Relational Analysis of Sensor Attacks on Cyber-Physical Systems

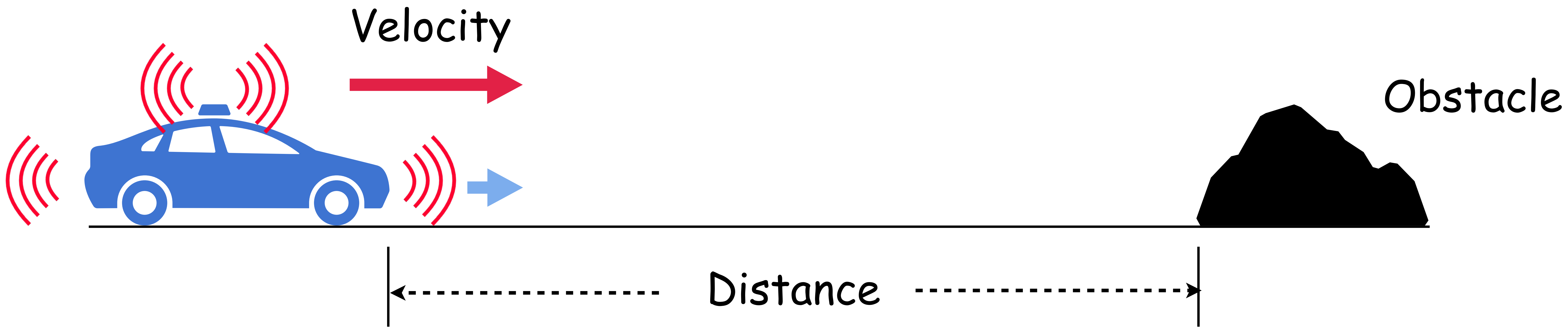
Jian Xiang*, Nathan Fulton†, Stephen Chong*

*Harvard University
jxiang, chong@seas.harvard.edu

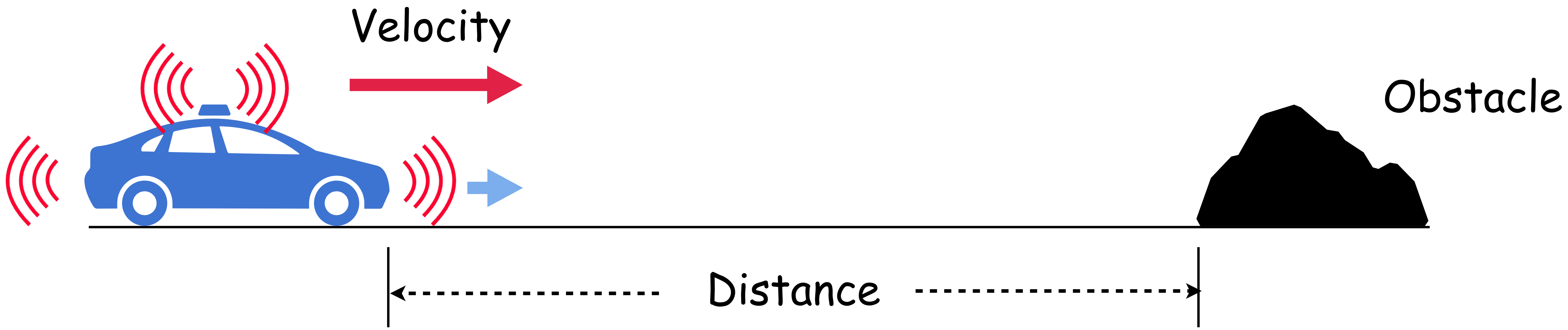
† MIT-IBM Watson AI Lab.
nathan@ibm.com

Sensor Attacks

Sensor Attacks

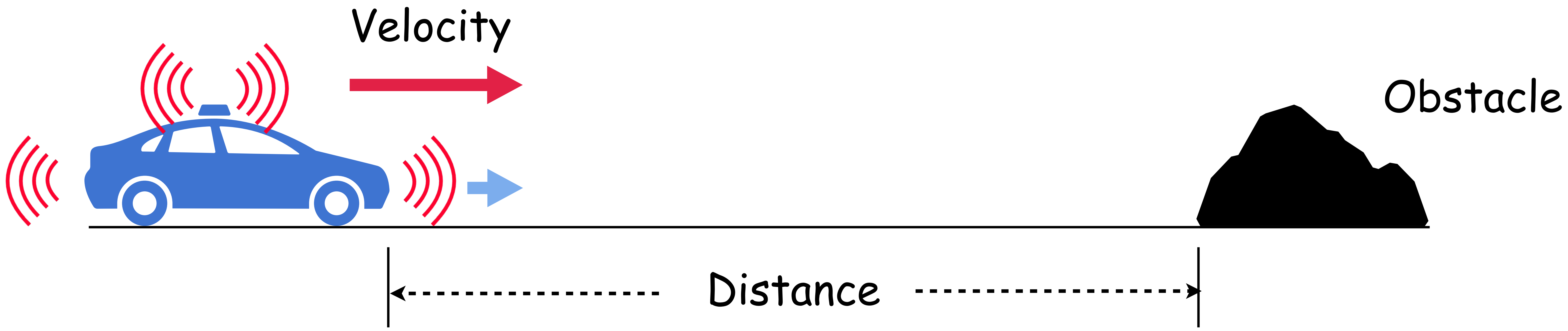


Sensor Attacks



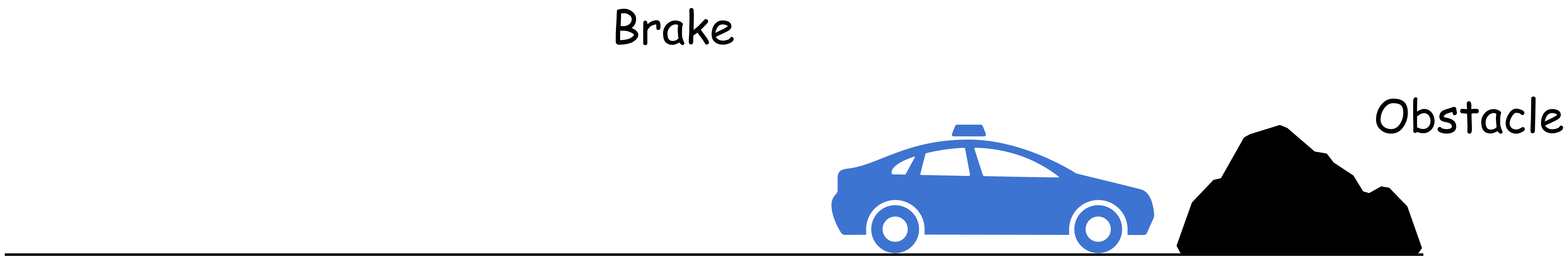
accelerate or brake?

Sensor Attacks



accelerate or brake? Decision is calculated based on sensed velocity and distance

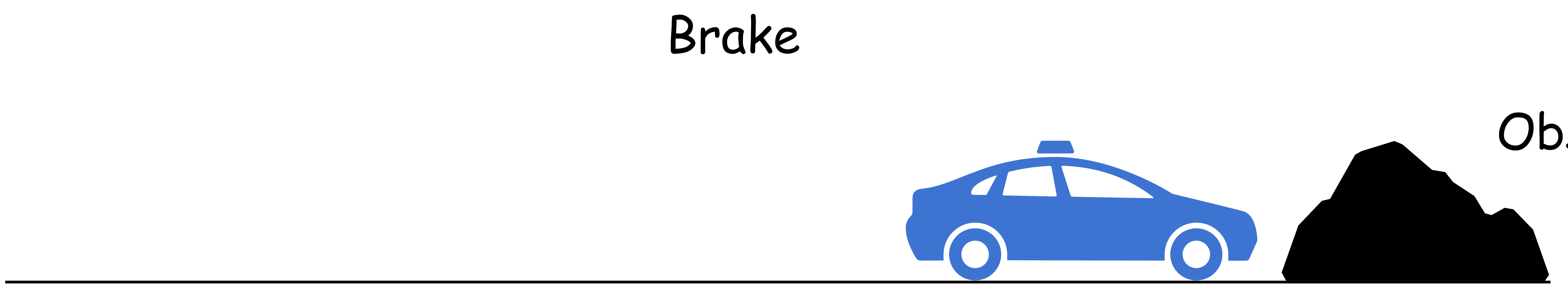
Sensor Attacks



Safe operation

accelerate or brake? Decision is calculated based on sensed velocity and distance

Sensor Attacks

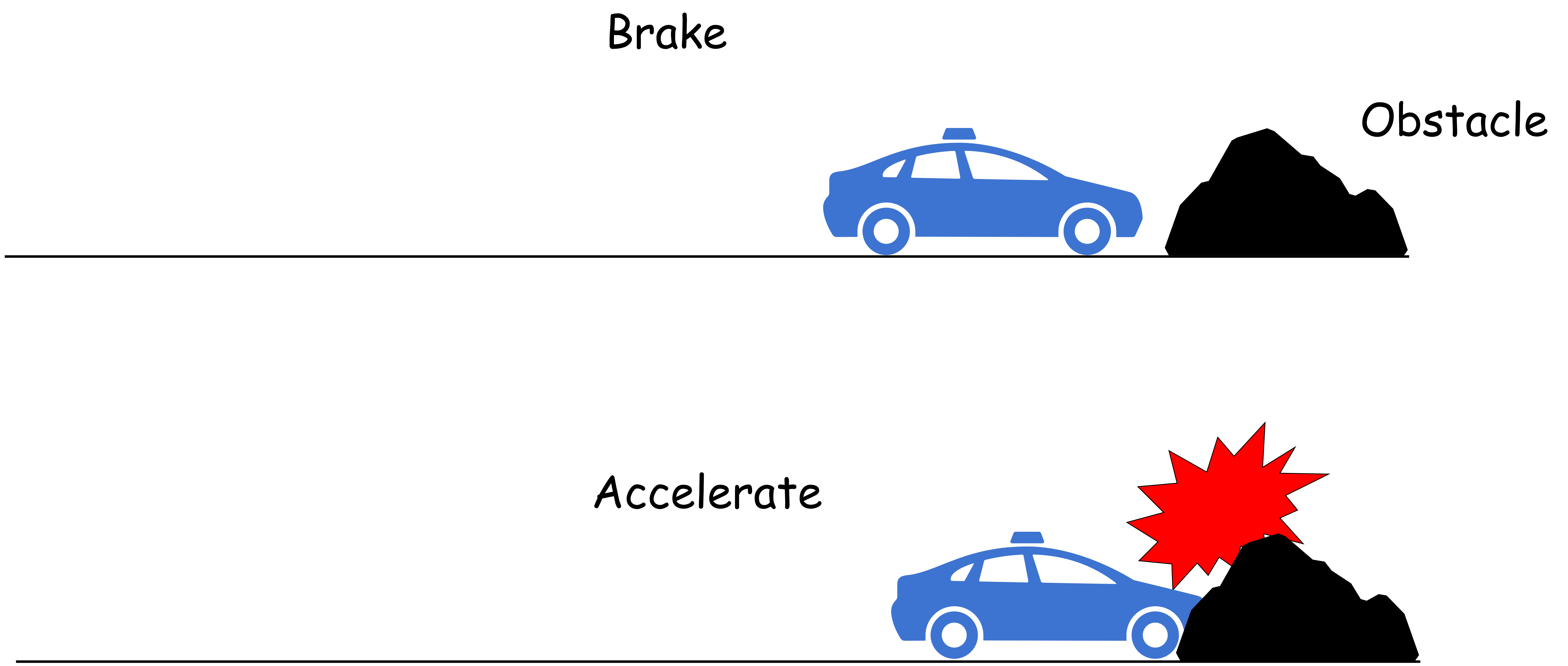


Safe operation

Sensor compromised

accelerate or brake? Decision is calculated based on sensed velocity and distance

