. Differential Dynamic Programming

$$V(\mathbf{x}, i) = \min_{\mathbf{u}}[\ell(\mathbf{x}, \mathbf{u}) + V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i+1)]$$

2. Differential Dynamic Programming

$$V(\mathbf{x}, i) = \min_{\mathbf{u}}[\ell(\mathbf{x}, \mathbf{u}) + V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i{+}1)]$$

Perturbations around i-th (x, u) pair

$$\begin{aligned} Q(\delta\mathbf{x}, \delta\mathbf{u}) &= \ell(\mathbf{x} + \delta\mathbf{x}, \mathbf{u} + \delta\mathbf{u}, i) - \ell(\mathbf{x}, \mathbf{u}, i) \\ &\quad + V(\mathbf{f}(\mathbf{x} + \delta\mathbf{x}, \mathbf{u} + \delta\mathbf{u}), i{+}1) - V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i{+}1) \end{aligned}$$

2. Differential Dynamic Programming

$$V(\mathbf{x}, i) = \min_{\mathbf{u}}[\ell(\mathbf{x}, \mathbf{u}) + V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i{+}1)]$$

$$\begin{aligned} Q(\delta\mathbf{x}, \delta\mathbf{u}) = &\ \ell(\mathbf{x} + \delta\mathbf{x}, \mathbf{u} + \delta\mathbf{u}, i) - \ell(\mathbf{x}, \mathbf{u}, i) \\ &+ V(\mathbf{f}(\mathbf{x} + \delta\mathbf{x}, \mathbf{u} + \delta\mathbf{u}), i{+}1) - V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i{+}1) \end{aligned}$$

Expand to second order

$$\approx \frac{1}{2} \begin{bmatrix} 1 \\ \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} 0 & Q_{\mathbf{x}}^{\mathsf{T}} & Q_{\mathbf{u}}^{\mathsf{T}} \\ Q_{\mathbf{x}} & Q_{\mathbf{xx}} & Q_{\mathbf{xu}} \\ Q_{\mathbf{u}} & Q_{\mathbf{ux}} & Q_{\mathbf{uu}} \end{bmatrix} \begin{bmatrix} 1 \\ \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}$$

2. Differential Dynamic Programming

$$\approx \frac{1}{2} \begin{bmatrix} 1 \\ \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} 0 & Q_{\mathbf{x}}^{\mathsf{T}} & Q_{\mathbf{u}}^{\mathsf{T}} \\ Q_{\mathbf{x}} & Q_{\mathbf{xx}} & Q_{\mathbf{xu}} \\ Q_{\mathbf{u}} & Q_{\mathbf{ux}} & Q_{\mathbf{uu}} \end{bmatrix} \begin{bmatrix} 1 \\ \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}$$

2. Differential Dynamic Programming

$$\approx \frac{1}{2} \begin{bmatrix} 1 \\ \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} 0 & Q_{\mathbf{x}}^{\mathsf{T}} & Q_{\mathbf{u}}^{\mathsf{T}} \\ Q_{\mathbf{x}} & Q_{\mathbf{xx}} & Q_{\mathbf{xu}} \\ Q_{\mathbf{u}} & Q_{\mathbf{ux}} & Q_{\mathbf{uu}} \end{bmatrix} \begin{bmatrix} 1 \\ \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}$$

$$\begin{aligned} Q_{\mathbf{x}} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}} \\ Q_{\mathbf{u}} &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}} \\ Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}} \\ Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\ Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}. \end{aligned}$$

