Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Cybernetics

BACHELOR THESIS

**Modelling of Dynamical Systems
for Generating of Behaviour of Artificial Creature**

Prague, 2013 **Martin Paňko**

Declaration

I declare that I have written my Bachelor Thesis myself and used only the sources listed in the enclosed bibliography and the appendixes.

In Prague 24.5. 2013

signature

Acknowledgement

I would like to thank Prof. Assoc. Pavel Nahodil for his experienced advices and positive attitude. Also I would like to thank Ing. Jaroslav Vítků for his time and significant help and guidance.

Abstrakt

Táto bakalárska práca sa zaoberá kontinuálnou identifikáciou "blackbox" systémov v diskrétnej časovej doméne. Hlavným cieľom je vyhodnotiť emergenciu komplexnej časovej dynamiky viacerých modelov jednoduchých dynamických systémov. Práca prezentuje zložený parametrický model, ktorý kombinuje jednoduché modely dynamických systémov. Parametre zloženého modelu sú prispôsobované kontinuálne prostredníctvom optimalizačnej procedúry reprezentovanej evolučným algoritmom. Model je algoritmizovaný a implementovaný ako neurónový model, ktorý môže byť zapojený do hybridnej neurónovej siete, ktorý reprezentuje architektúru autonómneho inteligentného agenta. Cieľom modulu je zlepšiť agentovu interpretáciu časovej dynamiky prostredia. Táto práca je súčasťou výskumu v oblasti umelého života, vedeného na katedre kybernetiky ČVUT v Prahe decentom Pavlom Nahodilom a v súčastnosti aj Ing. Jaroslavem Vítků, jeho doktorantom. Predložená práca prehlbuje niektoré nové prístupy navrhnuté Ing. Vítků v jeho PhD Thesis Proposal.

Abstract

This Bachelor Thesis investigates online identification of a blackbox system in discrete time domain. The main objective is to investigate the emergence of complex temporal dynamics from several models of simple dynamical systems. The ensemble parametric model, that combines simple models of dynamical systems is built. The parameters of the ensemble model are altered online by optimization procedure performed by evolutionary algorithm. The model is algorithmized and implemented as neural module that could be connected into framework of hybrid artificial neural network. This neural network represents an architecture of autonomous intelligent agent and the aim of the module would be to improve agents interpretation of temporal dynamics of the environment. This thesis deepens the research conducted for the last thirteen years by Prof. Assoc. Pavel Nahodil and recently by Ing. Jaroslav Vítků on Department of Cybernetics on CTU in Prague.

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

BACHELOR PROJECT ASSIGNMENT

Student: Martin Paňko

Study programme: Open Informatics

Specialisation: Computer and Information Science

Title of Bachelor Project: Modelling of Dynamical Systems for Generating of Behaviour of Artificial Creature

Guidelines:

1. Study the most commonly used models of dynamic systems.
2. Implement selected models of dynamic system as a re-usable modules in Robotic Operating System (ROS) so that the model parameters can be set using the I/O values of modules online.
3. Test the predicted behavior of each model compared to expectations.
4. Test alternative individual interconnections of modules among themselves and their resulting behavior.
5. Evaluate the possibility of using these modules to generate the behavior of artificial creatures.

Bibliography/Sources: Will be provided by the supervisor.

Bachelor Project Supervisor: doc. Ing. Pavel Nahodil, CSc.

Valid until: the end of the winter semester of academic year 2013/2014

prof. Ing. Vladimír Mařík, DrSc.
Head of Department

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE Fakulta elektrotechnická

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, January 10, 2013

Contents

List of Abbreviations

MSE	Mean squared error
EA	Evolutionary algorithm
ANN	Artificial Neural Network
ROS	Robotic Operating System

List of Symbols

M	Ensemble model operator
S	Blackbox system operator
$\mathbf{\Phi}$	Hyperparameter vector of the ensemble model M
$\mathbf{\Theta}$	Parameter vector of base models
$\|.\|_{mse}$	Mean squared error - optimisation criterion for evolutionary algorithm

List of Figures

List of Algorithms

Chapter 1

Introduction

The Study and design of intelligent agents has been the central problem in Artificial Intelligence for a several years. It is a very large topic combining several science fields and integrating many different approaches. Usually, it is expected that intelligent agent performs actions without relying on the knowledge of its designer (i.e. to be autonomous). In order to achieve that, the autonomous intelligent agent must gather knowledge by perceiving the environment and be able to put it into use to act on the environment. Designing the intelligent autonomous agent becomes a comprehensive problem and a real challenge when the agent is expected to act on the complex environment. In the complex environment, it is difficult for the autonomous agent to gather knowledge to be used to plan and perform its actions. One of the aspects of the complex environment are the unknown phenomena that occur in time domain. It is crucial for the autonomous intelligent agent to be able to continuously interpret these phenomena in order to recognize effects of his potential future actions. This thesis investigates online identification of temporal dynamics occurring in the environment by emergence of simple models of dynamical systems.

Chapter 2

Theoretical Foundation used in Modelling of Dynamical Systems

This chapter covers substantial theory of dynamical systems and its identification. Some of here presented principles are expanded in chapter 4.

2.1 System Identification

System Identification is process of determining mathematical description of a system[1] based on investigation of system's behavior. Investigation is based on two types of information: (1) a priori information about the system and (2) input-output data acquired by system observation. By investigation we usually refer to statistical inference of input-output data (Włodzimierz Greblicki, 2008; Ljung, 2008). The following figure 2.1 illustrates the concept of the system with its inputs and output.

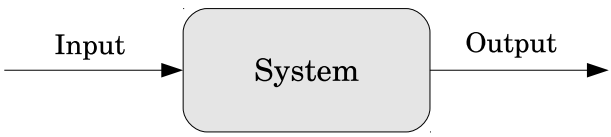

Figure 2.1: *System input-output scheme*

In general, the process of building mathematical description of a system is denoted

[1]Most authors in system identification and other scientific fields do not employ a formal definition of the concept of a system. Many use natural language definitions or descriptions (Pajonk, 2009). Formal definition is preseted e.g by (Backlund, 2000).

as mathematical modelling. Model is a mathematical representation of some aspects of a system and may be build using various degrees of mathematical formalism. Model is therefore an approximation of a system on level of abstaction that is specified by application use of model (Bobal, 2009). Model not necessarily replicates internal structure of the system and phenomena occuring inside of system but mimics external behavior of system which is observed by inputs and outputs. Model is characterized by model structure and parameters. Model structure represets relationships between variables and parameters represent coefficients.

2.1.1 The Methods of System Identification

System identification methods are divided into two groups: parametric and nonparametric. Parametric methods identify system model with an underlying mathematical structure that is associated with a coefficient set or parameters, whereas nonparametric methods model a system directly with its responses (Kashiwagi, 2009).

Depending on amount of the present information about the system, the identification can be denoted as whitebox or blackobox. Whitebox identification derive models from first principles in the respective scientific fields, with only a relatively small amount of parameters left. These parameters have an interpretation in the first principles used to derive the model. In blackbox identification the generic model structure or model family containing many parameters is chosen. These are to be determined in the inverse modelling step by taking measurements into account (Pajonk, 2009). Some whitebox systems may be too complex to be described mathematically on desired level of abstraction.

Sometimes during the process of system identification the concept of "true" description of the system is used. In most cases it is not realistic to achieve a "true" description of the system to be modelled. It is sometimes convenient to assume such a description as an abstraction. It is of the same character as a model but typically much more complex (Ljung, 2008).

System identification is very general and large topic reaching many science fields and application areas. It is extensively used in many subfields of engineering and also in natural (biology, ecology) and political sciences (mainly economics). Many fields has developed it's own nomenclatures, approaches and metodologies. Some are specific for the respective application, some have a broader use. Area of System identification is characterized by small number of leading principles (Pajonk, 2009; Ljung, 2008).

2.1.2 Model Classifications

The structure of this section is based on (Milan Vrožina, 2007; Bobal, 2009; Jiri Roubal, 2008). Some of the aspects in this section will be covered more formally in next section 2.2.2. It is possible to classify the models into several categories according to multiple criterions as following.

