

A Compositional Sheaf-Theoretic Framework for Event-Based Systems

Gioele Zardini

ETH Zürich
Zürich, Switzerland
gzardini@ethz.ch

David I. Spivak

Massachusetts Institute of Technology
Cambridge, MA, USA
dspivak@gmail.com

Andrea Censi

ETH Zürich
Zürich, Switzerland
acensi@ethz.ch

Emilio Frazzoli

ETH Zürich
Zürich, Switzerland
efrazzoli@ethz.ch

A compositional sheaf-theoretic framework for the modeling of complex event-based systems is presented. We show that event-based systems are machines, with inputs and outputs, and that they can be composed with machines of different types, all within a unified, sheaf-theoretic formalism. We take robotic systems as an exemplar of complex systems and rigorously describe actuators, sensors, and algorithms using this framework.

1 Introduction

This paper presents a unified modeling framework for event-based systems, focusing on the standard example of cybernetic systems. Specifically, we present event-based systems as machines, showing how to compose them in various ways and how to describe diverse interactions (between systems that are continuous, event-based, synchronous, etc). We showcase the efficacy of our framework by using it to describe a robotic system, and we demonstrate how apparently different robotic components, such as actuators, sensors, and algorithms, can be described within a common sheaf-theoretic formalism.

Related Work. Cyber-physical systems (CPSs), first mentioned by Wiener in 1948 [21], consist of an *orchestration of computers and physical systems* [9] allowing the solution of problems which neither part could solve alone. CPSs are complex systems including physical components, such as actuators, sensors, and computer units, and software components, such as perception, planning, and control modules. Starting from the first mention of CPSs, their formal description has been object of studies in several disciplines, such as control theory [2, 20, 5, 7], computer science [8, 16, 15, 10, 11], and applied category theory [1, 17, 18].

Traditionally, control theorists approached CPSs through the study of hybrid systems, analyzing their properties and proposing strategies to optimally control them. In [2], researchers identify phenomena arising in real-world hybrid systems and introduce a mathematical model for their description and optimal control. In [20], a compositional framework for the abstraction of discrete, continuous, and hybrid systems is presented, proposing constructions to obtain hybrid control systems. Furthermore, [5] defines event-driven control strategies for multi-agent systems, and [7] proposes an introduction to event- and self-triggered control systems, which are proactive and perform sensing and actuation when needed.

On the other hand, computer scientists focused on the study of verification techniques for CPSs. In [8], the researcher introduces a comprehensive theory of hybrid automata, and focuses on tools for the reliability analysis of safety-critical CPSs. Similarly, [16, 15] develop a differential dynamic logic for hybrid systems and introduce deductive verification techniques for their safe operation. Furthermore, [10, 11] underline that to achieve simplicity and understandability, *clear, deterministic modeling semantics*

have proven valuable in modeling CPSs, and suggest the use of modal models, ensuring that models are used within their validity regime only (e.g. discrete vs. continuous).

In 2006, [1] proposes the first category-theoretic framework for the study of hybrid systems, defining the category of hybrid objects and applying it to the study of bipedal robotic walking. Finally, [19, 17, 18] present temporal type theory and machines, which are shown to be able to describe discrete and continuous dynamical systems, and which lay the foundation for this work.

Organization of the Paper. Section 2 recalls the theory of sheaves and machines presented in [18] and includes explanatory examples. Section 3 shows how to model event-based systems within this framework, and provides practical tools and examples. Section 4 showcases the properties of the defined framework, by employing it to model the feedback control of a flying robot.

Acknowledgements We would like to thank Dr. Paolo Perrone for the fruitful discussions. DS acknowledges support from AFOSR grants FA9550-19-1-0113 and FA9550-17-1-0058.

2 Background

The reader is assumed to be familiar with category theory. An extended, self-contained version of this article is reported in [22]. In this section, we review the material needed to present Section 3. This paper builds on the theory presented in [17, 18]. Let $\mathbb{R}_{\geq 0}$ denote the linearly ordered poset of non-negative real numbers. For any $a \in \mathbb{R}_{\geq 0}$, let

$$\begin{aligned} \text{Ph}_a : \mathbb{R}_{\geq 0} &\rightarrow \mathbb{R}_{\geq 0} \\ \ell &\mapsto a + \ell. \end{aligned}$$

denote the translation-by- a function.

Definition 2.1 (Category of continuous intervals Int). The *category of continuous intervals* Int is composed of:

- Objects: $\text{Ob}(\text{Int}) := \mathbb{R}_{\geq 0}$. We denote such an object by ℓ and refer to it as a *duration*.
- Morphisms: Given two durations ℓ and ℓ' , the set $\text{Int}(\ell, \ell')$ of morphisms between them is

$$\text{Int}(\ell, \ell') := \{\text{Ph}_a \mid a \in \mathbb{R}_{\geq 0} \text{ and } a + \ell \leq \ell'\}.$$

- The identity morphism on ℓ is the unique element $\text{id}_\ell := \text{Ph}_0 \in \text{Int}(\ell, \ell)$.
- Given two morphisms $\text{Ph}_a : \ell \rightarrow \ell'$ and $\text{Ph}_b : \ell' \rightarrow \ell''$, we define their composition as $\text{Ph}_a \circ \text{Ph}_b = \text{Ph}_{a+b} \in \text{Int}(\ell, \ell'')$.

We often denote the interval $[0, \ell] \subseteq \mathbb{R}$ by $\tilde{\ell}$. A morphism Ph_a can be thought of as a way to include $\tilde{\ell}'$ into $\tilde{\ell}$, starting at a , i.e. the subinterval $[a, a + \ell'] \subseteq [0, \ell]$.¹

Proposition 2.2. Int is indeed a category: It satisfies associativity and unitality.

¹ Int is not just the category of intervals $[a, b]$ with inclusions, which we temporarily call Int' ; in particular Int' is a poset, whereas Int is not. For experts: Int' is the twisted arrow category of the poset (\mathbb{R}, \leq) , whereas Int is the twisted arrow category of the monoid $(\mathbb{R}, +, 0)$ as a category with one object.

Definition 2.3 (Int-presheaf). An *Int-presheaf* A is a functor

$$A: \text{Int}^{\text{op}} \rightarrow \text{Set},$$

where Set is the category of sets and functions. Given any duration ℓ , we refer to elements $x \in A(\ell)$ as *length- ℓ sections* (*behaviors*) of A . For any section $x \in A(\ell)$ and any map $\text{Ph}_a: \ell' \rightarrow \ell$ we write $x|_{[a, a+\ell']}$ to denote its *restriction* $A(\text{Ph}_a)(x) \in A(\ell')$.

