IGMN: A Connectionist Approach for Incremental Robotic Tasks

Milton Roberto Heinen\(^1\) and Paulo Martins Engel\(^2\) (advisor)

\(^1\) UDESC – Center of Technological Sciences
Joinville, CEP 89219-710, SC, Brazil
\(^2\) UFRGS – Informatics Institute
Porto Alegre, CEP 91501-970, RS, Brazil
miltonh@joinville.udesc.br, engel@inf.ufrgs.br

Abstract. This works presents IGMN (standing for Incremental Gaussian Mixture Network), a connectionist approach for incremental robotic tasks. IGMN is based on strong statistical principles (Gaussian mixture models) and asymptotically converges to the optimal regression surface as more training data arrive. Moreover, IGMN learns incrementally from data flows (each data can be immediately used and discarded), it is not sensible to initialization conditions, does not require fine-tuning its configuration parameters and has a good computational performance, thus allowing its use in real time control applications. Through several experiments using the proposed model it is demonstrated that IGMN can be successfully used in real time robotic tasks such as concept formation, robot control and mapping. Therefore, IGMN is a very useful machine learning tool for incremental and real time robotic tasks.

Keywords. Artificial neural networks, Bayesian methods, concept formation, incremental learning, Gaussian mixture models, mapping, autonomous robots.

Student level: PhD (Doutorado)
Date of conclusion: 15/03/2011 (This paper should be considered in CTDR)
URL: http://www.lume.ufrgs.br/bitstream/handle/10183/29015/000773175.pdf?sequence=1
Publications derived from this work: [5–17, 20]

1 Introduction

Artificial neural networks (ANNs) [3] are mathematical or computational models inspired by the structure and functional aspects of biological neural networks. Although in the last decades neural networks have been successfully used in several tasks, including signal processing, pattern recognition and robotics, most ANN models have some disadvantages that difficult their use in incremental function approximation and robotic tasks. The Backpropagation learning algorithm, for instance, requires several scans over all training data, which must be complete and available at the beginning of the learning process, to converge for a good solution. Moreover, after the end of the training process the synaptic weights are “frozen”, i.e., the network loses its learning capabilities. Therefore, these drawbacks difficult the use of ANNs in robotics because in this kind of application the training data are available just instantaneously to the robot, and in general an action must be taken using the sensory information available at the moment.
This work proposes a new artificial neural network model, called IGMN (standing for Incremental Gaussian Mixture Network) [4], which is able to tackle great part of these problems described above. IGMN is based on parametric probabilistic models (Gaussian mixture models), that have nice features from the representational point of view, describing noisy environments in a very parsimonious way, with parameters that are readily understandable. The main advantages of IGMN over other neural network models are: (i) IGMN learns incrementally using a single scan over the training data; (ii) the learning process can proceed perpetually as new training data arrive; (iii) IGMN handles the stability-plasticity dilemma [2] and does not suffer from catastrophic interference [3]; (iv) the ANN topology is defined automatically and incrementally; (v) IGMN is not sensible to initialization conditions; and (vi) IGMN has a good computational performance, which allows its use in real time robotics.

The remaining of this paper is organized as follows. Section 2 presents the ANN model proposed in this work; Section 3 describes some experiments performed to evaluate the proposed model in robotic tasks; and Section 4 provides some final remarks.

2 Incremental Gaussian Mixture Network

This section presents the neural network model proposed in this work, called IGMN. Unlike other neural network models, in IGMN we don’t use the words input and output to represent the data features of a training sample such as \( z = \{a, b\} \), for instance. Instead, we consider that the data vectors \( a \) and \( b \) are different sensory and/or motor modalities with distinct domains, and one modality (e.g. \( a \)) can be used to estimate another (e.g. \( \hat{b} \)). This occurs because in IGMN the information flow is bidirectional, i.e., \( a \) can be used to estimate \( \hat{b} \), \( b \) can be used to estimate \( \hat{a} \) and both can be used to compute the joint posterior probability \( p(j|z) \).

