Symbolic Domain Predictive Control

Johannes Löhr
University of Freiburg
Germany
loehr@tf.uni-freiburg.de

Martin Wehrle
University of Basel
Switzerland
martin.wehrle@unibas.ch

Maria Fox
King's College London
United Kingdom
maria.fox@kcl.ac.uk

Bernhard Nebel
University of Freiburg
Germany
nebel@tf.uni-freiburg.de

Abstract

Planning-based methods to guide switched hybrid systems from an initial state into a desired goal region opens an interesting field for control. The idea of the Domain Predictive Control (DPC) approach is to generate input signals affecting both the numerical states and the modes of the system by stringing together atomic actions to a logically consistent plan. However, the existing DPC approach is restricted in the sense that a discrete and pre-defined input signal is required for each action. In this paper, we extend the approach to deal with symbolic states. This allows for the propagation of reachable regions of the state space emerging from actions with inputs that can be arbitrarily chosen within specified input bounds. This symbolic extension enables the applicability of DPC to systems with bounded inputs sets and increases its robustness due to the implicitly reduced search space. Moreover, precise numeric goal states instead of goal regions become reachable.

Introduction

Dynamic systems are usually models derived from real world physics based on differential equations. Guiding such systems from an initial state into a desired goal state becomes difficult if the system can additionally switch between several *modes* containing e.g. sensors of different accuracies, actuators of different capabilities and controllers affecting the system dynamics in different ways. This class of systems is often referred to as switched dynamic systems (Liberzon 2003). Particularly challenging is the case of logical dependencies between the modes of a switched dynamic system.

A planning based method to control this kind of system is Domain Predictive Control (DPC) introduced by Löhr et al. (2012). It provides a domain-independent planning framework for modelling and solving planning problems derived from switched hybrid systems. Therefore, it deals with a set of actions called *domain model* each containing time discrete linear differential equations of each mode of the hybrid system. Heuristic search methods (Pearl 1984) are used to explore reachable states along minimal future cost estimates with the purpose to find a state satisfying the goal conditions. Preconditions and logical effects of actions ensure logically consistent state transitions. The resulting plan contains a sequence of actions connecting the initial state with the goal state and provides both a control signal and a mode switching strategy.

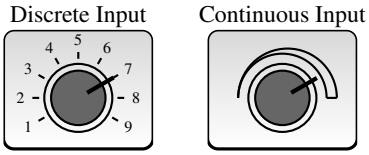

Figure 1: Modelling e.g. heaters, gas pedals or valves as parameters of actions leads to more realistic domains.

Although DPC has shown to find effective control strategies for switched hybrid systems in various domains (Löhr et al. 2012; 2013), there are two significant drawbacks which limit DPC's practical applicability: First, actions in the domain model rely on *input signals* of the system that have to be pre-specified for each action when setting up the domain model. While this is reasonable for actuators that can be switched between different inputs, actuators often provide a bounded continuous input spectrum as shown in Figure 1. In DPC, such bounded inputs are modelled by a sampling workaround, yielding a coarse approximation of the system's capabilities. Second, the desired goal state cannot be reached precisely, but is considered as reached if a state is found that is reasonably close. Both of these drawbacks stem from the discrete handling of input signals.

In this paper, we generalize DPC to deal with continuous bounded input signals and symbolic goal recognition. Overall, our approach connects planning, control theory and reachability analysis (Le Guernic 2009).

Scope

In this section we recall Domain Predictive Control and emphasize the need for more expressive domain models.

Domain Predictive Control

DPC propagates reachable states using time-discretized dynamics of continuous systems modelled as actions. A state $s = \langle \mathbf{x}, \mathbf{l} \rangle$ consists of a n dimensional numerical state vector $\mathbf{x} \in \mathbb{R}^n$ and a vector $\mathbf{l}$ containing the logical variables. Each

action $a_i = \langle P, E \rangle$ has the numerical effect E mapping the numerical state $\mathbf{x}$ to its successor state $\mathbf{x}' \in \mathbb{R}^n$ by

$$\mathtt{E}: \quad \mathbf{x}' = \Phi_i \mathbf{x} + \Psi_i \tag{1}$$

which corresponds to a linear transformation using the state transition matrix $\Phi_i \in \mathbb{R}^{n \times n}$ and the addition of an external input vector $\Psi_i \in \mathbb{R}^n$. The action can be applied to a state, if the preconditions P are fulfilled. Beside the numerical effect each action may have additional logical effects. For any linear time-invariant dynamical system $\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}$ (Kailath 1980) state transition matrices (Moler and van Loan 1978; 2003) can be obtained by

$$\Phi_i = e^{A_i \ \delta_i}, \tag{2}$$

where $A_i \in \mathbb{R}^{n \times n}$ corresponds to an arbitrary *dynamic matrix* and δ_i is the *duration* of an action. The dynamics is possibly influenced by feedback control in different modes. Therefore we can switch between several modes denoted by i from action to action. The effect of external inputs (e.g. forces acting on the system) over the duration δ_i is given by

$$\Psi_i = \int_0^{\delta_i} e^{A_i \ (\delta_i - \tau)} B_i \ \mathbf{u}_i \ d\tau. \tag{3}$$

where $\mathbf{u}_i \in \mathbb{R}^m$ is the *input vector* which has to be predefined over the duration δ_i for each action a_i. The state transition matrix Φ_i and the discrete input Ψ_i are computed in a preprocessing step in order to propagate the states quickly using Equation 1.

