Let's not Trust Experience Blindly: Formal Runtime Verification for CPS

Maximilian Schwenger

Joint work with Jan Baumeister, Peter Faymonville, Bernd Finkbeiner, Malte Schledjewski, Marvin Stenger, Leander Tentrup, Hazem Torfah

Saarland University

Static Verification

System S | Controller C | Specification φ

Verify:

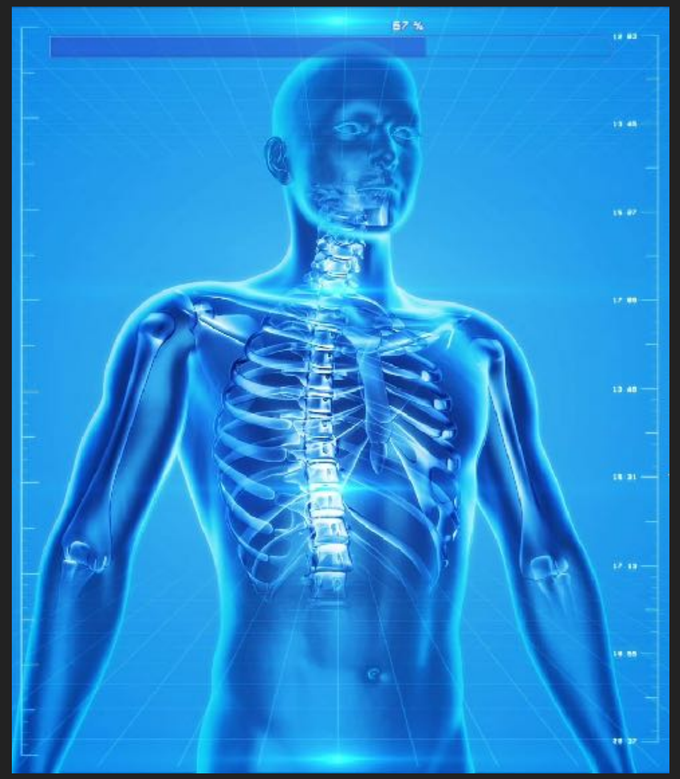
$$\forall \sigma \in \mathit{runs}(S \parallel C): \sigma \models \varphi$$

When Static Verification Fails

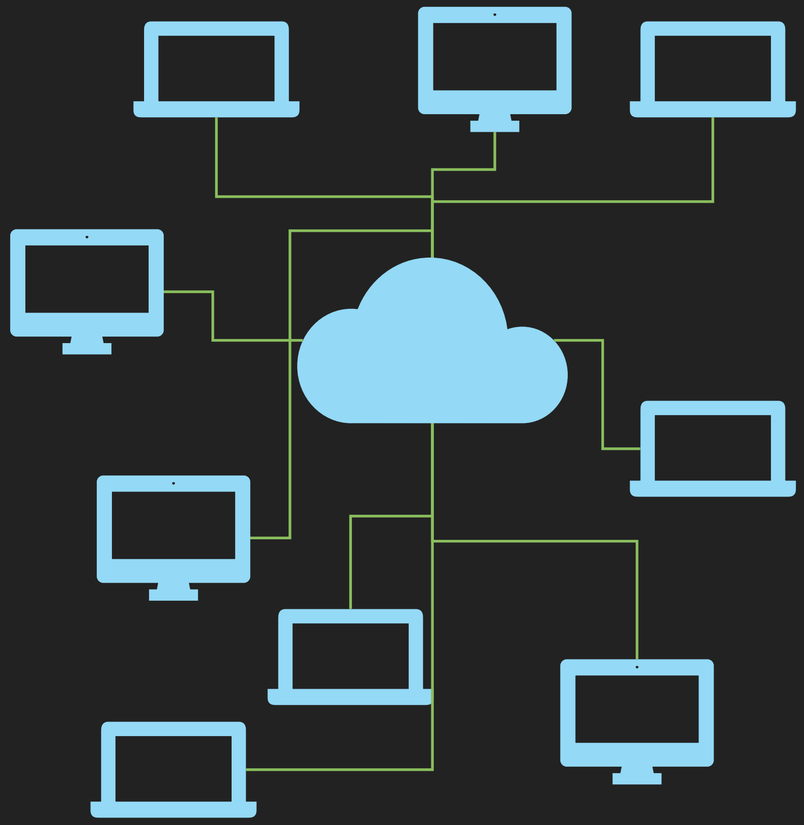
Complexity

$$\dot{p} = Rv$$
$$\dot{R} = R\hat{\omega}$$
$$\dot{v} = -\omega \times v + R^T \bar{g} + f_v(\omega, v, \alpha, \beta, \omega_r, \delta_c, \delta_r)$$
$$\dot{\omega} = -J^{-1}(\omega \times J\omega) + f_w(\omega, v, \alpha, \beta, \omega_r, \delta_c, \delta_r)$$
$$\dot{\alpha} = f_\alpha(\omega, v, \alpha, \beta, \omega_r, \delta_a, \delta_e)$$
$$\dot{\beta} = f_\beta(\omega, v, \alpha, \beta, \omega_r, \delta_a, \delta_e)$$
$$\dot{\omega}_r = f_r(\omega, v, \omega_r, \delta_c, \delta_r)$$

Lack Of Knowledge



Non-Determinism



Static Verification

System S

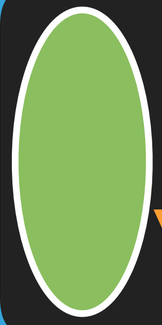
Controller C

Specification φ

Verify:

$\forall \sigma \in \mathit{runs}(S \parallel C): \sigma \models \varphi$

Testing



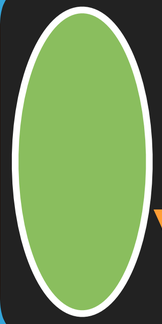
Verify:

$\forall \sigma \in runs(S \parallel C): \sigma \models \varphi$

$\exists S' \subseteq runs(S \parallel C):$

$\forall \sigma \in S': \sigma \models \varphi$

Runtime Verification



Verify:

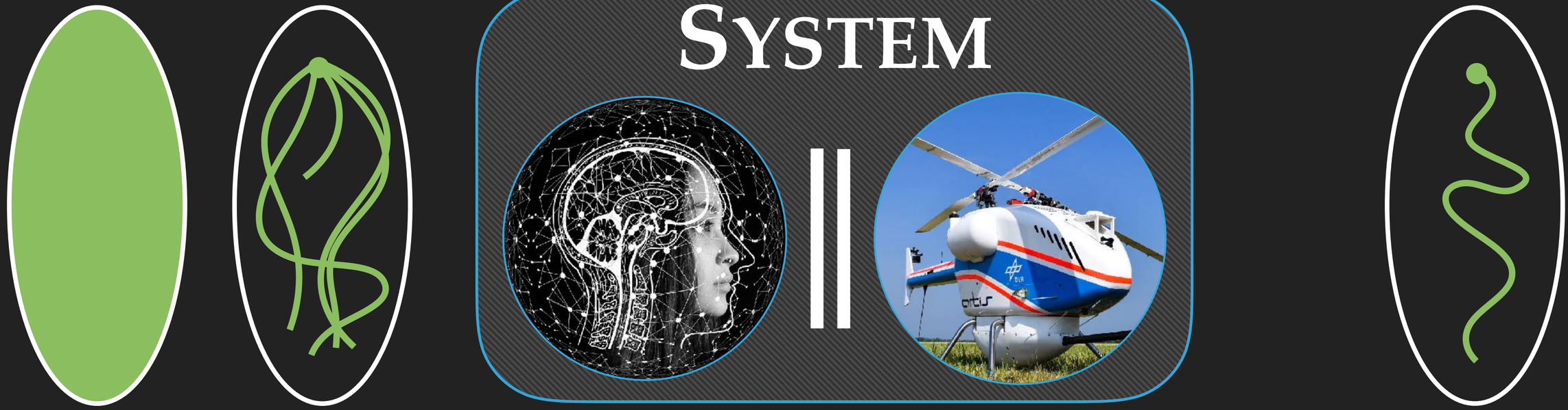
$\forall \sigma \in runs(S \parallel C): \sigma \models \varphi$

Given $\sigma \in runs(S \parallel C)$:

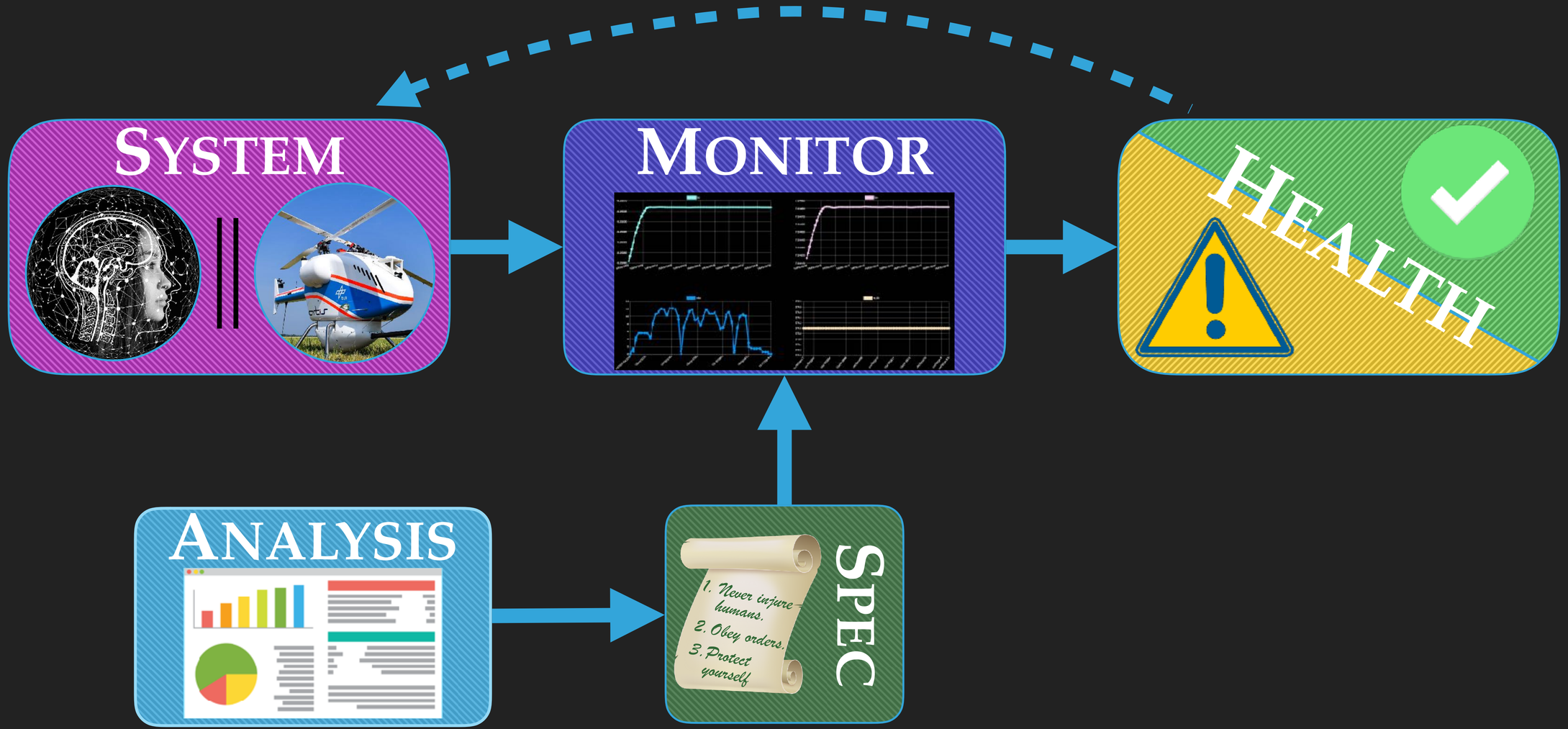
$\sigma \models \varphi$

Prior to Deployment

After Deployment



Our Take on Runtime Verification



Stream-based Runtime Verification

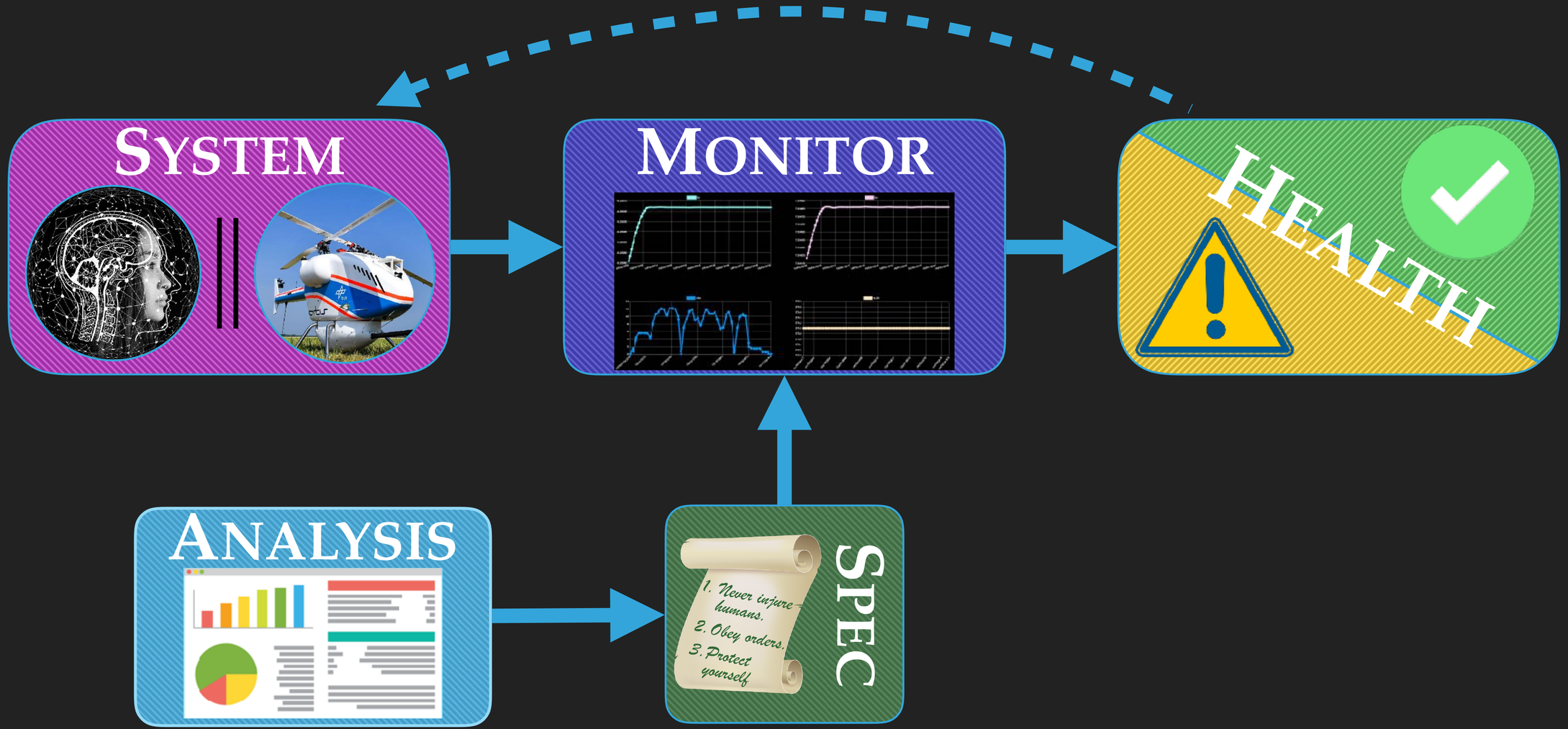
Output Streams

Trigger

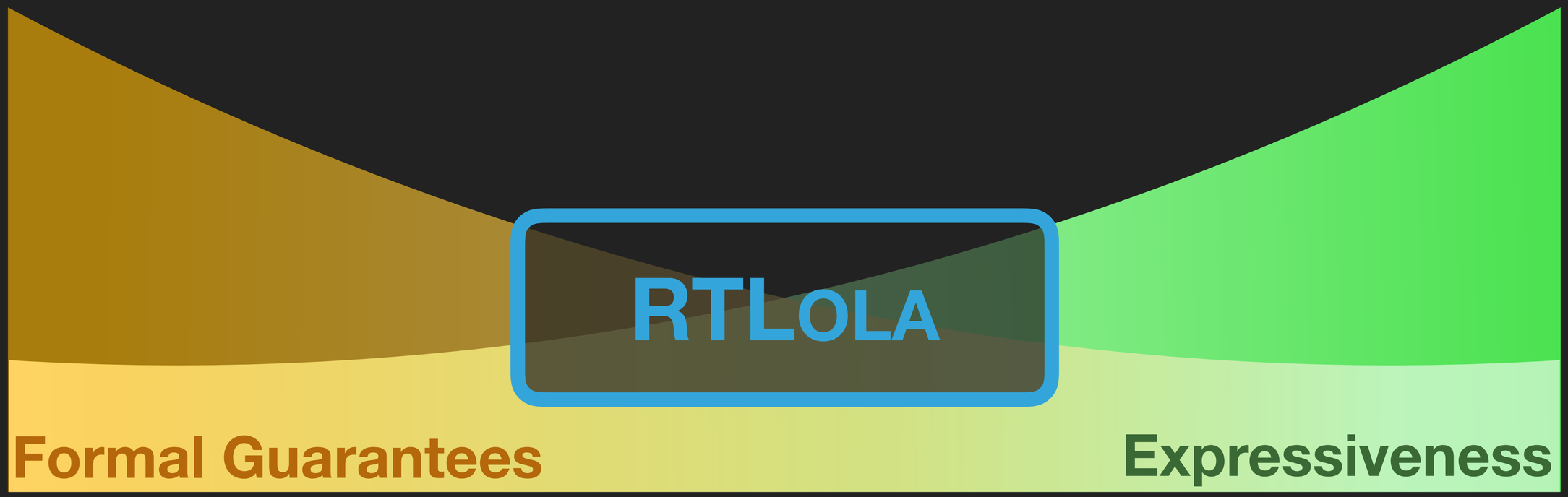
Trigger

Runtime Monitor

Our Take on Runtime Verification



Specification Languages



RTLola in a Nutshell

```
input lat, lon: Float64 // from GPS
input accel_x: Float64  // from accelerometer
input slow_down_cmd: Bool
```

The GPS module operates with at least 5Hz.

RTLola in a Nutshell

```
input lat, lon: Float64 // from GPS
input accel_x: Float64  // from accelerometer
input slow_down_cmd: Bool

output gps_samples @1Hz := lat.aggregate(over_exactly: 1s, using: count)
trigger gps_samples < 5 “GPS frequency less than 5 Hz.”
```

Accelerometer and GPS readings coincide.

RTLola in a Nutshell

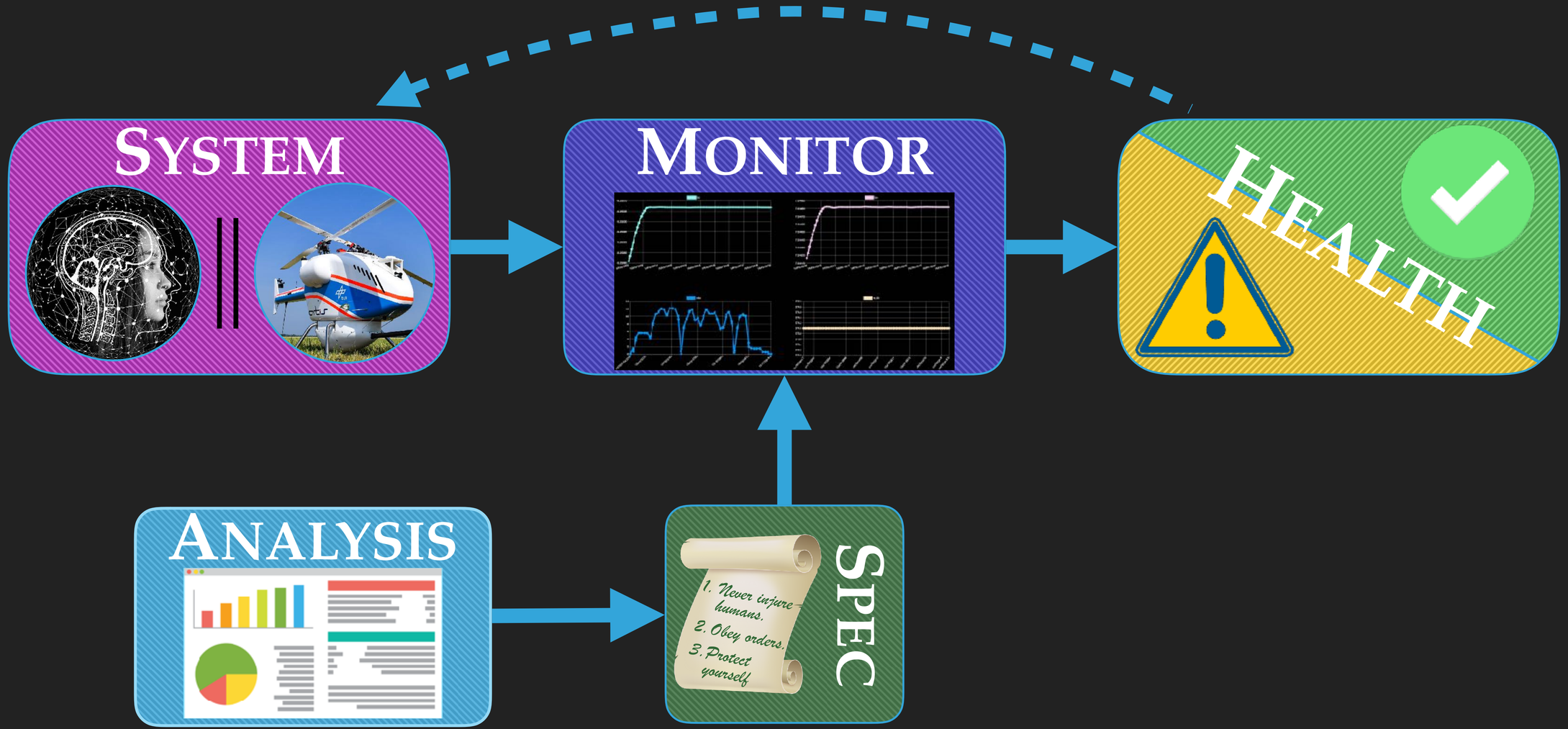
```
input lat, lon: Float64 // from GPS
input accel_x: Float64  // from accelerometer
input slow_down_cmd: Bool

output gps_samples @1Hz := lat.aggregate(over_exactly: 1s, using: count)
trigger gps_samples < 5 “GPS frequency less than 5 Hz.”

output accel_velo @1Hz := accel_x.aggregate(over: 5s, using: ∫)
output gps_velo @1Hz := lon.aggregate(over: 5s, using: ∇)
trigger abs(accel_velo - gps_velo) > 0.1 “Conflicting measurements for velocity.”
```

