A cognitive stochastic machine based on Bayesian inference: a behavioral analysis

Raphael Frisch, Marvin Faix
and Emmanuel Mazer
LIG - University Grenoble Alpes, France
raphael.frisch@univ-grenoble-alpes.fr
marvin.faix@inria.fr
emmanuel.mazer@inria.fr

Laurent Fesquet
TIMA, Grenoble, France
laurent.fesquet@univ-grenoble-alpes.fr

Augustin Lux
INRIA Grenoble Alpes, France
augustin.lux@inria.fr

Abstract—Bayesian models and stochastic computing form a promising paradigm for non-conventional, bio-inspired computation architectures. In particular, they are able to handle uncertainty and promise low power consumption. In this paper we study the application of such an architecture, the Sliced Bayesian Machine (SlicedBM) to a real-world problem, Sound Source Localization (SSL) for robots. We present an analysis of the quality of results and of computing time according to several parameters: sensor precision, result threshold, internal word length. Furthermore, we show that sensor data precision does not heavily influence the computation. On the opposite, the precision of the probability values plays an important role on result quality. This parameter also determines the circuit size. We also show that the higher the re-sampling threshold (RT), the better the distribution computed by the machine. Our results make it possible to choose optimal design parameters for a circuit along several trade-offs, and according to a given sensor fusion application.

I. Introduction

Are modern computers adapted to deal with sensor uncertainty, fault tolerance and low power requirements? As traditional computing approaches are reaching physical limits, Moore's law becomes difficult to follow [16]. Exploring alternative computation concepts nowadays becomes crucial to find solutions for the ever growing need of computation capacity. By nature, human beings use very energy efficient processing approaches to cope with all these objectives. Bayesian models may be a key to the problem since they take into account the incompleteness of their perceived environment and easily overcome errors in the sensor data. Due to their strengths, they are broadly used in many applications such as behavior prediction or decision making [27]. Probability may be a powerful tool to replace classical logic which is used in conventional computation architectures. It introduces a more plausible approach to address tasks such as robotics or artificial intelligence [13].

When looking at the human and their decision making process with all the neurons and the synapses, stochastic computing seems perfectly adapted to imitate the information traveling in such a network [8]. The information exchange between neurons can be represented by the binary telegraphic signal of a stochastic architecture. Performing Bayesian inference with stochastic computing leads to temporal encoding of the probabilities as stochastic bit streams. This results in an architecture without classical floating point units and only with simple electronic components, such as AND gates and counters [8].

In this paper, we focus on Sound Source Localization (SSL) for robots, such as humanoid, companion or telepresence robots. The goal is to calculate the azimuth of the sound source, typically a human speaker. Using this angular information, the robot can turn towards the person and interact with it. Since the robot is running on its own power supply, we intended to design a lightweight computation architecture to localize the speaker. We choose the SSL application since it is a non trivial problem to solve. Indeed, a Bayesian model was created containing many variables which will serve a representative benchmark.

The basic concept of Bayesian machines was presented in [5] and studies showed the robustness of the machine [4]. After dealing with simple examples of application, the Bayesian machine was adapted for SSL was presented in [10]. This introduced a new architecture, which is an extension of the standard version of the BM (Bayesian Machine), named the Sliced-BM. However, the development of the Sliced-BM also introduced many new parameters that were never studied before. The goal of this study is to analyze the impact of the different parameters of the Sliced-BM. Since our long term vision is to design the circuit corresponding to the BM, it is crucial to fully understand its behavior to be able to adapt a BM circuit to future applications, and to build optimal hardware for each use.

A. Related work

Probabilistic theory has gained interest in both the software and hardware community. Many probabilistic languages have been developed to learn Bayesian models and do inference on them. Furthermore, the interpretor of such a language enables the learning and the inference. Some examples of such developed languages are ProBT [1], Church [12], Venture [20], Figaro [24], Blog [22].

On the hardware side, the following results have been presented. Based on analog signals, Vigoda performs Bayesian inference using a message passing algorithm [28]. Moreover,

using optical hardware, [2] take advantage of the light intensity at different wave lengths to represent all the values defining a probability distribution simultaneously which allows to multiplex the processing on the same hardware. Furthermore, [15] encoded probability values using Strain switched Magneto Tunneling Junction (SMTJ) devices. A specific magneto-electric adder and multiplier was designed to manipulate this specific representation of probabilities. Based on these two operation, a inference circuit can easily be designed by organizing these blocks. Finally, Friedman et al. took C-Elements (also known as Muller gate) to achieve naive Bayes fusion for binary random variables by mixing stochastic signals [9].

When developing probabilistic hardware, many different aspects need to be considered. For example, probabilistic hardware often relies on entropy generators. Moreover, random number generators heavily influence the performance of the hardware. Therefore, physically based mechanism are explored to overcome this need. Different approaches exist: the more experimental MTJ [23] or the well-studied STRNG [3].

Working on stochastic hardware, a stochastic adder has been presented in [17] which gives new opportunities to develop stochastic hardware for Bayesian inference. The robustness of Bayesian machines has been demonstrated in [4] by fault injections. Results of fault injection at the RTL level provide the first evidences of the intrinsic robustness of such architectures.

