Exploring the trade-off between processing resources and settling time in image-based control through LQR tuning

Róbinson Medina*, Sander Stuijk*, Dip Goswami*, Twan Basten*†

* Eindhoven University of Technology
Eindhoven, The Netherlands

†TNO ESI
Eindhoven, The Netherlands

ABSTRACT

Image-Based control systems extract information by a camera and an image processing algorithm. The challenge of such controllers is that the sensing latency deteriorates the control performance. Multi-core technology can be used to implement the sensing algorithm in a pipelined fashion. More processing resources potentially lead to better settling time. This results in a trade-off between resources and performance. We present a method to analyse this trade-off.

CCS Concepts

•**Computer systems organization** → **Multicore architectures;** *Embedded hardware; Embedded software;* •**Hardware** → Yield and cost optimization;

Keywords

Image-based control; pipelined sensing control; Particle Swarm Optimization; trade-off analysis; LQR tuning.

1. MOTIVATION

Image-Based Control (IBC) is a common control strategy in fields such as Advanced Driver Assistance Systems (ADAS) [5] or visual servo control [2]. The main challenge in designing an IBC is to cope with the latency induced by the execution time of the image processing algorithm. To address this challenge, platforms with parallel processing capabilities can be used to implement the sensing algorithm in a pipelined fashion. For example, consider a second order under-damped system being controlled by an IBC with a sensing latency of 0.084 s. Using a non-pipelined (sequential) configuration of Fig. 1a and a pipelined sensing configuration of Fig. 1b, the settling is reduced by 30% as shown in Fig. 2.

The use of more pipes results in a shorter sampling period and potentially improves settling time. However, using more pipes implies more processing resources. This work focuses on the trade-off analysis between the number of pipes used and the settling time.

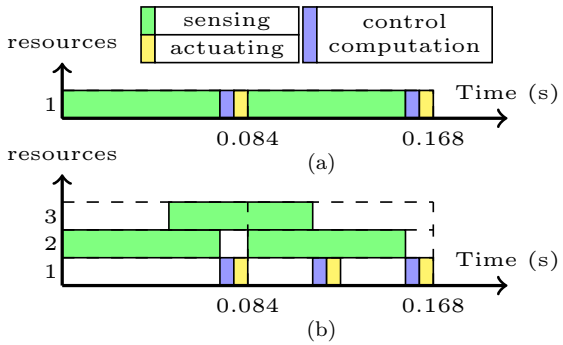

Figure 1: Example resource configuration of image-based control. a) Sequential configuration. b) Pipelined configuration with two sensing cores.

To analyse such a trade-off, a Linear Quadratic Regulator (LQR) controller has to be tuned for each resource configuration (i.e., number of pipes) such that the settling time is minimized. Tuning an LQR requires that its tuning matrices Q and R remain positive semi-definite. Classical tuning strategies for LQR, for reasons of simplicity, focus on finding the diagonal of such matrices. We use Particle Swarm Optimization (PSO) to find not only the diagonal of such matrices but also the off-diagonal elements to achieve better performance.

2. PROPOSED LQR TUNING

Pipelined sensing modelling. Linear time-invariant control applications of the following form are considered:

$$\dot{x}(t) = Ax(t) + Bu(t-\delta) \qquad (1)$$
$$y(t) = Cx(t)$$

where $A \in R^{n\times n}$ is the state matrix, $B \in R^{n\times i}$ the input matrix, $C \in R^{o\times n}$ the output matrix, $u(t-\delta) \in R^i$ the input vector, $x(t) \in R^n$ the state vector, $y(t) \in R^o$ the output vector, i the number of inputs, o the number of outputs, n the number of states, and δ the sensor-to-actuator delay which is the total of the sensing, control computation, and actuating latencies. The control application shown in Eq. 1