Static and dynamic models

Depending on whether the model describes statical or dynamical properties of system, we can classify models as static or dynamic. Static model describes system in which the outputs depend only on the present values of the inputs. Dependency between inputs and outputs is represented by algebraic equations in which time does not ascend as independent variable. On the other hand, dynamic model describes system in which outputs depend on the present and past values of the inputs. This relation is often in form of differential or difference equations (A. Terry Bahill, 2009).

Continuous and discrete models

Depending on the description of change of inputs and outpus we can classify models to countinuous or discrete. Continuous models describe relation between continuous inputs and outputs of a system while discreete models describe relation between inputs and outputs that change in discreete intervals of independent quantity.

Linear and nonlinear models

Model is linear if it satisfies the superposition principle i.e. if model is described by model operator F and its inputs $\mathbf{u}_1$, $\mathbf{u}_2$ it satisfies:

$$F(k_1\mathbf{u}_1 + k_2\mathbf{u}_2) = k_1F(\mathbf{u}_1) + k_2F(\mathbf{u}_2) \tag{2.1}$$

for any $k_1, k_2 \in \mathcal{R}$. If model is not linear it is nonlinear (Jiri Roubal, 2008).

Time-variant and time-invariant models

Parameters of models may be constat or variant in relation to independent quantity. If parameters are constat we refer to model as time-invariant and on the other hand if parameters change in dependence on some quantity we refer to model as time-variant.

Stochastic and determistic models

If model unambigously describes (determines) relation between inputs and outputs we denote model as deterministic. On the other hand we donote model as stochastic if relation between input and output it describes has element of randomness.

2.2 Mathematical Description of Dynamical Systems

There are two common mathematical formalisms for description of a dynamical systems:

State-space description

The state-space description express relation between input u and output y using state variable x: $u \rightarrow x \rightarrow y$.

Input-output description

Input-output description express relation between input u and output y directly: $u \rightarrow y$, usually expressed as transfer function.

The state-space description is more informative because it provides information about internal structure of dynamical system unlike input-output description. There is an infinite number of state-space descriptions for any system with its input u and output y that is described by input-output description (Jiri Roubal, 2008).

2.2.1 State-space Description

State-space description of dynamical system describe system by state $\mathbf{x}$. State is minumum set of variables (denoted as state variables) that fully describe the system and its output $\mathbf{y}$ to any given input $\mathbf{u}$. Input, output and state variables are variant due to some independent variable such as time and we usually refer to them as signals [1]. Input, output and state variables are usually denoted as vectors as following:

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} \mathbf{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_m(t) \end{bmatrix} \mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_p(t) \end{bmatrix} \tag{2.2}$$

where t is independent variable of dynamical system. Dynamical system usualy describe temporal dynamics so this variable often represnts time quantity. From now we will also denote t as time for simplification.

[1] A signal is a function that conveys information about the behavior of a system or attributes of some phenomenon (Priemer, 1991)

Mathematical description of the system in terms of a minimum set of variables $x_i(t), i = 1, 2, \ldots, n$, together with knowledge of those variables at an initial time t_0 and the system inputs for time $t \geq t_0$, are sufficient to predict the future system state and outputs for all time $t > t_0$. Number of state variables n is defined to be order of the system. (Rowell, 2002).

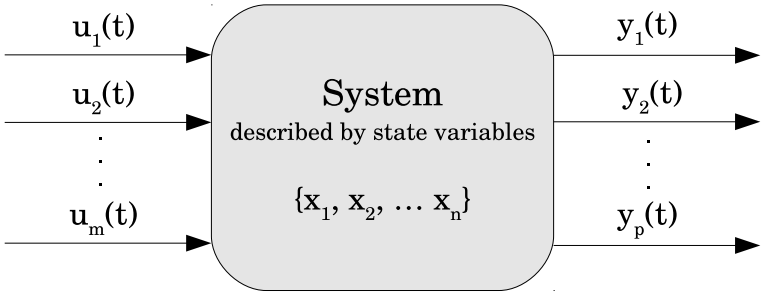

Figure 2.2: *System described by state variables*

2.2.2 Description Dynamical Systems

By following the previous notation, dynamical system can be described by differential equation that descibes the relation between input u, output y and state variable x.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \\ \mathbf{y} &= \mathbf{g}(\mathbf{x}, \mathbf{u}, t) \end{aligned} \tag{2.3}$$

where $\mathbf{x}(0) = \mathbf{x}_0$ is initial state. Sometimes dynamical system does not have input u - in such a case it is denoted as autonomous. If input and output dynamical system does not exciplity depend on time t it is refered to as time-invariant. In this case it is possible to simplify above equation into following form:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{u}), \\ \mathbf{y} &= \mathbf{g}(\mathbf{x}, \mathbf{u}) \end{aligned} \tag{2.4}$$

where $\mathbf{x}(0) = \mathbf{x}_0$ is initial state. It is possible to do another simplification of the description if the functions f and g are linear. In such a case, the dynamical system is

denoted as linear and it is described as follows:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \\ \mathbf{y} &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{aligned} \tag{2.5}$$

where $\mathbf{x}(0) = \mathbf{x}_0$ is initial state and the matrices are:
$A \in R^{n \times n}$ is the dynamics matrix
$B \in R^{n \times m}$ is the input matrix
$C \in R^{p \times n}$ is the output or sensor matrix
$D \in R^{p \times m}$ is the feedthrough matrix

Discrete dynamical system has the form:

$$\begin{aligned} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{aligned} \tag{2.6}$$

where $\mathbf{x}(0) = \mathbf{x}_0$ is the initial state and $t \in \mathcal{Z} = \{0, \pm 1, \pm 2 \dots\}$.
The above notation is based on (Boyd, 2012; Noskievic, 1999).

Chapter 3

State of the Art

In this section I will describe Evolutionary Algorithms and Ensemble learning. The evolutionary algorithm is the optimization procedure employed in learning of the ensemble module. Furthermore, some principles of ensemble learning that are also used in problem formulation are described.

3.1 Evolutionary Algorithms

Evolutionary algorithm is search and optimization procedure that finds its inspiration in Darwinian biological theory of evolution. The basic idea is that if only those individuals of a population reproduce, which meet a certain selection criteria, and the other individuals of the population die, the population will converge to those individuals that best meet the selection criteria. If imperfect reproduction is added the population can begin to explore the search space and will move to individuals that have an increased selection probability and that inherit this property to their descendants. These population dynamics can be described by principle in short as the "survival of the fittest" (Streichert, 2012).

This principle can be easily applied in computer science and applied to solve search and optimization problems from various application scince fields. Evolutionary algorithms operates on populations of data structures. Each data structure represents possible solution to given problem and it is possible to evaluate its quality in means of some particular criterion denoted as *fitness*. Evaluation is performed by function usually referred to fitness function. Depending on fitness of individuals, the selection is performed and then variation operators applied. Variation of population is accomplished by making random

changes in these data structures and by blending parts of different structures. These two processes are called *mutation* and *crossover* and together are referred to as variation operators (Ashlock, 2004).

The Flowchart 3.1 illustrates evolutionary loop of general continuance of standard genetic algorithm.

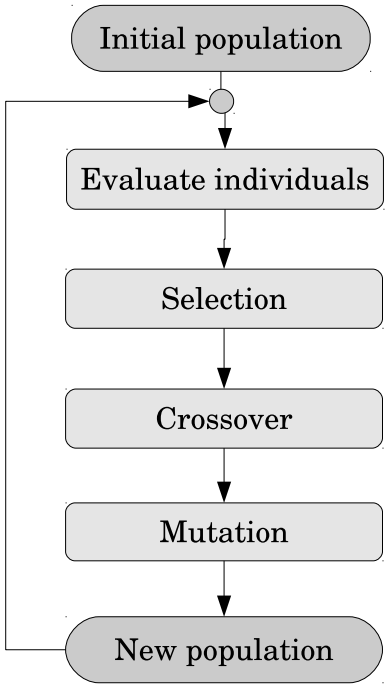

Figure 3.1: *Evolution algorithms flowchart*

3.1.1 The Main Components of Evolutionary Algorithms

I this section I will describe above mentioned main components of evolutionary algorithm. This include fitness function, selection and variation operators.