Figure 1 shows the IGMN architecture. It is composed by an association region \( \mathcal{P} \) (in the top of this figure) and many cortical regions, \( \mathcal{N}^A, \mathcal{N}^B, \ldots, \mathcal{N}^S \). All regions have the same number of neurons, \( M \). Initially there is a single neuron in each region (i.e., \( M = 1 \)), but more neurons are incrementally added when necessary using an error driven mechanism. Each cortical region \( \mathcal{N}^k \) receives signals from the \( k \)th sensory/motor modality, \( k \) (in IGMN there is no difference between sensory and motor modalities), and hence there is a cortical region for each sensory/motor modality.

Another important feature of IGMN is that all cortical regions \( \mathcal{N} \) execute a common function, i.e., they have the same kind of neurons and use the same learning algorithm. Moreover, all cortical regions can run concurrently, thus improving the performance on parallel architectures. Each neuron \( j \) of region \( \mathcal{N}^k \) performs the following operation:

\[
p(k|j) = \frac{1}{(2\pi)^{D^k/2} \sqrt{|C^k_j|}} \exp \left\{ -\frac{1}{2} (k - \mu^k_j)^T C_j^k (k - \mu^k_j) \right\}, \tag{1}
\]

i.e., a multivariate Gaussian distribution, where \( D^k \) is the dimensionality of \( k \) (different sensory/motor modalities \( k \) can have different dimensions \( D^k \)). Each neuron \( j \) maintains a mean vector \( \mu^k_j \) and a covariance matrix \( C^k_j \).
In IGMN the regions are not fully connected, i.e., the neuron \( j \) of \( N^K \) is connected just to the \( j \)th neuron of \( P \), but this connection is bidirectional. It is important to notice that there are no synaptic weights in these connections, i.e., all IGMN parameters are stored in the neurons themselves. A bottom-up connection between \( N^K \) and \( P \) provides the component density function \( p(k|j) \) to the \( j \)th neuron in \( P \). Therefore, a neuron \( j \) in the association region \( P \) is connected with the \( j \)th neuron of all cortical regions \( N \) via bottom-up connections and computes the a posteriori probability using the Bayes’ rule:

\[
p(j|z) = \frac{p(a|j) p(b|j) \ldots p(s|j) p(j)}{\sum_{q=1}^{M} p(a|q) p(b|q) \ldots p(s|q) p(q)},
\]

where it is considered that the neural network has an arbitrary number, \( s \), of cortical regions and \( z = \{a, b, \ldots, s\} \). The dotted lines in Figure 1 above indicate the lateral interaction among the association units for computing the denominator in (2).

Each neuron \( j \) of the association region \( P \) maintains its a priori probability, \( p(j) \), an accumulator of the a posteriori probabilities, \( s_{p,j} \), and an association matrix to store the correlations among each sensory/motor modality. If a neural network has two cortical regions, \( N^A \) and \( N^B \), for instance, then the association matrix \( C^A_j B \) will have two dimensions and size \( D^A \times D^B \). The top-down connections between \( P \) and \( N^K \), on the other hand, returns expectations to \( N^K \) which are used to estimate \( \hat{k} \) when \( k \) is missing.

### 2.1 IGMN operation

IGMN adopts an error-driven mechanism to decide if it is necessary to add a neuron in each region for explaining a new data vector \( z \). In mathematical terms, the ANN structure is changed (new neurons are added to the cortical and association layers) if the instantaneous approximation error \( \varepsilon \) is larger than a user specified threshold \( \varepsilon_{max} \).

The following subsections describe the IGMN operation during learning and recalling. To simplify our explanation, we will consider that the neural network has just two cortical regions, \( N^A \) and \( N^B \), that receive the stimuli \( a \) and \( b \), respectively. It will be also considered that we are estimating \( \hat{b} \) from \( a \) during recalling. But it is important to say that: (i) IGMN can have more than two cortical regions (one for each sensory/motor stimulus \( k \)); and (ii) after training the same neural network can be used to estimate either \( \hat{a} \) or \( \hat{b} \) (i.e., there is no difference between inputs and outputs).
2.2 Learning mode

