Software-based Realtime Recovery from Sensor Attacks on Robotic Vehicles

Choi, Hongjun, et al. 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)

Presenter: SangminWoo@Syssec

RAID 2020
PROCEEDINGS
A USENIX Publication

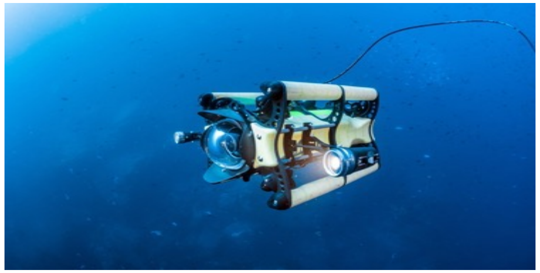
SYSSEC

Introduction

- RVs are becoming an integral part of our daily life.





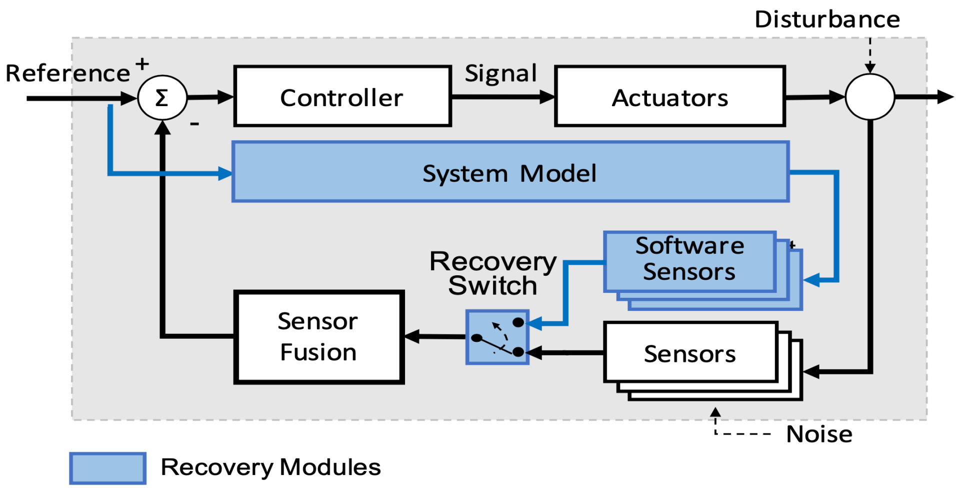


Safety Critical Systems

Malfunction -> Physical damages

Introduction

- ❖ Previous works only focused on detecting malfunctions.
- ❖ Proposed a new technique to recover from the malfunctions
  - – Software sensor: Software backup of physical sensors



Background: Multi-sensor RVs

- Heterogeneous Sensor and Sensor Fusion on UAV

Gyroscope

Accelerometer

Barometer

Magnetometer (Compass)

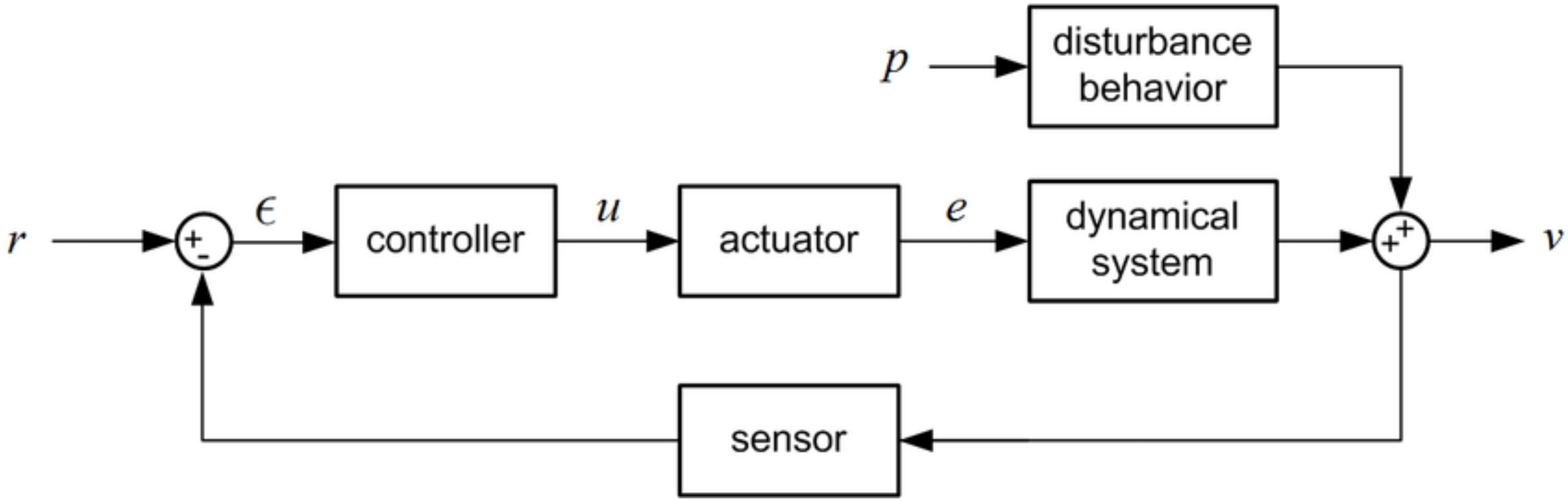
GPS

Distance Sensor (RangeFinder)

