**LTLCon: Software for control of linear systems from LTL formulas over linear predicates**  
Authors: Marius Kloetzer and Calin Belta  
Center for Information and Systems Engineering  
Boston University

**Problem formulation:** Consider the following affine control system in a full dimensional polytope *PN* in ***R****N*:  
 (1)  
where , , , and  is a given polyhedral subset of ***Rm*** capturing control constraints. Let  be an  formula over a set of atomic propositions given as arbitrary strict linear inequalities in ***R****N*:  
                          (2)  
where  and .

LTLCon finds sets of initial states and feedback control strategies for system (1) so that all trajectories of the corresponding closed loop system satisfy , while always staying inside .

An exact definition of semantics of LTL formulas over trajectories of the system can be found in [[3]](https://sites.bu.edu/hyness/ltl-control/#3),[[4]](https://sites.bu.edu/hyness/ltl-control/#4). Intuitively, a trajectory of (1) satisfies a formula if the word of satisfied predicates produced by the evolving trajectory satisfies the formula in the classical way defined in temporal logic. Some illustrative examples of words defined by continuous trajectories are given in Figure 1, where ,  are open half-spaces and , , , , , . Continuous trajectory  starts from the region where the predicates in  are true and converges to a point in the region where the predicates in  are true. The word  is . Trajectory  starts from  and loops infinitely as shown in the figure. The corresponding word  will be . For trajectory  originating in  and converging inside , the word  is . Finally, trajectory , which has a self-intersection, generates the word .

A black background with a black square

Description automatically generated with medium confidence  
Fig.1. Examples of continuous trajectories

**Implementation notes and acknowledgements:** LTLCon is implemented in Matlab. Even though this is transparent to the user, LTLCon also uses two free packages. The first one is a mex-file calling CDD in Matlab [[1]](https://sites.bu.edu/hyness/ltl-control/#1) and it is used to convert a polytope from V to H representations and vice-versa. The second one is LTL2BA [[2]](https://sites.bu.edu/hyness/ltl-control/#2), which is used to convert an LTL formula to a Büchi automaton.

**Usage and download**: LTLCon takes as input the polytope , the matrices , , and  of a system, and the LTL formula . If it finds a solution, it plots the produced trajectories corresponding to user defined initial states. Download LTLCon from [**here**](https://calinbelta.com/wp-content/uploads/2023/12/LTL_ctrl.zip). If you use our script files, please cite reference [[3]](https://sites.bu.edu/hyness/ltl-control/#3) from below.

**Example produced using LTLCon**: Consider the following system:  
 (3)

Polytope is specified in H form (), as the intersection of 8 closed half spaces, defined by: , , , , , , , , , , , , , , , . Control constraints are captured by the set 

We define a set  containing 10 predicates, as in equation (2), where: , , , , , , , , , , , , , , , , , , , .

Figure 2 depicts the bounding polytope , the vector field given by the drift of system (3), the predicates , , and the corresponding feasible subpolytopes (labeled , ).

A diagram of a geometrical figure

Description automatically generated  
Fig.2. The arrows represent the drift vector field of system (3). The yellow boxes mark the half-spaces corresponding to atomic propositions , . The regions to be visited are green, while the obstacles are gray.

We have chosen an LTL formula inspired from robot motion planning, which involves visiting a sequence of three regions infinitely often, while always avoiding three obstacles. The regions to be visited are, in order: ,  and , where  denotes the set of all points in the interior of the polytope with label . The obstacles are represented by the polyhedral regions ,  and . All regions to be visited and obstacles are represented in Figure 2. The LTL formula can be written as . By expressing interesting regions  and ,  in terms of predicates ,  we obtain    .

The set of initial states from which there exist continuous trajectories satisfying the formula is the union of the yellow polytopes in Figure 3 (a). A continuous trajectory starting from  (blue diamond) is also shown in Figure 3 (b).

|  |  |
| --- | --- |
| A yellow and blue hexagon with black lines and numbers  Description automatically generated | A diagram of a hexagon  Description automatically generated |
| (a) | (b) |

Fig.3. The set of initial states from which there exist continuous trajectories satisfying formula . (b) An example of such a trajectory.

**References:**

[1] Torrisi, F., Baotic, M.: Matlab interface for the cdd solver.  
[2] Gastin, P., Oddoux, D.: Fast LTL to Büchi automata translation. In G. Berry, H.C., Finkel, A., eds.: Proceedings of the 13th Conference on Computer Aided Verification (CAV’01). Number 2102 (2001) 53–65  
[3] Kloetzer, M., Belta, C.: A fully automated framework for control of linear systems from LTL specifications. In: Lecture Notes in Computer Science, Springer Berlin / Heidelberg, vol. 3927, pp. 333 – 347, 2006.  
[4] Kloetzer, M., Belta, C.: A fully automated framework for controller synthesis from linear temporal logic specifications. Technical Report CISE 2005-IR-0050, Boston University (2005).