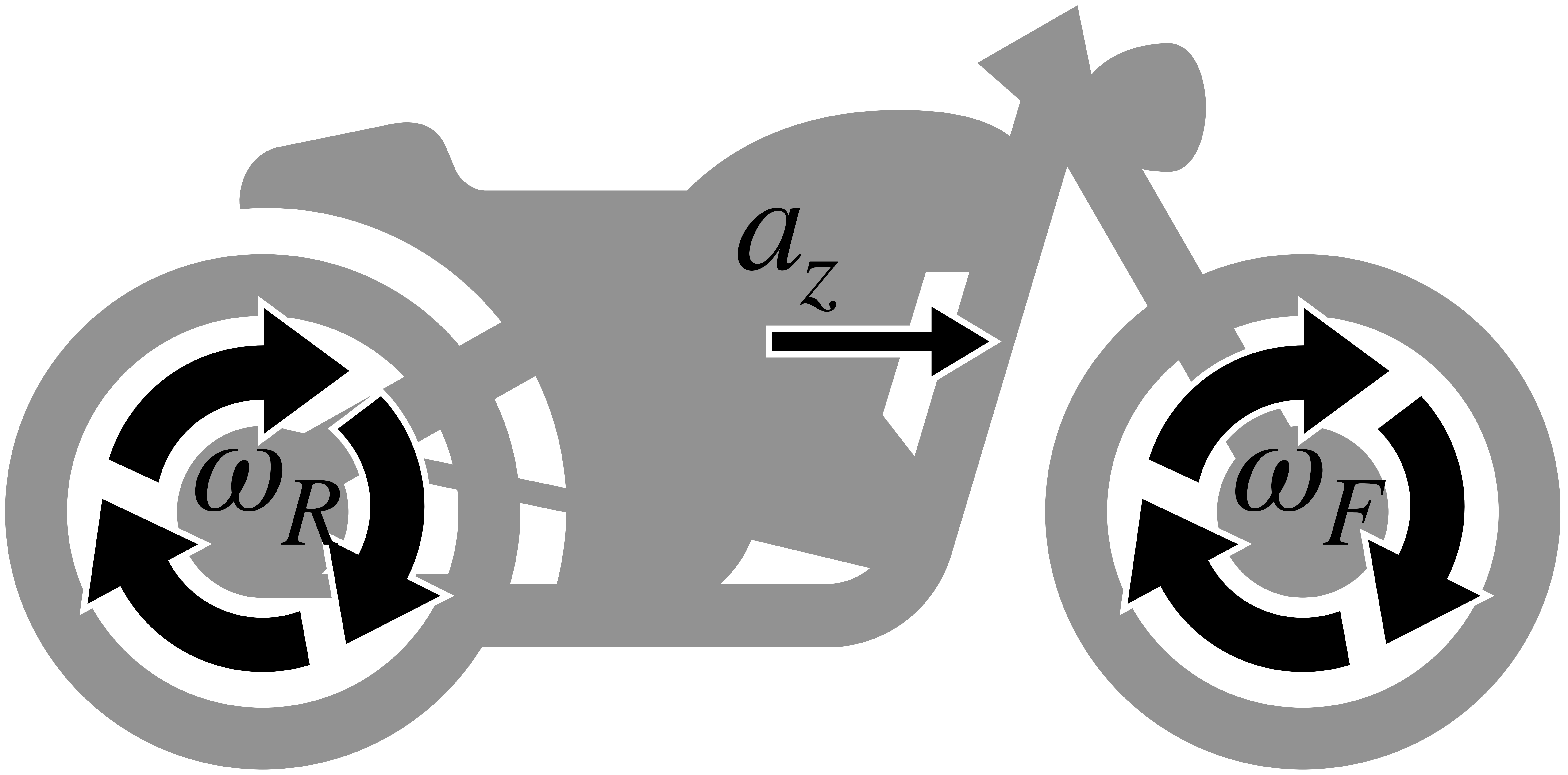
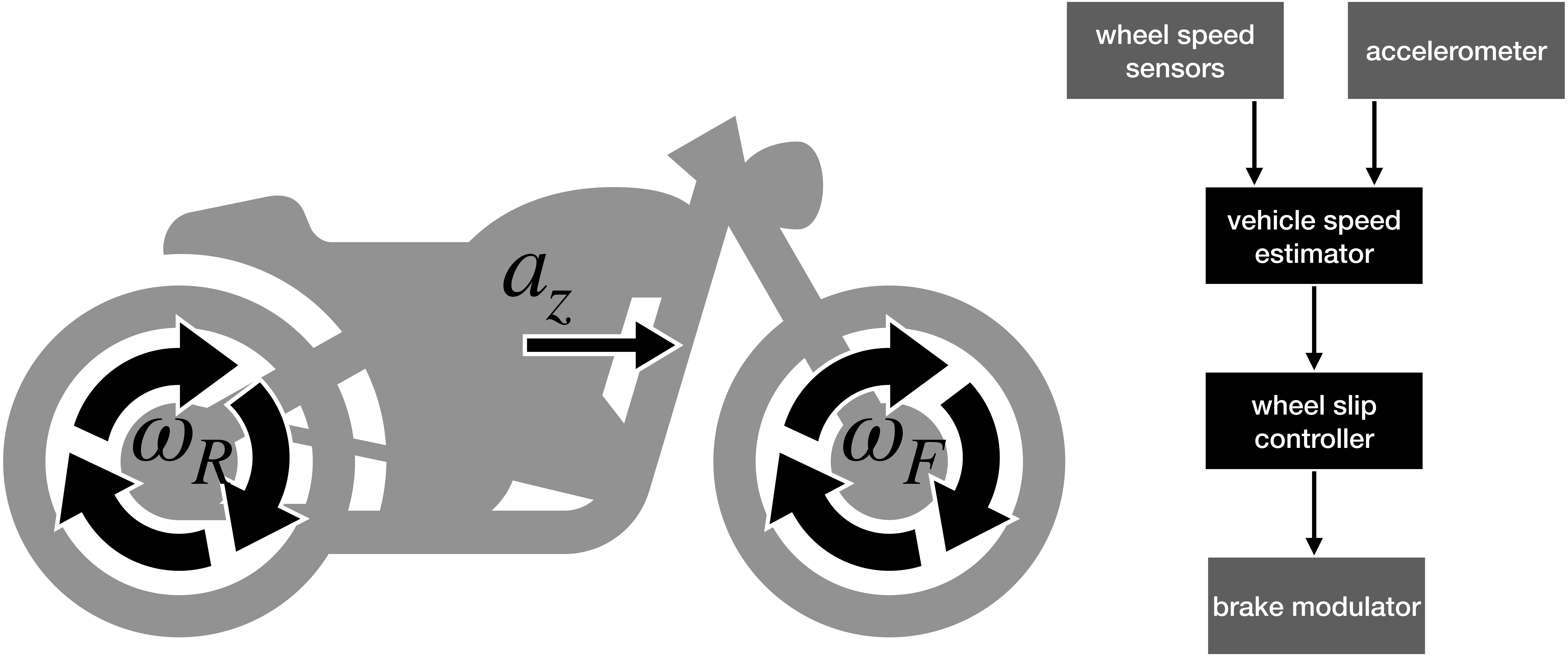
Pipit: reactive systems in F*

