Event-Triggered Control Design with Performance Barrier

Pio Ong and Jorge Cortés

Mechanical and Aerospace Engineering
University of California, San Diego
http://carmenere.ucsd.edu/jorge

57th IEEE Conference on Decision and Control:
Robust Event-Triggered Control
Miami, Florida
December 17-19, 2018

Motivating Example

Example from [P. Tabuada 2007], a well-cited paper in ET control

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

Motivating Example

Example from [P. Tabuada 2007], a well-cited paper in ET control

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

Continuous time: Pick $u = x_1 - 4x_2$

Motivating Example

Example from [P. Tabuada 2007], a well-cited paper in ET control

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

Continuous time: Pick $u = x_1 - 4x_2$

Event-triggered: Update u above when $\|e\| = \sigma \|x\|$, $e = x - x_k$

σ is a design parameter.
Lower $\rightarrow$ better **performance**
Higher $\rightarrow$ less trigger, **conserving resources**

Key Idea to Take Away from My Talk

- **Motivation:**
  1. Unclear on how to tune the design parameter to create a balance between trigger frequency and performance
  2. Standard ET design scheme can be inefficient in achieving desired performance
- **Assumption:**
  1. Desired performance can be achieved in continuous time
- **Approach:**
  1. Throw away the Lyapunov's criterion for stability, i.e. $\dot{V} \leq 0$
  2. Allow $\dot{V} > 0$
  3. Incorporate performance requirement into the trigger condition

Outline

Event-triggered control design overview:

- Linear system example
- Identify inefficiencies in satisfying a given desired performance

Our design:

- Incorporating performance requirement
  - Use barrier concept
- Advantages
- Apply our new design idea to distributed cases

Wrapping Up My Talk

- Simulations
- Conclusion and future ideas

Design Parameter

How might σ be picked?

From earlier example: Lyapunov function

$$V(x) = x^T \begin{bmatrix} 1 & 1/4 \\ 1/4 & 1 \end{bmatrix} x \implies \dot{V} \leq -0.44\|x\|^2 + 8\|e\|\|x\|$$

using ET control, $\dot{V} \leq -(0.44 - 8\sigma)\|x^2\|$, $\sigma = 0.05$ was picked, but why?

Design Parameter

How might σ be picked?

From earlier example: Lyapunov function

$$V(x) = x^T \begin{bmatrix} 1 & 1/4 \\ 1/4 & 1 \end{bmatrix} x \implies \dot{V} \leq -0.44\|x\|^2 + 8\|e\|\|x\|$$

using ET control, $\dot{V} \leq -(0.44 - 8\sigma)\|x^2\|$, $\sigma = 0.05$ was picked, but why?

Maybe because this σ guarantees the **performance** of

$$\dot{V} \leq -\frac{0.04}{3/4} V => V(x(t)) \leq V(x_0)\exp(-0.032t)$$

Design Parameter

How might σ be picked?

From earlier example: Lyapunov function

$$V(x) = x^T \begin{bmatrix} 1 & 1/4 \\ 1/4 & 1 \end{bmatrix} x \implies \dot{V} \leq -0.44\|x\|^2 + 8\|e\|\|x\|$$

using ET control, $\dot{V} \leq -(0.44 - 8\sigma)\|x^2\|$, $\sigma = 0.05$ was picked, but why?

Maybe because this σ guarantees the **performance** of

$$\dot{V} \leq -\frac{0.04}{3/4} V => V(x(t)) \leq V(x_0)\exp(-0.032t)$$

We can reverse the process.
Given performance specification $S(t) \leq V(x_0)\exp(-rt)$, one can find σ

Assumptions made

Assumptions so that we can reverse the process

Assumptions made

Assumptions so that we can reverse the process

- Known ISS Lyapunov function (same [P. Tabuada 2007])
  - $\dot{V} \leq -\alpha(\|x\|) + \gamma(\|e\|)$