Considering the task of sound source localization (SSL) for companion or telepresence robots, Keyrouz et al. developed a technique to perform localization using human inspired binaural techniques [14]. Sound source localization has also been studied using bio-inspired architectures. Using spiking neural networks, neuromorphic chips have been developed to imitate the locating technique of sound used by barn owls [25].

B. Paper outline

This paper is structured as follows. Section II provides a fast introduction of the Sound Source Localization (SSL) problem and introduces the stochastic approach and the resulting Bayesian Machine (BM) architecture. Section III explains the problems encountered when having too many evidences in a BM and introduces the enhanced architecture which is used for SSL More precisely, the temporal dilution problem is stated and the proposed solution using max-normalization to regenerate the stochastic signal is described. Experimental results are presented in section IV. The angular speaker localization is shown as well of a deep analysis of the impact of the different parameters of the Sliced-BM. Finally, some conclusion is drawn from this work and the ongoing work is stated.

II. BACKGROUND

A. Sound source localization

The task of Sound Source Localization (SSL) aims to estimate the position of a sound source in a room using the signals recorded by several microphones placed in the area. This task has been massively studied in the past. However, the interest in probabilistic models for SSL raised recently.
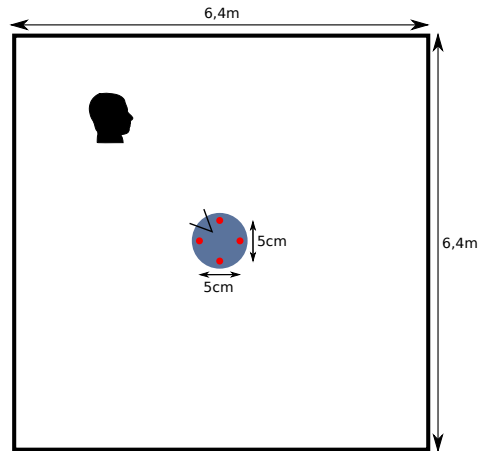

Fig. 1. Schema of the proposed sound source localization setup

These models associate the source position to acoustic inter-microphone features, e.g. [19], [26], [21], [29], [6], [7], [18].

In this study, we look at the application of SSL for telepresence or companion robots, e.g. [14]. The goal is to estimate the direction of the sound source to direct the robot to the source.

Fig. 1 presents the setup of the current study. A typical situation is replicated where the robot is placed in the middle of a room. A person speaks in this room of dimensions 6.4 m × 6.4 m × 3.1 m. The robot has 2 pairs of microphones mounted on its top with each a inter-microphone distance of 5cm (red dots on Fig. 1). As acoustic model for this work, the free-field model is assumed which expects the microphones to be omni-directional and to "float" in the room (their mounting does not affect the propagation of the sound waves). Moreover, this model assumes no reverberations on walls since they are considered negligible. However, experiments were run in real conditions with a realistic reverberation time computed by the room impulse response (RIR) simulator which is widely used in the research community of SSL.

The Bayesian Machine (BM) fuses the angular information given by each microphone pair. Among the various features used in SSL, we focus on the delay analysis between the recorded signals of each microphone pair. Namely, we use the Inter-channel Phase Difference (IPD). Considering one pair of microphones m, let $y_{1,m}(t)$ and $y_{2,m}(t)$ be the recorded signals from the two mikes. Let $Y_{2,m}(f,l)$ and $Y_{1,m}(f,l)$ be the Fourier transform calculated with the Short-Time Fourier Transform (STFT) with f the frequency and l the time frame. To calculate the IPD, we need the inter-channel coefficient ratio $R_m(f,l)$ which is defined by:

$$R_m(f,l) = \frac{Y_{2,m}(f,l)}{Y_{1,m}(f,l)} \qquad (1)$$

With (1), we can compute the desired angular information:

$$\phi_m(f,l) = \arg R_m(f,l) \qquad (2)$$

Since in our acoustic model we assume the free-field model, we can calculate the theoretical IPD $\phi_m^{\text{th}}(f,l)$ for each position
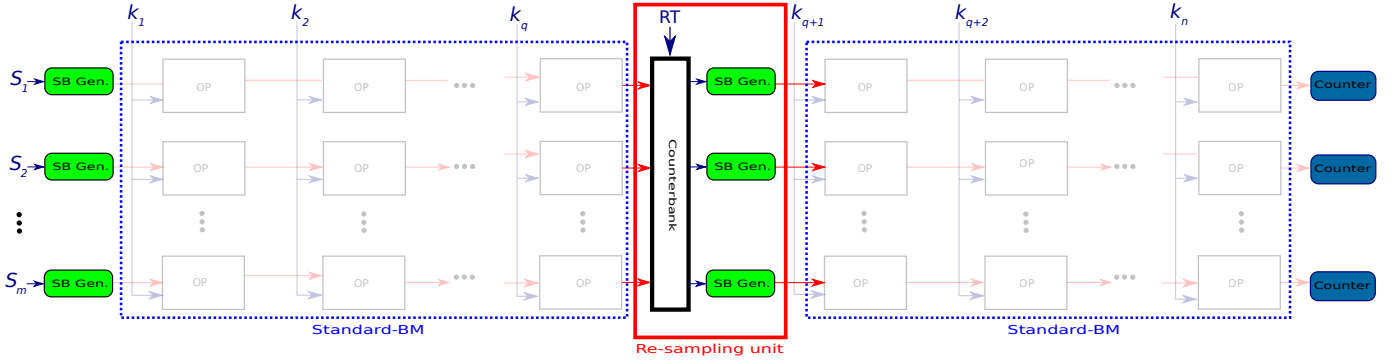
Fig. 2. Architecture of a Sliced-BM with 2 slices of $q = n/2$ columns each and 1 re-sampling unit between them. Originally published in [10]
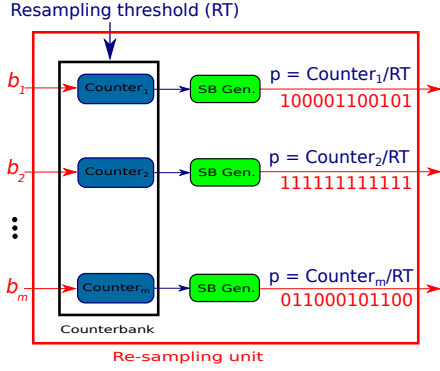

