PROCEEDINGS A

royalsocietypublishing.org/journal/rspa

Research

Cite this article: Qi Q, Terkaj W, Urgo M, Jiang X, Scott PJ. 2022 A mathematical foundation to support bidirectional mappings between digital models: an application of multi-scale modelling in manufacturing. *Proc. R. Soc. A* **478**: 20220156.
https://doi.org/10.1098/rspa.2022.0156

Received: 22 March 2022
Accepted: 17 June 2022

Subject Areas:
mechanical engineering, category theory

Keywords:
digital twin, information flow, delta lens, hybrid delta lens, bidirectional transition, assembly line

Author for correspondence:
P. J. Scott
e-mail: p.j.scott@hud.ac.uk

A mathematical foundation to support bidirectional mappings between digital models: an application of multi-scale modelling in manufacturing

Q. Qi[1], W. Terkaj[2], M. Urgo[3], X. Jiang[1] and P. J. Scott[1]

[1]EPSRC Future Advanced Metrology Hub, Centre for Precision Technologies, School of Computing and Engineering, University of Huddersfield, Huddersfield HD1 3DH, UK
[2]STIIMA-CNR, Institute of Intelligent Industrial Technologies and Systems for Advanced Manufacturing, National Research Council, Milan, Italy
[3]Department of Mechanical Engineering, Politecnico di Milano, Milan, Italy

PJS, 0000-0002-6092-3101

With manufacturing going through the Industry 4.0 revolution, a vast amount of data and information exchange leads to an increase in complexity of digitized manufacturing systems. To tackle such complexity, one solution is to design and operate a digital twin model under different levels of abstraction, with different levels of detail, according to the available information and scope of the model. To support efficient, coherent and stable information flows between models with different levels of detail, a mathematical structure, called a delta lens, has been explored and developed to support rigorous bidirectional transitions between the models. To support different types of abstractions in manufacturing, a hybrid delta lens has been proposed and its formal representation is developed to support the generalization of its structure and properties. Benefits of the proposed hybrid delta

THE ROYAL SOCIETY PUBLISHING

lenses are demonstrated through an application to an industrial case to support the modelling of an automatic, high-throughput assembly line.

1. Introduction

The design and analysis of engineered objects today have been developed with an impressive degree of complexity. In a wide range of areas, e.g. information technology, transportation, urban planning, manufacturing, logistics, etc., modern products, systems and services are the result of multiple components acting together to provide a global behaviour or functionality. This scheme falls into the area of complex systems, consisting of a large number of interacting components/agents exhibiting an emergent collective behaviour which is difficult to anticipate from the knowledge of each component/agent [1]. The design and analysis of complex systems require proper models, i.e. a simplified formal representation of a system taking into consideration the relevant features and behaviour of the complex system [2]. Pursuing this class of analyses for large systems using analytical methods is usually hopeless [1], thus, researchers and engineers must rely on computer-based approaches, e.g. numerical simulations.

Relevant examples of complex systems are manufacturing systems, comprising products, equipment, people, information, control and support functions contributing to the final goal of achieving the devising of products and services for the customers.

The complexity of manufacturing systems is due to a wide range of factors. Firstly, the need to take into consideration different types of behaviours: the physics linked to the manufacturing processes to be operated, the organization supporting the interactions among different agents (machine, transporters, human operators, etc.), the dynamics due to the continuous evolution of internal and external factors (supplies, failures, production volumes, etc.), management and control policies governing the behaviour of components and agents, information and data supporting many of the mentioned factors. Secondly, the need to address analyses and decisions at different levels in the system (e.g. machine, production cell, factory) with different levels of detail. As an example, fine-tuning a process requires us to take into consideration the detailed behaviour of a machine and the continuously sampled values of monitoring variables. On the contrary, planning the production can rely on a set of more aggregated datasets. Finally, as is usual in complex systems, the comprehensive behaviour can only be analysed by considering all the components/agents and their interactions at different levels of detail.

The use of digital technologies in industry, e.g. sensors, digital models, cloud computing, and integration of multiple heterogeneous data repositories, has opened a wide range of routes to manage and control manufacturing systems. Such enabling technologies, often referred to as Industry 4.0 [3], have facilitated the implementations of intelligent design [4,5], management [6,7], data analytics [8] and control approaches [9,10]. The Digital Twin (DT) plays a key role in this context [11] because it is the virtual (digital) counterpart of a real system, equipped with a set of models and algorithms to support decision-making, a continuous flow of data sending from the real manufacturing system and a control bus to implement decisions back to the system [12,13]. The bidirectional and continuous flow of data asks for appropriate communication technologies and algorithms. Various data, knowledge, models, services, etc., need to be linked to achieve more efficient, smart and sustainable manufacturing environments.

If all the resources of a DT are centrally stored and processed, it will cause a large burden for the transmission, computation usage, and maintenance. Thus, pursuing the achievement of accurate representations of a whole manufacturing system in a single monolithic digital twin is an unreasonable and unfeasible approach. On the contrary, a wise strategy is the definition of different interacting models designed to provide the desired level of details, matching the objective of the analysis or design phase. Indeed, a digital twin can be designed under a ladder of abstraction, where each abstraction represents a level of details, driving the need for multi-scale models where complex phenomena are abstracted into different levels exploiting data with

heterogeneous resolutions. This approach is applied in multi-fidelity models [14] consisting of multiple models with different levels of complexity or accuracy according to the intended use and the available data. Low-fidelity models (LFMs) are constructed by reducing certain details of a real system that is associated with a high-fidelity model (HFM). LFMs enable rapid analyses and low computational load, but cannot catch all the details of HFMs. The composition of models with different fidelity leads to a multi-fidelity surrogate model (MFSM), designed to bring in the desired degree of detail [15]. The seamless integration of these sub-models must guarantee that it is possible to go up and down the ladder of abstraction in a steady and secure way, i.e. to ensure bidirectional transitions with different levels of fidelity in an efficient, coherent and stable manner.

In this paper, a *digital* lens that enables the digital system to zoom in and out between models with different fidelities, is proposed as the mathematical foundation to formalize rigorous bidirectional transitions between models in a digital twin for manufacturing applications. The remainder of the paper is organized as follows. Section 2 describes basic characteristics of bidirectional information flow and what is a *digital* lens. Section 3 further extends the digital lens with an extra Delta structure, namely Delta lens, and why the Delta lens is suited for providing mathematical foundations for bidirectional information flow. To support different types of abstraction, a hybrid delta lens is proposed and its formal representation is developed to support the generalization of its structure and properties in §4. The proposed hybrid delta lens structure is then tested in §5 using an assembly line digital twin. Section 6 ends the paper with the conclusions.

2. Bidirectional information flow and lenses

With respect to modelling approaches for complex systems, the intrinsic complexity drives the need for hierarchical structured models consisting of sub-models interacting together in terms of information flows, propagating information within the model of the whole system. Moreover, as described in §1, models could support analyses and decisions at different scales and/or levels of detail. Thus, sub-models with different fidelity [14] could be interchanged, within the same structure of the global model, to pursue efficiency and tractability of the problems to be solved.

Information flow, in the context of hierarchical structured models, is considered as transformation/synchronization of information from one model to another, and vice versa when backwards. Therefore, the flows between the two models are intrinsically bidirectional, with forward flow and backward flow, in a sense that after the forward flow data in the sender and receiver co-exist and evolve independently, and the backward flow is able to bring the updated data back to a consistent state. It is essential to understand how information is transmitted from a sender to a receiver, and vice versa. To ensure stability of the flow, one requires small changes in either models to produce small changes in other model after the forward or backward flow [16].

Mathematically a bidirectional flow is considered as vertical, whenever the two models have different scales or levels of detail in the hierarchical structure, and flows between two models with the same scales or levels can be characterized as horizontal, provided that the two models share a common piece of information, but each of them contains something new not present in the other. Note that the definitions of vertical and horizontal information flows are distinct from information flows in the management of manufacturing as in [17], as in our case we focus on the mathematical characteristics of the flow rather than its management aspect. In real-case scenarios, without an explicitly defined hierarchical structure of the whole system, it is sometimes rather difficult to have a clear distinction about whether two models are from the same level (or with the same fidelity), and mathematically it is more challenging to characterize its general structure as sometimes it is heuristic and case based. In this paper, the focus will be on bidirectional information flows between two models with different fidelity, i.e. information flow between the global model and its sub-models, or flows between a sub-model and its sub-sub-model.

