04: Safety & Contracts
Logical Foundations of Cyber-Physical Systems

André Platzer
1. Learning Objectives
2. Quantum the Acrophobic Bouncing Ball
3. Contracts for CPS
   - Safety of Robots
   - Safety of Bouncing Balls
4. Logical Formulas for Hybrid Programs
5. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Notational Convention
6. Identifying Requirements of a CPS
7. Summary
Outline

1. Learning Objectives
2. Quantum the Acrophobic Bouncing Ball
3. Contracts for CPS
   - Safety of Robots
   - Safety of Bouncing Balls
4. Logical Formulas for Hybrid Programs
5. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Notational Convention
6. Identifying Requirements of a CPS
7. Summary
Learning Objectives

Safety & Contracts

rigorous specification
contracts
preconditions
postconditions
differential dynamic logic

discrete + continuous
analytic specification

model semantics
reasoning principles

CT
M&C  CPS
Outline

1. Learning Objectives

2. Quantum the Acrophobic Bouncing Ball

3. Contracts for CPS
   - Safety of Robots
   - Safety of Bouncing Balls

4. Logical Formulas for Hybrid Programs

5. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Notational Convention

6. Identifying Requirements of a CPS

7. Summary
Example (Quantum the Bouncing Ball)
Example (Quantum the Bouncing Ball)

\[ \{x' = v, v' = -g\} \]
Example (Quantum the Bouncing Ball)

\[ \{ x' = v, v' = -g \} \]
Quantum the Acrophobic Bouncing Ball

Example (Quantum the Bouncing Ball)

\[ \{ x' = v, v' = -g & x \geq 0 \} \]
Example (Quantum the Bouncing Ball)

\[ \{ x' = v, v' = -g \& x \geq 0 \} ; \\
\quad \text{if}(x = 0) \; v := -cv \]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\{ x' &= v, 
      v' &= -g \quad \& \quad x \geq 0 \}; \\
\text{if}(x = 0) \quad &v := -cv
\end{align*}
\]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\{x' = v, v' = -g & \land x \geq 0\}; \\
\text{if}(x = 0) & \quad v := -cv
\end{align*}
\]
Example (Quantum the Bouncing Ball)

\[
\{x' = v, v' = -g \& x \geq 0\};
\]

if \(x = 0\) \(v := -cv\)

*
Example (Quantum the Bouncing Ball)

\[
\{ x' = v, \quad v' = -g \quad \& \quad x \geq 0 \}; \\
\text{if}(x = 0) \quad v := -cv
\]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\{x' &= v, \ v' = -g \ & \ & \ x \geq 0\} ; \\
\text{if}(x = 0) \ & \ v := -cv
\end{align*}
\]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\{ &x' = v, \ v' = -g & \& \ x \geq 0 \}; \\
\text{if} (x = 0) \ (v := -cv \cup v := 0) \end{align*}
\]
Example (Quantum the Bouncing Ball)

\[
\{x' = v, \; v' = -g \; \& \; x \geq 0\};
\]

\[
\text{if}(x = 0) \; \; v := -cv^*\]
Outline

1. Learning Objectives
2. Quantum the Acrophobic Bouncing Ball
3. Contracts for CPS
   - Safety of Robots
   - Safety of Bouncing Balls
4. Logical Formulas for Hybrid Programs
5. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Notational Convention
6. Identifying Requirements of a CPS
7. Summary
Three Laws of Robotics Isaac Asimov 1942

1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.
2. A robot must obey the orders given to it by human beings, except where such orders would conflict with the First Law.
3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.

Three Laws of Robotics are not the answer. They are the inspiration!