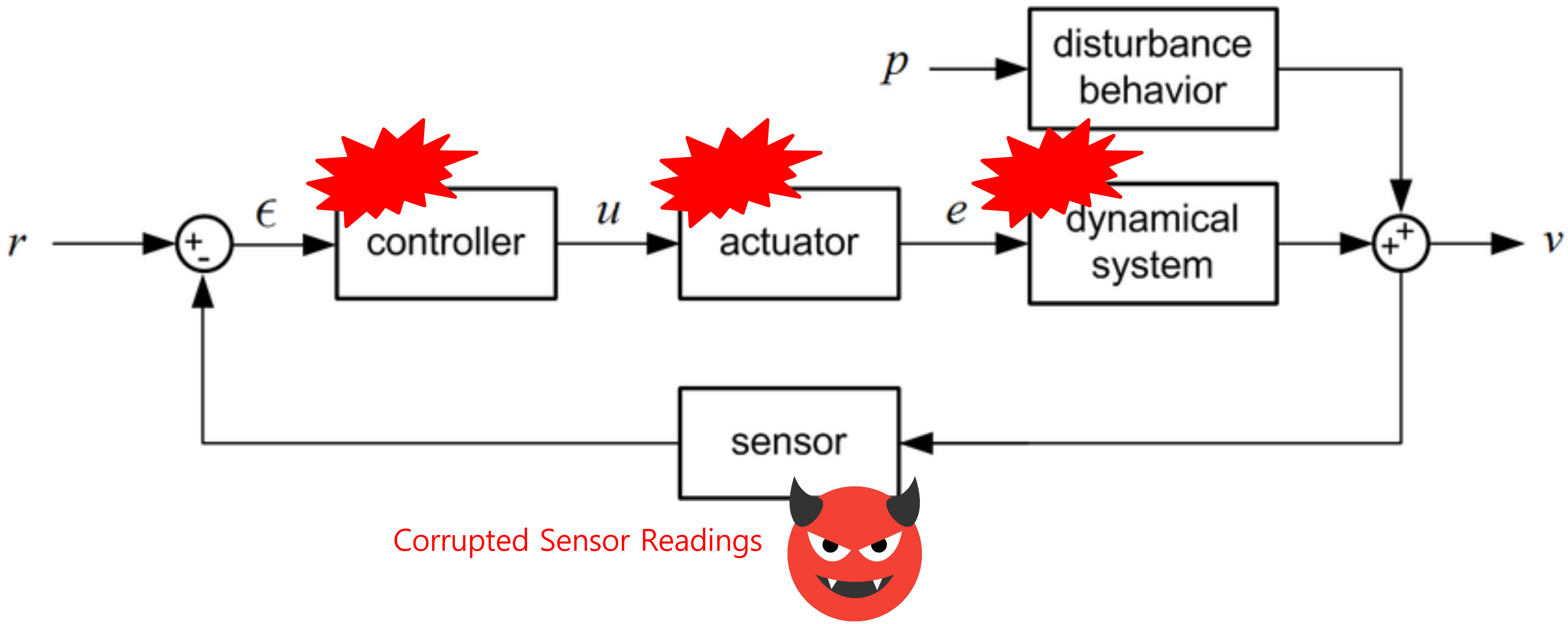
Sensor Fusion
(Converting and Weighted Sum)

Attitude / Position Control

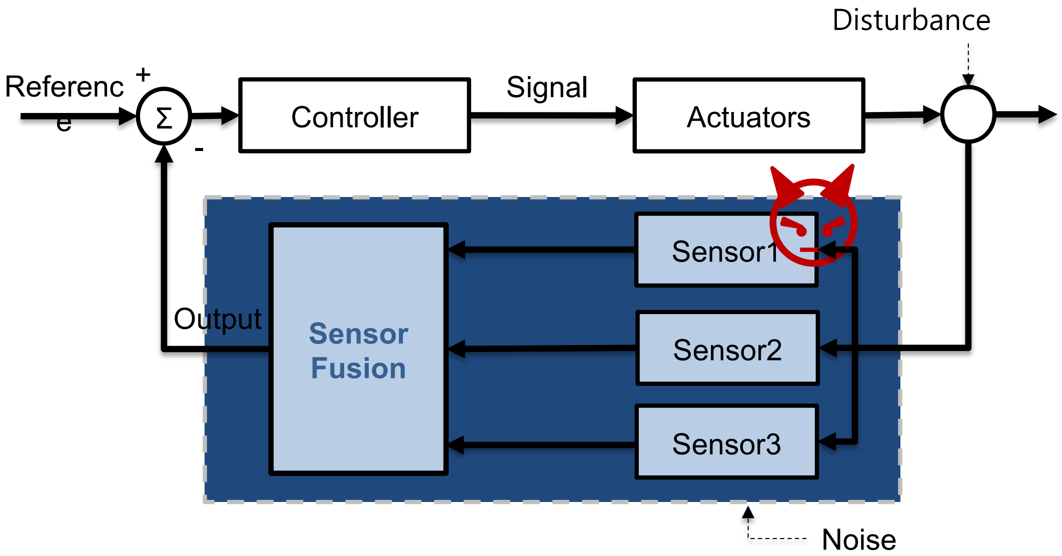
Background: Feedback Control Loop



Background: Sensor Attacks



Background: Existing Approaches



Sensor Fusion with Sensor Redundancy (TMR)

- Hardware Sensor Redundancy
  - Multiple HW sensors
  - Competitive (e.g., voting) or complementary way (e.g., weighted average)
- Heterogeneous Sensor Fusion
  - Use different types of sensors to measure states
  - Extended Kalman Filter
- Limitation
  - Attack resilient only for subset of sensors
  - Difficult to pinpoint the compromised sensor
  - cost