SAC 2017,April 03-07, 2017, Marrakech, Morocco

http://dx.doi.org/10.1145/3019612.3019862

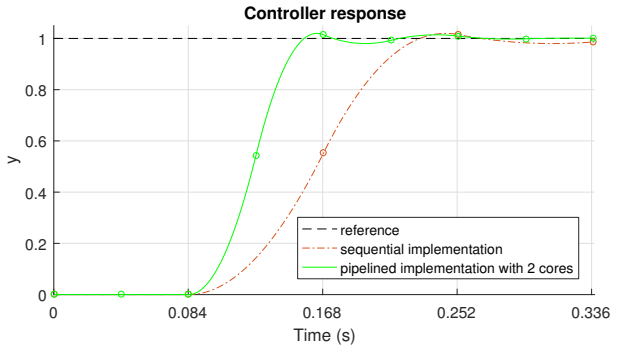

Figure 2: Controller response of motivational example with and without pipeline. The round markers denote multiples of the sampling period.

has an equivalent discrete-time pipelined model of the form:

$$z(\mathbb{K}+1) = \Phi z(\mathbb{K}) + \Gamma u(\mathbb{K}) \tag{2}$$

where $z(\mathbb{K}) \in \mathbb{R}^{n+l}$ is the augmented state vector, $\Phi \in \mathbb{R}^{(n+l)\times(n+l)}$ and $\Gamma \in \mathbb{R}^{(n+l)\times i}$ are the augmented discrete state and augmented input matrix respectively. l corresponds to the difference between the states of the discrete-time pipelined model and the continuous time model. l is found according to $l = i\gamma$ with γ the number of sensing pipes [7]. The controller sampling period is defined by:

$$h = \frac{\delta}{\gamma} \tag{3}$$

Note that a larger number of pipes leads to a smaller sampling period in Eq. 3.

LQR controller. An LQR controller uses the control law:

$$u(\mathbb{K}) = -Kz(\mathbb{K})$$

with K the feedback gain. K is found minimizing the cost function:

$$J = \min \sum_{\mathbb{K}=0}^{\infty} z^T(\mathbb{K})Qz(\mathbb{K}) + u^T(\mathbb{K})Ru(\mathbb{K}) \tag{4}$$

with $Q \in \mathbb{R}^{(n+l)\times(n+l)}$ and $R \in \mathbb{R}^{i\times i}$, the state and input weight matrices of the LQR which are symmetric positive semi-definite and symmetric positive definite matrices respectively, i.e. $Q = Q^T \succeq 0$ and $R = R^T \succ 0$ [6].

3. METHOD

Algorithm 1 PSO algorithm

```
Initialize random population X with m particles
while stopping criterion is not met do
    for each particle X_j do
        Compute fitness f
    end for
    for each particle X_j do
        Compute speed v
        Update swarm X
    end for
end while
```

PSO is used to tune an LQR with minimum settling time. Such an algorithm uses a population X of m individuals to explore in parallel the design space of a problem. The PSO algorithm evolves in an iterative manner. In each iteration, a fitness f is first computed to determine the quality of the solution found by each particle and then the population evolves according to a speed v. The process is repeated until a stop criterion is met. An example PSO algorithm is shown in Algorithm 1. PSO has been used to tune the LQR controller in several applications (e.g. [4]). In such applications, PSO minimizes a performance metric while tuning the diagonal elements of Q and R. These approaches restrict themselves to tuning the diagonal, because it is challenging to tune all matrix elements while keeping the matrices positive (semi-)definite.

Proposed approach. Our PSO algorithm finds not only the diagonal of Q and R but also all the elements of such matrices that minimize the settling time. To do so, each particle X_j in the population X is defined as the concatenation of all values of two random matrices α_j and β_j:

$$X_j = [\alpha_j \beta_j]$$

where $\alpha_j \in \mathbb{R}^{(n+l)\times(n+l)}$ and $\beta_j \in \mathbb{R}^{i\times i}$ are matrices of $rank(\alpha_j) \leq n+l$ and $rank(\beta_j) = i$ respectively. The Q of each particle (Q_j) is defined according to Theorem 1. R is defined in an analogue manner.

