ADVANCED ELECTRONIC MATERIALS Open Access
www.advelectronicmat.de

Hardware Implementation of Bayesian Decision-Making with Memristors

*Lekai Song, Pengyu Liu, Yang Liu, Jingfang Pei, Wenyu Cui, Songwei Liu, Yingyi Wen, Teng Ma, Kong-Pang Pun, Leonard W. T. Ng, and Guohua Hu**

Brains perform decision-making by Bayes theorem – events are quantified as probabilities and based on probability rules, computed to render the decisions. Learning from this, Bayes theorem may be applied to enable efficient user–scene interactions. However, given the probabilistic nature, implementing Bayes theorem with the conventional deterministic computing hardware can incur excessive computational cost and decision latency. Though challenging, here a probabilistic computing approach is presented based on memristors to implement Bayes theorem. Memristors are integrated with Boolean logic circuits and, by exploiting the volatile stochastic switching of the memristors, realize probabilistic Boolean logic operations, key for Bayes theorem hardware implementation. To empirically validate the efficacy of the hardware Bayes theorem in enabling user–scene interactions, lightweight Bayesian inference and fusion operators are designed using the probabilistic logic circuits and apply the operators in road scene parsing for self-driving, including route planning and obstacle detection. The results show the operators can achieve reliable decisions in less than 0.4 ms (or equivalently 2500 fps), outperforming human decision-making and the existing driving assistance systems.

1. Introduction

Brains are adept at making timely reliable decisions for events even with uncertainties. This ability is attributed to the *Bayes theorem*, a mathematical principle used to determine the probability of an event to occur when given certain conditions.[1] In Bayes theorem, the events, conditions, and their relations are all quantified and computed as probabilities, following the fundamental probability rules, to render timely reliable decisions. Bayes theorem therefore shows promise in visualizing the potential risks and decision confidence in addressing problems in, for instance, user–scene interactions.[2,3] Taking road scene parsing in self-driving as an example (**Figure 1a**), the car of investigation in blue based on Bayes theorem may make timely decision with reliability for lane changing when considering all relevant traffic conditions, e.g. the incoming cars, the pedestrian, and the ambient light and weather conditions. However, given the probabilistic nature, implementing Bayes theorem can be challenging due to the lack of specialized hardware to perform probabilistic data representation and computation. Using the conventional deterministic computing hardware systems for Bayes theorem implementation can inevitably incur excessive computational cost and decision latency.[4]

Idealistic Bayes theorem implementation demands a probabilistic computing solution to realize the probabilistic data representation and computation. One feasible approach is to encode the data into streams of random bits, termed *stochastic numbers*, where the probability of the bit 1s determines the values.[5] The stochastic numbers, when fed into standard Boolean logic gates, can undergo bitwise Boolean operations and yield reconstructed stochastic numbers as the computational results. The inputs and results obey statistical relations, thereby simplifying the logic operations[5–7] and facilitating Bayes theorem implementation. Early electronic circuits, e.g., those based on *linear feedback shift registers* (LFSR), have demonstrated the ability in encoding stochastic numbers by exploiting the stochasticity inherent in the circuits.[8–10] These circuits can be further integrated with Boolean logic circuits for stochastic number processing conforming to probabilistic computing rules. However, intricate

L. Song, P. Liu, Y. Liu, J. Pei, S. Liu, Y. Wen, K.-P. Pun, G. Hu
Department of Electronic Engineering
The Chinese University of Hong Kong
Shatin, N. T., Hong Kong SAR 999077, China
E-mail: ghhu@ee.cuhk.edu.hk
Y. Liu
Shun Hing Institute of Advanced Engineering
The Chinese University of Hong Kong
Shatin, N. T., Hong Kong SAR 999077, China
W. Cui, T. Ma
Department of Applied Physics
Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong SAR 999077, China
L. W. T. Ng
School of Materials Science and Engineering
Nanyang Technological University
Singapore 639798, Singapore

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aelm.202500134

DOI: 10.1002/aelm.202500134

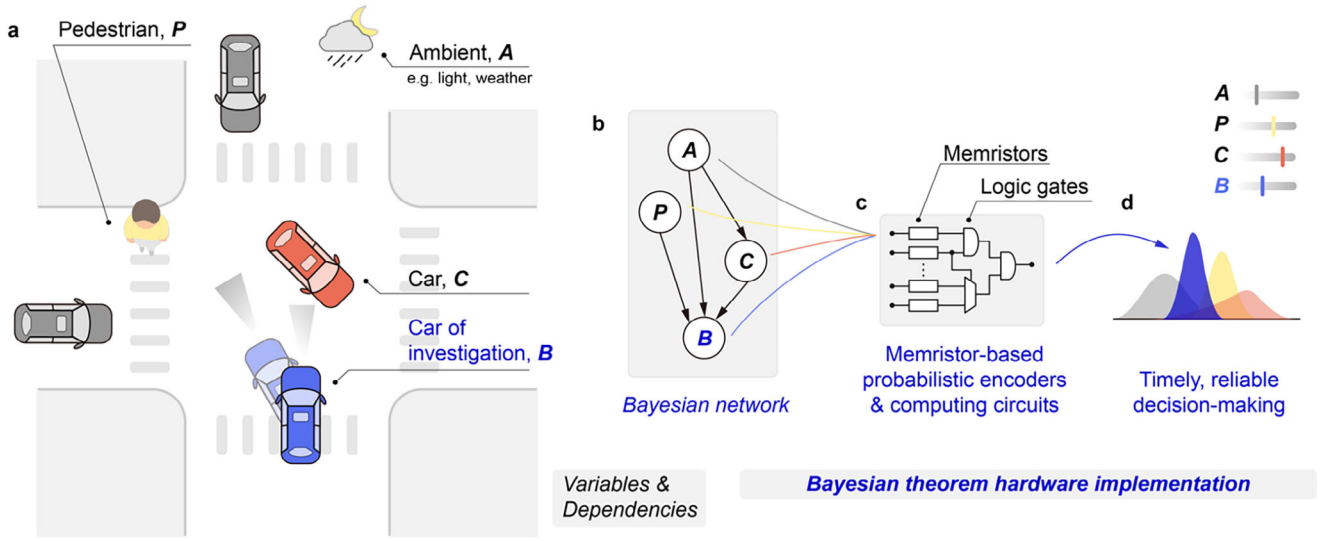

Figure 1. Bayesian decision-making. a) Road scene parsing in self-driving, showing the car of investigation decides lane changing based on Bayes theorem when considering all relevant traffic conditions, e.g. the incoming cars, the pedestrian, and the ambient light and weather conditions. b–d) Hardware implementation diagrams of Bayes theorem with memristors, where the memristors are used to enable probabilistic data encoding and logic operations and as such, render timely reliable Bayesian decision making based on probabilistic computational rules.

pre-/post-processing circuit designs are often required to ensure the results are not corrupted by improper correlations between the stochastic numbers.[11,12] As nanomaterial and electronics technology develop, devices with intrinsic stochastic properties were reported, e.g. memristive electronics based on 2D materials.[13–16] These devices, exploiting their stochastic properties, can inherently realize probabilistic data representation and computation and are, therefore, promising for designing compact hardware Bayes theorem systems.