André Platzer (CMU)
<table>
<thead>
<tr>
<th>Three Laws of Robotics</th>
<th>Isaac Asimov 1942</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.</td>
<td></td>
</tr>
<tr>
<td>2. A robot must obey the orders given to it by human beings, except where such orders would conflict with the First Law.</td>
<td></td>
</tr>
<tr>
<td>3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.</td>
<td></td>
</tr>
</tbody>
</table>
# Three Laws of Robotics

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A robot may not injure a human being or, through inaction, allow a human being to come to harm.</td>
</tr>
<tr>
<td>2</td>
<td>A robot must obey the orders given to it by human beings, except where such orders would conflict with the First Law.</td>
</tr>
<tr>
<td>3</td>
<td>A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.</td>
</tr>
</tbody>
</table>

*Isaac Asimov 1942*

---

Three Laws of Robotics are not the answer. They are the inspiration!
Example (Quantum the Bouncing Ball)

\[\{x' = v, v' = -g \& x \geq 0\};
\]

\[\text{if}(x = 0) \ v := -cv)^*\]
Example (Quantum the Bouncing Ball)

\[
\text{ensures}(0 \leq x)
\]

\[
\{x' = v, v' = -g \& x \geq 0\}; \\
\text{if}(x = 0) v := -cv
\]

\*
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\text{ensures}(0 \leq x) \\
\text{ensures}(x \leq H) \\
\{x' = v, v' = -g \& x \geq 0\}; \\
\text{if}(x = 0) v := -cv)^*
\end{align*}
\]
Example (Quantum the Bouncing Ball)

**requires**$(x = H)$

**ensures**$(0 \leq x)$

**ensures**$(x \leq H)$

$\{x' = v, \ v' = -g \& x \geq 0\};$

if$(x = 0) \ v := -cv)^*$
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\text{requires} & \quad (x = H) \\
\text{requires} & \quad (0 \leq H) \\
\text{ensures} & \quad (0 \leq x) \\
\text{ensures} & \quad (x \leq H) \\
& \quad (\{x' = v, v' = -g \land x \geq 0\}; \\
& \quad \text{if}(x = 0) v := -cv)^* 
\end{align*}
\]
Example (Quantum the Bouncing Ball)

- **requires** $(x = H)$
- **requires** $(0 \leq H)$
- **ensures** $(0 \leq x)$
- **ensures** $(x \leq H)$

$(\{x' = v, v' = -g & x \geq 0\};$

if $(x = 0) v := -cv)^* \text{@invariant}(x \geq 0)$
Example (Quantum the Bouncing Ball)

\textbf{Precondition:} \[ \text{requires} \left( 0 \leq H \right) \]

\textbf{FOL:} \[ x = H \land 0 \leq H \] in FOL

\textbf{Postcondition:} \[ \text{ensures} \left( 0 \leq x \right) \]

\textbf{FOL:} \[ 0 \leq x \land x \leq H \] in FOL

\[ \{x' = v, v' = -g \land x \geq 0 \}; \]

\[ \text{if} (x = 0) \ v := -cv \] *\textbf{Invariant:} \[ x \geq 0 \]

André Platzer (CMU)
CPS contracts are crucial for CPS safety. We need to understand CPS programs and contracts and how we can convince ourselves that a CPS program respects its contract.

Contracts are at a disadvantage compared to full logic.

**Logic is for Specification and Reasoning**

1. Specification of a whole CPS program.
2. Analytic inspection of its parts.
3. Argumentative relations between contracts and program parts.
   “Yes, this CPS program meets its contract, and here’s why . . .”
Example (Quantum the Bouncing Ball)

requires($x = H$)
requires($0 \leq H$)
ensures($0 \leq x$)
ensures($x \leq H$)

($\{x' = v, v' = -g \& x \geq 0\};$
  if($x = 0$) $v := -cv$)
Example (Quantum the Bouncing Ball)

\[ \text{requires}(x = H) \]
\[ \text{requires}(0 \leq H) \]
\[ \text{ensures}(0 \leq x) \]
\[ \text{ensures}(x \leq H) \]

\[
\{x' = v, v' = -g \land x \geq 0\}; \\
\text{if}(x = 0) v := -cv
\]

Precondition:
\[ x = H \land 0 \leq H \text{ in FOL} \]
Example (Quantum the Bouncing Ball)

