Feedback Refinement Relations for the Synthesis of Symbolic Controllers

Gunther Reissig\textsuperscript{1}, Alexander Weber\textsuperscript{1} and Matthias Rungger\textsuperscript{2}

\textsuperscript{1}: Chair of Control Engineering
Universität der Bundeswehr, München

\textsuperscript{2}: Hybrid Control Systems Group
Technische Universität München (TUM)
Controller Synthesis

A conceptual synthesis problem

Given a system $S$ and a desired behavior $\Sigma$, find a controller, i.e., a system $C$ so that the behavior of the closed loop satisfies $B(C \times S) \subseteq \Sigma$.

$\Sigma$, $B(C \times S)$ are subsets of $(U \times Y)^\infty := (U \times Y)^* \cup (U \times Y)^\omega$

- $U$ are the inputs
- $Y$ are the outputs

Examples

- Reference tracking

  $$(u, y) \in \Sigma \iff \lim_{t \to \infty} |y(t) - y_{\text{reference}}(t)| = 0$$

- Optimal control

  $$(u, y) \in \Sigma \iff |(u, y) - \inf J(u, y)| \leq \varepsilon$$

- Linear temporal logic specifications

  $$(u, y) \in \Sigma \iff (u, y) \text{ satisfies } \varphi$$
Abstraction and Refinement for Controller Synthesis

Usually, “direct” synthesis of $C$ from $(S, \Sigma)$ is impossible and a finite substitute $(\hat{S}, \hat{\Sigma})$ is used to determine $C$:

1. Compute a finite substitute $(\hat{S}, \hat{\Sigma})$ of $(S, \Sigma)$
   \- $\hat{S}$ is an “abstraction” or a “symbolic model” of $S$
2. Synthesize controller $\hat{C}$ to enforce $\hat{\Sigma}$ on $\hat{S}$
3. Refine solution $\hat{C}$ to $C$

\[
\begin{align*}
(S, \Sigma) & \quad \rightarrow \quad \hat{S} \quad \hat{\Sigma} \\
\hat{S} \quad \hat{\Sigma} & \quad \rightarrow \quad \hat{C} \\
\hat{C} & \quad \rightarrow \quad C
\end{align*}
\]

finite \quad 2.
abstract \quad \text{concrete}
finite \quad 1.
abstract \quad \text{concrete}
infinite \quad 3.
Correctness Reasoning

Given that \( \hat{C} \) enforces \( \hat{\Sigma} \) on \( \hat{S} \). How to ensure that \( C \) enforces \( \Sigma \) on \( S \)?

- For verification **simulation relations** are used:
  \[
  S \preceq_S \hat{S} \text{ and } B(\hat{S}) \subseteq \hat{\Sigma} \implies B(S) \subseteq \Sigma
  \]

  **Recall:** \( Q \subseteq X \times \hat{X} \) is a simulation relation from \( S \) to \( \hat{S} \) if
  - \( (x, \hat{x}) \in Q \) implies
  - \( \forall x \overset{u}{\longrightarrow} x' \)
  - \( \exists \hat{x} \overset{u}{\longrightarrow} \hat{x}' \)
  - \( (x', \hat{x}') \in Q \)

- For synthesis **alternating simulation relations** are used:
  \[
  \hat{S} \preceq_{AS} S \text{ and } B(\hat{C} \times \hat{S}) \subseteq \hat{\Sigma} \implies \exists C : B(C \times S) \subseteq \Sigma
  \]

  **Recall:** \( Q \subseteq X \times \hat{X} \) is an alternating simulation relation from \( \hat{S} \) to \( S \) if
  - \( (x, \hat{x}) \in Q \) implies
  - \( \forall \hat{u} \text{ “admissible”} \)
  - \( \exists u \text{ “admissible”} \)
  - \( \forall x \overset{u}{\longrightarrow} x' \)
  - \( \exists \hat{x} \overset{\hat{u}}{\longrightarrow} \hat{x}' \)
  - \( (x', \hat{x}') \in Q \)
The Refinement Procedure (\(\equiv\) Controller Transfer)

**ASR conditions**

1. \((x, \hat{x}) \in Q\) implies
2. \(\forall \hat{u} \text{ “admissible”}\)
3. \(\exists u \text{ “admissible”}\)
4. \(\forall x \xrightarrow{u} x'\)
5. \(\exists \hat{x} \xrightarrow{\hat{u}} \hat{x}'\)
6. \((x', \hat{x}') \in Q\)

