Year: 2015

A framework of relational networks to build systems with sensors able to perform the joint approximate inference of quantities

Martel, Julien ; Cook, Matthew

Abstract: Probabilistic approaches such as Bayesian inference have been extensively used to design systems able to operate in environments under uncertainty. Implementing these approaches on real-world systems constrained in their latencies or in their power-budget is a challenge because of the general computational intensity required by such methods. In this work, we propose a very simple yet efficient framework to perform approximate inference in a network of quantities between which relations are specified a-priori. We present how we can take advantage of computational features of our framework to implement it in dedicated hardware devices such as GPGPUs or Cellular Processor Arrays (CPAs) for which we demonstrate a simple vision system instantiating the principles of our approach.

A Framework of Relational Networks to Build Systems with Sensors able to Perform the Joint Approximate Inference of Quantities

Julien N.P. Martel
Institute of Neuroinformatics
University of Zürich / ETH-Zürich
Zürich, Switzerland
Email: jmartel@ini.ethz.ch

Matthew Cook
Institute of Neuroinformatics
University of Zürich / ETH-Zürich
Zürich, Switzerland
Email: cook@ini.ethz.ch

***Abstract*—Probabilistic approaches such as Bayesian inference have been extensively used to design systems able to operate in environments under uncertainty. Implementing these approaches on real-world systems constrained in their latencies or in their power-budget is a challenge because of the general computational intensity required by such methods. In this work, we propose a very simple yet efficient framework to perform approximate-inference in a network of quantities between which relations are specified a-priori. We present how we can take advantage of computational features of our framework to implement it in dedicated hardware devices such as GPGPUs or Cellular Processor Arrays (CPAs) for which we demonstrate a simple vision system instantiating the principles of our approach.**

I. Introduction

To autonomously behave in their environment we build systems equipped with sensors reporting noisy measurements. Achieving some perception in these system is a difficult but crucial step toward the construction of more complex cognitive architectures capable of taking decisions or performing actions in their environments. The goal of a sensory system is then not only to capture information but to “interpret” it by deriving other useful quantities for further processing. For instance, a visual system embodied in a mobile robot equipped with a conventional camera might have to interpret the captured light-intensity in other quantities related to the motion being imaged, its ego-motion or a depth map in order to perform obstacle avoidance.
Quantities we are interested in these sensory systems can generally be divided into two categories whether a sensor is present and is able to report related measurements or not. In the first case, the quantity is said to be observed, this would be the case of the light-intensity being reported by the camera in our example. In the other case, the quantity is latent or hidden such as the imaged motion –so-called optic-flow– as well as the depth map or the mobile robot’s ego-motion. The process of finding the possible values of a non-observed quantity is called inference. Inference is usually performed assuming interactions exist between observed and latent quantities such that the values for the latent ones conditioned on the observed ones can be derived. When building systems it is natural to express how quantities relate to each other based on our knowledge

This work was funded by SNF grant 143947.

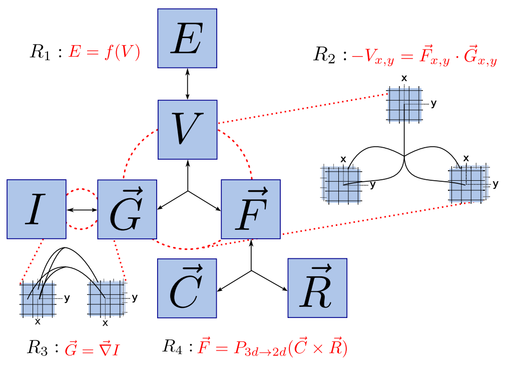
$R_1 : E = f(V)$
E
$R_2 : -V_{x,y} = \vec{F}_{x,y} \cdot \vec{G}_{x,y}$
V
x
y
I
$\vec{G}$
$\vec{F}$
$\vec{C}$
$\vec{R}$
$R_3 : \vec{G} = \vec{\nabla} I$
$R_4 : \vec{F} = P_{3d \rightarrow 2d}(\vec{C} \times \vec{R})$

Fig. 1: An illustration of a relational network modelling an observer rotating in front of a static scene. This network illustrates the quantities and relations detailed in Section IV. Each rectangle is a map of 128×128 pixel quantities corresponding to the resolution of the DVS sensor [1], with the exception of $\vec{R}$, a 3-dimensional vector.

of its design or from the assumptions we are ready to make on the nature of its environment. This led to the creation of a framework based on relational networks.