Fig. 3. The re-sampling unit of the Sliced-BM. Originally published in [10]

of the room very easily. In our probabilistic model, we assume the $\phi_m^{\text{meas}}(f,l)$ to follow a Gaussian distribution with a mean equal to $\phi_m^{\text{th}}(f,l)$ and a variance of σ_ϕ^2:

$$\phi_m^{\text{meas}}(f,l) \sim \mathcal{N}(\phi_m^{\text{th}}(f,l), \sigma_\phi^2). \qquad (3)$$

Moreover, the distribution conditioned on the source position is defined as:

$$p(\phi_m^{\text{meas}}(f,l)|x,y) = \frac{1}{\sqrt{2\pi}\sigma_\phi} \exp\left(-\frac{(\phi_m^{\text{meas}}(f,l)-\phi_m^{\text{th}}(f,l))^2}{2\sigma_\phi^2}\right) \qquad (4)$$

B. Bayesian machines

This section quickly introduces the concept of Bayesian machines (BM). The BMs were originally introduced in [5]. Later, in [10], the BM was adapted for Sound Source Localization (SSL) purposes. For more details, please refer to the mentioned articles.

Given a probabilistic model for the application, the goal of the BM is to compute the posterior probability distribution of this model provided the current sensor data. Lets S be the discrete searched variable of the model and K a discrete known variable. Notice that S and K can be themselves conjunction of variables. Following the Bayes rule, the inference over n known variables is done with:

$$P(S|K_1,\ldots,K_n) = \frac{1}{Z} P(S) \prod_{i=0}^{n-1} P(K_i|K_1,\ldots,K_{i-1},S) \qquad (5)$$

with $P(S)$ the prior, $P(K_i|K_1,\ldots,K_{i-1},S)$ the conditional distributions and Z the normalization constant.

Moreover, in our machines we mainly use the naive Bayesian fusion. In this case, each conditional distribution is treated as a likelihood of independent sensor variables (a so-called evidence) and (5) simplifies to:

$$P(S|K_1,\ldots,K_n) = \frac{1}{Z} P(S) \prod_{i=0}^{n-1} P(K_i|S). \qquad (6)$$

To compute the probability distributions as in (6), many multiplications are required. Therefore, the BM takes advantage of the stochastic bit streams. Contrary to conventional architectures, all probability values are encoded by streams of stochastic bits [11], drawn from a Bernoulli distribution. Each sample '0' or '1' represents $p = P(X = x_i)$. Discrete temporal integration over n_T steps gives an approximation of p: this is done by counting the number n_1 of '1' and dividing by n_T:

$$\frac{n_1}{n_T} \xrightarrow[n_T \to \infty]{} p. \qquad (7)$$

One main advantage of stochastic computing is that a multiplication between 2 probability values is supplied by an AND gate: given two probabilities p_1 and p_2 respectively encoded by their bit stream chain B_1 and B_2, the chain B_3 resulting from applying an AND gate over B_1 and B_2 encodes the probability $p_3 = p_1 \times p_2$. Hence, the computation of (6) is implemented by a succession of AND gates. The BM samples the desired distribution by computing in parallel its value for each possible value of the search variable in (6). The BM is organized as a matrix where each line represents the computation of (6) for each value $S_1, S_2, ..., S_m$ of the search variable S. At the end of each line, the result of the stochastic bit stream that went through the n successive AND gates is stored in counters. The probability distribution can be retrieved via temporal integration. Moreover, the final distribution is obtained after normalizing all the counter values with $\frac{counter_j}{\sum_l counter_l}$. This provides an approximation of the searched distribution $P(S = s_j|k_1,\ldots,k_n)$. The approximation is due to the stochastic approach in the machine.

III. Sliced-BM for SSL

Using stochastic bit streams as data representation introduced many advantages. However, an important drawback appears when dealing with low probabilities. As the BM is computing products of probabilities (encoded as stochastic bit streams), after a certain number of multiplications the resulting product has a very small value. The corresponding bit streams are mostly composed of '0's, they have low entropy, and the time to obtain significant output information grows exponentially. This problem is called the temporal dilution. To speed up the time needed to compute the temporal dilution, a more enhanced version of the BM was introduced in [10], named the Sliced-BM.

