3. The INSS system

The INSS system is composed of five modules: Symbolic-Module (Symbolic Inference Engine), NeuComp (Construction of a network from rules), NeuSim (ANN learning and recall), Extract (Rule extraction), and Valid (Validation of acquired knowledge, by means of study of relations between rules and examples). The INSS system components are represented in Figure 3.

Our system uses the CLIPS language (C Language Integrated Production System) [GIA93], developed by the STB-NASA, as its symbolic module. Our system also provides facilities to transfer rules and examples to/from the specific rule syntax used in this language and the syntax used in our tools (NeuComp/NeuSim/Extract). The NeuSim module can be also used as a forward-chaining inference engine once the symbolic rules have been transferred to the connectionist module. The activation of neurons (obtained by rule compilation) works exactly as a set of rules that are individually fired or not.

The NeuComp module can process elementary production rules of order 0 which are equivalent to IF/THEN forms such as:

\[ IF \text{ <Condition>}(\text{TRUE/FALSE}) AND/OR <\text{Condition}> (\text{TRUE/FALSE})... THEN <\text{Conclusion}> \]

The rule compilation follows the method described by Towell [TOW91, TOW93]. The result of the translation is a network composed of a set of units linked by weighted connections (see Figure 4). The activation of this network, before learning, leads exactly to the same results (outputs) as those obtained with the set of rules.

![Figure 4 - Rules to network translation ("rule insertion")](image)

\[
\text{Neuron output:} \quad \text{Out} = \frac{1}{1 + \exp(- \text{Sum})} \\
\text{Sum} = \sum_j W_{ij} \cdot \text{lj} + \Theta \\
\]

W: Connection Weight 
\text{lj}: Neuron Input [0..1] 
\Theta: Threshold (Bias) 
P: Number of positive antecedents
We also extended the rules used by KBANN to allow the application of INSS to robotic problems and to study what we called "high level rules" [REY97, REY97a]. Therefore, NeuComp accepts production rules of order 0+ (rules including value intervals). We implemented the usage of comparison functions of the following type:

\[
\text{<Feature> <Operator> <Value> or <Feature> <Operator> <Feature>},
\]

where Operator is Greater Than, Less Than or Equal.

Resulting in rules of this kind:

\[
\text{IF Greater Than(Sensor_S1, 1.0) AND Less Than(Sensor_S1, Sensor_S2) THEN Conclusion_C1}
\]

These rules can be compiled into an ANN composed by simple Perceptron like units (we create feed-forward multi-layer networks with sigmoid based units). A detailed description of all compilation processes, used within INSS, can be found in [OSO98].

As the symbolic rules allow to establish some initial knowledge and then give an initial structure to the network, this approach solves two important problems related to Artificial Neural Networks: on one hand this simplifies the choice of the number and distribution of units, on the other hand we obtain a good assignment of initial values to the connection weights.

The use of the Cascade-Correlation learning algorithm instead of Back-Propagation, in the NeuSim module, allows a quicker learning [FAH90, SCH93], with higher performance results [SCH93, THR91]. Figure 5 shows an example of the network structure evolution when we apply the Cascade-Correlation learning algorithm. It allows especially constructive learning where the initial knowledge is not mixed with the new acquired knowledge. The importance of such a choice of the learning method is reinforced by studies [SHU91, SHU94] showing that Cascade-Correlation networks can be used to model some aspects of human cognitive development.

![Figure 5 - ANN structure evolution using Cascade-Correlation](image-url)