In this work, we focus on formulating principles describing how we can build systems with sensors using relational networks. Our main goal is to create a framework that maps easily on parallel computing devices.
In Section II we present the principles of our framework, introducing the notion of quantities and relations and a simple message-passing algorithm to perform approximate inference. We discuss in Section III properties of our message passing algorithm as well as its relation to existing approaches and algorithms: we show how our approach can be interpreted from a Bayesian perspective as performing some kind of maximum likelihood inference. We present interesting computational features that our model exhibits such as its parallelism and distributivity. These properties are used to instantiate systems that have to meet real-world constraints such as having a low latency or running on a low power budget by taking advantage of dedicated hardware such as GPUs or CPAs.
Finally we illustrate in Section IV the principles introduced in this work by creating a relational network able to process visual information coming from a neuromorphic sensor reporting asynchronous events for changes in light-intensity.

II. A network of quantities and relations equipped with a message-passing algorithm

A. A network of quantities with relations modelling a system

In this work, we describe a problem in terms of observed and inferred quantities connected by relations. Relations are a-priori defined between the quantities –they are not learnt in the scope of this work– and connect them in a network. We then suggest a message passing algorithm that allows to perform inference over the network by prescribing the values of the quantities in order that they satisfy the relations they are involved in.

1) Quantities: $\{Q_i\}_{i\in\{1,...N\}}, N \in \mathbb{N}$ are random variables whose values $\{q_i\}_i \in \{1,...N\}$ typically lie in $\mathbb{R}^m, m \geq 1$. The systems we aim to build use sensors measuring certain quantities, these are the inputs of our framework. A quantity having its value fed with measurements coming from a sensor is said to be observed while a quantity with no measurement is said to be latent or hidden. The latter can only have its value inferred given the values of the quantities it is in relation with. Any quantity whose value is probed by a third-part process can be considered as an output of the framework.

2) Relations and networks: A relation $\mathcal{R}$ is defined over a non-empty set of quantities $\mathcal{D} = \{Q_i\}_{i\in\{1,...N\}}, N \in \mathbb{N}$, called the domain of the relation.
The relation $\mathcal{R}$ is a tuple $\mathcal{R} = (\mathcal{D}, R)$ where R is a subset of the Cartesian product of the elements of $\mathcal{D}$ (the quantities), $R \subseteq Q_1 \times Q_2 \times ... \times Q_N$.
The value $\{q_i\}_i$ for the quantities $\{Q_i\}_i$ are said to satisfy the relation $\mathcal{R}$ if the tuple $(q_1, ..., q_i, ..., q_N)$ is in R. To introduce a gradation in the "level of satisfaction" of a relation with respect to a quantity that is involved, one could define an error function defined over the relation. As a very simple example, one can imagine a simple relation between three scalar quantities $(A, B, C) \in \mathbb{R}^3$ stating that one of them has to be the sum of the two others say $C = A + B$. The tuple of values $(1.2, 3.4, 4.6)$ is an example of a tuple satisfying this relation.
A relational network is a pair consisting of a set of quantities and a set of relations defined over these quantities $\mathcal{N} = (\{Q_i\}_i, \{\mathcal{R}_r\}_r)$.

We model interactions between the quantities using these relations, they embed our knowledge about the system and its design as well as the assumptions we are ready to make about its environment.

3) Directions of a relation: The direction $\mathcal{R}_{Q_i\leftarrow\{Q_j\}_j}$ of a relation $\mathcal{R}$ is defined as the mapping between the source quantities $\{Q_j\}_j = \mathcal{D} \setminus \{Q_i\}$ and the target quantity $Q_i \in \mathcal{D}$. Note that this mapping is not necessarily a function since there might exist multiple values $\{q_{i,v}\}_v$ for the target quantity Q_i mapping to the values $\{q_j\}_j$ of the source quantities $\{Q_j\}_j$ such that the relation $\mathcal{R}$ is satisfied *i.e*, $\forall v \in \{1..., V\}, V \in \mathbb{N}$ the tuples $(q_1, ...q_{i,v}, ..., q_N) \in R$.
For a target quantity $Q_i \in \mathcal{D}$, we denote the set of possible values $\{q_{i,v}\}_v$ satisfying the relation $\mathcal{R}$ given the values $\{q_j\}_j, j \neq i$ associated to the source quantities $\{Q_j\}_j = \mathcal{D} \setminus \{Q_i\}$ as $\{q_{i,v}\}_v = \mathcal{R}_{Q_i\leftarrow\{Q_j\}_j}(Q_i|\{q_j\}_j)$.