Herein, we present a memristor-enabled probabilistic computing approach for hardware implementation of Bayes theorem, as illustrated in Figure 1b–d. We integrate volatile memristors with Boolean logic circuits and, by exploiting the volatile stochastic switching of the memristors, realize probabilistic data encoding and logic operations that are essential for Bayes theorem implementation. Using the memristor-enabled probabilistic logic circuits, we design lightweight Bayesian hardware operators capable of performing key Bayesian functions, e.g. inference and fusion. These hardware operators are applied in road scene parsing for self-driving, including route planning and obstacle detection, to empirically validate their effectiveness in practical user–scene interactions. Our results show the operators can perform reliable decisions in less than 0.4 ms (or equivalently 2500 fps). This outperforms human decision-making and the advanced driver assistance systems, underpinning the potential to enable efficient Bayesian user–scene interactions in, for instance, self-driving, virtual reality, and robotics.

2. Results

2.1. Volatile Memristors

Memristors are resistive switching devices, and the memory volatility varies depending on the switching mechanisms[17] – the volatile memristors behave with self-reset threshold switching.[18] In practical switching, memristors tend to exhibit nonideal stochasticity, as a result of the underlying stochastic switching dynamics. For example, arising from the stochastic diffusion of the conductive elements, filamentary memristors switch with stochasticity.[19] The stochasticity, however, along with the self-reset threshold switching behavior, can potentially enable the volatile memristors to perform probabilistic data encoding with reduced hardware and computational cost.[20–22]

Toward probabilistic data representation for hardware Bayes theorem implementation, we develop volatile filamentary memristors from solution-processed hexagonal boron nitride (hBN), following our previous report.[23,24] We start with liquid-phase exfoliation of hBN, followed by memristor fabrication via photolithography (Figure S1, Supporting Information). **Figure 2**a shows the memristors, where the top electrode develops with silver. To assess the switching behavior, we conduct sweeping test of the memristors (Figure 2b). In a typical switching, the memristor switches to a low resistive state when the bias is over a threshold voltage V_{th}, and then spontaneously resets to a high resistive state when the bias recedes below a hold voltage V_{hold}. Unlike the typical filamentary memristors where the interplay between the material defects and the filaments leads to non-volatile memory behavior,[25,26] the dangling-bond-free nature of hBN contributes to the volatile memory. Particularly, a thermal effect can participate in the switching process and the generation of abundant Joule heat can intensify the atomic oscillation.[27–29] As a result, the silver filaments through the hBN layer can maintain an equilibrium yet metastable state, leading to a fast threshold switching. With the decline and removal of the external bias, the memristors spontaneously reset as the filaments rupture. See also Figure S2 (Supporting Information) for the transient switching of the memristors, showing a switching time of ≈50 ns, a relaxation time of ≈1100 ns, and a switching energy of ≈0.16 nJ.

As the sweeping test continues, the memristor exhibits consistent cycle-to-cycle stochasticity in V_{th} (2.08 ± 0.28 V) and V_{hold}

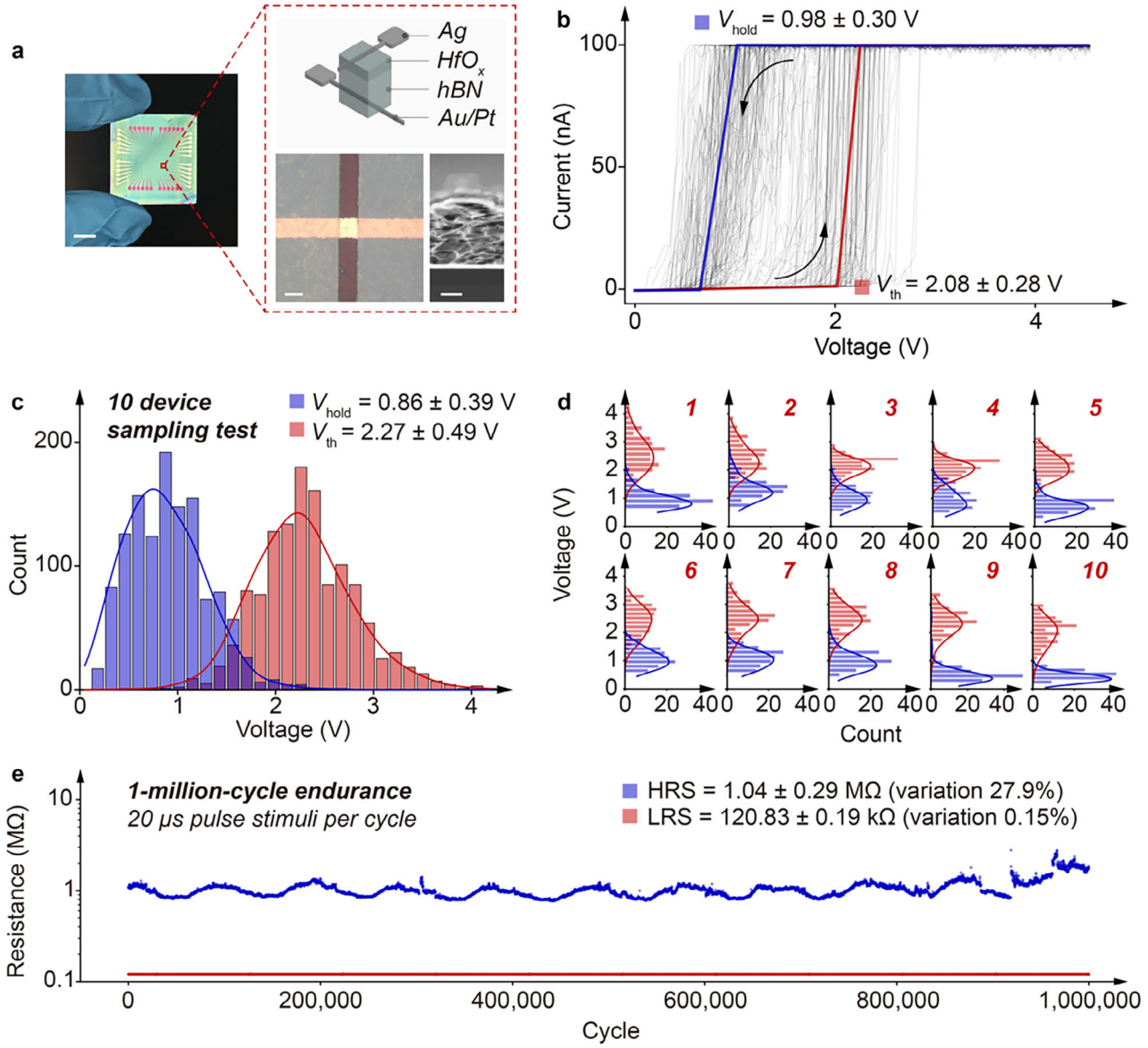

Figure 2. Volatile switching memristors. a) 12 × 12 hBN memristors in a crossbar configuration, and the optical microscopic and cross-sectional transmission electron microscopic images of a typical memristor. Device area ≈20 × 20 μm^2. Scale bars – 5 mm, 20 μm, and 100 nm. b) Current-voltage output from a typical memristor, showing 128-cycle stochastic yet stable switching with a switching ratio of ≈10^5. A compliance current of 100 nA is set. c) Overall and d) individual distributions of the measured V_{hold} and V_{th} of 10 randomly selected memristors in (a), along with the corresponding Gaussian fittings. 128 cycles are tested for each memristor. e) Endurance test of a typical memristor undergoing 10^6 consecutive test cycles under pulsed stimuli. For each test cycle, a 20 μs voltage pulse of 10 V is set to fully program the memristor and an 80 μs voltage pulse of 0.1 V is set to read the output. The output of the memristor is amplified by an operational amplifier and measured by an oscilloscope. The high and low resistance states in each test cycle are computed based on the oscilloscope measurement, and plotted in blue and red, respectively. Both the states remain stable throughout the endurance test.