Need for Effects on Sets of States

The main drawback of the current approach lies in Equation 3. Since the input signal $\mathbf{u}$ has to be pre-defined when setting up the domain model it is difficult to model inputs which can arbitrarily be chosen from a closed input set $\mathcal{U}$. A common workaround is to sample the input set $\mathbf{u}_j \in \mathcal{U}$ by $j \in \{1, ..., z\}$ input signals covering only a subset of z elements, see Figure 1. Furthermore, each sampled input is modelled as a separate action which leads to a heavily increased branching factor of the domain. To avoid the blow-up of the search space and to cope with the continuity of the input set we want do deal with bounded input sets as parameters of actions. This increases the applicability of Domain Predictive Control mainly due to two reasons:

- Real world actuators can provide an input spectrum ranging from a minimal input to a maximal input and it is the task of the control algorithm to choose a suitable input signal to bring the system into the desired goal state. Ideally, this should not be artificially restricted in advance when setting up the domain model.
- Reachability is another important aspect. In the previous DPC approach the goal is said to be satisfied if a planned state is "close enough" to the desired setpoint. The search can fail in cases of small target regions or coarse discretizations of the dynamics. With the propagation of symbolic states a plan is found if the target setpoint is part of the propagated reachable set of states. We will show later that we can deduce an explicit input signal from this plan which leads the system from the initial state to the goal state precisely.

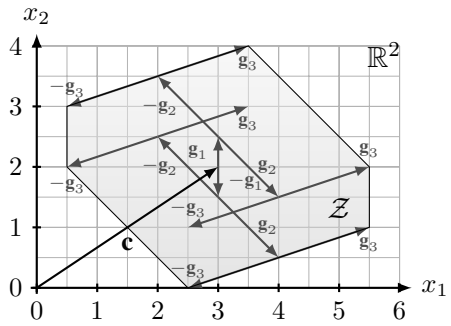

Figure 2: Illustration of a zonotope $\mathcal{Z}$ with three generators.

It is worth to mention that there are many possibilities to describe closed sets of a vector space e.g. using boxes, ellipsoids, or support functions. A comparison between set representations is given in Le Guernic (2009). In the next section we identify zonotopes (Grunbaum et al. 1967) as a suitable representation for the symbolic approach.

Methods

In this section we introduce methods known from reachability analysis to cope with the new class of action effects E that deal with symbolic states.

Zonotopes

Zonotopes $\mathcal{Z} = \langle \mathbf{c}, G \rangle$ are a sub-class of polytopes which consist of a center vector $\mathbf{c} \in \mathbb{R}^n$ and a set of k *generators* $\mathbf{g}_j$ which are also part of the n dimensional vector space and stored as column vectors in matrix G. It contains all points in the vector space reachable by the linear system of equations

$$\mathcal{Z} = \left\{ \mathbf{c} + \sum_{j=0}^{k} \alpha_j \ \mathbf{g}_j : \forall \alpha_j \in [-1, 1] \right\}, \tag{4}$$

as depicted in Figure 2. Visually the vertices of the set correspond to the convex hull of the points generated by adding and subtracting consecutively all generators $\mathbf{g}_j \in G$ to the ends of the emerging vector chains starting with the center vector $\mathbf{c}$. To avoid confusion we indicate the membership of the center vector or generators to a certain zonotope by $\mathcal{Z}.\mathbf{c}$ and $\mathcal{Z}.\mathbf{g}$ respectively when dealing with several zonotpes.

Linear Transformations of Zonotopes

Zonotopes can be linearly transformed by a state transition matrix Φ obtained from a dynamic system as described in Equation 2. It is defined as a linear transformation of all elements $\mathcal{X}' = \left\{ \Phi \mathbf{x} : \mathbf{x} \in \mathcal{X} \right\}$. The successor zonotope is obtained by the linear transformation of the center vector and all k generators of the zonotope describing $\mathcal{X}$, i.e.

$$\mathcal{X}.\mathbf{c}' = \Phi \mathcal{X}.\mathbf{c}, \text{ and } \mathcal{X}.\mathbf{g}' = \Phi \mathcal{X}.\mathbf{g} \quad \forall \mathbf{g} \in G.$$

This way the homogeneous evolution of sets of states without external input can be described. The homogeneous evolution of an initial set of states described by a zonotope is shown in Figure 3 for a simple mechanical system.