We define a direction to be able to retrieve the possible values –the ones that can satisfy the relation– of a target quantity given the values of the source quantities.
In practice, if multiple values for a target quantity can satisfy the relation, we could keep track of all of them or consider a single one. In the latter case, the direction of a relation could be thought as a function applied on the values of the source quantities which image is the value of the target quantity.

```
Require:
  •A network $\mathcal{N} = (\{Q_i\}_i, \{\mathcal{R}_j\}_j)$
  •A set of hyper-parameters $\{\eta_j\}_j$ controlling the update-rate of each relation
  •A maximum number of iterations (or any other stopping criterion) $T \geq 0$
 1: for all $Q_i \in \{Q_i\}_i$ do
 2:     $Q_i \leftarrow q_i^t \sim$ random()
        {Quantities can be initialized sensibly if knowledge or assumptions about their distributions can be made}
 3: end for
 4: $t \leftarrow 0$
 5: for all input quantities $\in \{Q_{i'}\}_{i'}$ do
 6:     $Q_{i'} \leftarrow q_{i'}^t \sim$ sensor$_{i'}$()
 7: end for
 8: while $t \leq T$ do
 9:     for all $\mathcal{R}_{Q_i\leftarrow\{Q_j\}_j}$ do
10:         $\mu^{t+1}(Q_i) \leftarrow \{q_j^t\}_j$
11:         $q_i^{t+1} \leftarrow (1-\eta_j)\cdot q_i^t + \eta_j \cdot \mathcal{R}_{Q_i\leftarrow\{Q_j\}_j}(Q_i|\mu^{t+1}(Q_i))$
12:     end for
13:     $t \leftarrow t + 1$
14: end while
```

Fig. 2: The algorithm describing the simple message-passing we suggest to equip our framework to perform approximate-inference.

Coming back to the example we introduced in II-A2 where three quantities A, B and C are related such that $C = A + B$, note that this equation for the relation naturally expresses one of the three directions of the relation. The values of the *target quantity* C are given by summing the *values* of A and B. Finding the two other directions is trivial, manipulating the equation one writes $A = C - B$ yielding the possible values when A is considered as a *target quantity* given the sources B and C. Similarly for B: $B = C - A$. In the general case, finding the form of a *direction* expressing the mapping between the *target quantity* and the *source quantities* might be difficult. Given that this mapping is generally not a function, one might actually want to find a single *value* for a *target quantity* and reduce the mapping to a function.

B. An update rule to change the value of a target quantity in a given direction

We suggest an approximate inference procedure for the framework to find the values of latent quantities by iteratively pushing quantities in the network to satisfy the relations they are involved in. First we introduce an update-rule treating an incoming message $\mu(Q_i)$ to update a target quantity Q_i as:

$$q_i^{t+1} = (1-\eta)\cdot q_i^t + \eta \cdot \mathcal{R}_{Q_i\leftarrow\{Q_j\}_j}(Q_i|\mu^{t+1}(Q_i)) \quad (1)$$
$$\text{with } \eta \in [0; 1]$$

The superscript t indicates an iteration of the message-passing algorithm we describe later in Section II-C. A message $\mu(Q_i)$ to update the target quantity Q_i using a particular direction

$\mathcal{R}_{Q_i \leftarrow \{Q_j\}_j}$ contains the values $\{q_j\}_j$ of the source quantities of this direction: $\mu^{t+1}(Q_i) = \{q_j^t\}_j$.

The update-rule has the form of a simple convex blending of parameter η. Intuitively, it smoothly changes the value maintained by the target quantity toward better satisfying the relation it is involved in by incorporating the suggested value from the direction applied to the source quantities.

One might wonder why the update-rule does not directly update the target quantity with the prescribed value coming from the direction given the values of the source quantities (*i.e* $\eta = 1$)? In fact, consider that a quantity might be involved in more than a single relation, therefore directly updating the value of the target quantity with the particular direction of a relation might lead to upset other relations the quantity is involved in.

C. A message passing algorithm for approximate inference

The objective of the inference procedure we describe in this section is to change the value of a quantity to better satisfy a direction using the above update-rule, and do it again for every direction until convergence, *i.e* until the values of every quantity satisfy the relations. If each direction is picked in turn, so that the value of the target quantities are slowly enforced, the network can be expected to converge to a state where all the quantities have their values satisfy their relations. This is only possible given that an input conform to the model's hypothesis is provided; in any other case, the relations might not be satisfiable at all. Because the values of the quantities are jointly pushed to optimize the relations they are involved in using a process that is only local it intuitively explains why the suggested inference is approximate, *i.e* non-optimal.