The concept of the Sliced-BM is based on regenerating the stochastic signal. A max-normalization is performed after a certain number of columns to regenerate the stochastic signal. To this purpose, the total amount of columns is divided into slices which typically contain 10 columns in our examples. At the end of each slice, counters capture the state of the probability distribution computed so far, and a max-normalization is done to start the next slice with a much higher amount of '1's in the stochastic signal.

The Sliced-BM architecture is shown in Fig. 2. A machine with m lines and n columns is shown with a total of 2 slices. The re-sampling unit between both slices regenerates the stochastic signal. At the end of the last slice, counters store the final probability distribution.

The max-normalization at the end of each slice is performed when the counter of one line in the slice reaches the re-sampling threshold (RT). After the max-normalization, this line starts the next slice with '1's as illustrated in Fig. 3 for line 2 with the bit stream b_2. Mathematically, all lines are divided by the RT and start the next slice with a prior equal to $Counter_i/RT$.

Concretely, when considering the Sliced-BM architecture, the main product in (6) is divided into a sequence of products each one of which represents the multiplication performed by a slice. This amounts to rewrite (6) as:

$$P(S|K_1,\ldots,K_n) = \frac{1}{Z'}P(S)\prod_{c=0}^{r-1}M_c\prod_{i=c\times q}^{(c+1)\times q-1}P(K_i|S). \tag{8}$$

where r is the number of slices with $q = n/r$ columns each. M_c represents the normalization factor of each slice.

For the SSL task, we take the IPD from both microphone pairs and fuse their information. The IPD is computed for each frequency bin. Since we mainly localize human speakers, we use the frequencies between 200Hz and 1000Hz to localize the source which are represented by 50 frequency bins. Hence, we have 50 IPDs per microphone pair. In total, the BM has $n = 100$ columns which were divided into $r = 10$ slices of $q = 10$ columns each. Each column contains the sensor data of the sensor K_i (known variable defined in (8)) which is the IPD for a certain frequency of a specific microphone pair. The lines of the BM represent the different possible positions in
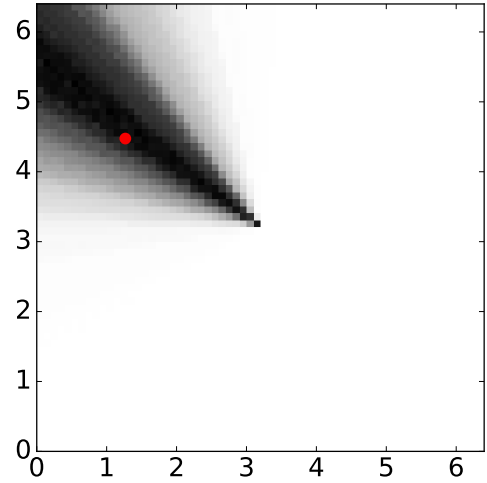

Fig. 4. Probability distribution of the sound source localization computed by the Sliced-BM

the room. Each line corresponds to a cell located at (x, y) which was defined as the searched variable S in (8).

IV. Experiments

This section presents the experiments that were made to analyze the behavior of the developed system. The localization performance of the system is shown as well as the impact of the different parameters that the new architecture Sliced-BM introduces. These parameters are:

- the number of bits used to represent probability values
- the number of bits for sensor data
- the number of slices
- the type of random number generator for bit stream creation

This section is structured as follows. First, the experimental setup is described. Second, an example of the localization is presented. Third, the definition of the Kullback-Leibler divergence is given. Fourth, the impact of the discretization of the probability values is observed. Fifth, the influence on the inference by the sensor precision is analyzed. Sixth, the importance of a good random number generator in the complete system is shown. Seventh, the effect on the computation when changing the number of slices is evaluated.

A. Experimental setup

In this study, the Sound Source Localization (SSL) task has been adapted to a companion or telepresence robot application. Therefore, the setup aims to reproduce a robot in a room which has to localize the sound source speaking from somewhere in that room. The setup, shown in Fig. 1, simulates a robot placed in the middle of a squared room of dimensions 6.4 m × 6.4 m × 3.1 m. The robot has been equipped with 2 pairs of microphones where the mikes of each pair are separated by 5 cm. The mikes are represented by the red dots in Fig. 1.