\textbf{Precondition:}  
\begin{align*}
\text{requires}(x = H) & \\
\text{requires}(0 \leq H) & \quad \text{(in FOL)} \\
\text{ensures}(0 \leq x) & \quad \text{(in FOL)} \\
\text{ensures}(x \leq H) & \\
\{x' = v, v' = -g \& x \geq 0\}; & \text{if}(x = 0) v := -cv)^* \\
\end{align*}

\textbf{Postcondition:}  
\begin{align*}
0 \leq x \land x \leq H & \quad \text{(in FOL)}
\end{align*}
Example (Quantum the Bouncing Ball)

**Precondition:**

\[ x = H \land 0 \leq H \] in FOL

**Postcondition:**

\[ 0 \leq x \land x \leq H \] in FOL

\[ \forall x \geq 0; \text{if} (x = 0) v := -cv \]
Example (Quantum the Bouncing Ball)

**Precondition:**
\[ x = H \land 0 \leq H \text{ in FOL} \]

**Postcondition:**
\[ 0 \leq x \land x \leq H \text{ in FOL} \]

\[ \{ x' = v, \ v' = -g \land x \geq 0 \}; \]
\[ \text{if}(x = 0) \ v := -cv \]
Example (Quantum the Bouncing Ball)

- **requires**($x = H$)
- **requires**($0 \leq H$)
- **ensures**($0 \leq x$)
- **ensures**($x \leq H$)

\[
\begin{align*}
\{x' = v, v' = -g & \land x \geq 0\}; \\
\text{if}(x = 0) v := -cv
\end{align*}
\]

Precondition:

\[x = H \land 0 \leq H\text{ in FOL}\]

Postcondition:

\[0 \leq x \land x \leq H\text{ in FOL}\]
Example (Quantum the Bouncing Ball)

- requires($x = H$)
- requires($0 \leq H$)
- ensures($0 \leq x$)
- ensures($x \leq H$)

$\{x' = v, v' = -g \& x \geq 0\};$

if($x = 0$) $v := -cv$*

Precondition:

$x = H \land 0 \leq H$ in FOL

Postcondition:

$0 \leq x \land x \leq H$ in FOL
Example (Quantum the Bouncing Ball)

requires\( (x = H) \)
requires\( (0 \leq H) \)
ensures\( (0 \leq x) \)
ensures\( (x \leq H) \)
\((\{x' = v, v' = -g \& x \geq 0\};\)
if\( (x = 0) v := -cv)^*\)

Precondition: \( x = H \land 0 \leq H \) in FOL

Postcondition: \( 0 \leq x \land x \leq H \) in FOL
Example (Quantum the Bouncing Ball)

\[
((x' = v, v' = -g \& x \geq 0); \text{if}(x = 0) v := -cv)^*]
\]

Precondition: \(x = H \land 0 \leq H\) in FOL

Postcondition: \(0 \leq x \land x \leq H\) in FOL

 Andre Platzer  
 LFCPS/04: Safety & Contracts  
 LFCPS/04 8 / 16
[\((\{x' = v, v' = -g \land x \geq 0\}; \text{if}(x = 0)\ v := -cv)^{\ast}\)](x \leq H)

Example (Quantum the Bouncing Ball)

- **requires**($x = H$)
- **requires**($0 \leq H$)
- **ensures**($0 \leq x$)
- **ensures**($x \leq H$)

($\{x' = v, v' = -g \land x \geq 0\}; \text{if}(x = 0)\ v := -cv)^{\ast}$)

Precondition:
$x = H \land 0 \leq H$ in FOL

Postcondition:
$0 \leq x \land x \leq H$ in FOL
Example (Quantum the Bouncing Ball)

**requires**($x = H$)

**requires**($0 \leq H$)

**ensures**($0 \leq x$)

**ensures**($x \leq H$)

\[
\begin{align*}
\{(x' = v, v' = -g \& x \geq 0); & \text{ if}(x = 0) \; v := -cv\}^* (0 \leq x) \\
\{(x' = v, v' = -g \& x \geq 0); & \text{ if}(x = 0) \; v := -cv\}^* (x \leq H)
\end{align*}
\]

Precondition:

$x = H \land 0 \leq H$ in FOL

Postcondition:

$0 \leq x \land x \leq H$ in FOL
Contracts for Quantum the Acrophobic Bouncing Ball

Example (Quantum the Bouncing Ball)

requires\((x = H)\)

requires\((0 \leq H)\)

ensures\((0 \leq x)\)

ensures\((x \leq H)\)

\(((\{x' = v, v' = -g \& x \geq 0\}; \text{if}(x=0) \ v := -cv)^*)\)(\(0 \leq x\))

\(((\{x' = v, v' = -g \& x \geq 0\}; \text{if}(x=0) \ v := -cv)^*)\)(\(x \leq H\))

\(((\{x' = v, v' = -g \& x \geq 0\}; \text{if}(x=0) \ v := -cv)^*)\)(\(0 \leq x \land x \leq H\))

Precondition: \(x = H \land 0 \leq H\) in FOL

Postcondition: \(0 \leq x \land x \leq H\) in FOL
Contracts for Quantum the Acrophobic Bouncing Ball

\[
[\{x' = v, v' = -g & x \geq 0\}; \text{if}(x=0) \ v := -cv]^*](0 \leq x)
\wedge
[\{x' = v, v' = -g & x \geq 0\}; \text{if}(x=0) \ v := -cv]^*(x \leq H)
\leftrightarrow
[\{x' = v, v' = -g & x \geq 0\}; \text{if}(x=0) \ v := -cv]^*[0 \leq x \land x \leq H]
\]

Example (Quantum the Bouncing Ball)

**Precondition:**

\[x = H \land 0 \leq H \text{ in FOL}\]

**Postcondition:**

\[0 \leq x \land x \leq H \text{ in FOL}\]

- **requires** \(x = H\)
- **requires** \(0 \leq H\)
- **ensures** \(0 \leq x\)
- **ensures** \(x \leq H\)

\(
\{x' = v, v' = -g & x \geq 0\}; \\
\text{if}(x = 0) \ v := -cv)^*
\)
Contracts for Quantum the Acrophobic Bouncing Ball

Example (Quantum the Bouncing Ball)

\[
[(\{x' = v, v' = -g \& x \geq 0\}; \text{if}(x=0) \ v := -cv)^*](0 \leq x)
\]

Precondition:

\[x = H \land 0 \leq H\] in FOL

Postcondition:

\[0 \leq x \land x \leq H\] in FOL

\[
\{x' = v, v' = -g \& x \geq 0\}; \text{if}(x = 0) \ v := -cv\]^*
\]
Example (Quantum the Bouncing Ball)

- **requires**\( (x = H) \)
- **requires**\( (0 \leq H) \)
- **ensures**\( (0 \leq x) \)
- **ensures**\( (x \leq H) \)

\[ \{ x' = v, \ v' = -g \land x \geq 0 \}; \ \text{if\( (x=0) \ v := -cv \}^* \right\} (0 \leq x) \]

**Precondition:**
\[ x = H \land 0 \leq H \text{ in FOL} \]

**Postcondition:**
\[ 0 \leq x \land x \leq H \text{ in FOL} \]
0 \leq x \land x = H \rightarrow [(\{x' = v, v' = -g \& x \geq 0\}; \text{if}(x=0) \ v := -cv)^*](0 \leq x)

Example (Quantum the Bouncing Ball)

**requires**($x = H$)  
**requires**($0 \leq H$)  
**ensures**($0 \leq x$)  
**ensures**($x \leq H$)  
($\{x' = v, v' = -g \& x \geq 0\}; \text{if}(x = 0) \ v := -cv)^*$

Precondition: $x = H \land 0 \leq H$ in FOL

Postcondition: $0 \leq x \land x \leq H$ in FOL
Outline

1. Learning Objectives
2. Quantum the Acrophobic Bouncing Ball
3. Contracts for CPS
   - Safety of Robots
   - Safety of Bouncing Balls
4. Logical Formulas for Hybrid Programs
5. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Notational Convention
6. Identifying Requirements of a CPS
7. Summary
Definition (Syntax of differential dynamic logic)