In more details, when an input feeds the values of one or several observed quantities we want to update all the quantities using each of the directions, in turn, and gently change the values of the associated target quantities by the mean of the update-rule we introduced in II-B. Precisely the message passing-algorithm picks a direction, creates a message emerging from the source quantities containing their values, apply the direction to this message and updates the target quantity using the update-rule. Another direction is then picked, creates another message emerging from the source quantities with the values that have been previously updated if updated, and updates the target quantity, and so forth. Once all the directions have updated their target quantity, an iteration of the message passing algorithm has been done. We iteratively repeat this procedure until convergence. We summarize our simple message passing algorithm in Figure 2.

III. Properties and comparisons of the framework with other approaches

A. Properties and computational features

In II-B we intuitively described how the update-rule changes the quantities using relations in the network. In this section we explain some of its properties and discuss how it can be interpreted using a probabilistic perspective. We then present some of the interesting computational features of the framework in terms of parallelism of the computation and distributivity of the components.

1) From a probabilistic perspective: Let us consider we have observed realizations of two random variables X and Y drawn from stationary distributions. Let us denote $\mathcal{X} = \{x_1, ..., x_i, ..., x_M\}$ and $\mathcal{Y} = \{y_1, ..., y_i, ..., y_N\}$ the realizations of these random variables.
What we call the value of a quantity X (resp. Y) we consider in our framework to be a point-wise estimate of the empirical mean (the first raw sample moment) $\bar{x}$ of the distribution of X (resp. Y). At any point in time when we receive a realization x_i of X we can compute a new estimate of $\bar{x}$.
If we have received M realizations of X and N realizations of Y that we assume come from the same random process, what would be the mean of a random variable Z relying on these two random variable realizations, *i.e* what would be the value of the quantity Z?

Property 1. *The update equation $\bar{z} = (1-\eta)\cdot\bar{x} + \eta\cdot\bar{y}$ with $\eta = \frac{N}{M+N}$ computes the update that gives the empirical mean point-wise estimate of a random variable Z combining M realizations of a random variable X and N realizations of Y as if they were coming from a single distribution.*

Proof: We can compute the empirical mean $\bar{x}$ of the distribution of the random variable X as $\bar{x} = \frac{1}{M}\sum_{\mathcal{X}} x_i$ analogously the empirical mean $\bar{y}$ for the random variable Y is $\bar{y} = \frac{1}{N}\sum_{\mathcal{Y}} y_j$.
Let us introduce a random variable Z drawn from a distribution for which we assume that the M realizations of X and the N realizations of Y have been drawn, we denote $\mathcal{Z} = \mathcal{X} \cup \mathcal{Y} = \{z_1, ..., z_k, ...z_{|\mathcal{Z}|}\} = \{x_1, ..., x_M, y_1, ..., y_N\}$ the realizations for Z.
We can now compute the empirical mean $\bar{z}$ of the random variable Z:

$$\begin{aligned}\bar{z} &= \frac{1}{|\mathcal{Z}|}\sum_{\mathcal{Z}} z_k = \frac{1}{M+N}\Big(\sum_{\mathcal{X}} x_i + \sum_{\mathcal{Y}} y_j\Big) \\ &= \frac{M\bar{x} + N\bar{y}}{M+N} = (1-\eta)\cdot\bar{x} + \eta\cdot\bar{y} \text{ with, } \eta = \frac{N}{M+N}\end{aligned}$$

■

Property 2. *If the random variables X and Y are assumed to be drawn independently from normal distributions $p(X = x_i|\bar{z}, \hat{z}) = \mathcal{N}(\bar{z}, \hat{z})$ and $p(Y_j|\bar{z}, \hat{z}) \sim \mathcal{N}(\bar{z}, \hat{z})$ then the update-equation $\bar{z} = (1-\eta)\cdot\bar{x} + \eta\cdot\bar{y}$ is maximizing the likelihood of $\bar{z}$ given the realizations $\mathcal{Z}$.*