The goal is to estimate the azimuth of the sound source by the robot using the mikes and making the inference with the
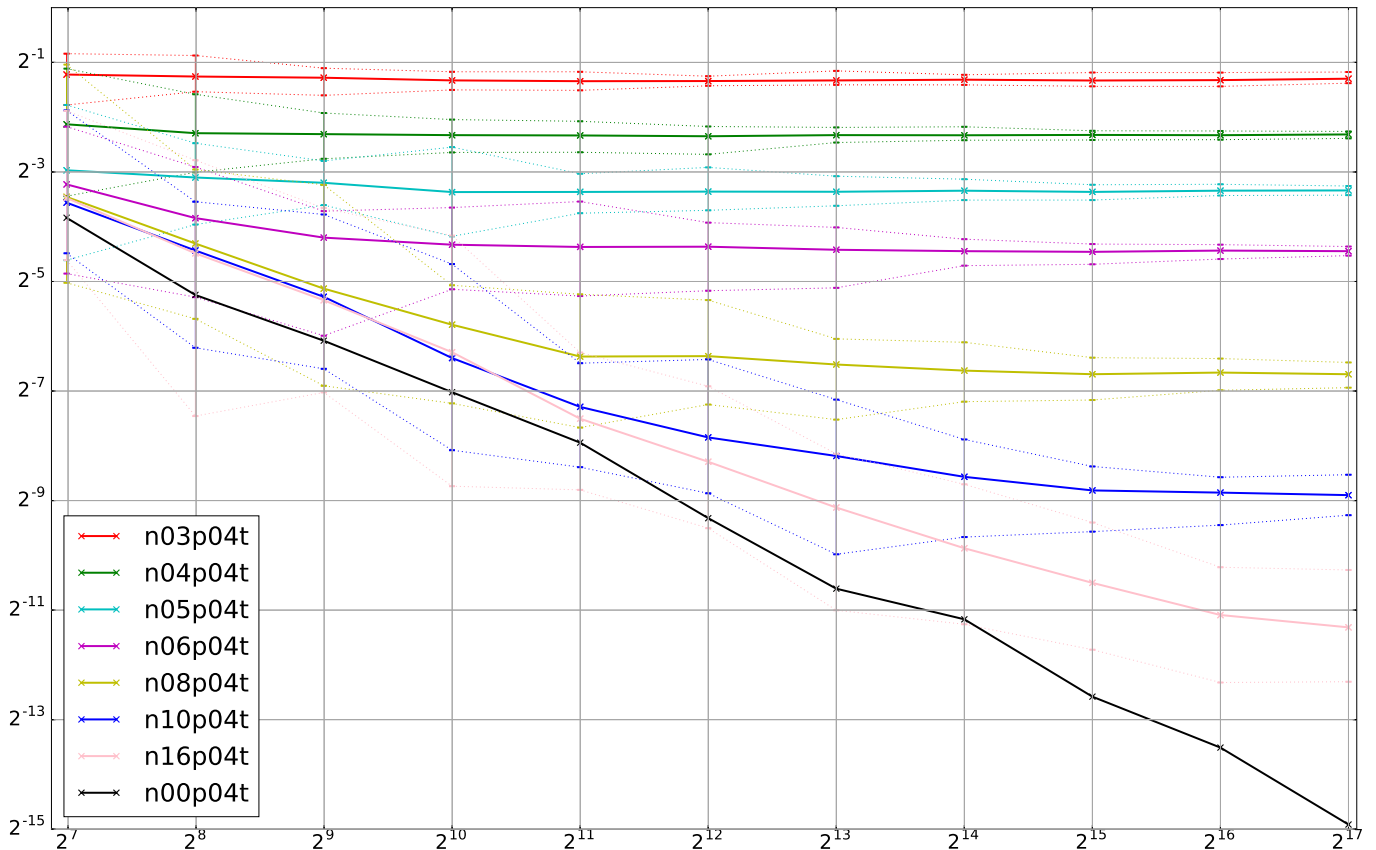
Fig. 5. KLD as a function of the re-sampling threshold value for different *n_bit*

BM. Therefore, the room was discretized into small cells, each $10 \text{ cm} \times 10 \text{ cm}$ insize, composing a grid of $64 \times 64 = 4,096$ possible positions.

The Sliced-BM has been implemented in C++ and Python to simulate the high-level behavior of the connected electronic components, which is typically done using a FPGA simulation. Using this simulator, fast experiments can be run. The mono-channel 16-kHz speech signals of the source recorded by each microphone have been generated using the room impulse response (RIR) simulator of AudioLabs Erlangen [1]. The reverberation time has been set to 400ms, an average value for such a room size. The IPDs have been calculated using the Short-Time Fourier Transform (STFT) with N = 1,024.

B. Localization performance

Regarding the application, the localization of the sound source has a big impact on the behavior of the robot. The BM computes the probability distribution from which the azimuth is easily derived to move the robot to the right position. In this experiment, the sound source was place at $X = 1.2m$ and $Y = 4.4m$.

The resulting map displaying the probability distribution of $P(x, y|K_i)$ is provided in Fig. 4. The shown distribution is computed using data from a single time frame (32ms) of the STFT and the described setup. It can be used to derive the speakers azimuth. The red dot gives the ground truth for the speakers position.

One can see that the angular localization is very precise. This provided information is good enough to control the companion robot of to move the camera of the telepresence robot to the speaker.

C. Kullback-Leibler divergence (KLD)

In the following, the quality of the computed distributions have to be evaluated. We use the well-known Kullback-Leibler divergence (KLD) to compare two probability distributions. More precisely, we compute the KLD between the experimental result distribution P_{exp} and and the theoretical distribution P_{th} as:

$$D_{KL}(P_{th}, P_{exp}) = \sum_i P_{th}(i) \log \frac{P_{th}(i)}{P_{exp}(i)}. \quad (9)$$

The value of the KLD is positive, a value of 0 means the two distributions are identical. The KLD is not a mathematical distance, in our experiments we use it as a quality measure. It is displayed on the Y-axis of the plots presented in the following sections. It compares the reference distribution obtained by multiplying all the evidences encoded as double floats P_{th} and the distribution computed by the BM P_{exp} in the given setup [2].

D. Impact of the probability discretization

This section will analyze how the machine behaves when varying the number of bits used to represent the probability values, namely *n_bit*. Moreover, the impact on the quality of the probability distribution is measured using the KLD. Notice, that the *n_bit* parameter directly impacts the size of the corresponding circuit of the machine and hence its power consumption. It also defines the width of the memory blocks of the BM since the evidences $P(K_i|S)$ are stored in registers in each OP block of the machine. Furthermore, we analyze how the KLD evolves for different values for the re-sampling threshold (RT). Since we target a low-power architecture, it is crucial to find the optimal trade-off between the resulting precision and the circuit size to minimize consumption.