For example, as shown in figure 1, under the same digital system, let model A be a global model, and model C be one of its sub-models, the bidirectional flows between model A and model C, written as $A \rightleftarrows C$, represent the exchange of information between the two models. As

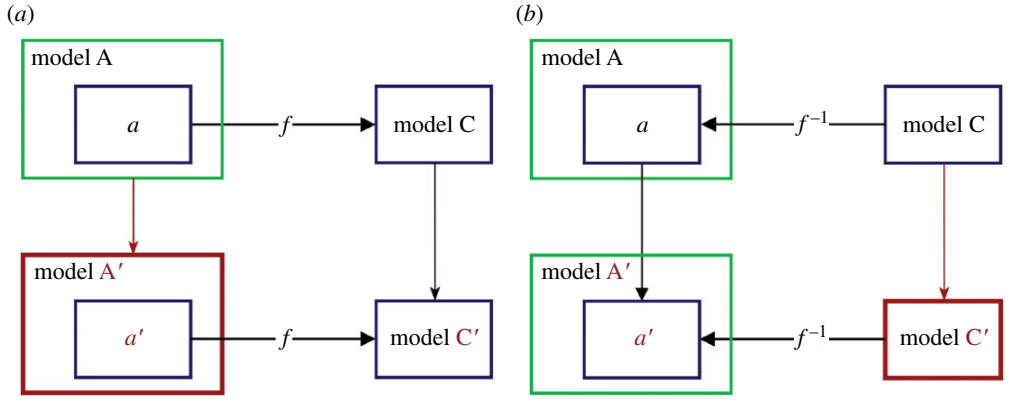

Figure 1. Bidirectional information flow between two models with different fidelity. (*a*) Forward information flow. (*b*) Backward information flow. (Online version in colour.)

model C takes a sub-set of information a from A, model C is generated by abstracting away some characteristics of model A, that is, C is deterministically computed from A. Mathematically the relationship between A and C is many-to-one, therefore different states of A can have the same states in C. Technically as the same model A can have many different sub-models other than C, understanding possible behaviour and outcomes of the bidirectional flow between A and C, or many others, will be helpful to aid with identifying a suitable sub-model.

Take the flows $A \rightleftarrows C$ as an example: as shown in figure 1*a*, the forward flow f from model A to C sends all information taken from the common part a in A to C; for the receiving end, model C gets all a from A and maps all things in a isomorphically to all things in C. Here, any updated A' results in an updated a', will be mapped into C to get a new status of C'. The backward flow f^{-1} from model C to A, as shown in figure 1*b*, sends all in C to the common part a in A; and any updated C' will be mapped isomorphically to the same part in a of A, resulting in an updated A'.

(a) Lenses

In a similar sense to a physical optic lens, a lens can be a mathematical structure which allows zooming into a small part of a larger system in the digital world. Here the zoomed-in part is the *view*, and the larger system the *source*. Any changes in the view will be reflected or zoomed out to the source. Lenses were originally defined to precisely present synchronization between a pair of sets, called state-based lenses [18] in which it is possible to update from one state of the system to another, and the only information retained on this update is its initial and final states. Here lenses are asymmetric, as the view is determined by the source, and the source has all the information required to reconstruct the view.

Formalized by category theory, let $\mathbf{S}$ be a set of source models, and $\mathbf{V}$ be a set of view models, a lens l between the two sets of models, write as $l : \mathbf{S} \rightleftarrows \mathbf{V}$, as shown in figure 2*a*, consists of two propagation functions *get* and *put*. The $get : S \to V$ function takes a source model $S \in \mathbf{S}$ and computes its view $V \in \mathbf{V}$, that is $V = get(S)$. The $put : S \times V \to S$ function takes an updated view $V'(V' \in \mathbf{V})$ of V and the original source model S, and computes an updated source model $S'(S' \in \mathbf{S})$ of S, that is $S' = put(V', S)$. The readers can refer to [19–25] for more information about lens and its different types.

The intended behaviour of a lens is specified by the set of lens laws, in which three lens laws (PutGet, GetPut, PutPut) are important to ensure a coherent and predictable exchange between $\mathbf{S}$ and $\mathbf{V}$. As shown in table 1, the PutGet law states that any update to V is represented faithfully in S, therefore it confirms that the systems S and V are indeed synchronized under the *get* and *put*

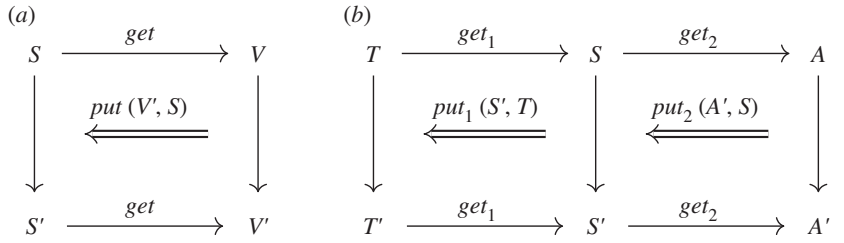

Figure 2. Lens and lenses' composition. (*a*) Lens $\mathbf{S} \rightleftarrows \mathbf{V}$. (*b*) Lenses composition $\mathbf{T} \rightleftarrows \mathbf{S} \rightleftarrows \mathbf{A}$.

Table 1. Lens laws.

laws	semantics
PutGet	$S = put(get(S), S)$
GetPut	$get(put(V', S)) = V'$
PutPut	$put(V'', put(V', S)) = put(V'', S)$

functions. The GetPut law represents that if V is not changed then neither is S. The PutPut law indicates that any update to V completely overwrites previous updates.

A lens is called *well-behaved* if it satisfies PutGet and GetPut laws; and a *well-behaved* lens is called *very well-behaved* if it also satisfies PutPut law.

When applying lenses in practical applications, such as database view updating, Haskell programs, systems interoperations, data sharing and model-driven engineering, compositionality is at the heart of lenses' applications. Given two lenses $l_1 : \mathbf{T} \rightleftarrows \mathbf{S}$ (with get_1 and put_1) and $l_2 : \mathbf{S} \rightleftarrows \mathbf{A}$ (with get_2 and put_2), the composition of l_1 and l_2 is $(l_1; l_2) : \mathbf{T} \rightleftarrows \mathbf{A}$, as shown in figure 2*b*. It consists of a new pair of *get* and *put* functions $get_{1\_2}(T) := get_2(get_1(T))$ and $put_{1\_2} := put_1(put_2(A', S), T)$. If both lenses satisfy a lens law, so does the composition. The composition of lenses is a powerful tool, as the updating of one dataset can affect all other datasets that it is related with, therefore compositing the lenses can ensure a coherent update synchronization across the whole system.

By analysing the characteristics of the bidirectional information flow and how information is transmitted and then updated between two models, one can identify that all aspects of such information flow match the structure of lenses. If combining figure 1*a*,*b* together, and making model A the source S and model C the view V, one can get the structure of a lens as shown in figure 2*a*. Therefore a basic information flow can be viewed as a concrete example of lenses, such that the structure of lenses can provide a mathematical foundation for bidirectional information flow.

3. Asymmetric delta lenses for vertical information flow

The notion of lens was developed originally as state-based, which concerns only the initial and final states of a system. It suffers from two problems in practice. The first problem is about the control of alignment [26]. To propagate changes from S to S', a tool should be used to align the two models. The alignment process should map elements in S' to the corresponding elements in S, such that changed elements in S' can be identified, and then the respective elements in S can be updated. Such alignment operation is computationally expensive and complex, and is usually based on fine-grained heuristics. As heuristics are not always reliable, model designers might prefer to use a different alignment strategy, or make corrections to automatically identified differences between S and S'. However, there is no room for such control of alignment with state-based lenses.

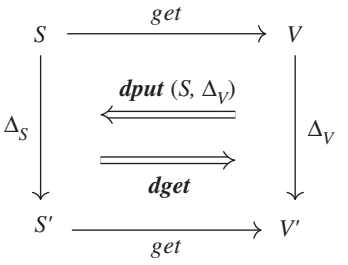

Figure 3. The structure of delta lens.

The second problem is the compatibility issue when composing multiple updates [26]. Owing to the complexity of the alignment operation, a single update propagation can become complicated and less predictable. Such a problem will be accumulated to the next lens after composition, and as such becomes more and more cumbersome whenever there are multiple compositions for models with complex structure.

(a) Delta lenses

To overcome the two above-mentioned issues, Diskin *et al.* [26] proposed a new type of lens, called the delta lens (*d*-lens). An inter-model *delta* was introduced to represent differences between a model and its updated version, followed by a delta propagation. A delta has a source model and a target model, and there may be multiple deltas between two models. For any model A, there is a special identity delta $\mathrm{id}_A: A \to A$ which specifies an idle (empty) update of model A resulting in the identity alignment of A to itself.