A slow-down is preceded by the respective command.

RTLola in a Nutshell

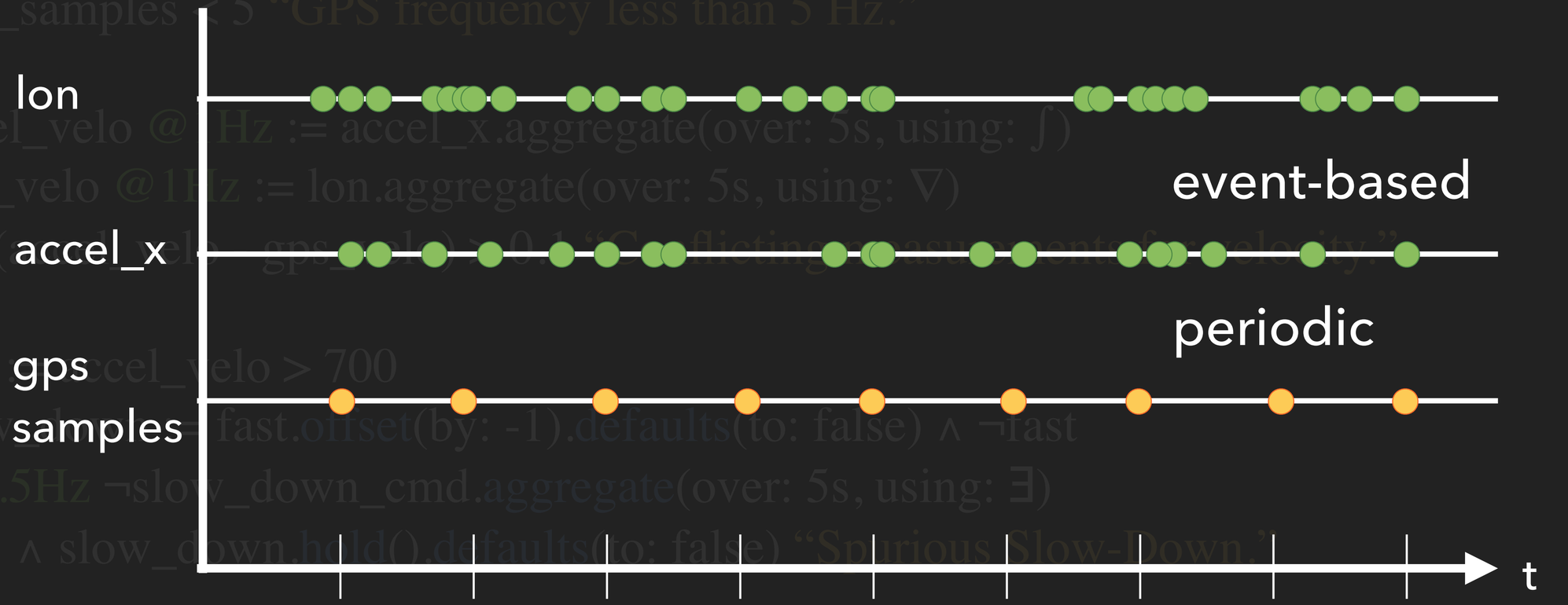
```
input lat, lon: Float64 // from GPS
input accel_x: Float64  // from accelerometer
input slow_down_cmd: Bool

output gps_samples @1Hz := lat.aggregate(over_exactly: 1s, using: count)
trigger gps_samples < 5 “GPS frequency less than 5 Hz.”

output accel_velo @1Hz := accel_x.aggregate(over: 5s, using: ∫)
output gps_velo @1Hz := lon.aggregate(over: 5s, using: ∇)
trigger abs(accel_velo - gps_velo) > 0.1 “Conflicting measurements for velocity.”

output fast := accel_velo > 700
output slow_down := fast.offset(by: -1).defaults(to: false) ∧ ¬fast
trigger @1Hz ¬slow_down_cmd.aggregate(over: 5s, using: ∃)
                ∧ slow_down.hold().defaults(to: false) “Spurious Slow-Down.”
```

Our Take on Runtime Verification



Strong Type System

```
input lat, lon: Float64 // from GPS
input accel_x: Float64  // from accelerometer
input slow_down_cmd: Bool

output gps_samples @1Hz := lat.aggregate(over_exactly: 1s, using: count)
trigger gps_samples < 5 “GPS frequency less than 5 Hz.”
output accel_velo @1Hz := accel_x.aggregate(over: 5s, using: ∫)
output gps_velo @1Hz := lon.aggregate(over: 5s, using: ∇)
trigger abs(accel_velo - gps_velo) > 0.1 “Conflicting measurements for velocity.”

output fast := accel_velo > 700
output slow_down := fast.offset(by: -1).defaults(to: false) ∧ ¬fast
trigger @0.5Hz ¬slow_down_cmd.aggregate(over: 5s, using: ∃)
    ∧ slow_down.hold().defaults(to: false) “Spurious Slow-Down.”
```



Strong Type System

```
input lat, lon: Float64 // from GPS
input accel_x: Float64  // from accelerometer
input slow_down_cmd: Bool
```

Float64	{lon}

Float64	{lat}
Float64	{acc}

Bool	{cmd}

```
output gps_samples @1Hz := lat.aggregate(over_exactly: 1s, using: count)
trigger gps_samples < 5 “GPS frequency less than 5 Hz.”
```

UInt64	1Hz

```
output accel_velo @1Hz := accel_x.aggregate(over: 5s, using: ∫)
output gps_velo @1Hz := lon.aggregate(over: 5s, using: ∇)
trigger abs(accel_velo - gps_velo) > 0.1 “Conflicting measurements for velocity.”
```

Float64	1Hz
Float64	1Hz

```
output fast := accel_velo > 700
output slow_down := fast.offset(by: -1).defaults(to: false) ∧ ¬fast
trigger @0.5Hz ¬slow_down_cmd.aggregate(over: 5s, using: ∃)
               ∧ slow_down.hold().defaults(to: false) “Spurious Slow-Down.”
```

