Towards Automatic Detection of Implicit Equality Constraints in Stability Verification of Hybrid Systems*

Eike Möhlmann and Oliver Theel
Carl von Ossietzky University of Oldenburg
Department of Computer Science
D-26111 Oldenburg, Germany
{*eike.moehlmann,theel*}*@informatik.uni-oldenburg.de*

Abstract—*We present a powerful heuristic that detects implicit equality constraints that may occur in the specification of systems of constraints in order to find Lyapunov-based certificates of stability for hybrid systems. A hybrid system is a fusion of systems exhibiting discrete-time as well as continuous-time behavior, e.g. embedded systems within a physical environment. Stability is a property which ensures that a system starting in any possible state will reach a desired state and remain there. Such systems are particularly useful where a certain autonomous operation is required, e.g. keeping a certain temperature or speed of a chemical reaction or steering a vehicle over a predefined track. Stable hybrid systems are extremely valuable because after an error has disturbed their normal operation, they automatically "steer back" to normal operation. Stability can be certified by finding a so-called Lyapunov function. The search for this kind of function usually involves solving systems of inequality constraints. We have identified and implemented a heuristic that detects implicitly specified equality constraints and tries to resolve them by substitution.*

Keywords—Hybrid Systems; Automatic Verification; Stability; Lyapunov Theory; Optimization; Sums-of-Squares

I. Introduction

We present a simple but yet powerful heuristic which detects implicitly specified equality constraints in a system of constraints that is generated for the search of Lyapunov functions. A *Lyapunov function* can be used to certify that a given hybrid system is indeed stable. It can be seen as a "generalized metric" whose value indicates for a given system state the "distance" to the desired system state, the so-called *equilibrium point*. The equilibrium point is w.l.o.g. assumed to be the origin, i.e. 0, of the state space. Lyapunov functions have a special shape which guarantee that while the system evolves, their values and, thereby, the "distance" decreases. Thus, any system for which a Lyapunov function can be found, eventually reaches the equilibrium point. For some systems, it is even possible to give a bound on the rate with which the system approaches the origin. In that case, one can compute an upper bound on the time required to reach a certain region around the equilibrium point. This is especially useful if one is interested in *region stability*.

The search for Lyapunov functions involves generating and solving sets of conditioned constraints, where each *conditioned constraint* has the form $\forall x \in P : 0 \preccurlyeq f(x)$. Here, "$0 \preccurlyeq f(x)$" is a linear constraint on polynomials involving some free parameters, where "$\preccurlyeq$" is the positive semi-definiteness operator, and P is the region in which the constraint has to be satisfied. Generating the constraints required by the famous Lyapunov theorem naively may lead to the following snippet:

$$\forall x \in P_1 : 0 \preccurlyeq f_1(x)$$
$$\forall x \in P_2 : 0 \preccurlyeq f_2(x)$$

where $P_1 = P_2$ and $f_1 = -f_2$. Conditioned constraints cannot directly be given to a solver (such as CSDP [1] or SDPA [2]). So, instead, one uses the so-called *S-Procedure* which then constructs the following representation:

$$0 \preccurlyeq f_1(x) - p_1(x)$$
$$0 \preccurlyeq f_2(x) - p_2(x)$$

* This work has been partly supported by the German Research Foundation (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS).

where p_1 (resp. p_2) encodes the condition $\forall x \in P_1$ (resp. $\forall x \in P_2$) by introducing new free parameters. This means that p_1 and p_2 contain different parameters (i.e. $p_1 \neq p_2$) which is good, because it allows more freedom in case $f_1 \neq -f_2$. But in the special case of $f_1 = -f_2$, it imposes unnecessary difficulties since a solver is required to assign some free parameters exact valuations, which is – unfortunately – bad.