THEOREM 1. *For a symmetric positive semi-definite matrix Q_j, there exists a square matrix α_j of $rank(\alpha_j) \leq n+l$, such that $Q_j = \alpha_j^T \alpha_j$.*

PROOF. [3, Proposition 6.62]. □

The definition of the population through α_j and β_j, allows the PSO algorithm to explore all values of Q and R in the range of all the real numbers, while guaranteeing $Q = Q^T \succeq 0$ and $R = R^T \succ 0$. This definition requires that after updating each particle X_j, the ranks of α_j and β_j follow Theorem 1. In case they do not, the particle is not updated and the speed of the particle X_j is set to zero. In practice, it rarely occurs that the rank condition is not satisfied.

The fitness metric is determined according to Eq. 5.

$$f = -S_t \tag{5}$$

where S_t is the closed loop settling time. The negation is to take into consideration the objective of minimizing settling time. The PSO algorithm evolves according to the well known equation Eq. 6.

$$X = X + v \tag{6}$$

where v is the PSO speed given by:

$$\begin{aligned} v = wv_{old} + C_p rnd_1(X - X_{pb}) + \\ C_g rnd_2(X - X_{gb}) \end{aligned} \tag{7}$$

where rnd_1 and rnd_2 are random uniformly distributed numbers in the range $[0, 1]$, v and v_{old} are the current and previous speeds respectively, X_{pb} and X_{gb}, the personal and global best fitness achieved by each particle respectively.

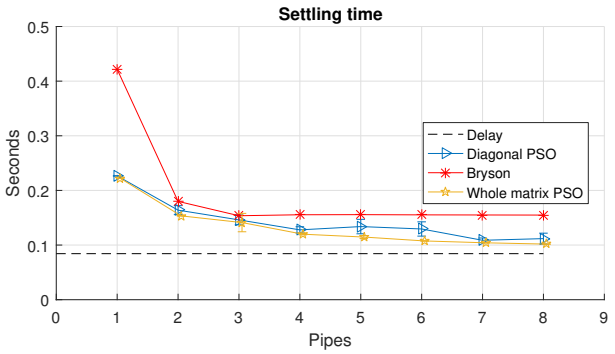

Figure 3: Tuning of LQR with different approaches.

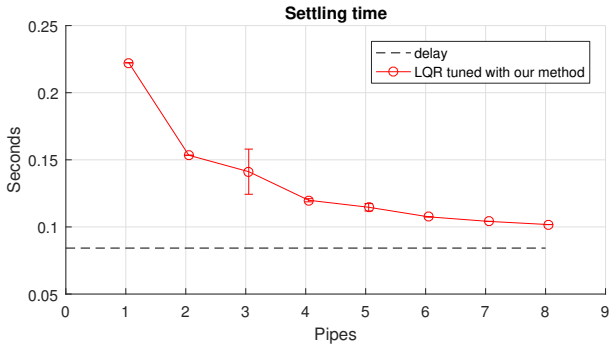

Figure 4: Trade-off between settling time and processing resources with fixed delay $\delta = 0.084$.

The other variables are tuning parameters of the speed equation: C_p and C_g the personal and global confidences respectively, and w the inertia.

Our algorithm follows the structure of Algorithm 1: during an initialization phase, the tuning parameters of the PSO Cp, Cg, m, and w are set and the population X is randomly initialized. During the iterations phase, for each particle in the population Q and R are extracted. The fitness is evaluated according to Eq. 5. The global best X_{gb} and the personal best of each particle X_{pb} are updated in case they improve their previously stored value. The particles are then updated according to Eqs. 6 and 7. The iterations are stopped when the global best remains unchanged during ten iterations or a maximum of 200 iterations is reached.

With this method, the settling time is minimized for a fixed number of pipes tuning all the elements of the full Q and R.

Example. An example of different tuning strategies of an LQR in the second order under-damped pipelined sensing described in Section 1 is shown in Fig. 3. Three techniques are considered: the well-known analytic tuning strategy proposed by Bryson [1], the classical PSO for tuning only the diagonal of Q and R of [4], and our approach based on PSO for tuning all the elements of Q and R.