3.1.1.1 Fitness and Selection

In every evolutionary step, the individuals in current population are evaluated according to some pre-defined quality criterion referred to as the fitness. This ensures, that expected number of times an individual is chosen is approximately proportional to its relative performance in the population. This ensures that high-fitness individuals stand a better chance of reproducing, while low-fitness ones are more likely to disappear (Tomassini,

2012). There are several variants of selection rule. The method of culling, in which all individuals below a given threshold are discarded, can be shown to converge faster than the random version (Eric B. Baum, 1995) as cited in (Russell, 2010).

3.1.1.2 Crossover

The crossover is binary variation operator performing exchange of material between individuals and simulating sexual reproduction. Crossover is stochastic operator that merges information from two parent gonetypes into one or two offspring genotypes. Recombination operators with a higher arity (using more than two parents) are mathematically possible and easy to implement but have no biological equivalent. They are not commonly used although several studies indicate that they have positive effects on evolution (Agoston E. Eiben, 1994). The crossover of two parents is performed by randomly selecting on or more crossover points in parents genotypes and exchanging materials between these points. Figure 3.2 illustrates two parents crossover.

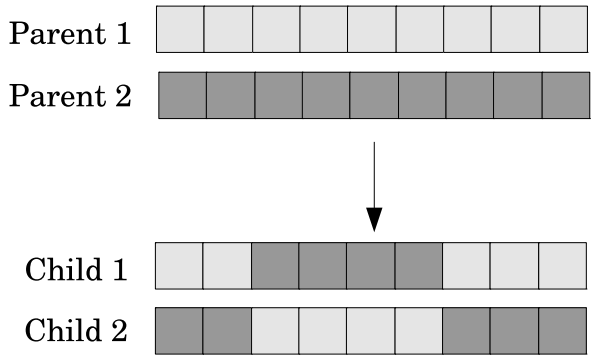

Figure 3.2: *Two point crossover of two parents*

3.1.1.3 Mutation

The mutation is unary variation operator performing stochastic modification of individual. It is applied to one genotype and delivers a slightly modified offspring. Mutation occurs during evolution according to defined mutation probability. This probability should not be high, otherwise, the evolution algorithm would debase into random search. The purpose of the mutation is to introduct diversity into population in order to increase overall fitness of the population. Mutation facilitates a local search and also helps prevent optimization processes to unstruck from local extrema. There are several variants of

implementation of mutation rule: single point mutation, multiple point mutation, probabilistic mutation, Lamackain mutation (Ashlock, 2004; Agoston E. Eiben, 2008). The mutation also has biological equivalent.

The following figure 3.2 illustrates the scheme of signle point crossover.

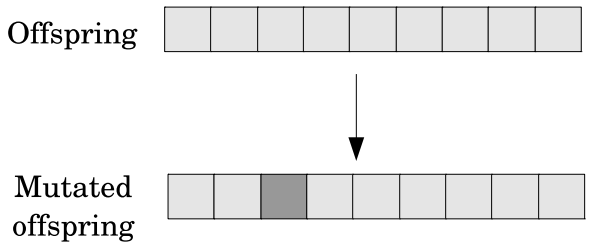

Figure 3.3: *Signgle point mutation scheme*

3.2 Ensemble Learning

3.2.1 General Principles of Ensemble Learning

Ensembles of learning machines constitute one of the main current directions in machine learning research, and have been applied to a wide range of real problems. Despite of the absence of an unified theory on ensembles, there are many theoretical reasons for combining multiple learners, and an empirical evidence of the effectiveness of this approach (Giorgio Valentini, 2002). The metodology of ensemble learning is elaborated mainly in fields of classification and regression.

Each learning algoritm dictates a certain model that comes with a set of assumptions. This inductive bias leads to error if the assumptions do not hold for the data. Each learning algorithm converges to different solution and fails under different circumstances (Alpaydin, 2010). The idea of ensemble methodology is to build a predictive model by combining multiple individual learning algorithms. The resulting model is expected to outperform each of the individual learning algorithms (Rokach, 2009). Figure 3.4 illustrates ensemble model consisting of combination of several learners.

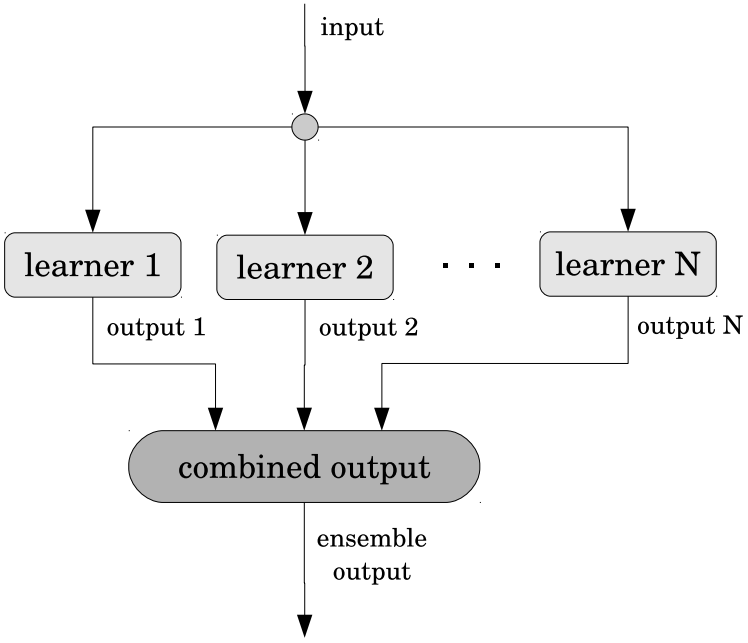

Figure 3.4: *Ensemble model of several learners*

3.2.2 Diversity and Combination of the Base Learners

There are basically two combination scenarios of base learners. In the first scenario, all the classifiers use the same representation of the input pattern. In the second scenario, each classifier uses its only representation of the input pattern i.e. the measurements extracted from the pattern are unique to each base learner (Kittler, 1998)

Different input represantation for each base learner helps to ensure diversity of base learners which is important. Combining several learners is useful only if they generate diverse outputs. Obviously combining the output of several identical classifiers produces no gain. The aim is to find set of diverse learners that complement each other. As a result, methods for creating ensembles center around producing classifiers that disgree on their predictions (David Opitz, 1999). The diversity of base-classifiers is usually achieved by using different learning algorithms that make different assumptions about the data and lead to different base-classifiers or training the learning algorithms on different data sets (Alpaydin, 2010).

The combination strategy of base learners is the is the key aspect of the ensemble learning. The objective of the combination strategy is to combine the base learners in such a way that the correct decisions are amplified and incorrect ones are cancelled out (Polikar, 2006).

Chapter 4

Approach to Problem and Implementation

This chapter explains the motivation behind the problem of system identification investigated by this thesis. Consecutively, the mathematical formulation of the problem is presented and the algorithmization and implementation details are covered.

4.1 Motivation

The aim of the computer model is to provide interpretation of temporal dynamics of the environment for the autonomous agent. It is important for the agent to learn the dynamical behaviour of certain phenomenons occurring in the surrounding environment. Learning of dynamics of the environment in domain of time allows the agent to predict temporal devolopment of the environment and helps to recognise effects of his potential future actions. This is crucial for agent to plan his future actions to carry out, in order to achieve selected goals.

The Model has to be implemented as neural module that will be integrated into framework of hybrid artificial neural network[1]. The purpose of this framework is to design complex modular systems based on concept of hybrid neural networks and use neuro-evolution to evolve into desired architecture of autonomous agent. The framework defines the following three components:

[1]This framework is being developed by Jaroslav Vítků as Nengoros project for his dissertation thesis (Vítků, 2013)

1. Representation of hybrid neural nodes in means of abstract neurons.

2. Information exchange between hybrid neural nodes by means of real-values or spiking communication.

3. Interconnection topology of neural nodes alterable by neuro-evolution.

The framework interconnects several heterogeneous abstract neural nodes of various complexity that implement subsystems with various functionalities such as automated planning, pattern recognition, etc. Each of the subsystems should have the ability to continuously process time-variant data stream received its by input and produce relevant output. The property of continuous processing of the input is determined by purpose of the hybrid modular system to be used for online data streams. The subsystems have form of explicit algorithms or subnetworks and serve as components of the agent architecture. On network interconnection level these neural nodes act as a blackbox systems that communicate only by predefined inputs and outputs by means of signals. Our model will be implemented as neural module that will represent one of these subsystems.