Sensor Attacks



Safe operation

Sensor compromised

accelerate or brake? Decision is calculated based on sensed velocity and distance

Our View

1. Formal approach is needed
2. Understanding the impact of a compromised sensor is important
3. Need to reason about relational properties

Contributions

A formal framework to model and analyze sensor attacks on cyber-physical systems

Methodology

Contributions

A formal framework to model and analyze sensor attacks on cyber-physical systems

A threat model

Methodology

Contributions

A formal framework to model and analyze sensor attacks on cyber-physical systems

A threat model	→	Two robustness properties

Methodology

Contributions

A formal framework to model and analyze sensor attacks on cyber-physical systems

A threat model → Two robustness properties → An equivalence relation

Methodology

Contributions

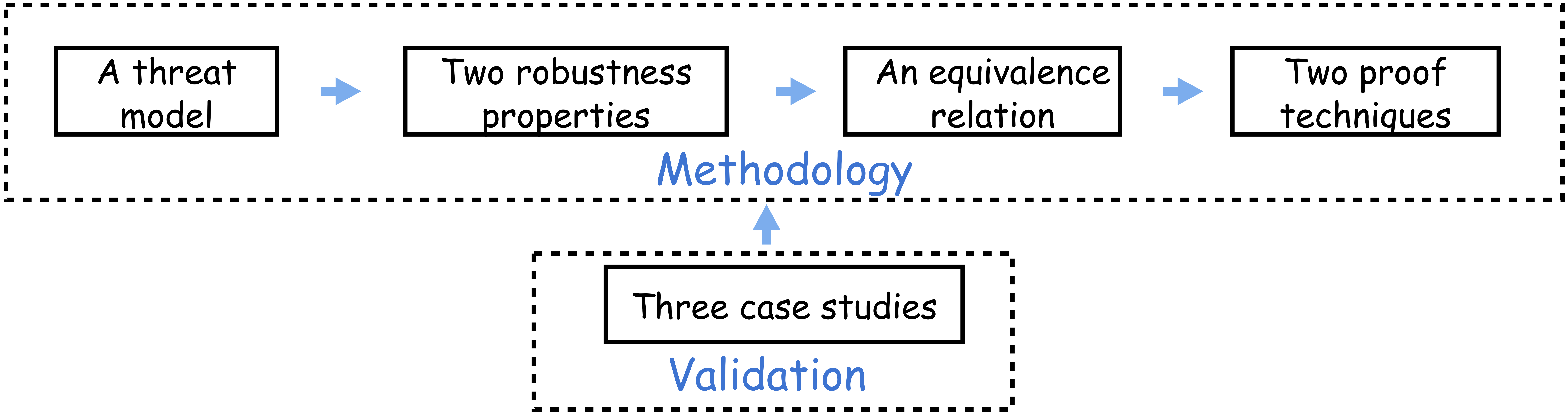
A formal framework to model and analyze sensor attacks on cyber-physical systems

A threat model → Two robustness properties → An equivalence relation → Two proof techniques

Methodology

Contributions

A formal framework to model and analyze sensor attacks on cyber-physical systems



Real-valued terms θ	
x	Program variable
c	Constant
$\theta_1 \oplus \theta_2$	Computation on terms

Hybrid Program α, β	
$x := \theta$	Deterministic assignment
$x := *$	Nondeterministic assignment
$x' = \theta \& \phi$	Continuous evolution
$?\phi$	Test if formula ϕ is true
$\alpha; \beta$	Sequential composition
$\alpha \cup \beta$	Nondeterministic choice
α^*	Nondeterministic repetition

Real-valued terms θ

x	Program variable
c	Constant
$\theta_1 \oplus \theta_2$	Computation on terms

Hybrid Program α, β

$\boxed{x := \theta}$	Deterministic assignment
$x := *$	Nondeterministic assignment
$x' = \theta \& \phi$	Continuous evolution
$?\phi$	Test if formula ϕ is true
$\alpha; \beta$	Sequential composition
$\alpha \cup \beta$	Nondeterministic choice
α^*	Nondeterministic repetition

Real-valued terms θ	
x	Program variable
c	Constant
$\theta_1 \oplus \theta_2$	Computation on terms

Hybrid Program α, β	
$x := \theta$	Deterministic assignment
$\boxed{x := *}$	Nondeterministic assignment
$x' = \theta \& \phi$	Continuous evolution
$?\phi$	Test if formula ϕ is true
$\alpha; \beta$	Sequential composition
$\alpha \cup \beta$	Nondeterministic choice
α^*	Nondeterministic repetition

Real-valued terms θ

x	Program variable
c	Constant
$\theta_1 \oplus \theta_2$	Computation on terms

Hybrid Program α, β

$x := \theta$	Deterministic assignment
$x := *$	Nondeterministic assignment
$\boxed{x' = \theta \& \phi}$	Continuous evolution
$?\phi$	Test if formula ϕ is true
$\alpha; \beta$	Sequential composition
$\alpha \cup \beta$	Nondeterministic choice
α^*	Nondeterministic repetition

Real-valued terms θ

x	Program variable
c	Constant
$\theta_1 \oplus \theta_2$	Computation on terms

Hybrid Program α, β

$x := \theta$	Deterministic assignment
$x := *$	Nondeterministic assignment
$x' = \theta \& \phi$	Continuous evolution
$?\phi$	Test if formula ϕ is true
$\alpha; \beta$	Sequential composition
$\alpha \cup \beta$	Nondeterministic choice
α^*	Nondeterministic repetition

Real-valued terms θ	
x	Program variable
c	Constant
$\theta_1 \oplus \theta_2$	Computation on terms

Hybrid Program α, β	
$x := \theta$	Deterministic assignment
$x := *$	Nondeterministic assignment
$x' = \theta \& \phi$	Continuous evolution
$?\phi$	Test if formula ϕ is true
$\boxed{\alpha; \beta}$	Sequential composition
$\alpha \cup \beta$	Nondeterministic choice
α^*	Nondeterministic repetition

Real-valued terms θ	
x	Program variable
c	Constant
$\theta_1 \oplus \theta_2$	Computation on terms

Hybrid Program α, β	
$x := \theta$	Deterministic assignment
$x := *$	Nondeterministic assignment
$x' = \theta \& \phi$	Continuous evolution
$?\phi$	Test if formula ϕ is true
$\alpha; \beta$	Sequential composition
$\boxed{\alpha \cup \beta}$	Nondeterministic choice
α^*	Nondeterministic repetition

Real-valued terms θ	
x	Program variable
c	Constant
$\theta_1 \oplus \theta_2$	Computation on terms

Hybrid Program α, β	
$x := \theta$	Deterministic assignment
$x := *$	Nondeterministic assignment
$x' = \theta \& \phi$	Continuous evolution
$?\phi$	Test if formula ϕ is true
$\alpha; \beta$	Sequential composition
$\alpha \cup \beta$	Nondeterministic choice
$\boxed{\alpha^*}$	Nondeterministic repetition

Real-valued terms θ

x	Program variable
c	Constant
$\theta_1 \oplus \theta_2$	Computation on terms

Hybrid Program α, β

$x := \theta$	Deterministic assignment
$x := *$	Nondeterministic assignment
$x' = \theta \& \phi$	Continuous evolution
$?\phi$	Test if formula ϕ is true
$\alpha; \beta$	Sequential composition
$\alpha \cup \beta$	Nondeterministic choice
α^*	Nondeterministic repetition

General form: (ctrl; plant)*

Differential Dynamic Logic ϕ, ψ

$\theta_1 \sim \theta_2$	Comparison between terms
$\neg\phi$	Negation
$\phi \wedge \psi$	Conjunction
$\phi \vee \psi$	Disjunction
$\phi \rightarrow \psi$	Implication
$\forall x.\ \phi$	Universal quantifier
$\exists x.\ \phi$	Existential quantifier
$[\alpha]\phi$	Program necessity
$\langle\alpha\rangle\phi$	Program existance

Differential Dynamic Logic ϕ, ψ

$\boxed{\theta_1 \sim \theta_2}$	Comparison between terms
$\neg\phi$	Negation
$\phi \wedge \psi$	Conjunction
$\phi \vee \psi$	Disjunction
$\phi \rightarrow \psi$	Implication
$\forall x.\ \phi$	Universal quantifier
$\exists x.\ \phi$	Existential quantifier
$[\alpha]\phi$	Program necessity
$\langle\alpha\rangle\phi$	Program existance

Differential Dynamic Logic ϕ, ψ

$\theta_1 \sim \theta_2$	Comparison between terms
$\neg\phi$	Negation
$\phi \wedge \psi$	Conjunction
$\phi \vee \psi$	Disjunction
$\phi \rightarrow \psi$	Implication
$\forall x.\ \phi$	Universal quantifier
$\exists x.\ \phi$	Existential quantifier
$\boxed{[\alpha]\phi}$	Program necessity
$\langle\alpha\rangle\phi$	Program existance