(0.98 ± 0.30 V) in switching. The stochasticity arises from the underlying switching dynamics of the memristors, i.e., the stochastic diffusion of the silver ions.[19,27,29] To assess the stochasticity further, we perform sampling test of ten randomly selected devices from the array (Figure S3, Supporting Information). As demonstrated, all the ten sampled devices exhibit a stable stochasticity in switching. We plot the overall device-to-device stochasticity in Figure 2c, with the distributions of the measured V_{hold} and V_{th} of the individual sampled memristors in Figure 2d. The device-to-device stochasticity is ≈8% in V_{th}, measured by the *coefficient of variation*, i.e., the ratio of the standard deviation to the mean of V_{th} in sampled memristors, proving a high device-to-device uniformity. Having confirmed the stable, uniform cycle-to-cycle and device-to-device stochastic switching, we perform stochastic process modeling on V_{th}. The measured V_{th} of each of the sampled memristors follows a mean-reverting behaviour with random fluctuations, conforming to the Ornstein-Uhlenbeck process (Figure S4, Supporting

Information). Ornstein-Uhlenbeck process describes a stochastic process in a dynamical system.[20] This indicates the stability of the switching stochasticity of our memristors in long-term switching. Indeed, pulsed cycling switching test proves a switching endurance of over 10^6 cycles (Figure 2e).

The stable volatile stochastic switching suggests that our memristors when exploited in probabilistic computing can encode the inputs into stochastic numbers in lightweight, where the volatile switching can eliminate resetting circuits and operations, while the switching stochasticity can facilitate stochastic number encoding. On the other hand, importantly, the device-to-device uniformity and high fabrication yield can allow simplified circuit design and calibration in practical probabilistic computing hardware implementation.

2.2. Probabilistic Logic Circuits

Using our volatile memristors, we then design stochastic number encoders (SNEs; **Figure** 3a) for stochastic number encoding with desired probabilities and correlations, aiming to realize probabilistic data representation as required by Bayes theorem. Here the memristors are integrated with comparators to implement lightweight SNEs. See also Figure S5 (Supporting Information) for the hardware implementation. When fed with pulsed signal V_{in}, the memristors are switched stochastically, and the output from the memristors carrying the stochasticity is binarized by the comparators via the reference V_{ref}. This yields stochastic numbers with probabilities, and the probabilities are well-regulated by V_{in} and V_{ref}. Notably, as designed, the stochastic numbers from an SNE are correlated, while those from two or more parallel SNEs are uncorrelated. Figure 3b presents the probability of uncorrelated stochastic numbers $P_{uncorrelated}$ with respect to V_{in}. As V_{in} increases, $P_{uncorrelated}$ is increased, as the memristors tend to switch on with a higher probability. This relation follows a sigmoidal fit, proving that V_{in} can effectively regulate $P_{uncorrelated}$. Similarly, Figure 3c presents the probability of correlated stochastic numbers $P_{correlated}$ with respect to V_{ref}. In this case, $P_{correlated}$ is decreased as V_{ref} increases, as V_{ref} serves as the threshold for binarization. This relation also follows a sigmoidal fit, proving that V_{ref} can also effectively regulate $P_{correlated}$. Here we note that $P_{uncorrelated}$ can exhibit a more prominent deviation than $P_{correlated}$, as two or more parallel independent memristors are required in implementing an uncorrelated SNE and the device-to-device variations among the memristors can lead to the more prominent deviation. The minor deviation in $P_{correlated}$ primarily arises from the noise and operation of the memristor and the correlated SNE circuit. Nevertheless, as demonstrated, our SNEs can perform stochastic number encoding with desired and well-regulated probabilities and correlations, suggesting that our SNEs can facilitate integration with Boolean logic circuits for performing probabilistic computing.

As examples, here we show the integration of our SNEs with compact Boolean logic circuits to build probabilistic AND and MUX in uncorrelation, as they can conduct the key Boolean operations for Bayes theorem (Figure 3d). For probabilistic AND, two parallel SNEs are connected to an AND gate, and the uncorrelated stochastic numbers from the SNEs serve as the inputs to the AND gate. Upon operation, based on the $P_{uncorrelated}$-V_{in} relation in Figure 3b, the SNEs when fed with pulsed signals output uncorrelated stochastic numbers, denoted as a and b, with probabilities of $P(a)$ and $P(b)$, respectively. a and b are then bit-by-bit fed into the AND gate to output a stochastic number, denoted as c, with a probability of $P(c)$. We show in Figure 3e the corresponding stochastic numbers and probabilities from the hardware test. The statistical relation, i.e., $P(a)P(b) \approx P(c)$, proves that the probabilistic AND successfully functions as a multiplier to conduct one-step multiplication of the stochastic numbers. Here we note that, compared to the conventional binary multipliers, the probabilistic AND not only simplifies the circuit design but also reduces the computational cost. The probabilistic AND can also be configured in correlation. In this case, SNEs are configured to output correlated stochastic numbers based on the $P_{correlated}$-V_{in} relation in Figure 3c, and the correlated probabilistic AND outputs the minimum of the inputs, i.e., $P(c) \approx min(P(a), P(b))$ (Figure 3e). We also design probabilistic MUX in both uncorrelation and correlation, and it performs one-step weighted addition (Figure 3e). Here the select and inputs of the probabilistic MUX must be uncorrelated to perform the operations properly (Figure S6, Supporting Information). We present in Table S1 (Supporting Information) a summary of the probabilistic logic circuits including AND, OR, XOR, and MUX in varying correlations.

The probabilistic logic circuits with the desired probabilities and correlations can potentially be integrated to enable probabilistic computing toward Bayes theorem implementation. Note that in the above demonstrations, the stochastic numbers are all encoded in 100-bit length. A longer bit length can render a higher precision in data representation and cancel the errors, however, with a higher computational cost.[5] Given the tradeoff, the bit length can be adjusted to accommodate tasks with different precision, error tolerance, and cost requirements.

2.3. Hardware Implementation of Bayesian Inference

To instantiate Bayes theorem, we apply our probabilistic logic circuits to implement Bayesian inference, a key decision-making method. Inference draws conclusions based on known facts. Despite being investigated in multiple disciplines including logic, neuroscience, and computer science, the essence of inference still remains elusive.[30–32] Nevertheless, the Bayes theorem provides a probabilistic solution to achieve inference.[16,33] In Bayesian inference, *prior knowledge* denotes the known facts, and *belief* the confidence (or reliability) of a certain decision. Importantly, as the prior knowledge and beliefs are quantified as probabilities, Bayesian inference can effectively revise the prior knowledge with the new information, based on the rules of probability, to make reliable decisions.

