Tables 2.1 and 3.4 Operations of hybrid programs (HPs), differential-algebraic programs (DAPs), quantified hybrid programs (QHPs)

HP Notation	KeYmaera Notation	Operation	Effect
$x := \theta$	`x := ` θ	discrete assign	assigns term θ to variable x
$x := *$	`x := *`	nondet. assign	assigns any real value to x
$x_1' = \theta_1, \dots, x_n' = \theta_n \,\&\, H$	`{x1'=`θ_1`,...,xn'=`θ_n`,H}`	continuous evolve	differential equations for x_j in first-order constraint H (evolution domain)
$?H$	`?H`	state test / check	test first-order formula H at current state
$\alpha; \beta$	α`;` β	seq. compose	HP β starts after HP α finishes
$\alpha \cup \beta$	α `++` β	nondet. choice	choice between alternatives HP α or HP β
α^*	α`*`	nondet. repeat	repeats HP α n-times for any $n \in \mathbb{N}$
`if`$(H)\,\alpha$	`if(H) then` α `fi`	if-then	runs α if H holds, otherwise does nothing
`if`$(H)\,\alpha$ `else` β	`if(H) then` α `else` β `fi`	if-then-else	runs α if H holds, otherwise runs β
`while`$(H)\,\alpha$	`while(H)` α `end`	while loop	repeats α as long as H holds, stops before doing α if H false.
$x_1' \geq \theta_1, \dots, x_n' \leq \theta_n \,\&\, H$	`{x1'>=`θ_1`,...,xn'<=`θ_n`,H}`	continuous evolve	differential inequalities for x_j within H
$\exists d\, x_1' = \theta_1(d) \land ..x_n' = \theta_n(d) \,\&\, H$	`{\exists R d.(x1'=`θ_1`&..xn'=`θ_n`&H)}`	continuous evolve	differential-algebraic constraints
$\forall i{:}C\ x(i) := \theta(i)$	`\forall C i . x(i) := `θ`(i)`	quantified assign	assigns terms $\theta(i)$ to $x(i)$ for all i
$\forall i{:}C\ x(i) := *$	`\forall C i . x(i) := *`	quantified nondet.	assigns any value to $x(i)$ for all i
$\forall i{:}C\ (x(i)' = \theta(i) \,\&\, H(i))$	`\forall C i . {x(i)'=`θ`(i),H(i)}`	quantified evolve	quantified differential equations for $x(i)$ at rate $\theta(i)$ within domain $H(i)$ for all i

Tables 2.3 and 4.1 Operators of differential dynamic logic (d$\mathcal{L}$), quantified differential dynamic logic (Qd$\mathcal{L}$)

d$\mathcal{L}$ Notation	KeYmaera Notation	Operator	Meaning
$\theta_1 = \theta_2$	θ_1 `=` θ_2	equality comparison	true iff θ_1 equals θ_2
$\theta_1 \geq \theta_2$	θ_1 `>=` θ_2	inequality comparison	true iff $\theta_1 \geq \theta_2$, similarly for $>, \leq, <, \neq$
$\neg\phi$	`!`ϕ	negation / not	true if ϕ is false
$\phi \land \psi$	ϕ `&` ψ	conjunction / and	true if both ϕ and ψ are true
$\phi \lor \psi$	ϕ `|` ψ	disjunction / or	true if ϕ is true or if ψ is true
$\phi \rightarrow \psi$	ϕ `->` ψ	implication / implies	true if ϕ is false or ψ is true
$\phi \leftrightarrow \psi$	ϕ `<->` ψ	bi-implication / equivalent	true if ϕ and ψ are both true or both false
$\forall x\, \phi$	`\forall R x .` ϕ	universal quantifier / for all	true if ϕ is true for all real values of variable x
$\exists x\, \phi$	`\exists R x .` ϕ	existential quantifier / exists	true if ϕ is true for some real value of variable x
$[\alpha]\phi$	`\[`α`\]` ϕ	$[\cdot]$ modality / box	true if ϕ is true after all runs of HP α
$\langle\alpha\rangle\phi$	`\<`α`\>` ϕ	$\langle\cdot\rangle$ modality / diamond	true if ϕ is true after at least one run of HP α
$\forall x\, \phi$	`\forall C x .` ϕ	universal quantifier / for all (Qd$\mathcal{L}$)	true if ϕ is true for all values of type C for variable x
$\exists x\, \phi$	`\exists C x .` ϕ	existential quantifier / exists (Qd$\mathcal{L}$)	true if ϕ is true for some value of type C for variable x
$[\alpha]\phi$	`\[`α`\]` ϕ	$[\cdot]$ modality / box (Qd$\mathcal{L}$)	true if ϕ is true after all runs of QHP α
$\langle\alpha\rangle\phi$	`\<`α`\>` ϕ	$\langle\cdot\rangle$ modality / diamond (Qd$\mathcal{L}$)	true if ϕ is true after some run of QHP α

```
\sorts { /* For QdL:   declare two sorts C and D in addition to built-in sort R of reals */
    C;                /* cars */
    L;                /* lanes */
}
\functions { /* declare function symbols for parameters or functions */
    /* symbolic parameter declarations, cannot change their values at runtime */
    R b;
    R A;
    /* declare a function as \external if interpreted by the arithmetic solver */
    \external R Sqrt(R);
    /* For QdL: declare a function f with 2 parameters and assignable return-value of type C */
    \nonRigid[Location] C f(C,L);
}
\programVariables { /* state variable declarations */
    R x;              /* real-valued position */
    R v;              /* real-valued velocity */
    R a;              /* real-valued acceleration */
}
\problem { /* dL or QdL or dTL formula to prove */
    v >=0 & b > 0 & A >= 0  ->  \[ ( (a:=-b ++ a:=A); {x'=v, v'=a, v>=0} )* \] (v >= 0)
}
```