Proof: Maximizing the likelihood is equivalent to minimizing the negative log-likelihood:

$$\bar{z}^* = \arg\min -\log\mathcal{L}$$

Also, since $\mathcal{Z} = \mathcal{X} \cup \mathcal{Y}$, and assuming measurements are independent allows to rewrite the minimized objective in terms of the probabilities of X and Y:

$$\begin{aligned}\mathcal{L} &= -\log\Big(\prod_k p(Z = z_k|\bar{z}, \hat{z})\Big) \\ &= -\log\Big(\prod_i p(X = x_i|\bar{z}, \hat{z})\prod_j p(Y = y_j|\bar{z}, \hat{z})\Big) \\ &= \sum_i -\log p(X = x_i|\bar{z}, \hat{z}) + \sum_j -\log p(Y = y_j|\bar{z}, \hat{z})\end{aligned}$$

Using the fact that X and Y are two Gaussian distributed random-variables, and simplifying constants in the objective:

$$\bar{z}^* = \arg\min \sum_i (x_i - \bar{z})^2 + \sum_j (y_j - \bar{z})^2$$

The optimum $\bar{z}^*$ of this objective is reached when the right hand term's derivative is null, yielding the desired result:

$$\begin{aligned}
&\frac{\partial}{\partial \bar{z}} \Big(\sum_i (x_i - \bar{z})^2 + \sum_j (y_j - \bar{z})^2 \Big) \Big|_{\bar{z}=\bar{z}^*} \overset{!}{=} 0 \\
&\Rightarrow \sum_i x_i - M \cdot \bar{z}^* + \sum_j y_j - N \cdot \bar{z}^* \overset{!}{=} 0 \\
&\Leftrightarrow \bar{z}^* = \frac{1}{M+N} \Big(\sum_i x_i + \sum_i y_i \Big) \\
&\Leftrightarrow \bar{z}^* = (1-\eta) \cdot \bar{x} + \eta \cdot \bar{y}, \text{ with } \eta = \frac{N}{M+N}
\end{aligned}$$

■

Using Property 2 the update-rule is interpreted from a probabilistic perspective as performing maximizing likelihood when the measurements $\mathcal{Z}$ are assumed to be drawn from a Gaussian distribution. In the case we are not ready to make such a strong assumption, the interpretation of the update-rule corresponds in computing the empirical mean of a distribution of unknown form as seen with Property 1. Note that we could write other update-rules to match higher-order moments of our distributions very similarly –here shown for second order moments–: $\hat{z} = (1-\eta) \cdot \hat{x} + \eta \cdot \hat{y} + (1-\eta)\eta \cdot (\bar{x} - \bar{y})^2$. The update-rule also generalizes to multidimensional random variables and multivariate distributions.

To interpret these results in terms of source and target quantities in our framework, let the random-variable X represent the target quantity to be updated and Y the direction applied to the source quantities (several random-variables that would be transformed by a functional mapping described by the direction). The meaning of the hyper-parameter η corresponds in this probabilistic view to how much old samples (from X) and new samples (from Y) we consider having gathered to estimate the value of the quantity Z represented via the point-wise estimate of its mean $\bar{z}$. The fewer samples for X and the more samples for Y naturally leads to update Z faster, which is what η describes.

2) *Computational features:* Using the simple message-passing algorithm and the structure of network we introduce offers interesting computational features to our system:

a) Joint approximate inference: is carried in the framework for all the quantities simultaneously. In the algorithm we present in Figure 2, when the input of the network is fed by the sensor measurements all the relations are picked in turn and update their quantities. Each relation is picked and updated T times before proceeding to the next measurement, meaning that for each measurement reported by a sensor, all the quantities –including the ones not directly connected to the sensor itself– are updated from all possible directions. This lets information flow in the network to allow the values of the quantities to converge using information from the different parts of the network. As we will show it in Section IV, this is a key feature that allows, for instance, to resolve under-constrained problems by letting the quantities slowly and consensually agree on their values.
In this respect, the framework we suggest is radically different from traditional approaches processing information in a pipeline fashion where the output of a "processing block" feeds the input of another one in a purely feed-forward manner. In this framework, information goes back and forth in the network and quantities that have updated their values are questioned back multiple times by their relations to other quantities. This whole process lets the network relax in a state where the values of the non-observed quantities have to mutually agree to interpret the sensory input: relations act as "soft-constraints".

b) Modularity: in the framework allows us to add a new quantity in the network without having to modify anything to its already existing structure, it can be introduced by adding a relation to other quantities already present in the network. Similarly adding a sensor is simply done by feeding its measurements in the appropriate quantity. As well, if a sensor would suddenly fail, the network continues to run even-though inference can be altered since it benefits from less information flowing in.

c) Distributed computation: can be achieved in the framework since each target quantity can be seen as an independent unit treating incoming messages and blending them with its own value using the suggested update-rule. Thus, a system that would run the whole network can be easily divided in sub-systems each responsible of their target quantity, the only necessary communication between them is the sharing of the messages.
Note that a target quantity that would be involved in multiple relations can treat differently messages coming from two different directions, hence not limiting the distributive feature of the network.