Before the learning process starts, the neural network is empty, i.e., all regions have \( M = 0 \) neurons. When the first training pattern \( z^1 = \{a^1, b^1\} \) arrives, a neuron in each region is created centered on \( z^1 \) and the neural network parameters are initialized as follows:

\[
sp_1 = 1.0; \quad C_i^{AB} = 0; \quad \mu_i^A = a^1; \quad \mu_i^B = b^1; \quad C_i^A = \sigma_{ini}^{A_1} I; \quad C_i^B = \sigma_{ini}^{B_2} I,
\]

where the subscript ‘1’ indicates the neuron \( j = 1 \) in each region, \( sp \) is an accumulator of posterior probabilities maintained in the association region \( \mathcal{P} \), \( 0 \) is a zero matrix of size \( D^A \times D^B \), \( \sigma_{ini}^A \) and \( \sigma_{ini}^B \) are diagonal matrices that define the initial radius of the covariance matrices (the pdf is initially circular but it changes to reflect the actual data distribution as new training data arrive) and \( I \) denotes the identity matrix. \( \sigma_{ini}^A \) and \( \sigma_{ini}^B \) are initialized using a user defined fraction \( \delta \) of the overall variance (e.g., \( \delta = 1/100 \)) of the corresponding attributes, estimated from the range of these values according to:

\[
\sigma_{ini}^{K_j} = \delta [\max(k) - \min(k)],
\]

where \([\min(k), \max(k)]\) defines the domain of a sensory/motor modality \( k \) (throughout this paper the symbol \( k \) will be used to indicate any sensory/motor modality, i.e., either \( a \) or \( b \) in this case). It is important to say that it is not necessary to know the exact minimum and maximum values along each dimension to compute \( \sigma_{ini}^{K_j} \), but just the approximate domain of each feature instead.

When a new training pattern \( z^t \) arrives, all cortical regions are activated, i.e., \( p(k|j) \) is computed using Equation 1 above, and the probabilities \( p(z^t|j) \) are sent to the association region \( \mathcal{P} \), which computes the joint posterior probabilities \( p(j|z^t) \) using the Bayes’ rule in (2). After this, the posterior probabilities \( p(j|z^t) \) are sent back to all cortical regions, i.e., \( \mathcal{N}^A \) and \( \mathcal{N}^B \), which compute their estimates as follows:

\[
\hat{b} = \sum_{j=1}^{M} p(j|z^t)[\mu_j^B + C_j^{BA} C_j^{-1}(\hat{a} - \mu_j^A)],
\]

\[
\hat{a} = \sum_{j=1}^{M} p(j|z^t)[\mu_j^A + C_j^{AB} C_j^{-1}(\hat{b} - \mu_j^B)],
\]

where \( C_j^{AB} \) is the \( j \)th association matrix maintained in association region \( \mathcal{P} \) and \( C_j^{BA} \) is its transpose. Using these estimates the normalized approximation error \( \epsilon \) is given by:

\[
\epsilon = \max_{k \in \mathbf{z}} \left\{ \max_{i \in \mathcal{D}^C} \left[ \frac{||k_i^t - \hat{k}_i||}{\max(k_i) - \min(k_i)} \right] \right\},
\]

where \([\min(k_i), \max(k_i)]\) defines the domain of the sensory/motor feature \( k_i \). Again \( \min(k_i) \) and \( \max(k_i) \) do not need to be the exact minimum and maximum values of \( k \) – they may be just approximations of the domain of each \( k_i \) feature (in fact \( \min(k_i) \) and \( \max(k_i) \) are used just to make IGMN more independent from the range of the data features). If \( \epsilon \) is larger than a user specified threshold, \( \epsilon_{max} \), than \( z^t \) is not considered as represented by any existing cortical neuron in \( \mathcal{N}^K \). In this case, a new unit \( j \) is created in each region and centered on \( z^t \) and all priors of \( \mathcal{P} \) are recomputed by:

\[
p(j)^* = \frac{sp_j}{\sum_{q=1}^{M} sp_q}.
\]
Otherwise (if \( z \) is well explained by any of the existing Gaussian units), the a posteriori probabilities \( p(\mathbf{j}|z^t) \) are added to the current value of the \( sp(j) \) on \( P \):

\[
sp^*_j = sp_j + p(\mathbf{j}|z^t), \quad \forall j,
\]

and the priors \( p(j) \) are recomputed using (7). Then \( \omega_j = p(\mathbf{j}|z^t)/sp^*_j \) is sent back to all cortical regions, and the parameters of all neurons in \( N^K \) are updated using the following recursive equations:

\[
\mu^K_j^* = \mu^K_j + \omega_j (\mu^K_j - \mu^K_j^*)
\]

