Research Proposal: Computational resource management of multi channel controller

Hodai Goldman (Hodaig@cs.bgu.ac.il)
Department of Computer Science,
Ben-Gurion University of the Negev, Beer Sheva, Israel
Supervisor: Dr. Gera Weiss

August 4, 2016

1 Background and Problem Formulation

Today's computer power allows for consolidation of controllers where a single computer can regulate many control loops, each with its varying needs of computation resources. This brings two research challenges that we intend to attack in the proposed thesis:

- How to schedule control tasks in order to achieve good performance in terms of control measures (overshoot, convergence speed, etc.)?
- What is a good interface for co-design of scheduling and control?

While it is possible to build control systems using standard operating systems, either real-time or desktop with static or with dynamic scheduling schemes, there is an agreed opinion in the control community that these do not serve well for the purpose outlined above [**?**]. Specifically, desktop type operating systems (Windows, Linux, etc.) schedule for computational efficiency, but do not allow for control performance guarantees of the individual loops. On the other hand, real-time operating systems sacrifice some efficiency for timing predictability, but they do relate timing information with control performance. When using such operating systems for control engineers usually apply controllers that work in a fixed periodic manner so that control behavior becomes deterministic and control performance can be guaranteed. This is not efficient because resources can be better utilized if controllers act at higher frequencies when only when needed. In this work we will develop methods to combine the efficiency of desktop operating systems with the predictability of real-time operating systems in a way that is more suitable for control systems then periods and deadlines

The control loops that we are analyzing are here of the form shown in Figure 1. A physical plant (controlled system) is connected via sensors and actuators...

The current state of the art is that control engineers design control tasks as periodic computations then they specify the required periodic frequency for the task and software engineers design a scheduler that ensures that the periodic frequency requirements are met usually using pre-computed knowledge of the expected (maximum) duration of the tasks. We claim that for control systems we can achieve better performance by using richer and more flexible set of requirements for the tasks. Specifically, we will develop tools with which the control engineers can specify in a natural way features of their control loop that the scheduler will use to allow for dynamic schedules that guarantee required control performance.

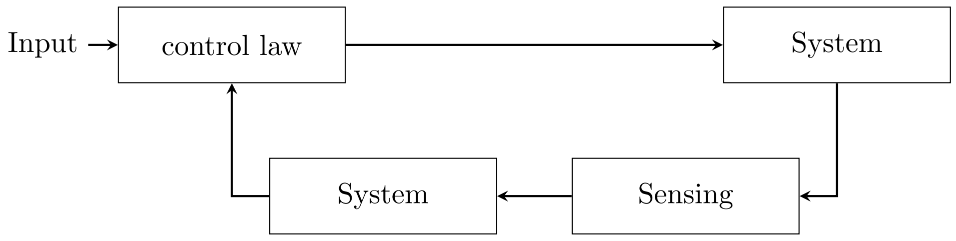

Figure 1: Typical control loop system.

2 Case study: Vision based controllers for drones

Tomatoes in green houses require much manual labor such as pollination. In fields Pollination is done by wind and by wild insects, In green houses by bumble bees or humans. Bumbles bees are the most popular In green houses but they are Sensitive to weather (heat) and Sensitive to pesticides.

The leading Case study scenario that we will be examine is a flying drones that will replace the bumbles. They goal is top identify the tomatoes flowers in a tomatoes row and then fly near the flowers in order to pollinate them using their natural rotors wind.

This task is very challenging for an autonomous drone, the hall between rows is very tight so the drone must be very stable and mast have accurate knowledge of his own state and location, therefore the drone use vision (cameras) to understand ware he is (state estimation) and to identify the flowers. This is lead to another challenging task, how can we construct real-time controller with such computational expensive *Computer Vision* tasks, without loose the stabilization (predictability) of the system. Tis is the main issue of this thesis, as show in section 3, we suggest a novel framework model for such systems.