Assumptions made

Assumptions so that we can reverse the process

- Known ISS Lyapunov function (same [P. Tabuada 2007])
  - $\dot{V} \leq -\alpha(\|x\|) + \gamma(\|e\|)$
- Given specification function
  - $\dot{S} = -h(S), \; S(x_0, 0) \geq V(x_0)$ where h locally Lipschitz, class $\mathcal{K}$
  - Note: earlier, special case $\dot{S} = -rS$

Assumptions made

Assumptions so that we can reverse the process

- Known ISS Lyapunov function (same [P. Tabuada 2007])
  - $\dot{V} \leq -\alpha(\|x\|) + \gamma(\|e\|)$
- Given specification function
  - $\dot{S} = -h(S), \; S(x_0, 0) \geq V(x_0)$ where h locally Lipschitz, class $\mathcal{K}$
  - Note: earlier, special case $\dot{S} = -rS$
- Performance achievable in continuous time
  - $-\alpha(\|x\|) < -h(\bar{V}(x))$

Derivative based Trigger

Design idea: **forget σ**, make $\dot{V} \leq -h(V)$, then $V \leq S$ (Comparison Lemma)

Derivative-based ET

$$t_{k+1} = \min_{t} \left\{ t > t_k \mid g(x(t), e(t)) + h(\bar{V}(x(t))) = 0 \right\}$$

where $\mathcal{L}_f V(x) \leq g(x, e) \leq -\alpha(\|x\|) + \gamma(\|e\|)$

Derivative based Trigger

Design idea: **forget σ**, make $\dot{V} \leq -h(V)$, then $V \leq S$ (Comparison Lemma)

Derivative-based ET

$$t_{k+1} = \min_t \left\{ t > t_k \mid g(x(t), e(t)) + h(\bar{V}(x(t))) = 0 \right\}$$

where $\mathcal{L}_f V(x) \leq g(x, e) \leq -\alpha(\|x\|) + \gamma(\|e\|)$

Ex. update u when $-0.44\|x\|^2 + 8\|e\|\|x\| + 0.032V(x) = 0$

Derivative based Trigger

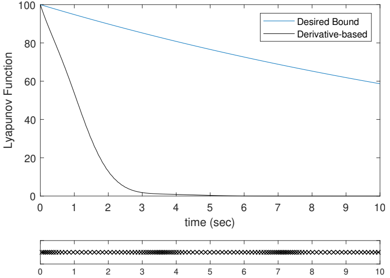
Design idea: **forget** σ, make $\dot{V} \leq -h(V)$, then $V \leq S$ (Comparison Lemma)

Derivative-based ET

$$t_{k+1} = \min_t \left\{ t > t_k \mid g(x(t), e(t)) + h(\bar{V}(x(t))) = 0 \right\}$$

where $\mathcal{L}_f V(x) \leq g(x, e) \leq -\alpha(\|x\|) + \gamma(\|e\|)$

Ex. update u when $-0.44\|x\|^2 + 8\|e\|\|x\| + 0.032V(x) = 0$



Derivative based Trigger

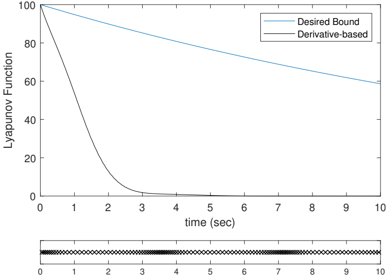
Design idea: **forget σ**, make $\dot{V} \leq -h(V)$, then $V \leq S$ (Comparison Lemma)

Derivative-based ET

$$t_{k+1} = \min_t \left\{ t > t_k \mid g(x(t), e(t)) + h(\bar{V}(x(t))) = 0 \right\}$$

where $\mathcal{L}_f V(x) \leq g(x, e) \leq -\alpha(\|x\|) + \gamma(\|e\|)$

Ex. update u when $-0.44\|x\|^2 + 8\|e\|\|x\| + 0.032V(x) = 0$



Problem? The trigger is too early. There is room for improvements.