Amos Robinson, Australian National University

Anti-lock brakes for a motorcycle



Anti-lock brakes for a motorcycle



Vehicle speed estimator

let veh_speed_estimator $\omega_F\ \omega_R\ a_z\ \lfloor \hat{v} \rfloor\ \lceil \hat{v} \rceil$ =

…called every 10ms…

let $\lfloor \hat{v}' \rfloor$ = …updated lower bound… **in**

let $\lceil \hat{v}' \rceil$ = …updated upper bound… **in**

$(\lfloor \hat{v}' \rfloor, \lceil \hat{v}' \rceil)$

Vehicle speed estimator

let veh_speed_estimator $\omega_F\ \omega_R\ a_z\ \lfloor \hat{v} \rfloor\ \lceil \hat{v} \rceil$ =

let $v_F = \omega_F \cdot$ radius **in**

let $v_R = \omega_R \cdot$ radius **in**

let $\lfloor \hat{v}' \rfloor$ = **if** $v_F \approx_\epsilon v_R$ **then** $\min v_F\ v_R$ **else** $\lfloor \hat{v} \rfloor + a_z - \epsilon$ **in**

let $\lceil \hat{v}' \rceil$ = **if** $v_F \approx_\epsilon v_R$ **then** $\max v_F\ v_R$ **else** $\lceil \hat{v} \rceil + a_z + \epsilon$ **in**

$(\lfloor \hat{v}' \rfloor, \lceil \hat{v}' \rceil)$

What properties do we want our estimator to have?