The closed loop \(C \times S\) follows to

**Construction of refined controller \(C\)**

1. Given \(x, Q\) acts as quantizer and picks \(\hat{x} \in Q(x)\)
2. In the abstract closed loop and admissible \(\hat{u}\) is proposed
3. In the concrete closed loop \(\hat{u}\) is matched by an admissible \(u\)
4. The concrete closed loop proceeds with \(x \xrightarrow{u} x'\)
5. The next \(\hat{x}'\) is determined by \(\hat{x}' \in Q(x')\) and \(\hat{x} \xrightarrow{\hat{u}} \hat{x}'\)
A Closer Look at the Refinement

The closed loop $C \times S$ follows to

Properties:
- Abstraction is contained in the refined controller
- The refined controller requires exact state information

Both properties are critical!
- Typically abstractions consists of $10^6$ states and $10^9$ transitions
- Usually only perturbed/quantized state information is available
- A static controller is not refined to a static controller!
In this Talk

1. A new notion of system relation, termed Feedback Refinement Relation, for controller refinement, so that the closed loop is given by

\[
\dot{\xi} \in f(\xi, u) + W
\]

where \( W \) is a bounded set of disturbances

Feedback refinement relations are necessary and sufficient for the illustrated refinement procedure

2. We show how feedback refinement relations can be used to
   - solve general control problems and
   - synthesize robust controllers

3. We show how to compute abstractions for control systems of the form

\[
\dot{\xi} \in f(\xi, u) + W
\]
Feedback Refinement Relations

- Systems, Solutions, Behavior
- Serial Composition, Feedback Composition
- Main Theorem
Systems: Informal Introduction

- We consider dynamical systems of the form
  \[ x(t + 1) \in F(x(t), u(t)) \]
  \[ y(t) \in H(x(t), u(t)) \]

- where
  - \( x \) is the state
  - \( u \) is the input
  - \( y \) is the output
  - \( F \) is the transition function
  - \( H \) is the output function

In order to define a meaningful serial/feedback composition of this general type of systems we need **internal variables**

In the first and second line, we need to pick the same \( u_2 \)!
Systems

\[ x(t + 1) \in F(x(t), v(t)) \]
\[ (y(t), v(t)) \in H(x(t), u(t)) \]
\[ x(0) \in X_0 \]

A system \( S \) is a tuple \( S = (X, X_0, U, V, Y, F, H) \) where

- \( X, U, V \) and \( Y \) are nonempty sets
  - \( X \) is the state alphabet
  - \( X_0 \subseteq X \) is the initial state alphabet
  - \( U \) is the input alphabet
  - \( V \) is the internal input alphabet
  - \( Y \) is the output alphabet
- \( F : X \times V \Rightarrow X \) is the transition function
- \( H : X \times U \Rightarrow Y \times V \) is the output function and is assumed to be strict, i.e.,
  \[ \forall (x, u) \in X \times U : H(x, u) \neq \emptyset \]

Notation

- We use \( F : X \Rightarrow Y \) to denote set-valued function from \( X \) to \( Y \)
Systems

We call a system $S = (X, X_0, U, V, Y, F, H)$

1. **static** if $X$ is a singleton

2. **Moore** if the output does not depend on the input, i.e.,
   \[(y, v) \in H(x, u) \land u' \in U \implies \exists v' (y, v') \in H(x, u');\]

3. **basic** if $U = V$ and
   \[(y, v) \in H(x, u) \implies v = u;\]

4. **Moore with state output** if $X = Y$ and
   \[(y, v) \in H(x, u) \implies y = x.\]

For a basic Moore system with state output and $X_0 = X$ we simply use

\[S = (X, U, F)\]

We define the set $U_S(x)$ of admissible inputs at the state $x \in X$ by

\[U_S(x) = \{ u \in U \mid F(x, u) \neq \emptyset \}\]
Solutions and Behavior

Let \( S = (X, X_0, U, V, Y, F, H) \) be given. A quadruple

\[
(u, v, x, y) \in (U \times V \times X \times Y)^{[0; T]}
\]

with \( T \in \mathbb{N} \cup \{\infty\} \) is a solution of the system \( S \) on \([0; T]\) if

