Semi-Human Instinctive Artificial Intelligence (SHI-AI)

Asadollah Norouzi\textsuperscript{1}, S. Mohammad Mohammadzadeh Ziabary\textsuperscript{2}

\textsuperscript{1}I. Azad University
Computer Department
asad.norouzi@gmail.com

\textsuperscript{2} I. Azad University
Computer Department
softnhard_ir@yahoo.com

Abstract. Providing robots (or any other intelligent embedded system) with manlike instincts will bring major issues of today artificial intelligence out of a deadlock. This paper proposes a nondeterministic decision making theory based on Semi Human Instincts implemented by learned potential fields, using neural networks and fuzzy logic offline and online learning algorithms, which enable the agent to perform in anonymous, dynamic and non-deterministic environments. SHI-AI is like a newly born baby who uses his/her instincts and will gradually become more and more intelligent as the brain learns more about its environment. The use of a new world modeling method called ARPL\textsuperscript{1} in SHI-AI enables the agent to perform better within anonymous environments where positioning is an important and complex issue.

1 Introduction

The research and work on this subject started from year 2000 and after 4 years of work and research and consulting with neurologists and psychologists resulted in presenting the MMLAI\textsuperscript{2} method. It was practically implemented and tested on the Middle-Size Soccer Robots (class F-2000) during RoboCup 2004 competitions, which revealed astonishing results some of which not even expected. This achievement encouraged us to work on it harder to cover its weaknesses and make it more optimized and adaptive to perform more efficient in noisy and anonymous environments such as soccer pitch. This led to the invention of SHI-AI that was, like MMLAI, practically implemented and tested on the Middle-Size Soccer Robots.

The first fundamental principle of this theory is based on instinct definition such that every problem has to be partitioned into its main and complex sections and then find a basic but reliable solution for each section having used the nature laws of whether physics, chemistry, or even mathematics. Providing the agent with these

\textsuperscript{1} Agent Relative Polar Localization (A localizing method that is impendent of stationary or known objects (flags) of the world designed and implemented by the authors)

\textsuperscript{2} Multi Magnetic layered Artificial Intelligence (A new AI approach for Autonomous soccer playing robots designed and implemented by the authors)
collections of instincts, we would have an agent that makes decisions without a particular knowledge and only by its defined instincts even if these decisions are false. For instance, imagine a baby who has just started walking and decides to travel along a path. He/she will not try to walk through obstacles but will instinctively choose a path to avoid or pass obstacles. In fact, this baby has not been taught how to avoid and pass obstacles but he/she instinctively knows that walking through solids is impossible. A similar case can be taken into consideration for animals and plants. This is due to the fact that instincts exist within all livings. The next principle of this theory is learning. Obviously an agent that only makes decisions based on instincts is not an intelligent one. We have two methods of learning in SHI-AI (see Sec.4.3-4.4). As a baby learns (meaning both learning with supervisor and without supervisor) and gains experience, he/she would be able to make more optimized decisions and the chosen traveling paths in case of object avoidance will be more accurate. The third principle of this theory is replacing quantity with quality even within calculations. That is, the volume of calculations is considerable reduced and is more similar to human brain. The last principle is decision making under any circumstances. In fact, with this theory we make sure that there is nothing as unexpected condition because basically no conditions are defined in this theory to have unexpected condition.

In SHI-AI depending on the area of performance basic instincts will be defined for the intelligent agent, and then the agent itself nourishes its instincts using learning techniques and special analytical process of the surrounding environment to make more optimized and realistic decisions. For example, in the model used for the RoboCup Middle Size Soccer Robots the pre-defined instincts for the intelligent agent were to achieve local aims (i.e. opponent goal or ball) and to avoid obstacles (i.e. opponents, teammates, etc). To be more precise, the intelligent agent will be able to dribble both stationary and mobile obstacles without having even a single line of programming code to make a specific decision on that. Passing and avoiding obstacles by robots is one of the main concerns of robotic issues. There are many algorithms developed with various levels of complexity based on different artificial intelligence theories, where we can strongly claim that none of those algorithms have the efficiency of SHI-AI. This is based on the hereunder facts:

- **ARPL** (see Sec.3) has eliminated the need for exact global positioning. Therefore, relative polar localization substitutes the global positioning where no complex algorithm is required which decreases calculation errors and speeds up the decision making system.
- **SHI-AI** is implemented with the least amount of programming code and execution time where there is no specific algorithm for dribbling or defending or such actions.