In practice, the problem is even worse since numerical solvers sometimes suffer from numerical inaccuracies. Therefore, additional "gaps" are added to the inequalities in order to make the constraints more robust against these numerical issues. Let g_i with $\forall x \in P_i : 0 \preccurlyeq g_i(x)$ where $i \in \{1, 2\}$ be such gap terms. Trying to solve the following constraints:

$$\forall x \in P_1 : 0 \preccurlyeq f_1(x) - g_1(x)$$
$$\forall x \in P_2 : 0 \preccurlyeq f_2(x) - g_2(x)$$

in the case where $P_2 = P_1$ and $f_2 = -f_1$ does not yield a solution since $\forall x \in P_1 : 0 \preccurlyeq g_1(x) \preccurlyeq f_1(x) \wedge 0 \preccurlyeq g_2(x) \preccurlyeq -f_1(x)$ is obviously a contradiction unless $P_1 \subseteq \{0\}$. But the intended conditioned constraint $\forall x \in P_1 : f_1(x) = 0$ may be easy to solve.

For our tool, Stabhyli [3], we have implemented a simple heuristic to detect and resolve such situations. The heuristic analyzes the conditioned constraints before the unconditioned constraints get generated and removes implicit equalities by substitution of free parameters in the other constraints.

Similar techniques to detect implicit equalities are used, for example, in satisfiability modulo theories solving for linear arithmetic [4]. However, we have to deal with polynomial constraints which we show how to handle in the special settings, where the constraints arise from proving stability using Lyapunov functions.

The remainder of the paper is organized as follows: Section II gives the theoretical background and Section III describes the detection and replacement heuristic as well as some examples. A final conclusion is given in Section IV.

II. PRELIMINARIES

In this section, we define the hybrid system model, stability and the techniques required to certify stability of a hybrid system.

Definition 1: A **Hybrid Automaton** is a quintuple

$$\mathcal{H} = (\mathcal{V}, \mathcal{M}, \mathcal{T}, \mathit{Flow}, \mathit{Inv}) \text{ where}$$

- $\mathcal{V}$ is a finite set of *variables* and $\mathcal{S} = \mathbb{R}^{|\mathcal{V}|}$ is the corresponding *continuous state space*,
- $\mathcal{M}$ is a finite set of *modes*,
- $\mathcal{T}$ is a finite set of *transitions* (m_1, G, R, m_2) where
  - $m_1, m_2 \in \mathcal{M}$ are the *source and target mode* of the transition, respectively,
  - $G \subseteq \mathcal{S}$ is a *guard* which restricts the valuations of the variables for which this transition can be taken,
  - $R : \mathcal{S} \to \mathcal{S}$ is the *reset function* which might reset some valuations of the variables,
- $\mathit{Flow} : \mathcal{M} \to [\mathcal{S} \to \mathcal{P}(\mathcal{S})]$ is the *flow function* which assigns a *flow* to every mode. A flow $f \subseteq \mathcal{S} \to \mathcal{P}(\mathcal{S})$ in turn assigns a closed subset of $\mathcal{S}$ to each $x \in \mathcal{S}$, which can be seen as the right hand side of a differential inclusion $\dot{x} \in f(x)$,
- $\mathit{Inv} : \mathcal{M} \to \mathcal{S}$ is the *invariant function* which assigns a closed subset of the continuous state space to each mode $m \in \mathcal{M}$, and therefore restricts valuations of the variables for which this mode can be active.

A *trajectory* of $\mathcal{H}$ is an infinite solution in form of a function $x(t)$ over time. Each solution has an associated (possibly infinite) sequence of modes visited by the trajectory.

Intuitively, stability is a property expressing that all trajectories of the system eventually reach an equilibrium point of the sub-state space and stay in that point forever, given the absence of errors. For technical reasons, the equilibrium point is usually assumed to be the origin of the continuous state space, i.e. 0. This is not a restriction, since a system can always be shifted such that the equilibrium is 0 via a coordinate transformation.

In the following, we refer to $x_{|\mathcal{V}'} \in \mathbb{R}^{|\mathcal{V}'|}$ as the *sub-vector* of a vector $x \in \mathbb{R}^{\mathcal{V}}$ containing only values of variables in $\mathcal{V}' \subseteq \mathcal{V}$.