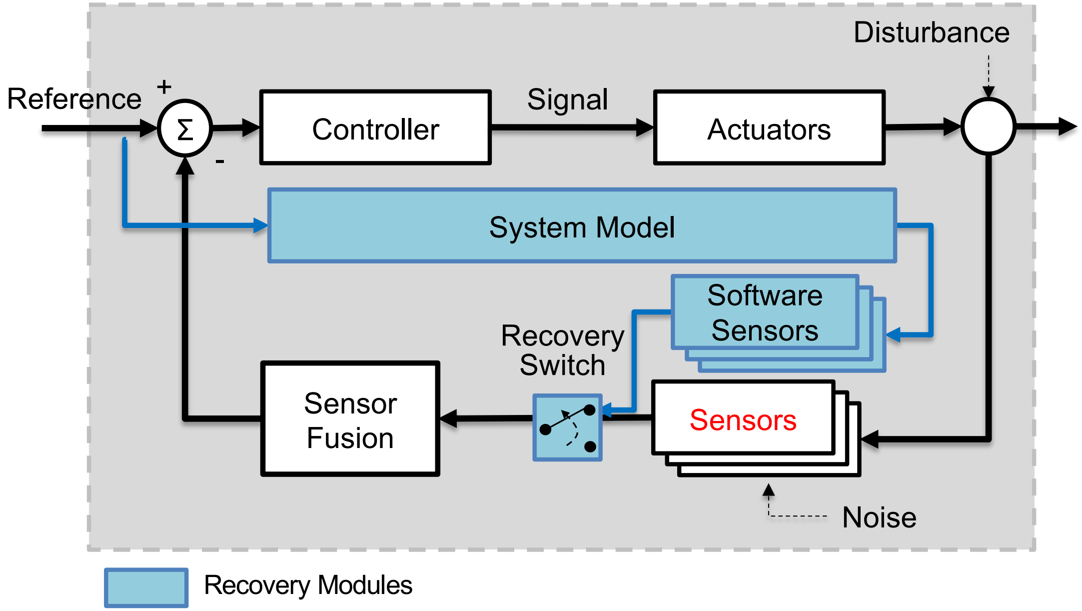
Contribution

- ❖ Propose a novel software-based technique: software sensors to recovery from sensor attacks
- ❖ Address prominent challenges:
  - How to generate software sensors using system identification?
  - How to recover from individual sensor failures?
  - How to improve software sensor accuracy considering external disturbances for practical usage?
- ❖ Comprehensive experiments on various RVs using attacks on one or multiple sensors

Software-sensor

```
main_loop() {

    // determines vehicle states
    angles = read_AHRS();

    // generates target values
    targets = navigation_logic();

    // generates actuation signal
    inputs = attitude_controller(targets, angles);

    // sends signals to actuators
    motor.update(inputs);
}
read_AHRS() {

    // read IMU sensor measurements
    for(i=0; i<num_gyro; i++) {
        gyros[i] = gyro_sensors[i].read(); // *attack*

        // *inserted code for attack recovery*
        if(abs(soft_gyro[i] - gyros[i]) > k)
            gyros[i] = soft_gyro[i];

        // weighted sum
        gyro += w[i] * gyros[i];
    }
    // return angles
    angles = convert2angle(gyro);
    return angles;
}
```

Control Program

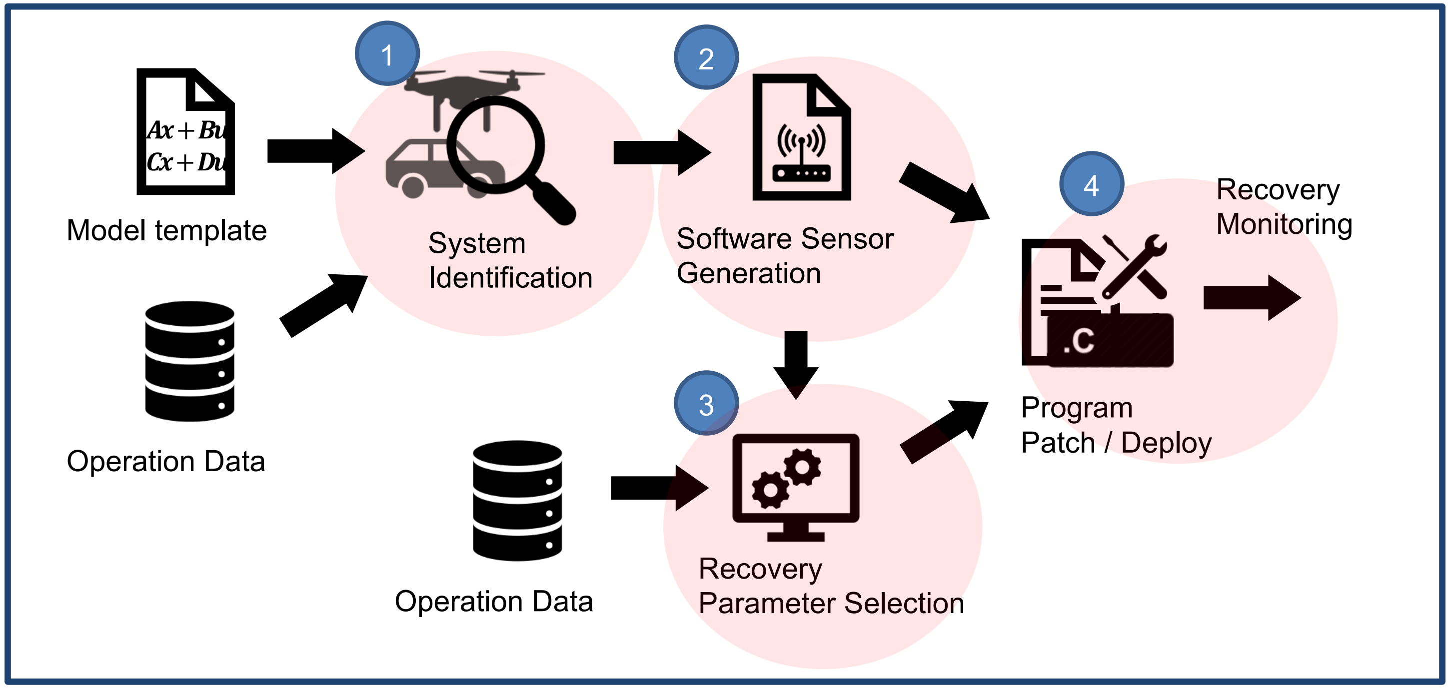


Feedback Control Loop

Technical Challenges

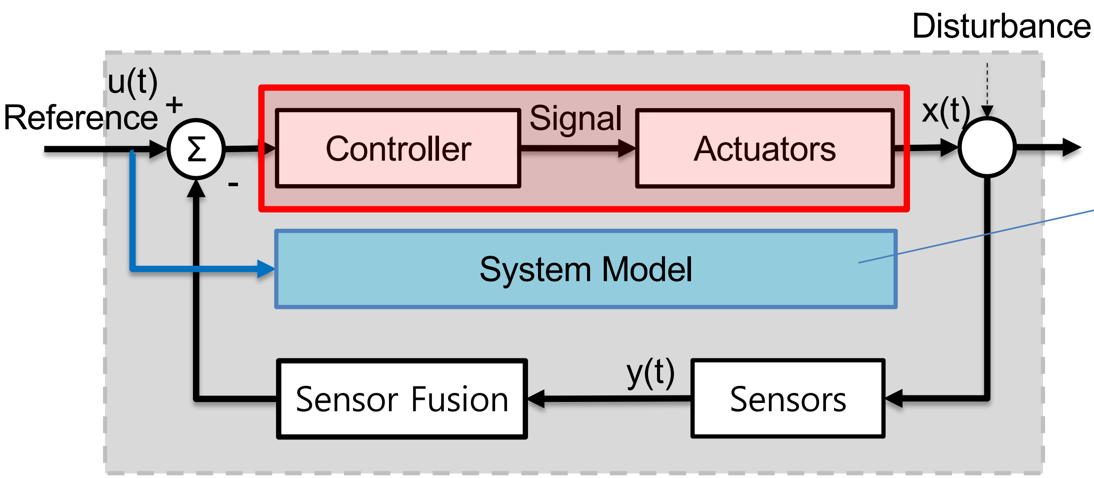
- Efficiency: Spatial & Temporal
- Intrinsic errors
  - Model inaccuracy
  - Conversion errors
  - External disturbances
- Determining parameters