- if the wheels agree, the estimate is pretty good

$$v_F \approx_\epsilon v_R \implies \lfloor \hat{v}' \rfloor \approx_\epsilon \lceil \hat{v}' \rceil$$

What properties do we want our estimator to have?

- if the wheels agree, the estimate is pretty good

  $$v_F \approx_\epsilon v_R \implies \lfloor \hat{v}' \rfloor \approx_\epsilon \lceil \hat{v}' \rceil$$

  easy proof:

  $\lfloor \hat{v}' \rfloor$ = **if** $v_F \approx_\epsilon v_R$ **then** $\min v_F \ v_R$ **else** …

  $\lceil \hat{v}' \rceil$ = **if** $v_F \approx_\epsilon v_R$ **then** $\max v_F \ v_R$ **else** …

What properties do we want our estimator to have?

- if the wheels agreed within time t, the estimate is not *too* bad

$$\blacklozenge_t \left(v_F \approx_\epsilon v_R\right) \implies \lfloor \hat{v}' \rfloor \approx_{t\epsilon} \lceil \hat{v}' \rceil$$

What properties do we want our estimator to have?

- if the wheels agreed within time t, the estimate is not *too* bad

$$\blacklozenge_t\,(v_F \approx_\epsilon v_R) \implies \lfloor \hat{v}' \rfloor \approx_{t\epsilon} \lceil \hat{v}' \rceil$$

how do we even state this? not trivial!

val veh_speed_estimator (ω_F ω_R: wheel) (a_z: accel) ($\lfloor \hat{v} \rfloor$ $\lceil \hat{v} \rceil$: vel)
: (vel & vel)

As a reactive system

let node veh_speed_estimator $\omega_F\ \omega_R\ a_z$=

let $v_F = \omega_F \cdot$ radius **in**

let $v_R = \omega_R \cdot$ radius **in**

let rec $\lfloor \hat{v} \rfloor$ = **if** $v_F \approx_\epsilon v_R$

then $\min v_F\ v_R$

else $(\min v_F\ v_R \rightarrow$ **pre** $\lfloor \hat{v} \rfloor) + a_z - \epsilon$ **in**

let rec $\lceil \hat{v} \rceil$ = **if** $v_F \approx_\epsilon v_R$

then $\max v_F\ v_R$

else $(\max v_F\ v_R \rightarrow$ **pre** $\lceil \hat{v} \rceil) + a_z + \epsilon$ **in**

$(\lfloor \hat{v} \rfloor, \lceil \hat{v} \rceil)$

As a reactive system

let node veh_speed_estimator $\omega_F\ \omega_R\ a_z$=

let $v_F = \omega_F \cdot$ radius **in**

let $v_R = \omega_R \cdot$ radius **in**

let rec $\lfloor \hat{v} \rfloor$ = **if** $v_F \approx_\epsilon v_R$

then $\min v_F\ v_R$

else $(\min v_F\ v_R \rightarrow$ **pre** $\lfloor \hat{v} \rfloor) + a_z - \epsilon$ **in**

let rec $\lceil \hat{v} \rceil$ = **if** $v_F \approx_\epsilon v_R$

then $\max v_F\ v_R$

else $(\max v_F\ v_R \rightarrow$ **pre** $\lceil \hat{v} \rceil) + a_z + \epsilon$ **in**

check $(\blacklozenge_t (v_F \approx_\epsilon v_R) \implies \lfloor \hat{v} \rfloor \approx_{t\epsilon} \lceil \hat{v} \rceil)$;