d) Parallelism: is also made possible in the framework thanks to a form of "conditional independence" in the network. In fact, any quantity in the network can be seen as independent of the others given the quantities involved in its relations. In some sense, and using the graphical models terminology, this defines a "Markov blanket" of a quantity.
Practically if one quantity is not in the Markov-blanket of the other, they can both be updated simultaneously. Note that sequential and parallel updates do not guarantee to have the same properties since changing the order in picking the relations typically changes the outcome of the approximate-inference.

B. *In relation to other approaches*

In this section we relate our framework to existing approaches, in the light of factor-graphs, and thus motivate the choices made in the representation of the values and of the quantities stored in the network of quantities.

1) *Modelling a problem in a network:* Having quantities connected in a graph over which an error function is defined locally for cliques of variables is the concept of factor-graph modelling. Algorithms such as belief propagation allow to efficiently perform exact inference for graph having a tree structures when it comes to marginalize (sum-product) or maximize (max-product) one or more quantities. Factor-graphs thus propose a unifying view generalizing concepts that first

emerged in statistical physics with Markov Random Fields [2], and are now used in more recent applications like error-correction and turbo-codes [3] or in Bayesian networks [4], Kalman filters [5] and Hidden Markov Models [6]. However, many practical models cannot be expressed using tree-like structures without making serious assumptions compromising their expressiveness. Typically, one likes to think of images as graphs where pixels are organized on an grid connected to their four-neighbours. In order to perform inference for these models, extensions to belief propagation like variants of the so-called loopy belief propagation were proposed, generally loosing all the convergence and exact inference guarantees that stood for trees. Therefore, our efforts are concentrated to suggest a framework in which models can be easily created and extended using a simple yet efficient inference algorithm.

2) Representing the values of the quantities and passing messages: One of the main issue in the instantiation and practical implementation of factor-graphs is the choice of a representation for the values of the quantities and for the messages that are passed. Should quantities be thought as random-variables for which assumed parametrized distributions are stored, like in Bayesian networks or Kalman filters? Is it better to store samples as realization of these variables, like in particle filters? Also, how to deal with quantities taking their values in continuous state-space: should we pass point-wise estimates, histograms or parametrized functions? To represent values of the quantities Kalman-filters –in their original formulation– restrict themselves in assuming random-variables drawn from Gaussian distributions, with linear relations so that the recursive computation of the quantities is closed under these transformations. In particle filters, samples (the particles) representing the values of the quantities can be propagated through non-linear relations, and can represent "arbitrary" distributions. A drawback is that many of them might be needed to faithfully run inference on a complex model consequently yielding intensive computation.
In this work, a choice we made in our framework is to compress the information of random-variable distributions in their moments. In III-A1 we show how we consider the values of our quantities to be representing the first order moments and explain the update-rule in that respect. We also show that if we meant to assume Gaussianity of the distributions, the update-rule can be seen as performing maximum-likelihood. We also work in maintaining and propagating second order moments in the network so that values, that would now be composed of these two numbers, can represent their own belief.

IV. Example and Results

As an instance of this framework we build a system that is able to infer the interpretation of a static scene (no object is assumed to move) imaged by an observer solely able to rotate, jointly in terms of its light-intensity, spatial gradient, temporal gradient, optic-flow and ego-motion.
To demonstrate the inference capabilities of our system, we attach a Dynamic Vision Sensor [1] that only reports pixel-wise temporal contrast changes in light-intensity in an asynchronous stream of events. Contrary to a conventional camera sensing an absolute amount of light intensity for a whole frame of pixels during a fixed period of time (typically at 22-25Hz), the DVS produces a stream of events asynchronously produced at each pixel beased on the contrast changes in the imaged scene.