The model will be connected into network as neural module by two inputs and one output as shown on figure 4.1.

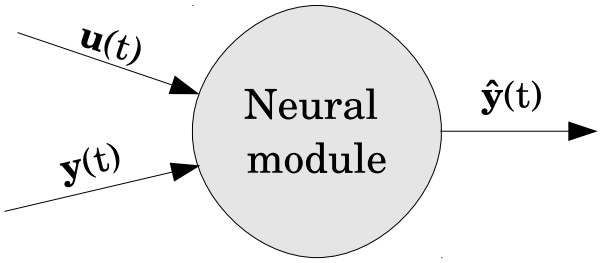

Figure 4.1: *Connection of neural module that implements model into hybrid neural network*

Input $\mathbf{u}(t)$ represents signal of the model that is determined by some of the other modules of hybrid ANN framework. These modules would represent agents high-level decision mechanism that is responsible planning future actions of the agents and acting on environment. Since each module is blackbox on interconnection level, our module does not have any a priori information about the dynamics of the input signal $\mathbf{u}(t)$. The second input $\mathbf{y}(t)$ represents signal that conveys information about temporal dynamics of the

environment captured by agent's sensors and interpreted by agent's perception subsystems. Output $\hat{\mathbf{y}}(t)$ represents models output that conveys information about estimated response of the environment to agents actions.

4.2 Problem Formulation

In the following section the formulation of the problem as online identification of blackbox system will be presented. The section will follow the same notation as was introduced in 4.1 but it will gain more mathematical interpretation.

4.2.1 Formulation as Optimisation Task

Let us consider a set of r time-variant variables and a set of n time-variant variables, both written as a vectors:

$$\mathbf{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_r(t) \end{bmatrix} \mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_n(t) \end{bmatrix} \tag{4.1}$$

where $t \in \mathbb{N}$ denotes independent discrete variable of time. We refer to vector $\mathbf{u}(t)$ as input vector and to $\mathbf{y}(t)$ as output vector of blackbox system. These vectors are related by system operator S which can be written as $\mathbf{y}(t) = S(\mathbf{u}(t))$. System operator S represents the blackbox system we are going to identify[1].

Further, let us have a set of n time-variant variables written as a vector $\hat{\mathbf{y}}(t) = [\hat{y_1}(t), \hat{y_2}(t), \dots \hat{y_n}(t)]^T$ and model operator $M(.|\mathbf{\Theta})$ with $\mathbf{\Theta} = [\theta_1, \theta_2, \dots \theta_L]^T$ denoting its parameters vector. This model generates output $\hat{\mathbf{y}}(t)$, which we denote as $\hat{\mathbf{y}}(t) = M(\mathbf{u}(t)|\mathbf{\Theta})$. We are in a search for a model operator $M(.|\mathbf{\Theta})$ with its parameter vector $\mathbf{\Theta}$ that approximates the system operator $S(.)$ (denoted as $M(.|\mathbf{\Theta}) \approx S(.)$) in means of

[1]The system that is going to be identified is often referred to as a plant in field of Dynamics and control.

minimizing[2] some particular norm:

$$\min \|\hat{\mathbf{y}} - \mathbf{y}\| \tag{4.2}$$

for all corresponding values of input $\mathbf{u}(t)$ and output vector $\mathbf{y}(t)$ that model obtains on its input due to some particular time t. As an approximation criterion for $M(.|\boldsymbol{\Theta}) \approx S(.)$, a commonly used Mean Squared Error criterion was employed. This criterion has the following form:

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_{mse} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{4.3}$$

This expression is optimization criterion we want to minimize. In addition, there is another optimization criterion that tells us how "good" the model is. It is the time $\tau \in \mathbb{N}$, in which we reach the state when $M(.|\boldsymbol{\Theta}) \approx S(.)$. Since we want fast approximation convergence, we want to minimize: $\|\tau\|$, where $\tau \in \mathbb{N}$. Accordingly, the resulting optimization task has the following form:

$$\arg\min_{M(.|\boldsymbol{\Theta})} \Big[\|\hat{\mathbf{y}} - \mathbf{y}\|_{mse} + \tau\Big] \tag{4.4}$$

for all corresponding vector values $\mathbf{u}(t)$ and $\mathbf{y}(t)$ that model obtains due to some particular time t.

4.2.2 Ensemble Model Operator

In this section the $M(.|\boldsymbol{\Theta})$ will be formalized as ensemble model operator.

Let us consider a set of base model operators $\mathcal{M} = \{m_1, m_2, \dots m_L\}$[3], each base model operator $m \in M$ is parametrized dynamical system generating time-variant output $\hat{\mathbf{y}}_m(t)$. We denote this as $\hat{\mathbf{y}}_m(t) = m(\mathbf{u}(t), \theta)$, where $\theta \in \mathbb{R}$ is the parameter.

We suppose that model operator $M(.|\boldsymbol{\Theta})$ that solves our optimization task 4.4 is possible to express as combination of base model operators as following:

$$M(\mathbf{u}(t)|\boldsymbol{\Theta}) = f(m_1(\mathbf{u}(t), \theta_1), m_2(\mathbf{u}(t), \theta_2), \dots m_L(\mathbf{u}(t), \theta_L)|\boldsymbol{\Phi}) \tag{4.5}$$

[2]We would be also satisfied with such a $M(.|\boldsymbol{\Theta})$, that evaluates criterion 4.2 to some "reasonable" value. What is "reasonable" would be determined by application in particular agents architecture

[3]Set of base-model operators is described in detail later in section 4.2.3

where $f(.)$ is base models combining function parametrized by hyperparameter vector $\mathbf{\Phi}$. More specifically, as a function $f(.|\mathbf{\Phi})$ I imposed the linear combination of the models with $\mathbf{\Phi} = [\phi_1, \phi_2, \dots \phi_L]^T$ denoting the weights vector:

$$f(m_1, m_2, \dots m_L|\mathbf{\Phi}) = \sum_{i=1}^{L} \phi_i m_i \tag{4.6}$$

where $\phi_i \in \mathbb{R}$. Accordingly, the resulting ensemble model operator can be expressed in the following form:

$$M(\mathbf{u}(t)|\mathbf{\Theta}) = \sum_{i=1}^{L} \phi_i m_i(\mathbf{u}(t), \theta_i) \tag{4.7}$$

where $\phi_i, \theta_i \in \mathbb{R}$ are parameters (more specifically ϕ_i are hyperparameters) to be optimized. The graphical schematic representation of ensemble model operator is shown on the following figure 4.2.