The delta propagation inputs and outputs not only models but also the deltas between them. In addition to the pair of *put* and *get* functions, *d*-lens introduces two higher order functions ***dget*** and ***dput***, as shown in figure 3. $\Delta_S : S \to S'$ is a set function which represents the difference from model S to its updated version S'. While the *get* function computes views from the source system, the ***dget*** translates deltas Δ_S in the source to deltas Δ_V in the view, that is $\Delta_V = \boldsymbol{dget}(\Delta_S)$. The backward update translation is given a higher-order function ***dput***, which takes a delta Δ_V in the view system and the original source S as input, and outputs a delta Δ_S in the source, that is $\Delta_S = \boldsymbol{dput}(\Delta_V, S)$. In other words, the ***dput*** not only considers the changes in the view, but also uses the original source system to retrieve information missing in the view.

The benefits of separating the process of identifying a delta (delta discovery) and delta propagation is that there are two distinguishing features of the delta lens over the state-based lenses. (1) It enables controlled alignment between the original and updated model. One can use different alignment strategies to generate a delta, and can also make corrections to any automatically discovered deltas. (2) As deltas produced by the ***dput*** can be passed to the input of other delta lenses, not only can it reduce the amount of expensive alignment processes, but also ensure correct composition at the same time.

(b) Delta lenses for information flow

Definition 3.1. Let $\alpha : \mathbf{S} \rightleftarrows \mathbf{V}$ and $\beta : \mathbf{V} \rightleftarrows \mathbf{T}$ be two delta lenses. The composition of α and β is a delta lens $(\alpha;\beta) : \mathbf{S} \rightleftarrows \mathbf{T}$. The ***dget*** function of $(\alpha;\beta)$ is the composition of the two ***dget*** functions, $\boldsymbol{dget}_{(\alpha;\beta)} = \boldsymbol{dget}_\alpha ; \boldsymbol{dget}_\beta$. The ***dput*** function of $(\alpha;\beta)$ is defined as follows. Let $\Delta_T : T \to T'$ be an update on T, such that $get_{(\alpha;\beta)}(S) = T$, here $T, T' \in \mathbf{T}$, that is, $get_\beta(V) = T$ with $V = get_\alpha(S)$, then $\boldsymbol{dput}_{(\alpha;\beta)}(\Delta_T, S) = \boldsymbol{dput}_\alpha(\Delta_V, S)$ with $\Delta_V = \boldsymbol{dput}_\beta(\Delta_T, V)$, that is $\boldsymbol{dput}_{(\alpha;\beta)} := \boldsymbol{dput}_\alpha(\boldsymbol{dput}_\beta(\Delta_T, V), S)$.

Table 2. Delta lens laws.

laws	semantics
GetId	$\boldsymbol{dget}(id_S) = id_V$, with $V = get(S)$
PutId	$id_S = \boldsymbol{dput}(id_V, S)$, with $V = get(S)$
PutGet	$\boldsymbol{dget}(\boldsymbol{dput}(\Delta_V, S)) = \Delta_v$
GetGet	$\boldsymbol{dget}(\Delta_S; \Delta_{S'}) = \boldsymbol{dget}(\Delta_S); \boldsymbol{dget}(\Delta_{s'})$
PutPut	$\boldsymbol{dput}((\Delta_V; \Delta_{V'}), S) = \boldsymbol{dput}(\Delta_V, S); \boldsymbol{dput}(\Delta_{V'}, S')$

As there are two types of information flow, vertical and horizontal, for each type the implementation of delta lenses will be different. For a vertical flow, the delta lens should be applied in an asymmetric way [26], and symmetrically [27] for a horizontal case. The asymmetric case will be the focus of our paper and the symmetric one will be studied in the future.

One of the major advantages of using delta lenses is that different delta lenses can be composed while maintaining the consistency of the semantics of the update procedures, thanks to the set of delta lens laws, as indicated in table 2. The GetId law requires that an idle update (identity delta) in the source model will translate into an idle update (identity delta) in the view; and the PutId says that an idle update in the view will generate an idle update in the source. In a similar sense to the PutGet law of the state-based lenses, the PutGet law on deltas also confirms the systems S and V are indeed synchronized under the ***dget*** and ***dput*** functions. The GetGet law ensures that the delta between two views arising from composition of updates in the source will be the same as computing delta incrementally over each update. Similarly, the PutPut law says that any new update $\Delta_{v'}$ in the view overwrites the previous update Δ_v.

A delta lens is called *well-behaved* if it satisfies GetId, PutId and PutGet laws; and a *well-behaved* delta lens is called *very well-behaved* if it also satisfies GetGet and PutPut laws.

Note that one does not require the GetPut law on deltas. As for an asymmetric delta lens, the view is generated from the source and therefore some information contained in the source might not be in the view, and some changes contained in the source delta cannot be retrieved from the ***dget*** function.

The above-mentioned laws provide a foundation for enabling transitions among many models in a rigorous manner via composition of delta lenses.

Here, GetGet and PutPut laws also ensure the composition of updates is rigorous during the composition of delta lenses. For example, two sequential updates of Δ_T and $\Delta_{T'}$ on T composed together under the PutPut laws can then use the backward propagation to get the composed Δ_S and $\Delta_{S'}$.

It is proved that the composition of two (very) well-behaved delta-lenses is also a (very) well-behaved delta lens.

During the composition, only the very first delta in a nesting of delta lenses needs to be computed. For example, in figure 4, updating a view in multiple steps and backward propagating the cumulative difference (composition of differences due to each update) produces the same result as backward propagating each difference in view individually and composing on the source side. Therefore, complex (multi-level modelling and nesting of lenses) information flow and synchronizations can become less computationally expensive and easier to manage, as the behaviour of the composed lenses is predictable under the set of delta lens laws.

4. Hybrid delta lenses = delta lenses + hybrid dput

Delta lenses can play a relevant role in the design and analysis of complex systems, supporting the development of digital models of real systems. In this scenario, decisions related to the requested level of details are fundamental, since they relate to the trade-off between the effort/cost to collect

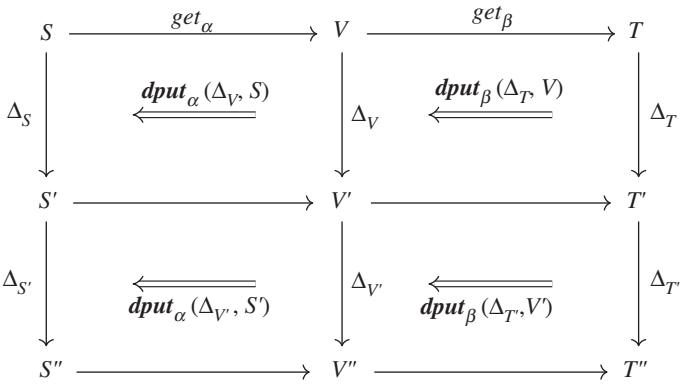

Figure 4. Delta lenses' composition.

data (e.g. installation of sensors and communication networks) and the desirable accuracy, and run elaborations (e.g. optimization, simulation, performance evaluation, planning, etc.).

Multi-fidelity models have been proposed to address these aspects [14], with the aim of exploiting multiple models of the same system with different levels of accuracy or complexity, and taking into consideration the specific availability of data and/or objectives of the analysis. Thus, while high-fidelity models (HFMs) consider the full set of features and information available for a manufacturing system, low-fidelity models (LFMs) are defined by reducing the level of details to enable rapid analyses and low computational load.

An LFM provides an abstraction of the HFM, entailing the need for mechanisms providing their reciprocal mapping and defining information and data flow to guarantee consistency.

The aforementioned asymmetric d-lens structure is applicable whenever the view embeds in the source faithfully, thus, it supports a fundamental class of abstraction taking a small part from a larger system. Nevertheless, different abstraction mechanisms can be needed in the modelling of a complex system. A very common one is the aggregation of a set of objects into a single one, entailing that the *get* function will be many-to-one, and there will be many possible ***dput*** functions for a given *get*, except if the mapping is defined by specific rules. While such rules can be difficult to define, a possible solution without defining the rules is to design an estimation algorithm to generate a possible mapping for the one-to-many relationship. Again, as there can be many estimation algorithms available, the way of hybridizing the ***dput*** with an arbitrary estimation algorithm might potentially affect the stability of a d-lens. To this end, a hybrid delta lenses structure is then proposed, and a formal representation of the hybrid delta lenses will be developed to support the generalization of its structure and properties.

