Technische
Universität
Braunschweig

Institut für
Regelungstechnik

Skill-Based Verification of Cyber-Physical Systems

Alexander Knüppel, Inga Jatzkowski, Marcus Nolte, Thomas Thüm, Tobias Runge, and Ina Schaefer

FASE'2021

(accepted at FASE'20)

Motivation: Cyber-Physical Systems

<https://waymo.com/journey/>

*„I really consider autonomous driving **a solved problem**. [...] I think we are probably **less than two years** away [...]"*

Elon Musk at Code 2016

Fundamental Research Questions Remain

Why formal?

Ensuring functional safety/correctness is indispensable!

SE

But...

- Scalability
- Expertise
- ...

Q: **How to integrate** functional safety & **provably correct** function development for safe behavior into software engineering processes?

Maneuver-Centric Formal Engineering Approach

Contributions of this work:

- Modelling notation for maneuvers based on *Skill Graphs*
- Formal underpinning for correctness proofs based on *Hybrid Programs*
- Tool support (*SKEDITOR*)

Modelling Foundation: Skill Graphs

- Graph-based representation for **functional dependencies** (nodes: skills, edges: dependencies) (A. Reschka et al., 2015)
- “**What activities** must the system perform to fulfill its goals?”
- Skill Graph:**
 - Used during **system design**
 - Mapping between **safety goals** & skills

A. Reschka, G. Bagschik, S. Ulbrich, M. Nolte and M. Maurer:

„Ability and skill graphs for system modeling, online monitoring, and decision support for vehicle guidance systems“

In: 2015 IEEE Intelligent Vehicles Symposium (IV 2015), Seoul, South Korea, 2015, pp. 933-939

Skill Description

Symbol	Description	Content
X_{def}	Defined variables or parameters	$\emptyset$
X_{req}	Required variables or parameters	$\{x, v, v_{\text{max}}\}$
α	Hybrid program	$\{\dots\}$
τ	Type	action
ψ	Safety obligation	$\{v \leq v_{\text{max}}(2.7\text{ms}^{-1})\}$
ϕ	Precondition	$\{v \leq v_{\text{max}} - ep * A\}$
	Contract-based design methodology	

Skill ID	Name		Requirements / Safety Obligation
...	...	...	...
S4	"Control longitudinal dynamics"	R1	"Vehicle speed must not exceed 2.7 m/s"
...	...	...	...

Hybrid Systems as a Mathematical Foundation for CPSs

Mixed modeling of continuous and discrete dynamics

Hybrid Programs and Differential Dynamic Logic

- **Differential Dynamic Logic:**

$$\Phi ::= \Theta_1 \sim \Theta_2 \mid \neg \Phi \mid \Phi \wedge \Psi \mid \Phi \vee \Psi \mid \Phi \rightarrow \Psi \mid \forall x \Phi \mid \exists x \Phi \mid [\alpha] \Phi$$

$$\sim \in \{<, \leq, >, \geq, =, \neq\}$$

- **Hybrid Program:**

$$\alpha ::= \alpha; \beta \mid \alpha \cup \beta \mid \alpha^* \mid x := \Theta \mid x := * \mid x' = \Theta \& H \mid ?H$$

Seq.
Compose

Nondet.
Choice

Nondet.
Repeat

Discrete
Assignment

Diff.
Equation

Test
Condition

Example: Hybrid Program of *Control Longitudinal Dynamics*

$$\begin{array}{l}
 X_{\text{def}} = \{x, v, v_{\max}\} \\
 X_{\text{req}} = \{a, A, B, ep, t\} \\
 \alpha ::= \left\{ \begin{array}{l}
 \text{init} \rightarrow [(ctrl; dyn)*](\text{guar}) \\
 \text{init} \equiv v \geq 0 \wedge v \leq v_{\max} \wedge A > 0 \wedge A \leq 4 \wedge B \geq 5 \wedge v_{\max} = 2.7 \\
 ctrl \equiv (?v_{\max} - v \leq \text{margin}); a = *; -B \leq a \leq 0; \\
 \quad \cup ?v_{\max} - v \geq \text{margin}; a = *; -B \leq a \leq A;) \\
 \text{margin} \equiv ep * A \\
 dyn \equiv t := 0; x' = v, v' = a, t' = 1 \& v \geq 0 \wedge t \leq ep \\
 \text{guar} \equiv v \leq v_{\max}
 \end{array} \right. \\
 \tau = \text{action} \\
 \Phi = \{v_s \leq v_{\max} \ (2.7 \text{ m s}^{-1})\}
 \end{array}$$