The formulas of differential dynamic logic are defined by the grammar:

\[ P, Q ::= e \geq \dot{e} \mid \neg P \mid P \land Q \mid P \lor Q \mid P \rightarrow Q \mid \forall x \ P \mid \exists x \ P \mid [\alpha]P \mid \langle \alpha \rangle P \]
Definition (Syntax of differential dynamic logic)

The formulas of differential dynamic logic are defined by the grammar:

\[ P, Q ::= e \geq \bar{e} \mid \neg P \mid P \land Q \mid P \lor Q \mid P \rightarrow Q \mid \forall x P \mid \exists x P \mid [\alpha]P \mid \langle\alpha\rangle P \]
Definition (dL Formulas)

\[ \omega \]

\[ [\alpha] P \]

\[ \beta \]

\[ P \]

\[ P \]

\[ P \]
Definition (dL Formulas)

\[ \langle \alpha \rangle P \]
Definition (dL Formulas)

\[
[\alpha]P
\]

\[\omega\]-span

\[\alpha\]-span
Definition (dL Formulas)

$[\alpha] P$

$\langle \beta \rangle P$

$\omega$

$\beta$-span

$\alpha$-span
Definition (dL Formulas)
The formulas of differential dynamic logic are defined by the grammar:

\[ P, Q ::= e \geq \bar{e} \mid \neg P \mid P \land Q \mid P \lor Q \mid P \rightarrow Q \mid \forall x P \mid \exists x P \mid [\alpha]P \mid \langle \alpha \rangle P \]

**Definition (dL semantics)**

\[
[ e \geq \bar{e}] = \{ \omega : \omega[e] \geq \omega[\bar{e}] \}
\]

\[
[\neg P] = [P]^C = S \setminus [P]
\]

\[
[P \land Q] = [P] \cap [Q]
\]

\[
[P \lor Q] = [P] \cup [Q]
\]

\[
[P \rightarrow Q] = [P]^C \cup [Q]
\]

\[
[\langle \alpha \rangle P] = [\alpha] \circ [P] = \{ \omega : \nu \in [P] \text{ for some } \nu : (\omega, \nu) \in [\alpha] \}
\]

\[
[[\alpha]P] = [\neg \langle \alpha \rangle \neg P] = \{ \omega : \nu \in [P] \text{ for all } \nu : (\omega, \nu) \in [\alpha] \}
\]

\[
[\exists x P] = \{ \omega : \omega^r_x \in [P] \text{ for some } r \in \mathbb{R} \}
\]

\[
[\forall x P] = \{ \omega : \omega^r_x \in [P] \text{ for all } r \in \mathbb{R} \}
\]

\[
\omega^d(y) = \begin{cases} 
  d & \text{ if } y = x \\
  \omega(y) & \text{ if } y \neq x
\end{cases}
\]
Differential Dynamic Logic: Syntax & Semantics

\[ [P] \] the set of states in which formula \( P \) is true
\[ \omega \in [P] \] formula \( P \) is true in state \( \omega \), alias \( \omega \models P \)
\[ \models P \] formula \( P \) is valid, i.e., true in all states \( \omega \), i.e., \( [P] = S \)

### Definition (dL semantics)

\[
\begin{align*}
[e \geq \bar{e}] &= \{ \omega : \omega[e] \geq \omega[\bar{e}] \} \\
\lnot P &= [P]^C = S \setminus [P] \\
[P \land Q] &= [P] \cap [Q] \\
[P \lor Q] &= [P] \cup [Q] \\
[P \to Q] &= [P]^C \cup [Q] \\
[\langle \alpha \rangle P] &= [\alpha] \circ [P] = \{ \omega : \nu \in [P] \text{ for some } \nu : (\omega, \nu) \in [\alpha] \} \\
[[\alpha]P] &= [\lnot \langle \alpha \rangle \lnot P] = \{ \omega : \nu \in [P] \text{ for all } \nu : (\omega, \nu) \in [\alpha] \} \\
\exists x P &= \{ \omega : \omega^r_x \in [P] \text{ for some } r \in \mathbb{R} \} \\
\forall x P &= \{ \omega : \omega^r_x \in [P] \text{ for all } r \in \mathbb{R} \}
\end{align*}
\]
Differential Dynamic Logic: Syntax & Semantics