(a) A formal representation of hybrid delta lenses

A hybrid delta lens, written as hd-lens, is defined under the formalization of category theory, as follows:

Definition 4.1. Let $l : \mathbf{K} \rightleftarrows \mathbf{L}$ be an asymmetric delta lens from a set of source models $\mathbf{K}$ to a set of view models $\mathbf{L}$, with $get : K \to L$, $put : K \times L \to K$, $\boldsymbol{dget} : [K, K] \to [L, L]$, and $\boldsymbol{dput} : K \times [L, L] \to [K, K]$, where $K \in \mathbf{K}$, $L \in \mathbf{L}$, $[K, K]$ the set of deltas from K to K, and $[L, L]$ is the set of functions (deltas) from L to L. Let $\boldsymbol{est}_Y^{K,L} : [K, K] \to [K, K]$ be an estimation algorithm which takes a delta (Δ_K) on K and computes an estimated new delta Δ_{K_Y}, here Δ_K, $\Delta_{K_Y} \in [K, K]$.

A hybrid delta lens $l_h : \mathbf{K} \rightleftarrows \mathbf{L}$, as shown in figure 5, is a new asymmetric delta lens constructed from l and the estimation algorithm $\boldsymbol{est}_Y^{K,L}$. The *get* function of l_h, denoted as get_h, is defined by the composition of a surjective mapping $\pi_1 : K \to K$ and *get*, that is $get_h(K) := get(\pi_1(K))$. The *put* function of l_h, is defined by the composition of an inverse of π_1 (which is to be decided by

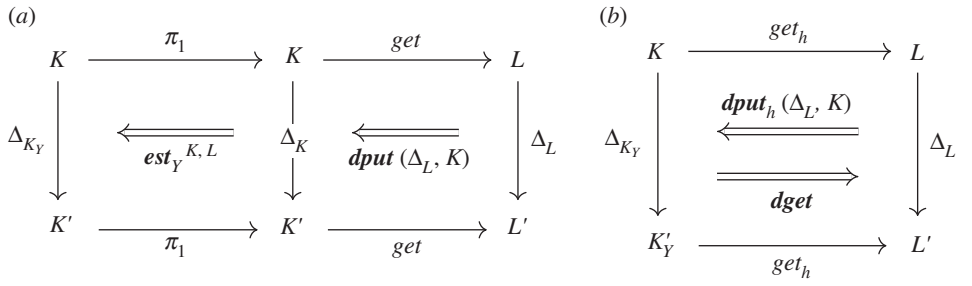

Figure 5. The structure of hybrid delta lens l_h: $\mathbf{H} \rightleftarrows \mathbf{L}$. (*a*) Hybrid delta lens—full map. (*b*) Hybrid delta lens—simple map.

$\boldsymbol{est}_Y^{K,L}$) and *put*, that is $put_h(L', K) := \pi_1^{-1}(put(L', K))$, with $L' \in \mathbf{L}$. The $\boldsymbol{dput}$ of l_h, denoted as $\boldsymbol{dput}_h$, is the sequential composition of $\boldsymbol{dput}$ and $\boldsymbol{est}_Y^{K,L}$, therefore $\boldsymbol{dput}_h(\Delta_L) := \boldsymbol{est}_Y^{K,L}(\boldsymbol{dput}(\Delta_L))$, where $\Delta_L \in [L, L]$.

Note that $\boldsymbol{dget}$ functions on l and l_h are the same, as the $\boldsymbol{est}_Y^{K,L}$ only deals with the backward propagation of update from L to K, and no estimation algorithm is required to generate L from K.

Similar to the set of delta lens laws, a set of hybrid delta lens laws are defined as follows:
GetId: $\boldsymbol{dget}(id_K) = id_L$, with $L = get(K)$
PutId: $id_K = \boldsymbol{dput}_h(id_L, K)$, with $L = get(K)$
PutGet: $\boldsymbol{dget}(\boldsymbol{dput}_h(\Delta_L, K)) = \Delta_L$
GetGet: $\boldsymbol{dget}(\Delta_K; \Delta_{K'}) = \boldsymbol{dget}(\Delta_K); \boldsymbol{dget}(\Delta_{K'})$
PutPut: $\boldsymbol{dput}_h((\Delta_L; \Delta_{L'}), K) = \boldsymbol{dput}_h(\Delta_L, K); \boldsymbol{dput}_h(\Delta_{L'}, K')$

Proposition 4.2. *Given a well-behaved asymmetric delta lens* $l : \mathbf{K} \rightleftarrows \mathbf{L}$ *and an estimation algorithm* $\boldsymbol{est}_Y^{K,L}$, *then the hybrid delta lens* $l_h : \mathbf{K} \rightleftarrows \mathbf{L}$ *is also a well-behaved asymmetric delta lens.*

Proof of proposition 4.2. From the definition of l_h we get $\boldsymbol{dget}$ on l_h is the same with $\boldsymbol{dget}$ on l, as $\boldsymbol{dget}(id_K) = id_L$, thus l_h satisfies GetId law.

From the definition we get $\boldsymbol{dput}_h(id_L, K) := \boldsymbol{est}_Y^{K,L}(\boldsymbol{dput}(id_L, K), K)$. As $\boldsymbol{dput}(id_L, K) = id_K$, and $\boldsymbol{est}_Y^{K,L}(id_K, K) = id_K$, we get $\boldsymbol{dput}_h(id_L, K) = id_K$, thus l_h satisfies PutId law.

We know $\boldsymbol{dput}_h(\Delta_L, K) := \boldsymbol{est}_Y^{K,L}(\boldsymbol{dput}(\Delta_L, K), K)$. Given $\boldsymbol{dput}(\Delta_L, K) = \Delta_K$, $\boldsymbol{dput}_h(\Delta_L, K) = \boldsymbol{est}_Y^{K,L}(\Delta_K, K) = \Delta_{K_Y}$. As $\boldsymbol{dget}$ on l_h is the same with $\boldsymbol{dget}$ on l, we have $\boldsymbol{dget}(\Delta_{K_Y}) = \Delta_L$, thus we have $\boldsymbol{dget}(\boldsymbol{dput}_h(\Delta_L, K)) = \Delta_L$, and l_h satisfies PutGet law. ■

Proposition 4.3. *Given a very well-behaved asymmetric delta lens* $l : \mathbf{K} \rightleftarrows \mathbf{L}$ *and an estimation algorithm* $\boldsymbol{est}_Y^{K,L}$, *the hybrid delta lens is also very well-behaved if the* $\boldsymbol{est}_Y^{K,L}$ *function satisfies PutPut law, that is* $\boldsymbol{est}_Y^{K,L}((\Delta_K, K); (\Delta_{K'}, K')) = \boldsymbol{est}_Y^{K,L}(\Delta_K, K); \boldsymbol{est}_Y^{K,L}(\Delta_{K'}, K')$.

Proof of proposition 4.3. If l is very well-behaved, l also satisfies PutPut and GetGet laws, and we get $\boldsymbol{dget}(\Delta_K; \Delta_{K'}) = \boldsymbol{dget}(\Delta_K); \boldsymbol{dget}(\Delta_{K'})$, and $\boldsymbol{dput}((\Delta_L; \Delta_{L'}), K) = \boldsymbol{dput}(\Delta_L, K); \boldsymbol{dput}(\Delta_{L'}, K)$.

We know $\boldsymbol{dget}$ on l_h is the same as $\boldsymbol{dget}$ on l, as we have $\boldsymbol{dget}(\Delta_K; \Delta_{K'}) = \boldsymbol{dget}(\Delta_K); \boldsymbol{dget}(\Delta_{K'})$, l_h also satisfies GetGet law.