Design Overview



System Identification

- ❖ System model predicts physical states changes



States [$x\ y\ z\ \boldsymbol{\phi}\ \boldsymbol{\theta}\ \boldsymbol{\psi}\ \dot{x}\ \dot{y}\ \dot{z}\ p\ q\ r$]

State-space equation

$$x' = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

System Identification determines ABCD matrices

$$x' = \begin{bmatrix} 0.9884 & -0.0493 & -0.0242 \\ 0.0025 & 0.9999 & 0 \\ 0 & 0.0025 & 1.0 \end{bmatrix} x(t) + \begin{bmatrix} 0.0025 \\ 0 \\ 0 \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} 1.8651 & 16.8655 & 10.0631 \end{bmatrix} x(t) + \begin{bmatrix} 0 \end{bmatrix} u(t)$$

Software Sensors

❖ Conversion Operation

– Convert predicted model states to sensor readings

– Conversion equation for each sensor with coordinate system transformation

Model States (12 states)

$[\ x\ y\ z\ \phi\ \theta\ \psi\ \dot{x}\ \dot{y}\ \dot{z}\ p\ q\ r\ ]$

angle roll
angle pitch
angle yaw
angular rate roll
angular rate pitch
angular rate yaw
position x
position y
position z
velocity x,
velocity y
velocity z

Transformation
(with error collection)

Sensor measurements

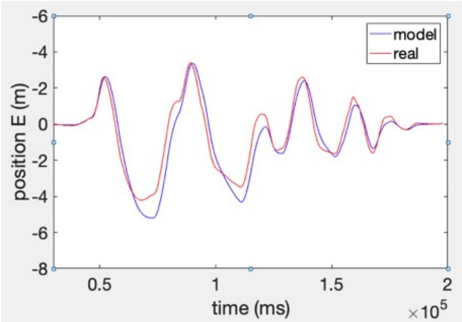
Gyroscope
(angular rates)

GPS
(position x y z)

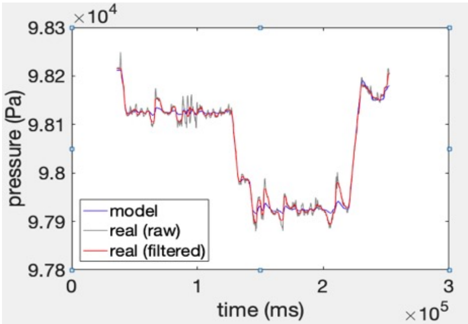
Accelerometer
acceleration $= \frac{(v_t - v_{t-1})}{dt}$

Barometer
pressure_from_base $P_h = P_0 \cdot exp\left[\frac{-g_0 \cdot M \cdot (z - h_0)}{R \cdot T_0}\right]$, z: position z

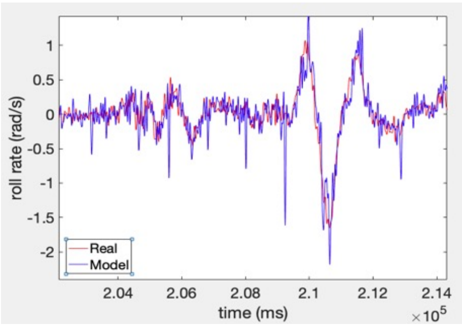
Software Sensors



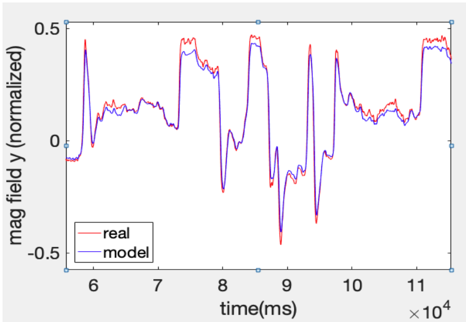
GPS sensor



Barometer



Gyroscope



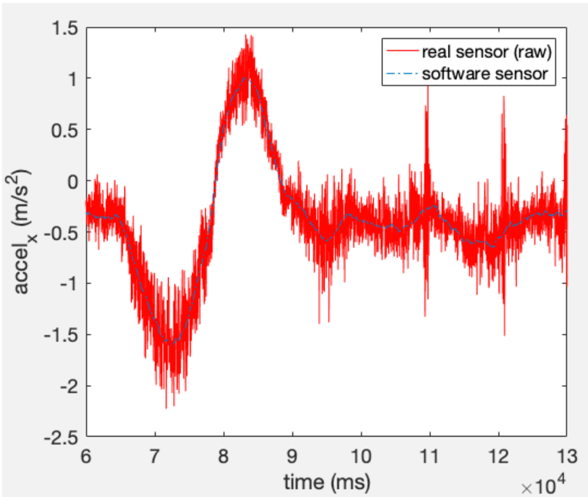
Magnetometer

Practical Challenges

- ❖ Practical Limitations – Inaccuracy
  - – Conversion Error
  - – Model Inaccuracy
  - – External disturbances
- ❖ Errors are accumulated over time