Fig. 5 shows the results of the experiments. The plot shows the KLD value (Y-axis) as a function of the re-sampling threshold value (X-axis) for the different values of *n_bit* (different lines). The sensor precision was set to 4 bits in all cases (*p04*) and a linear feedback shift register (LFSR) was used as a random number generator (RNG) which leads to the extension of the name *p04t* ('t' for use_LFSR = True). For each value of *n_bit* (*n03* to *n16*) 20 runs of the same experiment were made to calculate a mean value which is given by the thick line. The minimal and maximal KLD of each the 20 runs is provided by the dashed lines respectively underneath and above the line of the mean value (thick line). The black line (n00p04t) represents the float computation of the probability distribution as a reference of a maximal precision. Notice the logarithmic scale in both dimensions for a better presentation of the obtained data. The experiments were made for re-sampling thresholds varying from $2^7 = 128$ up to $2^{17} = 131.072$. The lower the KLD value, the better probability distribution.

Looking at the mean lines (thick lines), one can clearly observe a convergence to a maximal precision which depends on the value of *n_bit*. The precision of the distribution improves with *n_bit*. Moreover, in all cases, the increment of the re-sampling threshold (RT) leads to a better KLD. This is due to the fact that with a higher RT, the discretization of the probability value increases and hence distribution can be represented with more detail. However, an high RT means a long execution time for the inference since each slice of the machine will take longer to reach its RT. Therefore, it is crucial to adapt the RT to the given *n_bit* to avoid useless computation and power consumption.

E. Impact of the sensor precision

This section analyzes the impact on the computation when the sensor precision changes. Depending on the application that is treated by the BM, sensors can have different precisions. Therefore, it is important to analyze the overall result for different discretization of the sensor data.

Fig. 6 provides the results for different sensor precisions *p04* and *p06* which correspond respectively to 4 and 6 bits of precision for the sensor data. As the plot shows, the sensor precision does not massively impact the computation as the
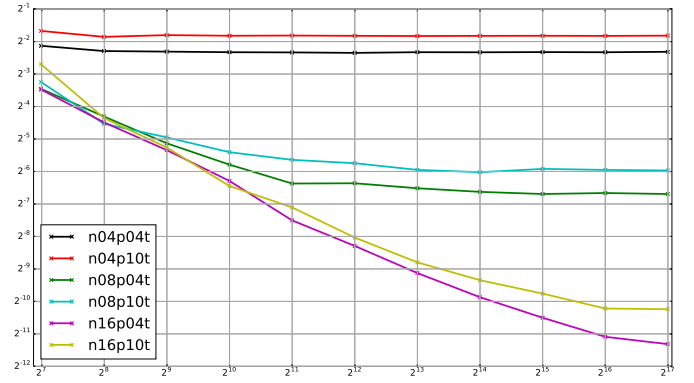

Fig. 6. KLD as a function of the re-sampling threshold value for different discretization of the sensor data

KLD value for each value of the RT does not vary much between *p04* and *p10* for a given value of *n_bit* (remember that both axes are in a logarithmic scale). Notice that for the clarity of the plot, we removed the plots for *p06* and *p08* since they were partially or completely overlapping with the lines for *p10*. As a conclusion, one can say that the sensor precision does not have a strong influence on the computation result whereas the choice of the *n_bit* heavily impacts the final result. This is not as astonishing as it might seem at first sight: given the large number of sensors in our application (the BM1 has a total of 100 columns), the exact product in (6) has a precision of 400 bits for 4-bit sensors, much more than can effectively be computed. Our computation is necessarily affected by rounding, and going to higher sensor precision cannot improve this situation.

F. Impact of the LFSR

Since in the current implementation of the circuit, linear feedback shift registers (LFSR) are used as random number generators (RBG) to obtain the stochastic bit streams. Therefore, we want to study the impact of the LFSR on the computed probability distribution. A succession of runs were made for fixed (*n02p08*, *n06p08* and *n08p08*) parameters of *n_bit* and sensor precision *p*. The computation is compared when using the LFSR as random number generators (RBG) in the BM (i.e. lines with *t* at the end) as opposed to when the BM uses the RBG included in C++ (i.e. lines with *f* at the end). Like in the previous sections, 20 runs were made for each value of RT (X-axis) and the mean value of all runs is plotted. Fig. 7 shows the KLD value as a function of the different values for the re-sampling threshold (RT). The precision of the computed distribution improves with increasing RT. However, one can see a notable difference in the KLD between the purple line (*n08p08f*) and the yellow line (*n08p08t*). At the RT value 2^{16}, a factor 16 between both lines can be observed which is clearly not negligible.