Definition 2: **Global Asymptotic Stability with Respect to a Subset of Variables [5].**
Let $\mathcal{H} = (\mathcal{V}, \mathcal{M}, \mathcal{T}, Flow, Inv)$ be a hybrid automaton, and let $\mathcal{V}' \subseteq \mathcal{V}$ be the set of variables that are required to converge to the equilibrium point 0. A continuous-time dynamic system $\mathcal{H}$ is called *globally stable (GS) with respect to* $\mathcal{V}'$ if for all functions $x_{|\mathcal{V}'}(t)$,

$$\forall \epsilon > 0 : \exists \delta > 0 : \forall t \geq 0 : \\ \Big|\Big|x(0)\Big|\Big| < \delta \Rightarrow \Big|\Big|x_{|\mathcal{V}'}(t)\Big|\Big| < \epsilon.$$

$\mathcal{H}$ is called *globally attractive (GA) with respect to* $\mathcal{V}'$ if for all functions $x_{|\mathcal{V}'}(t)$,

$$\lim_{t\to\infty} x_{|\mathcal{V}'}(t) = 0, \text{ i.e.,} \\ \forall \epsilon > 0 : \exists t_0 \geq 0 : \forall t > t_0 : \Big|\Big|x_{|\mathcal{V}'}(t)\Big|\Big| < \epsilon,$$

where 0 is the origin of $\mathbb{R}^{|\mathcal{V}'|}$. If a system is both, globally stable with respect to $\mathcal{V}'$ and globally attractive with respect to $\mathcal{V}'$, then it is called *globally asymptotically stable (GAS) with respect to* $\mathcal{V}'$.

Intuitively, GS is a boundedness condition, i.e. each trajectory starting δ-close to the origin will remain ϵ-close to the origin. GA ensures progress, i.e. for each ϵ-distance to the origin, there exists a point in time t_0 such that a trajectory always remains within this distance. By induction, it follows that every trajectory eventually approaches the origin. For a given hybrid system, this can be proven using Lyapunov Theory [6], which was originally restricted to continuous systems but has been lifted to hybrid systems.

We are handling polynomial hybrid systems. Thus, we assume that all guards, invariants, and flows are defined as expressions over polynomials. A *monomial* has the form $\prod_{v\in\mathcal{V}} v^{e_{v,j}}$ where all exponents $e_{v,j} \in \mathbb{N}$. A weighted sum of such monomials is called a *polynomial* $g : \mathbb{R}^{|\mathcal{V}|} \to \mathbb{R}$ and has the form $g(x) = \sum_j c_j \prod_{v\in\mathcal{V}} v^{e_{j,k}}$ where all $c_j \in \mathbb{R}$. Such a polynomial is called a *parameterized polynomial* if it has the form $f(x) = \sum_j c_j \rho_j \prod_{v\in\mathcal{V}} v^{e_{v,j}}$ where ρ_j is a *parameter* and $\rho_j \prod_{v\in\mathcal{V}} v^{e_{v,j}}$ is called a *parameterized monomial*. Indeed, the parameter in a parameterized monomial is rather optional, but to increase readability and shorten the formulas, we assume every summand in a parameterized polynomial to have a parameter where a "dummy" parameter might also represent a constant 1.

Theorem 1: **Discontinuous Lyapunov Functions for a Subset of Variables [5].**
Let $\mathcal{H} = (\mathcal{V}, \mathcal{M}, \mathcal{T}, Flow, Inv)$ be a hybrid automaton and let $\mathcal{V}' \subseteq \mathcal{V}$ be the set of variables that are required to converge. If for each $m \in \mathcal{M}$, there exists a set of variables $\mathcal{V}_m$ with $\mathcal{V}' \subseteq \mathcal{V}_m \subseteq \mathcal{V}$ and a continuously differentiable function $V_m : \mathcal{S} \to \mathbb{R}$ such that