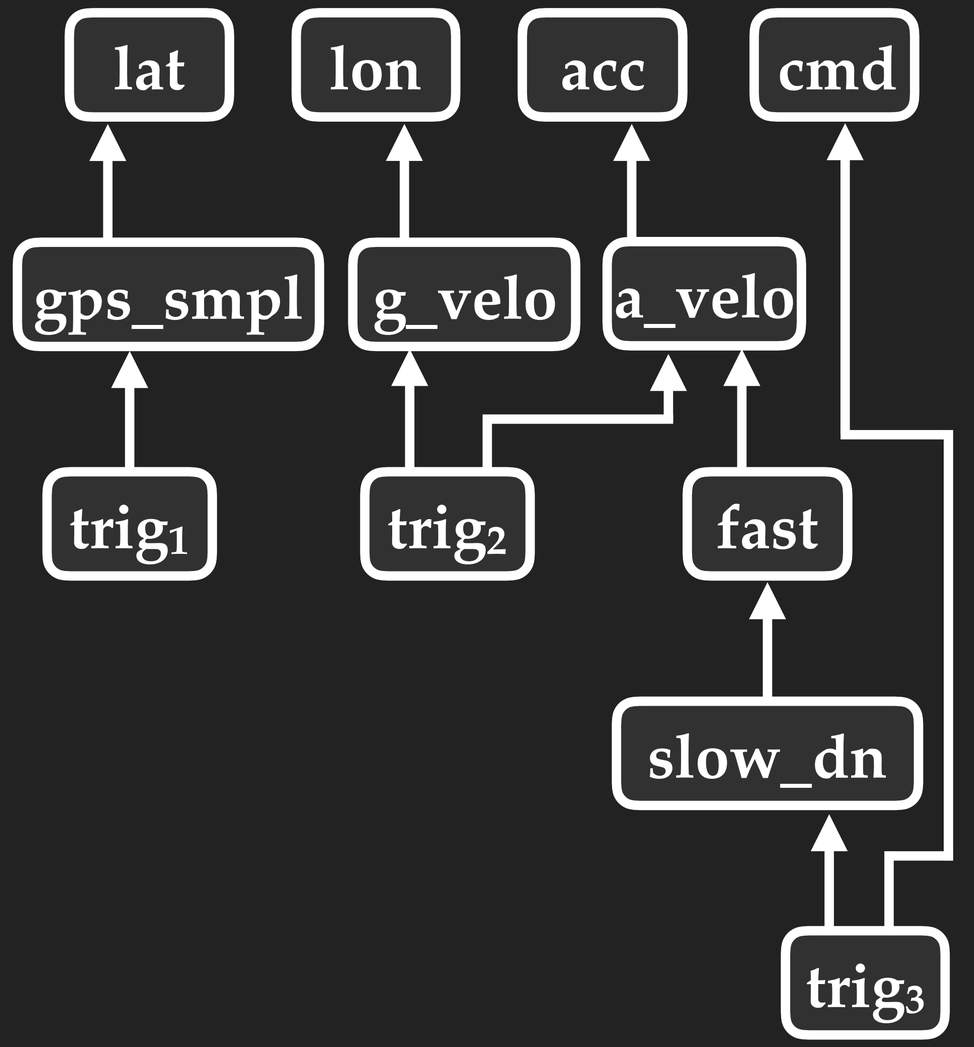
Bool	1Hz

Bool	1Hz
Bool	0.5Hz

Specification

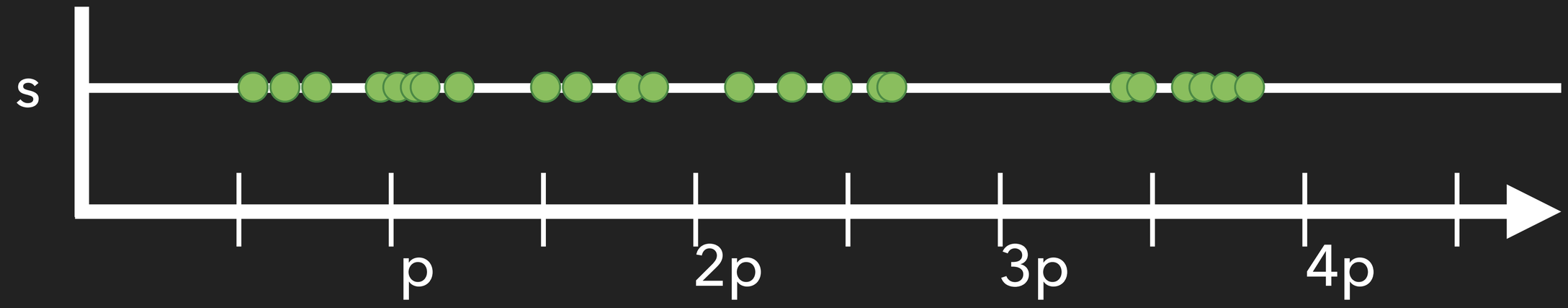
```
input lat, lon: Float64 // from GPS
input accel_x: Float64  // from accelerometer
input slow_down_cmd: Bool

output gps_samples @1Hz := lat.aggregate(over_exactly: 1s, using: count)
trigger gps_samples < 5 “GPS frequency less than 5 Hz.”

output accel_velo @1Hz := accel_x.aggregate(over: 5s, using: ∫)
output gps_velo @1Hz := lon.aggregate(over: 5s, using: ∇)
trigger abs(accel_velo - gps_velo) > 0.1
    “Conflicting measurements for velocity.”

output fast := accel_velo > 700
output slow_down := fast.offset(by: -1).defaults(to: false) ∧ ¬fast
trigger @1Hz ¬slow_down_cmd.aggregate(over: 5s, using: ∃)
    ∧ slow_down.hold().defaults(to: false) “Spurious Slow-Down.”
```

Dependency Graph



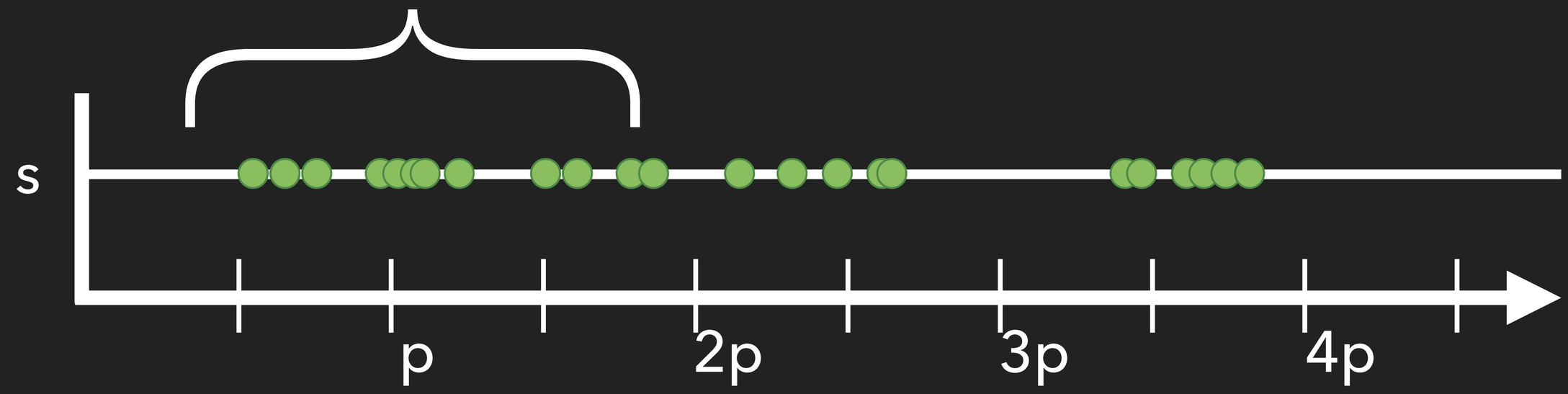
Sliding Windows

output h := **s**.aggr(over: 1.5p, using: γ)



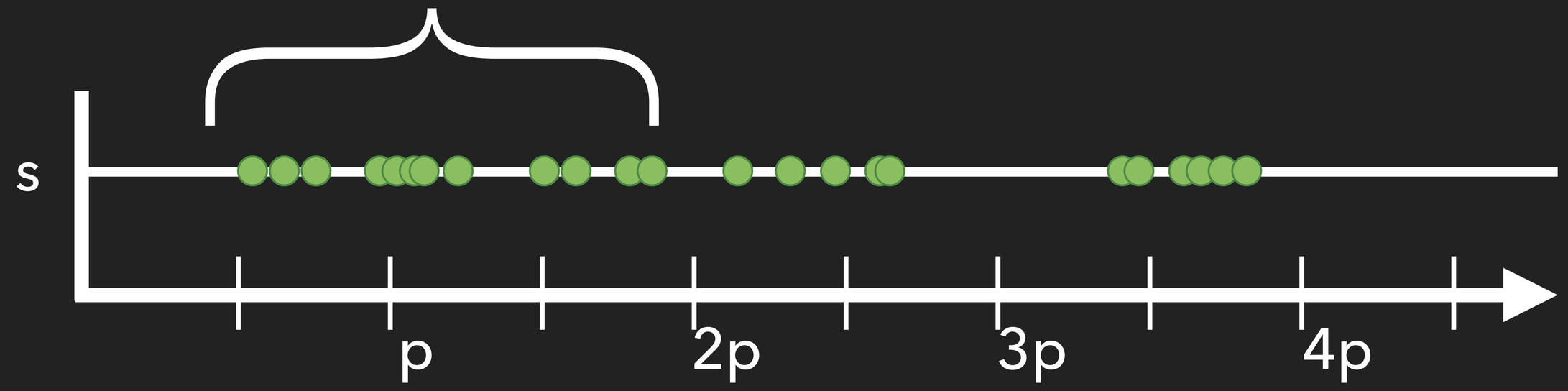
Sliding Windows

output h := **s**.aggr(over: 1.5p, using: γ)



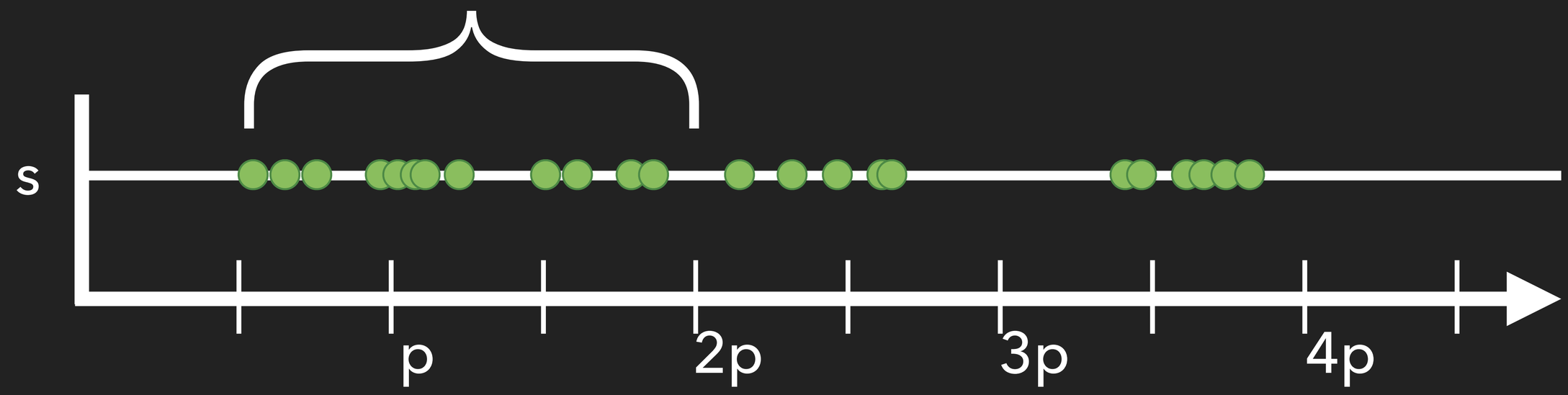
Sliding Windows

output h := **s**.aggr(over: 1.5p, using: γ)



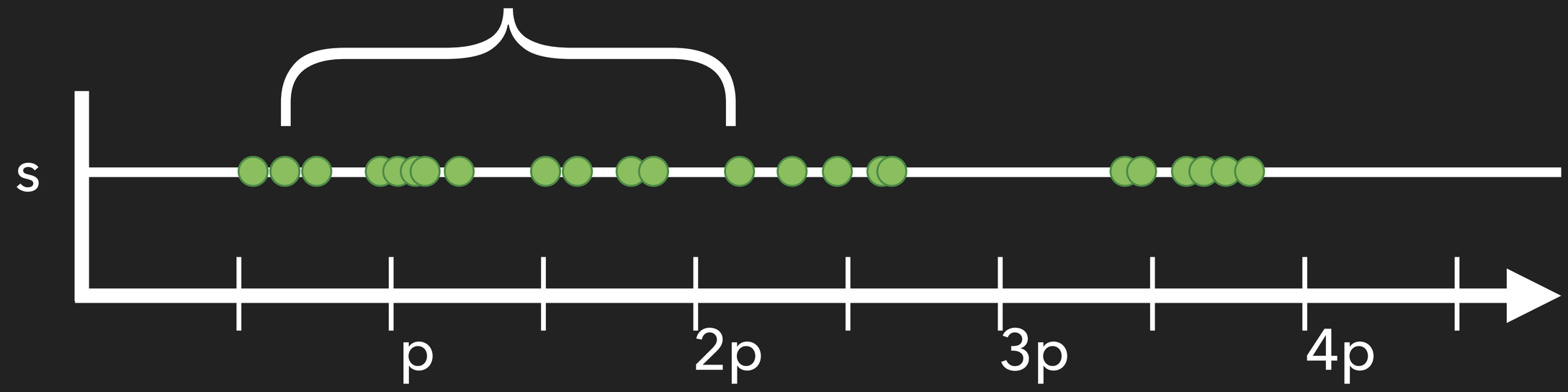
Sliding Windows

output h := **s**.aggr(over: 1.5p, using: γ)