However, the base algorithm of SHI-AI leads the agent to perform such tasks on different situations even though the performance might not be of the most optimized one. It is noteworthy to mention that the aim of the modern artificial intelligence is not to make optimized decisions but any decision that reaches the final goal with a high probability will be considered as optimized.
2 SHI-AI Layers

SHI-AI is consisted of five collaborating layers (see Fig. 1). The layers are:

1. Gate Layer (GL)
2. Transfer Layer (TL)
3. Decision Layer (DL)
4. Instinctive Virtual Layered World Modeler (IVLWM)
5. Predict Layer (PL)

The collaboration and communication between layers is done via defined protocols. These protocols have been defined to be compatible with any area of performance by only applying minor changes to the low level making.

The design and implementation of the layers has been to provide independent layers. Even though all layers are in connection with one another but the structure and detailed procedure of a layer is not an issue for other layers. This implementation enables a layer to be updated or changed in structure without the necessity to make changes to other layers, but only updating their protocols. It has been considered in the design of the layers that if a layer, under any unexpected circumstances, malfunctions; other layers will be aware of the situation and will report to the exception handler where necessary action will be taken depending on the application where SHI-AI is being implemented on.

2.1 Gate Layer

Gate Layer performs as a gateway between SHI-AI and the surrounding world where all communications between SHI-AI and the hardware world are done through this layer. In fact, this layer receives or retrieves all the inputs from the outside world and prepares executable outputs. This layer can be compatible with any hardware by making minor changes to the GL structure. Gate Layer is in contact with the Transfer Layer where gathered inputs by the Gate Layer are sent to Transfer Layer and the desired outputs are sent to the Gate Layer by the Transfer Layer.
2.2 Transfer Layer

Transfer Layer is responsible of parsing, correcting, and optimizing all the input and output data. This layer receives inputs from the Gate Layer and, if necessary, will make appropriate changes to the data formats and normalizes them to be ready to be sent to the upper and higher layers. Transfer Layer, also, recognizes errors in input data and will correct them before sending them to the upper or lower layers. This layer synchronously sends the same data that is being sent to the Decision Layer, to IVLWM, Learning and Predict Layers where data will be processed by each of the mentioned layers for different purposes. Finally when the optimum decision has been made by the Decision Layer the output will be sent to the Transfer Layer for optimization and then changed to be ready to be sent to the Gate Layer for final execution.

2.3 Decision Layer

The Decision Layer is consisted of two Low-Level and High-Level sub-layers.

The Low-Level Decision is based on static laws which we call instinctive decision making in the real world. This decision making method enables the agent to make logical (but not optimized) decisions without a prior learning process, and furthermore provides the agent with unconscious decision making. This unconscious decision making is similar to the case when an unwise baby touches a hot pot and immediately unconsciously removes his/her hand from the pot. As a result of this unconscious decision making the baby is kept away from danger and learns to avoid hot stuff. Unconscious decision making is inevitable in virtual world and specially anonymous and non-deterministic environments. This sub-layer creates the main output of decision layer which is passed to the Transfer Layer to be executed.

The High Level Decision recognizes and analyses its surrounding environment using appropriate data from the world. In this sub-layer decision making process, other active elements of the world are influential (e.g. in soccer coach and players of a team can and may give other individual players a better view of the playing status or even tell them what to do). The decisions made in this layer are directly influencing the IVLWM. In fact, the decision making process of the agent is to first make a high level decision having enough information from the world, predicted states, and additional information or commands from other active elements of the environment. This decision is then passed to IVLWM to model a world appropriate for the defined formulas of instincts. The part of the decision making process is done by the low level layer where the final decision is sent to Transfer Layer.

We should note that there will be a number decisions made by the agent which are equivalent to the number of available layers. These decisions along with decisions offered from other active elements are sent to a selective function where the final decision is selected based on the surrounding conditions. This function changes depending on the agent's area of performance (See formula 3 of Sec.4.1 to see the RoboCup example of this function).
2.3 Instinctive Virtual Layered World Modeler

This layer is the most important layer of SHI-AI. As the name implies, IVLWM is responsible for converting the agent’s surrounding world to a virtual world where affected by defined laws of instincts formation. The way instincts laws influence the real and virtual worlds depend on decision making conditions and learnings. Note that the virtual world layers number may alter depending on the number decisions agent can make.

This layer directly interacts with the learning layer. Thus, IVLWM generates more applicable and optimized virtual world having been fed by the learning process. As a result of this optimization the Decision Layer would be able to make more successful decisions.

2.5 Predict Layer

The Predict Layer is the forecasting side of information processing. The aim here is to derive information about how the surrounding world will be like at some time \( t_0 + \epsilon_t \) in the future, for some \( \epsilon_t > 0 \), by using data measured up to and including time \( \epsilon_t \). The predicted world is quiet useful for making high level decisions, specially in case of determining action strategies.