Verification Condition Generation

Theorem: $\models \phi_{\text{global}} \rightarrow [\rho]\psi_{\text{global}}$

where $\rho = \alpha_{\text{oeb}}^1; \dots; \alpha_{\text{oeb}}^n$

1. $\phi_{\text{global}} \rightarrow \text{assume}_s$ for all s
2. $\text{safe}_s \rightarrow \text{assume}_{s'}$ for all s and s'
iff s in { } and $(s', s) \in \text{Paths}$
3. $\text{safe}_s \rightarrow \psi_{\text{global}}$ for all s in { }
4. $\text{assume}_s \rightarrow [\alpha_{\text{oeb}}^i]\text{safe}_s$

depends on hybrid programs defined in - skills,
which can be integrated via method calling

- Main
- Observable
- External Behavior
- Action
- Planning
- Perception
- Sensor
- Actuator

Side note: dynamic systems are modeled in action-skills and propagated

Controller Assembly

Requires **program independence** at the moment, i.e.,

$$\text{Var}_{\text{mod}}(s_i) \cap \text{Var}_{\text{mod}}(s_j) = \emptyset$$

then $\alpha_{\text{root}} = \alpha_{s_1}; \dots; \alpha_{s_n}$

Follow hard shoulder may use controller implementations of child skills

➡ Modularity + reusability + saves verification effort

On Compositionality of Skill Graphs

Theorem 1 (Composition of Skill Graphs Retains Validity). *Let G_1 and G_2 be two compatible skill graphs and $G = G_1 \circ G_2$ their composition. Then, G is valid if G_1 and G_2 are valid (i.e., $\models G$ if $\models G_1$ and $\models G_2$).*

On Compositionality of Skill Graphs

Tool Support: Skeditor

<https://github.com/TUBS-ISF/Skeditor>

The screenshot displays the Skeditor IDE interface. On the left, the Package Explorer shows a project structure with folders like 'ExploreWorld', 'FollowMode', 'Obstacle Avoidance', 'Robot', 'src', 'graphs', 'ObstacleAvoidance.sked', 'SafeHalt.sked', 'Strong Compositionality', and 'WaterTank'. Below it, the Parameter List table is visible:

Name	Abbreviation	Unit	Value [range]	Type
Braking Force Range	B	m/s ²		Const.
Acceleration Range	A	m/s ²	[0:4]	Const.
Control cycle	ep	ms	(0:)	Const.
Position	x	m		Var.
Max Steering Rate	W	rad	0.5	Const.
Clock	t	s		Var.
Yaw rate	Psi2	m/s ²	0	Var.
Acceleration	a	m/s ²		Var.
Deceleration	b	m/s ²	(0:)	Const.
Speed	v	m/s		Var.
dx	dx	m/s ²		Var.
dy	dy	m/s ²		Var.
r	r	m/s ²		Var.
steering velocity	w	m/s ²		Var.
y position	y	m/s ²		Var.
Maximum steering	W	m/s ²		Const.
Maximum velocity	vmax	m/s	2.7	Const.

The main workspace shows a hierarchical diagram of a skill-based architecture. The 'Main' node points to 'Safe Halt', which points to 'Control dynamics'. 'Control dynamics' has four outgoing edges to 'Select target object', 'Estimate motion', 'Decelerate', and 'Accelerate'. 'Select target object' points to 'Perceive objects', which then points to 'Radar' and 'Camera'. 'Estimate motion' points to 'Inertial sensors'. 'Decelerate' points to 'Brake system', and 'Accelerate' points to 'Powertrain'. A 'Skill Description' dialog box is open in the foreground, showing the following content:

```
2 /* Empty Skill Description */
3
4 Variables { //interface, what about functions?
5   requires {
6     Double v;
7     Double a;
8     Double x;
9   }
10  provides {
11    //empty
12  }
13 }
14
15 Parameters {
16   Double b; // Deceleration constant
17   Double vmax; // maximal velocity
18   UInt32 t; // timing
19 }
```