Our lightweight memristor-based probabilistic logic circuits are proven capable of performing probabilistic data representation and computation. Herein, we design a hardware operator using our probabilistic logic circuits to perform Bayesian inference (**Figure** 4a; see also Figure S7 (Supporting Information) for the hardware implementation), following the Bayesian inference theorem:

$$P(A|B) = \frac{P(A)\,P(B|A)}{P(B)} = \frac{P(A)\,P(B|A)}{P(A)\,P(B|A) + P(\neg A)\,P(B|\neg A)} \quad (1)$$

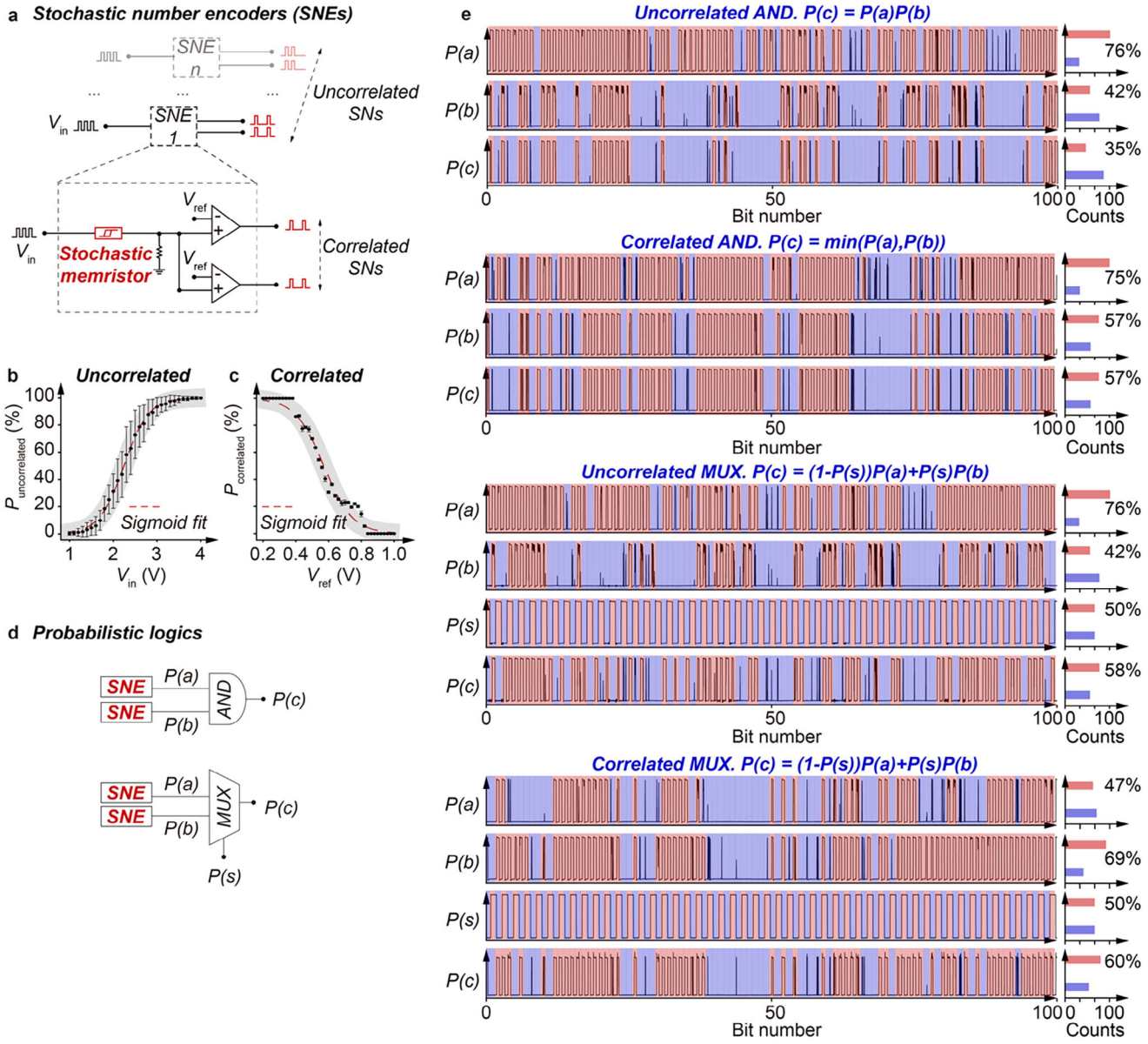

Figure 3. Memristor-enabled probabilistic logic circuits. a) Stochastic number encoders (SNEs), consisting of a memristor and a set of comparators. The output probability and correlation are regulated by the input V_{in} and reference V_{ref}. An SNE via circuit designs can output correlated stochastic numbers, while two or more parallel SNEs can output uncorrelated stochastic numbers. b) $P_{uncorrelated}$-V_{in} and c) $P_{correlated}$-V_{ref} relations of the SNEs in uncorrelation and correlation, well-fitting sigmoid functions $P_{uncorrelated} = 1/(1 + exp[-3.56(V_{in} - 2.24)])$ and $P_{correlated} = 1 - 1/(1 + exp[-11.5(V_{ref} - 0.57)])$. The error bar representing the mean and standard deviation at each data point is obtained from ten repeated samplings, where each sampling consists of 100-bit stochastic numbers. d) Probabilistic AND and MUX logic circuits in uncorrelation, implemented with the SNEs and standard AND and MUX logic circuits. The circuits can be reconfigured to implement other probabilistic logic circuits and accommodate varying correlations. e) Hardware tests of the probabilistic logic circuits in (d) in both uncorrelation and correlation. Bit 0s and 1s are marked in blue and red. For probabilistic MUX, the frequency of select s is half of the inputs to ensure both the inputs participate in the logic operations. The outputs of the probabilistic logic circuits in uncorrelation and correlation are consistent with the statistical relations in Table S1 (Supporting Information).

where the probabilistic AND and MUX are integrated to conduct one-step multiplication and weighted addition operations, respectively, for functioning as the numerator and denominator. The division is achieved with a probabilistic MUX plus a D-Flip-Flop, following a classic divider design, CORDIV.[34] When performing Bayesian inference, the prior knowledge with a prior probability $P(A)$ is updated when the new information with a marginal probability $P(B)$ is available, to yield a reliable decision with a posterior probability $P(A|B)$.

To investigate the effectiveness of our Bayesian inference operator in practical applications, we apply the operator in route planning for self-driving (Figure 4a). Route planning has been a challenging topic in self-driving, demanding great efforts to deal with complex traffic, vehicle driving, and lane-changing