1) for each $m \in \mathcal{M}$, there exist two class K^∞ functions α and β such that

$$\forall x \in Inv(m) : \quad \alpha\Big(\Big|\Big|x_{|\mathcal{V}_m}\Big|\Big|\Big) \preccurlyeq V_m(x) \\ \wedge\, V_m(x) \preccurlyeq \beta\Big(\Big|\Big|x_{|\mathcal{V}_m}\Big|\Big|\Big),$$

2) for each $m \in \mathcal{M}$, there exists a class K^∞ function γ such that

$$\forall x \in Inv(m) : \dot{V}_m(x) \preccurlyeq -\gamma\Big(\Big|\Big|x_{|\mathcal{V}_m}\Big|\Big|\Big)$$

for each $\dot{V}_m(x) \in \Big\{\Big\langle \frac{dV_m(x)}{dx}\Big| f\Big\rangle \Big| f \in Flow(m)\Big\}$,

3) for each $(m_1, G, R, m_2) \in \mathcal{T}$,

$$\forall x \in G : V_{m_2}(R(x)) \preccurlyeq V_{m_1}(x),$$

then $\mathcal{H}$ is globally asymptotically stable with respect to $\mathcal{V}'$ and V_m is called a *Local Lyapunov Function (LLF)* of mode m.

In Theorem 1, "$\Big\langle \frac{dV(x)}{dx}\Big| f\Big\rangle$" denotes the inner product between the gradient of a Lyapunov function V and a flow function f. Furthermore, each constraint has the form $0 \preccurlyeq c(x)$ where "$\preccurlyeq$" is a positive semi-definiteness operator which requires that $c(x)$ is non-negative almost everywhere and $c(0) = 0$.

Definition 3: A **Constraint** is either called

- *unconditioned* iff it exhibits the form $0 \preccurlyeq f(x)$ or

- *conditioned* iff it exhibits the form $\forall x \in P : 0 \preccurlyeq f(x)$

where $f(x)$ is a parameterized polynomial, $P \subset \mathcal{S}$ is a subset of the continuous state space, and $\forall x \in S$ is called the condition.[1]

Definition 4: Using the **S-Procedure [7],** a conditioned constraint can be transformed into an unconditioned constraint. The S-Procedure restricts a constraint to some region by exploiting the fact that finding a solution for

$$\left(\sum_i a_i \cdot g_i(x)\right) + \left(\sum_i b_i \cdot h_i(x)\right) \preccurlyeq g(x)$$

with $a_i \geq 0$ implies

$$\left(\bigwedge_i 0 \preccurlyeq g_i(x)\right) \bigwedge \left(\bigwedge_i 0 = h_i(x)\right)$$
$$\Rightarrow 0 \preccurlyeq g(x).$$

Note that the parameters b_i of the equality conditions $0 = h_i(x)$ are not required to be non-negative.

To find a set of Lyapunov functions for a hybrid system, one also needs to supply the solver with *templates*, which are parameterized polynomials, which usually contain all monomials up to some certain degree constructible as combinations of variables. The following steps are performed using the templates: Generate the constraint system described by Theorem 1, then use the S-Procedure to cast the conditioned constraints into unconditioned ones, and then try to solve[2] them. Upon success, we have a certificate of stability. Note that one cannot conclude non-stability from failing to solve the constraints since the described method is sound but incomplete. Higher degree templates could still render the problem solvable.

[1]We assume the conditions to be expressed as a conjunction of equalities and inequalities over polynomials, i.e. $\bigwedge_i g(x) \triangle 0$ where $g(x)$ is a polynomial and $\triangle$ is a relation with $\triangle \in \{=, \geq, >\}$.

[2]Further steps are required like translating the polynomial constraints to linear matrix inequalities (LMIs) which, in turn, can be solved using solvers for positive semi-definite problems. This type of translation can be done by using the Sums-of-Squares decomposition [8].

III. Simplifying Constraint Systems

In this section, we give the concrete problem statement and then describe the heuristic used to simplify the constraint systems.