3 Research Plan

3.1 the controller control system framework

As show in Figure 2 the suggested control system framework have strong relation between the tasks scheduler to the control loops, in this case to the estimation task (*State Estimation*) and the sensing task (*Computer Vision*), in this framework the scheduler is part of the control logic and therefore it can make scheduling decisions based on the current control state, in the above example (Figure 2) the scheduling of *Computer Vision* task is de-

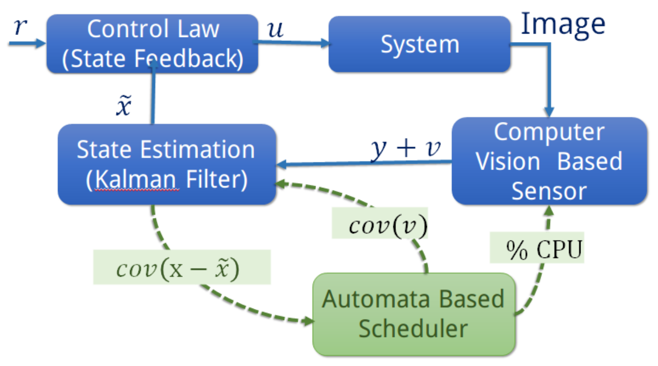
r
Control Law (State Feedback)
u
System
Image
$\tilde{x}$
State Estimation (Kalman Filter)
$y+v$
Computer Vision Based Sensor
$cov(v)$
% CPU
$cov(x-\tilde{x})$
Automata Based Scheduler

Figure 2: Our proposal of the controller control system framework

pended on the accuracy of the state estimation. For example if the vision is clear the *Computer Vision* will produce good measurement of the System and therefore good accuracy will be achieved so the *Scheduler* can allocate less computation time to the heavy *Computer Vision* task and still remain stable and allocate more CPU to others control loops or some background tasks (like navigation).

The above collaboration require to re-adjust some parts of the controlling system, e.g. the *State Estimator* need to work with variable error of *Computer Vision* measurement and the *Computer Vision* need to be able to run within variable time limits, and also the tasks pre-defined requirements need to be re-adjust in order to define the relation between the tasks like *Computer Vision* and *State Estimator*. now we will dive in each part of the above system (Figure 2) and explain how it should be adjusted (changed) and how we will achieve that adjustment in our demonstration (see 2).

Finally, we need to re-consider the system Analyzing technique because variable sensing characteristics (*Computer Vision* time limits) will lead to variable estimation characteristics...

3.1.1 Sensors (*Computer Vision Based Sensor*)

Computer Vision Based Sensor is the module that responsible to take picture or series of pictures and produce the goal measurement. For that example we will use the *Computer Vision* in order to detect 2 dimensional movement in the camera surface (i.e. the drone speed). There are lots of such algorithms (optical flow), they differ by their running time and by the accuracy of the solution, mostly more time lead to more accuracy.

In our case we need to be able to control the *Computer Vision* running time and to have some god knowledge of the solution accuracy in order to match the correspond state estimation (see 3.1.2) logic. We assume that the *Computer Vision* error is spread Normally, this is acceptable assumption because ??? [**?**], so we will use the error Covariance as the solution accuracy.

In order to control the running time, we will use an Anytime vision algorithms show by ??? [**?**]. This type of algorithms will run till we stop them, and wen we stop them they will provide a solution with accuracy that is proportional to the amount of time he was running, that way we can control the solution accuracy by controlling the running time. In our implementation, in order to lower the complexity, we will pre-define few different operation modes" of the *Computer Vision* task, that differ by their running time and they identified by a pair $(RunTime, Covariance)$ where *Covariance* is the error covariance of running time *RunTime*.

3.1.2 State Estimator

3.1.3 Control Tasks

3.1.4 Automata Based Scheduler

3.2 experimental environment

3.2.1 drones

3.2.2 Ardu-Pilot-Mega as base controller software

4 Preliminary Results

Blablabla said Nobody [3]. Blablabla said Nobody [1]. Blablabla said Nobody [2].

5 References

[1] Ardupilotmega (apm): open source controller for quad-copters.

[2] Rajeev Alur and Gera Weiss. Rtcomposer: A framework for real-time components with scheduling interfaces.

[3] Merav Bukra. Gamecomposer: A framework for dynamic scheduling. Master's thesis.