- \( x(0) \in X_0 \)
- \( x(t + 1) \in F(x(t), v(t)) \) holds for all \( t \in [0; T - 1] \)
- \( (y(t), v(t)) \in H(x(t), u(t)) \) holds for all \( t \in [0; T] \)

Behavior \( \mathcal{B}(S) \) of \( S \)

- An element of the behavior \( (u, y) \in \mathcal{B}(S) \) is an input/output sequence which is “generated” by a solution of the system \( S \)
- The generating solution either is infinite or ends in a blocking input/state pair

\( (u, y) \in \mathcal{B}(S) \) iff

- \( \exists v, x, T(u, v, x, y) \) is a solution of \( S \) on \([0; T]\)
- \( T < \infty \) implies that \( F(x(T - 1), u(T - 1)) = \emptyset \)
Serial Composition

Let $S_i = (X_i, X_{i,0}, U_i, V_i, Y_i, F_i, H_i), \ i \in \{1, 2\}$ be two systems.

- $S_1$ is serial composable with $S_2$
  - if $Y_1 \subseteq U_2$
- The serial composition of $S_1$ and $S_2$ is denoted by $S_2 \circ S_1$
Serial Composition: Example

Sample-and-hold system $S = (\mathbb{R}^n, \mathbb{R}^m, F)$ with

$$F(x, u) = \left\{ e^{A\tau}x + \left( \int_0^{\tau} e^{A(\tau-s)}ds \right) u \right\}$$

Quantizer

- $Q : \mathbb{R}^n \Rightarrow \mathbb{Z}^n$
- $Q(x) = \{ \hat{x} \in \mathbb{Z}^n \mid |x - \hat{x}|_\infty \leq 1 \}$

is identified with the static system $Q = (\{q\}, \{q\}, \mathbb{R}^n, \mathbb{R}^n, \mathbb{Z}^n, F_q, H_q)$

- $F_q(q, x) = \{q\}$ for all $x \in \mathbb{R}^n$
- $H_q(q, x) = Q(x) \times \{x\}$

$S$ is serial composable with $Q$ and $Q \circ S = (\mathbb{R}^n, \mathbb{R}^n, \mathbb{R}^m, \mathbb{R}^m, \mathbb{Z}^n, F, H)$ is basic with

$$H(x, u) = Q(x) \times \{u\}$$
Feedback Composition

Let $S_i = (X_i, X_{i,0}, U_i, V_i, Y_i, F_i, H_i), \ i \in \{1, 2\}$ be two systems.

- $S_1$ is feedback composable with $S_2$ if $Y_2 \subseteq U_1$ and $Y_1 \subseteq U_2$.
- $S_2$ is Moore if $y_2 \rightarrow H_1 \rightarrow y_1 \rightarrow H_2 \rightarrow y_2$.
- $S_1 \times S_2$ is nonempty-valued.
- $S_1 \times S_2$ is a system.
- $S_1$: controller, $S_2$: plant.
Feedback Refinement Relations

Let $S_i = (X_i, U_i, F_i), \ i \in \{1, 2\}$ be two systems and assume $U_2 \subseteq U_1$
A strict relation

$$Q \subseteq X_1 \times X_2$$

is a feedback refinement relation from $S_1$ to $S_2$, denoted by

$$S_1 \preceq_Q S_2$$

if the following holds for all $(x_1, x_2) \in Q$:

1. $U_{S_2}(x_2) \subseteq U_{S_1}(x_1)$
2. $u \in U_{S_2}(x_2) \implies Q(F_1(x_1, u)) \subseteq F_2(x_2, u)$

In words

1. every admissible input of $S_2$ at $x_2$ is an admissible input of $S_1$ at $x_1$
2. every successor $x_1' \in F_1(x_1, u)$ when mapped to $X_2$ via $Q$ is contained in $F_2(x_2, u)$
Comparison with Alternating Simulation Relations