Definition 2.4 (Compatible sections). If A is an Int-presheaf, we say that sections $a \in A(\ell)$ and $a' \in A(\ell')$ are *compatible* if the right endpoint of a matches the left endpoint of a' , i.e.:

$$a|_{[\ell, \ell]} = a'|_{[0, 0]}.$$

Definition 2.5 (Int-sheaf). An Int-presheaf

$$P: \text{Int}^{\text{op}} \rightarrow \text{Set}.$$

is called an *Int-sheaf* if, for all ℓ, ℓ' and compatible sections $p \in P(\ell)$, $p' \in P(\ell')$ (i.e., with $p|_{[\ell, \ell]} = p'|_{[0, 0]}$), there exists a unique $\bar{p} \in P(\ell + \ell')$ such that

$$\bar{p}|_{[0, \ell]} = p \quad \text{and} \quad \bar{p}|_{[\ell, \ell + \ell']} = p'.$$

Morphisms of Int-sheaves are just morphisms of their underlying Int-presheaves. We denote the category of Int-sheaves as $\text{Shv}(\text{Int}) := \widetilde{\text{Int}}$.

Example 2.6 (Initial and terminal objects in $\widetilde{\text{Int}}$). The terminal object in $\widetilde{\text{Int}}$ is called 1 , and it assigns to each interval ℓ the one-element set $\{1\}$. Similarly $\emptyset \in \widetilde{\text{Int}}$ is the initial object and sends each interval ℓ to the empty set \emptyset .

Example 2.7 (Period- d clock). For any $d > 0$ define Clock_d to be the presheaf with

$$\text{Clock}_d(\ell) := \{ \{t_1, \dots, t_n\} \subseteq \tilde{\ell} \mid t_1 < d, \ell - t_n < d, \text{ and } t_{i+1} - t_i = d \text{ for all } 1 \leq i \leq n-1 \}.$$

Note that $(\ell/d) - 1 \leq n \leq \ell/d$. We denote an element of $\text{Clock}_d(\ell)$ by $\phi = \{t_1, \dots, t_n\} \subseteq \tilde{\ell}$; it's the set of “ticks” of the clock, spaced d -apart. Given ϕ and $\text{Ph}_a: \ell' \rightarrow \ell$, the restriction is given by taking those “ticks” that are in the smaller interval:

$$\text{Clock}_d(\text{Ph}_a)(\phi) = \phi|_{[a, a+\ell']} := \phi \cap \tilde{\ell}'.$$

Definition 2.8 (Machine). Let $A, B \in \widetilde{\text{Int}}$. An (A, B) *machine* is a span in $\widetilde{\text{Int}}$

$$\begin{array}{ccc} & C & \\ f^{\text{in}} \swarrow & & \searrow f^{\text{out}} \\ A & & B \end{array}$$

Equivalently, it is a sheaf C together with a sheaf map $f: C \rightarrow A \times B$.² We refer to A as the *input sheaf*, to B as the *output sheaf*, and to C as the *state sheaf*.

²Technically, we identify spans (C, f) and (C', f') if there is an isomorphism $i: C \rightarrow C'$ with $i \circ f' = f$.

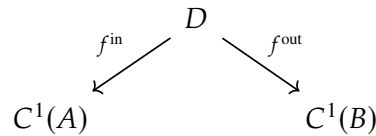
Remark 2.9. Definition 2.8 is symmetric, whereas machines are generally considered causal: inputs affect later outputs. This is captured formally by the notion of being *total*, *deterministic*, and *inertial*, defined in [18]. Totalness (resp. determinism) means that given $c \in C(\ell)$ and $a' \in A(\ell')$ with $f^{\text{in}}(c)|_{[0,\ell]} = a$, there is at least one (resp. at most one), $c' \in C(\ell)$ such that $c'|_{[0,\ell]} = c$ and $f^{\text{in}}(c') = a'$. In other words, there is a unique way that the internal behavior can accommodate any input coming in. A machine is ε -inertial if its internal state on an interval $[a, b]$ determines its output on $[a, b + \varepsilon]$.

In this paper, all of our machines will be total and deterministic; however we will not mention this explicitly. Each can also be made ε -inertial by composing it (Definition 2.12) with an ε -delay (Example 2.11). A result of [18] says that if all the machines in a network are deterministic, total, and inertial, then their composite is as well (Definition 2.12).

Example 2.10 (Continuous dynamical system). For Euclidean spaces A, B , an (A, B) -continuous dynamical system (CDS) consists of

- $S = \mathbb{R}^n$, called the *state space*.
- The *dynamics* $\dot{s} = f^{\text{dyn}}(s, a)$, for any $a \in A$, $s \in S$, and smooth function f^{dyn} .
- The *readout* $b = f^{\text{rdt}}(s)$, with $b \in B$ and smooth function f^{rdt} .

We can write this as a machine



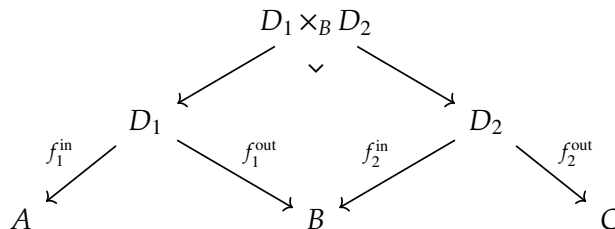
where the apex is given by

$$D(\ell) = \{(a, s, b) \in C^1(A) \times S \times C^1(B) \mid \dot{s} = f^{\text{dyn}}(a, s) \text{ and } b = f^{\text{rdt}}(s)\},$$

and f^{in} and f^{out} are the projections $(a, s, b) \mapsto a$ and $(a, s, b) \mapsto b$. Note that $C^1(A), C^1(B)$ represent the continuously differentiable functions on A and B , respectively.

Example 2.11 (ε -delay). Given any sheaf A and positive real $\varepsilon > 0$, we define the A -type ε -delay machine Del_ε^A to be the span $A \xleftarrow{f^{\text{in}}} A_\varepsilon \xrightarrow{f^{\text{out}}} A$, where $A_\varepsilon(\ell) := A(\ell + \varepsilon)$ is the sheaf of ε -extended behaviors, $f^{\text{in}}(a) = a|_{[0,\ell]}$, and $f^{\text{out}}(a) = a|_{[\varepsilon,\ell+\varepsilon]}$.