Sliding Windows

output h := **s**.aggr(over: 1.5p, using: γ)



List Homomorphisms

output h := s.aggr(over: 1.5p, using: γ)

$\gamma : A^* \to B$

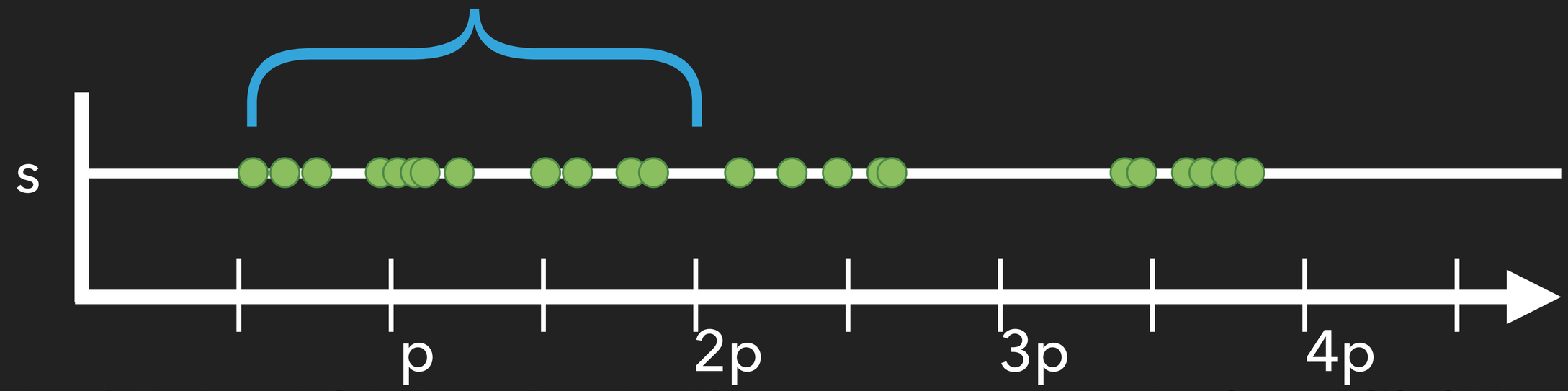
$\mathsf{map}_\gamma : A \to T \qquad \mathsf{fin}_\gamma : T \to B \qquad \circ_\gamma : T \times T \to T$

$\gamma(v_1, \ldots, v_n) = \mathsf{fin}_\gamma(\mathsf{map}_\gamma(v_1) \circ_\gamma \ldots \circ_\gamma \mathsf{map}_\gamma(v_n))$

- Meertens: "*Algorithmics: Towards programming as a mathematical activity*", 1986

Sliding Windows

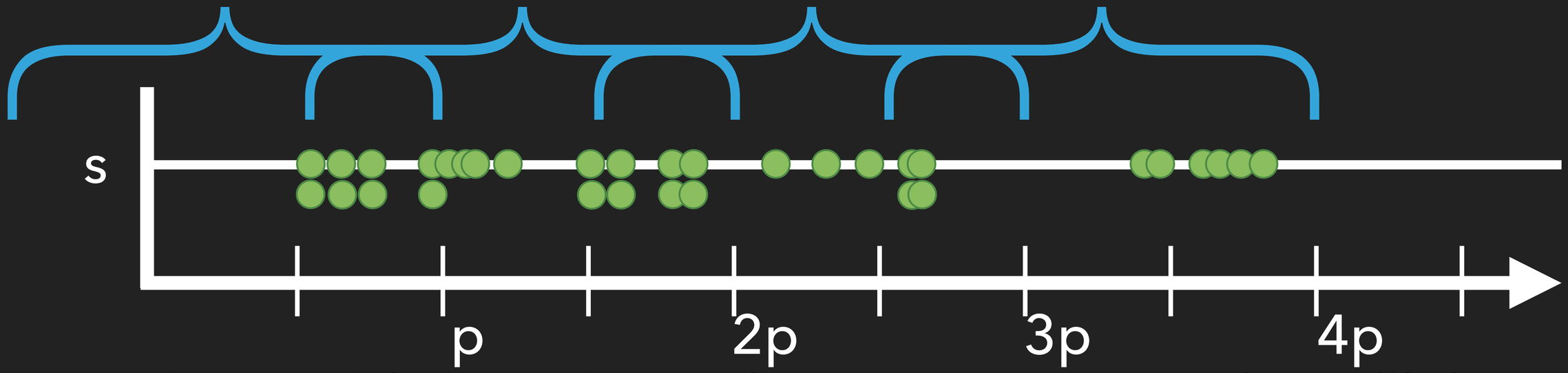
output **h** $@p^{-1}Hz$:= **s**.aggr(over: 1.5p, using: γ)



- Li et al.: "*No Pane, No Gain: Efficient Evaluation of Sliding-window Aggregates over Data Streams*", SIGMOD Rec. 2005
- Schwenger: "*Let's not Trust Experience Blindly: Formal Monitoring of Humans and other CPS*", Master Thesis 2019

Sliding Windows

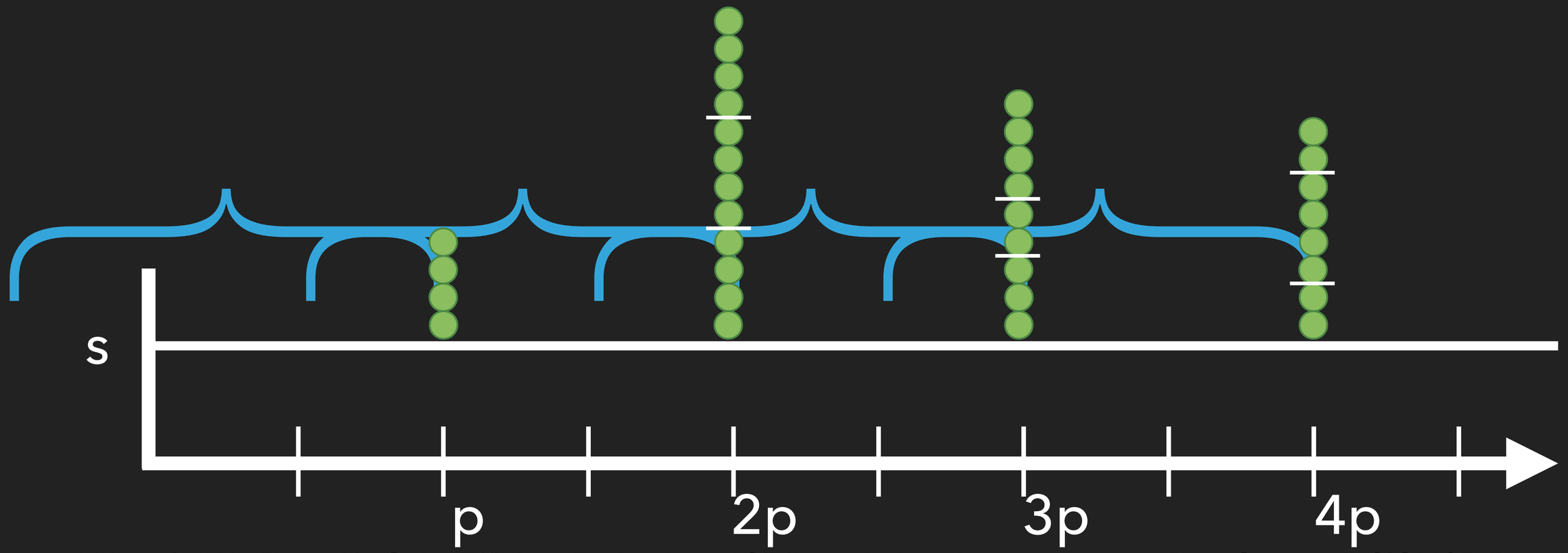
output **h** $@p^{-1}Hz$:= **s**.aggr(over: 1.5p, using: γ)



- Li et al.: "*No Pane, No Gain: Efficient Evaluation of Sliding-window Aggregates over Data Streams*", SIGMOD Rec. 2005
- Schwenger: "*Let's not Trust Experience Blindly: Formal Monitoring of Humans and other CPS*", Master Thesis 2019

