Synthesizing Reactive Motion Plans for UAVs

Tristan Knoth and Gaurav Mahajan

October 20, 2017

Autonomous drones operate in dynamic environments. As these UAVs gain prevalence, safe and reactive motion plans become imperative. Writing safe controllers and motion plans, despite advances in verification, still requires significant effort on the part of the developer [1] [2]. Most research towards robot motion plan synthesis is either not reactive [3], or uses a "receding horizon model" [4]. Since UAVs are naturally constrained by their relatively short battery life, we will be able to generate complete high-level motion plans with state-of-the art program synthesis techniques. We will use a small domain-specific language describing the motion and sensing capabilities of a UAV, as well as the motion capabilities of any potential obstacles to describe the scenario. Given a starting position a goal, and a set of safety conditions, we will synthesize a reactive controller directing the robot to the target and back to the starting position to recharge its battery.

The following grammar for our DSL, inspired by previous work on high-level reactive controller synthesis [3], describes a UAV moving through a 2-dimensional $p \times q$ grid, with n obstacles and m motion primitives available.

$$
\begin{aligned}
E &::= E; E \mid P \\
P &::= \texttt{if}\ B\ \texttt{then}\ E\ \texttt{else}\ E \mid M \\
M &::= m_1 \mid m_2 \mid \ldots \mid m_m \\
B &::= B \parallel B \mid B \&\& B \mid \neg B \mid I \leq I \mid I = I \\
I &::= 0 \mid 1 \mid \ \ldots \ \mid \max(p, q) - 1 \mid I + I \mid I - I \mid \texttt{getX}\ R \mid \texttt{getY}\ R \\
R &::= r_0 \mid ob_0 \mid ob_1 \mid \ \ldots \ \mid ob_n
\end{aligned}
$$

The most interesting part of the language is the "motion primitives", generated by the variable M above. Our language is agnostic to the specific actions allowed, and should be able to be instantiated with any movements. In fact, we hope to allow any number of functions taking an (x, y) coordinate and outputting a new position, including transcendental functions.

In the simplest case, when motion primitives are simple linear transformations of the current coordinates, we will be able to leverage state-of-the art SMT solvers to do most of the synthesis. Since the UAVs are restricted to paths of a certain maximum length given by their battery capacity, the synthesizer should have only a finite number of language components available. Thus, it should be relatively straightforward to use an SMT solver to synthesize a satisfactory program.

However, we hope to also extend our synthesizer to more complex scenarios, in which the motion primitives can be all manner of functions, including transcendental functions. Here, we will need a more complex model for the UAV's resource use as well. We hope to use the `dReal` SMT solver to find a "δ-satisfactory" program [5]. Here, the notion of δ-satisfiability refers to the idea of meeting the success conditions within some small δ. This means that "hill-climbing" algorithms, while normally intractable for synthesis problems due to discrete definitions of correctness, may prove efficient and effective. Similarly, we may be able to leverage a resource aware type system [6] to guide our synthesis process when using a more complex model for UAV resources.

In the long run, this work may even extend to more complex and realistic scenarios. It extends naturally to 3-dimensional space, but may also apply to the increasingly-common scenario in which multiple drones work together towards a common goal. For now, however, we will start with simpler cases in order to investigate the utility of SMT solver-guided program synthesis for reactive UAV controllers.

References

[1] Ankush Desai, Indranil Saha, Jianqiao Yang, Shaz Qadeer, and Sanjit A Seshia. DRONA: A Framework for Safe Distributed Mobile Robotics ACM Reference format. 2017.

[2] Matthew O'Kelly, Houssam Abbas, Sicun Gao, Shiraishi Shiraishi, Shinpei Kato, and Rahul Mangharami. APEX: Autonomous Vehicle Plan Verification and Execution. *SAE*, 2016.

[3] Sarah Chasins and Julie L Newcomb. Using SyGuS to Synthesize Reactive Motion Plans. *EPTCS*, 229, 2016.

[4] Tichakorn Wongpiromsarn, Ufuk Topcu, and Richard M Murray. Receding Horizon Temporal Logic Planning. 2012.

[5] Sicun Gao, Soonho Kong, and Edmund M Clarke. dReal: An SMT Solver for Nonlinear Theories over the Reals. *CADE*, 2013.

[6] Jan Hoffmann, Klaus Aehlig, and Martin Hofmann. Resource Aware ML. In *24rd International Conference on Computer Aided Verification (CAV'12)*, volume 7358 of *Lecture Notes in Computer Science*, pages 781–786. Springer, 2012.