$$(\neg\text{r not right})\ \frac{\Gamma, \phi \vdash \Delta}{\Gamma \vdash \neg\phi, \Delta} \qquad (\vee\text{r or right})\ \frac{\Gamma \vdash \phi, \psi, \Delta}{\Gamma \vdash \phi \vee \psi, \Delta} \qquad (\wedge\text{r and right})\ \frac{\Gamma \vdash \phi, \Delta \quad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \phi \wedge \psi, \Delta} \qquad (\rightarrow\text{r imply right})\ \frac{\Gamma, \phi \vdash \psi, \Delta}{\Gamma \vdash \phi \rightarrow \psi, \Delta}$$

$$(\neg\text{l not left})\ \frac{\Gamma \vdash \phi, \Delta}{\Gamma, \neg\phi \vdash \Delta} \qquad (\vee\text{l or left})\ \frac{\Gamma, \phi \vdash \Delta \quad \Gamma, \psi \vdash \Delta}{\Gamma, \phi \vee \psi \vdash \Delta} \qquad (\wedge\text{l and left})\ \frac{\Gamma, \phi, \psi \vdash \Delta}{\Gamma, \phi \wedge \psi \vdash \Delta} \qquad (\rightarrow\text{l imply left})\ \frac{\Gamma \vdash \phi, \Delta \quad \Gamma, \psi \vdash \Delta}{\Gamma, \phi \rightarrow \psi \vdash \Delta}$$

$$(ax \text{ close})\ \frac{}{\Gamma, \phi \vdash \phi, \Delta} \qquad (cut)\ \frac{\Gamma \vdash \phi, \Delta \quad \Gamma, \phi \vdash \Delta}{\Gamma \vdash \Delta}$$

$$(\langle;\rangle \text{ compose})\ \frac{\langle\alpha\rangle\langle\beta\rangle\phi}{\langle\alpha;\beta\rangle\phi} \qquad (\langle{}^{*n}\rangle \text{ unwind})\ \frac{\phi \vee \langle\alpha\rangle\langle\alpha^*\rangle\phi}{\langle\alpha^*\rangle\phi} \qquad (\langle{:=}\rangle \text{ assign})\ \frac{\phi_{x_1}^{\theta_1}\cdots{}_{x_n}^{\theta_n}}{\langle x_1 := \theta_1, .\,.\,, x_n := \theta_n\rangle\phi}$$

$$([;] \text{ compose})\ \frac{[\alpha][\beta]\phi}{[\alpha;\beta]\phi} \qquad ([{}^{*n}] \text{ unwind})\ \frac{\phi \wedge [\alpha][\alpha^*]\phi}{[\alpha^*]\phi} \qquad ([{:=}] \text{ assign})\ \frac{\langle x_1 := \theta_1, .\,.\,, x_n := \theta_n\rangle\phi}{[x_1 := \theta_1, .\,.\,, x_n := \theta_n]\phi}$$