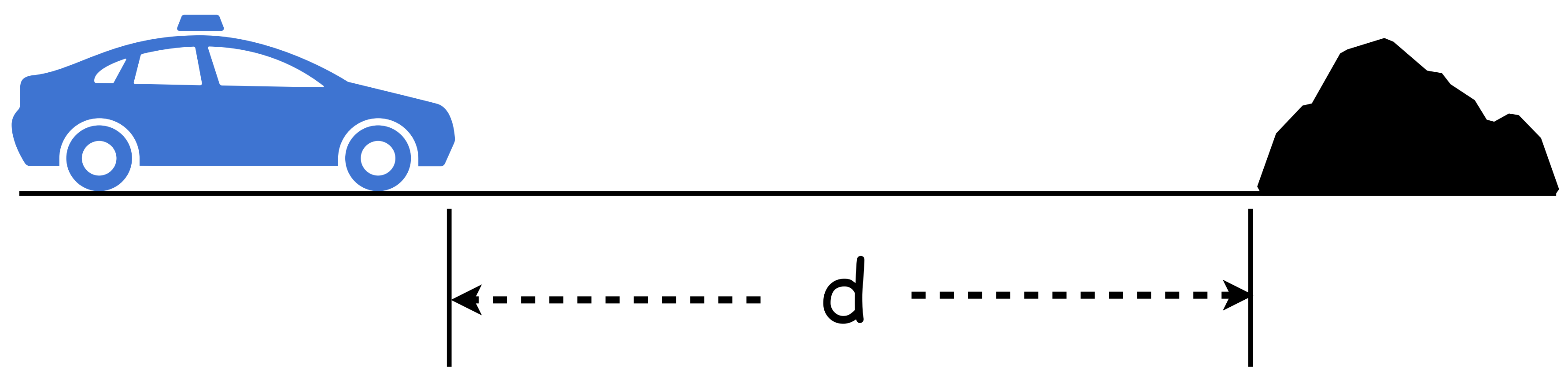
Differential Dynamic Logic ϕ, ψ

$\theta_1 \sim \theta_2$	Comparison between terms
$\neg\phi$	Negation
$\phi \wedge \psi$	Conjunction
$\phi \vee \psi$	Disjunction
$\phi \rightarrow \psi$	Implication
$\forall x.\ \phi$	Universal quantifier
$\exists x.\ \phi$	Existential quantifier
$\boxed{[\alpha]\phi}$	Program necessity
$\langle\alpha\rangle\phi$	Program existance