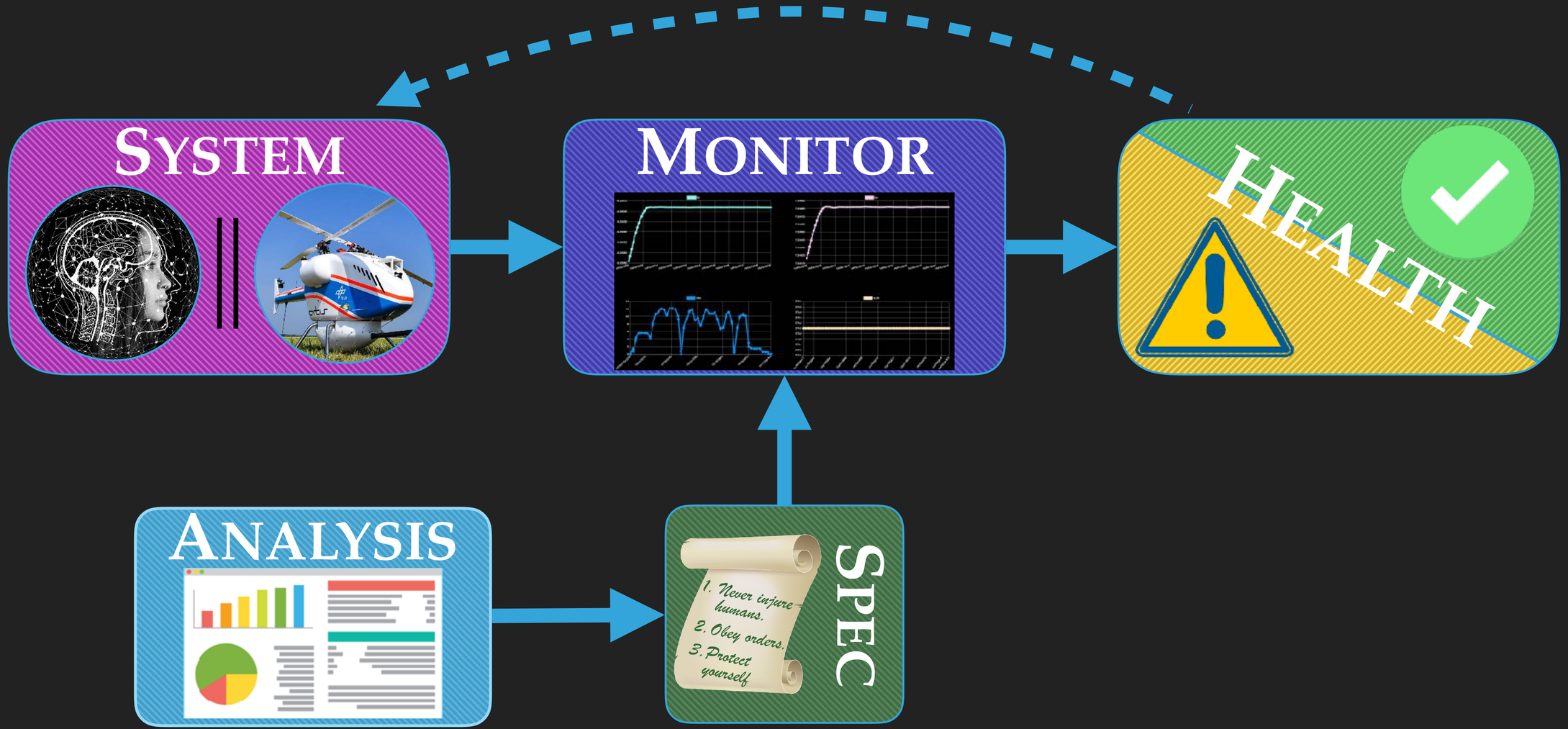
Sliding Windows

output **h** $@p^{-1}Hz$:= **s**.aggr(over: 1.5p, using: γ)

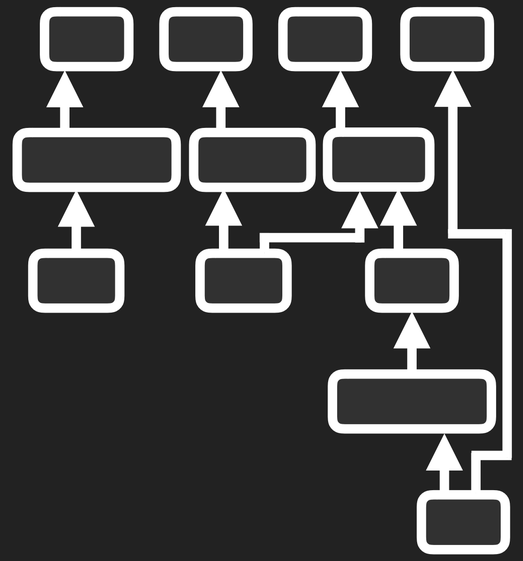


- Li et al.: "*No Pane, No Gain: Efficient Evaluation of Sliding-window Aggregates over Data Streams*", SIGMOD Rec. 2005
- Schwenger: "*Let's not Trust Experience Blindly: Formal Monitoring of Humans and other CPS*", Master Thesis 2019

Our Take on Runtime Verification



StreamLAB

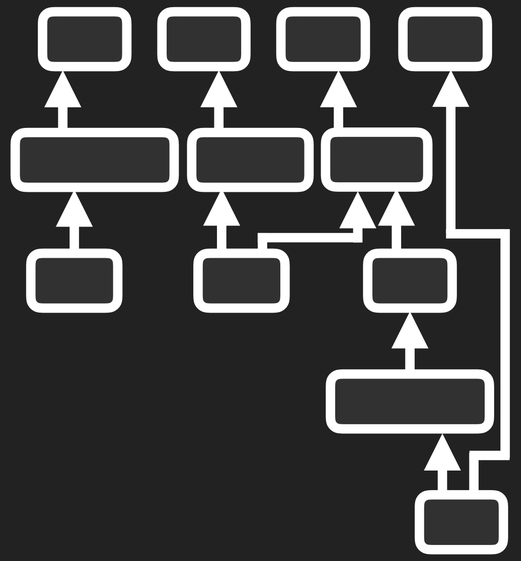


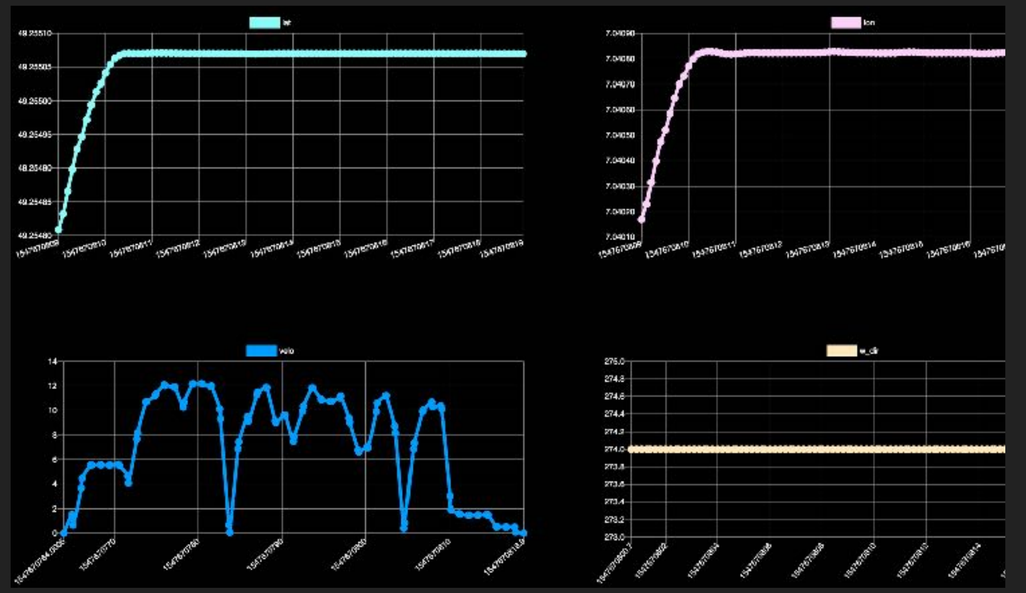
RTLola Specification

Annotated DG Intermediate Rep.

Backend

StreamLAB





Graphical UI developed by Sanny schmitt

RTLola Specification

Annotated DG Intermediate Rep.

Rust Interpretation

Rust Interpreter

Specification:

GPS frequency validation

GPS/IMU jump detection

Hover phase detection

Results:

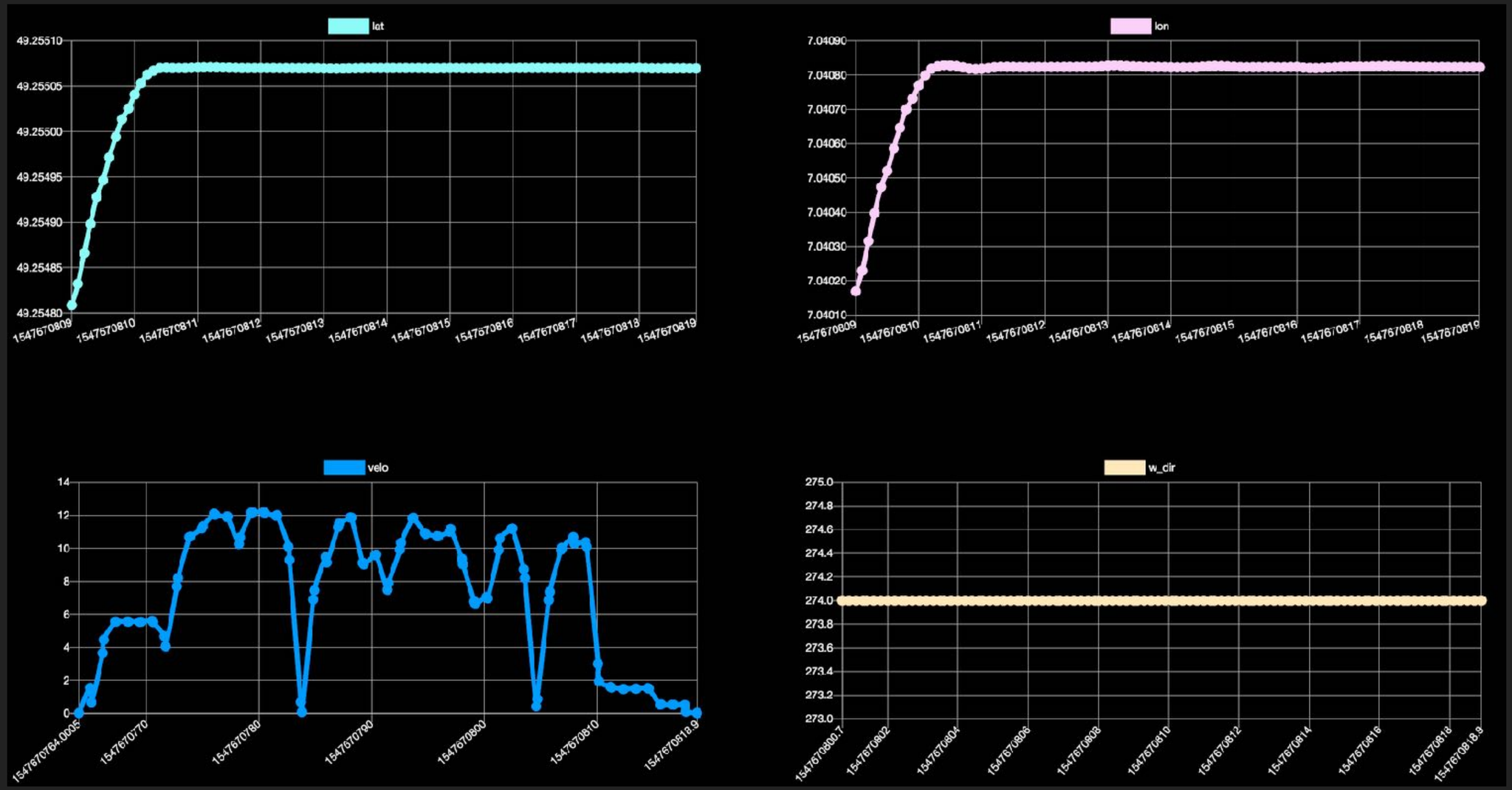
433,000 events

1,545ns per event @ 146%

Stack size < 1kB, no heap

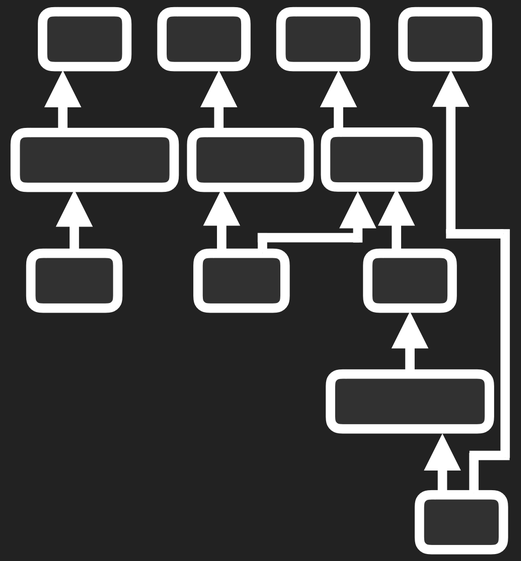
- Faymonville, Finkbeiner, Schledjewski, Schwenger, Stenger, Tentrup, Torfah, "*StreamLAB: Stream-based Monitoring of Cyber-physical Systems*", CAV 2019