$$(\langle\cup\rangle \text{ choice})\ \frac{\langle\alpha\rangle\phi \vee \langle\beta\rangle\phi}{\langle\alpha \cup \beta\rangle\phi} \qquad (\langle?\rangle \text{ test})\ \frac{H \wedge \psi}{\langle ?H\rangle\psi} \qquad (\langle'\rangle \text{ ODE solve})\ \frac{\exists t{\geq}0\, \big((\forall 0{\leq}\tilde{t}{\leq}t\, \langle\mathcal{S}(\tilde{t})\rangle H) \wedge \langle\mathcal{S}(t)\rangle\phi\big)}{\langle x_1' = \theta_1, .\,.\,, x_n' = \theta_n \,\&\, H\rangle\phi}$$ [1]

$$([\cup] \text{ choice})\ \frac{[\alpha]\phi \wedge [\beta]\phi}{[\alpha \cup \beta]\phi} \qquad ([?] \text{ test})\ \frac{H \rightarrow \psi}{[?H]\psi} \qquad (['] \text{ ODE solve})\ \frac{\forall t{\geq}0\, \big((\forall 0{\leq}\tilde{t}{\leq}t\, \langle\mathcal{S}(\tilde{t})\rangle H) \rightarrow \langle\mathcal{S}(t)\rangle\phi\big)}{[x_1' = \theta_1, .\,.\,, x_n' = \theta_n \,\&\, H]\phi}$$ [1]

$$(\forall\text{r all right})\ \frac{\Gamma \vdash \phi(s(X_1, .\,.\,, X_n)), \Delta}{\Gamma \vdash \forall x\, \phi(x), \Delta}$$ [2]

$$(\exists\text{r exists right})\ \frac{\Gamma \vdash \phi(X), \Delta}{\Gamma \vdash \exists x\, \phi(x), \Delta}$$

$$(\exists\text{l exists left})\ \frac{\Gamma, \phi(s(X_1, .\,.\,, X_n)) \vdash \Delta}{\Gamma, \exists x\, \phi(x) \vdash \Delta}$$ [2]

$$(\forall\text{l all left})\ \frac{\Gamma, \phi(X) \vdash \Delta}{\Gamma, \forall x\, \phi(x) \vdash \Delta}$$

$$(\text{i}\forall \text{ quantifier elimination})\ \frac{\Gamma \vdash \mathrm{QE}(\forall X\, (\Phi(X) \vdash \Psi(X))), \Delta}{\Gamma, \Phi(s(X_1, .\,.\,, X_n)) \vdash \Psi(s(X_1, .\,.\,, X_n)), \Delta}$$ [3]

$$(\text{i}\exists \text{ eliminate existential})\ \frac{\Gamma \vdash \mathrm{QE}(\exists X\, \bigwedge_i (\Phi_i \vdash \Psi_i)), \Delta}{\Gamma, \Phi_1 \vdash \Psi_1, \Delta \quad \ldots \quad \Gamma, \Phi_n \vdash \Psi_n, \Delta}$$ [4]

$$([]\text{ generalization})\ \frac{\Gamma \vdash [\alpha]\phi, \Delta \qquad \Gamma \vdash \forall^\alpha(\phi \rightarrow \psi), \Delta}{\Gamma \vdash [\alpha]\psi, \Delta} \qquad (\langle\rangle\text{ generalization})\ \frac{\Gamma \vdash \langle\alpha\rangle\phi, \Delta \qquad \Gamma \vdash \forall^\alpha(\phi \rightarrow \psi), \Delta}{\Gamma \vdash \langle\alpha\rangle\psi, \Delta}$$

$$(ind \text{ loop invariant})\ \frac{\Gamma \vdash \phi, \Delta \qquad \Gamma \vdash \forall^\alpha(\phi \rightarrow [\alpha]\phi), \Delta \qquad \Gamma \vdash \forall^\alpha(\phi \rightarrow \psi), \Delta}{\Gamma \vdash [\alpha^*]\psi, \Delta}$$