Definition 2.12 (Composition of machines). Given two machines $M_1 = (D_1, f_1^{\text{in}}, f_1^{\text{out}})$ and $M_2 = (D_2, f_2^{\text{in}}, f_2^{\text{out}})$ of types (A, B) and (B, C) respectively, their *composite* is the machine $M = (D_1 \times_B D_2, f^{\text{in}}, f^{\text{out}})$ of type (A, C) , namely the span given by pullback:



Once we demand all our machines to be inertial, they do not form a category because there is no identity machine. However they do form an algebra on an operad of wiring diagrams; see [17] for details.

3 Event-based Systems

Definition 3.1 (Event stream). Let A be a set and $\ell \geq 0$. We define a length- ℓ *event stream* of type A to be an element of the set

$$\text{Ev}_A(\ell) := \{(S, a) \mid S \subseteq \tilde{\ell}, S \text{ finite}, a: S \rightarrow A\}.$$

For an event stream (S, a) we refer to elements of $S = \{s_1, \dots, s_n\} \subseteq \tilde{\ell}$ as *time-stamps*, we refer to a as the *event map*, and for each time-stamp s_i we refer to $a(s_i) \in A$ as its *value*.

If the set of time-stamps is empty, there is a unique event map, and we refer to $(\emptyset, !)$ as an *empty event stream*.

Example 3.2. Consider a Swiss traffic light and its set of color transitions

$$A = \{\text{redToOrange}, \text{orangeToGreen}, \text{greenToOrange}, \text{orangeToRed}\}.$$

We will give an example of an element $(S, a) \in \text{Ev}_A(60)$, i.e. a possible event stream of length 60. Its set of time-stamps is $S = \{20, 25, 45, 50\}$, and the event map is given by

$$a: S \rightarrow A$$

$$s \mapsto \begin{cases} \text{redToOrange}, & \text{if } s = 20, \\ \text{orangeToGreen}, & \text{if } s = 25, \\ \text{greenToOrange}, & \text{if } s = 45, \\ \text{orangeToRed}, & \text{if } s = 50. \end{cases}$$

Definition 3.3 (Restriction map on event streams). For any event $e = (S \subseteq \tilde{\ell}, a: S \rightarrow A) \in \text{Ev}_A(\ell)$ and any $0 \leq t \leq t' \leq \ell$, let $S_{t,t'} := \{1 \leq i \leq n \mid t \leq s_i \leq t'\} \subseteq S$, and let $a_{t,t'}: S_{t,t'} \rightarrow A$ be the composite. Then we define the *restriction* of e along $[t, t'] \subseteq \tilde{\ell}$ to be $e|_{[t,t']} := (S_{t,t'}, a_{t,t'})$.

Proposition 3.4. Ev is functorial: Given a function $f: A \rightarrow B$ there is an induced morphism $\text{Ev}_f: \text{Ev}_A \rightarrow \text{Ev}_B$ in $\widetilde{\text{Int}}$, and this assignment preserves identities and composition.

Proposition 3.5. For any set A , the presheaf Ev_A is in fact a sheaf.

Remark 3.6. There is a monoidal structure (\emptyset, \odot) on Set : Its unit is \emptyset and the monoidal product of A and B is $A \odot B := A + B + A \times B$.

Proposition 3.7. $\text{Ev}: (\text{Set}, \odot, \emptyset) \rightarrow (\widetilde{\text{Int}}, \times, 1)$ is a strong monoidal functor: $1 \cong \text{Ev}_\emptyset$ and for any sets A, B , we have

$$\text{Ev}_A \times \text{Ev}_B \cong \text{Ev}_{A \odot B}.$$

Definition 3.8 (Event-based system). Let A, B be sets. An *event-based system* $P = (C, f^{\text{in}}, f^{\text{out}})$ of type (A, B) is a machine

$$\begin{array}{ccc} & C & \\ f^{\text{in}} \swarrow & & \searrow f^{\text{out}} \\ \text{Ev}_A & & \text{Ev}_B \end{array} \quad (1)$$

between two event streams. For any input event stream $e \in \text{Ev}_A(\ell)$, the preimage $(f^{\text{in}})^{-1}(e)$ is the set of all internal behaviors consistent with e .

Remark 3.9. An event-based system P of type (A, B) is graphically represented as

$$\text{Ev}_A \text{---} \boxed{P} \text{---} \text{Ev}_B$$

Example 3.10 (Discrete dynamical systems as event-based systems). Let A, B be sets. Then an (A, B) -discrete dynamical system (DDS) consists of:

- A set S , elements of which are called *states*.
- A function $f^{\text{upd}}: A \times S \rightarrow S$, called the *update function*.
- A function $f^{\text{rdt}}: S \rightarrow B$, called the *readout function*.

We can transform any (A, B) dynamical system into an (A, B) -event-based system as follows. Define D to be the sheaf with the following sections:

$$D(\ell) := \{T \subseteq \tilde{\ell}, (a, s): T \rightarrow A \times S \mid T \text{ finite and } s_{i+1} = f^{\text{upd}}(a_i, s_i) \text{ for all } 1 \leq i \leq n-1\}.$$

The restriction map is the same as for the underlying event stream (Definition 3.3). We define the span (1) as follows. Given $(T, a, s) \in D(\ell)$, we have $f^{\text{in}}(T, a, s) = (T, a)$ and $f^{\text{out}}(T, a, s) = (T, (s \circ f^{\text{rdt}}))$.

Given a function $f: S \rightarrow A$ and a subset $A' \subseteq A$, we can take the preimage $f^{-1}(A') \subseteq S$ and get a function $f|_{A'}: f^{-1}(A') \rightarrow A'$. This can be used to define a filter for event-based systems.

Definition 3.11 (Filter for event-based systems). Let A and $A' \subseteq A$ be sets. An (A, A') -filter is an (A, A') -event-based system with $D = A$, $f^{\text{in}} = \text{id}$, and $f^{\text{out}}(\ell): \text{Ev}_A \rightarrow \text{Ev}_{A'}$ defined on (S, a) , where $S \subseteq \tilde{\ell}$ and $a: S \rightarrow A$, as follows:

$$f^{\text{out}}(\ell)(S, a) := (a^{-1}(A'), a|_{A'}).$$

In other words it consists of the subset (the preimage of $a^{-1}(A') \subseteq S$) of those time-stamps whose associated values are in A' , together with the original event map on that subset.

