Certification of Online Optimization Algorithms Within Control Systems

Raphael Cohen, *Georgia Institute of Technology*,
Guillaume Davy, *École normale supérieure de Cachan,*
Pr. Eric Feron, *Georgia Institute of Technology*

Situation:

Optimization algorithms used in a real-time and safety-critical context offer the potential for considerably advancing robotic and autonomous systems by improving their ability to execute complex missions.
However, this promise cannot happen without proper attention to the considerably stronger operational constraints that real time, safety-critical applications must meet, unlike their non-real-time, desktop counterparts.
Advanced real-time algorithms are growing in complexity and length, related to the growth in autonomy, which allows aircraft, automobile, and medical devices to plan paths of their own.
On the other hand, the productivity of safety-critical software developers remains fairly constant at 0.6 to 1 line of code per hour. Knowing that software verification and validation represent fifty percent of their entire engineering development budget, it is then obvious that unless something is done soon, advanced real-time and safety-critical cost development using today's technologies will be unsustainable, if not impossible in the years to come.
In this study, we look for some methods of certification, including formal methods to tackle these issues. These methods targeting critical systems, we are really interested in applying it on real physical systems as the project goes forward.

Goals:

•Demonstrate the relevance and feasibility of embedding modern optimization (and control) algorithms in real-time applications, with strong theoretical guarantees.
•Support the expression of proof elements (including on-line optimization modules) to compile those enriched models down to code, carrying along proof elements.
•Develop the capability to re-check this information of proof elements for other purposes, such as verification and documentation.

Hoare Triple:

To express property on code we use Hoare Triple : *{P} C {Q}*.

- ***C* is the program you want to put a property on**
- ***P* is called the precondition it's an hypothesis on the state of your program before the execution of *C*.**
- ***Q* is the post condition, it specify how the state of your program must be after the execution of *C*.**

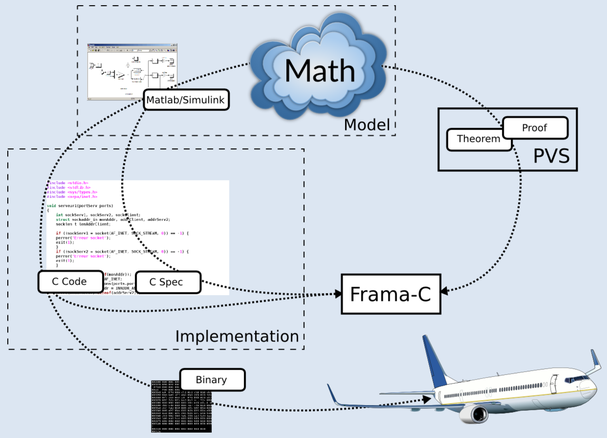
A Hoare triple is valid if and only if for all memory *M* that satisfy *P*, the memory *M'* you get after running *C* on *M satisfy Q*.

In this example you assert that if a is greater than 2 and b is greater than 3 then after the execution of the program you will have y greater than zero.

```
1. { a > 2 and b > 3}
2.
3. x = min(a, b)
4. y = 3 * x - 5
5.
6. {y > 0 }
```

Towards a Formally Verified Process:

- ***PVS is a proof assistant, it permits to write math in a computer understandable-language and also to prove them, to be sure that you wrote them correctly.***
- ***Frama-C is a tool that takes annotated C code and permits to check that the annotation are correct.***



This diagram represents the process of development of an embedded software as we are doing it:

- **Write the model of the physical system, the properties that it will verify, and the model of the software used. First in math language and then implemented in Matlab.**
- **AutoCode in C this Simulink model along with some annotation expressing Hoare triple on the C code.**
- **Check all the Hoare triples with Frama-C toolkit using math theorems expressed and proved in PVS.**

3 DOF Helicopter Quanser:

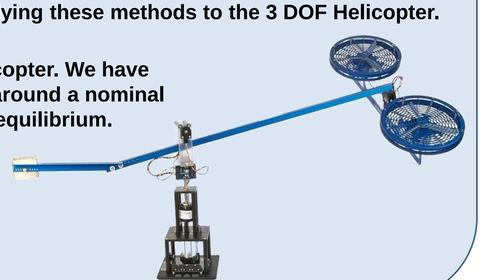
• 3 Degrees of Liberty robotic arm (Elevation λ, Pitch φ, Travel θ), supposed to model a helicopter.

• The Attitude control is done by the two DC motors and propellers

• In this study, we will focus on applying these methods to the 3 DOF Helicopter.

• We use a Linear Model of the Helicopter. We have linearized the equations of motion around a nominal input such that the Helicopter is at equilibrium.

(Thrust required on both motors to cancel out gravity)



Explanation of the Experiment:

• The goal is to have the helicopter going from one point to another (far away from each other) with avoidance of obstacles.
• In this study, we considerer the obstacle to be « Ceiling » and a « Ground ». Formulation of the Obstacle as Linear Constraints:
• To tackle these issues, we decided to implement an optimization algorithm on the 3 DOF Helicopter Quanser as part of the MPC concept.

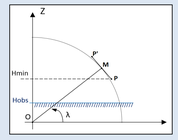


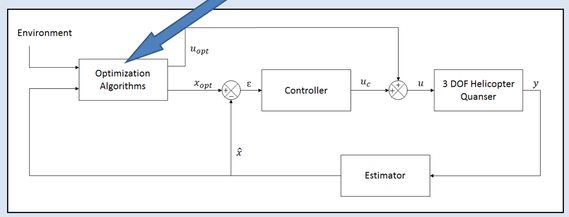
Figure of the Quanser with an obstacle (Ground)

Linearized constrains

$$l.\lambda - r.\psi \geq h_{obs}$$
$$l.\lambda + r.\psi \geq h_{obs}$$

Linearized model of the Quanser

$$\begin{cases} \dot{x} = A.x + B.u \\ y = C.x \end{cases}$$



Optimization problem:

$$Min \sum_{i=1}^{N} x_i{}^T Q\, x_i + u_i{}^T R\, u_i$$

Such that

- x_i *and* u_i *satisfy model dynamic constraint*
- x_i *satisfy the obstacle constraints*

Optimization algorithm property:

We are going to implement an interior point optimization algorithm and we want to verify the following constraints:

- **Output a feasible solution**
- **Bounded amount of time**
- **ε-close solution**

Future Tasks:

• Implementing the whole online optimization control system on the Quanser

• Developing an autocoder a proof carrying support for the linear programing optimization algorithms and their semantics

• Build an Analyzer allowing us to demonstrate that the output of the autocoder is indeed analyzable, and provably correct.