Error Correction Techniques

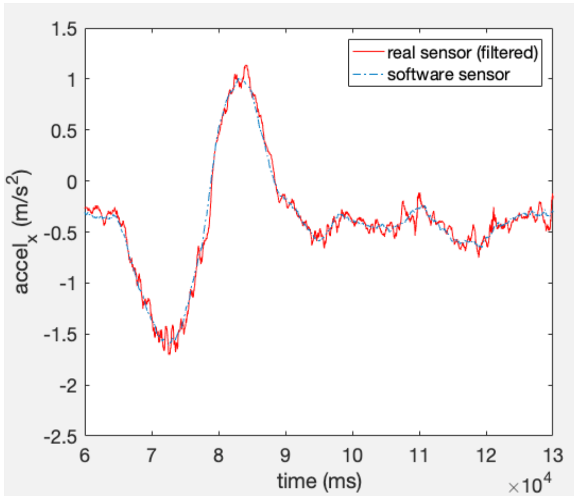
❖ Conversion error correction



Raw measurement

Low-pass filter

Smooth noise-robust differentiator

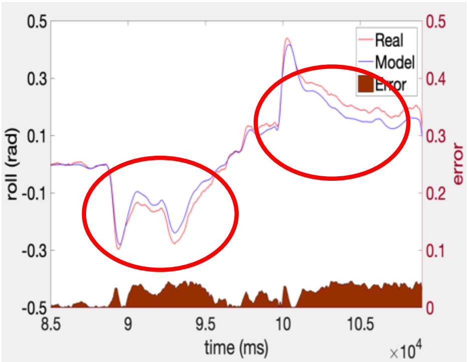


Corrected

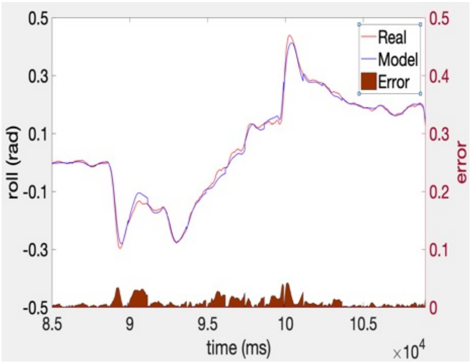
Error Correction Techniques

❖ Model error correction

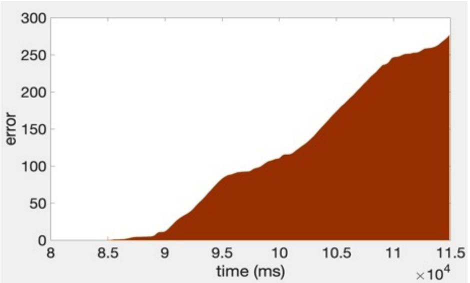
Prediction drift
Approximation Error
b/w Real and Model



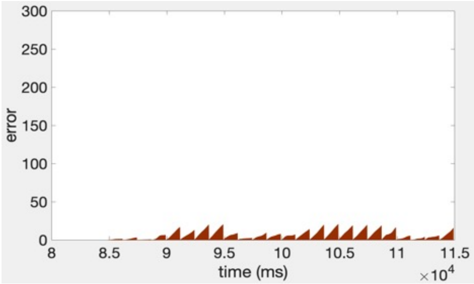
Synchronization
with small time windows



Accumulated error

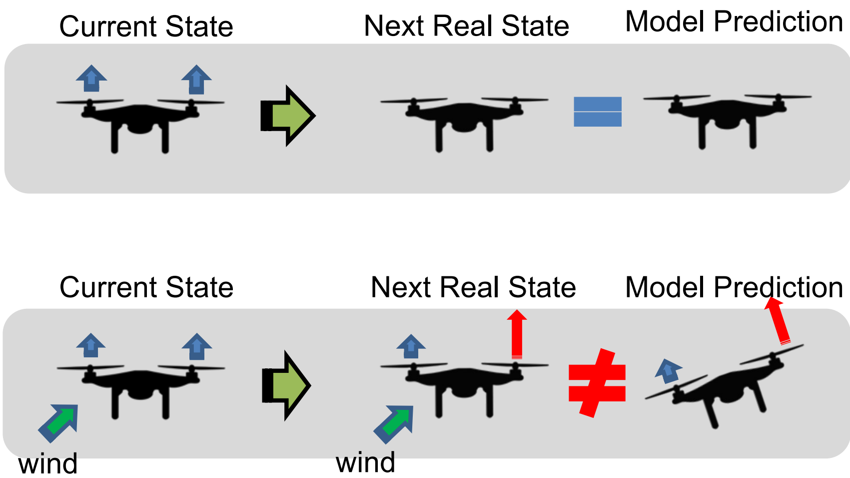


Error reset
at every window

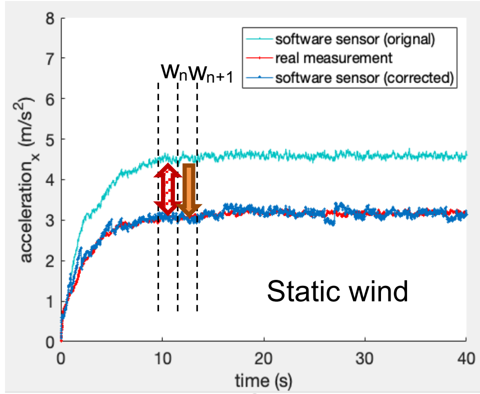


Error Correction Techniques

❖ External Error Correction

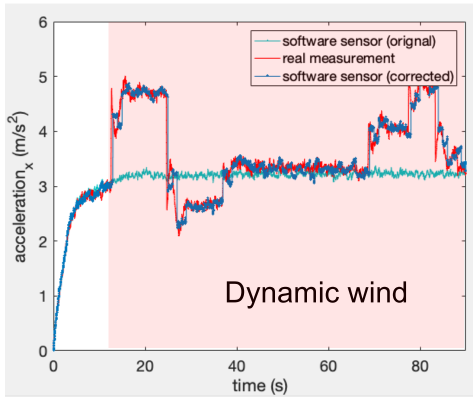


state differences due to external forces



Calculate an effect of wind forces in W_n

Compensate the forces in W_{n+1}



Evaluation: Subject Systems

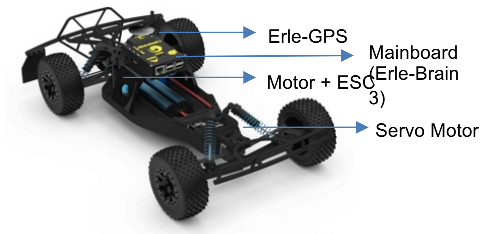
❖ 6 Vehicles (2 real / 4 simulated vehicles)