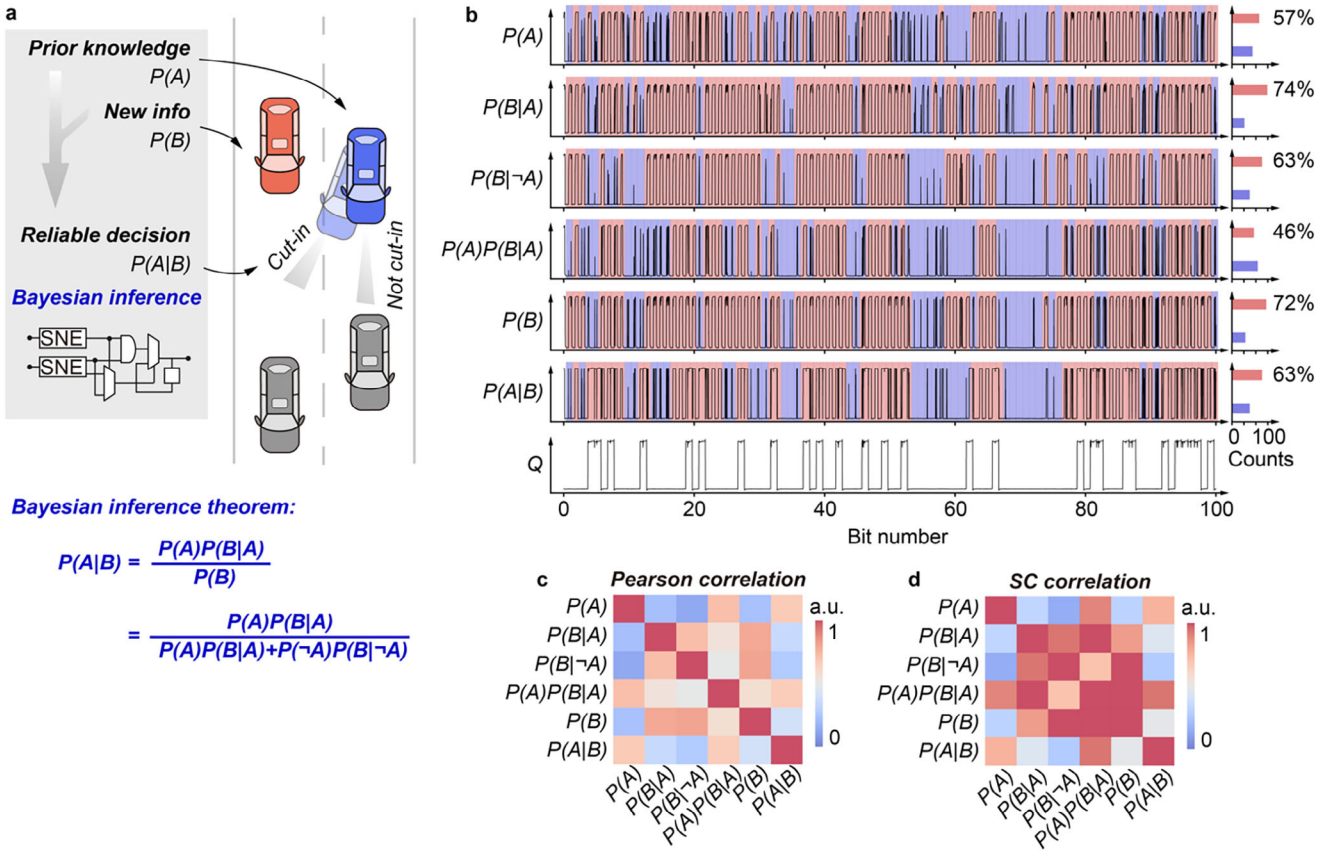

Figure 4. Hardware implementation of Bayesian inference. a) Memristor-enabled Bayesian inference operator and its workflow to perform Bayesian inference for route planning. Prior knowledge with a prior probability $P(A)$ is updated as the new information with probability $P(B)$ becomes available, yielding a reliable decision with a posterior probability $P(A|B)$. Here $P(A)$ represents the initial belief of a vehicle (blue) trying to cut in lanes, and $P(B)$ represents the probability of the vehicle spotting another vehicle (red) on the target lane. The Bayesian inference operator is implemented with the memristor-enabled probabilistic AND and MUX logic circuits. The probabilistic logic circuits are used to conduct multiplication and weighted addition operations respectively for functioning as the numerator and denominator conforming to the Bayesian inference theorem. The division is achieved with a probabilistic MUX plus a D-Flip-Flop, following a classic divider design for stochastic numbers, CORDIV.[34] See Figure S7 (Supporting Information) for the detailed circuit design and hardware implementation. b) Hardware test of the Bayesian inference operator to perform route planning for self-driving vehicles. $P(A)$ and $P(B)$ are manually set to initialise the operator. Stochastic numbers at the key nodes are plotted accordingly. Bit 0s and 1s are marked in blue and red. The result $P(A|B) > P(A)$ means the red vehicle in (a) can cut in lanes with higher confidence. This enhanced decision reliability is attributed to the comprehensive evaluation of both the vehicle itself and the traffic situation. Pairwise (c) Pearson and (d) SC correlations between the stochastic numbers in (b).

conditions. Particularly, given that any delay in decision making can lead to severe traffic accidents, achieving timely reliable decisions is critical in route planning. Bayesian inference has been considered promising for performing route planning,[35] as it can potentially make reliable decisions with prior knowledge (e.g., the traffic rules, road structures, and driving behaviors) and the latest lane information (e.g., the obstacles, accidents, and weather conditions). Here we assume a vehicle (blue) considering changing its lane, with probability $P(A)$ representing its initial belief to cut in the lane, based on the prior knowledge, i.e., the traffic conditions before lane changing. To ensure a safe lane change, the vehicle may evaluate the situation in the target lane to determine whether to cut in through Bayesian inference. In this case, $P(B)$ represents a marginal probability related to the latest lane condition, i.e., an incoming vehicle (red) on the target lane. With $P(A)$ and $P(B)$, the vehicle may update its belief based on the Bayesian inference theorem and get a posterior probability $P(A|B)$. For illustration, $P(A) = 57\%$ and $P(B) = 72\%$ are assumed here for Bayesian inference (Figure 4b). The operator integrates the prior knowledge with the latest lane information to make a final decision with a probability $P(A|B) = 63\%$. This probability aligns with the theoretical result in Equation 1, ≈61%, validating the effectiveness of our hardware operator in performing Bayesian inference. With the updated belief, i.e., $P(A) < P(A|B)$, the vehicle may make a lane-changing decision with a higher reliability. However, here we must note the decision-making is essentially not limited to lane changing. For example, when $P(A) > P(A|B)$, the hardware operator can decide to maintain its current lane with a higher reliability.

Note that, to make our hardware operator lightweight for timely decision-making, we maximize the sharing of the SNEs. This is possible with a well regulation of the probabilities and correlations of the stochastic numbers that ensures the proper functions of the Multiplier, Adder, and Divider. To clarify, we quantify