2. Differential Dynamic Programming

$$\approx \frac{1}{2}\begin{bmatrix}1\\ \delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}0 & Q_{\mathbf{x}}^{\mathsf{T}} & Q_{\mathbf{u}}^{\mathsf{T}}\\ Q_{\mathbf{x}} & Q_{\mathbf{xx}} & Q_{\mathbf{xu}}\\ Q_{\mathbf{u}} & Q_{\mathbf{ux}} & Q_{\mathbf{uu}}\end{bmatrix}\begin{bmatrix}1\\ \delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}$$

$$\begin{aligned}
Q_{\mathbf{x}} &= \ell_{\mathbf{x}} \; + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}}\\
Q_{\mathbf{u}} &= \ell_{\mathbf{u}} \; + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}}\\
Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}}\\
Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}\\
Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.
\end{aligned}$$

$$\delta\mathbf{u}^* = \underset{\delta\mathbf{u}}{\operatorname{argmin}}\, Q(\delta\mathbf{x}, \delta\mathbf{u}) = -Q_{\mathbf{uu}}^{-1}(Q_{\mathbf{u}} + Q_{\mathbf{ux}}\delta\mathbf{x})$$

Minimizing wrt u

2. Differential Dynamic Programming

$$\approx \frac{1}{2} \begin{bmatrix} 1 \\ \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} 0 & Q_{\mathbf{x}}^{\mathsf{T}} & Q_{\mathbf{u}}^{\mathsf{T}} \\ Q_{\mathbf{x}} & Q_{\mathbf{xx}} & Q_{\mathbf{xu}} \\ Q_{\mathbf{u}} & Q_{\mathbf{ux}} & Q_{\mathbf{uu}} \end{bmatrix} \begin{bmatrix} 1 \\ \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}$$

$$\begin{aligned} Q_{\mathbf{x}} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}} \\ Q_{\mathbf{u}} &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}} \\ Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}} \\ Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\ Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}. \end{aligned}$$

open-loop term $\mathbf{k} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}}$

feedback gain term $\mathbf{K} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}$

$$\delta\mathbf{u}^* = \operatorname*{argmin}_{\delta\mathbf{u}} Q(\delta\mathbf{x}, \delta\mathbf{u}) = -Q_{\mathbf{uu}}^{-1}(Q_{\mathbf{u}} + Q_{\mathbf{ux}}\delta\mathbf{x})$$

2. Differential Dynamic Programming

$$\approx \frac{1}{2}\begin{bmatrix}1\\ \delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}0 & Q_{\mathbf{x}}^{\mathsf{T}} & Q_{\mathbf{u}}^{\mathsf{T}}\\ Q_{\mathbf{x}} & Q_{\mathbf{xx}} & Q_{\mathbf{xu}}\\ Q_{\mathbf{u}} & Q_{\mathbf{ux}} & Q_{\mathbf{uu}}\end{bmatrix}\begin{bmatrix}1\\ \delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}$$

$$\begin{aligned}
Q_{\mathbf{x}} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}}\\
Q_{\mathbf{u}} &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}}\\
Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}}\mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}}\cdot\mathbf{f}_{\mathbf{xx}}\\
Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}}\mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}}\cdot\mathbf{f}_{\mathbf{uu}}\\
Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}}\mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}}\cdot\mathbf{f}_{\mathbf{ux}}.
\end{aligned}$$

$$\delta\mathbf{u}^* = \underset{\delta\mathbf{u}}{\operatorname{argmin}}\, Q(\delta\mathbf{x},\delta\mathbf{u}) = -Q_{\mathbf{uu}}^{-1}(Q_{\mathbf{u}} + Q_{\mathbf{ux}}\delta\mathbf{x})$$

2. Differential Dynamic Programming

$$\approx \frac{1}{2}\begin{bmatrix}1\\ \delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}0 & Q_{\mathbf{x}}^{\mathsf{T}} & Q_{\mathbf{u}}^{\mathsf{T}}\\ Q_{\mathbf{x}} & Q_{\mathbf{xx}} & Q_{\mathbf{xu}}\\ Q_{\mathbf{u}} & Q_{\mathbf{ux}} & Q_{\mathbf{uu}}\end{bmatrix}\begin{bmatrix}1\\ \delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}$$

$$\begin{aligned}\Delta V(i) &= -\tfrac{1}{2}Q_{\mathbf{u}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{u}}\\ V_{\mathbf{x}}(i) &= Q_{\mathbf{x}} - Q_{\mathbf{u}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{ux}}\\ V_{\mathbf{xx}}(i) &= Q_{\mathbf{xx}} - Q_{\mathbf{xu}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{ux}}.\end{aligned}$$