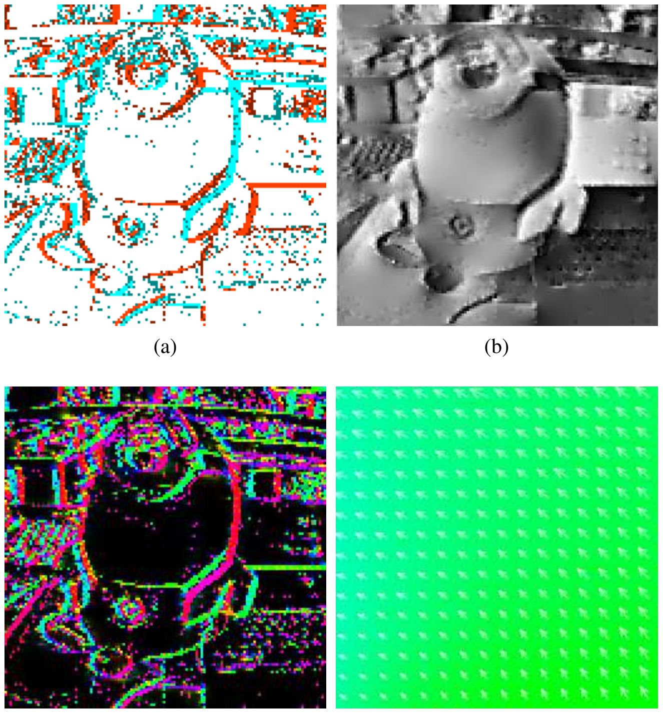
(a) (b) (c) (d)

Fig. 3: We ran the network presented in Section IV for $T = 50$ iterations or our message passing algorithm. (a) shows the Dynamic Vision Sensor (DVS) events: this the only input to the network, blue (resp. red) encodes "positive" (resp. negative) contrast changes. (b) shows the intensity inferred from the network. (c) shows the spatial gradient of the light intensity, the hue encodes the gradient direction. (d) shows the optic-flow, the hue encodes its direction while the saturation encodes its magnitude.

Thus, the output of the sensor is a stream of events located by their (x,y) coordinates with signed polarities (+1 / -1) indicating positive or negative changes in the scene luminance, timestamped with a high time resolution ($1\mu s$).

A. A simple vision system using visual quantities & relations

In the following we detail how we built a network, as seen in Figure 1 to model the simplified vision task described above.

1) R_1: Events coming from the DVS are related to temporal variations of light intensity: The first relation we can state is between the events $E_{x,y}$, or contrast changes produced by the DVS we use in our system and the temporal variations of the light intensity $V_{x,y} = \frac{\partial I_{x,y}}{\partial t}$ at a pixel location (x, y), namely: $E_{x,y} = f(V_{x,y})$. The function f describes the internal processing mechanism of our sensor to binarize variations of light-intensity –modelling it is out of the scope of this work–. $E_{x,y}$ is an instance of a scalar quantity in our framework. It can be thought as being organized on a grid, each pixel (x, y) in this grid contains its estimated value of $E_{x,y}$. It is connected via a relation to an homologous quantity at position (x, y) in another grid $V_{x,y}$. We call these grids: quantity maps. The same relation exists between all (x, y) belonging to the map E and their homologous in map V.

2) R_2: Spatial variations of the light intensity are related to its temporal variations by the optic-flow equation: Computer vision is traditionally working with images using grey-level

intensities. Inferring these intensities $I_{x,y}$ from the events $E_{x,y}$ we get from the DVS is a non-trivial problem and would typically require a dedicated, computationally expensive algorithm as presented in [7].
However there is a simpler relation between the spatial variations of the light intensity, also called spatial gradient and denoted $\vec{G}_{x,y} = (\frac{\partial I_{x,y}}{\partial x}, \frac{\partial I_{x,y}}{\partial y})^T$ and the temporal variations of the light intensity $V_{x,y}$. More specifically the total variation of light intensity can be written with the chain rule as the sum of its spatial and temporal variations: $\frac{dI_{x,y}}{dt} = \frac{\partial I_{x,y}}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial I_{x,y}}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial I_{x,y}}{\partial t}$. Assuming now that luminance in the scene is constant on a small amount of time leads to write $\frac{dI_{x,y}}{dt} = 0$, yielding the classical optic-flow constraint equation (OFCE) [8] after rearranging and identifying the terms with V and $\vec{G}_{x,y}$: $V_{x,y} = -\vec{G}_{x,y} \cdot \vec{F}_{x,y}$, denoting the optic-flow $\vec{F}_{x,y} = (\frac{\partial x}{\partial t}, \frac{\partial y}{\partial t})^T$.
Note that given an image gradient $\vec{G}_{x,y}$ and temporal variations of the light intensity $V_{x,y}$, one can only solve for the flow $\vec{F}_{x,y}$ in the gradient direction. Across intensity regions where there is no spatial gradient, *i.e* with light intensity being uniform in space, the problem is even worse since the OFCE is "ill-conditioned" and cannot be solved for a particular optic-flow. The above equation is under-constrained and this problem is known as the aperture problem [9]. A variety of methods have been designed to regularize the OFCE, amongst them are the popular variational methods [10] or multi-grid differential methods [11].