We have $\boldsymbol{dput}_h((\Delta_L; \Delta_{L'}), K) := \boldsymbol{est}_Y^{K,L}(\boldsymbol{dput}((\Delta_L; \Delta_{L'}), K), K))$. From PutPut law on l we get $\boldsymbol{dput}((\Delta_L; \Delta_{L'}), K) = \boldsymbol{dput}(\Delta_L, K); \boldsymbol{dput}(\Delta_{L'}, K')$. Let $\boldsymbol{dput}(\Delta_L, K) = \Delta_K$ and $\boldsymbol{dput}(\Delta_{L'}, K') = \Delta_{K'}$, then we have $\boldsymbol{dput}_h((\Delta_L; \Delta_{L'}), K) = \boldsymbol{est}_Y^{K,L}((\Delta_K, K); (\Delta_{K'}, K'))$. As the $\boldsymbol{est}_Y^{K,L}$ function satisfies PutPut law, we have $\boldsymbol{dput}_h((\Delta_L; \Delta_{L'}), K) = \boldsymbol{est}_Y^{K,L}(\Delta_K, K); \boldsymbol{est}_Y^{K,L}(\Delta_{K'}, K')$. As $\boldsymbol{est}_Y^{K,L}(\Delta_K, K) = \boldsymbol{dput}_h(\Delta_L, K)$, and $\boldsymbol{est}_Y^{K,L}(\Delta_{K'}, K') = \boldsymbol{dput}_h(\Delta_{L'}, K')$, we can then get $\boldsymbol{dput}_h((\Delta_L; \Delta_{L'}), K) = \boldsymbol{dput}_h(\Delta_L, K); \boldsymbol{dput}_h(\Delta_{L'}, K')$, thus l_h satisfies PutPut law. ■

5. Implementing lenses in an assembly line digital twin

As introduced in §1, a digital twin model for a manufacturing system is defined in terms of different interacting sub-models designed to provide the desired level of details, matching the objective of the analysis or design phase. A major engineering problem in the instantiation of this class of models is coping with information and data coming from different sources and characterized by heterogeneous resolution and sampling rates. At the same time, different types of decisions require specific levels of modelling details. Thus, a digital twin of a real manufacturing system is likely to contain a mixture of different types of models and abstractions, as described in §4, and would benefit from the adoption of hybrid delta lenses.

The proposed mathematical foundation is tested and demonstrated to collect and elaborate monitoring data streamed from installed sensors in manufacturing systems that are modelled with a multi-scale approach [28]. The HFM describes all the resources in the system (e.g. machines and buffers), while an LFM is triggered by the position of the available sensors monitoring the flow of parts between the resources in the system. The actual flow of parts can be observed only where a sensor is installed. On the contrary, if no sensor is present, the interfaces among the resources cannot be observed, determining a simplified model of the system (LFM) where some resources are clustered into a single one.

An exemplary case is provided by an assembly line represented in figure 6. The shown assembly line is a simplified version of an automatic assembly line producing furniture components, consisting of 13 resources, specifically seven assembly machines (from M1 to M7) and six buffers (from B1 to B6). The main raw artefact (*partA*) enters machine M1, while another component (*compB*) to be assembled enters machine M2. Buffers are specified with transport time t_j and capacity c_j, and machines with service time s_j and the number of servers k_j. The property values of the resources are reported in table 3. All machines are single-server and the transport time of buffers is negligible.

Taking this assembly line (figure 6) as the high-fidelity model (HFM), a low-fidelity model (LFM) can be represented as a partition of the HFM where resources are grouped into mutually exclusive clusters.

(a) Implementation of backward transition: a hybrid dput function

Owing to the abstraction of multiple machine tools and buffers into single clusters, it is clear that the GetPut law cannot be satisfied. In this case, as shown in figure 8, the exact position of each artefact in the abstracted machine tools and buffers will not be determined, for example as M2 and M3 in the HFM have been aggregated into M2_1, at a certain time T_i, an artefact located in either M2 or M3 can be recovered, but the exact position is unclear.

Though the GetPut law cannot be satisfied, one can estimate where an artefact is by considering requirements and constraints imposed by capacity and minimum transition time. This estimation, which can be implemented via alternative algorithms, is denoted as $\boldsymbol{est}_Y^{H,L}$. In this case, under the hd-lens structure, as shown in figure 7, the $\boldsymbol{dput}_h$ function will be a sequence of two functions: a normal ***dput*** function to generate an intermediate delta: $\Delta_{H'} = \boldsymbol{dput}(\Delta_L, \mathrm{HFM})$; and an estimation algorithm $\boldsymbol{est}_Y^{H,L}$ to allocate each artefact's exact location on the aggregated machines/buffers, here Y represent different algorithms. The estimation algorithm $\boldsymbol{est}_Y^{H,L}$ takes $\Delta_{H'}$ and a collection of properties for the aggregated buffers/machines $\{c_j, t_j, k_j, s_j\}$ as input, and output a final delta Δ_H. The get_h is a sequential composition from π_1 to the get of its normal d-lens structure. Here π_1 maps each resource (M1 − M7 and B1 − B6) into its aggregated group. In the case shown in figure 8, π_1 presents the mappings of M2 → {M2, M3}, M3 → {M2, M3}, M1 → {M1, B1}, B1 → {M1, B1}, and so on.

Formally, written as

$$\Delta_H = \boldsymbol{dput}_h(\Delta_L, \mathrm{HFM}) = \boldsymbol{est}_Y^{H,L}(\boldsymbol{dput}(\Delta_L, \mathrm{HFM}), \{c_j, t_j, k_j, s_j\}). \tag{5.1}$$

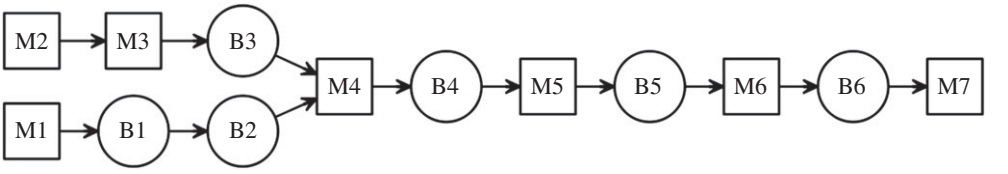

Figure 6. Representation of the assembly line. The squares denote machines and the circles as buffers.

Table 3. Properties of (*a*) machines and (*b*) buffers in the assembly line.

(*a*) machines			(*b*) buffers	
ID	operation	service time (s)	ID	capacity
M1	machining	5	B1	5
M2	machining	3	B2	4
M3	machining	3	B3	10
M4	assembly	10	B4	7
M5	machining	11	B5	6
M6	machining	12	B6	7
M7	machining	13		

As shown in figure 8, for example, whenever a part (compB.1) exits from M2_1 (representing the aggregation of M2 and M3), the time it leaves M2 to enter M3 must be determined by an estimation algorithm, exploiting the available information, i.e. the number of servers k_2 and k_3, service times s_2 and s_3, and capacity c_2 and c_3 for M2 and M3, respectively.

To support the testing and validation of the described approach, two estimation algorithms have been used to implement the $\boldsymbol{dput}_h$ function.

(i) $dput_h$ by queuing reasoning

An estimation algorithm can be obtained based on queuing theory. Generally speaking, a part enters an element k (a buffer or a machine) at time t_k and leaves it at time t'_k. The resulting lead time $t'_k - t_k$ can be partitioned in a time waiting in the queue w_k and service time s_k. Let t_A and t'_A be the enter and exit time for a part $part_x$ flowing through an aggregated set of elements A_l described by an ordered list $\mathcal{L}_A$ of buffers and machines. While the service times for each element k in the list are known, the waiting times can be estimated through the following:

$$w_k = \left(\frac{c_i}{k_i}\sum_{i\in\mathcal{L}}\frac{c_i}{k_i}\right) \times \left(t' - t - \sum_{k\in\mathcal{L}} s_i\right) \tag{5.2}$$

Thus, the enter and exit times of each part in an element $h \in A$ can be estimated, starting from t, considering the waiting and service time for all the elements preceding h in the ordered list $\mathcal{L}_A$. Finally, starting from this estimation, a further step applied to fine-tune the enter and exit times to match the capacity limit of each buffer/machine. At each time stamp, the number of parts located within a buffer/machine is calculated and, if the capacity limit of the buffer/machine is violated, the last part entered is moved back to the previous element until there is a vacancy in the next buffer/machine, granted that the preceding element belongs to A.