$(\lfloor \hat{v} \rfloor, \lceil \hat{v} \rceil)$

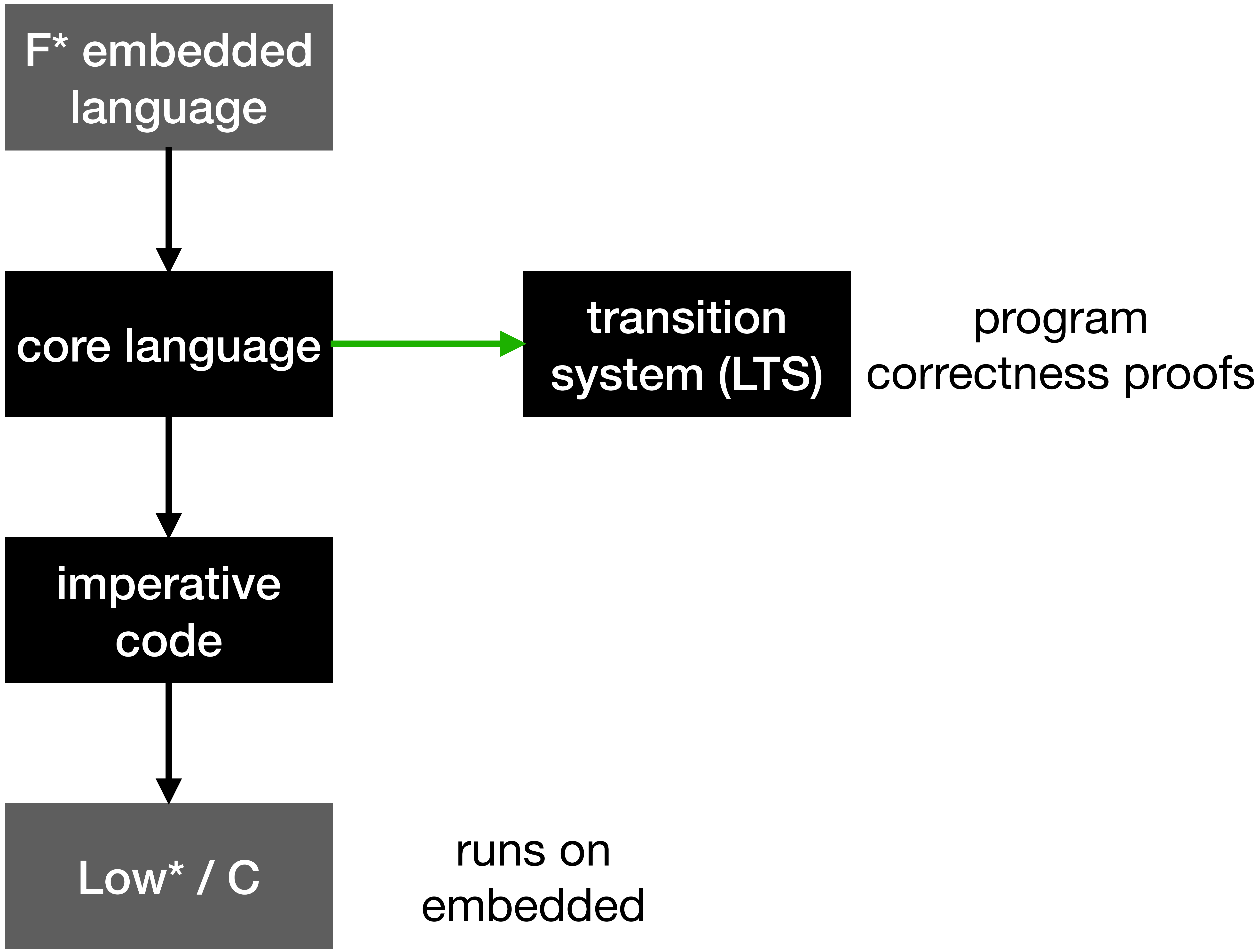
Past-time modal logic

let node $\blacklozenge_t(p) =$

if $t = 0$ **then** false **else** $\blacklozenge_{t-1}$(false $\rightarrow$ **pre** p)