Safety: $\phi \rightarrow [\alpha]\psi$

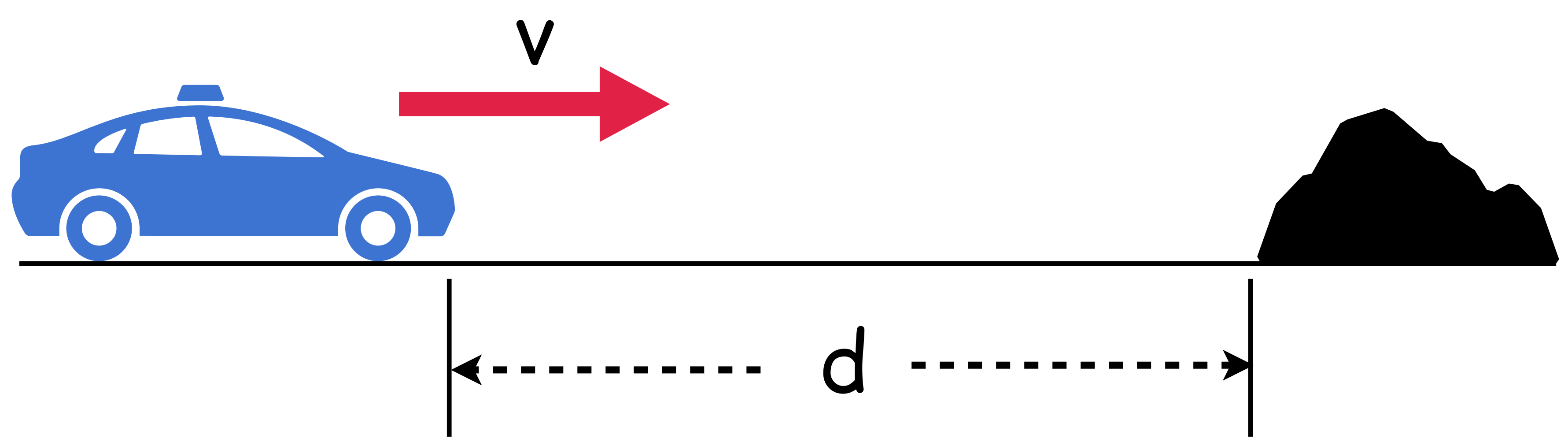
Example Hybrid Program Model

Example Hybrid Program Model



d : distance to obstacle

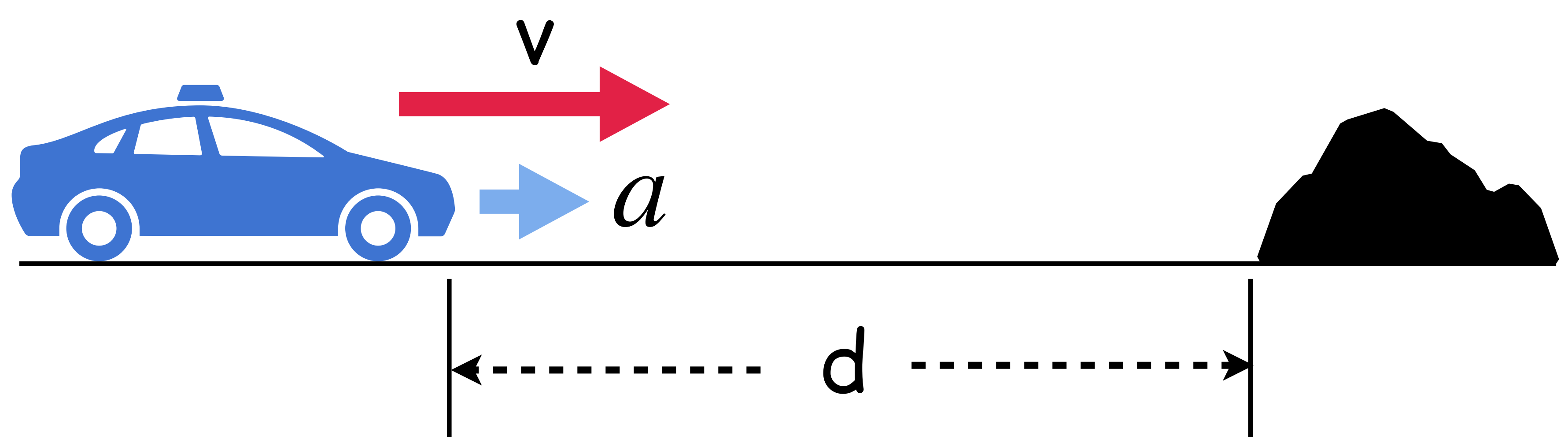
Example Hybrid Program Model



d : distance to obstacle

v : vehicle velocity

Example Hybrid Program Model

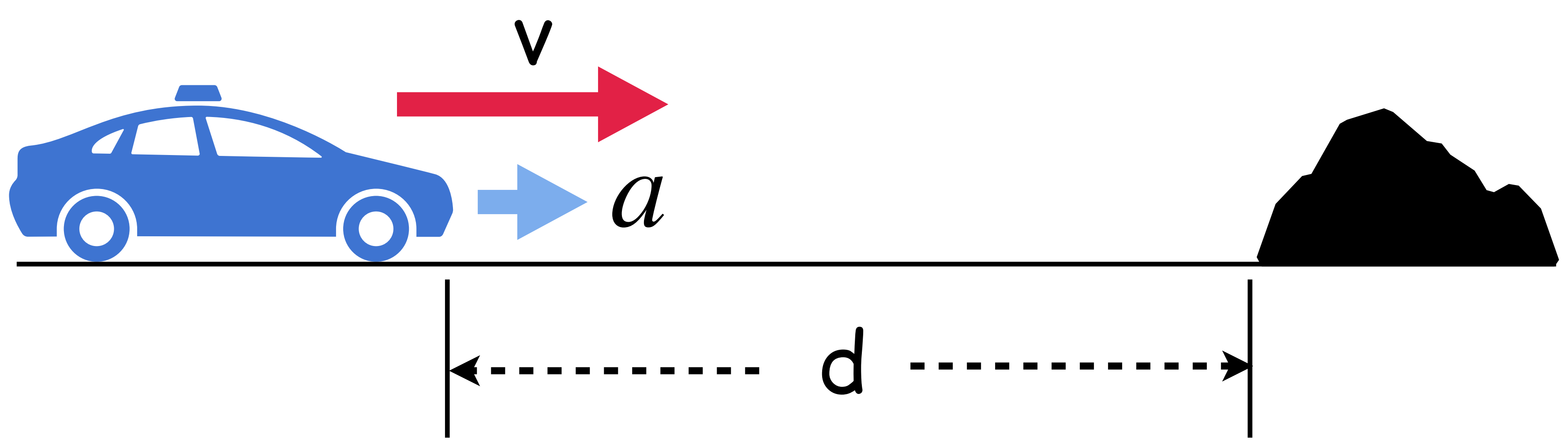


d : distance to obstacle

v : vehicle velocity

a : acceleration

Example Hybrid Program Model



d : distance to obstacle

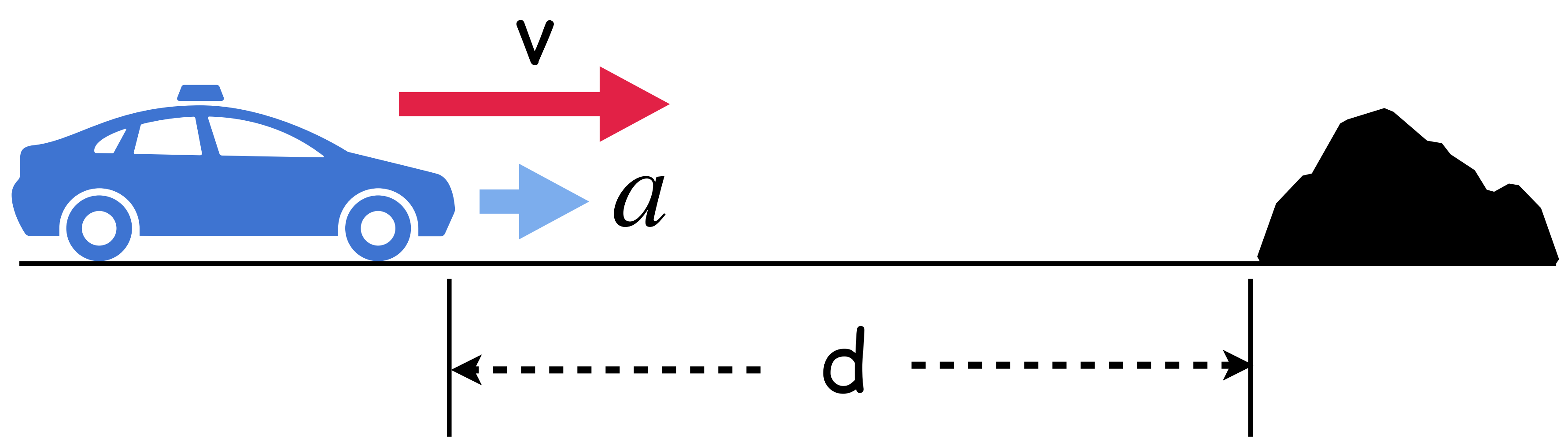
v : vehicle velocity

a : acceleration

t : clock variable

$$\text{ctrl} \equiv ((\text{accel} \cup \text{brake});\ t := 0)$$

Example Hybrid Program Model



d : distance to obstacle

v : vehicle velocity

a : acceleration

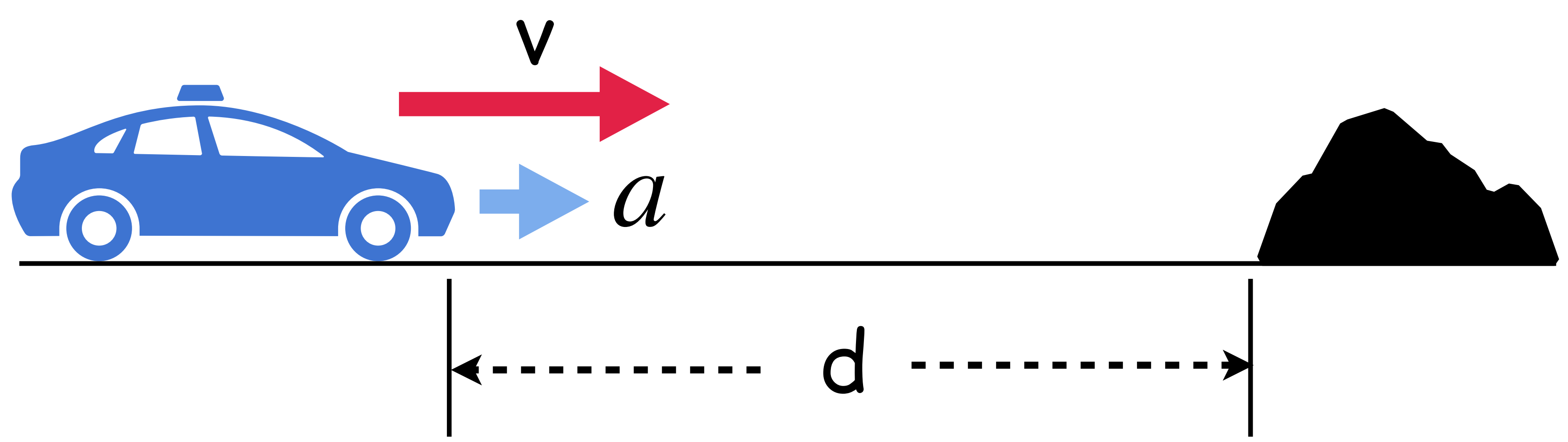
t : clock variable

B : braking rate

$$\text{ctrl} \equiv ((\text{accel} \cup \text{brake});\ t := 0)$$

$$\text{brake} \equiv a := -B$$

Example Hybrid Program Model



d : distance to obstacle

v : vehicle velocity

a : acceleration

t : clock variable

B : braking rate

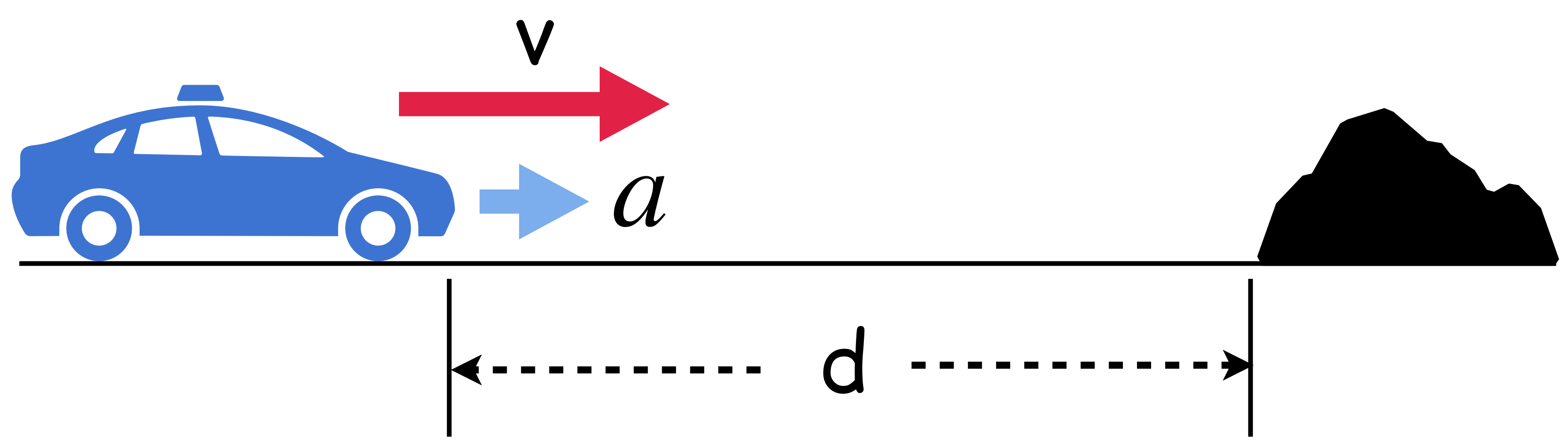
A : acceleration rate

$$\text{ctrl} \equiv ((\text{accel} \cup \text{brake});\ t := 0)$$

$$\text{brake} \equiv a := -B$$

$$\text{accel} \equiv ?\psi; a := A$$

Example Hybrid Program Model



d : distance to obstacle

v : vehicle velocity

a : acceleration

t : clock variable

B : braking rate

A : acceleration rate

ϵ : control interval

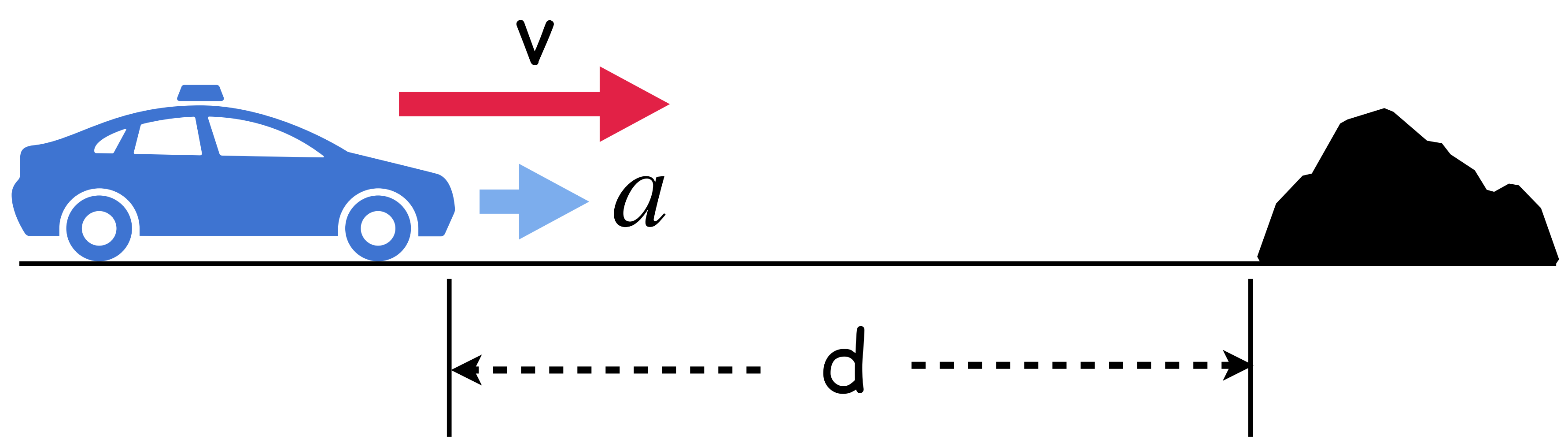
$$\text{ctrl} \equiv ((\text{accel} \cup \text{brake});\ t := 0)$$

$$\text{brake} \equiv a := -B$$

$$\text{accel} \equiv ?\psi; a := A$$

$$\psi \equiv 2Bd > v^2 + (A + B)(A\epsilon^2 + 2v\epsilon)$$

Example Hybrid Program Model



d : distance to obstacle
v : vehicle velocity
a : acceleration

t : clock variable
B : braking rate
A : acceleration rate
ϵ : control interval

$$\text{ctrl} \equiv ((\text{accel} \cup \text{brake});\ t := 0)$$

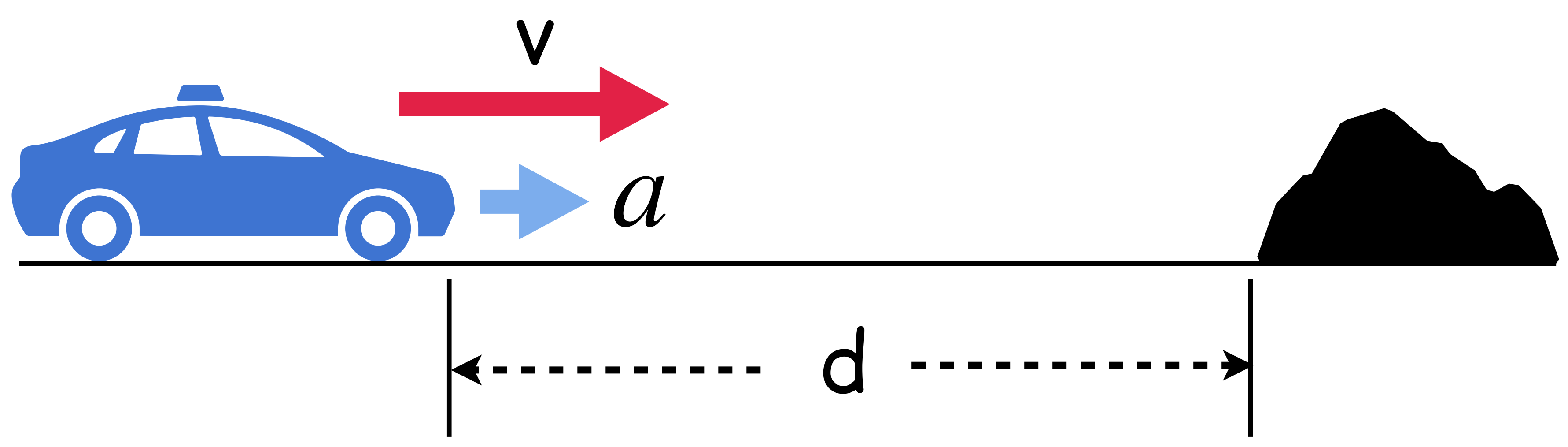
$$\text{brake} \equiv a := -B$$

$$\text{accel} \equiv ?\psi; a := A$$

$$\psi \equiv 2Bd > v^2 + (A+B)(A\epsilon^2 + 2v\epsilon)$$

$$\text{plant} \equiv d' = -v, v' = a, t' = 1 \ \&\ (v \geq 0 \wedge t \leq \epsilon)$$