<table>
<thead>
<tr>
<th>ASR</th>
<th>VS</th>
<th>FRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>• ((x_1, x_2) \in Q) implies</td>
<td>• ((x_1, x_2) \in Q) implies</td>
<td>• ((x_1, x_2) \in Q) implies</td>
</tr>
<tr>
<td>• (\forall u_2 \in U_{S_2}(x_2))</td>
<td>• (\forall u \in U_{S_2}(x_2))</td>
<td>• (\forall u \in U_{S_2}(x_2))</td>
</tr>
<tr>
<td>• (\exists u_1 \in U_{S_1}(x_1))</td>
<td>• (u \in U_{S_1}(x_1))</td>
<td>• (u \in U_{S_1}(x_1))</td>
</tr>
<tr>
<td>• (\forall x'_1 \in F_1(x_1, u_1)) we have</td>
<td>• (\forall x'_1 \in F_1(x_1, u)) we have</td>
<td></td>
</tr>
<tr>
<td>(asr) (Q(x'_1) \cap F_2(x_2, u_2) \neq \emptyset)</td>
<td></td>
<td>(Q(x'_1) \subseteq F_2(x_2, u)) (frr)</td>
</tr>
</tbody>
</table>

Difference

1. ASR: inputs \(U_1\) and \(U_2\) can be different
2. (asr) vs (frr)

What is the problem with ASR?

• In the refinement with ASR
  1. for \((x_1, x_2) \in Q\) and \(u_2\) to pick the matching \(u_1\) we need to know concrete state \(x_1\)
  2. and to pick the right \(x'_2 \in Q(x'_1)\) we need to know \(F_2(x_2, u_2)\) so that \(x'_2 \in Q(x'_1)\) and \(x'_2 \in F_2(x_2, u_2)\)

• In the refinement with FRR
  1. \(u\) can directly be applied to \(S_1\)
  2. we can pick any \(x'_2 \in Q(x'_1)\) and use (frr) to ensure \(x'_2 \in F_2(x_2, u)\)
Feedback Refinement Relations: Necessary Conditions

Consider

- \( S_i = (X_i, U_i, F_i), \ i \in \{1, 2\} \)
- \( C = (X_c, X_{c,0}, U_c, V_c, Y_c, F_c, H_c) \)
- strict \( Q \subseteq X_1 \times X_2 \)

and the statements

1. \( C \) is feedback composable with \( S_2 \)
2. \( C \) is feedback composable with \( Q \circ S_1 \)
3. \( B(C \times (Q \circ S_1)) \subseteq B(C \times S_2) \)

**Theorem:** If 1. implies 2. and 3. then \( S_1 \preceq_Q S_2 \).

We have to show

- \( U_2 \subseteq U_1 \)
- \( (x_1, x_2) \in Q \) implies \( U_{S_2}(x_2) \subseteq U_{S_1}(x_1) \)
- \( (x_1, x_2) \in Q \) and \( u \in U_{S_2}(x_2) \) implies \( Q(F_1(x_1, u)) \subseteq F_2(x_2, u) \)

**U_2 \subseteq U_1**

- 1. implies \( Y_c \subseteq U_2 \)
- 2. implies \( Y_c \subseteq U_1 \)
- Since 1. implies 2. we have: \( Y_c \subseteq U_2 \) implies \( Y_c \subseteq U_1 \). Hence \( U_2 \subseteq U_1 \)

We have to show
Feedback Refinement Relations: Sufficient Conditions

Consider

- \( S_i = (X_i, U_i, F_i), \ i \in \{1, 2\} \)
- \( C = (X_c, X_{c,0}, U_c, V_c, Y_c, F_c, H_c) \)
- strict \( Q \subseteq X_1 \times X_2 \)

and the statements

1. \( C \) is feedback composable with \( S_2 \)
2. \( C \) is feedback composable with \( Q \circ S_1 \)
3. \( B(C \times (Q \circ S_1)) \subseteq B(C \times S_2) \)

**Theorem:** If \( S_1 \preceq_Q S_2 \), then (1. and \((*)\)) imply (2. and 3.) where

\[
(y_c, v_c) \in H_c(x_c, x_2) \land F_2(x_2, y_c) = \emptyset \implies F_c(x_c, v_c) = \emptyset
\]

\((*)\)

- \((*)\): \( C \) non-blocking \( \implies S_2 \) non-blocking
- In the proof we need \( C \times (Q \circ S_1) \) non-blocking \( \implies C \times S_2 \) non-blocking
Feedback Refinement Relations: Sufficient Conditions