Definition 3.12 (Continuous stream). Let A be a topological space. We define a *continuous stream* of type A to be

$$\text{Cnt}_A(\ell) := \{a \mid a: \tilde{\ell} \rightarrow A \text{ continuous}\}.$$

Definition 3.13 (Lipschitz continuous function). Given two metric spaces (A, d_A) and (B, d_B) , a function $f: A \rightarrow B$ is called *Lipschitz continuous* if there exists a $K \in \mathbb{R}$, $K \geq 0$, such that for all $a_1, a_2 \in A$:

$$d_B(f(a_1), f(a_2)) \leq K d_A(a_1, a_2).$$

Remark 3.14 (Lipschitz continuous stream). In Definition 3.12, if A is a metric space, then we can consider the subsheaf consisting of only those streams $a: \tilde{\ell} \rightarrow A$ that are Lipschitz continuous, i.e.

$$\text{LCnt}_A(\ell) = \{a: \tilde{\ell} \rightarrow A \mid a \text{ Lipschitz continuous}\} \subseteq \text{Cnt}_A(\ell).$$

Example 3.15. For any S , there is a *codiscrete* topological space \hat{S} with points S and only two open sets: \emptyset and S . For any topological space X , the functions from the underlying set of X to S are the same as the continuous maps $X \rightarrow \hat{S}$.³ Thus, we have

$$\begin{aligned} \text{Cnt}_{\hat{S}}(\ell) &= \{a \mid a: \tilde{\ell} \rightarrow \hat{S}, a \text{ continuous}\} \\ &= \{a \mid a: \tilde{\ell} \rightarrow S\}. \end{aligned}$$

We sometimes denote $\text{Cnt}_{\hat{S}}$ simply by Cnt_S .

³Technically, one can say that the underlying set functor is left adjoint to the codiscrete functor.

Definition 3.16 (Sampler). Let A be a topological space and choose $d \in \mathbb{R}_{\geq 0}$, called the *sampling time*. A *period- d A -sampler* is a span

$$\begin{array}{ccc} & \text{Clock}_d \times \text{Cnt}_A & \\ f^{\text{cnt}} \swarrow & & \searrow f^{\text{evt}} \\ \text{Cnt}_A & & \text{Ev}_A \end{array}$$

where $f^{\text{cnt}}, f^{\text{evt}}$ are morphisms:

$$\begin{aligned} f^{\text{cnt}}: \text{Clock}_d \times \text{Cnt}_A &\rightarrow \text{Cnt}_A \\ (\phi, a) &\mapsto a, \\ f^{\text{evt}}: \text{Clock}_d \times \text{Cnt}_A &\rightarrow \text{Ev}_A \\ (\phi, a) &\mapsto \phi \circledast a. \end{aligned}$$

The second formula is the composite $\phi \subseteq \tilde{\ell} \xrightarrow{a} A$, which takes the value of a at d -spaced intervals.

Definition 3.17 (L -level-crossing sampler). Let (A, dist) be a metric space and consider a Lipschitz input stream $\text{LCnt}_A(\ell)$. Consider the *level* $L \in \mathbb{R}$. A *L -level-crossing sampler* of type A is a machine

$$\begin{array}{ccc} & P & \\ f^{\text{cnt}} \swarrow & & \searrow f^{\text{evt}} \\ \text{LCnt}_A & & \text{Ev}_A \end{array}$$

with $P(\ell) := \{(c, a_0) \mid c \in \text{LCnt}_A(\ell), a_0 \in A\}$. Given $p = (c, a_0) \in P(\ell)$, we have $f^{\text{cnt}}(c, a_0) = c$. Furthermore, either $\text{dist}(c(t), a_0) < L$ for all $t \in \tilde{\ell}$ or there exists $t \in \tilde{\ell}$ with $\text{dist}(c(t_1), a_0) \geq L$. In the first case, take $f^{\text{evt}}(c) = (\emptyset, !)$ to be the empty event stream. In the second case, define

$$t_1 = \inf\{t \in \tilde{\ell} \mid \text{dist}(c(t_1), a_0) \geq L\}$$

and let $a_1 = c(t_1)$. Recursively, define $t_{i+1} \in \tilde{\ell}$ to be the least time such that $\text{dist}(c(t_{i+1}), a_i) \geq L$ (if there is one). There will be a finite number of these because c is Lipschitz. We denote the last of such times by t_n . Then, we define

$$\begin{aligned} f^{\text{evt}}: P(\ell) &\rightarrow \text{Ev}_A \\ (c, a_0) &\mapsto \{t_1, \dots, t_n, c(t_1), \dots, c(t_n)\}. \end{aligned}$$

Definition 3.18 (Reconstructor). Let A be a set. A *reconstructor* of type A is a span

$$\begin{array}{ccc} & C & \\ f^{\text{evt}} \swarrow & & \searrow f^{\text{cnt}} \\ \text{Ev}_A & & \text{Cnt}_A \end{array}$$

with $C(\ell) = \{(S, a_0, a) \mid S \subseteq \tilde{\ell} \text{ finite}, a_0 \in A, a: S \rightarrow A\}$, where $f^{\text{evt}}: C \rightarrow \text{Ev}_A$ is given by $f^{\text{evt}}(S, a_0, a) := (S, a)$, and where $a' := f^{\text{cnt}}(a_0, s_1, \dots, s_n, a_1, \dots, a_n): \tilde{\ell} \rightarrow A$ is given by

$$a'(t) := \begin{cases} a_0, & 0 \leq t < s_1 \\ a(s_i), & s_i \leq t < s_{i+1}, \quad i \in \{2, \dots, n-1\} \\ a(s_n), & s_n \leq t \leq \ell. \end{cases}$$

Remark 3.19. The reconstructed stream is not continuous, it is only piecewise continuous. Luckily, it is continuous with respect to the codiscrete topology. We denote \hat{A} simply by A .

Remark 3.20. The reconstructor is known in signal theory as the zero-order-hold (ZOH), and represents the practical signal reconstruction performed by a conventional digital-to-analog converter (DAC).

Definition 3.21 (Composition of event-based systems). Given two event-based systems $P_1 = (D_1, f_1^{\text{in}}, f_1^{\text{out}})$ and $P_2 = (D_2, f_2^{\text{in}}, f_2^{\text{out}})$ of types (A, B) and (B, C) , they compose as machines do (Definition 2.12). Their *composite* is the (A, C) -event-based system $P = (D_1 \times_B D_2, f^{\text{in}}, f^{\text{out}})$.