Example Hybrid Program Model



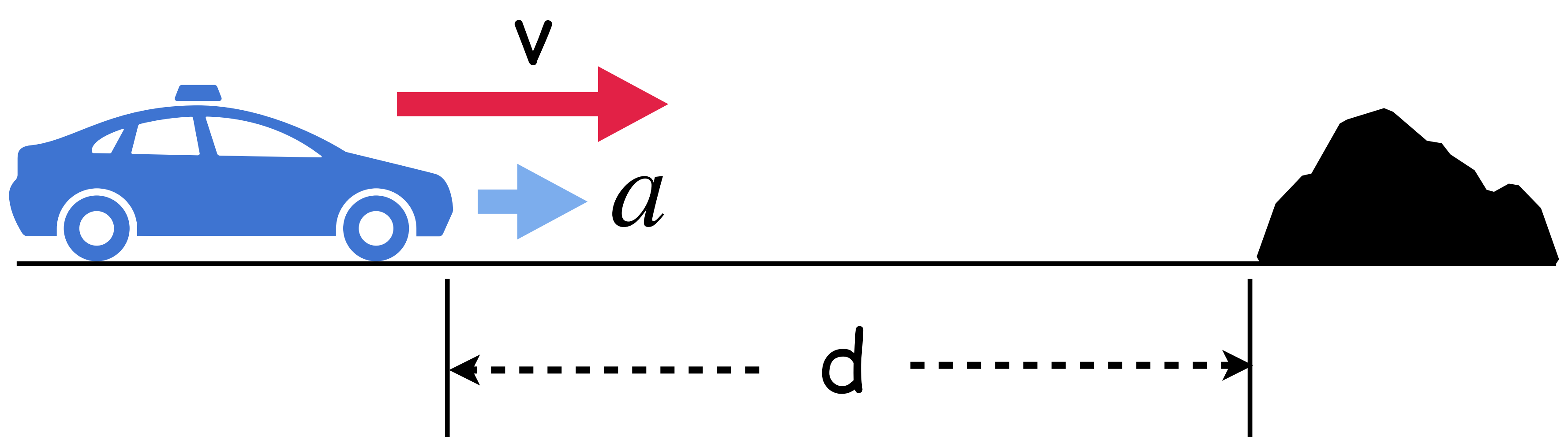
d : distance to obstacle
v : vehicle velocity
a : acceleration

t : clock variable
B : braking rate
A : acceleration rate
ϵ : control interval

$$\text{ctrl} \equiv ((\text{accel} \cup \text{brake});\ t := 0)$$
$$\text{brake} \equiv a := -B$$
$$\text{accel} \equiv ?\psi; a := A$$
$$\psi \equiv 2Bd > v^2 + (A+B)(A\epsilon^2 + 2v\epsilon)$$
$$\text{plant} \equiv d' = -v, v' = a, t' = 1 \ \&\ (v \geq 0 \wedge t \leq \epsilon)$$

$$(A \geq 0 \wedge B \geq 0 \wedge 2Bd > v^2)$$

Example Hybrid Program Model



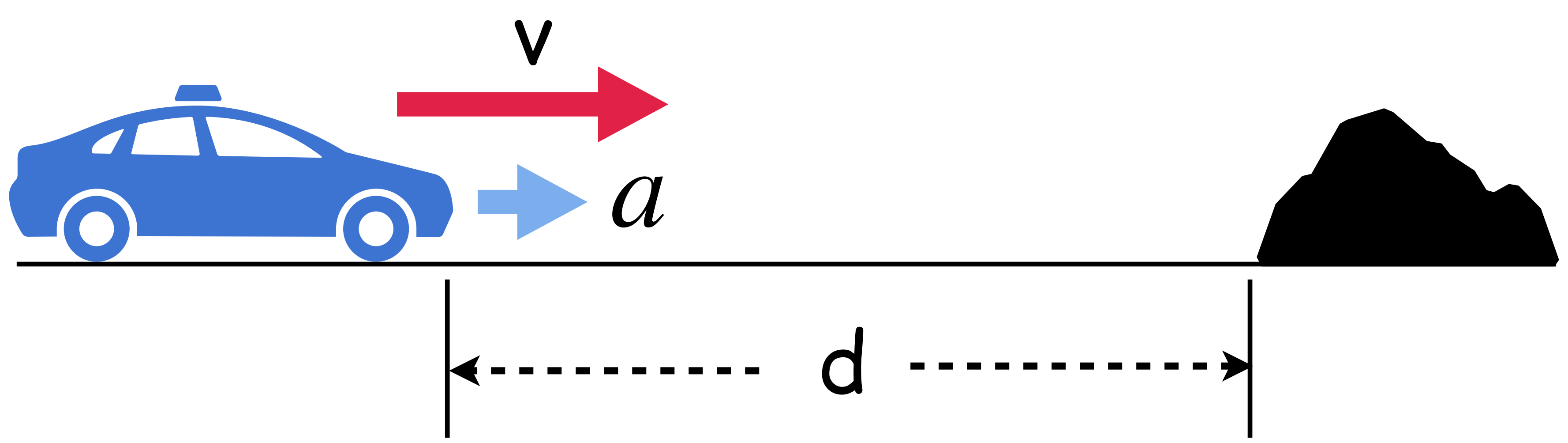
d : distance to obstacle
v : vehicle velocity
a : acceleration

t : clock variable
B : braking rate
A : acceleration rate
ϵ : control interval

$$\text{ctrl} \equiv ((\text{accel} \cup \text{brake});\ t := 0)$$
$$\text{brake} \equiv a := -B$$
$$\text{accel} \equiv ?\psi; a := A$$
$$\psi \equiv 2Bd > v^2 + (A + B)(A\epsilon^2 + 2v\epsilon)$$
$$\text{plant} \equiv d' = -v, v' = a, t' = 1 \ \&\ (v \geq 0 \wedge t \leq \epsilon)$$

$$(A \geq 0 \wedge B \geq 0 \wedge 2Bd > v^2) \rightarrow [(\text{ctrl}; \text{plant})^*]$$

Example Hybrid Program Model



d : distance to obstacle
v : vehicle velocity
a : acceleration

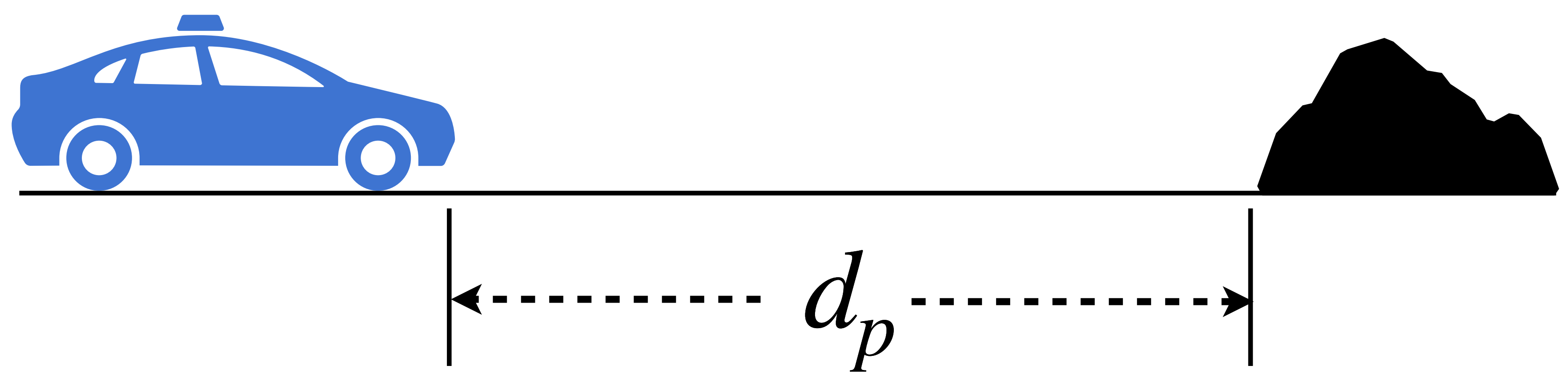
t : clock variable
B : braking rate
A : acceleration rate
ϵ : control interval

$$\text{ctrl} \equiv ((\text{accel} \cup \text{brake});\ t := 0)$$
$$\text{brake} \equiv a := -B$$
$$\text{accel} \equiv ?\psi; a := A$$
$$\psi \equiv 2Bd > v^2 + (A+B)(A\epsilon^2 + 2v\epsilon)$$
$$\text{plant} \equiv d' = -v, v' = a, t' = 1 \ \&\ (v \geq 0 \land t \leq \epsilon)$$

$$(A \geq 0 \land B \geq 0 \land 2Bd > v^2) \rightarrow [(\text{ctrl};\text{plant})^*]\ (d > 0)$$