Another pipit

Pipit structure



Core language

$e :=$

$| \; v \; | \; x \; | \; e \; e'$

$| \; \text{pre} \; e$

$| \; e \to e'$

$| \; \mu x. \; e[x]$

$| \; \text{let} \; x = e \; \text{in} \; e'[x]$

$| \; \text{check} \; p \; e \; \text{in} \; e'$

Core language

$e :=$

$\mid v \mid x \mid e\; e'$

$\mid$ pre e

$\mid e \to e'$

$\mid \mu x.\; e[x]$

$\mid$ let $x = e$ in $e'[x]$

$\mid$ check $p\; e$ in e'

$$\frac{}{\Sigma \vdash v \Downarrow v}\;\text{(Value)}$$

$$\frac{}{\Sigma_\perp ; \sigma \vdash x \Downarrow \sigma(x)}\;\text{(Var)}$$

Core language

$e :=$

$| \; v \; | \; x \; | \; e \; e'$

$| \; \text{pre} \; e$

$| \; e \to e'$

$| \; \mu x. \; e[x]$

$| \; \text{let} \; x = e \; \text{in} \; e'[x]$

$| \; \text{check} \; p \; e \; \text{in} \; e'$

$$\frac{\Sigma \vdash e \Downarrow v}{\Sigma; \sigma \vdash \text{pre} \; e \Downarrow v} \; \text{(Pre)}$$

Core language

$e :=$

$\mid v \mid x \mid e\ e'$

$\mid$ pre e

$\mid e \rightarrow e'$

$\mid \mu x.\ e[x]$

$\mid$ let $x = e$ in $e'[x]$

$\mid$ check $p\ e$ in e'

$$\frac{\sigma \vdash e \Downarrow v}{\sigma \vdash e \rightarrow e' \Downarrow v}\ (\rightarrow_1)$$

$$\frac{\Sigma; \sigma \vdash e' \Downarrow v'}{\Sigma; \sigma \vdash e \rightarrow e' \Downarrow v'}\ (\rightarrow_S)$$

Core language

e :=

$| \, v \, | \, x \, | \, e \; e'$

$|$ pre e

$| \, e \to e'$

$| \, \mu x. \; e[x]$

$|$ let $x = e$ in $e'[x]$

$|$ check $p \; e$ in e'

$$\frac{\Sigma \vdash e[x := \mu x. \, e] \Downarrow v}{\Sigma \vdash \mu x. \, e \Downarrow v} \, (\mu)$$

Core language

$e :=$

$\mid v \mid x \mid e\ e'$

$\mid \text{pre}\ e$

$\mid e \rightarrow e'$

$\mid \mu x.\ e[x]$

$\mid \text{let}\ x = e\ \text{in}\ e'[x]$

$\mid \text{check}\ p\ e\ \text{in}\ e'$

$$\frac{\Sigma \vdash e'[x := e] \Downarrow v}{\Sigma \vdash \text{let}\ x = e\ \text{in}\ e' \Downarrow v}\ \text{(Let)}$$

Core language

$e :=$

$\mid v \mid x \mid e\ e'$

$\mid \text{pre}\ e$

$\mid e \rightarrow e'$

$\mid \mu x.\ e[x]$

$\mid \text{let}\ x = e\ \text{in}\ e'[x]$

$\mid \text{check}\ p\ e\ \text{in}\ e'$

$$\frac{\Sigma \vdash e \Downarrow \top \qquad \Sigma \vdash e' \Downarrow v'}{\Sigma \vdash \text{check}\ p\ e\ \text{in}\ e' \Downarrow v'}\ \text{(Check)}$$

Metatheory: evaluation

$$\frac{\Sigma \text{ non-empty} \qquad e \text{ causal}}{\exists v \,.\, \Sigma \vdash e \Downarrow v}$$

Metatheory: causality

$\mu x.\; x + 1$ not causal

$\mu x.\, (0 \to \text{pre}\; (x + 1))$ causal

Metatheory: substitution

$$\frac{\Sigma \vdash e \Downarrow^{*} vs \quad \Sigma\#[x := vs] \vdash e' \Downarrow v}{\Sigma \vdash \textsf{let}\ x = e\ \textsf{in}\ e' \Downarrow v}$$

Metatheory: substitution

$$\frac{\Sigma \vdash e \Downarrow^* vs \quad \Sigma\#[x := vs]; \sigma\#[x := v_\perp] \vdash e \Downarrow v}{\Sigma; \sigma \vdash \mu x.\, e \Downarrow v}$$

Flying pipit

Future work

- verification:
  - proof of imperative codegen
- case studies:
  - anti-lock braking?
- improvements:
  - common subexpression elimination
- language features:
  - clocks, letrecs and contracts

Last pipit