Remark 3.22. We refer to the composition of event-based systems as putting them in *series*, and represent it graphically as

$$\text{Ev}_A \text{---} \boxed{P_1} \xrightarrow{\text{Ev}_B} \boxed{P_2} \text{---} \text{Ev}_C \equiv \text{Ev}_A \text{---} \boxed{P} \text{---} \text{Ev}_C$$

Definition 3.23 (Tensor Product of event-based systems). Given two event-based systems $P_1 = (E_1, f_1^{\text{in}}, f_1^{\text{out}})$ and $P_2 = (E_2, f_2^{\text{in}}, f_2^{\text{out}})$ of types (A, B) and (C, D) , their tensor product is an event-based system $P = (E_1 \times E_2, f^{\text{in}}, f^{\text{out}})$ of type $(A \odot C, B \odot D)$, i.e. a span

$$\begin{array}{ccc} & E_1 \times E_2 & \\ f_1^{\text{in}} \times f_2^{\text{in}} \swarrow & & \searrow f_1^{\text{out}} \times f_2^{\text{out}} \\ \text{Ev}_A \times \text{Ev}_C & & \text{Ev}_B \times \text{Ev}_D \end{array}$$

This is an event-based system because $\text{Ev}_A \times \text{Ev}_C \cong \text{Ev}_{A \odot C}$ and $\text{Ev}_B \times \text{Ev}_D \cong \text{Ev}_{B \odot D}$.

Remark 3.24. We refer to the tensor product of event-based systems as putting them in *parallel*, and represent it graphically as

$$\begin{array}{ccc} \text{Ev}_A \text{---} \boxed{P_1} \text{---} \text{Ev}_B & & \\ \text{Ev}_C \text{---} \boxed{P_2} \text{---} \text{Ev}_D & \equiv & \text{Ev}_{A \odot C} \text{---} \boxed{P} \text{---} \text{Ev}_{B \odot D} \end{array}$$

Definition 3.25 (Trace of event-based system). Given an event-based system $P = (D, f^{\text{in}}, f^{\text{out}})$ of type $(A \times C, B \times C)$, its trace is an event-based system $P' = (E_{\text{tr}}, f_{\text{tr}}^{\text{in}}, f_{\text{tr}}^{\text{out}})$ of type (A, B) given by

$$E_{\text{tr}}(\ell) = \{d \in D(\ell) \mid \pi_2(f^{\text{in}}(d)) = \pi_2(f^{\text{out}}(d))\}$$

with $f_{\text{tr}}^{\text{in}}(d) = \pi_1(f^{\text{in}}(d)) \in \text{Ev}_A$ and $f_{\text{tr}}^{\text{out}}(d) = \pi_1(f^{\text{out}}(d)) \in \text{Ev}_B$.

Remark 3.26. We depict an event-based system with trace as having *feedback* or a *loop*, and represent it graphically as

$$\begin{array}{ccc} \text{Ev}_A \text{---} \boxed{P} \text{---} \text{Ev}_B & & \\ \text{Ev}_C \text{---} \overleftarrow{\quad} \text{Ev}_C & & \\ & \equiv & \text{Ev}_A \text{---} \boxed{P'} \text{---} \text{Ev}_B \end{array}$$

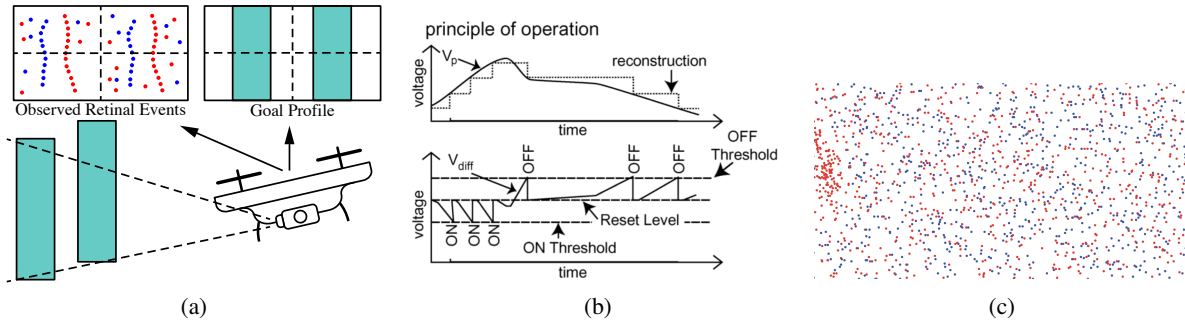


Figure 1: (a) Illustration of the neuromorphic heading regulation problem. (b) Working principle of the Dynamic Vision Sensor [13]. (c) Events over a given time-window [3].

4 Example: Neuromorphic Optomotor Heading Regulation

In the following, we want to show that complex CPSs can be modeled using the framework presented in Section 3. To do so, we consider the neuromorphic optomotor heading regulation problem studied in [3]. Specifically, we consider the case of a body moving in the plane and changing its orientation, expressed as an element of $SO(2)$, based on the scene observed by an event camera mounted on it. The result is reported in Figure 2, which shows that this complex system can be understood as the composition (Definition 2.12) of machines.

4.1 Background on Event Cameras

Event cameras are *asynchronous* sensors which have introduced a completely new acquisition technique for visual information [13]. This new type of sensors samples light depending on the scene dynamics and therefore differs from standard cameras. In particular, event cameras show notable advantages such as high temporal resolution, low latency (in the order of microseconds), low power consumption, and high dynamic range, all of which naturally encourage their employment in robotic applications. An exhaustive review of the existing applications of event cameras has been reported in [6].

Event Generation Model

In the following, we review the event generation model presented in [3]. An event camera [13] is composed of a set \mathcal{S} , elements of which are called pixels, reacting (independently) to changes in light *brightness*. There is a function $\text{dir}: \mathcal{S} \rightarrow \mathbb{S}^1$, representing the *direction* of each pixel in the event camera's field of view. The environment reflectance is a function $m: \mathbb{S}^1 \rightarrow \mathbb{R}_{\geq 0}$, such that $m(\text{dir}(s))$ represents the intensity of light from direction $\text{dir}(s) \in \mathbb{S}^1$ at any given moment of time. The light field is a map

$$I: \mathbb{R}_{\geq 0} \times \mathcal{S} \rightarrow \mathbb{R}_{\geq 0} \quad (2)$$

$$(t, s) \mapsto I_t^s,$$

where I_t^s represents the intensity of light reaching the event camera at time t in direction $\text{dir}(s) \in \mathbb{S}^1$. Brightness is expressed as $L := \log(I_t^s)$. An event $e = (s, t, p)$ is generated at pixel location s at time t if the change

$$\Delta L(s, t) := L(s, t) - L(s, t')$$

in brightness at that pixel since the last event t' was fired reaches a threshold $\pm C$ (Figure 1b), i.e. $|\Delta L(s, t)| = pC$, where $p \in \{-1, 1\}$ represents the event's polarity [6] (blue and red in Figure 1).

Remark 4.1. Note that the contrast sensitivity C can be tuned depending on the applications through the pixel bias currents, as explained in [14].