For a constraint $0 \preccurlyeq f(x)$, a constraint $0 \preccurlyeq g(x)$ is called a *matching constraint* iff $0 \preccurlyeq f(x) \wedge 0 \preccurlyeq g(x)$ implies that $0 = f(x) = g(x)$. Note the special case $g(x) = -f(x)$.

Problem:
Given a set of conditioned constraints $C_1, \ldots, C_n$, each of the form $C_i = (P, f_i(x), g_i(x))$. We want to find a conditioned constraint $C_k = (P', f_k(x), g_k(x))$ such that $\exists d_1, \ldots, d_n : \sum_i d_i \cdot f_i(x) = -f_k(x)$ with $d_i \geq 0$.

Roughly speaking, we are searching for conditioned constraints for which one can obtain a matching constraint via a conic combination of the other constraints. In theory, the factors d_i are not restricted to scalar factors and can be arbitrary positive semi-definite monomials since it is a well-known fact that the result of a product of positive semi-definite functions is again positive semi-definite. This allows us to find the implicit equality in the following example:

Example 1:
Given the following system of conditioned constraints:

$$\forall x \in P : \rho_1 x^2 \preccurlyeq \rho_2 x^2 \quad (1)$$
$$\forall x \in P : \rho_2 x^4 \preccurlyeq \rho_1 x^4 \quad (2)$$

by multiplying the inequality in Constraint 1 with x^2, we obtain

$$\forall x \in P : \rho_1 x^4 \preccurlyeq \rho_2 x^4 \wedge \rho_2 x^4 \preccurlyeq \rho_1 x^4$$

from which we conclude that

$$\forall x \in P : \rho_1 x^4 = \rho_2 x^4.$$

However, we restrict the factors to scalars since implicit equality constraints are mainly caused by the constraints of Type 3 in Theorem 1. Since these constraints are relating the different templates only, a free parameter (ρ) usually occurs only in a single parameterized monomial, so a situation such as

in Example 1 does not need to be handled. Furthermore, due to this restriction, a simple linear program (LP) can be used to search for implicit equality constraints which we show how to obtain in the following.

We start with a system of conditioned constraints obtained by generating all constraints required by Theorem 1. Then, we group conditioned constraints by their condition, thereby obtaining a finite number of groups of conditioned constraints. Each such group has the form:

$$\begin{aligned}\forall x \in P :& \\ & 0 \preccurlyeq f_1(x) = \sum_j c_{(j,1)} \rho_{(j,1)} \prod_{v \in \mathcal{V}} v^{e_{(v,j,1)}} \\ \wedge \quad & \dots \\ \wedge\ & 0 \preccurlyeq f_n(x) = \sum_j c_{(j,n)} \rho_{(j,n)} \prod_{v \in \mathcal{V}} v^{e_{(v,j,n)}}.\end{aligned}$$

Here, a $c_{j,k}$ is the j-th coefficient of the k-th conditioned constraint belonging to the j-th parameterized monomial of the same constraint.

In each group, we now search for a conditioned constraints C_k for which we can syntactically construct a matching constraint as a conic combination of the other constraints:

$$\exists d_1, \dots, d_n \geq 0 : \sum_i d_i f_i(x) = -f_k(x)$$

or equivalently:

$$\exists d_1, \dots, d_n \geq 0 : \sum_i d_i f_i(x) + f_k(x) = 0$$

By syntactically replacing all parameterized monomials $\rho_{(j,i)} \prod_{v \in \mathcal{V}} v^{e_{(v,j,i)}}$ with a new symbol $z_{(j,i)}$, we have:

$$\begin{aligned}\exists d_1, \dots, d_n \geq 0 : & \sum_i d_i \sum_j c_{(j,i)} z_{(j,i)} \\ & + \sum_j c_{(j,k)} z_{(j,k)} = 0.\end{aligned}$$