\[
C^K_j^* = C^K_j - (\mu^K_j^* - \mu^K_j)(\mu^K_j^* - \mu^K_j)^T + \omega_j \left[ (z - \mu^K_j^*)(z - \mu^K_j^*)^T - C^K_j \right],
\]

where the superscript ‘*’ refers to the new (updated) values. Finally the association matrix \( C^{AB}_j \) is updated using the following recursive equation:

\[
C^{AB}_j^* = C^{AB}_j - (\mu^A_j^* - \mu^A_j)(\mu^B_j^* - \mu^B_j)^T + \omega_j \left[ (a^t - \mu^A_j^*)(b^t - \mu^B_j^*)^T - C^{AB}_j \right].
\]

### 2.3 Recalling mode

In the recalling mode, a stimulus (e.g., \( a \)) is presented to a partially trained neural network (as the learning process proceeds perpetually, in IGMN we never consider that the training process is over), which computes an estimate for another stimulus (e.g., \( \hat{b} \)). As said before, IGMN can be used to estimate either \( \hat{a} \) or \( \hat{b} \), but to simplify our explanation we will consider that we are estimating \( \hat{b} \) from \( a \).

Initially the stimulus \( a \) is received in the cortical region \( N^A \), where each neuron \( j \) computes \( p(a|j) \) using (1). These predictions are sent to the association region \( P \) through the bottom-up connections, which is activated using just \( p(a|j) \):

\[
p(j|a) = \frac{p(a|j)p(j)}{\sum_{q=1}^{M} p(a|q)p(q)}.
\]

After this, \( p(j|a) \) is sent to the cortical region \( N^B \) via the top-down connections, and \( N^B \) computes the estimated stimulus \( \hat{b} \) by:

\[
\mathbf{x}_j^B = \mu^B_j + C^{BA}_j C^{-1}_j (\mathbf{a} - \mu^A_j),
\]

\[
\hat{b} = \sum_{j=1}^{M} p(j|a) \mathbf{x}_j^B.
\]

### 3 Experiments

This section describes the use of IGMN in some applications such as concept formation, robotic control and mapping. Subsection 3.1 discusses the use of IGMN for incremental concept formation, which is an important task in machine learning and robotics. Subsection 3.2 shows how IGMN can be used to compute the control actions for a mobile robot performing a wall following behavior. Finally, Subsection 3.3 demonstrates how IGMN can be used as a feature-based mapping algorithm.
3.1 Incremental concept formation

In the experiments described in this section the data consist of 10 continuous values provided by the Pioneer 3-DX simulator software ARCOS (Advanced Robot Control & Operations Software). A Pioneer 3-DX robot has 8 sonar sensors, disposed in front of the robot at regular intervals, and a two-wheel differential, reversible drive system with a rear caster for balance.

The IGMN network used in these experiments has two cortical regions, $\mathcal{N}^S$ and $\mathcal{N}^V$. The cortical region $\mathcal{N}^S$ tackles the sonar readings $s = \{s_1, s_2, \ldots, s_8\}$, and the cortical region $\mathcal{N}^V$ receives the speeds applied at the robot wheels at time $t$, i.e., $v = \{v_1, v_2\}$. To decide what is the most active concept at time $t$, the maximum likelihood (ML) hypothesis $\ell = \arg \max_j p(j|z)$, where $z = \{s, v\}$, is used. It is important to note that IGMN computes and maintains the a posteriori probabilities of all concepts at each time, and hence it can be used in applications such as the so called multi-hypothesis tracking problem in robotic localization domains [1, 21]. The configuration parameters used in the following experiments are $\delta = 0.01$ and $\epsilon_{max} = 0.1$. It is important to say that no exhaustive search was performed to optimize the configuration parameters.

The first experiment was accomplished in an environment composed of six corridors (four external and two internal), and the robot performed a complete cycle in the external corridors. Figure 2(a) shows the segmentation of the trajectory obtained by IGMN when the robot follows the corridors of this environment. IGMN created four units, corresponding to the concepts “corridor” (1: plus sign), “wall at right” (2: circle), “corridor / obstacle front” (3: asterisk) and “curve at left” (4: cross). The symbols in the trajectory of Figure 2(a) represent the ML hypothesis in each robot position, and the black arrow represents the robot starting position and direction.

