A COMPONENT-BASED APPROACH TO HYBRID SYSTEMS SAFETY VERIFICATION

Andreas Müller – andreas.mueller@jku.at
Werner Retschitzegger – werner.retschitzegger@jku.at
Wieland Schwinger – wieland.schwinger@jku.at

Johannes Kepler University, Linz
Department of Cooperative Information Systems
http://cis.jku.at/

Stefan Mitsch – smitsch@cs.cmu.edu
André Platzer - aplatzer@cs.cmu.edu

Carnegie Mellon University, Pittsburgh
Computer Science Department
http://www.ls.cs.cmu.edu

OVERVIEW

- Background
  - Cyber-Physical System
  - Hybrid System Models
  - Component-based Modeling

- Component-based Modeling and Verification Approach
  - Components
  - Interfaces
  - Contracts
  - Composition Retains Contract

- Conclusion and Future Work

OVERVIEW

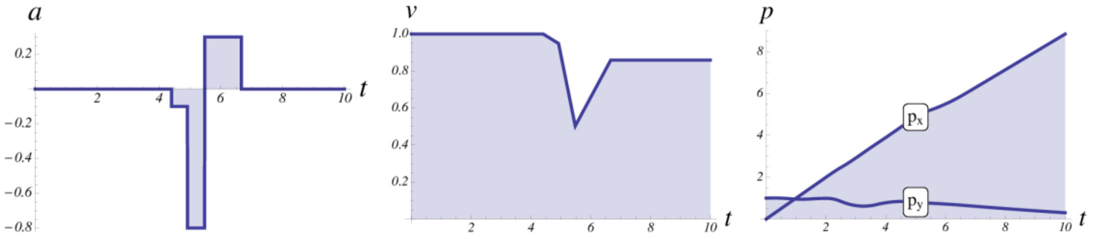
- Background
  - ☐ Cyber-Physical System
  - ☐ Hybrid System Models
  - ☐ Component-based Modeling

- Component-based Modeling and Verification Approach
  - ☐ Components
  - ☐ Interfaces
  - ☐ Contracts
  - ☐ Composition Retains Contract

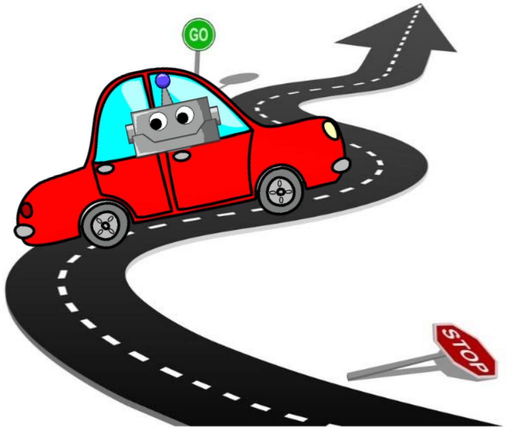
- Conclusion and Future Work

BACKGROUND

- Cyber-physical systems (CPS)
  - **Cyber** and **physical** capabilities
  - Continuous physical-part: vehicle movement,…
  - Discrete cyber-part: vehicle steering,…
  - Often safety-critical!



- **Hybrid system models** – Model and analyze CPS
  - Hybrid programs: program notation for hybrid system modeling
  - Safety Analysis:
    - $\Phi \rightarrow [\alpha]\Psi$ …starting in Φ, each run of α leads to a safe state Ψ
    - Verified using Theorem Prover – KeYmaera
  - Challenging for large monolithic models



- **Component-based** hybrid system modeling and **verification**
  - Component verification results do **not always transfer** to composite

→ **Component-based approach to hybrid system safety verification**

OVERVIEW

- Background
  - Cyber-Physical System
  - Hybrid System Models
  - Component-based Modeling

- <u>Component-based Modeling and Verification Approach</u>
  - Components
  - Interfaces
  - Contracts
  - Composition Retains Contract

- Conclusion and Future Work

RUNNING EXAMPLE - VEHICLE CRUISE CONTROL

- Vehicle Cruise Control System
  - Overall Safety Property: Keep vehicle’s velocity within bounds
  - Split into two components

- Actuator Component
  - Receives target velocity
  - Chooses target acceleration, such that target velocity can be reached
  - Outputs actual velocity
- Cruise Controller Component
  - Receives actual velocity
  - Chooses target velocity
  - Outputs target velocity

DEFINITION 2: COMPONENT

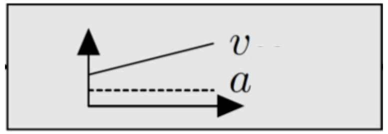
- Component $C = (ctrl, plant)$

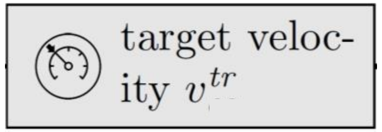
- $ctrl$
  - Discrete control part
  - NO continuous parts

- $plant$
  - Continuous part
  - $\{x_1' = \theta_1, \dots, x_n' = \theta_n \;\&\; H\}$
  - Ordinary differential equations
  - Evolution domain H