Lyapunov Function Trigger

Design idea: just make $V \leq S$

Function-based ET

$$t_{k+1} = \left\{ t > t_k \mid S(x_0, t) - V(x(t)) = 0 \right\}$$

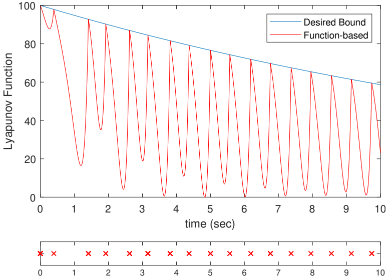
Lyapunov Function Trigger

Design idea: just make $V \leq S$

Function-based ET

$$t_{k+1} = \left\{ t > t_k \mid S(x_0, t) - V(x(t)) = 0 \right\}$$

Straightforward. Performance immediately satisfied



Efficient, less triggers, but there is no **robustness** to time delay

Performance Barrier Design

Design idea: combine the two schemes, but how?

Performance Barrier Design

Design idea: combine the two schemes, but how?

- Answer: Use the concept from control barrier function

Performance Barrier Design

Design idea: combine the two schemes, but how?

- Answer: Use the concept from control barrier function

Performance Barrier ET

$$t_{k+1} = \min_t \left\{ t > t_k \mid \underbrace{g(x(t), e(t)) + h(\bar{V}(x(t)))}_{derivative-based} = \beta\big(S(x_0, t) - \bar{V}(x(t))\big) \right\}$$

where β is class-$\mathcal{K}_\infty$

Performance Barrier Design

Design idea: combine the two schemes, but how?

- Answer: Use the concept from control barrier function

Performance Barrier ET

$$t_{k+1} = \min_t \left\{ t > t_k \mid \underbrace{g(x(t), e(t)) + h(\bar{V}(x(t)))}_{derivative-based} = \beta\big(S(x_0, t) - \bar{V}(x(t))\big) \right\}$$

where β is class-$\mathcal{K}_\infty$

What have we done here?

- We allow $\dot{V} > -h(V)$ given some performance "residual", $S - V > 0$

Performance Barrier Design

Design idea: combine the two schemes, but how?

- Answer: Use the concept from control barrier function

Performance Barrier ET

$$t_{k+1} = \min_{t} \left\{ t > t_k \mid \underbrace{g(x(t), e(t)) + h(\bar{V}(x(t)))}_{derivative-based} = \beta\big(S(x_0, t) - \bar{V}(x(t))\big) \right\}$$

where β is class-$\mathcal{K}_\infty$

What have we done here?

- We allow $\dot{V} > -h(V)$ given some performance "residual", $S - V > 0$
- We satisfy $V < S$ because it must be the case that $\dot{V} < -h(V)$ when $V = S$

Performance Barrier Design

Design idea: combine the two schemes, but how?

- Answer: Use the concept from control barrier function

Performance Barrier ET

$$t_{k+1} = \min_t \left\{ t > t_k \mid \underbrace{g(x(t), e(t)) + h(\bar{V}(x(t)))}_{derivative-based} = \beta\big(S(x_0, t) - \bar{V}(x(t))\big) \right\}$$

where β is class-$\mathcal{K}_\infty$

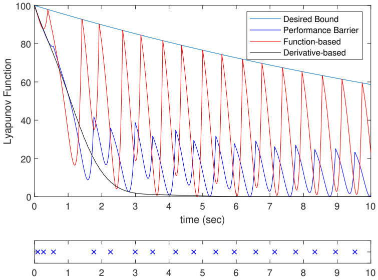
What have we done here?

- We allow $\dot{V} > -h(V)$ given some performance "residual", $S - V > 0$
- We satisfy $V < S$ because it must be the case that $\dot{V} < -h(V)$ when $V = S$
- It's like we set a **barrier** on V with S

Example

For the earlier example, update u when

$$\underbrace{-0.44\|x\|^2 + 8\|e\|\|x\|}_{g(x,e)} + \underbrace{0.032V(x)}_{h(V(x))} = \underbrace{10(V(x_0)\exp(-0.032t) - V(x))}_{\beta(S(x_0,t)-V(x))}$$



Maintain some level of **robustness** to time delay, not too inefficient in triggering

Advantages

Advantages of **performance barrier** design include:

- compared to **derivative-based**, guarantee higher minimum **interevent time**
  - because in each interval, derivative-based has to happen first
  - we provide the bound for the interevent time for linear case
  - we do not know by how much for general nonlinear case (future work)