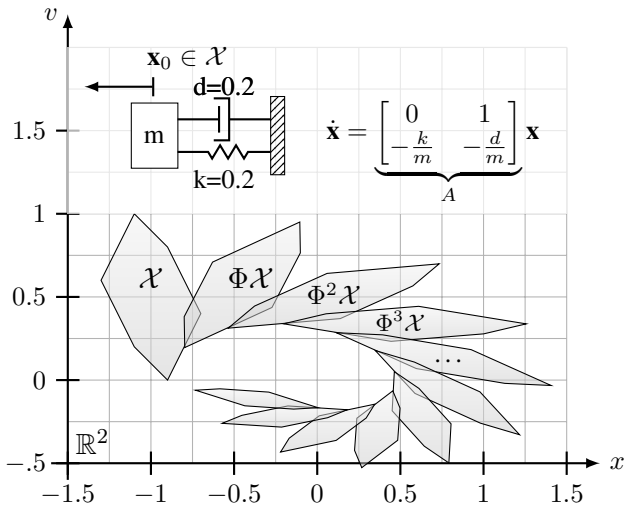

Figure 3: Successive linear transformation by $\Phi = e^{A\delta}$ of a set of initial states modelled as zonotope $\mathcal{X}$ to their successors after $\delta = 1$ seconds.

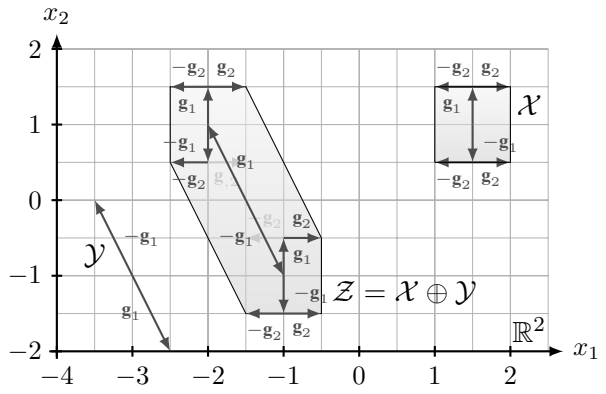

Figure 4: Illustration of the Minkowski sum $\mathcal{Z} = \mathcal{X} \oplus \mathcal{Y}$

Minkowski Sum

The Minkowski sum $\mathcal{X} \oplus \mathcal{Y} = \left\{ \mathbf{x}+\mathbf{y} : \mathbf{x} \in \mathcal{X} \text{ and } \mathbf{y} \in \mathcal{Y} \right\}$ is a new set which contains the sum of all elements contained in $\mathcal{X}$ and $\mathcal{Y}$.

Adding zonotopes under the Minkowski sum yields zonotopes again which is a clear advantage to other set representations like e.g. ellipsoids. The Minkowski sum $\mathcal{Z} = \mathcal{X} \oplus \mathcal{Y}$ can be obtained by

$$\mathcal{Z}.\mathbf{c} = \mathcal{X}.\mathbf{c} + \mathcal{Y}.\mathbf{c}, \text{ and } \mathcal{Z}.G = \{\mathcal{X}.G, \mathcal{Y}.G\}$$

with a new set of generators $\mathcal{Z}.G$ containing all generators from $\mathcal{X}.G$ and $\mathcal{Y}.G$, as illustrated in Figure 4.

Planning with Reachable Sets

From the methods section we have all ingredients available to plan with symbolic sets. We define a symbolic state as a tuple $s = \langle \mathbf{l}, \mathcal{X} \rangle$, which consists of a state variables vector $\mathbf{l}$ containing logical variables and a set of state variable vectors $\mathcal{X}$ each containing numerical variables.

Planning Domain

In this section, we model the planning domain expressing the capabilities of the hybrid system. Let $\mathcal{X} \subset \mathrm{R}^n$ be a zonotope describing a symbolic set of state variable vectors each consisting of n numeric variables. The closed input set of an actuator with a minimum possible actuation $\mathbf{u}_{min}$ and a maximum possible actuation $\mathbf{u}_{max}$ is symbolically described as zonotope $\mathcal{U}$ with

$$\mathcal{U}.\mathbf{c} = \frac{1}{2}(\mathbf{u}_{min} + \mathbf{u}_{max}), \text{ and } \mathcal{U}.\mathbf{g} = \frac{1}{2}(\mathbf{u}_{max} - \mathbf{u}_{min}),$$

where $\mathcal{U}.\mathbf{c}$ is the center vector and $\mathcal{U}.\mathbf{g}$ is the generator. If the system can use more actuators at once the feasible inputs can be chosen from a combined input set generated by the Minkowski sum of the respective zonotopes. The actuation of such a continuous actuator for a duration δ_i shall be modelled as action a_i. A finite set of such actions $\mathcal{O}$ describes the *domain model*. We define the effect E of action a_i on the symbolic state $\mathcal{X}$ by

$$E: \quad \mathcal{X}' = \Phi_i \mathcal{X} \oplus \mathcal{U}_{d,i}, \tag{5}$$

where $\mathcal{X}'$ is the closed set of successor states which are reachable from $\mathcal{X}$. The linear transformation corresponds to the natural evolution of the symbolic state over the duration of one action while the Minkowski sum describes the additionally reachable set of states if an actuator is available for this time. The state transition matrix Φ_i can be obtained by Equation 2 and $U_{d,i}$ is the reachable set driven by an arbitrary input $\mathbf{u}$ chosen from $\mathcal{U}_i$. Due to the linearity of the integral the entities of $\mathcal{U}_{d,i}$ can be obtained from $\mathcal{U}_i$ by

$$\mathcal{U}_{d,i}.\mathbf{c} = \int_0^{\delta_i} e^{A_i(\delta_i - \tau)} B_i \, \mathcal{U}_i.\mathbf{c} \, d\tau \tag{6}$$

$$\mathcal{U}_{d,i}.\mathbf{g} = \int_0^{\delta_i} e^{A_i(\delta_i - \tau)} B_i \, \mathcal{U}_i.\mathbf{g} \, d\tau, \quad \forall \mathbf{g} \in G.$$