$$\begin{aligned}Q_{\mathbf{x}} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}}V'_{\mathbf{x}}\\ Q_{\mathbf{u}} &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}V'_{\mathbf{x}}\\ Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}}V'_{\mathbf{xx}}\mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}}\cdot\mathbf{f}_{\mathbf{xx}}\\ Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}V'_{\mathbf{xx}}\mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}}\cdot\mathbf{f}_{\mathbf{uu}}\\ Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}V'_{\mathbf{xx}}\mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}}\cdot\mathbf{f}_{\mathbf{ux}}.\end{aligned}$$

$$\delta\mathbf{u}^* = \underset{\delta\mathbf{u}}{\operatorname{argmin}}\, Q(\delta\mathbf{x},\delta\mathbf{u}) = -Q_{\mathbf{uu}}^{-1}(Q_{\mathbf{u}} + Q_{\mathbf{ux}}\delta\mathbf{x})$$

2. Differential Dynamic Programming

$$Q_{\mathbf{x}} = \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{u}} = \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{xx}} = \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}}$$
$$Q_{\mathbf{uu}} = \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}$$
$$Q_{\mathbf{ux}} = \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.$$

2. Differential Dynamic Programming

$$
\begin{aligned}
Q_{\mathbf{x}} &= \boxed{\ell_{\mathbf{x}}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{u}} &= \boxed{\ell_{\mathbf{u}}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{xx}} &= \boxed{\ell_{\mathbf{xx}}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}} \\
Q_{\mathbf{uu}} &= \boxed{\ell_{\mathbf{uu}}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\
Q_{\mathbf{ux}} &= \boxed{\ell_{\mathbf{ux}}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.
\end{aligned}
$$

Cost function

2. Differential Dynamic Programming

$$Q_{\mathbf{x}} = \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}}$$

$$Q_{\mathbf{u}} = \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}}$$

$$Q_{\mathbf{xx}} = \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}}$$

$$Q_{\mathbf{uu}} = \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}$$

$$Q_{\mathbf{ux}} = \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.$$

Dynamics

2. Differential Dynamic Programming

$$
\begin{aligned}
Q_{\mathbf{x}} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{u}} &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}} \\
Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\
Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.
\end{aligned}
$$

Next value

2. Differential Dynamic Programming

$$
\begin{aligned}
Q_{\mathbf{x}} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{u}} &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}} \\
Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\
Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.
\end{aligned}
$$

Cost function, dynamics and next value

$\downarrow$

Q

2. Differential Dynamic Programming

$$Q_{\mathbf{x}} = \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{u}} = \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{xx}} = \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}}$$
$$Q_{\mathbf{uu}} = \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}$$
$$Q_{\mathbf{ux}} = \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.$$

$$\downarrow$$

$$\mathbf{k} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}}$$
$$\mathbf{K} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}$$

Cost function, dynamics and next value

$$\downarrow$$

Q

$$\downarrow$$

Control modifications

2. Differential Dynamic Programming

$$Q_{\mathbf{x}} = \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}}$$

$$Q_{\mathbf{u}} = \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}}$$

$$Q_{\mathbf{xx}} = \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}}$$

$$Q_{\mathbf{uu}} = \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}$$

$$Q_{\mathbf{ux}} = \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.$$

$$\mathbf{k} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}}$$

$$\mathbf{K} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}$$

$$\Delta V(i) = -\tfrac{1}{2} Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}}$$

$$V_{\mathbf{x}}(i) = Q_{\mathbf{x}} - Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}$$

$$V_{\mathbf{xx}}(i) = Q_{\mathbf{xx}} - Q_{\mathbf{xu}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}.$$

Cost function, dynamics and next value

↓

Q

↓

Control modifications

Local quadratic models of V(i)

2. Differential Dynamic Programming

$$Q_{\mathbf{x}} = \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{u}} = \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{xx}} = \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}}$$
$$Q_{\mathbf{uu}} = \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}$$
$$Q_{\mathbf{ux}} = \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.$$