Type	Model	Controller Software	Number of Sensors				
			G	A	M	B	P
Quadrotor	APM SITL	ArduCopter 3.4	2	2	1	1	1
Hexacopter	APM SITL	ArduCopter 3.6	2	2	1	1	1
Rover	APM SITL	APMrover2 2.5	2	2	1	1	1
Quadrotor	Erle-Copter	ArduCopter 3.4	2	2	1	1	1
Rover	Erle-Rover†	APMrover2 3.2	1	1	2	1	1
Quadrotor	3DR Solo†	APM:solo 1.3.1	3	3	3	2	1

* G: gyroscope, A: accelerometer, M: magnetometer, B: barometer, P: GPS

† Real Vehicles

3DR Solo



Erle-Rover

Evaluation: Setting

❖ Attack

- Simulate the physical attack with an attack code
- Modify sensor readings in sensor interfaces
- Controlled attack (e.g., random, selected values)

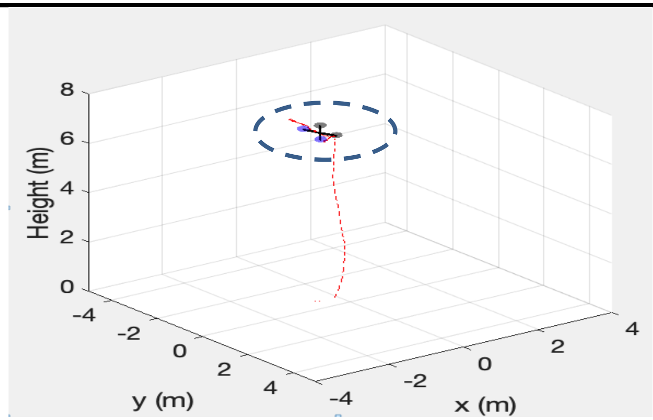
❖ Recovery

$$R_{succ} := |Y_t - \bar{Y}_t| \leq \varepsilon, t \in [1 \ldots k]$$

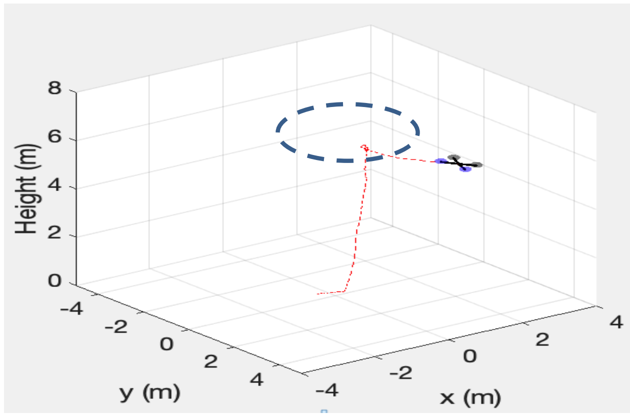
Y_t : real state $\overline{Y}_t$: prediction
$\boldsymbol{\varepsilon}$: error margin k : time for recovery success