Therefore, as future work, we have to explore other and better ways to generate random bits while keeping the power consumption low. RNGs can sometimes require a large area on the circuit. For this reason, trying new methods based on
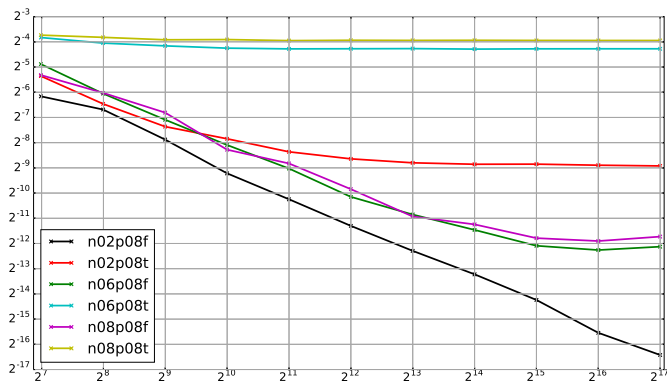
Fig. 7. KLD as a function of the re-sampling threshold value for runs with LFSR as random number generators (t) and with the random number generator of C++ (f)
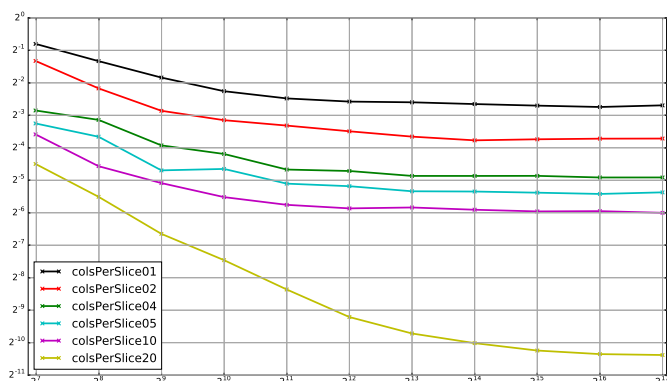

Fig. 8. KLD as a function of the re-sampling threshold value for different number of columns per slice

physical phenomenons for RBGs might be an important task in future for us.

G. Computation accuracy depending on the number of slices

One major parameter which was introduced with the Sliced-BM architecture is the number of slices contained in the machine. The number of slices directly defines the number of columns in each slice which determines the circuit size of the machine. Moreover, in terms of computation speed, the number of slices is a key parameter since a low number of slices means that re-sampling is performed only a few times and hence the risk of loosing time due to temporal dilution grows. On the opposite, when having many slices in the machine, the inference will take more time because the re-sampling will occupy a large part of the total computation time. The goal is to find a trade-off between the minimal circuit size and a reasonable computation time.

Fig. 8 shows the evolution of the precision for different numbers of columns per slice in the machine considering the machine has 100 columns in total. The graph shows that the quality of the results improves with the number of columns per slice. However, if the slices become too large, the computation time of the machine can significantly increase due to temporal dilution. Hence, the need to find a reasonable trade-off.

V. CONCLUSION & FUTURE WORK

In this work a method for SSL for companion or telepresence robots has been presented. The SSL computes the probability distribution and the angular information of the speaker located somewhere around the robot. Furthermore, the heart of the computation architecture has been deeply analyzed to observe the impact of the modification of the different parameters that were introduced by this new architecture, namely the Sliced-BM. In this enhanced architecture the slicing concept has been defined. Experiments have been made to study the behavior of the architecture in terms of computation speed, circuit size and computation accuracy. It has been shown that the discretization of sensor data does not have a big influence on the computation. However, the discretization of the probability values in the machine has an important impact on the quality of results. This parameter directly defines the circuit size. Moreover, it has been shown that the higher the re-sampling threshold (RT), the better the distribution computed by the machine. Furthermore, the impact of the quality of the random number generator (RBG) has been illustrated. Lastly, the evolution of the accuracy of the computation when the number of slices changes has been observed. Overall, one can say that depending on an application and on the used sensors, we can determine the discretization of the probability values in the machine and the optimal RT for this situation. This determination is made as a trade-off between circuit size and runtime. These results will valuable for designing future chips dedicated to new applications.

As future work, we consider exploring other types of RBG to replace the LFSR since the impact of the RBG on the overall computation is very impressive as shown in the experiments. Furthermore, the current acoustic model may be replaced by a learned probabilistic model which can be adapted to a specific room. Moreover, we will have to measure the resource consumption of the BM in terms of memory and energy. Finally, we will adapt the Sliced-BM to other applications in order to validate the ease to adapt and find the optimal circuit to overcome another task.

REFERENCES

[1] P. Bessière, E. Mazer, J. M. Ahuactzin, and K. Mekhnacha, *Bayesian programming*. CRC Press, 2013.
[2] P. Blanche, M. Babaeian, M. Glick, J. Wissinger, R. Norwood, N. Peyghambarian, M. Neifeld, and R. Thamvichai, "Optical implementation of probabilistic graphical models," 2016.
[3] K. Cherkaoui, V. Fischer, A. Aubert, and L. Fesquet, "A very high speed true random number generator with entropy assessment," *Workshop on Cryptographic Hardware and Embedded Sys.*, pp. 179–196, 2013.
[4] A. Coelho, R. Laurent, M. Solinas, Jr., J. Fraire, E. Mazer, N. E. Zergainoh, S. Karaoui, and R. Velazco, "On the robustness of stochastic bayesian machines," *IEEE Trans. Nucl. Sci.*, vol. PP, no. 99, pp. 1–1, 2017.
[5] A. Coninx, P. Bessière, E. Mazer, J. Droulez, R. Laurent, M. A. Aslam, and J. Lobo, "Bayesian sensor fusion with fast and low power stochastic circuits," in *Proc. of IEEE Int. Conf. on Rebooting Computing*, 2016.