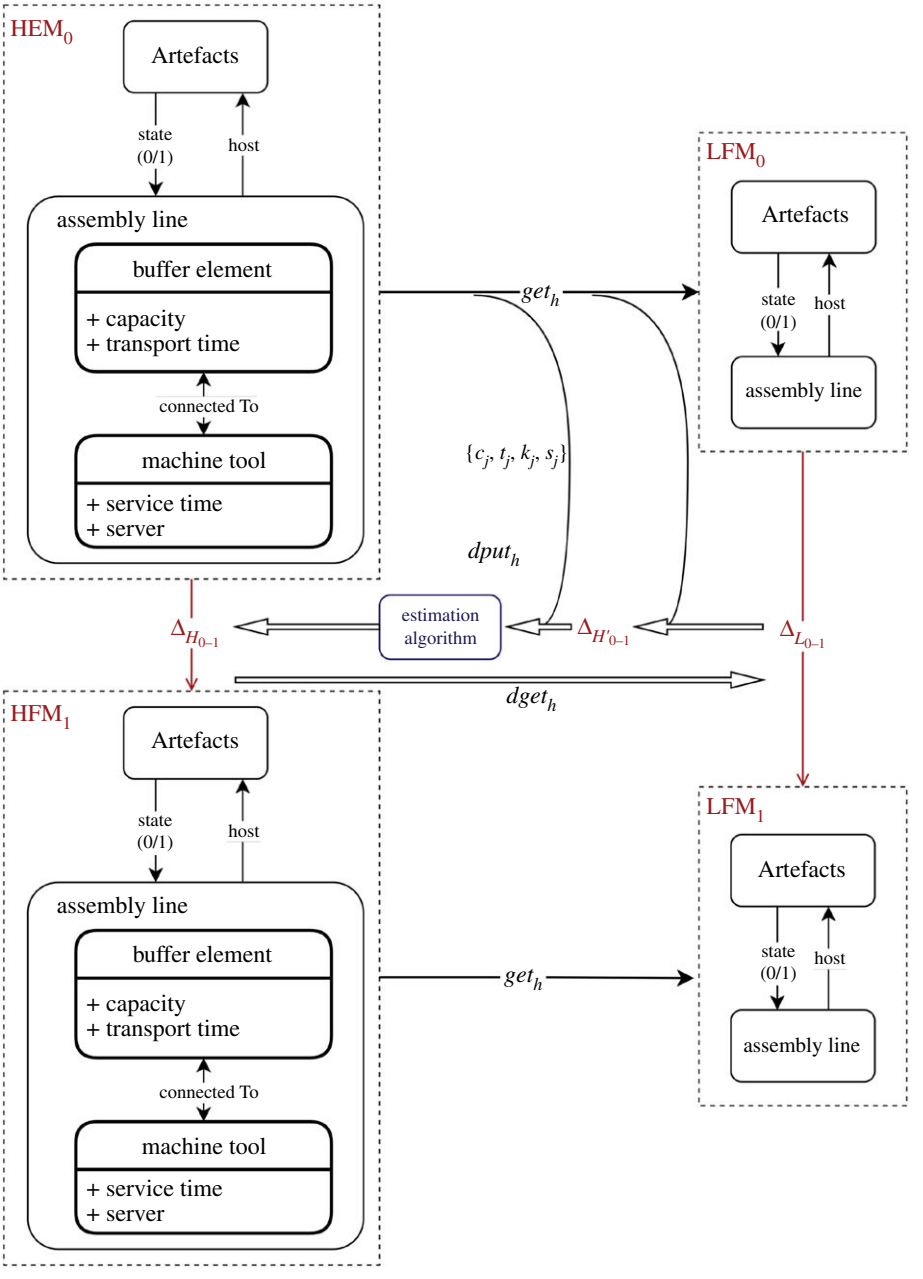

Figure 7. The hd-lens structure between the HFM and LFM. (Online version in colour.)

(ii) $dput_h$ by forward scheduling

Another estimation algorithm, based on forward scheduling [29], is also developed to calculate the earliest enter/exit times of parts for each element $k \in A$. Thus, a permutation flow shop scheduling approach is used. Modelled as a flow shop, the portion of the system represented by A, to obtain the earliest time schedule, i.e. the earliest t'_k for each part matching both t and t' as well as the constraints related to the element and the process (capacity, number of servers and service times) are all modelled.

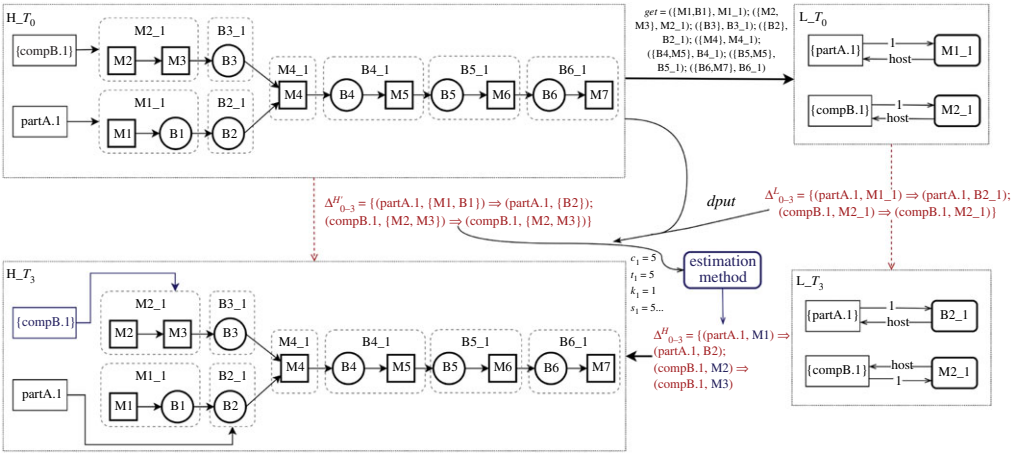

Figure 8. The hd-lens structure in the example—using an estimation method. (Online version in colour.)

(b) Performance evaluation of hd-lenses for multi-fidelity transitions

The performance of the hd-lens for these multi-fidelity transitions is evaluated in terms of the differences between the actual high-fidelity data and the result of the estimated backward mapping. The difference can be evaluated only if the high-fidelity log is available. Typically, this log can be the output of simulation during the design phase of the manufacturing system. The reconstructed high-fidelity log and actual high-fidelity log are compared by calculating the difference between the actual T_i and the estimated T_i' entry time of each part at each machine/buffer; furthermore, it is possible to calculate the difference between the actual number of parts N_{Ti} and the estimated number of parts N'_{Ti} for each T_i on each machine/buffer.

The performance indicator $|\overline{\Delta}|_j$ denotes the mean absolute difference between the actual and estimated number of parts on resource j during the entire log; $|\overline{\Delta}|_{\text{tot}}$ represents the sum of the $|\overline{\Delta}|_j$ for all the resources j and it is used to measure the global magnitude of the average errors obtained after applying the $\boldsymbol{dput}_h$ function. Similarly, given $\sigma(\Delta)_j$ the standard deviation of the difference between the actual and estimated number of parts in resource j, σ_{tot} denotes their sum over all the resources in the system, which offers a measurement of the global variability of the estimation error. Finally, Δ_{UB}^{LB} specifies the minimum and maximum deviation between actual and estimated number of parts over all system resources. The described evaluation is operated for the two estimation approaches considered, namely Method I based on queuing reasoning and Method II using forward scheduling.

Considering all the possible partitions, a total of 511 LFMs have been generated for the assembly system defined in figure 6 and table 3. The performance of the proposed approach has been assessed by elaborating the simulation log of 200 parts flowing through the assembly system.

Table 4 reports an excerpt of the results, where columns *ID*, *Agg*, *Clusters* and *LFM* stipulate the identifier of the LFM, the number of aggregated resources, the size of each aggregation cluster and the configuration of the LFM using round brackets to identify clusters. For instance, LFM n.5 (shown in figure 9*a*) has only one cluster with three resources (B5, M6, B6), thus there are two aggregated resources ($Agg = 2$) with respect to the HFM. The performance of the two $\boldsymbol{dput}_h$ functions is reported in columns Method I and Method II.

The aggregation number can be a benchmark to compare the performance of the estimation on both methods. In general, the larger the aggregation numbers, the greater the estimated errors are expected to be. However, there are a few exceptions. For example, LFM n.5 with a low aggregation ($Agg = 2$) and a single cluster has larger errors on both Method I and II compared with other LFMs with equal or even larger aggregation (e.g. n.75 as shown in figure 9*c*). This can be at least partially

Table 4. Low-fidelity models.