This layer plays an important role for error detection and correction. The presence of errors is inevitable and cannot be avoided, however must be controlled and reduced. If we could detect and correct the errors and provide the Decision Layer with more real information of the world, then, obviously, the decisions made by the D.L. would have been more efficient. Thus, we designed this layer to take control of this task. P.L. receives information about the surrounding world from the Transfer Layer and will approximate the state of the world for \( n \) steps ahead. The number of \( n \) can be set and changed depending on the area of performance in the Predict Layer Settings. If the difference between the newly received information from the TL and the predicted information of the future world in the PL exceeds the difference factor defined in the PL Settings, then PL sends a signal to the Exception Handler that there might be a problem with one of the input mediums (e.g. camera) and will replace the incorrect data with an average value between the incorrect data and the predicted one. This average is not necessarily the best but it will, surely, assist the agent to make logical decisions in case of erroneous or unavailable input data. In this case a noise factor will be calculated and produced by the PL which will be used for future calculations. This noise factor can be retrieved by other layers in case needed. According to SHI-AI structure the presence of PL is not vital and it can perform well enough even without this layer.
3 Agent Relative Polar Localization (ARPL)

ARPL is a method for modeling agent’s surrounding world based on polar coordinates of $r$ and $\theta$ where $r$ represents distance and $\theta$ represents angle. In this method, the agent retrieves the location of surrounding objects using the above mentioned coordinate relative to itself. That is, each object will have a distance and angle relative to the agent that results a polar position vector. The collection of these polar position vectors will make the agent's world. (See Fig.2)

![Fig. 2.](image)

To have this method better understood we should now refer to RoboCup implementation of ARPL. In RoboCup implementation of ARPL, we may have two ways of expressing the values of distance. The first one would be exact logarithmic value which is actually the logarithmic position of the object in a parabolic mirror, where robots vision is through an omni-directional parabolic mirror (this position is not the exact metric position of the object since it is not re-calculated through the parabolic formula of the mirror). The latter one which is used by Decision Layer is the linguistic fuzzy representation of the distance. This is done by dividing the circular visible area of the robot into several logarithmic sections defined as linguistic quantities like "close", "near", "far", etc. (See Fig.2 (left picture bottom). The magnitude of these ranges are increased exponentially from the closest point (tangent point) of the agent to the defined far most point. ARPL eliminates the need of having constant and shared references in the surrounding world that allows the agent to avoid using Cartesian calculations ($x = r \cdot \cos(\theta + \hat{\lambda}), y = r \cdot \sin(\theta + \hat{\lambda})$). This ability decreases localization and decision making inevitable errors significantly. Having designed and implemented the Agent Relative Polar Localization we successfully managed to eliminate the need of Exact Global Positioning. In other words, in this method world modeling is not dependent on stationary points (flags), and consequently makes it more reliable compared to other positioning and localizing techniques where not seeing a flag would cause major errors in world modeling. It has been seen during many games that when the human referee is standing in front of a corner post, the attacker robots might suddenly be unable to make correct decision since the flag post would not be visible.
A sensible example of ARPL in the real world would be the case when a pedestrian intends to walk across a street. He sees a street with moving objects (vehicles) as obstacles in his way to pass across the street. Naturally the pedestrian does not know the exact distances of the moving objects from himself. The only information needed to complete the decision is either being close to a moving object or far from it. The rate of this measurement would be “how” (e.g. how close, how far …) and naturally there is no need to know any exact value.

4 SHI-AI Implementation on RoboCup

As we have mentioned earlier, SHI-AI is the developed version of MMLAI. This theory was first implemented and tested on RoboCup 2003 Soccer Simulation League, which resulted in presenting the MMLAI for the following year on RoboCup 2004 Middle Size League. The results and tests taken from the mentioned events some of which not even expected led us to make changes in its performance structure and finally presenting the new and improved version of our theory which we now call Semi-Human Instinctive Artificial Intelligence (SHI-AI). One of the most significant improvements of SHI-AI that was used and tested during RoboCup 2005 MSL competitions is the skill of unsupervised agent learning in addition to the previously implemented offline human learning. This new skill improved the cooperative behavior of our agents considerably. Having 2005 experiences we combined D.L. and IVLWM into a XML based decision tree which can be modified by human using the graphical interface or can be modified by robots accessing the XML script of the decision tree (this is more explained further on).