[6] A. Deleforge, R. Horaud, Y. Y. Schechner, and L. Girin, "Co-localization of audio sources in images using binaural features and locally-linear regression," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 23, no. 4, pp. 718–731, 2015.

[7] Y. Dorfan and S. Gannot, "Tree-based recursive expectation-maximization algorithm for localization of acoustic sources," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 23, no. 10, pp. 1692–1703, 2015.

[8] M. Faix, E. Mazer, R. Laurent, M. O. Abdallah, R. Le Hy, and J. Lobo, "Cognitive computation: a bayesian machine case study," in *IEEE Trans. Cogn. Inf. & Cogn. Comp.* IEEE, 2015, pp. 67–75.

[9] J. S. Friedman, L. E. Calvet, P. Bessière, J. Droulez, and D. Querlioz, "Bayesian inference with muller c-elements," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 63, no. 6, pp. 895–904, 2016.

[10] R. Frisch, R. Laurent, M. Faix, L. Girin, L. Fesquet, A. Lux, J. Droulez, P. Bessière, and E. Mazer, "A bayesian stochastic machine for sound source localization," in *2017 IEEE International Conference on Rebooting Computing (ICRC)*, Nov 2017, pp. 1–8.

[11] B. Gaines, "Stochastic computing systems," in *Advances in information systems science.* Springer, 1969, vol. 2, pp. 37–172.

[12] N. Goodman, V. Mansinghka, D. Roy, K. Konawitz, and D. Tarlow, "Church: a language for generative models," *arXiv preprint arXiv:1206.3255*, 2012.

[13] E. T. Jaynes, *Probability Theory: the Logic of Science.* Cambridge University Press, 2003.

[14] F. Keyrouz and K. Diepold, "Binaural source localization and spatial audio reproduction for telepresence applications," *Presence*, vol. 16, no. 5, pp. 509–522, Oct 2007.

[15] S. Khasanvis, M. Li, M. Rahman, A. K. Biswas, M. Salehi-Fashami, J. Atulasimha, S. Bandyopadhyay, and C. A. Moritz, "Architecting for causal intelligence at nanoscale," *Computer*, vol. 48, no. 12, pp. 54–64, 2015.

[16] L. B. Kish, "End of moore's law: thermal (noise) death of integration in micro and nano electronics," *Physics Letters A*, vol. 305, no. 34, pp. 144 – 149, 2002.

[17] V. T. Lee, A. Alaghi, J. P. Hayes, V. Sathe, and L. Ceze, "Energy-efficient hybrid stochastic-binary neural networks for near-sensor computing," *Des. Autom. Test Eur. Conf. Exhib. (DATE), 2017*, pp. 13–18, mar 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7926951/

[18] X. Li, L. Girin, R. Horaud, and S. Gannot, "Estimation of the direct-path relative transfer function for supervised sound-source localization," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 24, no. 11, pp. 2171–2186, 2016.

[19] M. I. Mandel, R. J. Weiss, and D. P. Ellis, "Model-based expectation-maximization source separation and localization," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 18, no. 2, pp. 382–394, 2010.

[20] V. K. Mansinghka, D. Selsam, and Y. Perov, "Venture: A higher-order probabilistic programming platform with programmable inference," *arXiv preprint*, vol. arXiv:1404.0099, 2014.

[21] T. May, S. van de Par, and A. Kohlrausch, "A probabilistic model for robust localization based on a binaural auditory front-end," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 19, no. 1, pp. 1–13, 2011.

[22] B. Milch, B. Marthi, S. Russell, D. Sontag, D. L. Ong, and A. Kolobov, "BLOG: Probabilistic models with unknown objects," *Statistical relational learning*, p. 373, 2007.

[23] A. Mizrahi, N. Locatelli, R. Matsumoto, A. Fukushima, H. Kubota, S. Yuasa, V. Cros, J. V. Kim, J. Grollier, and D. Querlioz, "Magnetic stochastic oscillators: Noise-induced synchronization to underthreshold excitation and comprehensive compact model," *IEEE Trans. Magn.*, vol. 51, no. 11, pp. 1–4, 2015.

[24] A. Pfeffer, "Practical probabilistic programming," in *Inductive Logic Programming.* Springer, 2011, pp. 2–3.

[25] T. Pfeil, A. C. Scherzer, J. Schemmel, and K. Meier, "Neuromorphic learning towards nano second precision," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, Aug 2013, pp. 1–5.

[26] M. Raspaud, H. Viste, and G. Evangelista, "Binaural source localization by joint estimation of ILD and ITD," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 18, no. 1, pp. 68–77, 2010.

[27] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman, "How to grow a mind: Statistics, structure, and abstraction," *Science*, vol. 331, no. 6022, pp. 1279–1285, 2011. [Online]. Available: http://science.sciencemag.org/content/331/6022/1279

[28] B. Vigoda, "Analog logic: Continuous-time analog circuits for statistical signal processing," Ph.D. dissertation, Massachusetts Institute of Technology, 2003.

[29] J. Woodruff and D. Wang, "Binaural localization of multiple sources in reverberant and noisy environments," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 20, no. 5, pp. 1503–1512, 2012.