				Method I			Method II		
ID	Agg	Clusters	LFM	$\lvert\overline{\Delta}\rvert_{\text{tot}}$	σ_{tot}	Δ_{UB}^{LB}	$\lvert\overline{\Delta}\rvert_{\text{tot}}$	σ_{tot}	Δ_{UB}^{LB}
5	2	3	{M1},{B1},{B2},{M2},{M3},{B3}, {M4},{B4},{M5},{B5,M6,B6},{M7}	1.9516	1.8979	−3, 3	5.9046	5.2409	−7, 7
11	2	3	{M1},{B1},{B2},{M2},{M3},{B3}, {M4},{B4},{M5,B5,M6},{B6},{M7}	0.4793	1.1050	−1, 2	0.0015	0.0642	−1, 1
21	3	2;3	{M1},{B1},{B2},{M2},{M3},{B3}, {M4},{B4,M5},{B5,M6,B6},{M7}	1.9867	2.3742	−3, 3	5.9050	5.2683	−7, 7
23	2	3	{M1},{B1},{B2},{M2},{M3},{B3}, {M4},{B4,M5,B5},{M6},{B6},{M7}	1.1370	1.7662	−3, 3	0.0004	0.0291	−1, 1
30	5	6	{M1},{B1},{B2},{M2},{M3},{B3},{M4}, {B4,M5,B5,M6,B6,M7}	2.7019	3.2362	−3, 3	6.0778	6.5980	−7,7
63	1	2	{M1},{B1},{B2},{M2,M3},{B3},{M4}, {B4},{M5},{B5},{M6},{B6},{M7}	0.0000	0.0000	0,0	0.0003	0.0262	−1, 1
75	3	2;3	{M1},{B1},{B2},{M2,M3},{B3},{M4}, {B4},{M5,B5,M6},{B6},{M7}	0.3982	1.0157	−1, 2	0.0018	0.0904	−1, 1
123	5	3;4	{M1},{B1},{B2},{M2,M3,B3},{M4}, {B4,M5,B5,M6},{B6},{M7}	2.8636	3.5138	−3, 3	0.0018	0.0947	−1, 1
190	7	2;2;6	{M1},{B1,B2},{M2},{M3,B3}, {M4},{B4,M5,B5,M6,B6,M7}	4.8996	5.5505	−3, 3	6.0785	6.6504	−7, 7
202	5	2;2;2;3	{M1},{B1,B2},{M2,M3},{B3},{M4}, {B4},{M5,B5},{M6,B6,M7}	1.6620	2.7764	−3, 2	0.2127	0.8066	−1, 2
208	4	2;2;2;2	{M1},{B1,B2},{M2,M3},{B3},{M4}, {B4,M5},{B5},{M6},{B6,M7}	1.5282	2.4171	−3, 2	0.0016	0.1111	−1, 1
222	7	2;2;6	{M1},{B1,B2},{M2,M3},{B3},{M4}, {B4,M5,B5,M6,B6,M7}	3.8589	4.5863	−3, 3	6.0785	6.6504	−7, 7
318	7	2;2;6	{M1,B1},{B2},{M2},{M3,B3},{M4}, {B4,M5,B5,M6,B6,M7}	3.9315	4.8043	−3, 3	6.0785	6.6507	−7, 7
340	5	2;2;2;2;2	{M1,B1},{B2},{M2,M3},{B3},{M4}, {B4,M5},{B5,M6},{B6,M7}	0.7458	2.2933	−1, 1	0.0030	0.1645	−1, 1
350	7	2;2;6	{M1,B1},{B2},{M2,M3},{B3},{M4}, {B4,M5,B5,M6,B6,M7}	2.8619	3.8517	−3, 3	6.0785	6.6507	−7, 7
411	5	3;4	{M1,B1,B2},{M2},{M3},{B3},{M4}, {B4,M5,B5,M6},{B6},{M7}	2.4762	3.8494	−3, 3	0.0018	0.0966	−1, 1
502	8	3;3;2;4	{M1,B1,B2},{M2,M3,B3},{M4}, {B4,M5},{B5,M6,B6,M7}	5.0667	5.8518	−3, 3	5.9059	5.4666	−7, 7
508	8	3;3;4;2	{M1,B1,B2},{M2,M3,B3},{M4}, {B4,M5,B5,M6},{B6,M7}	4.0486	5.6866	−3, 3	0.0027	0.1567	−1, 1
510	9	3;3;6	{M1,B1,B2},{M2,M3,B3},{M4}, {B4,M5,B5,M6,B6,M7}	5.5294	6.4392	−3, 3	6.0785	6.6575	−7, 7

explained by the fact that LFM n.5 aggregation ($\{B5, M6, B6\} \rightarrow B5\_1$) includes two buffers with large capacity.

LFM n.23 (shown in figure 9*b*) has an aggregation comparable to LFM n.5 with a similar error on Method I, but with a much lower error on Method II. This case demonstrates the impact that

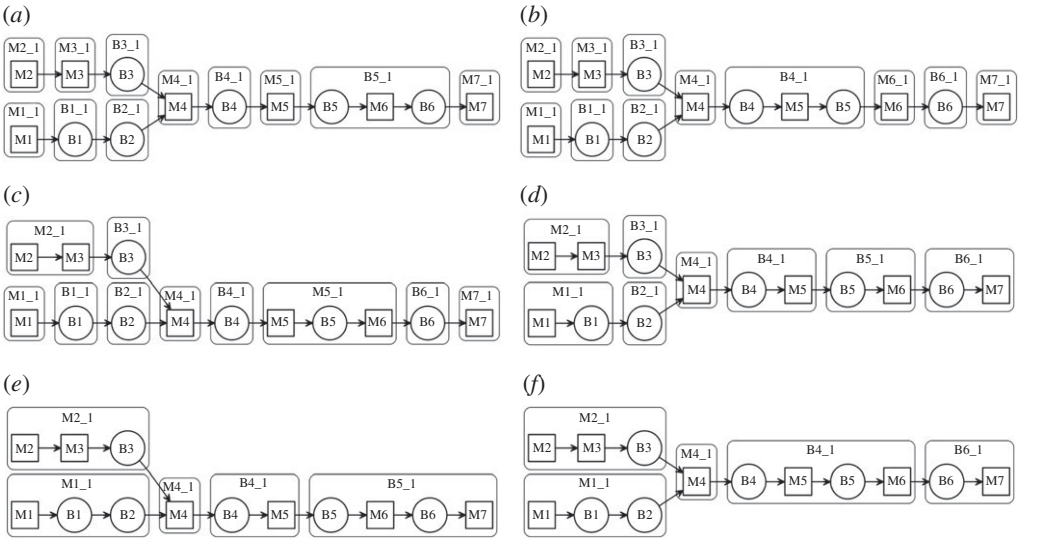

Figure 9. Examples of different aggregations of the assembly line. (*a*) LFM n.5, (*b*) LFM n.23, (*c*) LFM n.75, (*d*) LFM n.340, (*e*) LFM n.502 and (*f*) LFM n.508.

the specific estimation algorithm may have on the performance of ***dput***. Similarly, LFMs n.502 (figure 9*e*) and n.508 (figure 9*f*) have the same type of clusters, but different estimation errors, in particular for Method II. This difference is explained by the fact that Method II can be more or less influenced by the transient behaviour, depending on the position and size of the clusters.

Furthermore, it can be observed that typically the estimation error is lower when the aggregated resources are split into smaller clusters. This is particularly true for Method I (e.g. LFMs n.123, n.202, n.340), and less evident for Method II (e.g. LFMs n.123 and n.202 are an exception).

(c) Discussion

The results show that the approximation of an LFM can be accessed, thus providing a valuable tool to select the most suited LFM with considerations of requirements related to accuracy. It must be highlighted that the choice of the estimation algorithm may have a significant impact on the approximation error of LFMs. Herein, Method I is less sensitive towards the transient changes of the system because the estimation is based on a fusion of the scheduling and queuing approaches. However, both the methods suffer from an accumulated delayed effect, as more estimated errors will be accumulated towards the end of the assembly line when an increasing number of parts have been loaded into the system. It might be possible to eliminate such effect via an advanced estimation method which is tailored for the specific characteristics of the system under study. The estimation error can be further reduced using more sophisticated methods, but this is out of the scope of this paper. The two estimation methods have been adopted out of simplicity to test and to validate the stability of the hd-lens structure on the assembly line case, providing that the estimation algorithm complies with the structural properties of the constructed hd-lens.

The developed hd-lens structure provides a formal way to construct bidirectional information flow between multi-level digital models. The lens structure itself does not enclose any information on how to develop the low-fidelity model, more specifically, defining the *get* function.

The experiment results from the designated 511 aggregations provide a comprehensive analysis of all possible partitions of the 13 connected buffers/machines. Regardless of the estimation method used, there are a number of patterns which can be summarized: (1) if

the aim is to minimize the number of sensors used on the assembly line, large aggregation can be considered with as many clusters as possible to reduce the overall estimation error; (2) if the aim is to minimize the estimation error, it is better to avoid aggregating several resources with relatively large capacities (i.e. it is better to aggregate machines than buffers).