4.2 Robotic System Description

In this section, we describe the robotic system presented in [3]. The system is composed of a body, which is able to move on the plane and to change its orientation (heading), expressed as an element of $SO(2)$. An event camera is mounted on the body, and allows for it to perceive the environment. Furthermore, heading regulation happens in image space via feedback from the event camera through a decision process (regulator). A graphical representation of the robotic system is shown in Figure 2.

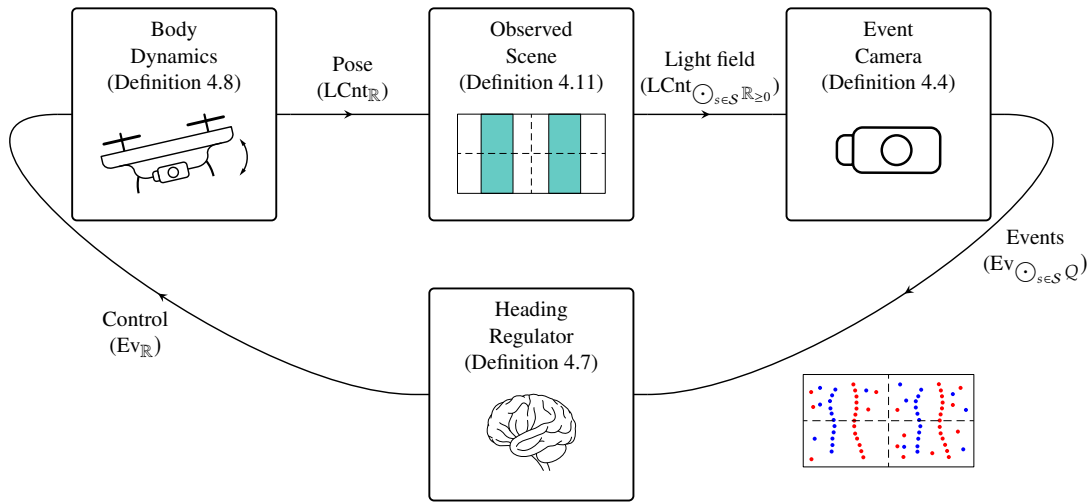


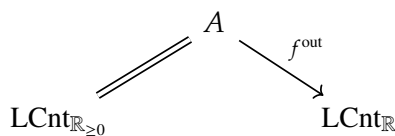
Figure 2: Graphical representation of the neuromorphic heading regulation problem as a composition of machines, together with input-output stream types.

In the following, each one of the aforementioned components will be described as a machine, showing the unifying properties of our framework. The components' interactions will be represented through composition and product of machines. The diagram reported in Figure 2 contextualizes the following definitions.

4.2.1 Event Camera

To represent an event camera in our framework, we first consider a single pixel as a machine. In Definition 4.4 we will define an event camera to be the product (Definition 3.23) of n independent event camera pixels.

Definition 4.2 (Event camera pixel). First, define a $(LCnt_{\mathbb{R}_{\geq 0}}, LCnt_{\mathbb{R}})$ machine P_1



with $A(\ell) := \{a: \tilde{\ell} \rightarrow \mathbb{R} \mid a \text{ Lipschitz continuous}\}$. For $a \in A(\ell)$, define $f^{\text{in}}(a) = a$ (i.e. the identity, from now on as in the given span) and $f^{\text{out}}(a) = \log(a)$. Then, for any $C \in \mathbb{R}_{\geq 0}$, called the *contrast sensitivity*, let P_2 be a C -level-crossing sampler (Definition 3.17) of type \mathbb{R} , with input $\text{LCnt}_{\mathbb{R}}$ and output $\text{Ev}_{\mathbb{R}}$. Finally, let P_3 be the (\mathbb{R}, Q) -event-based system corresponding to the DDS (Example 3.10) of input-output type (\mathbb{R}, Q) , $Q = \{-1, 1\}$, with state set $S = \mathbb{R}_{\geq 0} \times Q$, consisting of pairs (r, q) , readout $f^{\text{rdt}}: S \rightarrow Q$ defined as $f^{\text{rdt}}(r, q) = q$, and update function defined as

$$f^{\text{upd}}: \mathbb{R} \times S \rightarrow S$$

$$(r', r, q) \mapsto \begin{cases} (r', 1), & r' - r \geq C \\ (r', -1), & r' - r \leq -C. \end{cases}$$

An *event-camera pixel* for an event camera with contrast sensitivity C is the composite (Definition 2.12) machine $P_C = P_1 \circ P_2 \circ P_3$ with input of type $\text{LCnt}_{\mathbb{R}_{\geq 0}}$ and output of type Ev_Q .

Remark 4.3 (Interpretation). Recalling the working principle of an event camera pixel, the continuous input of P_C represents the light intensities measured by a pixel at a specific time (Equation 2). The machine P_1 computes the log of the intensities, and the C -level-crossing sampler P_2 determines when the changes in intensities are sufficient to generate events. The DDS P_3 determines the polarity of the events, by storing it in the state set together with the brightness of the previously fired event.

We are now ready to define an event camera as a machine, arising from the product of the event camera pixels (machines) composing it.

Definition 4.4 (Event camera). Let \mathcal{S} be a set, elements of which we think of as pixels. For any contrast sensitivity $C \in \mathbb{R}_{\geq 0}$, define the *event camera with \mathcal{S} pixels and contrast sensitivity C* to be the product (Definition 3.23) $\prod_{s \in \mathcal{S}} P_C = (P_C)^{\mathcal{S}}$, where P_C is as in Definition 4.2. The input stream of P_C is of type $\text{LCnt}_{\bigcirc_{s \in \mathcal{S}} \mathbb{R}_{\geq 0}}$ and the output stream is of type $\text{Ev}_{\bigcirc_{s \in \mathcal{S}} Q}$.

Remark 4.5. Recall the monoidal structure presented in Remark 3.6. The strong monoidality of Ev (Proposition 3.7) means that each event of the event camera consists of an event at one or more of its pixels.

4.2.2 Heading Regulator

Given the body with heading θ_t , one wants to steer it toward some “goal” heading θ_g , i.e. one wants to reach $\theta_{t+\Delta t} \in [\theta_g - \delta, \theta_g + \delta]$, for some probability $1 - \varepsilon(\delta)$. This is achieved by a *heading regulator*, which only considers the events measured by the event camera, and takes decisions in real time. To do so, one needs a function $f: \mathcal{S} \rightarrow \mathbb{R}$, called the *estimator*, which for each event observed on pixel $s_i \in \mathcal{S}$ at time t_j gives an estimate of the current heading of the body $\theta_{t_j} = f(s_i)$. As shown in [3], it is sufficient to find an f such that

$$\int_{\mathcal{S}} f(s) p_e(s, t) ds = \theta_t, \quad (3)$$

in a neighborhood of θ_g , where $p_e(s, t)$ represents the probability of event e being generated at pixel s at time t .