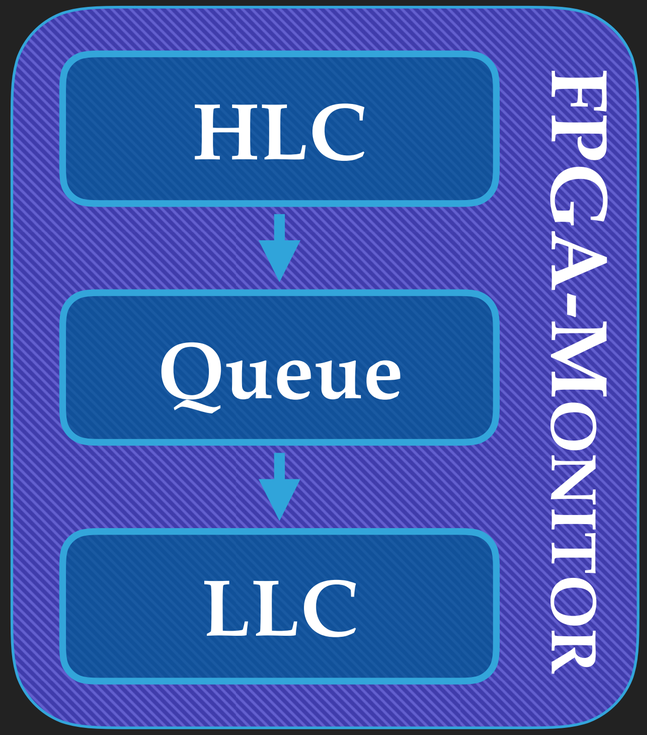
Rust Interpreter



Thanks to Sanny Schmitt for designing the interface!

StreamLAB



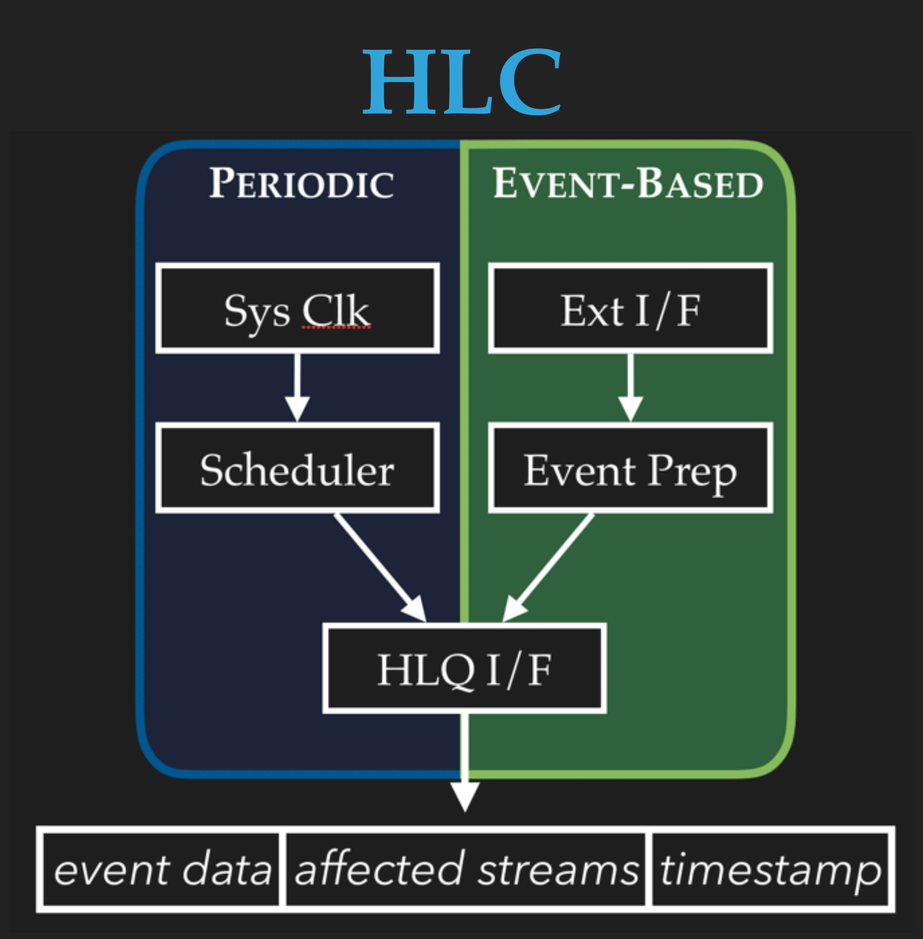


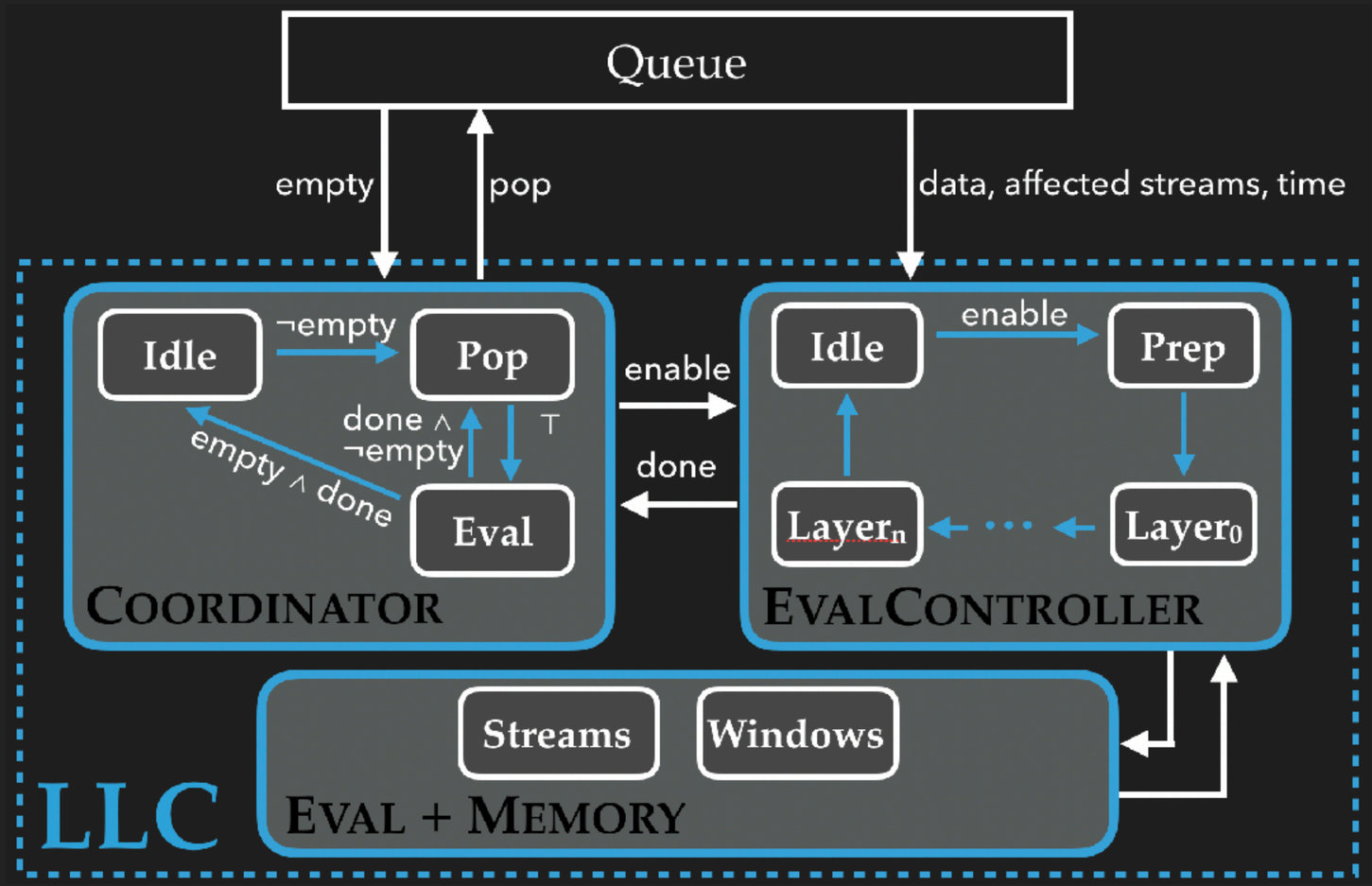
RTLola Specification

Annotated DG Intermediate Rep.