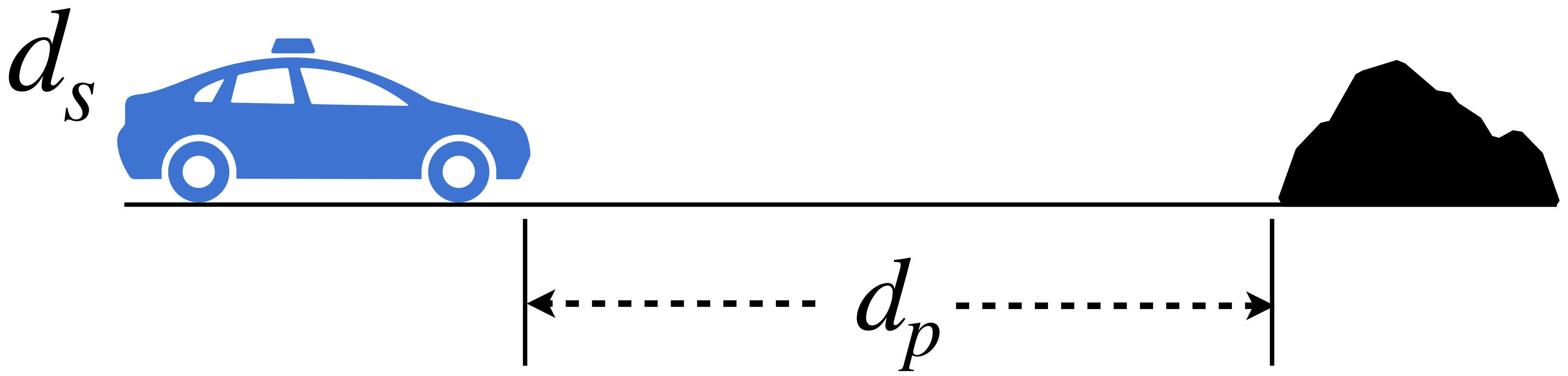
Example with Sensing

Example with Sensing



d_p : distance (physical)

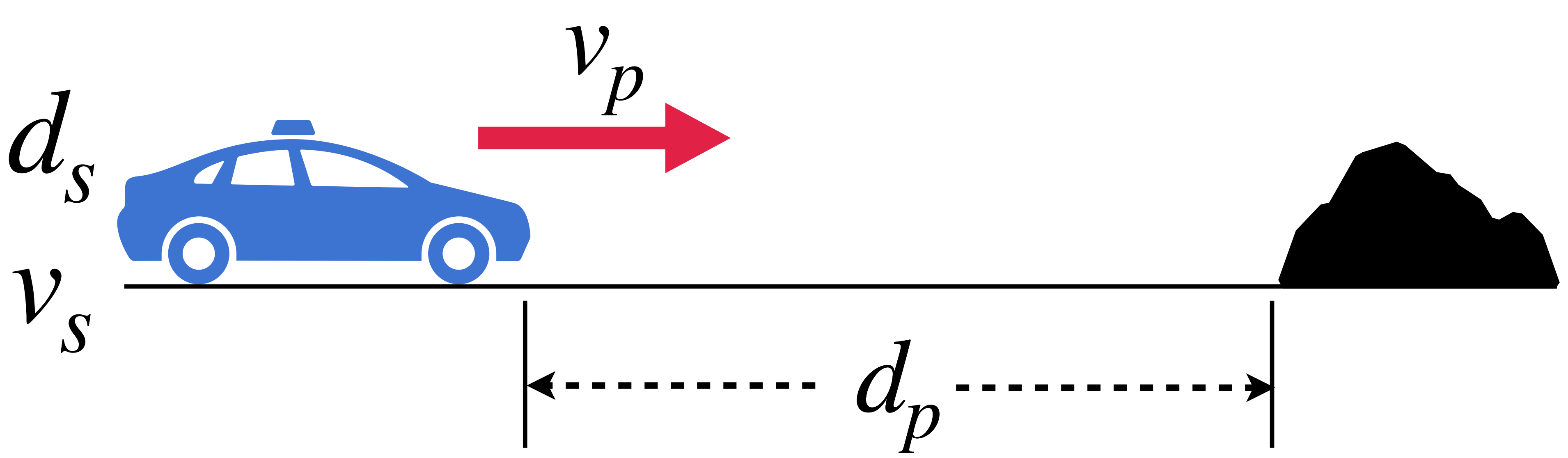
Example with Sensing



d_p : distance (physical)

d_s : distance (sensed)

Example with Sensing



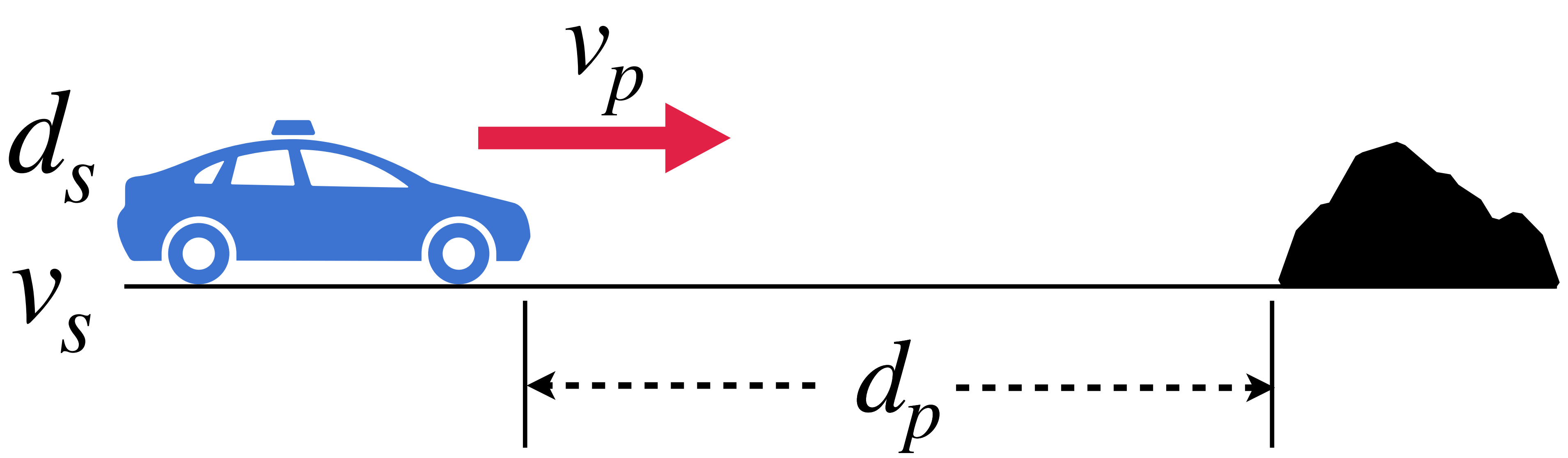
d_p : distance (physical)

d_s : distance (sensed)

v_p : velocity (physical)

v_s : velocity (sensed)

Example with Sensing



d_p : distance (physical)

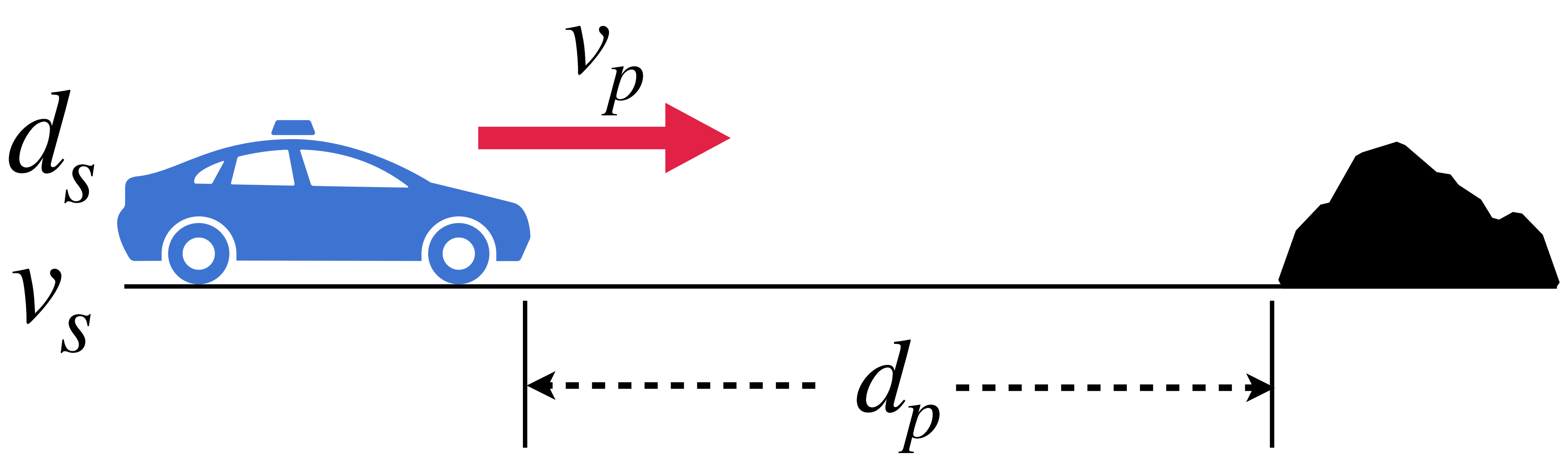
d_s : distance (sensed)

v_p : velocity (physical)

v_s : velocity (sensed)

$$\text{plant} \equiv d_p' = -v_p, v_p' = a, t' = 1 \,\&\, (v_p \geq 0 \wedge t \leq \epsilon)$$

Example with Sensing



d_p : distance (physical)
d_s : distance (sensed)

v_p : velocity (physical)
v_s : velocity (sensed)

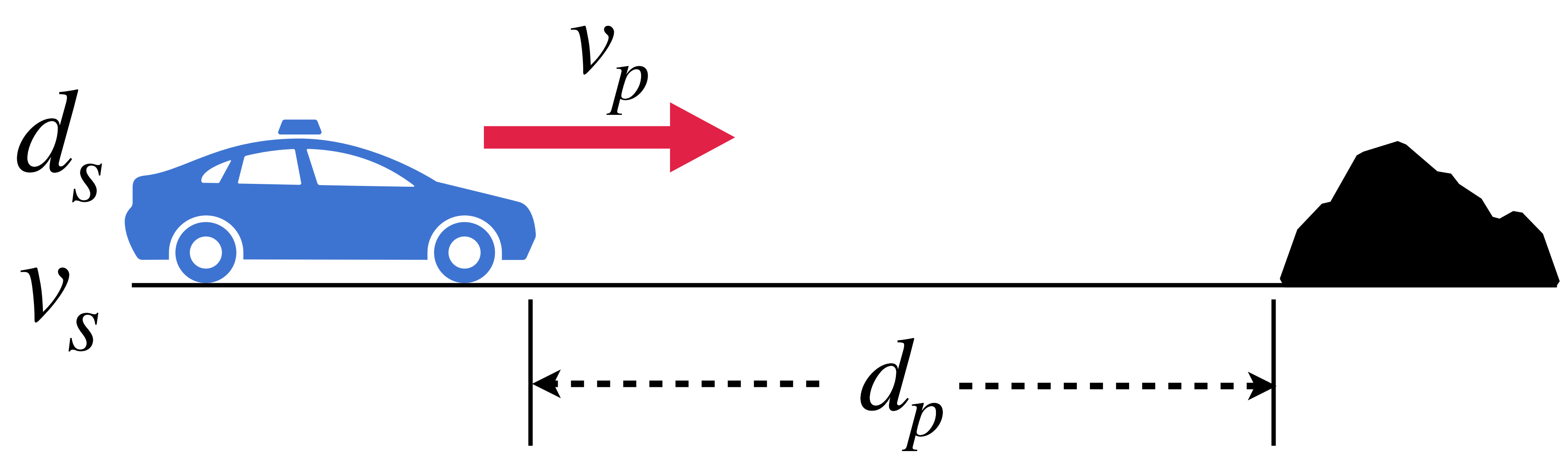
ctrl $\equiv$ ((accel $\cup$ brake); $t := 0$)

brake $\equiv$ $a := -B$

accel $\equiv$ $?\psi; a := A$

plant $\equiv$ $d_p' = -v_p, v_p' = a, t' = 1 \;\&\; (v_p \geq 0 \wedge t \leq \epsilon)$