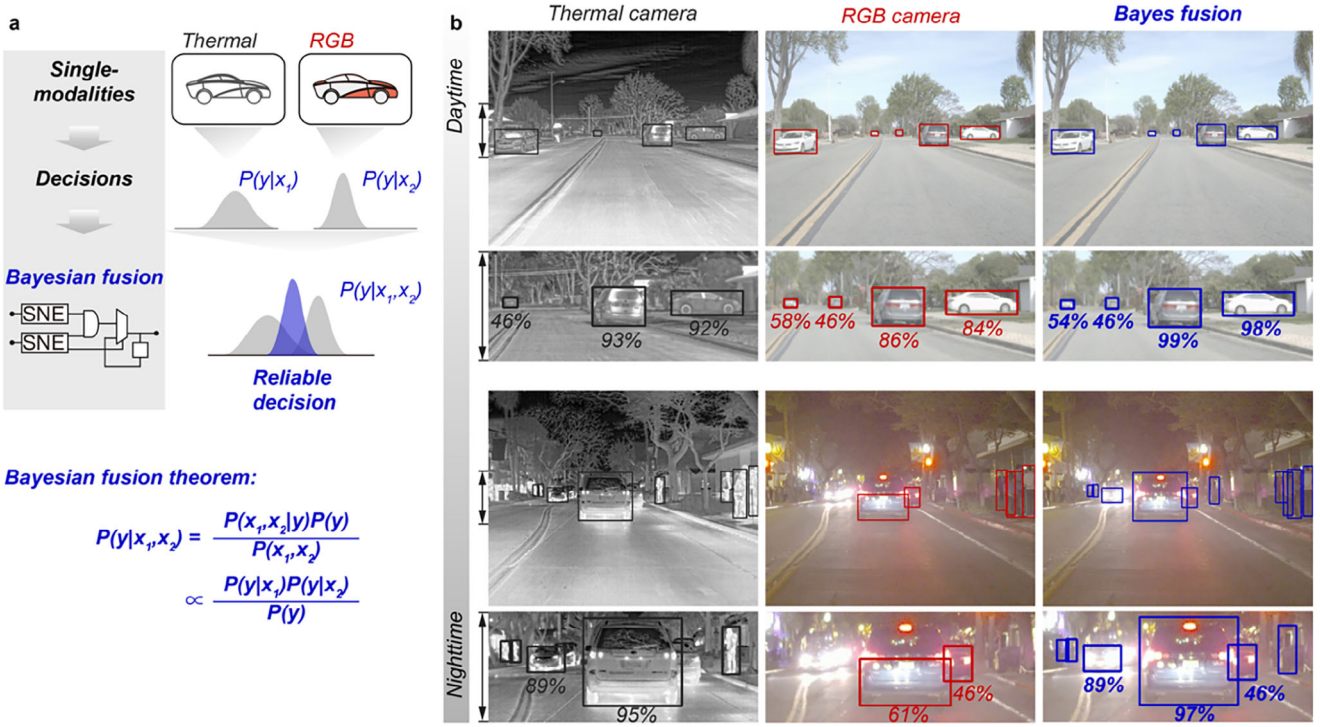

Figure 5. Hardware implementation of Bayesian fusion. a) Memristor-enabled Bayesian fusion operator and its workflow to perform Bayesian fusion. Pre-trained neural networks tailored for single modalities receive RGB and thermal information and output single-modal obstacle detection decisions, denoted as $P(y|x_1)$ and $P(y|x_2)$, where y represents the detected obstacle, and x_1 and x_2 represent the RGB and thermal information, respectively. The Bayesian fusion operator fuses the single-modal decisions into a reliable decision, denoted as $p(y|x_1,x_2)$, using the Bayesian fusion theorem. The Bayesian fusion operator is implemented with the memristor-enabled probabilistic AND and MUX logic circuits. The probabilistic logic circuits are used to conduct multiplication and weighted addition operations respectively for functioning as the numerator and denominator conforming to the Bayesian fusion theorem. The division is achieved with a probabilistic MUX plus a D-Flip-Flop, following a classic divider design for stochastic numbers, CORDIV.[34] See Figure S9 (Supporting Information) for the detailed circuit design and hardware implementation. b) Obstacle detection results before and after the Bayesian fusion of the RGB and thermal information under the different visibility conditions during daytime and nighttime. Before fusion, single-modal (RGB or thermal) networks typically lose certain target obstacles and make decisions with insufficient confidence. The Bayesian fusion operator effectively overcomes the single-modal shortages, addresses the target-missing and low-confidence issues, and achieves much more accurate and reliable results.

pairwise correlations between the stochastic numbers involved in the Bayesian inference using *Pearson correlation* and *SC correlation* (Figure 4c,d). The results confirm that all the probabilistic logic circuits in our operator work in the desired correlations and, as such, conduct the desired Boolean operations for performing Bayesian inference. Though our route planning studies the case where the decision-making is governed by one variable only (denoted as one-parent-one-child, i.e., A→B), our operator can be readily generalized to more cases, for instance, two-parent-one-child (A_1→B←A_2) and one-parent-two-child (B$_1$←A→B$_2$) (Figure S8, Supporting Information). In addition, arising from the fast switching of the memristors (<4 µs in total per bit, Figure S2, Supporting Information), our Bayesian inference operator with 100-bit stochastic number encoding can achieve Bayesian inference with less than 0.4 ms per frame, or equivalently 2500 fps. Note that here we neglect the delays of the comparators and Boolean logic circuits, as the switching of the memristors is the bottleneck of the operator. In this consideration, our operator outperforms human decision-making (0.7–1.5 ms reaction time[36]) and the advanced driver assistance systems (30–45 fps[37]) in route planning.

2.4. Hardware Implementation of Bayesian Fusion

Fusion is another key Bayesian decision-making. While inference integrates the past and present information, fusion considers the information from multiple modalities. Given that each modality represents fragmented information, Bayesian fusion fuses multimodal information to address the single-modal shortages and as such, achieve more reliable decisions.[30] As with inference, the information in Bayesian fusion is represented and computed by probabilities and can lead to excessive computational cost and latency. Here we investigate the potential of our probabilistic logic circuits in implementing Bayesian multimodal fusion in hardware. We design a fusion operator conforming the Bayesian fusion theorem (**Figure** 5a). See also S9 for the hardware implementation.

To investigate the effectiveness of our hardware Bayesian fusion operator in practical applications, we apply the operator in obstacle detection in self-driving (Figure 5a). Here we assume a self-driving vehicle equipped with RGB and thermal cameras, with the RGB camera to extract the obstacle signatures mainly in good visibility and the thermal camera obstacle signatures only

with heat emission. We test with a real dataset, *FLIR*,[38] that consists of aligned RGB-thermal images captured at different visibility conditions. Pre-trained edge networks tailored for processing RGB and thermal signals are used to output single-modal detection decisions, denoted as $P(y|x_1)$ and $P(y|x_2)$, where y represents the detected obstacles, and x_1 and x_2 the RGB and thermal inputs. Considering the limitations of the RGB and thermal cameras, the single-modal decisions tend to be unreliable. As shown (Figure 5b), the thermal camera loses certain obstacles, as a result of insufficient thermal emissions from the obstacles or received by the camera. RGB camera also misses obstacles, particularly during low-visibility nighttime. Besides, these two single modalities sometimes fail to provide confident decisions.

Given this, RGB-thermal Bayesian fusion has been considered promising for realizing reliable decision-making, particularly, during low-visibility nighttime, rain, and fog.[2,3] Our Bayesian fusion operator can fuse the single-modal RGB and thermal decisions into $p(y|x_1,x_2)$, following:

$$p\left(y|x_1,x_2\right)=\frac{p\left(x_1,x_2|y\right)p\left(y\right)}{p\left(x_1,x_2\right)}\propto p\left(x_1,x_2|y\right)p\left(y\right) \tag{2}$$

Assuming x_1 and x_2 are conditionally independent with given y, we have:

$$p\left(x_1,x_2|y\right)=p\left(x_1|y\right)p\left(x_2|y\right) \tag{3}$$

Substituting Equation 3 into 2, we get the multimodal fusion decision:

$$p\left(y|x_1,x_2\right)\propto p\left(x_1|y\right)p\left(x_2|y\right)p\left(y\right)=\frac{p\left(x_1|y\right)p\left(y\right)p\left(x_2|y\right)p\left(y\right)}{p\left(y\right)}$$
$$\propto\frac{p\left(y|x_1\right)p\left(y|x_2\right)}{p\left(y\right)} \tag{4}$$

The RGB-thermal Bayesian fusion results prove our operator is effective in addressing the single-modal detection issues (Figure 5b). On the one hand, our operator can integrate the single-modal RGB and thermal decisions to generate a final decision, resolving the issue of target missing. On the other hand, our operator can make more confident decisions by leveraging the Bayesian fusion theorem.

As with the Bayesian inference operator, our Bayesian fusion operator also maximizes the sharing of the SNEs and makes good use of the correlations to realize lightweight circuit design. Similarly, arising from the fast switching of the memristors, our Bayesian fusion operator with 100-bit stochastic number encoding can also easily achieve fusion with less than 0.4 ms per frame (equivalently 2500 fps). Considering the sampling rate of the cameras (10–30 fps[39]) and the processing speed of the pre-trained networks (300 fps[40]) deployed in real-world self-driving, our operator is a promising timely reliable obstacle detection solution for self-driving.