$$(con \text{ loop convergence})\ \frac{\Gamma \vdash \exists v\, \varphi(v), \Delta \qquad \Gamma \vdash \forall^\alpha \forall v{>}0\, (\varphi(v) \rightarrow \langle\alpha\rangle\varphi(v-1)), \Delta \qquad \Gamma \vdash \forall^\alpha(\exists v{\leq}0\, \varphi(v) \rightarrow \psi), \Delta}{\Gamma \vdash \langle\alpha^*\rangle\psi, \Delta}$$

$$(DI \text{ differential invariant})\ \frac{\Gamma, H \vdash F, \Delta \qquad \Gamma \vdash \forall^\alpha(H \rightarrow F'{}_{x_1'}^{\theta_1}\cdots{}_{x_n'}^{\theta_n}), \Delta}{\Gamma \vdash [x_1' = \theta_1, .\,.\,, x_n' = \theta_n \,\&\, H]F, \Delta}$$

$$(DV \text{ differential variant})\ \frac{\Gamma \vdash [x_1' = \theta_1, .\,.\,, x_n' = \theta_n \,\&\, {\sim}F]H, \Delta \qquad \Gamma \vdash \exists\varepsilon{>}0\, \forall^\alpha(\neg F \wedge H \rightarrow (F' \geq \varepsilon)_{x_1'}^{\theta_1}\cdots{}_{x_n'}^{\theta_n}), \Delta}{\Gamma \vdash \langle x_1' = \theta_1, .\,.\,, x_n' = \theta_n \,\&\, H\rangle F, \Delta}$$ [5]

$$(DW \text{ differential weaken})\ \frac{\Gamma \vdash \forall^\alpha(H \rightarrow \phi), \Delta}{\Gamma \vdash [x' = \theta \,\&\, H]\phi, \Delta}$$

$$(DC \text{ differential cut})\ \frac{\Gamma \vdash [x' = \theta \,\&\, H]C, \Delta \qquad \Gamma \vdash [x' = \theta \,\&\, (H \wedge C)]\phi, \Delta}{\Gamma \vdash [x' = \theta \,\&\, H]\phi, \Delta}$$

$$(DA \text{ differential auxiliaries})\ \frac{\phi \leftrightarrow \exists y\, \psi \qquad \Gamma \vdash [x' = \theta, y' = \vartheta \,\&\, H]\psi, \Delta}{\Gamma \vdash [x' = \theta \,\&\, H]\phi, \Delta}$$ [6]

$$(IA \text{ auxiliary variable})\ \frac{\Gamma \vdash [y := \theta]\phi, \Delta}{\Gamma \vdash \phi, \Delta}$$ [7]

$$(\langle{:}{*}\rangle \text{ random})\ \frac{\exists X\, \langle x := X\rangle\phi}{\langle x := *\rangle\phi}$$ [8]

$$([{:}{*}] \text{ random})\ \frac{\forall X\, [x := X]\phi}{[x := *]\phi}$$ [8]

[1] t and $\tilde{t}$ are new logical variables and $\langle\mathcal{S}(t)\rangle$ is the discrete assignment set $\langle x_1 := y_1(t), .\,.\,, x_n := y_n(t)\rangle$ with simultaneous solutions $y_1, .\,.\,, y_n$ of the respective differential equations with constant symbols x_i as symbolic initial values.

[2] s is a new (Skolem) function symbol and $X_1, .\,.\,, X_n$ are all free logical variables of $\forall x\, \phi(x)$.

[3] X is a new logical variable. Further, QE needs to be defined for the formula in the premise.

[4] Among all open branches, free logical variable X only occurs in the branches $\Gamma, \Phi_i \vdash \Psi_i, \Delta$. Further, QE needs to be defined for the formula.

[5] F contains no equalities and the differential equations are Lipschitz continuous.

[6] y new program variable and $y' = \vartheta, y(0) = y_0$ has a solution $y : [0, \infty) \rightarrow \mathbb{R}^n$ for each y_0

[7] y new program variable

[8] X new logical variable

Figs. 2.11 and 3.9 KeYmaera implementation of proof rules for differential dynamic logic ($\mathsf{d}\mathcal{L}$)

[1] André Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, Heidelberg, 2010.

[2] André Platzer. The structure of differential invariants and differential cut elimination. *Logical Methods in Computer Science*, 2012.

[3] André Platzer. A complete axiomatization of quantified differential dynamic logic for distributed hybrid systems. *Logical Methods in Computer Science*, 2012. Special issue for selected papers from CSL'10.