Example with Sensing



d_p : distance (physical)

d_s : distance (sensed)

v_p : velocity (physical)

v_s : velocity (sensed)

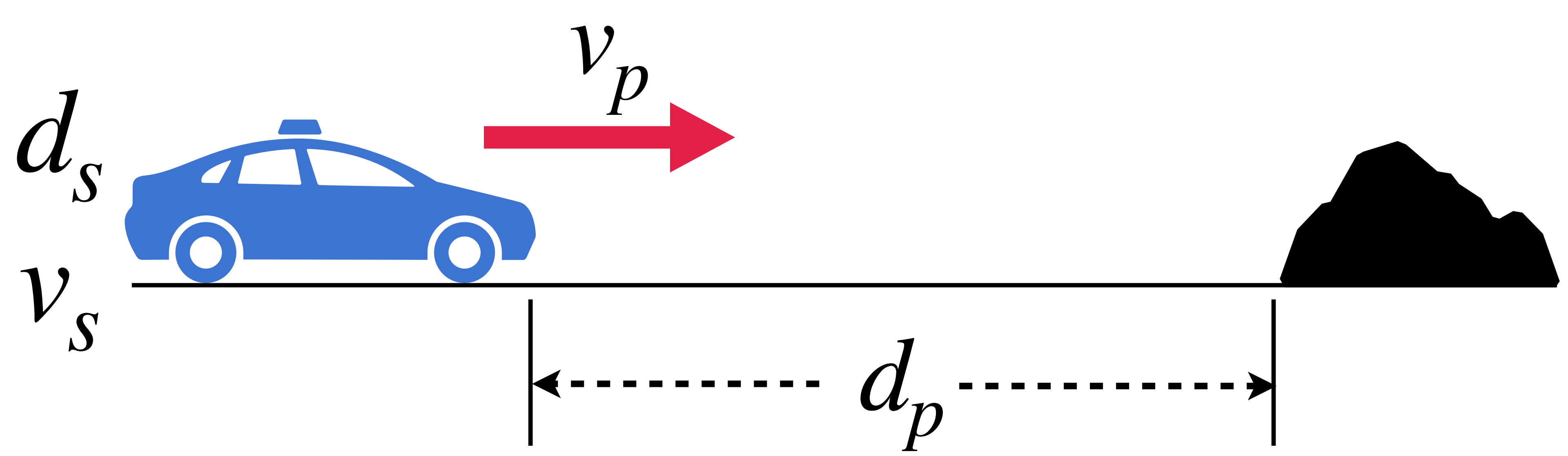
ctrl $\equiv v_s := v_p; d_s := d_p;$ ((accel ∪ brake); $\ t := 0)$

brake $\equiv \ a := -B$

accel $\equiv ?\psi; a := A$

plant $\equiv d_p' = -v_p, v_p' = a, t' = 1\ \&\ (v_p \geq 0 \wedge t \leq \epsilon)$

Example with Sensing



d_p : distance (physical)

d_s : distance (sensed)

v_p : velocity (physical)

v_s : velocity (sensed)

$$\text{ctrl} \equiv v_s := v_p; d_s := d_p; ((\text{accel} \cup \text{brake}); \; t := 0)$$

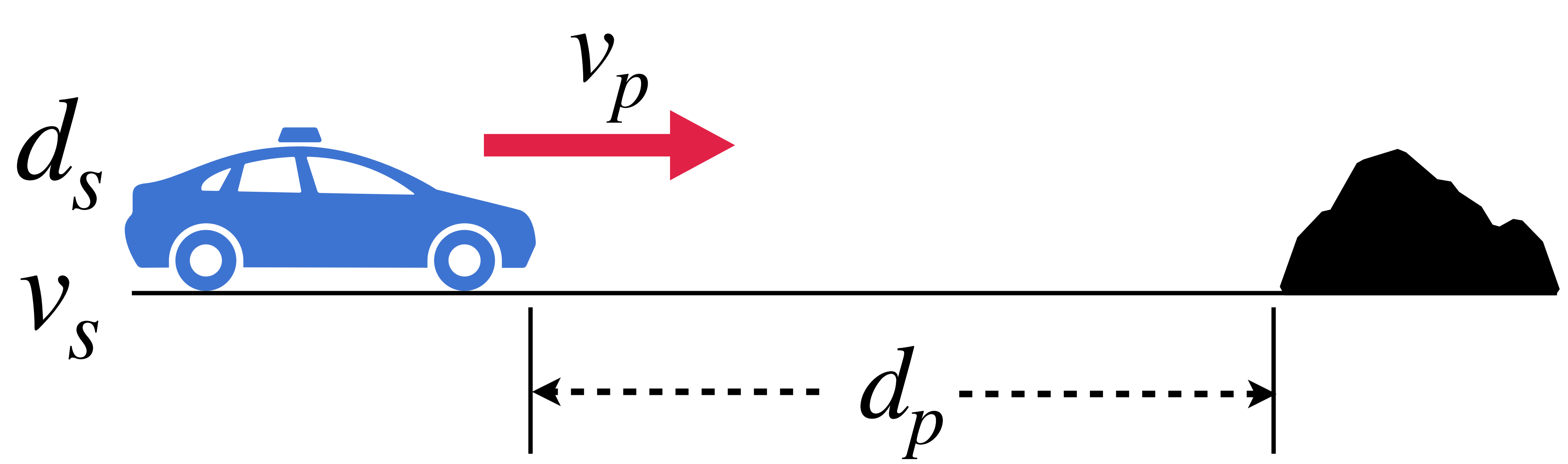
$$\text{brake} \equiv a := -B$$

$$\text{accel} \equiv ?\psi; a := A$$

$$\psi \equiv 2Bd_s > v_s^2 + (A+B)(A\epsilon^2 + 2v_s\epsilon)$$

$$\text{plant} \equiv d_p' = -v_p, v_p' = a, t' = 1 \;\&\; (v_p \geq 0 \land t \leq \epsilon)$$

Example with Sensing



d_p : distance (physical)
d_s : distance (sensed)
v_p : velocity (physical)
v_s : velocity (sensed)

$$\text{ctrl} \equiv v_s := v_p;\ d_s := d_p;\ ((\text{accel} \cup \text{brake});\ \ t := 0)$$

$$\text{brake} \equiv\ \ a := -B$$

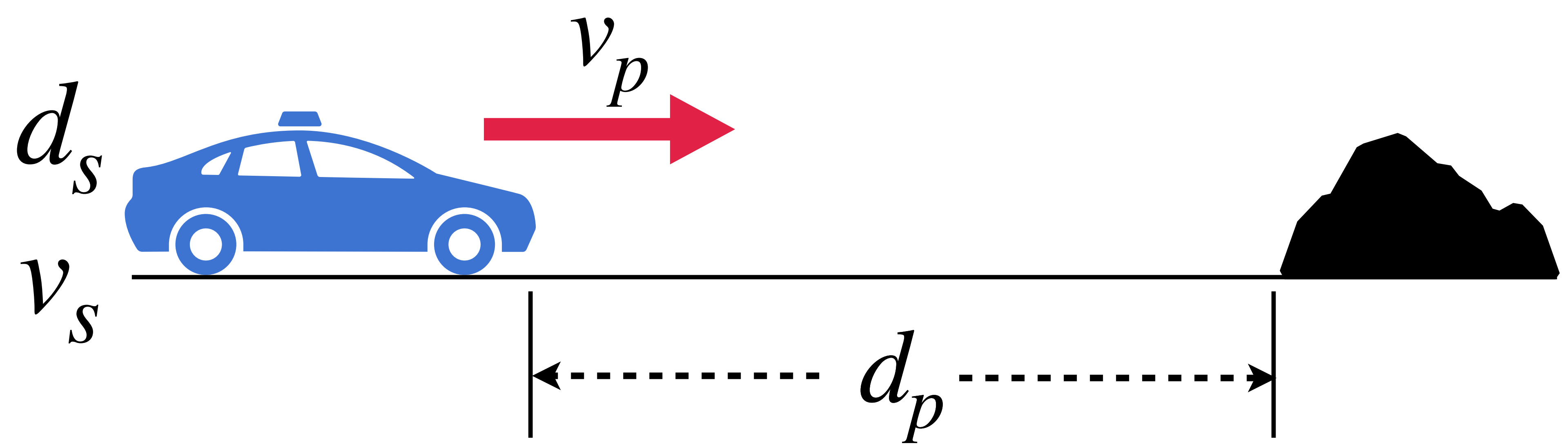
$$\text{accel} \equiv ?\psi;\ a := A$$

$$\psi \equiv 2Bd_s > v_s^2 + (A+B)(A\epsilon^2 + 2v_s\epsilon)$$

$$\text{plant} \equiv d_p' = -v_p,\ v_p' = a,\ t' = 1\ \&\ (v_p \geq 0 \land t \leq \epsilon)$$

$$(A \geq 0 \land B \geq 0 \land 2Bd_p > v_p^2) \quad \rightarrow [(\text{ctrl};\text{plant})^*]\ (d_p > 0)$$