![Segmentations obtained by IGMN](image)

Fig. 2. Segmentations obtained by IGMN

The next experiment was performed in a more complex environment, composed by two different sized rooms connected by a short corridor. This environment was originally used in [19] and [18]. Figure 2(b) shows the segmentation performed by IGMN in
this experiment. IGMN has created seven clusters, corresponding to the concepts "wall at right" (1: plus sign), "corridor" (2: circle), "wall at right / obstacle front" (3: asterisk), "curve at left" (4: cross), "bifurcation / obstacle front" (5: square), "bifurcation / curve at right" (6: five-pointed star) and "wall at left / curve at right" (7: hexagram).

Comparing these experiments, it can be noticed that some similar concepts, like "curve at left" and "obstacle front", were discovered in both experiments, although these environments are different (the environment shown in Figure 2(a) has many corridors whilst that one shown in Figure 2(b) has two large rooms and just one short corridor). This points out that concepts extracted from a data flow corresponding to a specific sensed environment are not restricted to this environment, but they form an alphabet that can be reused in other contexts. This is a useful aspect, that can improve the learning process in more complex environments.

3.2 Estimating the desired speeds in a mobile robotics application

In the experiments described in the previous section the main goal was to identify natural groupings in the input space, which were represented by the IGMN Gaussian units, and therefore the IGMN estimation/prediction capabilities were not used. In this section we will use these estimation/prediction capabilities to compute the desired actions (i.e., the wheel speeds) for a mobile robot performing a wall following behavior in a simulated environment. These experiments are relevant because in robotic control tasks usually it is not possible to predict all situations that occur in the real world, and hence the robot needs to learn from experience while interacting with the environment.

In the next experiment IGMN was used to estimate the desired speeds in the same environment described above (Figure 2(b)) and using a IGMN neural network with two cortical regions, $N^S$ and $N^V$. Figure 3(a) shows the results obtained in this experiment, where the x axis corresponds to the time index of the sensor readings (the training database is composed by 2070 data samples) and the y axis corresponds to the difference between the right and left motor speeds, i.e., $y_d(t) = v_1 - v_2$. A positive value in $y_d(t)$ corresponds to a left turn in the robot trajectory and a negative value corresponds to a right turn. The solid gray line in Figure 3(a) represents the desired $y_d(t)$ values and the dashed black line represents the difference between the actual IGMN outcomes, i.e.: $y_o(t) = \hat{v}_1 - \hat{v}_2$. It is important to say that the region $N^V$ has size $D^V = 2$, i.e., the difference $y_o(t)$ was computed just to improve the visualization in Figure 3(a).

The configuration parameters used in this experiment are $\delta = 0.01$ and $\varepsilon_{max} = 0.01$. We can notice in this figure that the approximation is quite good: the NRMS error is just 0.034598. The time required for learning was 0.351 seconds, and 40 Gaussian units were added in each region. Moreover, if we use this trained network to control the robot, it will follow the same trajectory shown in Figure 2(b), i.e., the robot follows the "target trajectory" very well. These results are impressive because the robot does not receive any information about its own position (it just receives the sonar readings).

The next experiment was performed in a more complex and irregular environment, shown in Figure 3(b), where the robot was preprogrammed to follow the external walls of the simulated environment. IGMN was trained using the data corresponding to one lap in the environment (1631 samples), and was tested using another independent lap (1551 samples). The solid gray curve in Figure 3(b) shows the trajectory followed by
the robot during the learning phase, and the dashed black curve shows the trajectory followed by the robot using the trained IGMN network to control its actions. We can notice in this figure that the robot have not simply repeated the “target trajectory” after training. On the contrary, it follows a softer trajectory, which indicates that IGMN has learned a wall-following behavior rather than just reproducing the target trajectory.

3.3 Robotic mapping

This subsection describes some experiments in which IGMN is used as a feature-based mapping algorithm, i.e., the multivariate Gaussian units represent the main features (objects, walls, etc) of the environment. In these experiments were used two kinds of sensory information: (i) data provided by a laser scanner; and (ii) data provided by sonar sensors. The robotic platform used in these experiments is a real Pioneer 3-DX mobile robot. This robot has 8 sonars, disposed in front of the robot at regular intervals, which sensitivity ranges from 10 centimeters to over four meters. The laser scanner installed in the robot is a Sick LMS-200, which in ideal conditions is capable of measuring out to 80m over a 180° arc. Figure 4 shows the real environment used in the simulation. It is composed by two long corridors of 2.3 × 30 meters liked by two short corridors of 2.3 × 10 meters, as shown in the schematic map presented in Figure 5.

