Reactive Scheduler

Gera Weiss
Ben-Gurion University
Beer-Sheva, Israel
geraw@cs.bgu.ac.il

Hodai Goldman
Ben-Gurion University
Beer-Sheva, Israel
hodaig@cs.bgu.ac.il

ABSTRACT

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

- How to schedule control tasks in order to achieve good performance in terms of control measures (overshoot, convergence time, etc.), within the still limited computation resources?
- What is a good interface for co-design of scheduling and control?

Desktop type operating systems, like Windows and Linux, schedule for computational efficiency but do not allow for worst-case performance guarantees. Real-time operating systems, on the other hand, sacrifice some efficiency for timing predictability, but the type of timing guarantees that such systems provide usually are not directly guaranteeing the control performance of the system. When using such operating systems for control, engineers usually apply controllers that work in a fixed periodic manner witch the 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 only when needed.

In this work we show how to combine the efficiency of dynamic scheduling with the predictability of real-time scheduling, in a way that is more suitable for control systems then periods and deadlines. We show that applying control computations at dynamically adjustable periodic, and with variable computational demands, based on real-time information, allows for better utilization of the computational resources and therefor better control performance.

1. INTRODUCTION

Sections go here.

2. ARCHITECTURE: AUTOMATA BASED SCHEDULER

Sections go here.

3. SIMULATIONS

4. PROPOSED METHODOLOGY

1. **Controller design:** Based on the separation principle, we propose to design the controller to achieve the control objectives assuming a perfect observation. In practice, this may not be feasible because controller designs such as PID require a system to experiment with. In this case, as demonstrated in the case-study below (see Section 5), we propose to work with one of the observation modes. If the system is close to linear, this should result with a near optimal design.
2. **Observers design:** Specification of sensor modes and observers design
3. **Performance analysis:** Now, we can perform some experiments with the different observer modes and analyze transient behaviors. Specifically, as shown in the case study **??**, we can measure how long it takes for the error to accumulate after switching to a lesser observation mode and formulate how this error affects the control objectives.
4. **Scheduling automata design:** Base on the analysis we can specify the resource scheduling requirements in the form of *specification automaton*. The

goal is to design flexible specification that allow dynamic scheduling in order to adapt the environment and the system state, this will improve the system efficient.

5. APPLICATION TO AUTONOMOUS QUADROTOR FLYING IN-DOOR

TODO - The quadrotor basics

The specific case study we used to test our concept is the ability of fly thru windows, this task require a very stable hovering in front of the window. We implement Autonomous controller for that task, and we evaluate the performance in terms of the horizontal linear axis parallel to the window surface, as this is the most critical axis in this task, this axis noted by x see Figure **??**.

5.1 Controller Design

The drone consist of Raspberry pi with navio [**?**, **?**] and is controlled by modified ArduPilot [**?**]. We implement standard PID [**?**] controller that regulate the position and attitude of the drone relatively to the window. We used inertial sensors (Gyroscope and Accelerometer) for attitude control relative to earth, and we use camera [**?**] and simple image processing for position and attitude relatively to the window **??**. The image processing have few different operation modes which have different image resolution, better resolution image produce more accurate measurement but consume more processing time, changing operation modes allows us to control the trade off between mesurment quality and processing time as shown in Table **??**, for simplicity we examine only two modes, *High quality* with resolution of 960p and *Low quality* with resolution of 240p.

As shown in the simulation at Section 3 we should use Kalman filter estimator, in our case we have non-linear system... For simplicity, inspiring from Kalman filter [**?**], we use two steps estimator that *predict* the current state evolved from the previous state using the system model ($x_{k|\hat{k-1}}$) and then *update* the prediction with current state measurement from image (z_k). The result estimation ($x_{\hat{k|k}}$) is weighted average: $x_{\hat{k|k}} = Kx_{k|\hat{k-1}} + (1-K)z_k$, the K gain is defined different for each resolution for achieving estimation in each resolution.

5.2 Analysis and Specification Automata

The objective of the system is to maintain stable hovering in front of the window, and the performances of the system is measured by the amount of deviation from the center line of the window in the critical axis x (see Figure **??**), and in other hand we gauge efficiency by the amount of processing resources is used by the heavy observation tasks (CPU percentage).

Analyzing the test results showing in Table **??** we see that High quality observation mode provide 0.4 meter error tolerance meaning we can fly this mode in 1.2 meter wide corridor [1], with the cost of 30% CPU usage. With adaptive scheduling specification we lower the CPU usage without significant worsening of High quality performances.

From the Low quality experiment graph we can see that measurement post-fit residual ($\tilde{y}_{k|k}$) is accumulate proportional to the deviation from the center of x axis, and we can use $\tilde{y}_{k|k}$ vales to predict

TODO - position ($x_{k|k}$) derived schedulers

TODO - complex / agregated schedulers

TODO - gyro derived??

5.2.1 results

TODO - graphs and tables

6. CONCLUSIONS

Conclusion goes here.

Acknowledgments

Acknowledgement goes here.

APPENDIX

Appendix goes here.

[1]Experiments was done with 0.4 meter wide quad-rotor