Advantages

Advantages of **performance barrier** design include:

- compared to **derivative-based**, guarantee higher minimum **interevent time**
  - because in each interval, derivative-based has to happen first
  - we provide the bound for the interevent time for linear case
  - we do not know by how much for general nonlinear case (future work)
- compared to **function-based**, maintain some level of **robustness** to delays
  - have some time to update u after the trigger
  - depending on the function β, we can control how fast V is increasing
  - we do not know how much time delay we can tolerate exactly (future work)

Advantages

Advantages of **performance barrier** design include:

- compared to **derivative-based**, guarantee higher minimum **interevent time**
  - because in each interval, derivative-based has to happen first
  - we provide the bound for the interevent time for linear case
  - we do not know by how much for general nonlinear case (future work)
- compared to **function-based**, maintain some level of **robustness** to delays
  - have some time to update u after the trigger
  - depending on the function β, we can control how fast V is increasing
  - we do not know how much time delay we can tolerate exactly (future work)
- flexibility for distributed implementation
  - can extend performance barrier design to distributed scenarios
  - some interesting things can happen... (future work)

Distributed Systems

For the distributed system

$$\dot{x}_i = f_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i}^{(i)})$$

Under the following **assumptions**:

- V separable into $V = \sum_i V_i(x_{\mathcal{N}_i})$
- $\dot{V}$ separable into $\dot{V} = \sum_i \dot{V}_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i}^{(\mathcal{N}_i)})$
- S separable into $S = \sum_i S_i$

Distributed Systems

For the distributed system

$$\dot{x}_i = f_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i}^{(i)})$$

Under the following **assumptions**:

- V separable into $V = \sum_i V_i(x_{\mathcal{N}_i})$
- $\dot{V}$ separable into $\dot{V} = \sum_i \dot{V}_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i}^{(\mathcal{N}_i)})$
- S separable into $S = \sum_i S_i$
- $\dot{V}_i(x_{\mathcal{N}_i}, 0) \leq g_i(x_{\mathcal{N}_i}, 0) < -h(\bar{V}_i(x_{\mathcal{N}_i}))$

Distributed Systems

For the distributed system

$$\dot{x}_i = f_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i}^{(i)})$$

Under the following **assumptions**:

- V separable into $V = \sum_i V_i(x_{\mathcal{N}_i})$
- $\dot{V}$ separable into $\dot{V} = \sum_i \dot{V}_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i}^{(\mathcal{N}_i)})$
- S separable into $S = \sum_i S_i$
- $\dot{V}_i(x_{\mathcal{N}_i}, 0) \leq g_i(x_{\mathcal{N}_i}, 0) < -h(\bar{V}_i(x_{\mathcal{N}_i}))$
- Access to state information of two-hop neighbors

Distributed Systems

For the distributed system

$$\dot{x}_i = f_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i}^{(i)})$$

Under the following **assumptions**:

- V separable into $V = \sum_i V_i(x_{\mathcal{N}_i})$
- $\dot{V}$ separable into $\dot{V} = \sum_i \dot{V}_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i}^{(\mathcal{N}_i)})$
- S separable into $S = \sum_i S_i$
- $\dot{V}_i(x_{\mathcal{N}_i}, 0) \leq g_i(x_{\mathcal{N}_i}, 0) < -h(\bar{V}_i(x_{\mathcal{N}_i}))$
- Access to state information of two-hop neighbors
- Interevent time is lower bounded
  - Decentralized ET implementation can lead to Zeno (known in literature)
  - Trigger barrier can help! (in some situation)

Distributed Systems

For the distributed system

$$\dot{x}_i = f_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i}^{(i)})$$

Under the following **assumptions**:

- V separable into $V = \sum_i V_i(x_{\mathcal{N}_i})$
- $\dot{V}$ separable into $\dot{V} = \sum_i \dot{V}_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i}^{(\mathcal{N}_i)})$
- S separable into $S = \sum_i S_i$
- $\dot{V}_i(x_{\mathcal{N}_i}, 0) \leq g_i(x_{\mathcal{N}_i}, 0) < -h(\bar{V}_i(x_{\mathcal{N}_i}))$
- Access to state information of two-hop neighbors
- Interevent time is lower bounded
  - Decentralized ET implementation can lead to Zeno (known in literature)
  - Trigger barrier can help! (in some situation)

Distributed Performance Barrier Design

$$t_{i_{k+1}} = \min_t \left\{ t > t_{i_k} \mid g_i(x_{\mathcal{N}_i^2}, e_{\mathcal{N}_i^2}^{(\mathcal{N}_i)}) + h(\bar{V}_i(x_{\mathcal{N}_i})) = \beta(S_i(t) - \bar{V}_i(x_{\mathcal{N}_i})) \right\}$$

Simulations

Consider a full-state controlled system

$$\dot{x} = \begin{bmatrix} 2 & 2 & 1 & 0 & 0 \\ -3 & -3 & 0 & 2 & 0 \\ 1 & 0 & -2 & 3 & 1 \\ 0 & 1 & 3 & -4 & 5 \\ 0 & 0 & 1 & -2 & 1 \end{bmatrix} x + u, \;\; x, u \in \mathbb{R}^5$$

Simulations

Consider a full-state controlled system

$$\dot{x} = \begin{bmatrix} 2 & 2 & 1 & 0 & 0 \\ -3 & -3 & 0 & 2 & 0 \\ 1 & 0 & -2 & 3 & 1 \\ 0 & 1 & 3 & -4 & 5 \\ 0 & 0 & 1 & -2 & 1 \end{bmatrix} x + u, \;\; x, u \in \mathbb{R}^5$$

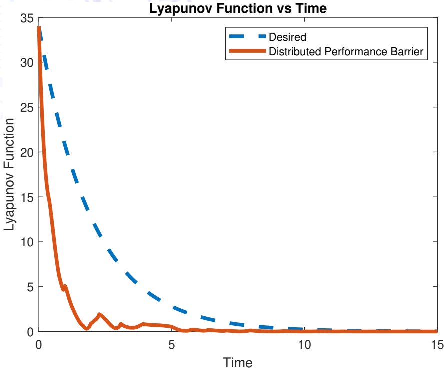
For continuous signal, cancel the off-diagonal

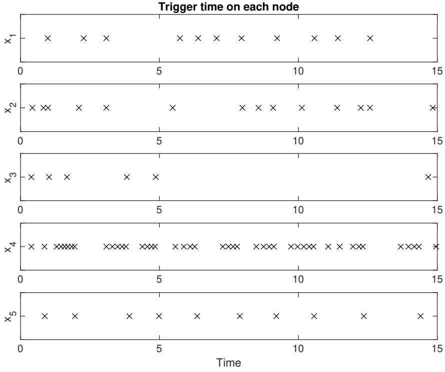
$$u = -\begin{bmatrix} 3 & 2 & 1 & 0 & 0 \\ -3 & 0 & 0 & 2 & 0 \\ 1 & 0 & 0 & 3 & 1 \\ 0 & 1 & 3 & 0 & 5 \\ 0 & 0 & 1 & -2 & 2 \end{bmatrix} x \implies \dot{x} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -3 & 0 & 0 & 0 \\ 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & 0 & -4 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix} x$$

This satisfy $S(x_0, t) = V(x_0)\exp(-0.5t)$ with some margin.
But, using **derivative-based** design $\rightarrow$ **Zeno!**

Simulations (cont.)

Using **performance barrier** design,





No Zeno Behavior!

Conclusions

New event-triggered design schemes - **Performance Barrier**

- Increase minimum interevent time by relaxing condition on $\dot{V}$
- Maintain some level of robustness
- Interesting application possibility in distributed scenarios

Future Work

- Characterize increase in interevent time and tradeoff with robustness
- Explore the benefits in distributed settings
  1. When can performance barrier fix Zeno behavior?
  2. Can distributed system communicate the residual and collaborate?

Question?

Questions and feedback are welcome!

Extra Slide

Possible future work: **Rebalancing residuals** between nodes