The first experiments performed using sensor data provided by the Sick LMS-200 laser scanner installed on the Pioneer 3-DX robot. In these experiments, the robot was manually controlled to perform one loop in the environment shown in Figure 4. A complete laser scan is received at each 100 milliseconds, and the matching and merging processes were performed at each second (i.e., using 10 complete scans). The first experiment was conducted using $\tau_{nov} = 1.0e - 8$, and this small value had produced few large clusters, as can be seen in Figure 6.

In this figure the occupancy probabilities are represented by shades of gray, where darker regions represent higher occupancy probabilities (close to 1) and lighter regions correspond to probabilities close to 0. It can be noticed in Figure 6 that the proposed
The model was able to model with reasonable accuracy the main features of the environment, because even with many irregularities each wall was modeled using a single component. It is important to highlight that the proposed model does not have any random initialization and/or decision, and thus the obtained results are always identical for the same dataset and parameters.

The next experiment was performed using the same conditions described above, but using a larger $\tau_{nov} = 1.0\epsilon - 2$ value which makes the system more sensible to small variations in the laser data. The obtained results are shown in Figure 7. It can be noticed that much more Gaussian components were generated in this experiment (76 Gaussian components were generated). Nevertheless, these components fit very well the environment features, existing almost one cluster for each feature (doors, saliences in the walls, etc.). Moreover, each wall is modeled by a thin, long cluster which closely represents the center of the wall. These experiments show that the proposed model is able to create useful representations of the environment, and these representations can be coarse (Figure 6) or fine (Figure 7) depending on the $\tau_{nov}$ configuration value.

Although these experiments were performed using high quality laser range data, the proposed model is not restricted to this kind of sensory data. In the next experiment the proposed model is evaluated using data provided by the sonar sensor array of the Pioneer 3-DX robot. The time interval of each scan is 100 milliseconds, and the local mixture model is generated using 100 complete scans (i.e., the local model is matched against the global model at each 10 seconds). Figure 8 illustrates the occupancy prob-
abilities of this map (again darker regions represents higher occupancy probabilities). Although in this experiment more Gaussian distributions were generated (216), this is yet a low value compared to the number of cells necessary to map this environment using grid-based mapping techniques. The configuration parameters used in this experiment are the same of the previous one (i.e., $\tau_{nov} = 1.0e - 2$).

We can notice that the proposed model was able to create a reasonable map of the environment even using a very noisy sensory data. The results obtained in Figure 8
show that the proposed model does not require only laser scanner sensors, but of course the quality of the maps is superior when high quality sensor data is used.

Although in the last experiment many Gaussian distributions were generated (216), this is yet a small number compared to the number of cells necessary to map this environment using grid-based mapping techniques. To demonstrate this, Figure 9 shows a grid map generated using the same sonar readings that generated the map presented in Figure 8. Each cell in Figure 9 represent an area of $20 \times 20$ centimeters, and $200 \times 80 = 16000$ were necessary to represent this map. It can be noticed in Figure 9 that even using 16000 cells the map resolution is inferior than that presented in Figure 8. Moreover, the number and size of the grid cells must be previously informed and kept fixed.

![Grid map generated using the same data of Figure 9](image)

**Fig. 9.** Grid map generated using the same data of Figure 9

### 4 Conclusion

This work presented IGMN, a connectionist approach for incremental function approximation and prediction. IGMN is based on strong statistical principles and asymptotically converges to the optimal regression surface as training data arrive. To validate the proposed model some experiments were performed, and these experiments have demonstrated that: (i) IGMN learns incrementally using a single scan over the training data; (ii) It does not require to fine tune its configuration parameters; (iii) the IGMN performance is comparable to those of other ANN models but without requiring that the training data set be complete and available at the beginning of the learning process. Therefore, IGMN can be used successfully in many applications such as concept formation, robotic control and mapping.

### References