Cost function, dynamics and next value

↓

Q

↓

Control modifications

Local quadratic models of V(i)

$$\mathbf{k} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}}$$
$$\mathbf{K} = -Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}$$

$$\Delta V(i) = -\tfrac{1}{2} Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}}$$
$$V_{\mathbf{x}}(i) = Q_{\mathbf{x}} - Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}$$
$$V_{\mathbf{xx}}(i) = Q_{\mathbf{xx}} - Q_{\mathbf{xu}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}.$$

Backward Pass

2. Differential Dynamic Programming

$$\hat{\mathbf{x}}(1) = \mathbf{x}(1)$$
$$\hat{\mathbf{u}}(i) = \mathbf{u}(i) + \mathbf{k}(i) + \mathbf{K}(i)(\hat{\mathbf{x}}(i) - \mathbf{x}(i))$$
$$\hat{\mathbf{x}}(i+1) = \mathbf{f}(\hat{\mathbf{x}}(i), \hat{\mathbf{u}}(i))$$

Forward Pass

3. Iterative LQR

3. Iterative LQR

$$
\begin{aligned}
Q_{\mathbf{x}} &= \ell_{\mathbf{x}} \ +\mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{u}} &= \ell_{\mathbf{u}} \ +\mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}} \\
Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\
Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.
\end{aligned}
$$

3. Iterative LQR

$$
\begin{aligned}
Q_{\mathbf{x}} &= \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{u}} &= \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}} \\
Q_{\mathbf{xx}} &= \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}} \\
Q_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\
Q_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.
\end{aligned}
$$

4. Regularization

4. Regularization

$$
\begin{aligned}
\widetilde{Q}_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}(V'_{\mathbf{xx}} + \mu \mathbf{I}_n)\mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\
\widetilde{Q}_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}(V'_{\mathbf{xx}} + \mu \mathbf{I}_n)\mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}} \\
\mathbf{k} &= -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{u}} \\
\mathbf{K} &= -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{ux}}
\end{aligned}
$$

4. Regularization

$$\widetilde{Q}_{\mathbf{uu}} = \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}(V'_{\mathbf{xx}} + \mu \mathbf{I}_n)\mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}$$

$$\widetilde{Q}_{\mathbf{ux}} = \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}(V'_{\mathbf{xx}} + \mu \mathbf{I}_n)\mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}$$

$$\mathbf{k} = -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{u}}$$

$$\mathbf{K} = -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{ux}}$$

As role of Levenberg-Marquardt parameter

4. Regularization

$$
\begin{aligned}
\widetilde{Q}_{\mathbf{uu}} &= \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}(V'_{\mathbf{xx}} + \mu \mathbf{I}_n)\mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}} \\
\widetilde{Q}_{\mathbf{ux}} &= \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}}(V'_{\mathbf{xx}} + \mu \mathbf{I}_n)\mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}} \\
\mathbf{k} &= -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{u}} \\
\mathbf{K} &= -\widetilde{Q}_{\mathbf{uu}}^{-1}\widetilde{Q}_{\mathbf{ux}}
\end{aligned}
$$

$$
\begin{aligned}
\Delta V(i) &= \quad + \tfrac{1}{2}\mathbf{k}^{\mathsf{T}} Q_{\mathbf{uu}} \mathbf{k} + \mathbf{k}^{\mathsf{T}} Q_{\mathbf{u}} \\
V_{\mathbf{x}}(i) &= Q_{\mathbf{x}} + \mathbf{K}^{\mathsf{T}} Q_{\mathbf{uu}} \mathbf{k} + \mathbf{K}^{\mathsf{T}} Q_{\mathbf{u}} + Q_{\mathbf{ux}}^{\mathsf{T}} \mathbf{k} \\
V_{\mathbf{xx}}(i) &= Q_{\mathbf{xx}} + \mathbf{K}^{\mathsf{T}} Q_{\mathbf{uu}} \mathbf{K} + \mathbf{K}^{\mathsf{T}} Q_{\mathbf{ux}} + Q_{\mathbf{ux}}^{\mathsf{T}} \mathbf{K}.
\end{aligned}
$$