Reordering the m syntactical different monomials $z_1, \dots, z_m$ leads to:

$$\begin{aligned}\exists d_1, \dots, d_n \geq 0 : & \sum_i d_i \sum_j^m c_{(j,i)} z_j \\ & + \sum_j^m c_{(j,k)} z_j = 0.\end{aligned}$$

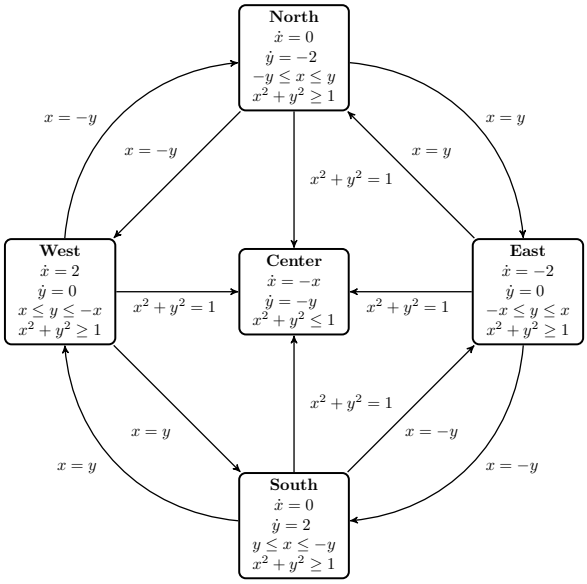

Fig. 1. Hybrid system describing a robot automatically approaching a certain region **`Center`**

By grouping the monomials z_j, we obtain

$$\begin{aligned}\exists d_1, \dots, d_n \geq 0 : & \\ & \sum_j^m \left(\left(\sum_i d_i c_{(j,i)} \right) + c_{(j,k)} \right) z_j = 0.\end{aligned}$$

Now, observe that a solution for the LP

$$\forall 1 \leq j \leq m : \sum_i d_i c_{(j,i)} = -c_{(j,k)}$$

with $d_i \geq 0$ yields a valid solution for the above constraint system and

$$\forall x \in P : 0 \preccurlyeq \sum_i d_i f_i(x)$$

is a constructible matching constraint for the k-th constraint. Note, that in order to ease finding a solution for the LP, the parameter d_k should be set to 0, a priori. Also, note that if it is known that the templates are not scaled, i.e. the constraints are not normalized, the problem can even be reduced to a binary program where $d_i \in \{0, 1\}$.

Example 2:
Figure 1 shows a hybrid automaton modeling a simple robot whose goal is to reach a certain target region **`Center`** independent of the starting point (somewhere in **`North`**, **`East`**, **`South`**, or **`West`**) with a maximal velocity of $2km/h$. Whenever it is not yet close to the **`Center`**, it drives full throttle in the cardinal direction of the center (thus its direction depends only on the quadrant the robot is in). Being close (less than one

kilometer) to the target, the robot's strategy changes and it follows a radio signal directing it directly towards the target. The vector field is visualized in Figure 2.

Generating the constraint system for this hybrid automaton leads to three constraints per mode and one constraint per transition – thus 27 constraints in total. Here we focus only on the constraints generated for the transitions which are:

$$x = y \Rightarrow V_N \preccurlyeq V_E \tag{3}$$
$$x = y \Rightarrow V_E \preccurlyeq V_N \tag{4}$$
$$x = y \Rightarrow V_S \preccurlyeq V_W \tag{5}$$
$$x = y \Rightarrow V_W \preccurlyeq V_S \tag{6}$$
$$x = -y \Rightarrow V_N \preccurlyeq V_W \tag{7}$$
$$x = -y \Rightarrow V_W \preccurlyeq V_N \tag{8}$$
$$x = -y \Rightarrow V_E \preccurlyeq V_S \tag{9}$$
$$x = -y \Rightarrow V_S \preccurlyeq V_E \tag{10}$$
$$x^2 + y^2 = 1 \Rightarrow V_C \preccurlyeq V_N \tag{11}$$
$$x^2 + y^2 = 1 \Rightarrow V_C \preccurlyeq V_E \tag{12}$$
$$x^2 + y^2 = 1 \Rightarrow V_C \preccurlyeq V_S \tag{13}$$
$$x^2 + y^2 = 1 \Rightarrow V_C \preccurlyeq V_W \tag{14}$$