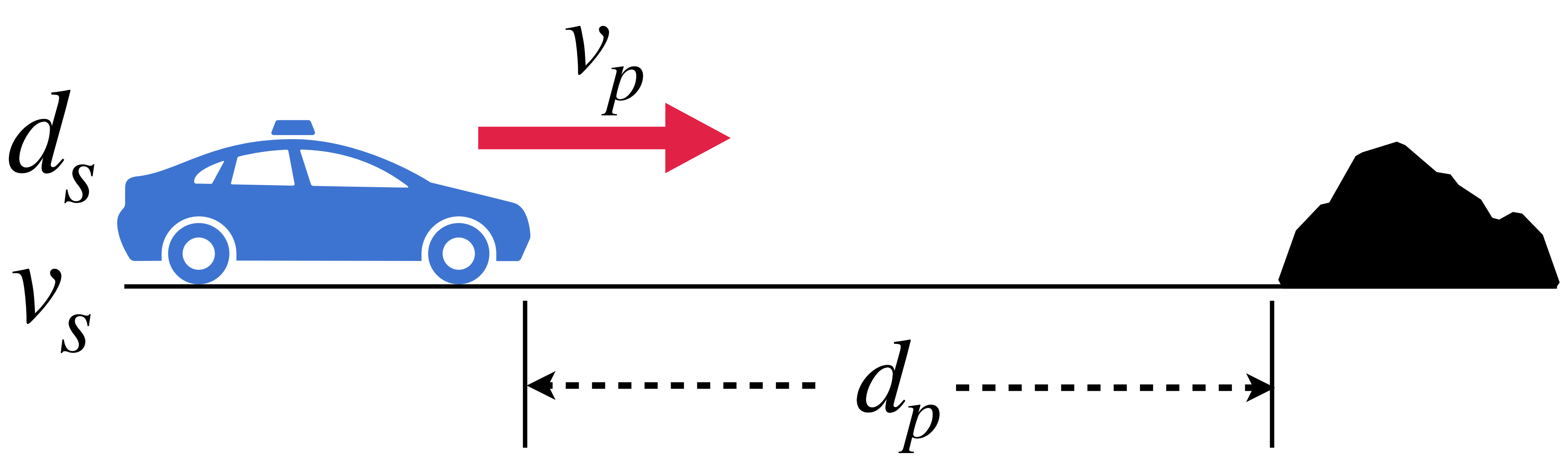
Example with Sensor Attack



d_p : distance (physical)
d_s : distance (sensed)
v_p : velocity (physical)
v_s : velocity (sensed)

$$\text{ctrl} \equiv v_s := v_p; d_s := d_p; ((\text{accel} \cup \text{brake}); \ t := 0)$$
$$\text{brake} \equiv a := -B$$
$$\text{accel} \equiv ?\psi; a := A$$
$$\psi \equiv 2Bd_s > v_s^2 + (A + B)(A\epsilon^2 + 2v_s\epsilon)$$
$$\text{plant} \equiv d_p' = -v_p, v_p' = a, t' = 1 \ \& \ (v_p \geq 0 \land t \leq \epsilon)$$

Example with Sensor Attack



d_p : distance (physical)

d_s : distance (sensed)

v_p : velocity (physical)

v_s : velocity (sensed)

$$\text{ctrl} \equiv d_s := d_p; ((\text{accel} \cup \text{brake}); \ t := 0)$$

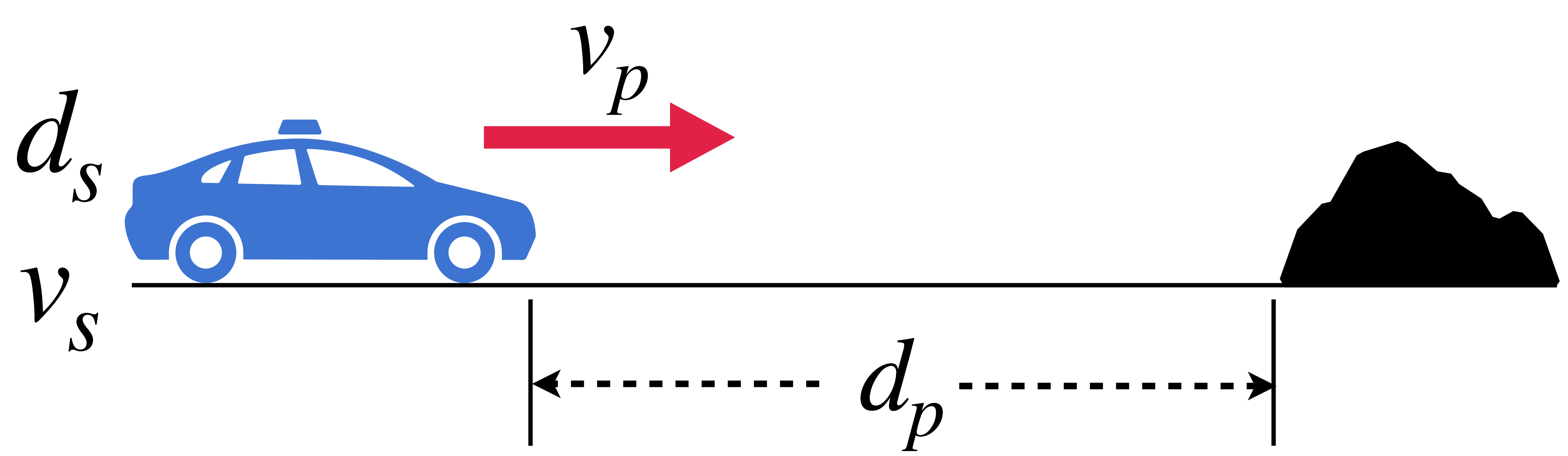
$$\text{brake} \equiv a := -B$$

$$\text{accel} \equiv ?\psi; a := A$$

$$\psi \equiv 2Bd_s > v_s^2 + (A+B)(A\epsilon^2 + 2v_s\epsilon)$$

$$\text{plant} \equiv d_p' = -v_p, v_p' = a, t' = 1 \ \& \ (v_p \geq 0 \land t \leq \epsilon)$$

Example with Sensor Attack



d_p : distance (physical)

d_s : distance (sensed)

v_p : velocity (physical)

v_s : velocity (sensed)

$$\text{ctrl} \equiv v_s := *;\ d_s := d_p; ((\text{accel} \cup \text{brake});\ t := 0)$$

$$\text{brake} \equiv a := -B$$

$$\text{accel} \equiv ?\psi; a := A$$

$$\psi \equiv 2Bd_s > v_s^2 + (A+B)(A\epsilon^2 + 2v_s\epsilon)$$

$$\text{plant} \equiv d_p' = -v_p, v_p' = a, t' = 1\ \&\ (v_p \geq 0 \wedge t \leq \epsilon)$$

Two Robustness Properties

P

$P_{Attacked}$

Two Robustness Properties

P

$P_{Attacked}$

Robustness of Safety

P is robustly safe: if P is safe, then $P_{Attacked}$ is safe

Two Robustness Properties

P

$P_{Attacked}$

Robustness of Safety $\neq$ **Safety**

P is robustly safe: if P is safe, then $P_{Attacked}$ is safe

Two Robustness Properties

P

$P_{Attacked}$

Robustness of Safety $\neq$ **Safety**

P is robustly safe: if P is safe, then $P_{Attacked}$ is safe

$$(\phi_{pre} \rightarrow [P]\phi_{safe})$$

Two Robustness Properties

P

$P_{Attacked}$

Robustness of Safety $\neq$ **Safety**

P is robustly safe: if P is safe, then $P_{Attacked}$ is safe

$$(\phi_{pre} \rightarrow [P]\phi_{safe}) \qquad \rightarrow \qquad (\phi_{pre} \rightarrow [P_{Attacked}]\phi_{safe})$$

Two Robustness Properties

P

$P_{Attacked}$

Robustness of Safety $\neq$ Safety

P is robustly safe: if P is safe, then $P_{Attacked}$ is safe

$$(\phi_{pre} \rightarrow [P]\phi_{safe}) \quad \rightarrow \quad (\phi_{pre} \rightarrow [P_{Attacked}]\phi_{safe})$$

Robustness of high-integrity state

H-Equivalence of Two Programs

H-Equivalence of Two Programs

$P_1 = \alpha *$ and $P_2 = \beta *$

H-Equivalence of Two Programs

$P_1 = \alpha *$ and $P_2 = \beta *$ $\quad P_1 \approx_H P_2$

H-Equivalence of Two Programs

$P_1 = \alpha^*$ and $P_2 = \beta^*$ $\quad P_1 \approx_H P_2$

$$\omega_0 \xrightarrow{\alpha} \omega_1 \xrightarrow{\alpha} \omega_2 \xrightarrow{\alpha} \omega_3 \cdots$$

H-Equivalence of Two Programs

$P_1 = \alpha *$ and $P_2 = \beta *$ $\quad P_1 \approx_H P_2$

$$\begin{tikzcd}
\omega_0 \arrow[d, "\alpha"'] & \nu_0 \arrow[d, bend left, "\beta"] \\
\omega_1 \arrow[d, "\alpha"'] & \nu_1 \arrow[d, bend left, "\beta"] \\
\omega_2 \arrow[d, "\alpha"'] & \nu_2 \arrow[d, bend left, "\beta"] \\
\omega_3 & \nu_3 \\
\vdots & \vdots
\end{tikzcd}$$

H-Equivalence of Two Programs

$P_1 = \alpha^*$ and $P_2 = \beta^*$ $\quad P_1 \approx_H P_2$

$$
\begin{tikzcd}
\omega_0 \arrow[d, "\alpha"'] & \approx_H & \nu_0 \arrow[d, bend left, "\beta"] \\
\omega_1 \arrow[d, "\alpha"'] & \approx_H & \nu_1 \arrow[d, bend left, "\beta"] \\
\omega_2 \arrow[d, "\alpha"'] & \approx_H & \nu_2 \arrow[d, bend left, "\beta"] \\
\omega_3 & \approx_H & \nu_3 \\
\vdots & & \vdots
\end{tikzcd}
$$

$\omega_i \approx_H \nu_i$ Two states agree on values of all variables in set H

Proving H-equivalence: Self-Composition

Proving: $P \approx_H P_{Attacked}$

Challenges:

Proving H-equivalence: Self-Composition

Proving: $P \approx_H P_{Attacked}$

Challenges:

1. Non-determinism,
   e.g., $accel \cup brake$

Proving H-equivalence: Self-Composition

Proving: $P \approx_H P_{Attacked}$

Challenges:

1. Non-determinism, e.g., $accel \cup brake$

2. Duration of continuous evolution, e.g., $d' = -v, v' = a \,\&\, (v \geq 0 \wedge t \leq \epsilon)$

Main Results & Future Work

1. A formal threat model
2. Two robustness properties
3. An equivalence relation for reasoning robustness
4. Two proof techniques
5. Three case studies

Future Work:

We are working on a more expressive relational logic

Thank you!