1. Image-based Control (IBC)

- IBC systems are a class of data-intensive feedback control systems whose feedback is provided by image-based sensing [1].
- Embedded platform (discrete-time)

![Camera](image)

- Sensing and processing
- Control computation
- Actuation

Dynamic system (continuous time)

Figure 1. An image-based control system: block diagram

- Here, the sampling period (h) and the sensor-to-actuator delay (τ) is greater than the frame arrival duration (see Fig. 2), i.e. h > fps⁻¹. fps denotes the camera frame rate per second.

![Camera input](image)

![Time](image)

- SPADe (continuous time)

- Controller design is a black box

2. Controller Design vs Implementation

<table>
<thead>
<tr>
<th>Control Design Engineer</th>
<th>Embedded Systems Engineer</th>
</tr>
</thead>
<tbody>
<tr>
<td>The timing values for the worst-case (WC) workload are used for controller design [2].</td>
<td>Allocates sufficient platform resources to ensure the timing for the worst-case workload.</td>
</tr>
<tr>
<td>Sensing task is a black box</td>
<td>Controller design is a black box</td>
</tr>
</tbody>
</table>

The Gap

worst-case workload rarely occurs inefficient resource utilisation

![Gantt chart](image)

- Figure 2. IBC system gantt chart

- Inputs: SPADe
- Formal Modelling
- Analysis and Design
- Implementation

![Model](image)

- Application
- Workload Variations
- Platform Allocation
- Camera Input fps

- Timing Scheduling
- Reconfiguration Mechanism
- Controller Configuration

- Analysis and Design: Analyse application and platform models to design system configurations.
- Reconfiguration mechanism for run-time implementation.

3. Bridging The Gap

Can we jointly optimise control performance and platform resource utilisation considering workload variations?

4. Approach

- We propose a structured Scenario- and Platform-Aware Design (SPADe) flow for IBC systems (assuming it is a white box) [1] that:
  1. optimises control performance or quality-of-control (QoC),
  2. maximises effective resource utilisation and
  3. adheres to platform constraints (given allocation and fps).

- Application Model
- Platform Model

- The SPADe approach involves the following aspects (see Fig. 4):
  a) Formal Modelling: i) identify and model the parameters that characterise workload variations, and ii) model application considering workload variations and platform considering platform constraints.
  b) Analysis and Design: Analyse application and platform models to design system configurations.
  c) Reconfiguration mechanism for run-time implementation.

5. Results & Conclusion

- SPADe maximises effective resource utilisation and improves the settling time for the control system compared to WC design.
- Considering workload variations is definitely beneficial for design.

![Graph](image)

- Figure 3. Design-Implementation Gap for worst-case (WC) workload.

- The execution time for sensing task depends on image workload variations which can be statistically analysed (e.g. as a PERT [3]).
- The designer can classify frequently occurring workload scenarios and always we observe that worst-case workload rarely occurs.
- An average workload scenario results in idling of the resource.
- A WC workload-based implementation means that frames have to be dropped even though the resource is idle/available.

6. Next Challenges

Extend SPADe approach for: i) (reconfigurable) pipelined controller design and implementation; ii) approximated image processing algorithms; and iii) communication-aware design for distributed IBC.

![Graph](image)

References


Acknowledgement

This project has received funding from the European Union’s Horizon 2020 Framework Programme for Research and Innovation under grant agreement no 674875.