Tool Support: Skeditor

<https://github.com/TUBS-ISF/Skeditor>

The screenshot displays the Skeditor software interface. A prominent yellow warning box at the top states: "Warning: Limited functionality: Neither Mathematica nor Wolfram Engine is configured. [Configure now.](#)". Below this, an update notification indicates that version 4.9.3 is now available from KeYmaeraX.org. The main editing area shows a logical formula:
$$\vdash \text{init}(v) \wedge \text{consts} \rightarrow [\{ \text{ctrl}; \text{drive}; \}^*] \text{safe}(v)$$
 with a hint to "implyR | chaseAt |". The interface includes a Package Explorer on the left, a Parameter List at the bottom left, and a Palette on the right. A small dialog box at the bottom right contains the text "KeYmaera X 4.9.1 (version 4.9.3 is now available from KeYmaeraX.org). © Carnegie Mellon University, Logical Systems Lab 2014-2020" and "zum bekannten Fußballmagazin: Wie Holstein Kiel das Top...".

Name	Abbreviation
Braking Force Range	B
Acceleration Range	A
Control cycle	ep
Position	x
Max Steering Rate	W
Clock	t
Yaw rate	Psi2
Acceleration	a
Deceleration	b
Speed	v
dx	dx
dy	dy
r	r
steering velocity	w
y position	y
Maximum steering	W
Maximum velocity	vmax

Var.	Const.	Value
$\dot{v}$		
$\dot{a}$		
$\dot{b}$		
$\dot{w}$		
$\dot{y}$		
$\dot{W}$		
$\dot{vmax}$		

Safety obligation	Assumption (computed)	Guarantee (computed)
$v \leq v_{max}$	$((v \geq 0.0 \ \& \ v \leq v_{max}) \mid (v + (A * ep) \leq v_{max}))$	$v \geq 0.0 \ \& \ v \leq v_{max}$

```
14 Parameters {
15   Double b; // Deceleration constant
16   Double vmax; // maximal velocity
17   UInt32 t; // timing
18 }
```


Reducing Verification Effort through Composition – Eval.

RQ: To what extent can skill-based compositional verification reduce the verification effort?

Reducing Verification Effort through Composition - Results

***RQ:** To what extent can skill-based compositional verification reduce the verification effort?*

	Verified Skill				Proof steps					Σ_{reuse}
	s_1	s_6	s_{13}	s_{14}	s_1	s_6	s_{13}	s_{14}	Σ	
G_1	✓	✓			4,746	3,769			8,515	8,515
G_2		✓	✓	✓		3,769	16,924	7,223	27,916	24,147
$G_1 \circ G_2$	✓	✓	✓	✓	4,746	3,769	16,924	7,223	32,662	0*
Σ_{total}									69,093	32,662

*No re-verification of skills with Theorem 1

➡ Approximately 53%

Summary

<https://github.com/TUBS-ISF/Skeditor>

a.knueppel@tu-bs.de

+

```


$$\begin{aligned}
 X_{def} &= \{x, v, v_{max}\} \\
 X_{req} &= \{a, A, B, ep, t\} \\
 \text{init} &\rightarrow \{ctrl; dyn\} \{guar\} \\
 \text{init} &\equiv v \geq 0 \wedge v \leq v_{max} \wedge A > 0 \wedge A \leq 4 \wedge B \geq 5 \wedge v_{max} = 2.7 \\
 ctrl &\equiv (!v_{max} - v \leq \text{margin}); a = s; -B \leq a \leq 0; \\
 &\quad \vee (!v_{max} - v \geq \text{margin}); a = s; -B \leq a \leq A; \\
 \alpha ::= & \\
 \text{margin} &\equiv ep * A \\
 dyn &\equiv t := 0; a' = v, v' = a, t' = 1 \wedge v \geq 0 \wedge t \leq ep \\
 guar &\equiv v \leq v_{max} \\
 \tau &= \text{action} \\
 \phi &= \{v_s \leq v_{max} (2.7 \text{ m s}^{-1})\}
 \end{aligned}$$


```

Current/Future Work:

- Case Studies/modelled maneuvers
- Code generation + ROS/AirSim Simulation
- and more...