- Actuator: $C_{ac} = (ctrl_{ac}, plant_{ac})$
  - $ctrl_{ac} \equiv$ choose a, such that v^{tr} is reached until ϵ
  - $plant_{ac} \equiv$ evolve v with rate a for at most ϵ

- Cruise Control Component
  - Choose target velocity





DEFINITION 3: INTERFACE

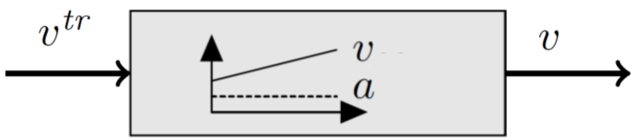
- Interface $I = (V^{in}, \pi^{in}, V^{out}, \pi^{out})$

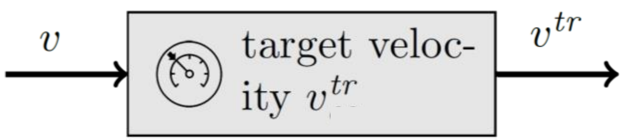
- V^{in}…variables for input ports
- π^{in}…input assumptions

- V^{out}…variables for output ports
- π^{out}…output guarantees

- Actuator: I_{ac}
  - $V^{in} = \{v^{tr}\}$…target velocity
  - $\pi^{in}(v^{tr}) \equiv$ target velocity v^{tr} in velocity interval
  - $V^{out} = \{v\}$…current velocity
  - $\pi^{out}(v) \equiv$ current velocity v in velocity interval

- Cruise Control Component
  - Reads current velocity
  - Provides calculated target velocity





JKU

DEFINITION 4: CONTRACT

- Contract
  - ☐ Initial state ϕ
  - ☐ Target state ψ
- $\mathrm{Cont}(C, I) \equiv$

$$\underbrace{t = 0 \land \phi}_{\text{valid initial state}} \rightarrow [\underbrace{(\underbrace{in}_{\text{read inputs}};\underbrace{ctrl}_{\text{run ctrl}};\underbrace{\{t' = 1, plant\}}_{\text{run plant}})*}_{\text{repeat 0…n times}}]\underbrace{\psi}_{\text{must hold after all runs}}$$

- $\psi \equiv \psi^{safe} \land \Pi^{out}$

- Actuator: (1)
  - ☐ $\phi \equiv$ Vehicle initially stopped and …
  - ☐ $\psi \equiv$ vehicle velocity always in interval
- Cruise Controller Component:
  - ☐ Target velocity always in interval
- Verified using KeYmaera

(1) Properties coincide due to simple example. Not necessarily the case!

THEOREM 1: COMPOSITION RETAINS CONTRACTS

- Let…
  - (C_1, I_1) and (C_2, I_2) be Components with Interfaces
  - $Cont(C_1, I_1)$ and $Cont(C_2, I_2)$ verified
  - Compatible (Def. 6)
  - $(C_3, I_3) = (C_1, I_1) \,||\, (C_2, I_2)$ (Def. 5)
- Then $Cont(C_3, I_3)$ is also valid, with…
  - $\phi_3 \equiv \phi_1 \wedge \phi_2$
    both initial states hold
  - $\psi_3 \equiv \psi_1 \wedge \psi_2$
    both safety properties and all output properties hold

- Two Components
  - Actuator and Cruise Controller
- Actuator Contract verified
  - $\psi_{ac} \equiv$ vehicle velocity always in interval
- Cruise Controller Contract verified
  - $\psi_{cc} \equiv$ target velocity always in interval
- Compatible Composite

- $(C_{sys}, I_{sys}) = (C_{ac}, I_{ac}) \,||\, (C_{cc}, I_{cc})$
  - $\phi_{sys} \equiv \phi_{ac} \wedge \phi_{cc}$
  - $\psi_{sys} \equiv \psi_{ac} \wedge \psi_{cc}$
  - ➔ vehicle velocity always in interval

Overall System Property!

OVERVIEW

- Background
  - ☐ Cyber-Physical System
  - ☐ Hybrid System Models
  - ☐ Component-based Modeling

- Component-based Modeling and Verification Approach
  - ☐ Components
  - ☐ Interfaces
  - ☐ Contracts
  - ☐ Composition Retains Contract

- <u>Conclusion and Future Work</u>

CONCLUSION AND FUTURE WORK

- We presented a technique to model and verify component-based CPS
  - Split system into components
  - Verify Components
  - Rebuild system from components
  - → Transfer Verification Results!

- Future Work
  - Extend interface and port capabilities
  - Implement framework as tool
  - Add further composition operations
    - Delayed transmission
    - Erroneous transmission

A COMPONENT-BASED APPROACH TO HYBRID SYSTEMS SAFETY VERIFICATION

Andreas Müller – andreas.mueller@jku.at
Werner Retschitzegger – werner.retschitzegger@jku.at
Wieland Schwinger – wieland.schwinger@jku.at

Johannes Kepler University, Linz
Department of Cooperative Information Systems
http://cis.jku.at/

Stefan Mitsch – smitsch@cs.cmu.edu
André Platzer - aplatzer@cs.cmu.edu

Carnegie Mellon University, Pittsburgh
Computer Science Department
http://www.ls.cs.cmu.edu