$\boldsymbol{\varepsilon}$ = 3
k = 10

Success

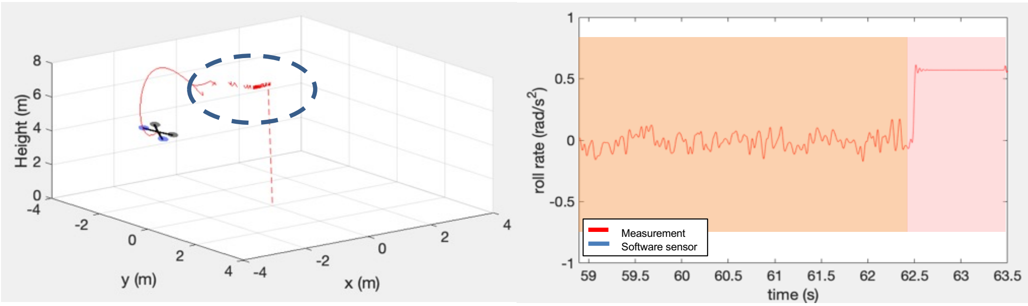


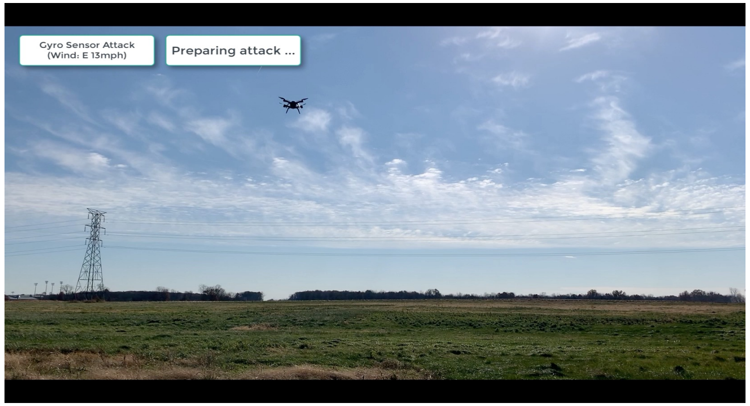
Fail



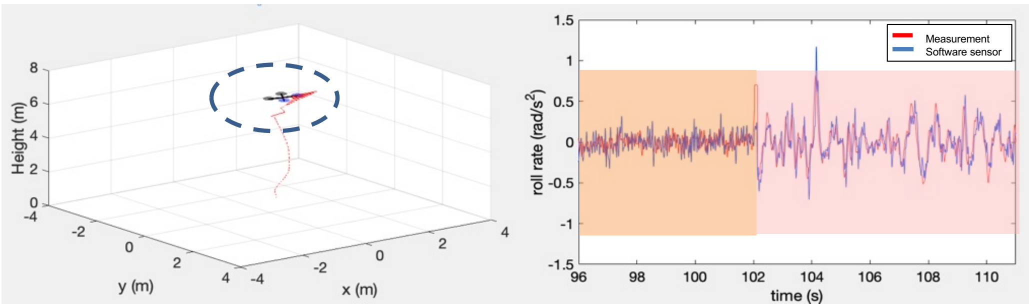
Gyro Attack Recovery on 3DR Solo

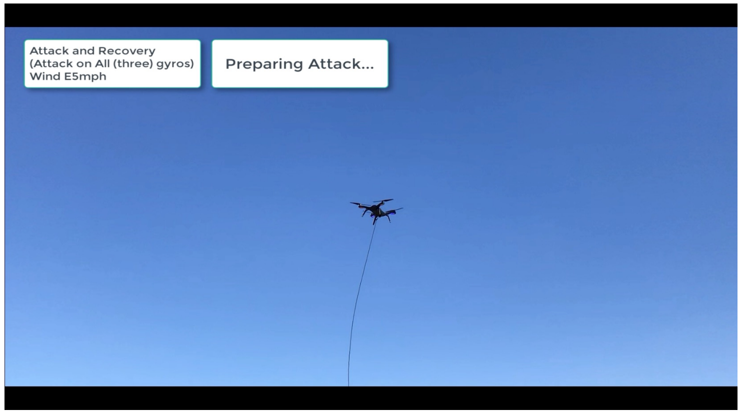
Gyro Attack





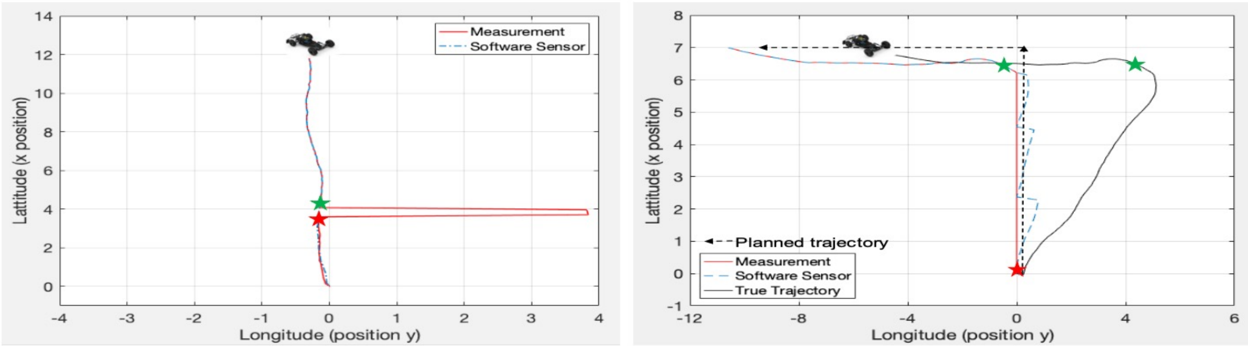
Gyro Attack Recovery





Stealthy GPS Attack on Erle-rover

Advanced Stealthy GPS attack:
Random/Controlled Attack and Recovery



(a) Random Attack (b) Controlled Attack

Attack Combination and Result Highlights

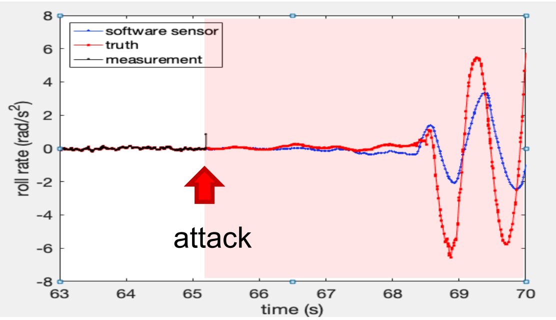
Test#	GPS	Barometer 1 2	Gyroscope 1 2 3	Recovered
C1	Compromised	Benign	Benign	✓
C2	Benign	Compromised	Benign	✓
C3	Benign	Benign	Compromised	✓†
C4	Compromised	Compromised	Benign	✓
C5	Compromised	Benign	Compromised	✓†
C6	Compromised	Compromised	Compromised	✓†

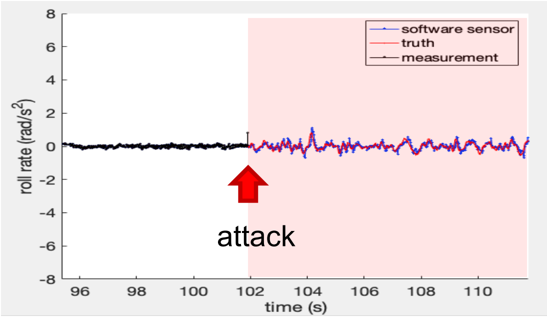
✓: success, †Supplementary Compensation Applied