While the Equations 2 and 6 can be calculated in a preprocessing step when setting up the domain, effects as in Equation 5 can be calculated quickly during planning since the linear transformation and the Minkowski sum are simple mathematical operations on zonotopes.

Planning Problem

The planning problem is given by an initial state s_0 and a goal state $s_\star = \langle \mathbf{l}_\star, \mathbf{x}_\star \rangle$. The logical part $\mathbf{l}_\star$ can be partially defined over the logical variables while the numerical part $\mathbf{x}_\star$ of the goal state is an assignment over the numeric variables.

A *trajectory* $\mathcal{S} = \langle s_0, s_1, \ldots, s_z \rangle$ connects the initial state with the goal state such that each $1 \leq i \leq z$ intermediately generated state s_i can be generated out of its predecessor state s_{i-1} by applying an action $a \in \mathcal{O}$. The trajectory is *valid* if all the conditions of all involved actions are satisfied at their appropriate time. The sequence of actions generating a valid trajectory is called a *plan* if s_z complies with the goal

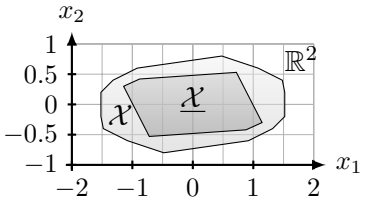

Figure 5: Zonotope $\mathcal{X}$ and the subset $\underline{\mathcal{X}}$ reachable by using pseudo inverse to obtain the generator coefficients $\boldsymbol{\alpha}$.

specification $s_\star$. This means that for the logical state $\mathbf{l}_z$ is part of the partial state $\mathbf{l}_\star$ and for the numerical part that the propagated symbolic states $\mathcal{X}_z$ contains $\mathbf{x}_\star$.

Checking the Goal

A key question is how we can check whether a propagated set contains the goal state. Using the set definition from Equation 4, we let $G = [\mathbf{g}_1, ..., \mathbf{g}_k] \in \mathbb{R}^{n \times k}$ be a matrix containing all k generators as column vectors and define $\boldsymbol{\alpha}$ as a vector of k generator coefficients. We want to check the set $\mathcal{X} = \langle \mathbf{c}, G \rangle$ for containing the goal state. Since

$$\mathbf{x}_\star = \mathbf{c} + G\,\boldsymbol{\alpha}, \tag{7}$$

the goal $\mathbf{x}_\star$ is part of $\mathcal{X}$ if there exists a solution for $\boldsymbol{\alpha}$ under the constraint $\alpha_j \in [-1, 1]$ for all $j \in \{1, \ldots, k\}$.

We use a straight forward approach utilizing the *pseudo inverse*[1] G^+ which fulfils $G^+G = I$, where I is the unity matrix. From Equation 7 we get

$$\boldsymbol{\alpha} = G^+(\mathbf{x}_\star - \mathbf{c}) \tag{8}$$

and $\mathbf{x}$ is part of the set if $||\boldsymbol{\alpha}||_\infty \leq 1$.

The use of the pseudo inverse needs G to be of full rank[2]. If $k = n$ the pseudo inverse becomes the regular inverse and $\boldsymbol{\alpha}$ can be determined exactly. If $k > n$ there is an infinite number of possible solutions for $\boldsymbol{\alpha}$ and the pseudo inverse returns the solution with minimal Euclidean norm. We induce conservatism by saying that the minimum Euclidean norm of $\boldsymbol{\alpha}$ obtained by the pseudo inverse is also an acceptable solution regarding the minimum infinity norm. The result is that $||\boldsymbol{\alpha}||_2 \leq 1$ only triggers if $\mathbf{x} \in \underline{\mathcal{X}} \subseteq \mathcal{X}$ as depicted in Figure 5.

Interpreting the Plan

If we have found a plan with trajectory $\mathcal{S} = \langle s_0, s_1, \ldots, s_z \rangle$ then $\mathbf{x}_\star \in \underline{\mathcal{X}}_z$ and therefore also $\mathbf{x}_\star \in \mathcal{X}_z$. This means only that the numerical goal is reachable in principle. To actually control the hybrid system into the goal state we have to determine the concrete input signal $\mathbf{u}_i \in \mathcal{U}_i$ which has to be applied for each action during the plan execution such that the numerical system state $\mathbf{x}$ equals $\mathbf{x}_\star$ after plan execution. The knowledge of $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1^T, \boldsymbol{\alpha}_2^T, ..., \boldsymbol{\alpha}_z^T)^T$ enables us to determine the input signal guiding the system into the goal by $\mathbf{u}_i = \mathcal{U}_i.\mathbf{c} + \mathcal{U}_i.G\boldsymbol{\alpha}_i \in \mathcal{U}$ which is valid during the execution of action a_i in the time interval $t \in [t_i, t_i + \delta_i]$.