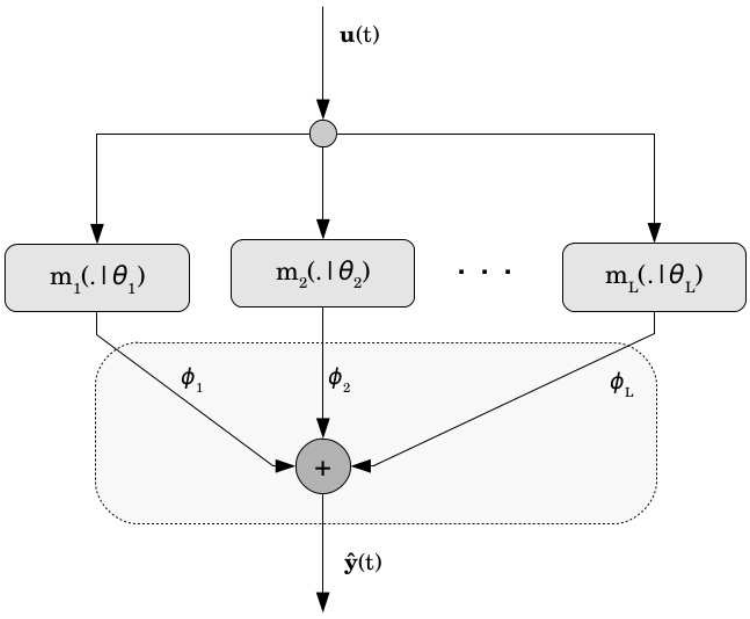

Figure 4.2: *The Scheme of the ensemble model.*

4.2.3 Set of Base-model Operators

In previous section 4.2.2 we expressed the ensemble model operator $M(.|\mathbf{\Theta})$, as a linear combination of some particular parametrized base models $m \in \mathcal{M}$. Each model m

represents "simple" temporal dynamics and we suppose, that by proper optimization of parameters ϕ_i, θ_i the emergence of complicated dynamical behaviour would emerge. Each model m has only one free parameter to make optimization procedure less complicated. In this section I am going to describe the structure of each base model m using the formalism of discrete linear time-invariant dynamical systems. Set $\mathcal{M}$ consists of three models m_1, m_2, m_3 that are referred as integrator, derivator and constant respectively.

Integrator m_1 is parametrized model of a dynamical system that may be described by state-space description as following

$$\begin{aligned} \mathbf{x}_1(t+1) &= \mathbf{x}_1(t) + \theta_1 \mathbf{u}(t) \\ \hat{\mathbf{y}}_1(t) &= \mathbf{x}_1(t) \end{aligned} \tag{4.8}$$

where $\mathbf{x}_1(t)$ is vector of state variables and parameter $\theta_1 \in \mathbb{R}$ is denoted as input gain.

Derivator m_2 has the following form:

$$\hat{\mathbf{y}}_2(t) = \theta_2 \mathbf{u}(t) - \mathbf{u}(t-1) \tag{4.9}$$

where $\theta_2 \in \mathbb{R}$ is the input gain.

Constant m_3 has the following form:

$$\hat{\mathbf{y}}_3(t) = \theta_3 \mathbf{u}(t) \tag{4.10}$$

where $\theta_3 \in \mathbb{R}$ is the input gain.

4.3 The Implementation of Mathematical Model

In this section, the implementation details and algoritmization of mathematical model into the ensemble neural module will be described. This module consists of several simple models interconnected by combination function $f(.)$.

4.3.1 Technical Background

From the technical point of view, the framework of hybrid artificial neural networks is built using Nengoros which is software tool based on combination large-scale neural networks

simulator Nengo[1] and Robotic Operating System (ROS)[2]. Advantage of combination of both systems is "blackbox-like" optimization of artificial neural network topologies provided by Nengo and functionality of explicit modular design of modular components provided by ROS (Vítků, 2012).

Model is implemented as neural module in Nengo scripting system. Scripting is realized in programming language Jython[3], which is implementation of the Python programming language that is designed to run on the Java Platform. This enables to create and combine both Python and Java implementation.

4.3.2 The Framework for Developement of the Ensemble Module

For algoritmization, testing and evaluation of the module the framework was developed. It is implemented as small hybrid artificial neural network in Nengoros. The topology of the framework components is shown on figure 4.3. The framework provides the environment for devolepement of the neural module by simulating the connection of the module into artificial hybrid network and allows the performance evaluation of the ensemble module. The framework consists of several components:

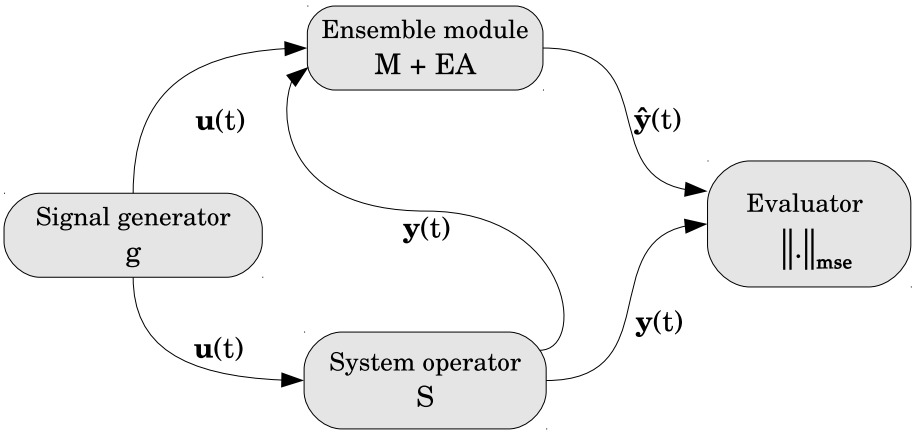

Figure 4.3: *The framework provides the environment for development of the ensemble module*

[1]Nengo is developed by Centre for Theoretical Neuroscience (CTN) at the University of Waterloo, Canada, its homepage is *http://nengo.ca/*

[2]The official project homepage is *http://www.ros.org*

[3]The project homepage is *http://www.jython.org/*

Signal generator $g(.)$

It is a function $g : \mathbb{N} \to \mathbb{R}$, that generates discrete time signal that represents input signal $\mathbf{u}(t)$ generated by other subsystems of hybrid Artificial neural network. Description of role of the signal in frame of the agents architecture is present in previous section 4.1. For module testing, generators of various signals will be used.

Enseble module $M + EA$

This component is described in the following section 4.3.3.

Unknown system operator S

This component represents the response of the enviroment perceived by agent.

Evaluator $\|.\|_{mse}$

This component evaluates performance of operator M by computing mean squared error of signals $\hat{\mathbf{y}}(t)$ and $\mathbf{y}(t)$.

4.3.3 Module Structure

I this section the complementary information to mathematical formulation of model operator M is presented. The structure of the module is shown on figure 4.4. The module consists of the following two main components:

The Evolutionary Algorithm (EA)

It provides optimization of $\mathbf{\Phi}$ and $\mathbf{\Theta}$ by employing $\|.\|_{mse}$ as the optimisation criterion for k past inputs u and y of the model. The parameters are updated in constant rate t_r by generating a new population $\mathcal{P}$ and selecting parameter $\rho \in \mathcal{P}$ with highest fitness.

The Enseble model M

We have described this component in previous sections 4.2 denoted to problem formulation.

This module would be connected into hybrid neural network as shown on figure 4.1.

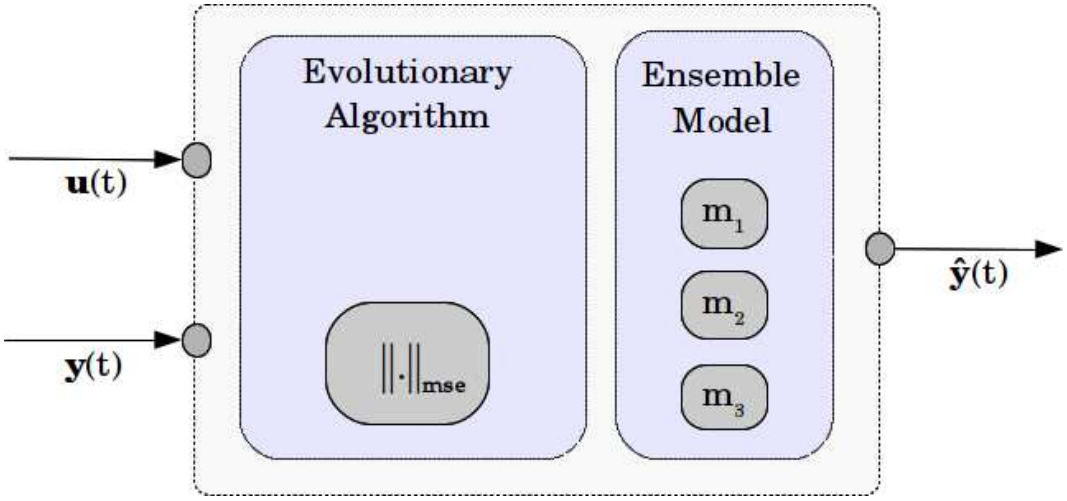

Figure 4.4: *The Structure of the ensemble module.*

4.3.4 How does it work

The general procedure the module is shown on the Algorithm 1. The algorithm updates parameters $\mathbf{\Phi}$ and $\mathbf{\Theta}$ in constant rate t_r. These parameters are stored in $\mathcal{P}$ as a touple $\rho = (\mathbf{\Phi}, \mathbf{\Theta}, fit)$[4]. The evaluation procedure of each parameter ρ[5] in the population is based on k previous values of inputs u, y where $k \in \mathbb{N}$ is constant in the algorithm runtime. The evaluation procedure calculates fitness for each parameter $\rho \in \mathcal{P}$ as:

$$fit = \frac{k}{\sum^{k}(\tilde{y} - y)^2} \tag{4.11}$$

where each $\tilde{y}$ is value of ensemble model $M(.)$ with particular parameter $\rho = (\mathbf{\Phi}, \mathbf{\Theta}, .)$ for particular $u \in U$ and corresponding $y \in Y$. Procedure of evaluation of the parameters set $\mathcal{P}$ is shown as Algorithm 2.