The method of Bryson improves settling time up to three pipes. However, further increasing the number of pipes has no effect on the settling time because the design space is too large, leading to controllers incorrectly tuned.

Tuning the diagonal of Q and R using PSO outperforms the settling time found by the method of Bryson because of the more exhaustive search in the design space. A trade-off analysis with this method suggests that the performance of the controller is improved with every new pipe, except when adding 5 and 6 pipes.

Tuning the whole matrix performs as least as good as tuning only the diagonal. The result of a trade-off analysis with this method differs from the previous cases. The settling time of the controller is always improved with each newly added pipe. However, after a certain point adding more pipes only generates a minimal improvement in the settling time at a cost of extra sensing hardware. This example illustrates that our method is therefore more suitable to analyse the trade-offs of a pipelined IBC than existing approaches.

4. TRADE-OFF ANALYSIS

This section presents the trade-off between processing resources and settling time. The analysis is obtained after simulations on the second order under-damped dynamic model presented in Section 1. We show that the amount of sensing delay is a critical factor in such a trade-off.

Settling time vs number of pipes. Fig. 2 shows that having more sensing pipes allows to shorten the settling time. To this end, an initial resource configuration of one pipe is considered. Our method is used to tune an LQR with minimum settling time. The processing resources are increased and the process is repeated until the improvement on settling time is negligible. The result is shown in Fig. 4. The vertical bars on each point denotes the standard deviation which is obtained after running the same simulation 10 times. Note that this deviation is virtually zero in all the experiments except in the case of three pipes, where the PSO algorithm showed slightly different results.

Fig. 4 shows that the settling time gets shorter with each added pipe. However, the improvement does not grow proportionally with the number of pipes. For example, after four pipes the settling time improvement is reduced to an order of milliseconds: therefore there is little effect from the extra processing resources. Clearly, for the system under consideration and the given sensing delay, a higher number of pipes is (practically) beneficial until four pipes. In general, such region of interest depends on the system dynamics and should be considered in the design phase of a pipelined controller.

Settling time vs delay. The trade-off between processing resources and settling time depends not only on the model dynamics but also on the sensing delay. We illustrate this part of the trade-off simulating a variation of the sensing delay in the example provided in Section 1. Fig. 5 shows that the improvement in settling time depends on the sensing delay. The shortest settling time is achieved with the shortest delay, because the resulting sampling period is capable of sensing the fastest change in the model. A large delay with insufficient pipelining results in a low sampling frequency and an under-sampled model, which potentially increases the settling time. Therefore, given a sensing delay the number of sensing pipes should be chosen to make sure that the sampling period is capable of sensing the fastest

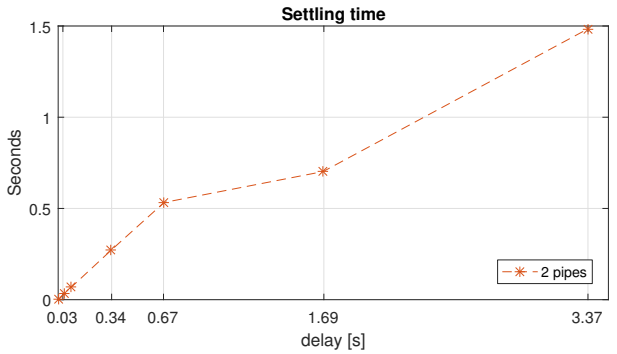

Figure 5: Impact of sensing delay on the settling time (with delay subtracted for comparison purposes) using two sensing cores as processing resources.

change in the system.

In Fig. 6 the settling times are shown when two different delays are considered while the sampling period (i.e. processing resources) is varied. The settling time gets longer proportionally with the sampling period. Note that if different delays have the same sampling period due to differences in processing resources (e.g. in Eq. 3 $\delta = 0.084\ s$ with $\gamma = 2$ produces $h = 0.042\ s$ and $\delta = 0.042\ s$ with $\gamma = 1$ also produces $h = 0.042$) the resulting settling time is the same. Consequently, if a system has the possibility of a short and a long sensing delay (e.g. by using different processors with different speeds), the settling time achieved with the short delay can also be achieved with the long delay, if sufficiently many processing resources are added. Alternatively, if a certain settling time is needed, it can be achieved by increasing processing resources irrespective of the sensing delay (assuming that a feasible solution exists).