To accommodate larger-scale systems with higher-order conditional probabilities and allow fusion with more modalities, Equation 4 can be easily generalized by, according to ref. [41].

$$p\left(y|x_1,x_2\ldots x_{\mathrm{M}}\right)=\frac{\prod_{i=1}^{M}p\left(y|x_i\right)}{p(y)^{M-1}} \tag{5}$$

In this case, as more variables and dependencies are involved in the computation, the probabilistic results may become minimal. Hence, longer stochastic numbers can be required to represent the results to avoid underflow during the computation. This may gradually undermine the advantages of stochastic computing over the conventional binary computing in terms of energy consumption and latency, as we discuss in Note S1 (Supporting Information) and show in Figure S11 (Supporting Information). The correlations between the stochastic numbers in this case may also need to be re-evaluated to ensure correct stochastic computing due to the longer and sparser stochastic numbers. Here, "sparser" means a decreased ratio of bit 1s in stochastic numbers. Given the above considerations and concerns, the Bayesian operator needs to be optimized carefully before being deployed in large-scale systems.

3. Conclusion

In this work, we have implemented a memristor-enabled probabilistic computing approach for performing timely reliable Bayesian decision-making in hardware. We develop lightweight probabilistic logic circuits using volatile stochastic memristors and achieve probabilistic data representation and logic operations to facilitate hardware Bayes theorem implementation. As practical applications of the hardware Bayes theorem, we design Bayesian inference and fusion operators using the probabilistic logic circuits and prove their remarkable performance in road scene parsing for self-driving, including route planning and obstacle detection.

Given the demonstrated remarkable Bayesian decision-making performance and the compact circuit designs, as well as the scalability of our memristors, our hardware Bayes theorem approach is readily scaled up and generalized to develop Bayesian systems for real-world user–scene interactions. To explore the feasibility, we show via simulation that a large-scale Bayesian fusion can process high-throughput road scene video (Movie S1, Supporting Information). As demonstrated, the fusion resolves the target missing issue, with significantly higher chances to detect the obstacles than the single-modal thermal (by 85%) or RGB (by 19%). For example, the running child obscured by the harsh lighting is hardly detected by RGB. Besides, importantly, the fusion achieves decisions with higher confidence. Arising from the fast switching of the memristors, the large-scale Bayesian fusion can achieve user–scene interactions with a response time of less than 0.4 ms, or equivalently over 2500 fps, fulfilling the requirements of widespread applications including self-driving, virtual reality, robotics, and beyond. Though promising, realizing large-scale Bayesian systems requires large-scale fabrication of the memristors and high-level integration of the memristors with logic circuits, as well as parallel operations of the large-scale circuits. Amongst these, the fabrication yield and

uniformity of the memristors are still key concerns in large-scale manufacturing in current technological advances, and collaborative efforts from both the academia and industry are demanded to address the challenges. Hardware and algorithm codesigns are also needed to address or accommodate the non-idealities, e.g. noises and delays from the memristors and circuits.

4. Experimental Section

Materials and Memristor Fabrication: Pristine hBN powder and all chemicals were purchased from Sigma–Aldrich and used as received. Liquid-phase exfoliation, ink formulation, and deposition of hBN follow the method previously reported in ref. [23]. In a typical process (Figure S1, Supporting Information), the memristor was fabricated in a vertical Pt/Au/hBN/HfO_x/Ag configuration, where hBN was deposited by slot-die coating, the HfO_x layer (20 nm) was deposited by atomic layer deposition, and the metal electrodes (5/15 nm Pt/Au and 30 nm Ag) were patterned by photolithography and deposited by electron beam evaporation. During device fabrication, the hBN layer after deposition was baked at 200 °C for 2 h.

Stochastic Number Encoders and Probabilistic Logic Circuits: To build the SNEs and probabilistic logic circuits, the memristors were tested on a probe station and connected to the logic gates and other electronic devices on a breadboard. Tektronix Keithley 4200A-SCS parameter analyzer with pulse measure units was used to measure the electrical characteristics of the memristors. Siglent arbitrary waveform generators and digital storage oscilloscopes were used to output signals and measure waveforms. To endow the stochastic numbers with a certain probability, based on the $P_{uncorrelated}$-V_{in} and $P_{correlated}$-V_{in} relations in Figure 3b,c, each stochastic number encoder was fed with *n*-cycle pulsed signals of the corresponding V_{in} to encode *n*-bit stochastic numbers.

Stochastic Number Correlation: The correlation between the stochastic numbers S_x and S_y is quantified using the *Pearson correlation* (ρ) and *SC correlation* (*SCC*),[71] following:

$$\rho\left(S_x, S_y\right) = \frac{ad - bc}{\sqrt{(a+b)\,(a+c)\,(b+d)\,(c+d)}} \tag{6}$$

and

$$SCC\left(S_x, S_y\right) = \begin{cases} \frac{ad-bc}{(a+b+c+d)\,\min(a+b,a+c)-(a+b)(a+c)}, & ad \geq bc \\ \frac{ad-bc}{(a+b)(a+c)-(a+b+c+d)\,\max(a-d,0)}, & ad < bc \end{cases} \tag{7}$$

where *a*, *b*, *c*, and *d* represent the counts of 1-1, 1-0, 0–1, and 0-0 pairs of two stochastic numbers S_x and S_y, respectively.

Bayesian Inference and Fusion: Each operator is built with two randomly selected memristor-based SNEs. The two SNEs exhibit similar $P_{uncorrelated}$-V_{in} and $P_{correlated}$-V_{in} relations. No specific or additional memristor calibration, circuit redesigns, or testing of the relations are required to implement the SNEs and operators given the high device uniformity and fabrication yield of the memristors. For Bayesian fusion, the pre-trained RGB-only and thermal-only networks used are the *Ultralytics YOLOv8*[42] and *Roboflow flir-data-set/22*,[38] respectively. The prior of the obstacle $p(y)$ is assumed to be uniform for the convenience of circuit designs. Limited by the scalability of the lab-based realization of the hardware Bayesian operator, the hardware Bayesian fusion is conducted on individual RGB-thermal image pairs only. These Bayesian demonstrations are implemented by hardware. On this basis, large scale Bayesian computing is implemented via simulation in Python. Movie S1 (Supporting Information) shows large-scale Bayesian fusion on videos. Considering the output probability may exceed one, a normalization module is integrated to ensure reasonable outputs as the final multimodal fusion decisions (Figure S10, Supporting Information).

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

G.H. acknowledges support from CUHK (4055227) and RGC (24200521), Y.L. from SHIAE (RNE-p3-21), J.P. and Y.W. from RGC (24200521), T.M. from RGC (15306824) and ITC (ITS/150/23FP), L.N. from Tier 1 MOE Grant (RG86/23 and RS14/23).