\([P]\) the set of states in which formula \(P\) is true
\(\omega \in [P]\) formula \(P\) is true in state \(\omega\), alias \(\omega \models P\)
\(\models P\) formula \(P\) is valid, i.e., true in all states \(\omega\), i.e., \([P] = S\)

\[\exists d \[x := 1; x' = d\]x \geq 0\] and \([x := x + 1; x' = d]\]x \geq 0 and \(\langle x' = d\rangle x \geq 0\)

**Definition (dL semantics)**

\([e \geq \tilde{e}] = \{\omega : \omega[e] \geq \omega[\tilde{e}]\}\)
\([\neg P] = [P]^c = S \setminus [P]\)
\([P \land Q] = [P] \cap [Q]\)
\([P \lor Q] = [P] \cup [Q]\)
\([P \to Q] = [P]^c \cup [Q]\)
\([\langle \alpha \rangle P] = \langle \alpha \rangle \circ [P] = \{\omega : \nu \in [P] \text{ for some } \nu : (\omega, \nu) \in [\alpha]\}\)
\([\exists x P] = \{\omega : \omega^r_x \in [P] \text{ for some } r \in \mathbb{R}\}\)
\([\forall x P] = \{\omega : \omega^r_x \in [P] \text{ for all } r \in \mathbb{R}\}\)
the set of states in which formula $P$ is true

$\omega \in \mathcal{L}[P]$ formula $P$ is true in state $\omega$, alias $\omega \models P$

$\models P$ formula $P$ is valid, i.e., true in all states $\omega$, i.e., $\mathcal{L}[P] = S$

$\models \exists d \left[ x := 1; x' = d \right] x \geq 0$ and $\not\models \left[ x := x + 1; x' = d \right] x \geq 0$ and $\not\models \langle x' = d \rangle x \geq 0$

Definition (dL semantics) ($\mathcal{L} : \text{Fml} \to \wp(S)$)

$$\mathcal{L}[e \geq \tilde{e}] = \{ \omega : \omega[e] \geq \omega[\tilde{e}] \}$$

$$\mathcal{L}[\neg P] = \mathcal{L}[P]^c = S \setminus \mathcal{L}[P]$$

$$\mathcal{L}[P \land Q] = \mathcal{L}[P] \cap \mathcal{L}[Q]$$

$$\mathcal{L}[P \lor Q] = \mathcal{L}[P] \cup \mathcal{L}[Q]$$

$$\mathcal{L}[P \rightarrow Q] = \mathcal{L}[P]^c \cup \mathcal{L}[Q]$$

$$\mathcal{L}[\langle \alpha \rangle P] = \langle \alpha \rangle \circ \mathcal{L}[P] = \{ \omega : \nu \in \mathcal{L}[P] \text{ for some } \nu : (\omega, \nu) \in \mathcal{L}[\alpha] \}$$

$$\mathcal{L}[\neg \langle \alpha \rangle \neg P] = \{ \omega : \nu \in \mathcal{L}[P] \text{ for all } \nu : (\omega, \nu) \in \mathcal{L}[\alpha] \}$$

$$\mathcal{L}[\exists x P] = \{ \omega : \omega^r_x \in \mathcal{L}[P] \text{ for some } r \in \mathbb{R} \}$$

$$\mathcal{L}[\forall x P] = \{ \omega : \omega^r_x \in \mathcal{L}[P] \text{ for all } r \in \mathbb{R} \}$$
Notational Conventions: Precedence

Convention (Operator Precedence)

1. Unary operators (e.g., *, ¬, ∀x, ∃x, [α], ⟨α⟩) bind stronger than binary
2. ∧ binds stronger than ∨, which binds stronger than →, ↔
3. ; binds stronger than ∪
4. Arithmetic operators +, −, · associate to the left
5. Logical and program operators associate to the right