Remark 4.6. As shown in [3], a possible choice for f is

$$f: \mathcal{S} \rightarrow \mathbb{R}$$

$$s \mapsto f(s) := \int_{s-\delta}^{s+\delta} p_{\theta}(s-v) |\nabla m^v|(s-v) dv,$$

where each pixel v contributes with $(s - v)$, weighted by the probability of generating an event (∇m^v) , given the probability $p_\theta(s - v)$ of θ being $s - v$.

The events used to regulate the heading are described through statistics $S_t \in \mathbb{R}$, which are computed asynchronously, when an event is observed. Given a function f , one can define the heading regulator as an event-based system.

Note that $\bigodot_{s \in \mathcal{S}} Q \cong \{(S, q) \mid S \subseteq \mathcal{S} \text{ non-empty}, q: S \rightarrow Q\}$. In fact, the heading regulator will not use the polarities q , but only the firing sets S . We will follow the style of Example 3.10, but what follows does not arise from a DDS because we explicitly use the time-stamps.

Definition 4.7 (Heading regulator). Given an event-camera (Definition 4.4), we define an $(\bigodot_{s \in \mathcal{S}} Q, \mathbb{R})$ event-based system $H = (D, f^{\text{in}}, f^{\text{out}})$ (the *heading regulator*) as follows. Let $X = \mathbb{R}_{\geq 0} \times \mathbb{R}$ and define

$$D(\ell) := \{t_1, \dots, t_n, (S_1, q_1), \dots, (S_n, q_n), x_1, \dots, x_n \mid x_i = f^{\text{upd}}((S_i, q_i), t_i, x_{i-1})\},$$

where $\{t_1, \dots, t_n\} \subseteq \tilde{\ell}$, $x_i \in X$, $(S_i, q_i) \in \bigodot_{s \in \mathcal{S}} Q$, and

$$f^{\text{upd}}: \bigodot_{s \in \mathcal{S}} Q \times \mathbb{R}_{\geq 0} \times X \rightarrow X$$

$$((S, q), t, x) \mapsto \left(t, \sum_{s_i \in S} e^{-a(t - \pi_1(x))} \pi_2(x) - \frac{\kappa}{a} f(\text{dir}(s_i)) \right),$$

with f satisfying Equation 3, and $a > 0, \kappa > 0$ tunable parameters. Then define $f^{\text{rdt}}(x) = \pi_2(x)$ and

$$f^{\text{in}}: D(\ell) \rightarrow \text{Ev} \bigodot_{s \in \mathcal{S}} Q$$

$$d \mapsto f^{\text{in}}(d) := \{t_1, \dots, t_n, (S_1, q_1), \dots, (S_n, q_n)\}, \quad (S_i, q_i) \in \bigodot_{s \in \mathcal{S}} Q,$$

and

$$f^{\text{out}}: D(\ell) \rightarrow \text{Ev} \mathbb{R}$$

$$d \mapsto f^{\text{out}}(d) := \{t_1, \dots, t_n, f^{\text{rdt}}(x_1), \dots, f^{\text{rdt}}(x_n)\}, \quad x_i \in X.$$

4.2.3 Body Dynamics

For small variations, the body orientation $\theta_t \in \mathbb{R}$ and its dynamics are expressed through the law

$$d\theta_t = \text{sat}_b(u) dt,$$

where $u \in \mathbb{R}$ represents the input received from the controller and

$$\text{sat}_b: \mathbb{R} \rightarrow \mathbb{R}_{[-b, b]}$$

$$r \mapsto \text{sat}_b(r) := \begin{cases} r, & \text{if } r \in [-b, b], \\ -b, & \text{if } r < -b, \\ b, & \text{if } r > b \end{cases}$$

represents the saturation of the actuators, which receiving a control $r \in \mathbb{R}$, are only able to commute it to an actuation $\text{sat}_b(r) \in [-b, b]$. The body dynamics can be written as a machine.

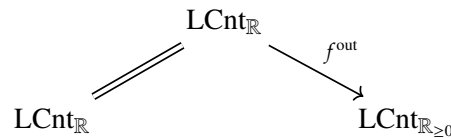
Definition 4.8 (Body dynamics). Consider a reconstructor (Definition 3.18) P_1 of type \mathbb{R} with input $\text{Ev}_{\mathbb{R}}$ and output $\text{Cnt}_{\mathbb{R}}$. Furthermore, consider a continuous dynamical system P_2 (Example 2.10) of input-output type $(C^1(\mathbb{R}), C^1(\mathbb{R}))$, with state space $X = \mathbb{R}$ representing the pose of the robot. Then, define the dynamics as $\dot{s} = \text{sat}_b(u)$ and the readout as the identity, i.e. $f^{\text{rdt}}(s) = s$. The *body dynamics* P are given by the composition (Definition 2.12) of machines $P_1 \circ P_2$. It has input stream of type $\text{Ev}_{\mathbb{R}}$ and output stream of type $\text{LCnt}_{\mathbb{R}}$.

Remark 4.9 (Interpretation). The heading regulator produces an event stream of type $\text{Ev}_{\mathbb{R}}$. However, the body dynamics are expressed in continuous time and therefore one needs a reconstructor P_1 . Then, P_2 just represents the continuous dynamics.

4.2.4 Observed Scene

Considering the variations in the dynamics, resulting in the current pose θ_t , one can write the variations of the light intensities for each pixel $s \in \mathcal{S}$ with direction $\text{dir}(s) \in \mathbb{S}^1$ as $I_t^s = m(\theta_t + \text{dir}(s))$, where m represents the environment reflectance introduced in Section 4.1.

Definition 4.10 (Scene observed by an event-camera pixel). Given a pixel $s \in \mathcal{S}$ of an event-camera, let $\text{dir}(s)$ be its fixed direction. The *scene observed by an event-camera pixel* is an $(\text{LCnt}_{\mathbb{R}}, \text{LCnt}_{\mathbb{R}_{\geq 0}})$ machine



For $\theta \in \text{LCnt}_{\mathbb{R}}(\ell)$, define

$$\begin{aligned} f^{\text{out}} : \text{LCnt}_{\mathbb{R}}(\ell) &\rightarrow \text{LCnt}_{\mathbb{R}_{\geq 0}} \\ \theta &\mapsto (\theta + \text{dir}(s)) \circledast m. \end{aligned}$$

Definition 4.11 (Scene observed by an event-camera). Consider an event camera as in Definition 4.4. The scene observed by each pixel of the camera is a machine P of type $(\text{LCnt}_{\mathbb{R}}, \text{LCnt}_{\mathbb{R}_{\geq 0}})$ (Definition 4.10). The *scene observed by an event camera* is the machine given by the product (Definition 3.23) of machines $\prod_{s \in \mathcal{S}} P = P^{\mathcal{S}}$.