The selection of the pair of parameters (ρ_1, ρ_2) where $\rho_1, \rho_2 \in \mathcal{P}$ is performed in stochastic manner depending on fitness of each parameter ρ. The selection probability of ρ from the $\mathcal{P}$ is proportional to its fitness.

The crossover of the pair (ρ_1, ρ_2) is performed as:

$$min(x_1, x_2) + (r(\delta + 1)) \tag{4.12}$$

[4]If $\mathbf{\Phi}$ or $\mathbf{\Theta}$ is fixed and not to be optimized, it is not stored in ρ.

[5]Each ρ represents population individual in terminology of EA .

where r is random number $r \in \langle 0, 1)$, $\delta = \max(x_1, x_2) - \min(x_1, x_2)$ for each corresponding components x_1, x_2 of (ρ_1, ρ_2). This procedure is similair to uniform crossover.

The mutation update on each component x in ρ consists of performing one of the following processes, each determined its constant probability p_i:

1. Scaling value of x is performed as following $x := xr$ where $r \in (0, c)$ where c is constant during runtime.
2. Inverting its sign as $x := -1x$.
3. Resetting the parameter is performed as $x := 0$ if x is component of $\mathbf{\Phi}$ or $x := 1$ if x is component of $\mathbf{\Theta}$.

After mutation, the offspring is added to population $\mathcal{P}$ and the output $\hat{y}$ is calculated using parameter ρ_{best} with highest fitness.

$\mathcal{P}$ - set of parameters (i.e. population in EA terminology)
U, Y - FIFO queues that store k previous u and y respectively
N_{cross} - number of crossovers, depends on $|\mathcal{P}|$
t_r - parameters update rate
$rand$ - random number from $\langle 0, 1)$
p_{mut} - probability of mutation

initialize $\mathcal{P}$; // initialize population randomly
while *run* **do** // run until interrupted
  $u_t \to U$, $y_t \to Y$; // add input values into queues
  if $t \mod t_r == 0$ **then** // if true, update parameters
    evaluate $\mathcal{P}$; // evaluate each parameter
    for 1 *to* N_{cross} **do**
      (ρ_1, ρ_2) := select pair from $\mathcal{P}$; // select based on fitness
      offspr := perform crossover of (ρ_1, ρ_2) ; // get offspring by crossover
      if $rand < p_{mut}$ **then**
        mutate *offspr*
      end
      offspr $\to \mathcal{P}$; // add offspring into population
    end
    ρ_{best} := get best from $\mathcal{P}$; // get best parameter from population
  end
  $\hat{y}$:= calculate output of M using ρ_{best}
  $t := t + 1$; // increment timestep variable
end

Algorithm 1: General procedure of the ensemble module.

$\mathcal{P}$ - set of parameters (i.e. population in EA terminology)
$\rho = (\mathbf{\Phi}, \mathbf{\Theta}, fit)$ - parameters of $M(.)$ with associated fitness
U, Y - FIFO queues that store k previous u and y respectively
err - mean square error

for $p \in \mathcal{P}$ **do**
  for $y \in Y$, $u \in U$ **do**
    $\tilde{y}$:= evaluate $M(.)$ using $(\mathbf{\Phi}, \mathbf{\Theta})$; `// evaluate ensemble model`
    $err := err + (\tilde{y} - y)^2$; `// update MSE`
  end
  $\rho \leftarrow k/err$; `// associate parametres with new fitness`
  $\rho \rightarrow \mathcal{P}$; `// save parameters associated with fitness`
end

Algorithm 2: Evaluation of the parameters

Chapter 5

Experiments

In order to examine the behaviour of the ensemble module, several experiments were performed. The experiments should indicate capability of the ensemble module to continuously learn the parameters by optimization procedure in order to approximate certain systems. As system operator S a particular neural network is selected and its approximation by employing single base model and ensemble model is compared. As input signal $\mathbf{u}(t)$, sinusoidal and step signal are used.

The system operator S corresponds to spiking neural network in Nengo. The experiments consist of approximation of the relatively small reccurent neural network consisting of 150 neurons. The network consists of population of neurons that are connected to most of the other neurons within the population. The reccurent connections of this neural network store the information stabily over time and allows the network to implement particular temporal integration. The post-synaptic time constant of the neurons is $pstc = 10ms$.

5.1 Approximation by Single Base Model

In this experiment the approximation of the system S is performed only by single base model m_1, the integrator i.e. the values of hyperparameters are fixed to $\phi_1 = 1, \phi_2 = \phi_3 = 0$. The objective is to determine if particular neural network presented above may be approximated by m_1. In next section the same neural network will be approximated by whole ensemble module and the results will be compared. The update rate of the ensemble module is set to $t_r = 20$, the number of previous inputs is set to $k = 1500$, the

simulation is stopped after $t = 3000$ time steps.

5.1.1 Sinusoidal Input Signal

In this case the input $\mathbf{u}(t)$ is sinusoidal signal with angular frequency $\omega = 10\pi$ and amplitude $A = 20$. This input signal is shown on the figure 5.1.

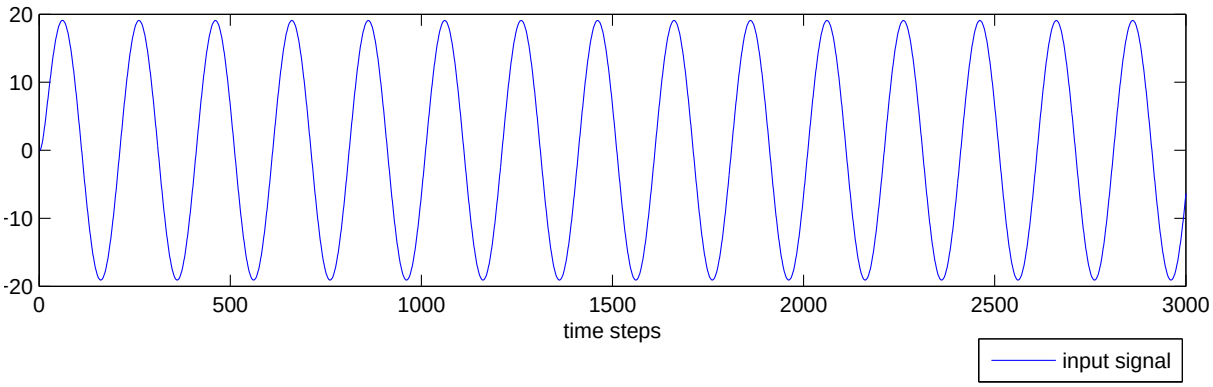

Figure 5.1: *Sinusoidal input signal $u(t)$*

The output of the module $\hat{\mathbf{y}}(t)$ and output of the system $\mathbf{y}(t)$ obtained by simulation are shown on the figure 5.2. The figure shows, that value of the parameter θ_1 stabilize approx. after $t = 750$ time steps.