The implementation of SHI-AI for RoboCup field is consisted of a base namespace that all SHI-AI sections and layers are defined as sub-namespaces and classes. SHI-AI implemented platform is any UNIX compatible operating system using object-oriented programming technique. However, it can also be executed on MS-Windows machines using Cygwin UNIX emulator.

4.1 The decision making process

In SHI-AI’s RoboCup implementation, the robot’s instinct is made using the coulomb charge law. That is, and since our instincts are based on electric fields from now on we call IVLWM as LEFG (Layered Electric Field Generator). This name variation would help better understanding of the hereunder formulas:

$$F = k \frac{q_0 q_1}{r^2}$$  \hspace{1cm} (1)

Where \( q_0 \) is the agent’s charge which will dynamically vary by the DL, \( q_1 \) is the charge of surrounding objects where the magnitude of this charge will be determined
by the LEFG, \( r \) is the parametric distance between the agent and the object, \( k \) is a number that is used instead of \( \frac{1}{4\pi \varepsilon_0} \). This value is set to 1 by default.

Generalizing the formula (1) to all the surrounding objects of the agent, we deduce the hereunder formula for each sub-layer:

\[
\tilde{F}_j = k_j \sum_{i=1}^{n} \left( \frac{q_i q_j}{\rho_i^2}, \theta_i \right)
\]

(2)

Where \( \theta_i \) is the surrounding objects angles relative to the agent, \( n \) is the number of surrounding objects, \( j \) is the sub-layer’s number id, \( \tilde{F}_j \) is the resultant vector for the \( j \)th sub-layer, \( k_j \) is a number that dynamically changes determining the brevity coefficient of robots specially when revealing from a deadlock situation.

As a result of formula (2) two vectors of \( \tilde{F}_1 \) and \( \tilde{F}_2 \) are calculated where the first one represents the move vector and the latter one represents a possible kick vector (for either passing or shooting). Finally \( \tilde{F}_T \) is chosen from either of the calculated vectors using formula (3):

\[
\tilde{F}_T = \Psi(\tilde{F}_1, \tilde{F}_2, \tilde{\Omega})
\]

(3)

Where \( \tilde{\Omega} \) is the unconscious decisioning vector, \( \Psi \) is a selective function that determines the optimum vector from the input vectors where the \( \tilde{\Omega} \) vector has a higher priority.

As was mentioned in Sec.2.3 the decision is made in two levels. In RoboCup implementation the high level decision is consisted of three types of decisions. Strategic Decisions are for instance attack, defend, etc. which are engaged for a particular period of the play for all robots and are usually triggered by the coach. Local Decisions are for instance GotoBall, LocalAttack, GotoGoal, etc. which are directly launched by the robot and are temporarily valid for execution under defined conditions. Electric Fields are generated depending on these Local Decisions. Atomic Decisions are for instance Move, MoveNKick, Stop, MoveNHandleOn, etc. these decisions are considered to be the final output and understandable by Transfer Layer and also determine execution type of \( \tilde{F}_T \). These decisions are sequentially dependent to one another. In fact, having made the Strategic Decision and considering play conditions a Local Decision will be made and following this decision and other play states the Atomic Decision will be made and the Electric Field Layers are generated. For example, if the Strategic Decision is attack, and ball is free the decision for players who see the ball would be as follows: Their first Local Decision is GotoBall, and for executing it the Atomic Decision of MoveNHandleOn will be launched and path to reach the ball will be found by the Electric Fields.
4.2 The Object-Oriented Graphical Decision Tree

One of the most complex parts of SHI-AI implementation for RoboCup05 MSL was coding implementation of robot's main decision tree. In 2004, we have had a very limited decision condition area for robot, and in fact robot had only five Local Decisions and three Atomic Decisions that were combined and consequently don’t care states were eliminated. Therefore, the coding implementation using switch-case statement seemed to be reasonable. However, in 2005 this area of conditions was increased to six Strategic Decisions, twelve Local Decisions and five Atomic Decisions. Thus, a rather huge condition area was formed. Anyhow, this decision tree, too, was coded using switch-case statements. This complexity of the code urged us to find a solution to simplify generating, correcting, tracing, and manipulating the decision tree. Providing easy manipulation of the decision tree was of greater importance when we wanted professional soccer coach and players to modify this decision tree.