Example (Operator Precedence)

\[ [α]P ∧ Q \equiv ([α]P) ∧ Q \quad \forall x P ∧ Q \equiv (\forall x P) ∧ Q \quad \forall x P → Q \equiv (\forall x P) → Q \]
\[ α; β ∪ γ \equiv (α; β) ∪ γ \quad α ∪ β; γ \equiv α ∪ (β; γ) \quad α; β^* \equiv α; (β^*) \]
\[ P → Q → R \equiv P → (Q → R). \]

But →, ↔ expect explicit parentheses. Illegal: \[ P → Q ↔ R \quad P ↔ Q → R \]
Outline

1. Learning Objectives
2. Quantum the Acrophobic Bouncing Ball
3. Contracts for CPS
   - Safety of Robots
   - Safety of Bouncing Balls
4. Logical Formulas for Hybrid Programs
5. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Notational Convention
6. Identifying Requirements of a CPS
7. Summary
Example (Bouncing Ball)

\[(\{x' = v, v' = -g \ & \ x \geq 0\};
\text{if}(x = 0) \ v := -cv)^*\]
Example (Bouncing Ball)

\[ H = x \geq 0 \rightarrow [(\{ x' = v, v' = -g \land x \geq 0 \}; \text{if}(x = 0) \ v := -cv)^*] \ 0 \leq x \leq H \]
Example (Bouncing Ball)

\[ H = x \geq 0 \quad \rightarrow \quad \left[ \left\{ x' = v, v' = -g \land x \geq 0 \right\}; \right. \]
\[ \left. \text{if}(x = 0) \quad v := -cv \right] \quad 0 \leq x \leq H \]
Example (Bouncing Ball)

\[ H = x \geq 0 \land g > 0 \rightarrow \left[ \left\{ x' = v, v' = -g \land x \geq 0 \right\}; \right. \\
\left. \text{if}(x = 0) \; v := -cv \right\}^* \] \; 0 \leq x \leq H
Quantum the Acrophobic Bouncing Ball

Not if \( c > 1 \) for anti-damping

Example (Bouncing Ball)

\[
H = x \geq 0 \land g > 0 \rightarrow \left[ \begin{array}{l}
\{ x' = v, v' = -g & x \geq 0 \} ; \\
\text{if} (x = 0) v := -cv \end{array} \right] 0 \leq x \leq H
\]
Example (Bouncing Ball)

\[ 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \{ x' = v, v' = -g \land x \geq 0 \}; \right. \\
\left. \text{if}(x = 0) v := -cv \right] 0 \leq x \leq H \]
Quantum the Acrophobic Bouncing Ball

Example (Bouncing Ball)

\(\begin{align*}
1 \leq c &\geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \{ x' = v, v' = -g \land x \geq 0 \} ;
\text{if}(x = 0) \ v := -cv \right]^* \right] 0 \leq x \leq H
\end{align*}\)

Not if \(v > 0\) initial climbing
Example (Bouncing Ball)

\[ v \leq 0 \land 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \{ x' = v, v' = -g \land x \geq 0 \}; \right. \\
\left. \text{if}(x = 0) v := -cv \right]\] 0 \leq x \leq H
Quantum the Acrophobic Bouncing Ball

Example (Bouncing Ball)

\[\nu \leq 0 \land 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \{x' = v, v' = -g \land x \geq 0 \}; \right.\]

\[\text{if}(x = 0) \nu := -cv \right] 0 \leq x \leq H\]
Quantum the Acrophobic Bouncing Ball

Example (Bouncing Ball)

\[ v = 0 \land 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \left\{ x' = v, v' = -g \land x \geq 0 \right\}; \right. \\
\text{if} (x = 0) \ v := -cv \right]^* 0 \leq x \leq H \]
Example (Runaround Robot)

\[(x, y) \not= o \rightarrow \begin{bmatrix}
\omega := -1 \\
\omega := 1 \\
\omega := 0
\end{bmatrix}; \quad \begin{bmatrix}
x' = v \\
y' = w \\
v' = \omega w \\
w' = -\omega v
\end{bmatrix}
\]
Example (Runaround Robot)

\[
((\omega := -1 \cup \omega := 1 \cup \omega := 0);
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Example (Runaround Robot)

\[(x, y) \neq o \rightarrow [(\omega := -1 \cup \omega := 1 \cup \omega := 0);\]
\[\{x' = v, y' = w, v' = \omega w, w' = -\omega v\}]^* (x, y) \neq o\]
Example (Runaround Robot)

\[(x, y) \neq o \rightarrow \left[\left((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0);\right)\right.\]
\n\[
\left.\{x' = v, y' = w, v' = \omega w, w' = -\omega v\}\right) * \] (x, y) \neq o
Outline