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Keywords

bayes theorem, Bayesian decision-making, memristors, probabilistic Boolean logic circuits, switching stochasticity

Received: April 30, 2025
Revised: July 23, 2025
Published online: September 1, 2025

[1] D. C. Knill, A. Pouget, *Trends Neurosci.* **2004**, *27*, 712.
[2] W. Zhou, X. Lin, J. Lei, L. Yu, J. N. Hwang, *IEEE Trans. Mult.* **2022**, *24*, 2526.
[3] Y. Sun, W. Zuo, P. Yun, H. Wang, M. Liu, *IEEE Trans. Auto. Sci. Engineer.* **2021**, *18*, 1000.
[4] D. Bonnet, T. Hirtzlin, A. Majumdar, T. Dalgaty, E. Esmanhotto, V. Meli, N. Castellani, S. Martin, J.-F. Nodin, G. Bourgeois, J.-M. Portal, D. Querlioz, E. Vianello, *Nat. Commun.* **2023**, *14*, 7530.
[5] A. Alaghi, W. Qian, J. P. Hayes, *IEEE Trans. Comp.-Aided Des. Int. Circ. Syst.* **2018**, *37*, 1515.
[6] R. K. Budhwani, R. Ragavan, O. Sentieys, *Proc. – IEEE Int. Symp. Circuits and Systems*, IEEE, Baltimore, MD, USA **2017**, 7–10.
[7] A. Alaghi, J. P. Hayes, *ACM Trans. Embedded Comp. Syst.* **2013**, *12*, 1.
[8] P. Knag, W. Lu, Z. Zhang, *IEEE Trans. Nanotechnol.* **2014**, *13*, 283.
[9] S. Gaba, P. Sheridan, J. Zhou, S. Choi, W. Lu, *Nanoscale* **2013**, *5*, 5872.
[10] S. A. Salehi, *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2020**, *28*, 992.
[11] A. Alaghi, J. P. Hayes, *2013 IEEE 31st Int. Conf. Comp. Design, ICCD*, IEEE, Asheville, NC, USA **2013**, 39–46.
[12] H. Ichihara, S. Ishii, D. Sunamori, T. Iwagaki, T. Inoue, *2014 32nd IEEE International Conference on Computer Design, ICCD*, IEEE, Seoul, Korea (South) **2014**, 361.
[13] H. Ravichandran, T. Knobloch, S. Subbulakshmi Radhakrishnan, C. Wilhelmer, S. P. Stepanoff, B. Stampfer, S. Ghosh, A. Oberoi, D. Waldhoer, C. Chen, J. M. Redwing, D. E. Wolfe, T. Grasser, S. Das, *Nat. Commun.* **2024**, *15*, 1.

[14] H. Ravichandran, Y. Zheng, T. F. Schranghamer, N. Trainor, J. M. Redwing, S. Das, *Adv. Mater.* **2023**, *35*, 1.
[15] A. Sebastian, R. Pendurthi, A. Kozhakhmetov, N. Trainor, J. A. Robinson, J. M. Redwing, S. Das, *Nat. Commun.* **2022**, *13*, 1.
[16] Y. Zheng, H. Ravichandran, T. F. Schranghamer, N. Trainor, J. M. Redwing, S. Das, *Nat. Commun.* **2022**, *13*, 1.
[17] L. O. Chua, *Nat. Rev. Electr. Engineer.* **2024**, *1*, 614.
[18] S. Kumar, X. Wang, J. P. Strachan, Y. Yang, W. D. Lu, *Nat. Rev. Mater.* **2022**, *7*, 575.
[19] J. Tang, F. Yuan, X. Shen, Z. Wang, M. Rao, Y. He, Y. Sun, X. Li, W. Zhang, Y. Li, B. Gao, H. Qian, G. Bi, S. Song, J. J. Yang, H. Wu, *Adv. Mater.* **2019**, *31*, 1902761.
[20] S. Dutta, G. Detorakis, A. Khanna, B. Grisafe, E. Neftci, S. Datta, *Nat. Commun.* **2022**, *13*, 2571.
[21] W. A. Borders, A. Z. Pervaiz, S. Fukami, K. Y. Camsari, H. Ohno, S. Datta, *Nature* **2019**, *573*, 390.
[22] Q. Xia, J. J. Yang, *Nat. Mater.* **2019**, *18*, 309.
[23] L. Song, P. Liu, J. Pei, F. Bai, Y. Liu, S. Liu, Y. Wen, L. W. T. Ng, K. Pun, S. Gao, M. Q.-H. Meng, T. Hasan, G. Hu, *Adv. Electron. Mater.* **2024**, *10*, 1.
[24] L. Song, P. Liu, J. Pei, Y. Liu, S. Liu, S. Wang, L. W. T. Ng, T. Hasan, K. P. Pun, S. Gao, G. Hu, *Nat. Commun.* **2025**, *16*, 1.
[25] W. Chen, L. Song, S. Wang, Z. Zhang, G. Wang, G. Hu, S. Gao, *Adv. Electron. Mater.* **2023**, *9*, 2200833.
[26] S. Wang, L. Song, W. Chen, G. Wang, E. Hao, C. Li, Y. Hu, Y. Pan, A. Nathan, G. Hu, S. Gao, *Adv. Electron. Mater.* **2023**, *9*, 2200877.
[27] Z. Wang, S. Joshi, S. E. Savel'ev, H. Jiang, R. Midya, P. Lin, M. Hu, N. Ge, J. P. Strachan, Z. Li, Q. Wu, M. Barnell, G. L. Li, H. L. Xin, R. S. Williams, Q. Xia, J. J. Yang, *Nat. Mater.* **2017**, *16*, 101.
[28] J. Song, J. Woo, A. Prakash, D. Lee, H. Hwang, *IEEE Electron Device Lett.* **2015**, *36*, 681.
[29] C. P. Hsiung, H. W. Liao, J. Y. Gan, T. B. Wu, J. C. Hwang, F. Chen, M. J. Tsai, *ACS Nano* **2010**, *4*, 5414.
[30] U. Noppeney, *Annu. Rev. Neurosci.* **2021**, *44*, 449.
[31] T. Rohe, A. C. Ehlis, U. Noppeney, *Nat. Commun.* **2019**, *10*, 1.
[32] Y. Lecun, Y. Bengio, G. Hinton, *Nature* **2015**, *521*, 436.
[33] R. van de Schoot, S. Depaoli, R. King, B. Kramer, K. Märtens, M. G. Tadesse, M. Vannucci, A. Gelman, D. Veen, J. Willemsen, C. Yau, *Nat. Rev. Methods Primers* **2021**, *1*, 1.
[34] T. H. Chen, J. P. Hayes, *Proc. IEEE Comp. Soc. Annual Symp. VLSI, ISVLSI*, IEEE, Pittsburgh, PA, USA **2016**, 116–121.
[35] K. Wang, Y. Wang, B. Liu, J. Chen, *IEEE Trans. Instrum. Meas.* **2023**, *72*, 1.
[36] M. Green, *Transp. Human Factors* **2000**, *2*, 195.
[37] D. Gehrig, D. Scaramuzza, *Nature* **2024**, *629*, 1034.
[38] T. Imaging, *FLIR data set Dataset* **2024**, Roboflow Universe.
[39] D. J. Yeong, G. Velasco-hernandez, J. Barry, J. Walsh, *Sensors* **2021**, *21*, 1.
[40] H. Wang, C. Liu, Y. Cai, L. Chen, Y. Li, *IEEE Trans. Instrum. Meas.* **2024**, *73*, 1.
[41] Y.-T. Chen, J. Shi, Z. Ye, C. Mertz, D. Ramanan, S. Kong, *Computer Vision – ECCV 2022*, Springer, Cham Switzerland **2022**, pp. 139–158.
[42] G. Jocher, J. Qiu, A. Chaurasia, *Ultralytics YOLO*, **2023**.