6. Conclusion and future work

To address the growing challenges of managing complexity imposed by ever-advancing digital technologies in manufacturing, designing and operating digital models under multi-levels of abstraction is one of the solutions. There is however a lack of mathematical foundations to support information flow between models with different levels of detail in an efficient, coherent and stable manner. In this paper, a mathematical structure, called Delta lens, has been explored and used to provide mathematical foundations to support rigorous bidirectional transitions between digital models with different fidelities. In order to support different types of abstractions in manufacturing, e.g. aggregation of many objects into a single one, a hybrid delta lenses structure (hd-lenses) has been proposed and its formal representation is developed to support the generalization of its structure and properties. To validate the proposed formal framework, the proposed hd-lenses structure was then tested and demonstrated via a multi-level digital model of an assembly line. An hd-lens has been constructed to support information flow between a high-fidelity model and a low-fidelity model of the system. In the experiment, the performance of 511 different LFMs, communicating with the same HFM, has been analysed under two different estimation methods. The results have shown that even though the estimation methods behave differently in terms of errors, the general properties of the constructed hd-lens are rigorously complied. This gives us insight into finding the most suitable abstractions for the LFM, providing a way to aid decision-making in sensor configurations on the assembly line.

The proposed formal structure to support information flow between models with different fidelities paves the way to many possible further studies. As the hd-lens allows different estimation algorithms to be employed under the constraint of the hd-lens properties, to further refine the demonstrated test case, future work might involve exploration of the estimation algorithms and finding the optimal algorithms based on the specific application. Theoretically, the power of compositionality of *d*-lenses and the proposed hd-lenses have been discussed but not yet tested. Future work will include exploration of the compositionality of hd-lenses to support more complex transitions between multi-level digital models. Furthermore, this paper has a particular focus on vertical information flow between models with different fidelities; the next major focus will be the study of Delta lenses and other structures to provide formal foundations for horizontal information flow.

Data accessibility. Data used in this article are available from the authors upon request. Data and code: https://github.com/Stefanie8283/hd-lens-in-manufacturing.

Authors' contributions. Q.Q.: conceptualization, formal analysis, funding acquisition, methodology, resources, software, validation, writing—original draft, writing—review and editing; W.T.: conceptualization, formal analysis, methodology, software, validation, writing—original draft, writing—review and editing; M.U.: conceptualization, formal analysis, methodology, software, validation, writing—original draft, writing—review and editing; X.J.: conceptualization, formal analysis, methodology, resources, validation, writing—review and editing; P.J.S.: conceptualization, formal analysis, methodology, resources, validation, writing—review and editing.

All authors gave final approval for publication and agreed to be held accountable for the work performed therein.

Conflict of interest declaration. We declare that we have no competing interests.

Funding. The Huddersfield team (Q.Q., X.J. and P.J.S.) would like to acknowledge the funding support from the EPSRC UKRI Innovation Fellowship (ref. EP/S001328/1), EPSRC Future Advanced Metrology Hub (ref. EP/P006930/1) and EPSRC Fellowship in Manufacturing (ref. EP/R024162/1).

References

1. Boccara N. 2010 *Modeling complex systems*. 2nd edn. Graduate Texts in Physics. New York, NY: Springer.
2. Aumann CA. 2007 A methodology for developing simulation models of complex systems. *Ecol. Model.* **202**, 385–396. (doi:10.1016/j.ecolmodel.2006.11.005)
3. Philbeck T, Davis N. 2018 The fourth industrial revolution. *J. Int. Aff.* **72**, 17–22.
4. Terkaj W, Gaboardi P, Trevisan C, Tolio T, Urgo M. 2019 A digital factory platform for the design of roll shop plants. *CIRP J. Manuf. Sci. Technol.* **26**, 88–93. (doi:10.1016/j.cirpj.2019.04.007)
5. Gödri I, Kardos C, Pfeiffer A, Váncza J. 2019 Data analytics-based decision support workflow for high-mix low-volume production systems. *CIRP Ann.* **68**, 471–474. (doi:10.1016/j.cirp.2019.04.001)
6. Terkaj W, Tolio T, Urgo M. 2015 A virtual factory approach for in situ simulation to support production and maintenance planning. *CIRP Ann.* **64**, 451–454. (doi:10.1016/j.cirp.2015.04.121)
7. Ma J, Chen H, Zhang Y, Guo H, Ren Y, Mo R, Liu L. 2020 A digital twin-driven production management system for production workshop. *Int. J. Adv. Manuf. Technol.* **110**, 1385–1397. (doi:10.1007/s00170-020-05977-5)
8. Jain S, Shao G, Shin SJ. 2017 Manufacturing data analytics using a virtual factory representation. *Int. J. Prod. Res.* **55**, 5450–5464. (doi:10.1080/00207543.2017.1321799)
9. Frazzon EM, Kück M, Freitag M. 2018 Data-driven production control for complex and dynamic manufacturing systems. *CIRP Ann.* **67**, 515–518. (doi:10.1016/j.cirp.2018.04.033)
10. Urgo M, Terkaj W. 2020 Formal modelling of release control policies as a plug-in for performance evaluation of manufacturing systems. *CIRP Ann.* **69**, 377–380. (doi:10.1016/j.cirp.2020.04.007)
11. Stark R, Damerau T. 2019 *Digital twin*, pp. 1–8. Berlin, Germany: Springer.
12. Jiang Y, Yin S, Li K, Luo H, Kaynak O. 2021 Industrial applications of digital twins. *Phil. Trans. R. Soc. A* **379**, 20200360. (doi:10.1098/rsta.2020.0360)
13. Vogel-Heuser B, Ocker F, Weiß I, Mieth R, Mann F. 2021 Potential for combining semantics and data analysis in the context of digital twins. *Phil. Trans. R. Soc. A* **379**, 20200368. (doi:10.1098/rsta.2020.0368)
14. Forrester AI, Sóbester A, Keane AJ. 2007 Multi-fidelity optimization via surrogate modelling. *Proc. R. Soc. A* **463**, 3251–3269. (doi:10.1098/rspa.2007.1900)
15. Alexander F, Andras Sobester AK. 2008 *Engineering design via surrogate modelling: a practical guide*, 1st edn. New York, NY: Wiley.
16. Scott PJ. 2004 Pattern analysis and metrology: the extraction of stable features from observable measurements. *Proc. R. Soc. Lond. A* **460**, 2845–2864. (doi:10.1098/rspa.2004.1291)
17. Forza C, Salvador F. 2001 Information flows for high-performance manufacturing. *Int. J. Prod. Econ.* **70**, 21–36. (doi:10.1016/S0925-5273(00)00038-4)
18. Bohannon A, Pierce BC, Vaughan JA. 2006 Relational lenses: a language for updatable views. In *Proc. of the Twenty-Fifth ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems, Chicago, IL, 26–28 June 2006*, pp. 338–347. New York, NY: ACM.
19. Johnson M, Rosebrugh RD. 2016 Unifying set-based, delta-based and edit-based lenses. *Bx@ ETAPS* **1571**, 1–13.
20. Johnson M, Rosebrugh RD. 2017 Symmetric delta lenses and spans of asymmetric delta lenses. *J. Object Technol.* **16**, 2–1. (doi:10.5381/jot.2017.16.1.a2)
21. Johnson M, Rosebrugh R. 2013 Delta lenses and opfibrations. *Electron. Commun. EASST* **57**, 1–18.
22. Foster JN, Greenwald MB, Moore JT, Pierce BC, Schmitt A. 2007 Combinators for bidirectional tree transformations: a linguistic approach to the view-update problem. *ACM Trans. Program. Lang. Syst. (TOPLAS)* **29**, 17-es. (doi:10.1145/1232420.1232424)
23. Ahman D, Uustalu T. 2017 Taking updates seriously. In BX@ ETAPS, pp. 59–73.
24. Clarke B. 2020 Internal lenses as functors and cofunctors. Preprint. (https://arxiv.org/abs/2009.06835v1)
25. Clarke B. 2020 Internal split opfibrations and cofunctors. Preprint. (https://arxiv.org/abs/2004.00187)
26. Diskin Z, Xiong Y, Czarnecki K. 2011 From state-to delta-based bidirectional model transformations: the asymmetric case. *J. Obj. Technol.* **10**, 1–25. (doi:10.5381/jot.2011.10.1.a6)

27. Diskin Z, Xiong Y, Czarnecki K, Ehrig H, Hermann F, Orejas F. 2011 From state-to delta-based bidirectional model transformations: the symmetric case. In *Int. Conf. on Model Driven Engineering Languages and Systems, Wellington, NZ, 16–21 October 2011*, pp. 304–318. Berlin, Germany: Springer.
28. Terkaj W, Qi Q, Urgo M, Scott PJ, Jiang X. 2021 Multi-scale modelling of manufacturing systems using ontologies and delta-lenses. *CIRP Ann.* **70**, 361–364. (doi:10.1016/j.cirp.2021.04.047)
29. Demeulemeester EL, Herroelen WS. 2002 *Project scheduling—a research handbook*. International Series in Operations Research & Management Science. Boston, MA: Springer.