where V_C, V_N, V_E, V_S, V_W are the templates for the Lyapunov function for the individual modes **`Center`**, **`North`**, **`East`**, **`South`**, and **`West`**, respectively. By applying the procedure described above, we derive that Constraint 3 and Constraint 4, Constraint 5 and Constraint 6, Constraint 7 and Constraint 8, and Constraint 9 and Constraint 10 are pairwise matching constraints. In the textual presentation above, this can be easily seen, but matching constraints can be much harder to detect in practice, since the templates are not given in this explicit form but instead are given as parameterized polynomials. The templates might be $V_i = \rho_{(i,1)}x^2 + \rho_{(i,2)}xy + \rho_{(i,3)}y^2 + \rho_{(i,4)}x + \rho_{(i,5)}y + \rho_{(i,6)}$ for $i \in \{N, E, S, W, C\}$, which serve as good candidates in this example. However we conclude that

$$x = y \Rightarrow V_N = V_E, \tag{15}$$
$$x = -y \Rightarrow V_N = V_W, \tag{16}$$
$$x = -y \Rightarrow V_E = V_S, \tag{17}$$
$$x = y \Rightarrow V_S = V_W. \tag{18}$$

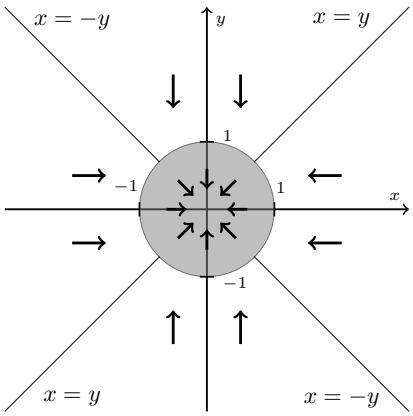

Fig. 2. Visualization of the vector field of the movement of the robot defined in Figure 1

Thus in case of Equality 15 together with the condition, we have

$$\begin{aligned} x = y \Rightarrow\ & (\rho_{(N,1)} - \rho_{(E,1)})x^2 \\ & + (\rho_{(N,2)} - \rho_{(E,2)})xy \\ & + (\rho_{(N,3)} - \rho_{(E,3)})y^2 \\ & + (\rho_{(N,4)} - \rho_{(E,4)})x \\ & + (\rho_{(N,5)} - \rho_{(E,5)})y \\ & + (\rho_{(N,6)} - \rho_{(E,6)}) = 0. \end{aligned}$$

By eliminating the condition, we get the unconditioned constraint

$$\begin{aligned} & (\rho_{(N,1)} + \rho_{(N,2)} + \rho_{(N,3)} \\ & \quad - \rho_{(E,1)} + \rho_{(E,2)} + \rho_{(E,3)})y^2 \\ & + (\rho_{(N,4)} + \rho_{(N,5)} - \rho_{(E,4)} + \rho_{(E,5)})y \\ & + (\rho_{(N,6)} - \rho_{(E,6)}) = 0 \end{aligned}$$

in which we can now isolate one of the parameterized monomials and use it to substitute it in the other constraints. Here, a promising candidate is to replace $\rho_{(E,6)}$ with $(\rho_{(N,1)} + \rho_{(N,2)} + \rho_{(N,3)} - \rho_{(E,1)} + \rho_{(E,2)} + \rho_{(E,3)})y^2 + (\rho_{(N,4)} + \rho_{(N,5)} - \rho_{(E,4)} + \rho_{(E,5)})y + \rho_{(N,6)}$. By doing so, we reduce the number of free parameters by one and the number of constraints by two. We can repeat the procedure with the other three implicit Equalities 16-18 and remove six more constraints and three more parameters.