Tracing of the numerical states visited during plan execution can be done by using the recursive function generating the successors $\mathbf{x}(t_i + \delta_i) = \mathbf{x}(t_i) + \mathcal{U}_{d,i}.\mathbf{c} + \mathcal{U}_{d,i}.G\boldsymbol{\alpha}_i$ from the previous numerical state, starting at the initial numerical state $\mathbf{x}(t_0)$ and ending at $\mathbf{x}(t_z) = \mathbf{x}_\star$.

Guiding the Search

In order to efficiently find goal states, we need a heuristic estimation of the effort to reach the numerical goal. The numerical system state is either a certain point in the state space $\mathbf{x}$ or a reachable set $\mathcal{X}$.

The DPC approach dealt with points in the state space and positive experience was made with heuristic functions that were based on a weighted error metric. Therefore we obtain the error $\mathbf{e} = \mathbf{x}^* - \mathcal{X}.\mathbf{c}$ and choose a weighted vector norm as heuristic value

$$h = ||W\mathbf{e}||_\circ, \tag{9}$$

where W is a weighting matrix and $\circ \in \{1, 2, \infty\}$ indicates the absolute norm, the Euclidian norm or the infinity norm of the weighted error. A weighting is very convenient for dynamic systems since not all values in $\mathbf{e}$ have the same importance. Via the weighting matrix W the errors can be scaled to an appropriate ratio.

What is the estimated cost to go for a set $\mathcal{X}$ with generators G? Visually it is the part of the error vector that is not covered by the zonotope. In the case that the zonotope contains no generators we can directly use Equation 9. If it also contains a set of generators G we use

$$h = ||W\mathbf{e}||_\circ - \frac{1}{||\boldsymbol{\alpha}||_\infty}||W\mathbf{e}||_\circ \tag{10}$$

as heuristic value. Note that the goal is reached if the term $\frac{1}{||\boldsymbol{\alpha}||_\infty}$ becomes larger than one. Thus the heuristic value stays positive during the search and becomes zero or negative if the goal was reached. The larger the distance between the predicted zonotope and the goal state, the larger is $||\boldsymbol{\alpha}||_\infty$ and the less is the heuristic value influenced by the generators of the zonotope.

Case Study

We demonstrate the symbolic DPC approach in a orbital manoeuvre domain similar to the domain used by Löhr et al. (2013). A satellite is ideally located at the origin of the *Local Vertical Local Horizontal frame* (LVLH) after orbit injection and separation. Deviations in position and velocity due to the separation from the launcher shall be compensated by the symbolic DPC approach. The LVLH origin moves with the orbital velocity around Earth while the V-axis points into flight direction and the R-axis points towards Earth (Fehse 2003), see Figure 6.

[1] The pseudo inverse G^+ is defined as PS^+Q^T, where P and Q stem from the singularity decomposition $G = QSP^T$ and $S^+ \in \mathbb{R}^{k \times n}$ is a diagonal matrix with the inverse singular values of G as diagonal elements $[S^+]_{ii}$.

[2] If $rank(G) < n$ the solution of Equation 8 using the pseudo inverse leads to an $\boldsymbol{\alpha}$ with least square result for $(\mathbf{x}_\star - \mathbf{c})$. One has to double check Equation 7 by inserting $\boldsymbol{\alpha}$ again to ensure that the goal was reached.

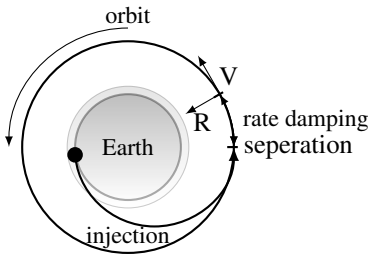

Figure 6: Injection of a satellite to a circular target orbit with subsequent separation and rate damping.

Planning Domain

We are considering a medium sized satellite of 1000 kg mass. The spacecraft is equipped with one orbit manoeuvre thruster which can provide significant force of 2.24 N up to 10 N but only in one direction of the spacecraft body frame, which means that the satellite has to be slewed before a force can be applied into a certain direction. The attitude control system can reach any attitude within 200 seconds. The orbital relative dynamics are linearized at the LVLH frame yielding the dynamic and input matrices

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 2(\frac{2\pi}{T_P}) \\ 0 & 3(\frac{2\pi}{T_P})^2 & -2(\frac{2\pi}{T_P}) & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{m} & 0 \\ 0 & \frac{1}{m} \end{bmatrix},$$

where T_P is the orbit period and m is the mass of the spacecraft. The system of ordinary differential equations are known as Hill equations (Hill 1878). We consider a circular orbit with period of 6000 seconds which corresponds to a orbit height of about 765 km. The state $\mathbf{x} = (x, y, v_x, v_y)^T$ contains the position and velocity components along the R-direction and the V-direction of the spacecraft.