Consider

- \( S_i = (X_i, U_i, F_i), \ i \in \{1, 2\} \)
- \( C = (X_c, X_{c,0}, U_c, V_c, Y_c, F_c, H_c) \)
- strict \( Q \subseteq X_1 \times X_2 \)

Corollary: If \( S_1 \preceq_Q S_2 \), then 1. and (\( \ast \)) imply

- \( C \circ Q \) is feedback composable with \( S_1 \)
- for every \( (u, x_1) \in B((C \circ Q) \times S_1) \) exists \( (u, x_2) \in B(C \times S_2) \) so that

\( \forall t \in \text{dom}(u, x_1) : (x_1(t), x_2(t)) \in Q \)
Applications

- Symbolic Synthesis
- Robust Synthesis
Specifications

Consider a system $S = (X, X_0, U, V, Y, F, H)$:
- any subset $\Sigma \subseteq (U \times Y)^\infty$ is a specification for $S$
- $S$ satisfies $\Sigma$ if $B(C \times S) \subseteq \Sigma$
- a system $C$ solves the control problem $(S, \Sigma)$ if
  - $C$ is feedback composable with $S$
  - $S$ and $C$ satisfy $(\ast)$ (non-blocking requirement)
  - $C \times S$ satisfies $\Sigma$

Consider
- two systems $S_i = (X_i, U_i, F_i), i \in \{1, 2\}$ with $U_2 \subseteq U_1$
- a specification $\Sigma_1$ for $S_1$
- a strict relation $Q \subseteq X_1 \times X_2$

A specification $\Sigma_2 \subseteq (X_2 \times X_2)^\infty$ for $S_2$ is an abstract specification associated with $S_2$, $S_1$, $\Sigma_1$ and $Q$ if for any $(u, x_1, x_2) \in (U \times X_1 \times X_2)^{[0; T]}, \ T \in \mathbb{N} \cup \{\infty\}$ with

$$(u, x_2) \in \Sigma_2 \text{ and } \forall_{t \in [0; T]} : (x_1(t), x_2(t)) \in Q \implies (u, x_1) \in \Sigma_1$$

We use

$$(S_1, \Sigma_1) \lessdot_Q (S_2, \Sigma_2)$$

to denote the fact that $S_1 \lessdot_Q S_2$ and $\Sigma_2$ is an abstract specification associated with $S_2$, $S_1$, $\Sigma_1$ and $Q$
Symbolic Synthesis

As a direct consequence of the behavioral inclusion $\mathcal{B}(C \times (Q \circ S_1)) \subseteq \mathcal{B}(C \times S_2)$ for any system $C$ and the definition of abstract specification we get:

**Theorem:** Let $(S_1, \Sigma_1) \preceq_Q (S_2, \Sigma_2)$. If $C$ solves $(S_2, \Sigma_2)$ then $C \circ Q$ solves $(S_1, \Sigma_1)$.
We consider

- system $S_1 = (X_1, U_1, F_1)$
- specification $\Sigma_1$ for $S_1$
- quantizer $Q$
- perturbations
  - $P_1$ accounts for actuator imprecisions
  - $P_2$ accounts for measurement noise
  - $P_3$ and $P_4$ are useful to robustify the spec

We show how to construct

- auxiliary system $\hat{S}_1$
- auxiliary quantizer $\hat{Q}$
- auxiliary specification $\hat{\Sigma}_1$

so that if $C \circ \hat{Q}$ solves $(\hat{S}_1, \hat{\Sigma}_1)$ then the behavior of the Perturbed System satisfies $\Sigma_1$.

$C \circ \hat{Q}$ can be computed by an abstract problem $(S_2, \Sigma_2)$ that satisfies

$$(\hat{S}_1, \hat{\Sigma}_1) \preceq_{\hat{Q}} (S_2, \Sigma_2)$$
Computation of Abstractions of Perturbed Control Systems
Computation of Abstractions

We focus on

- \( X_2 \) is a **cover** of \( X_1 \), i.e., \( X_2 \) is a set of non-empty subsets of \( X_1 \) and \( X_1 \subseteq \bigcup_{x_2 \in X_2} x_2 \)
- every cell \( x_2 \in X_2 \) is a subset of \( X_1 \)
- the quantizer \( Q \) is the **set membership relation** \( \in \)
- for a given \( x_1 \in X_1 \) the quantizer \( Q \) picks \( x_2 \in X_2 \) in a non-deterministic fashion so that \( x_1 \in x_2 \)