3) R_3: Light intensity and its gradient: Writing the relation between the light-intensity and its spatial gradient choosing the gradient as a target quantity is trivial since by definition: $\vec{G} = \vec{\nabla} I$. How to update the intensity based on the spatial gradient is a slightly more complicated issue. By taking the divergence on each side the following relation appears: $\triangle I = \nabla \cdot \vec{G}$ which is in the form of a Poisson equation and yields a natural candidate for the update of I.

4) R_4: Assuming a rotation in front of a static scene regularizes the optic-flow: The DVS only produces events when there is a change in intensity of the imaged scene: for instance when the observer moves. If we restrict our visual system to rotations in a static scene, the instantaneous motion of the system is entirely specified with a single rotation described by a pseudo-vector $\vec{R}$ whose orientation is the axis of the rotation and which magnitude corresponds to the angular velocity.
Given these two simplifying assumptions for our system: a perfect rotation and a static scene it is now possible for a particular rotation $\vec{R}$ to influence the optic-flow that a pixel looking in the direction of $\vec{C}_{x,y}$ should experience.

B. Inference performed in the visual network

We use the message-passing algorithm we introduced in Section II and show images of the resulting inference carried in the network in Figure 3. We binned 20ms of events in 128×128 frames and ran $T = 50$ iterations for each relation in the network. This is achieved at approximately 60Hz on a recent Intel mobile processor, and about 170Hz on a GPU allowing real-time applications. This network is the object of an implementation on an analogue CPA [12], simulations and preliminary results on an actual device indicate the system could run in real-time using about $3W$ of power.

V. Conclusion

In this work we introduced a framework that allows to perform approximate inference in the form of a network of quantities and relations. Observable quantities reported by sensors are used, jointly with a set of a-priori defined relations, to infer other quantities of interest. The relations are used to softly constrain the system to find a consistent solution, *i.e* an assignment of the values satisfying the relations for all the quantities in the network. We suggested a very simple update-equation, that smoothly blends the actual values of the quantities with the values imposed by the relations. We showed that a simple message-passing algorithm performing the update of all the quantities several times for all the relations picked is performing a maximum-likelihood update. Such a framework equipped with the message passing algorithm we propose enables computation to be performed in parallel and can trivially be distributed allowing to execute it on dedicated hardware such as GPGPUs or CPAs. As inference is performed jointly for all the quantities our approach differs from common feed-forward approaches where neither loops nor feedbacks between quantities are used.
We demonstrated the capabilities of a small visual system built from the principles of the framework we developed in this work. Using the inference algorithm we proposed, we successfully inferred a visual interpretation in terms of optic-flow, ego-motion, light intensity and spatio-temporal derivatives of the light intensity from a DVS input. The solution of such problems is usually addressed by dedicated algorithms that are "hot" topics of on-going vision research.

References

[1] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128×128 120 db 15 μs latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Feb. 2008.

[2] E. Ising, "Beitrag zur theorie des ferromagnetismus," *Zeitschrift für Physik*, vol. 31, no. 1, pp. 253–258, 1925.

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, *Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-codes*, 1993, vol. 2.

[4] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[5] R. Kalman, "A new approach to linear filtering and prediction problems," *Trans. of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[6] L. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state markov chains," *Ann. Math. Statist.*, vol. 37, no. 6, pp. 1554–1563, 1966.

[7] H. Kim, A. Handa, R. Benosman, S.-H. Ieng, and A. Davison, "Simultaneous mosaicing and tracking with an event camera," in *Proc. of the British Machine Vision Conference, BMVC 2014*, 2014.

[8] B. Horn and B. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.

[9] T. Poggio, V. Torre, and C. Koch, "Computational vision and regularization theory," *Nature 317*, pp. 314–319, Sept. 1985.

[10] J. Weickert and C. Schnörr, "A theoretical framework for convex regularizers in pde-based computation of image motion," *Internat. Journal of Computer Vision*, vol. 45, no. 3, pp. 245–264, 2001.

[11] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," *Proc. of Imaging Understanding Workshop*, vol. 17, pp. 121–130, 1981.

[12] J. Martel, M. Chau, P. Dudek, and M. Cook, "Toward joint approximate inference of visual quantities on cellular processor arrays," in *Proc. of the IEEE Internat. Symp. on Circuits and Systems, ISCAS 2015*, 2015.