Using thruster pulsing the provided force can be scaled within a continuous input spectrum from $\mathbf{u}_{min} = 2.24$ N to $\mathbf{u}_{max} = 10$ N. We allow the spacecraft to slew into attitudes where the force can be applied in flight or opposite to flight direction ($\pm V$-axis) and vertically towards nadir and zenith ($\pm R$-axis). This leads to four actions. The duration of each action is chosen as $\delta = 50$ seconds. The related state transition matrix Φ is obtained by Equation 2 and the discrete input set is obtained by Equation 6. Additionally we model two actions: a 50 seconds coast action where the thruster is switched off and a slew action of 200 seconds duration performed from the attitude control system. The corresponding state transition matrix for the slew action is $\Phi_{slew} = \Phi^4$ where also no translational thrust is applied. All actions including also the logical effects E_l and preconditions P are summarized in Table 1.

Planning Problem

The position and velocity after the separation and rate damping phase may vary significantly from the desired one at the origin of the LVLH frame $\mathbf{x}_\star = (0, 0, 0, 0)^T$. We randomly initialize a set of simulations with an initial position error $x_{1,2} \in [-500\text{ m}, 500\text{ m}]$ along V-bar and R-bar and a velocity error of $x_{3,4} \in [-1\ \frac{\text{m}}{\text{s}}, 1\ \frac{\text{m}}{\text{s}}]$ in both directions, see Table 2. The attitude after rate damping is known as $\texttt{pV}$.

action	P	E_n	E_l
a_1	-	$\mathcal{X}' = \Phi\mathcal{X}$	-
a_2	-	$\mathcal{X}' = \Phi_{slew}\mathcal{X}$	$\texttt{pV}, \texttt{mV}, \texttt{pR}, \texttt{mR}$
a_3	$\texttt{pV}$	$\mathcal{X}' = \Phi\mathcal{X} \oplus \mathcal{U}_{d,pV}$	$\neg\texttt{mV}, \neg\texttt{pR}, \neg\texttt{mR}$
a_4	$\texttt{mV}$	$\mathcal{X}' = \Phi\mathcal{X} \oplus \mathcal{U}_{d,mV}$	$\neg\texttt{pV}, \neg\texttt{pR}, \neg\texttt{mR}$
a_5	$\texttt{pR}$	$\mathcal{X}' = \Phi\mathcal{X} \oplus \mathcal{U}_{d,pR}$	$\neg\texttt{pV}, \neg\texttt{mV}, \neg\texttt{mR}$
a_6	$\texttt{mR}$	$\mathcal{X}' = \Phi\mathcal{X} \oplus \mathcal{U}_{d,mR}$	$\neg\texttt{pV}, \neg\texttt{mV}, \neg\texttt{pR}$

Table 1: Action set defining the domain model $\mathcal{O}$.

Simulation

The simulation part consists of three steps. First a symbolic reachability analysis is performed using the domain model and greedy search. Once we have found a plan we generate the continuous input signal. Finally we perform a numeric time simulation to obtain the continuous trajectory. The heuristic search is guided by Equation 10. Errors of the velocity are stronger weighted compared to the position errors by using the diagonal weighting matrix W with elements $W_{11} = 1, W_{22} = 1, W_{33} = 10, W_{44} = 10$ to obtain the heuristic value. It is important to point out that the weighting is domain dependent and subject to optimization. Analogous to the design of weighting matrices in optimal controller synthesis some engineering knowledge is needed to weight the states properly.

The results are shown in Figure 7. On the left hand side the trajectory of the numerical time simulation is shown for each planning problem. They start at the corresponding initial state (the initial velocity is marked by an arrow) and ends precisely at the goal state at the origin of the LVLH frame.

The input signal guiding the dynamic system into the goal state is shown on the right hand side of Figure 7. All objectives are fulfilled, since the input is not exceeding the actuator capacity of 10 N at all times. Moreover, the minimum input signal does not undercut 2.24 N corresponding to the minimum force of the thruster. If the direction of the input signal is changed, the attitude control system performs a slew manoeuvre with a predefined duration of 200 seconds where no force is applicable. Again, the plans found provide a control signal and a switching strategy that is compliant with the specifications modelled in the domain, here corresponding to different attitudes of the spacecraft.

The computational effort is shown in Table 3 in terms of explored nodes. Comparing breadth-first search to the greedy-search using the presented heuristics one can say that the heuristics work very effectively.

Related Work

The concept of *flow tubes* is a related idea for planning with continuous dynamics. It is used by Li and Williams (2008) to obtain an optimal trajectory within a hull of legal trajectories called hybrid flow graphs represented by cones for bounded velocity systems. The solution is obtained by solving mixed logic (non-) linear programs.