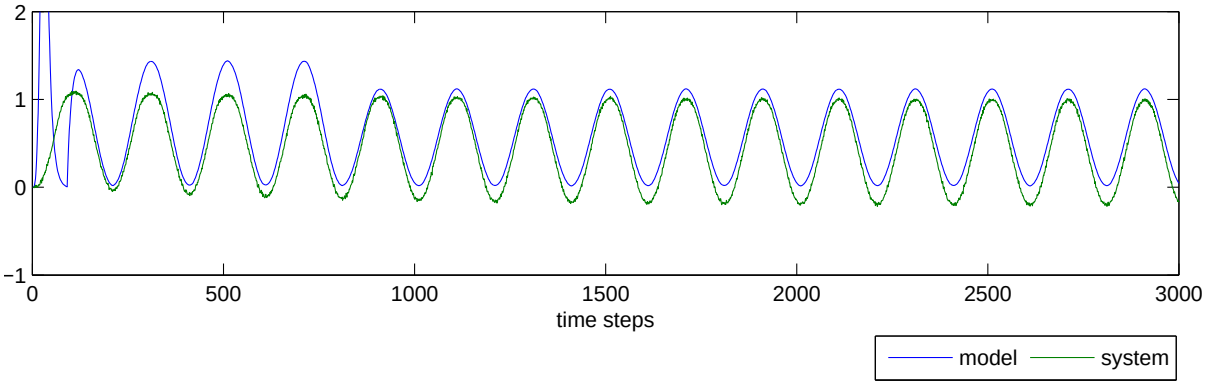

Figure 5.2: *Outputs of model and plant*

The value of mean squared error of the signals $\mathbf{y}$ and $\hat{\mathbf{y}}$ is shown on the figure 5.3. The value of MSE rappidly grows for low values of t until maximum value is reached.

After the peak point the module has captured enough input-output data in order to approximation of the system.

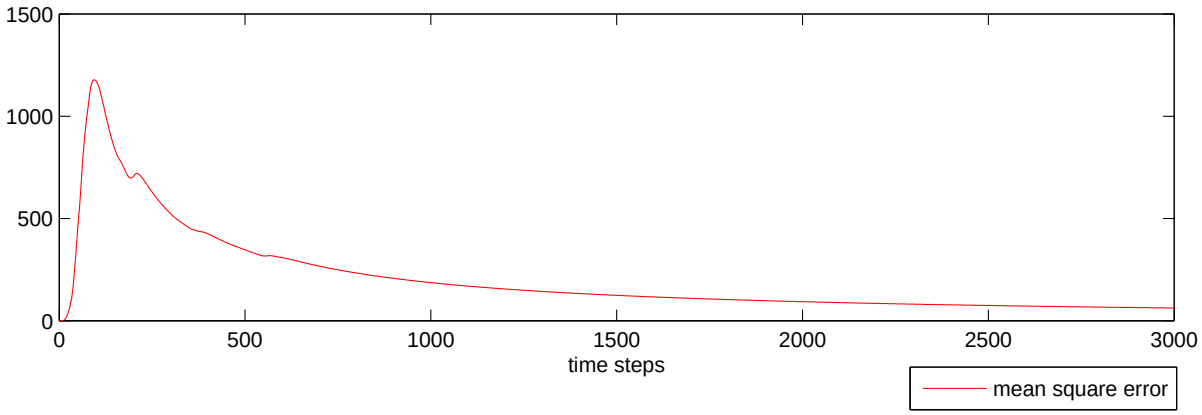

Figure 5.3: *Mean square error of the output signals*

5.1.2 Step Input Signal

This simulation is performed using random periodic step input signal $\mathbf{u}(t)$ as shown on the figure 5.4.

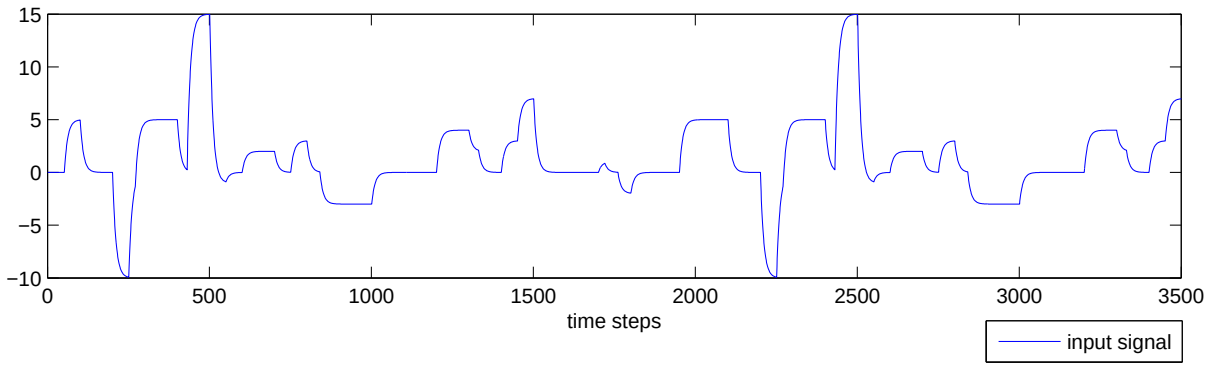

Figure 5.4: *Input signal* $u(t)$

The output signals $\hat{\mathbf{y}}(t)$ and $\mathbf{y}(t)$ obtained by simulation are shown on the figure 5.5. The approximation of the system is for this given input signal is not as accurate and signals do not overlap as in the previous case but breach points of signals are located in equal time steps t. Given step signal $\mathbf{u}(t)$ together with the system S generate less

information for approximation for low t and the module does not converge as fast as in previous case.

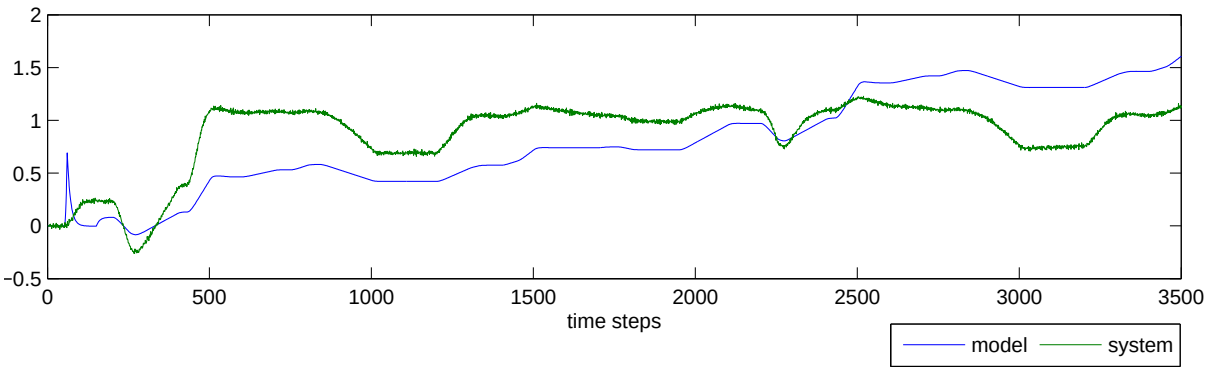

Figure 5.5: *Outputs of model and plant*

The value of mean squared error of the signals $\mathbf{y}$ and $\hat{\mathbf{y}}$ is shown on figure 5.6.