Hardware Compilation

VHDL/FPGA Compilation



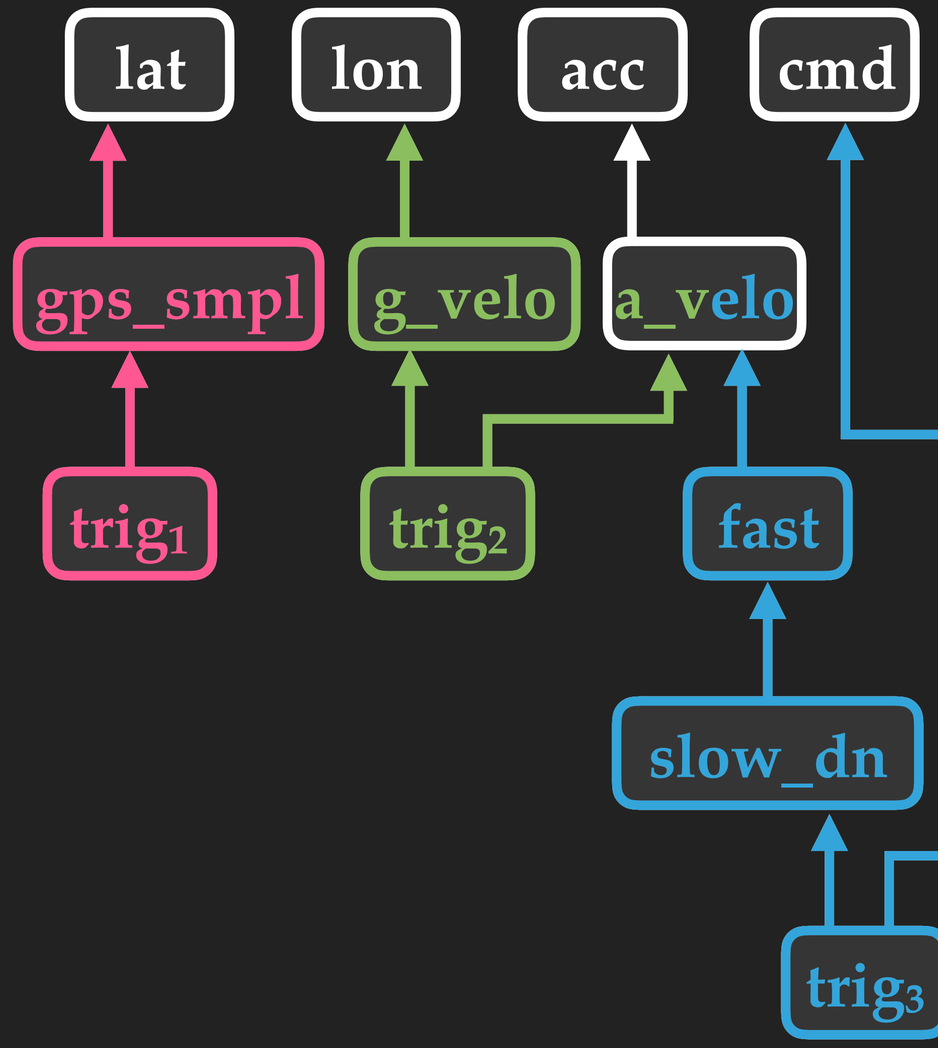


- Baumeister, Finkbeiner, Schwenger, Torfah, "*FPGA Stream-Monitoring of Real-Time Properties*", EMSOFT 2019
- Baumeister, Finkbeiner, Schwenger, Torfah, "*On the Similarities of Aircraft and Humans*", CyberCardia@ESWeek2019

Specification

```
input lat, lon: Float64 // from GPS
input accel_x: Float64  // from accelerometer
input slow_down_cmd: Bool

output gps_samples @1Hz := lat.aggregate(over_exactly: 1s, using: count)
trigger gps_samples < 5 “GPS frequency less than 5 Hz.”

output accel_velo @1Hz := accel_x.aggregate(over: 5s, using: ∫)
output gps_velo @1Hz := lon.aggregate(over: 5s, using: ∇)
trigger abs(accel_velo - gps_velo) > 0.1
    “Conflicting measurements for velocity.”

output fast := accel_velo > 700
output slow_down := fast.offset(by: -1).defaults(to: false) ∧ ¬fast
trigger @1Hz ¬slow_down_cmd.aggregate(over: 5s, using: ∃)
    ∧ slow_down.hold().defaults(to: false) “Spurious Slow-Down.”
```

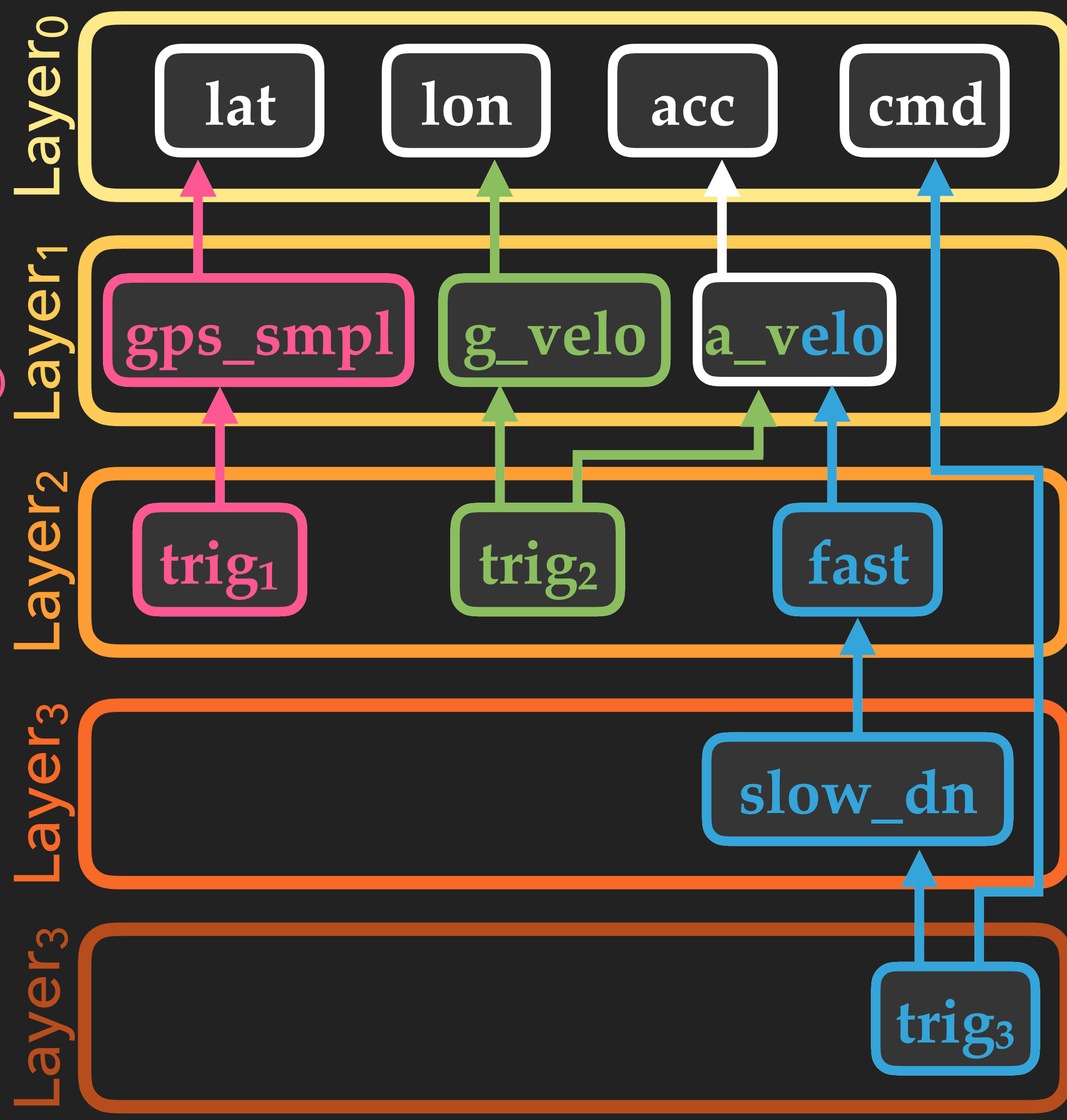
Dependency Graph



Specification

```
input lat, lon: Float64 // from GPS
input accel_x: Float64  // from accelerometer
input slow_down_cmd: Bool

output gps_samples @1Hz := lat.aggregate(over_exactly: 1s, using: count)
trigger gps_samples < 5 “GPS frequency less than 5 Hz.”

output accel_velo @1Hz := accel_x.aggregate(over: 5s, using: ∫)
output gps_velo @1Hz := lon.aggregate(over: 5s, using: ∇)
trigger abs(accel_velo - gps_velo) > 0.1
    “Conflicting measurements for velocity.”

output fast := accel_velo > 700
output slow_down := fast.offset(by: -1).defaults(to: false) ∧ ¬fast
trigger @1Hz ¬slow_down_cmd.aggregate(over: 5s, using: ∃)
    ∧ slow_down.hold().defaults(to: false) “Spurious Slow-Down.”
```

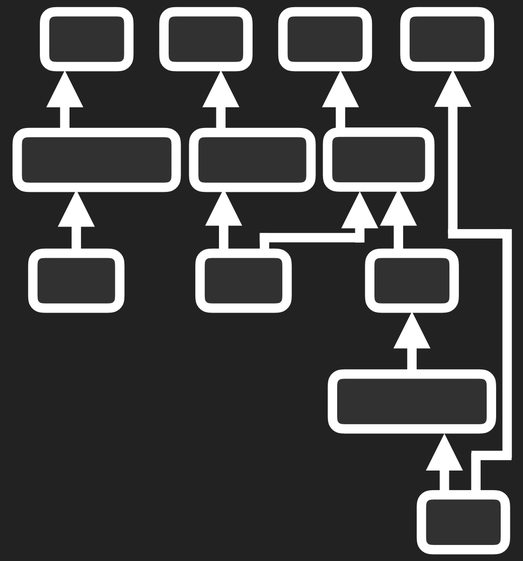
Dependency Graph



Evaluation

		FF	LUT	MUX	CA	MULT	Pwr [W]	Time [µs]
Drone	Mon	3036	3685	26	656	10	1.620	4.28
	HLC	901	156	0	22	0		
	Q	543	442	0	43	0		
	LLC	1281	2820	0	576	10		
Network	Mon	1905	1533	23	226	23	1.570	3.20
	HLC	550	161	0	37	0		
	Q	330	342	0	28	0		
	LLC	895	927	0	161	0		
Cmd-Resp Parallel	Mon	6379	13794	0	849	0	1.582	3.77
	HLC	936	232	0	30	0		
	Q	540	326	0	28	0		
	LLC	4903	13236	0	971	0		
Cmd-Resp Sequential	Mon	6909	14768	0	851	0	1.581	43.83
	HLC	936	232	0	30	0		
	Q	534	326	0	28	0		
	LLC	5433	14210	0	973	0		

StreamLAB



```
invariant gm_s3[i-2] == r.s3_mem[0]
invariant |trigger1_ghost| == i-4
invariant |trigger2_ghost| == i-4
invariant forall j:Int :: {t3[j],t2[j]
invariant forall j:Int :: {t3[j],t2[j]
invariant forall j:Int :: {t3[j],t2[j]

var s1_i: Int := t1[i] + r.s2_mem[0]
var s2_i: Int := t2[i] + r.s3_mem[0]
var s3_i: Int := t3[i] + r.s1_mem[0]

var trigger1: Bool := s3_i -s2_i < 0
var trigger2: Bool := s1_i - t1[i] ==

assert trigger1 <==> (t3[i] + r.s1_mem
3  silicon  ✓ Successfully verified negative-cycle.vpr in
```

Developed by Stefan Oswald,
co-advised by Noemi Passing

RTLola Specification

Annotated DG Intermediate Rep.

Viper Compilation

Our Vision

System

||

+

Formal Guarantees on Runtime Behavior

¯_(ツ)_/¯
It's ML

Trust me,
I'm an engineer

Outlook