Hybrid planning as model checking (Della Penna, Magazzeni, and Mercorio 2012; Bogomolov et al. 2014) has been used to solve problems expressed in PDDL+ (Fox and Long

No	initial numerical state
P_1	$\mathbf{x}_0 = (-370.9106, 104.8375, 0.0079, 0.9009)^T$
P_2	$\mathbf{x}_0 = (-163.2896, -407.6480, -0.6806, 0.5617)^T$
P_3	$\mathbf{x}_0 = (192.5291, -163.7162, 0.8430, 0.4272)^T$
P_4	$\mathbf{x}_0 = (-50.7718, -321.6447, 0.2343, -0.3626)^T$
P_5	$\mathbf{x}_0 = (-39.6511, -416.6847, -0.4382, 0.4576)^T$
P_6	$\mathbf{x}_0 = (108.2207, -135.2274, 0.4653, 0.7224)^T$
P_7	$\mathbf{x}_0 = (433.3637, -456.3211, -0.7784, -0.3377)^T$

Table 2: Initial states defining one planning problem each.

2006). The work is also related to model checking for hybrid automata (Henzinger 2000) with the objective to show correctness by proving that error states are not reachable. Tool supported symbolic search for this purpose has be done by Frehse et al. (2011) in the SpaceEx project. The increasing complexity of the representing sets of states are tackled with tight over-approximations which avoids a growing error known as wrapping effect (Kühn 1999).

The idea to *plan* with control inputs based on the future state evolution of dynamic systems is borrowed from model predictive control methods (Wang 2009) where errors and control efforts are optimized over a finite time horizon. Polytopes are also utilized by Rubagotti, Trimboli, and Bemporad (2013) to model additive uncertainties and by Trimboli, Rubagotti, and Bemporad (2011) to cope with parametric uncertainties in order to deduce stability properties of piecewise affine hybrid systems by analysing the bounds of the emerging reachable sets.

Conclusion

We presented a symbolic extension to the Domain Predictive Control framework that enables the modelling of input signals as a numeric parameter of actions. We use set representations from reachability analysis to handle the resulting symbolic analysis of the reachable states. We successfully guide the search based on weighted error norms and pseudo inverse, while breadth-first search does not provide results in reasonable time. Current work involves the development of admissible heuristics for optimal search.

While earlier work on DPC was expressible in standard planning languages (Fox and Long 2003), this is difficult for the symbolic extension, since the complexity of the symbolic state expressed by zonotopes increases with the plan length. Tight under-approximations of the zonotopes as used by Girard, Le Guernic, and Maler (2006) could become important to keep the complexity of the set representation constant. An additional drawback is the requirement of an obstacle-free numeric state space for the symbolic approach since the calculation of intersections of zonotopes is known to be a crucial operation.

However, important issues of earlier work by Löhr et al. (2012) on DPC could efficiently be tackled by the presented symbolic approach. First, the precise goal reachability could be achieved. Second, oversized domains and the corresponding oversized search space can be avoided by expressing bounded input-sets as parameters of actions. Both is indispensable to provide input signals in reasonable time.

No	breadth-first	$\|\|W\mathbf{e}\|\|_2$	$\|\|W\mathbf{e}\|\|_1$	$\|\|W\mathbf{e}\|\|_\infty$
P_1	404647 (11)	290 (16)	**250** (16)	509 (12)
P_2	37759 (9)	**92** (11)	4162 (23)	251 (11)
P_3	464666 (11)	202 (16)	282 (19)	**130** (12)
P_4	1623085 (12)	598 (18)	**348** (18)	416 (18)
P_5	8146 (8)	**54** (9)	298 (21)	69 (9)
P_6	1237236 (12)	**989** (24)	1093 (25)	1626 (22)
P_7	48069 (9)	**63** (16)	1841 (19)	150 (19)

Table 3: Explored nodes for breadth-first and different error norm heuristics using greedy-search. Quickest results bold. The respective plan length is given within the brackets.