XML\textsuperscript{3} [7] was one of the best solutions to reach the above mentioned goals. XML is very similar to everyday language conversation where tree structures can be well implemented in Object-Oriented environments. We made unlimited area of conditions by generating an appropriate DOM\textsuperscript{4} whilst only having a limited code volume of XML. This method also provided us with the hereunder privileges:

- XML code is a more human language friendly language compared to programming languages such as C/C++
- XML code has a script nature structure. That is, any modification and correction does not require re-compilation.
- Independent programs or cross platform programs may use the same reference XML code. This becomes valuably important when it comes to robot learning.
- Due to the nature of Object-Oriented structure, the decision tree would be shown graphically for easy manipulation.
- Implementation of learning algorithms for tracing or correcting the tree over a XML script is simpler and there is no need for having them within robot execution code.

To generate the tree and its graphical interface we used Dia software with own developed plugins. Fig.3 represents a section of the decision tree.

---

3 eXtensible Mark-up Language
4 Data Object Model
Each time robot makes decision from the start point, it goes through the conditions and will reach one of the passive states, active states, or memory states and then finally reaches a leaf. Passive states are considered to be as AND operation between two conditions and are only transactions. Active states are weighted states that act like passive states but learning algorithms use these weights to alter and modify the decision tree. Memory states are states like local strategies and are stored to be used for next decision making. Leaves are consisted of two sections. One is the virtual electric charge generation table for world objects. The latter one is the Atomic Decision of the robot.

As can be seen, using XML the D.L. and LEFG are combined and the decision making process is converted into parsing the tree and calculating $F_T$ based on the virtual electric charge generation table returned by the XML parser. This is then passed to T.L along with the Atomic Decision.

### 4.3 Supervised Online Learning

Fig. 4 shows a block diagram that illustrates the SHI-AI’s Supervised Online Learning. In conceptual terms, the supervisor or the coach is an intelligent agent that analyzes the performance of the playing agents outside the playing field. This online agent is able to interact with the playing agents and leading them to a better and effective play having them learned from the play. These learning are based on neural networks learning algorithms.

The form of supervised learning we have implemented is the Error-Correction Learning. It is a closed-loop feedback system, but the unknown environment is not in the loop. As a performance measure for the system we may think in terms of the mean-square error or the sum of squared errors over the training sample, defined as a function of the free parameters of the system. This function may be visualized as a multidimensional error-performance surface or simply error surface, with the free
parameters as coordinates. The true error surface is averaged over all possible input-output examples. Any given operation of the system under the teacher’s supervision is represented as a point on the error surface. For the system to improve performance over time and therefore learn from the teacher, the operating point has to move down successively toward a minimum point of the error surface; the minimum point may be a local minimum or a global minimum. Our supervised learning system is able to do this with the useful information it has about the gradient of the error surface corresponding to the current behavior of the system. The gradient of an error surface at any point is a vector that points in the direction of steepest descent. In fact, in the supervised learning from examples, the system may use an instantaneous estimate of the gradient vector, with the example indices presumed to be those of time. The use of such an estimate results in a motion of the operating point on the error surface that is typically in the form of a random walk. Nevertheless, we have implemented an algorithm designed to minimize the cost function. Thus, having this algorithm, using an adequate set of input-output examples, and enough time permitted to do the training, enables our supervised learning system to perform such tasks as pattern classification and function approximation.

**Fig. 4.** Block diagram of learning with a coach

### 4.4 Unsupervised Online Learning

SHI-AI’s Online Learning comes into action within the performing field of SHI-AI. Our Online Learning or learning without a teacher plays an important role enabling SHI-AI to be adaptive with environment variables. Online Learning enables the agent that is using SHI-AI to increase its level of intelligence learning and experiencing from its action environment. In Online Learning or learning without a teacher, as the name implies, there is no teacher to oversee the learning process. That is to say, there are no labeled examples of the function to be learned by the network. Here we use the reinforcement learning. In reinforcement learning, the learning of an input-output mapping is performed through continued interaction with the environment in order to minimize a scalar index of performance. Fig.5 shows the
block diagram of our implemented reinforcement learning system built around a critic that converts a primary reinforcement signal received from the environment into a higher quality reinforcement signal called the heuristic reinforcement signal, both of which are scalar inputs. This system is designed to learn under delayed reinforcement, which means that the system observes a temporal sequence of stimuli (i.e. state vectors) also received from the environment, which eventually result in the generation of the heuristic reinforcement signal. Notwithstanding the difficulties of the delayed-reinforcement learning, this learning is very appealing. It provides the basis for the system to interact with its environment, thereby developing the ability to learn to perform a prescribed task solely on the basis of the outcomes of its experience that result from the interaction. In fact, the Decision Layer and LEFG here are in contact with one another so that the LEFG learning process will be aware of the non-successful decisions made by DL.

**Fig. 5.** Block diagram of reinforcement learning

**References:**