1 Learning Objectives
2 Quantum the Acrophobic Bouncing Ball
3 Contracts for CPS
   - Safety of Robots
   - Safety of Bouncing Balls
4 Logical Formulas for Hybrid Programs
5 Differential Dynamic Logic
   - Syntax
   - Semantics
   - Notational Convention
6 Identifying Requirements of a CPS
7 Summary
Definition (Hybrid program $\alpha$)

$$\alpha, \beta ::= x := f(x) \mid ?Q \mid x' = f(x) \& Q \mid \alpha \cup \beta \mid \alpha ; \beta \mid \alpha^*$$

Definition (dL Formula $P$)

$$P, Q ::= e \geq \bar{e} \mid \neg P \mid P \land Q \mid \forall x \ P \mid \exists x \ P \mid [\alpha] P \mid \langle \alpha \rangle P$$
### Differential Dynamic Logic $dL$: Syntax

#### Definition (Hybrid program $\alpha$)

\[
\alpha, \beta ::= x := f(x) \mid ?Q \mid x' = f(x) & Q \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^*
\]

#### Definition (dL Formula $P$)

\[
P, Q ::= e \geq \tilde{e} \mid \neg P \mid P \land Q \mid \forall x \ P \mid \exists x \ P \mid [\alpha]P \mid \langle \alpha \rangle P
\]
### Differential Dynamic Logic dL: Semantics

**Definition (Hybrid program semantics)**  
\[
\begin{align*}
[x := f(x)] & = \{ (\omega, \nu) : \nu = \omega \text{ except } \nu[x] = \omega[f(x)] \} \\
[?Q] & = \{ (\omega, \omega) : \omega \in [Q] \} \\
[x' = f(x)] & = \{ (\varphi(0), \varphi(r)) : \varphi \models x' = f(x) \text{ for some duration } r \} \\
[\alpha \cup \beta] & = [\alpha] \cup [\beta] \\
[\alpha ; \beta] & = [\alpha] \circ [\beta] \\
[\alpha^*] & = [\alpha]^* = \bigcup_{n \in \mathbb{N}} [\alpha^n] 
\end{align*}
\]

**Definition (dL semantics)**  
\[
\begin{align*}
[e \geq \tilde{e}] & = \{ \omega : \omega[e] \geq \omega[\tilde{e}] \} \\
[\neg P] & = [P]^c \\
[P \land Q] & = [P] \cap [Q] \\
[\langle \alpha \rangle P] & = [\alpha] \circ [P] = \{ \omega : \nu \in [P] \text{ for some } \nu : (\omega, \nu) \in [\alpha] \} \\
[[\alpha]P] & = [\neg \langle \alpha \rangle \neg P] = \{ \omega : \nu \in [P] \text{ for all } \nu : (\omega, \nu) \in [\alpha] \} \\
[\exists x P] & = \{ \omega : \omega_x^r \in [P] \text{ for some } r \in \mathbb{R} \}
\end{align*}
\]

These definitions provide the semantics for hybrid programs and differential dynamic logic (dL), respectively, allowing for the formal verification of hybrid systems.
André Platzer.
*Logical Foundations of Cyber-Physical Systems.*
URL: http://www.springer.com/978-3-319-63587-3,
doi:10.1007/978-3-319-63588-0.

André Platzer.
*Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics.*
doi:10.1007/978-3-642-14509-4.

André Platzer.
Logics of dynamical systems.

André Platzer.
Differential dynamic logic for hybrid systems.
André Platzer.
A complete uniform substitution calculus for differential dynamic logic.  