All Recovered

Gyroscope ← Accelerometer + Magnetometer

Supplementary Compensation Applied



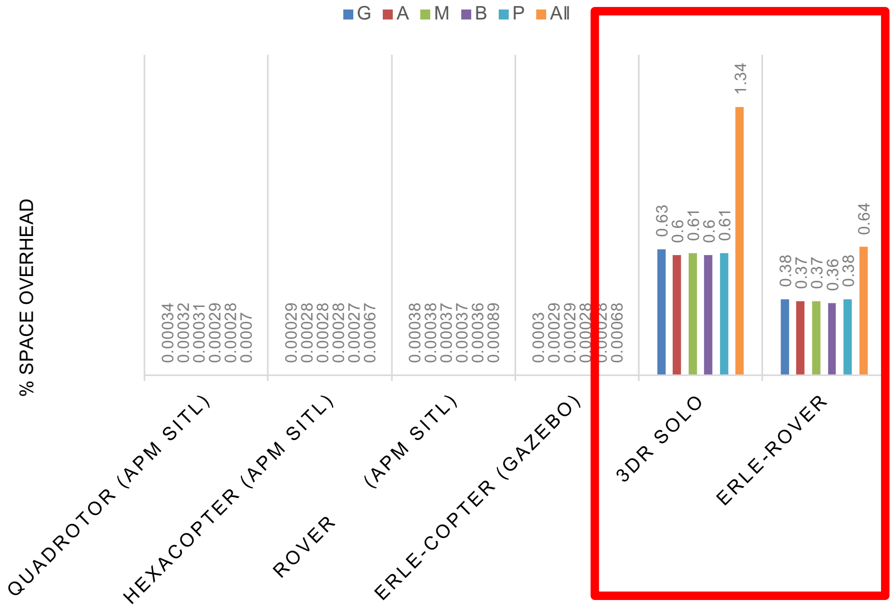


Without compensation

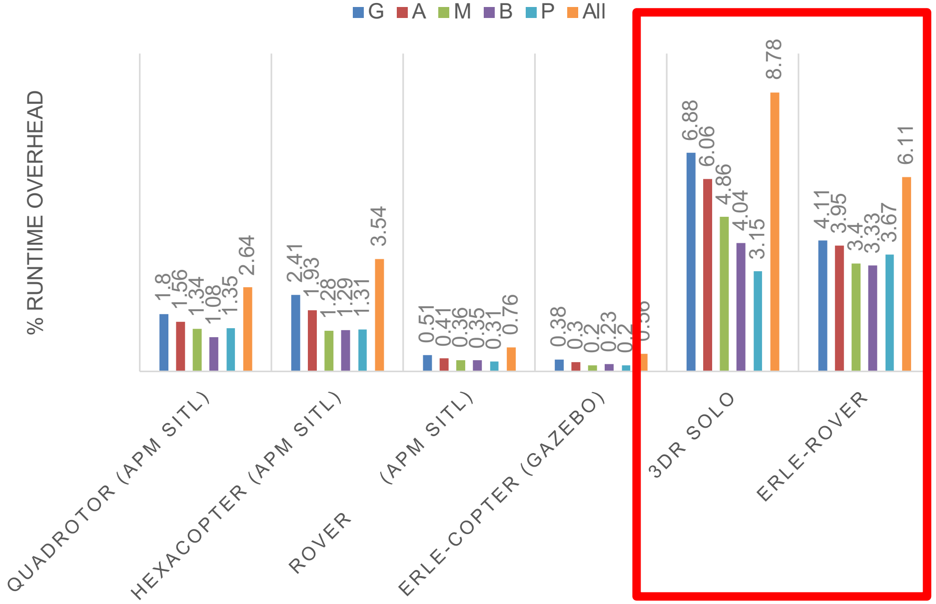
With compensation

Performance Overhead

❖ Space Overhead



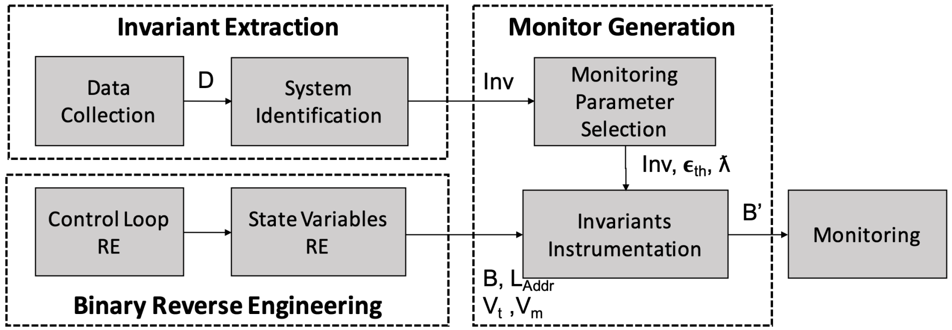
❖Runtime Overhead



Related Work (Previous)

❖ Choi, Hongjun, et al. "Detecting attacks against robotic vehicles: A control invariant approach." *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 2018.

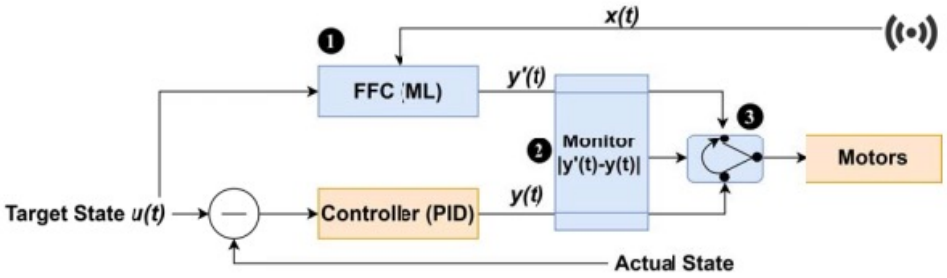
- Expect state output based on system modeling
- If the accumulated error in monitor window exceed a threshold, alarms the attack attempt.



Related Work (Work after this paper)

❖ Dash, Pritam, et al. "Pid-piper: Recovering robotic vehicles from physical attacks." *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2021.

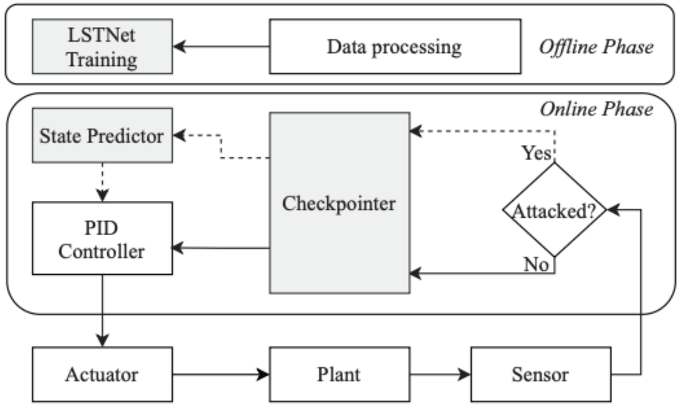
- ML-based Feed Forward Controller
- FFC replaces PID controller if an attack is detected



Related Work (Work after this paper)

❖ Akowuah, Francis, et al. "Recovery-by-learning: Restoring autonomous cyber-physical systems from sensor attacks." *2021 IEEE 27th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*. IEEE, 2021.

- LSTNet training for prediction model that exploits the temporal correlation among heterogeneous sensors
- Checkpointer saves normal behavior if no attack detected.
- If an attack is detected, state predictor generates proper input based on checkpoints.



Conclusion

- ❖ They proposed a novel software-sensor based real-time recovery technique for RVs
  - Support heterogeneous multiple sensor recovery
- ❖ The technique can’t recover from..
  - Accumulated error during the recovery window
  - Undetectable small error attacks
- ❖ Evaluations were not persuasive
  - Why not real attack?
  - The explanation of attacks are not specific
  - Why only hovering?

Thank You!