Note that the output of the machine presented in Definition 4.11 is of the same type of the input of the machine presented in Definition 4.4. This allows us to close the loop reported in Figure 2 using the trace (Definition 3.25).

Consider an event camera C (Definition 4.4), a heading regulator H (Definition 4.7), the body dynamics B (Definition 4.8), and the scene observed by the event camera O (Definition 4.11). In order to take the trace (and have the result be deterministic and total) we compose with a \mathbb{R} -type ε -delay machine (Example 2.11) $\text{Del}_{\varepsilon}^{\mathbb{R}}$. Then the trace

$$\text{Tr}^{\mathbb{R}}(B \circledast O \circledast C \circledast H \circledast \text{Del}_{\varepsilon}^{\mathbb{R}}),$$

of the composite machine (Definition 2.12), is the desired closed-loop behavior of the robotic system.

5 Conclusion and future Work

In this paper, we presented a framework characterized by high descriptive power and formality, and we showed how event-based systems can be modeled using it. However, the framework does not subsume the literature presented in Section 1 yet. In particular, we look forward to exploring three extensions. First, we would like to explicitly introduce a notion of uncertainty, which would allow for a more accurate description of particular systems. Second, we would like to explore the implications of super-dense time, mentioned in [12]. Third, we aim at developing tools for the synthesis of event-based systems, using the internal language of the topos of behaviour types [17], and the mathematical theory of co-design [4].

References

- [1] Aaron David Ames (2006): *A categorical theory of hybrid systems*. Ph.D. thesis.
- [2] Michael S Branicky, Vivek S Borkar & Sanjoy K Mitter (1998): *A unified framework for hybrid control: Model and optimal control theory*. *IEEE transactions on automatic control* 43(1), pp. 31–45, doi:10.1109/9.654885.
- [3] Andrea Censi (2015): *Efficient neuromorphic optomotor heading regulation*. In: *2015 American Control Conference (ACC)*, IEEE, pp. 3854–3861, doi:10.1109/ACC.2015.7171931.
- [4] Andrea Censi (2015): *A mathematical theory of co-design*. *arXiv preprint arXiv:1512.08055*.
- [5] Dimos V Dimarogonas, Emilio Frazzoli & Karl H Johansson (2011): *Distributed event-triggered control for multi-agent systems*. *IEEE Transactions on Automatic Control* 57(5), pp. 1291–1297, doi:10.1109/TAC.2011.2174666.
- [6] Guillermo Gallego, Tobi Delbruck, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Joerg Conradt, Kostas Daniilidis et al. (2019): *Event-based vision: A survey*. *arXiv preprint arXiv:1904.08405*.
- [7] WPMH Heemels, Karl Henrik Johansson & Paulo Tabuada (2012): *An introduction to event-triggered and self-triggered control*. In: *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, IEEE, pp. 3270–3285, doi:10.1109/CDC.2012.6425820.
- [8] Thomas A Henzinger (2000): *The theory of hybrid automata*. In: *Verification of digital and hybrid systems*, Springer, pp. 265–292, doi:10.1007/BFb0032003.
- [9] Edward A Lee (2015): *The past, present and future of cyber-physical systems: A focus on models*. *Sensors* 15(3), pp. 4837–4869, doi:10.3390/s150304837.
- [10] Edward A Lee (2016): *Fundamental limits of cyber-physical systems modeling*. *ACM Transactions on Cyber-Physical Systems* 1(1), pp. 1–26.
- [11] Edward A Lee (2018): *Models of Timed Systems*. In: *International Conference on Formal Modeling and Analysis of Timed Systems*, Springer, pp. 17–33, doi:10.1109/RTAS.2007.5.
- [12] Edward A Lee, Jaijeet Roychowdhury & Sanjit A Seshia (2010): *Fundamental Algorithms for System Modeling, Analysis, and Optimization*.
- [13] Patrick Lichtsteiner, Christoph Posch & Tobi Delbruck (2008): *A 128×128 120 dB 15 μs Latency Asynchronous Temporal Contrast Vision Sensor*. *IEEE journal of solid-state circuits* 43(2), pp. 566–576, doi:10.1109/JSSC.2007.914337.
- [14] Yuji Nozaki & Tobi Delbruck (2017): *Temperature and parasitic photocurrent effects in dynamic vision sensors*. *IEEE Transactions on Electron Devices* 64(8), pp. 3239–3245, doi:10.1109/TED.2017.2717848.
- [15] André Platzer: *Lecture Notes on Foundations of Cyber-Physical Systems*.
- [16] André Platzer (2008): *Differential dynamic logic for hybrid systems*. *Journal of Automated Reasoning* 41(2), pp. 143–189, doi:10.1007/s10817-008-9103-8.

- [17] Patrick Schultz & David I Spivak (2019): *Temporal Type Theory: A topos-theoretic approach to systems and behavior*. 29, Springer, doi:10.1007/978-3-030-00704-1_5.
- [18] Patrick Schultz, David I Spivak & Christina Vasilakopoulou (2020): *Dynamical systems and sheaves*. *Applied Categorical Structures* 28(1), pp. 1–57, doi:10.1016/j.jpaa.2016.10.009.
- [19] Alberto Speranzon, David I Spivak & Srivatsan Varadarajan (2018): *Abstraction, Composition and Contracts: A Sheaf Theoretic Approach*. *arXiv preprint arXiv:1802.03080*.
- [20] Paulo Tabuada, George J Pappas & Pedro Lima (2004): *Compositional abstractions of hybrid control systems*. *Discrete event dynamic systems* 14(2), pp. 203–238, doi:10.1023/B:DISC.0000018571.14789.24.
- [21] Norbert Wiener (1948): *Cybernetics or Control and Communication in the Animal and the Machine*. MIT press.
- [22] Gioele Zardini, David I. Spivak, Andrea Censi & Emilio Frazzoli (2020): *A Compositional Sheaf-Theoretic Framework for Event-Based Systems (Extended Version)*. *arXiv preprint arXiv:2005.04715*.