Let

- \( S_i = (X_i, U_i, F_i) \) be two systems \( i \in \{1, 2\} \)
- \( X_2 \) be a cover by non-empty sets of \( X_1 \)

**Theorem:** \( S_1 \preceq_\in S_2 \) if and only if

1. \( U_2 \subseteq U_1 \) and \( x_1 \in x_2 \in X_2 \) implies \( U_{S_2}(x_2) \subseteq U_{S_1}(x_1) \)
2. \( x_2, x'_2 \in X_2, u \in U_{S_2}(x_2) \) and \( x'_2 \cap F_1(x_2, u) \neq \emptyset \) implies \( x'_2 \in F_2(x_2, u) \)

Computation of abstraction reduces to computation (overapproximation) of reachable sets!
Abstractions of Perturbed Control Systems

We consider a differential inclusion

\[ \dot{\xi} \in f(\xi, u) + W \]  

(\text{**})

We cast the sample-and-hold behavior of (\text{**}) with \( \tau \in \mathbb{R}_{>0} \) as system

\[ S_1 = (\mathbb{R}^n, \mathbb{R}^m, F_1) \]

The abstraction \( S_2 = (X_2, U_2, F_2) \) is given by

- \( U_2 \subseteq \mathbb{R}^m \)
- \( X_2 \) is a cover of \( \mathbb{R}^n \)
- cells are hyper-rectangles with center \( c \in \mathbb{R}^n \) and radius \( r \in \mathbb{R}^n_{>0} \)

\[ x_2 = c + [-r, r] = [c_1 - r_1, c_1 + r_1] \times \ldots \times [c_n - r_n, c_n + r_n] \]

- we overapproximate the reachable set by another hyper-rectangle \( c' + [-r', r'] \)

\[ F_1(c + [-r, r], u) \subseteq c' + [-r', r'] \]

- and then determine the cells \( \bar{c} + [-\bar{r}, \bar{r}] \in X_2 \) with

\[ \bar{c} + [-\bar{r}, \bar{r}] \cap c' + [-r', r'] \neq \emptyset \]
How to compute $c'$ and $r'$?

Theorem:
- consider (***) and $S_1 = (\mathbb{R}^n, \mathbb{R}^m, F_1)$ for $\tau \in \mathbb{R}_{>0}$
- $u \in U$ and $f(\cdot, u)$ is continuously differentiable
- disturbances $W = [-w, w]$, $w \in \mathbb{R}_{\geq 0}$
- a cell $c + [-r, r] \subseteq K$, with $K \subseteq \mathbb{R}^n$ is convex
- $\xi$ sol. of (***), with $\xi(0) \in K \implies \xi(t) \in K$ for all $t \in [0, \tau] \cap \text{dom} \xi$
- let $L \in \mathbb{R}^{n \times n}$ satisfy for all $x \in K$

$$L_{i,j}(u) \geq \begin{cases} D_j f_i(x, u) & \text{if } i = j \\ |D_j f_i(x, u)| & \text{if } i = j \end{cases}$$

$D_j f_i$: partial derivative of $f_i$ w.r.t. $j$th component of 1st argument

Then

$$F_1(c + [-r, r], u) \subseteq c' + [−r', r']$$

where $c'$ and $r'$ follow by the solution at time $\tau$ of

$$\begin{align*} c' : \quad & \dot{y} = f(y, u) \quad y(0) = c \\ r' : \quad & \dot{z} = Lz + w \quad z(0) = r \end{align*}$$

Overapproximation of reachable set is reduced to
- estimation of partial derivatives of $f$
- solution of two unperturbed initial value problems
Numerical Examples

- A planning problem for a mobile robot
- An aircraft landing maneuver
Reach-Avoid Specifications

The desired behavior $\Sigma$ in both examples is defined in terms of three sets

- $A_{init}$: the set of initial states
- $A_{avoid}$: the set of obstacles
- $A_{reach}$: the target set

Given $S = (X, U, F)$, we define a reach-avoid specification $\Sigma \subseteq (U \times X)^\infty$ for $S$ by

$$(u, x) \in \Sigma \iff x(0) \in A_{init} \implies \exists T \ x(T) \in A_{reach} \wedge \forall t \in [0; T] \ x(t) \notin A_{avoid}$$

