Detection is not enough: Attack Recovery for Safe and Robust Autonomous Robotic Vehicles

Pritam Dash, University of British Columbia
Perception in Robotic Vehicles (RV)

Turn Right in 50 meters

Roll rate = 0.5 rad/s
Sensor Attacks Against Robotic Vehicles (RV)

GPS Spoofing.
Transmit malicious GPS Signals

Tippenhauer et. al. On the requirements for successful GPS spoofing attacks. CCS’11
Sensor Attacks Against Robotic Vehicles (RV)

Signal Injection.
Optical, Magnetic or Acoustic noise

Son et. al. Rocking Drones with Intentional Sound Noise on Gyroscopic Sensors. Usenix Security’2015
Sensor Attacks and Consequences

Iran–U.S. RQ-170 incident

GPS Cyberattack Falsely Placed U.K. Warship Near Russian Naval Base
Risks for Industrial/Civilian Robotic Vehicles?
Efforts in Securing RVs
Attack Detection, Anomaly Detection

Invariant Based Detection

Model based Detection

“Very Effective in Detecting Attack”

Choi et. al., Detecting Attacks against Robotic Vehicles: a Control Invariant Approach, CCS’18
Quinonez et. al., SAVIOR: Securing Autonomous Vehicles with Robust Physical Invariants, Usenix Security’20
Detection is not Enough ...
Failsafe is not enough either...
How to safeguard RVs and ensure safety?
Attack Recovery not just Attack Detection
Feed-forward Control

Historic State Replay
Feed-forward Control

Historic State Replay
How sensor attacks disrupt RVs?
Sensor ➔ PID Control ➔ Actuator Signal

PID Control (Proportional Integral Derivative)
Sensor → PID Control → Actuator Signal
Sensor $\rightarrow$ PID Control $\rightarrow$ Actuator Signal
Sensor $\rightarrow$ PID Control $\rightarrow$ Actuator Signal
RV under Attack
PID Over-Compensates under Attacks
PID Over-Compensates under Attacks

![Diagram showing position error over time with attacks.](image-url)
PID Over-Compensates under Attacks
PID Over-Compensates under Attacks
PID Over-Compensates under Attacks
PID Over-Compensates under Attacks

PID compensation $\rightarrow$ handling faults ✓
under attacks $\times$
Approach to design Recovery Techniques

**Recovery Requirements**

R1: Handle persistent errors → erroneous physical states

R2: Prevent erroneous actuator Signals
Feedforward Controller (FFC) Design

Feedforward Control

Recovery Requirements

R1: Prevent erroneous physical states
R2: Prevent erroneous actuator signals
Recovery Framework

Feedforward Control

Feedback Control

FFC (ML) → command

command → FFC (ML)

e(t) → PID Control

T → e(t)

State Estimation → RV's state

RV's state → State Estimation

PID Control → e(t)

e(t) → PID Control
Recovery Framework

Feedforward Control

Feedback Control

FFC (ML) → PID Control → State Estimation → RV's state

command

$T$, $e(t)$, $R\hat{V}'s\ state$
Recovery Framework

Feedforward Control

Feedback Control

Feedforward Control

FFC (ML)

PID Control

State Estimation

RV’s state
Feed-Forward Control

• Prevents crashes – 0 crash.
• Ensure mission success despite attacks.
• ~1% energy overhead.
• ~7% performance overhead.
Feed-Forward Control

Single Sensor under attack ✓

Multiple Sensors under attack x
Multiple Sensor under Attack

Cao et. al., Invisible to both Camera and Lidar, IEEE S&P 2021

- Manipulate camera and LiDAR
- GPS and Gyroscope
Attack Setting

Limited to a Range

Cao et. al., Invisible to both Camera and Lidar, IEEE S&P 2021
Recovery Goal

Cao et. al., Invisible to both Camera and Lidar, IEEE S&P 2021

Prevent disruptions
Prevent Erroneous Actuator Signals

Recovery Requirements

R1: Prevent erroneous physical states
R2: Prevent erroneous actuator signals
Prevent Erroneous Actuator Signals

R1: Prevent erroneous physical states
R2: Prevent erroneous actuator signals
Identify the Sensor(s) under attack

Recovery Requirements

R1: Prevent erroneous physical states

R2: Prevent erroneous actuator signals
Isolate Sensor(s) from Control Process

Recovery Requirements

R1: Prevent erroneous physical states

R2: Prevent erroneous actuator signals
Substitute Input Sequence

Recovery Requirements

R1: Prevent erroneous physical states
R2: Prevent erroneous actuator signals
Substitute Input Sequence: Historic States

Record Historical States

- Position,
- Velocity,
- Angular rates...

Throttle
Substitute Input Sequence: Historic States

Record Historical States

Replay Historical States
Substitute Input Sequence: Historic States

Record Historical States

Replay Historical States
Isn’t replaying a type of attack?

How does this work?
Recovery with Historic State Input

\[ y = P \text{ (desired – actual)} \]

Where:

\[ x(t_0 - n) \]

\[ x(t_0 - 1) \]

\[ x(t_0 + 1) \]

\[ x(t_n) \]

\[ x(t_0) - x(t_0 + 1) \]

\[ x(t_0 - 1) - x(t_0) \]

\[ x(t_0 - n) - x(t_0 - n + 1) \]

\[ x(t_0) - x(t_n) \]
Recovery with Historic State Input

\[ x(t_0 - n) \]

\[ x(t_0 - 1) \]

\[ x(t_0 + 1) \]

\[ x(t_n) \]

\[ t_0 \]

\[ t_n \]

\[ y(t_0 + 1) \approx 1000 \text{ rpm} \]

\[ y(t_n) \approx 1000 \text{ rpm} \]

\[ y = P \left( \text{desired} - \text{actual} \right) \]

\[ y = P \left( \left( \text{desired} - \text{actual} \right) - \left( \text{desired} - \text{actual} \right) \right) \]

Replay Historical States
Without Recovery
Without Recovery

Attack Detected
With Recovery
With Recovery

Attack Detected

Software Recovery
Detection is not Enough

Attack Resilience is the step towards safe and robust autonomous RVs

Software solutions can offer low-cost attack recovery.

Contact Pritam Dash, pdash@ece.ubc.ca