The above analysis clearly demonstrates the trade-off between sensing delay, number of sensing pipes and settling time. The fact that the effect of sensing delay can be mitigated with additional sensing pipes holds in a specific region of operation of a system which further depends on the nature of the system dynamics. For example, in Fig. 6, using more than five sensing cores does not show significant improvement for the system under consideration. The improvement due to pipelined control is mainly visible with one to four sensing cores. The analysis essentially identifies such a region which is of main design interest. Moreover, the effect of a long sensing delay becomes more prominent in presence of noise and disturbances which are usually present in real-life systems. The influence of system dynamics on the trade-off and the effect of disturbance/noise are part of our future work.

5. CONCLUSIONS AND FUTURE WORK

We presented a method to analyse the trade-off between processing resources and settling time in a pipelined Image-Based Control (IBC). The method uses Particle Swarm Optimization for tuning a Linear Quadratic Regulator (LQR) that minimizes the settling time. Our method shows that finding all the values of the tuning matrices in the LQR potentially improves the settling time compared to tuning only

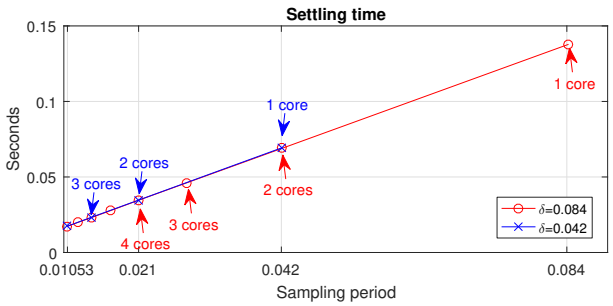

Figure 6: Impact of sampling period on settling time (with sensing delay subtracted for comparison purposes).

the diagonal elements.

We analysed the resource usage-quality of control trade-off in a second order under-damped system. Three factors need to be considered in an IBC implemented in a pipelined fashion: (i) sensing delay, (ii) number of sensing pipes, and (iii) settling time (quality of control in general). Settling time can potentially be shortened with extra resources, leading to a shorter sampling period. The effect of a higher sensing delay (e.g. resulting from a compute-intensive image processing algorithm) can be mitigated by additional sensing pipes. Experiments on other types of systems (e.g. over-damped, third order, fourth order system dynamics, not reported in this paper for space reasons) provide similar results.

Future work includes trade-off analysis over a wider range of systems including the effect of model uncertainties and experimental implementations.

6. ACKNOWLEDGMENT

This work was funded by the Dutch Technology Foundation STW as part of the Robust Cyber-Physical Systems (rCPS) program, project 12697.

7. REFERENCES

[1] A. E. Bryson. *Applied optimal control: optimization, estimation and control*. CRC Press, 1975.

[2] F. Chaumette and S. Hutchinson. Visual servo control. I. Basic approaches. *IEEE Robotics and Automation Magazine*, 13:82–90, 2006.

[3] P. J. Dhrymes. *Mathematics for econometrics*. Springer, 1978.

[4] H. Duan and C. Sun. Pendulum-like oscillation controller for micro aerial vehicle with ducted fan based on LQR and PSO. *Science China Technological Sciences*, 56:423–429, 2013.

[5] K. Jo et al. Development of autonomous car—part I: Distributed system architecture and development process. *Industrial Electronics, IEEE Trans. on*, 61:7131–7140, 2014.

[6] F. L. Lewis and V. L. Syrmos. *Optimal control*. John Wiley & Sons, 1995.

[7] R. Medina, S. Stuijk, D. Goswami, and T. Basten. Reconfigurable pipelined sensing for image-based control. In *Proc. SIES*. IEEE, 2016.