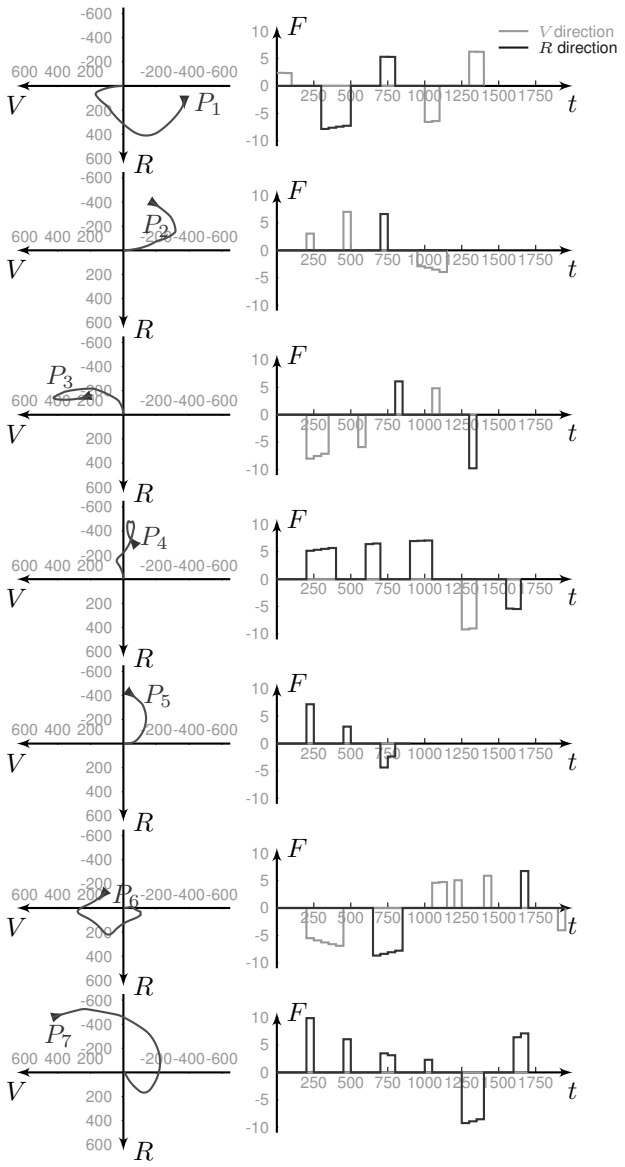

Figure 7: Simulation results. The trajectory of each planning problem leading to the origin of the LVLH frame is shown on the left hand side. On the right hand side the planned input signal is shown.

Acknowledgements

This work was supported by the German Aerospace Center (DLR) and Airbus Defence and Space as part of the project "Kontiplan" (50 RA 1221) and partially by EPSRC project EP/J012211".

References

Bogomolov, S.; Magazzeni, D.; Podelski, A.; and Wehrle, M. 2014. Planning as model checking in hybrid domains. In *Twenty-Eighth Conference on Artificial Intelligence*.

Della Penna, G.; Magazzeni, D.; and Mercorio, F. 2012. A universal planning system for hybrid domains. *Applied intelligence* 36(4):932–959.

Fehse, W. 2003. *Automated rendezvous and docking of spacecraft*. Cambridge and and New York: Cambridge Univeristy Press.

Fox, M., and Long, D. 2003. PDDL2.1: An extension to PDDL for expressing temporal planning domains. *J. Artif. Intell. Res. (JAIR)* 20:61–124.

Fox, M., and Long, D. 2006. Modelling mixed discrete-continuous domains for planning. *J. Artif. Intell. Res.(JAIR)* 27:235–297.

Frehse, G.; Le Guernic, C.; Donzé, A.; Cotton, S.; Ray, R.; Lebeltel, O.; Ripado, R.; Girard, A.; Dang, T.; and Maler, O. 2011. SpaceEx: Scalable verification of hybrid systems. In *Computer Aided Verification*, 379–395. Springer.

Girard, A.; Le Guernic, C.; and Maler, O. 2006. Efficient computation of reachable sets of linear time-invariant systems with inputs. In *Hybrid Systems: Computation and Control*. Springer. 257–271.

Grunbaum, B.; Klee, V.; Perles, M. A.; and Shephard, G. C. 1967. *Convex polytopes*, volume 2. Springer.

Henzinger, T. A. 2000. *The theory of hybrid automata*. Springer.

Hill, G. W. 1878. Researches in the lunar theory. *American Journal of Mathematics* 1(1):5–26.

Kailath, T. 1980. *Linear systems*, volume 1. Prentice-Hall Englewood Cliffs, NJ.

Kühn, W. 1999. Towards an optimal control of the wrapping effect. In *Developments in Reliable Computing*. Springer. 43–51.

Le Guernic, C. 2009. Reachability analysis of hybrid systems with linear continuous dynamics. *Univerit Joseph Fourier*.

Li, H. X., and Williams, B. C. 2008. Generative planning for hybrid systems based on flow tubes. In *International Conference on Automated Planning and Scheduling (ICAPS)*, 206–213.

Liberzon, D. 2003. *Switching in systems and control*. Springer.

Löhr, J.; Eyerich, P.; Keller, T.; and Nebel, B. 2012. A planning based framework for controlling hybrid systems. In *International Conference on Automated Planning and Scheduling (ICAPS)*.

Löhr, J.; Eyerich, P.; Winkler, S.; and Nebel, B. 2013. Domain predictive control under uncertain numerical state information. In *Twenty-Third International Conference on Automated Planning and Scheduling*.

Moler, C., and van Loan, C. 1978. Nineteen dubious ways to compute the exponential of a matrix. *SIAM review* 20(4):801–836.

Moler, C., and van Loan, C. 2003. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM review* 45(1):3–49.

Pearl, J. 1984. Heuristics: intelligent search strategies for computer problem solving.

Rubagotti, M.; Trimboli, S.; and Bemporad, A. 2013. Stability and invariance analysis of uncertain discrete-time piecewise affine systems.

Trimboli, S.; Rubagotti, M.; and Bemporad, A. 2011. Stability and invariance analysis of uncertain pwa systems based on linear programming. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, 7398–7403. IEEE.

Wang, L. 2009. *Model Predictive Control System Design and Implementation Using MATLAB®*. Springer.