Given $\hat{S} = (\hat{X}, \hat{U}, \hat{F})$ with $S \preceq \in \hat{S}$ the abstract specification is defined by the reach-void specification induced by the sets

- $\hat{A}_{init} = \{ \hat{x} \in \hat{X} \mid \hat{x} \cap A_{init} \neq \emptyset \}$
- $\hat{A}_{avoid} = \{ \hat{x} \in \hat{X} \mid \hat{x} \cap A_{avoid} \neq \emptyset \}$
- $\hat{A}_{reach} = \{ \hat{x} \in \hat{X} \mid \hat{x} \subseteq A_{reach} \}$
Robot Path Planning

- unicycle/segway dynamics

\[
\begin{align*}
\dot{\xi}_1 &= u_1 \cos(\alpha + \xi_3) \cos(\alpha)^{-1} \\
\dot{\xi}_2 &= u_1 \sin(\alpha + \xi_3) \cos(\alpha)^{-1} \\
\dot{\xi}_3 &= u_1 \tan(u_2)
\end{align*}
\]

where \(\alpha = \arctan(\tan(u_2)/2)\)

- \(\xi_1\) and \(\xi_2\) coordinates in the plane
- \(\xi_3\) orientation
- \(u_1\) forward velocity
- \(u_2\) steering angle

- sampling time \(\tau = 0.3\) sec

- computation of \(S_2 = (X_2, U_2, F_2)\)
  - number of states \(|X_2| \approx 91 \cdot 10^3\)
  - number of inputs \(|U_2| = 49\)
  - number of transitions \(\approx 28.4 \cdot 10^6\)

- computation times
  - 2.33 sec to compute the abstract problem
  - 0.22 sec to solve the abstract problem
Aircraft Landing Maneuver

- aircraft dynamics
  \[
  \dot{\xi}_1 = \frac{1}{m} (u_1 \cos u_2 - D(u_2, \xi_1) - mg \sin \xi_2)
  \]
  \[
  \dot{\xi}_2 = \frac{1}{mx_1} (u_1 \sin u_2 + L(u_2, \xi_1) - mg \cos \xi_2)
  \]
  \[
  \dot{\xi}_3 = \xi_1 \sin \xi_2
  \]

- \(\xi_1\) forward velocity
- \(\xi_2\) flight path angle
- \(\xi_3\) altitude
- \(u_1 \in [0, 16 \cdot 10^4]\) engine thrust
- \(u_2 \in [0, 10^\circ]\) angle of attack

- actuator and measurement errors
  \[P_1(u) = (u + [-5 \cdot 10^3, 5 \cdot 10^3] \times [-0.25^\circ, 0.25^\circ]) \cap U\]
  \[P_2(x) = x + \frac{1}{20} [-0.25, 0.25] \times \frac{1}{20} [-0.05^\circ, 0.05^\circ] \times \frac{1}{20} [-1, 1]\]

- sampling time \(\tau = 0.25\) sec
- computation of \(S_2 = (X_2, U_2, F_2)\)
  - number of states \(|X_2| \approx 9.3 \cdot 10^6\)
  - number of inputs \(|U_2| = 20\)
  - number of transitions \(\approx 9.4 \cdot 10^9\)
- \(\approx 11\) min to compute \((S_2, \Sigma_2)\)
- 26 sec to solve \((S_2, \Sigma_2)\)

- aircraft

- specification:
  steer the aircraft from \(\approx 55\) m close to the ground with a given total and horizontal touchdown velocity

- landing maneuver
Summary and Related Work

We have seen

- Feedback refinement relations
  - Solve general control problems via abstraction and refinement
  - Account for various perturbations

- Computation of abstractions

Related and ongoing work

- Abstraction-based solution of optimal control problems
- Combination of the computation of abstractions within existing toolboxes to solve synthesis problems with more complex specifications
- Various work to reduce the complexity of the computation of the abstraction
  - compositional construction of abstractions
  - merge the computation of the abstraction with the solution of the abstract control problem
  - pick a cell radius so as to minimize number of transitions
  - Gunther: local refinement of the abstract state space

- Gunther: Validated numerics: account for approximation errors in the solution of ODE and other computations
- ...
- ...