Example 3:
The third example given in Figure 3 does not describe a system for a particular purpose. But it shows that implicit equality constraints

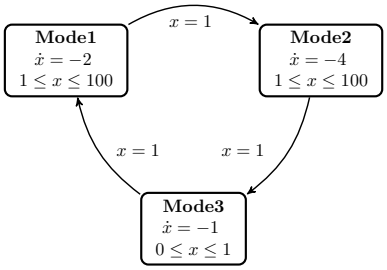

Fig. 3. Hybrid system showing that implicit equality constraints might involve more than two transitions

can involve more than two constraints. Here, the transitions would require:

$$x = 1 \Rightarrow V_2 \preccurlyeq V_1 \tag{19}$$

$$x = 1 \Rightarrow V_3 \preccurlyeq V_2 \tag{20}$$

$$x = 1 \Rightarrow V_1 \preccurlyeq V_3 \tag{21}$$

where V_1, V_2, and V_3 are the Lyapunov function templates for **`Mode1`**, **`Mode2`**, and **`Mode3`**, respectively. And by combination of the constraints, we can conclude:

$$x = 1 \Rightarrow V_1 = V_2 = V_3 \tag{22}$$

which allows us to remove three constraints and two free parameters from the constraint system.

For all three examples, Lyapunov functions can only be found if (a) no gaps are used to strengthen the constraints which, then, risks that the values returned by a solver do not form a valid Lyapunov function or (b) implicit equality constraints are detected and resolved which, thereby, discards the need for gaps on these constraints.

IV. Conclusion

We have shown a simple but yet powerful heuristic that detects implicit equality constraints. Our heuristic is sound but incomplete and searches for equality constraints using linear programming. In our case, the constraints that we have to consider, do have a special shape. Thanks to this shape; the heuristic is sufficient in many cases and especially in the settings of finding Lyapunov functions. After applying the heuristic we can use the gained knowledge, i.e., the detected equality constraints, to replace free parameters in the constraint system which not only reduces the number of free parameters but also the number of constraints. We have presented three examples where the search for Lyapunov functions certifying stability of these hybrid systems would fail because of additional gaps that are required to make the constraint system robust against numerical issues. But by using our heuristic, it is easy to find Lyapunov functions for two of them. Future work will include to generate quadratic monomials and the inclusion of the products between the constraints and these monomials in the linear problem. This will make the heuristic applicable to a more general case.

References

[1] B. Borchers, "CSDP, a C Library for Semidefinite Programming." *Optimization Methods and Software*, vol. 10, pp. 613–623, 1999.

[2] K. Fujisawa, K. Nakata, M. Yamashita, and M. Fukuda, "SDPA Project : Solving Large-Scale Semidefinite Programs," *Journal of the Operations Research Society of Japan*, vol. 50, no. 4, pp. 278–298, Dec. 2007.

[3] E. Möhlmann and O. E. Theel, "Stabhyli: a Tool for Automatic Stability Verification of Non-Linear Hybrid Systems," in *Hybrid Systems: Computation and Control*, C. Belta and F. Ivancic, Eds. ACM, 2013, pp. 107–112.

[4] L. Li, K.-D. He, M. Gu, and X.-Y. Song, "Equality Detection for Linear Arithmetic Constraints," *Journal of Zhejiang University SCIENCE A*, vol. 10, pp. 1784–1789, 2009.

[5] J. Oehlerking, "Decomposition of Stability Proofs for Hybrid Systems," Ph.D. dissertation, Carl von Ossietzky University of Oldenburg, Department of Computer Science, Oldenburg, Germany, 2011.

[6] M. Lyapunov, "Problème général de la stabilité du movement," in *Ann. Fac. Sci. Toulouse, 9*. Université Paul Sabatier, 1907, pp. 203–474.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, Mar. 2004.

[8] S. Prajna and A. Papachristodoulou, "Analysis of Switched and Hybrid Systems - Beyond Piecewise Quadratic Methods," *Proceedings of the ACC*, 2003.