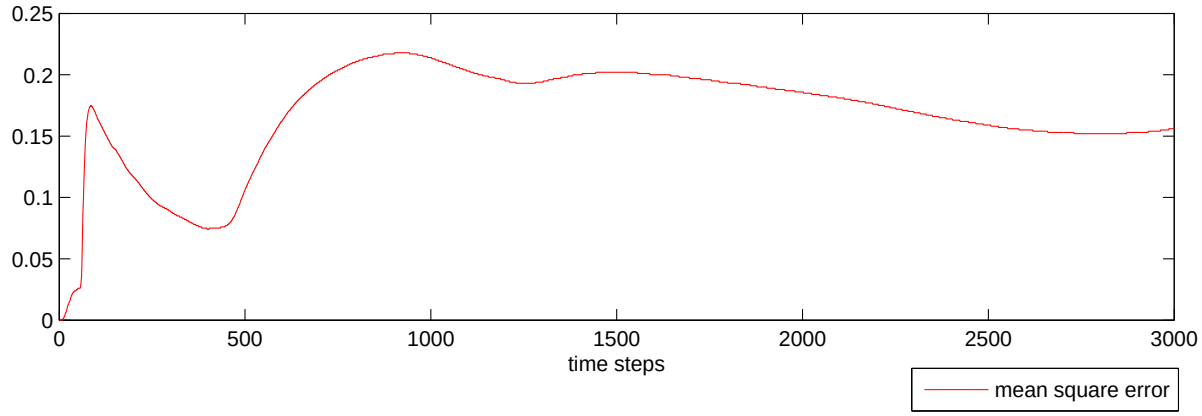

Figure 5.6: *Mean squared error*

5.2 Approximation by the Ensemble Model

In this section the approximation of the same above described system S is investigated. The simulation is run with the hyperparameters $\mathbf{\Phi}$ and model parameters $\mathbf{\Theta}$ that are continuously optimized. The objective is to find out if it is possible to improve the approximation of the given system S by employing other base models.

5.2.1 Sinusoidal Input Signal

The input signal is the same as in experiment described above 5.1.1.

The output of the module $\hat{\mathbf{y}}(t)$ and output of the system $\mathbf{y}(t)$ represented by neural network is shown on figure 5.7. For low t the signal $\hat{\mathbf{y}}(t)$ is characterized by fluctuation, the regularity of $\hat{\mathbf{y}}(t)$ begins to emerge approx. at $t = 1400$ time steps. After sinusoidal pattern is captured by the ensemble module, the parameters of the module are unable to converge further to improve the approximation. The effective change of the optimization rules could resolve this problem.

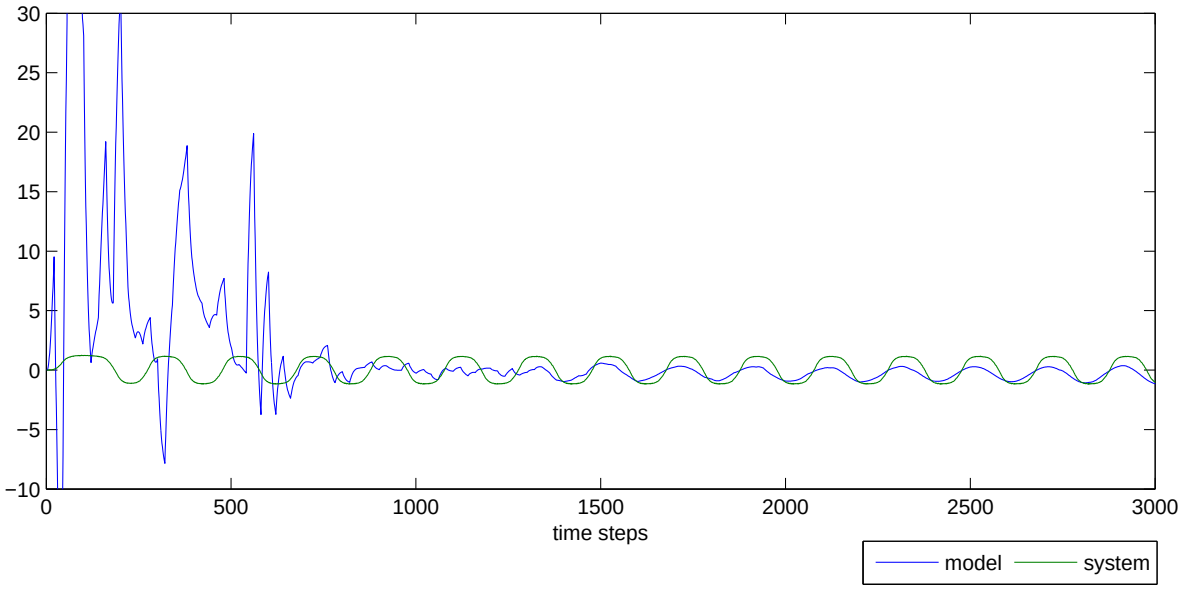

Figure 5.7: *Outputs of model and plant*

The value of mean squared error of the signals $\mathbf{y}$ and $\hat{\mathbf{y}}$ is shown on figure 5.8. The curve has similair shape to curve 5.3 measured for the previous sinusoidal input signal.

5.2.2 Step Input Signal

The input step signal is again identical to the signal employed in the previous experiment 5.1.2.

The output of the module $\hat{\mathbf{y}}(t)$ and output of the system $\mathbf{y}(t)$ shown on figure 5.9 is characterized by irregular fluctuation for low values of t. After the fluctuation peroid, the module adjust its parameters and roughly approximates the system output $\mathbf{y}(t)$. The value of MSE of the signals $\mathbf{y}$ and $\hat{\mathbf{y}}$ is shown on figure 5.10.

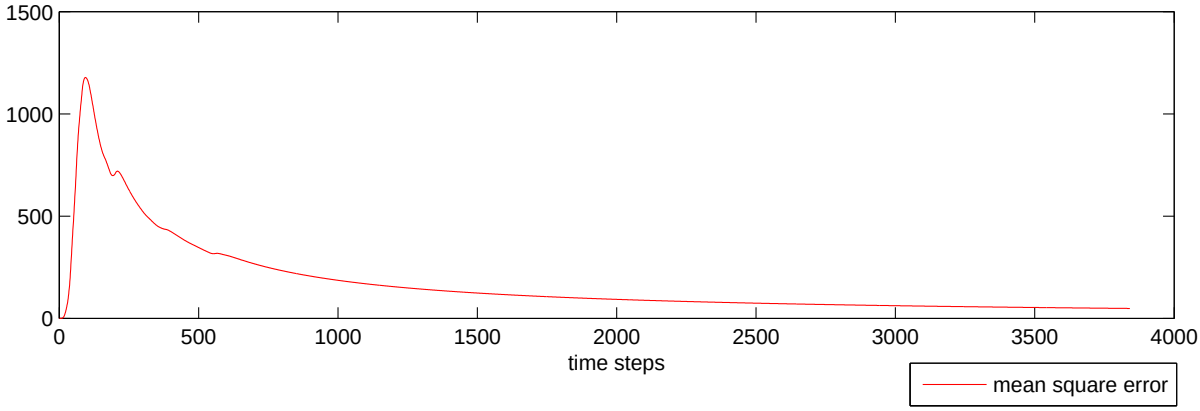

Figure 5.8: *Mean squared error*

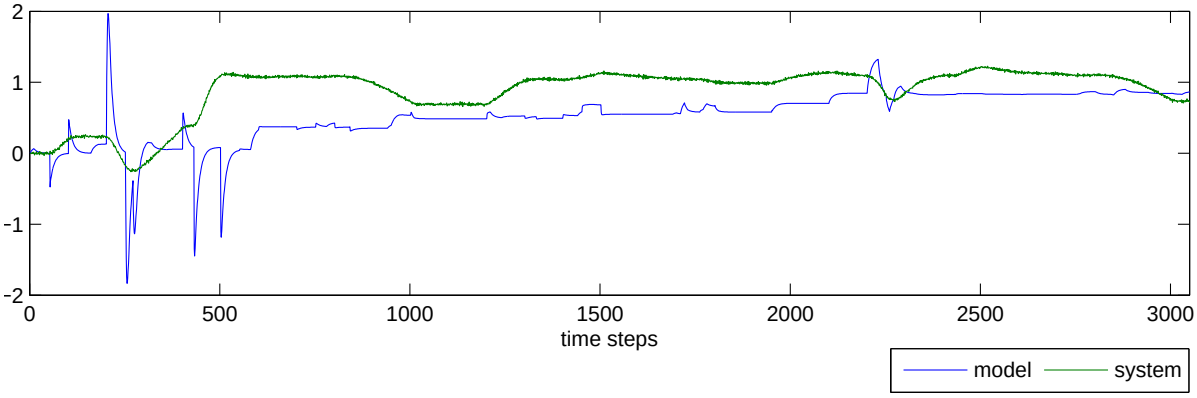

Figure 5.9: *Outputs of model and plant*

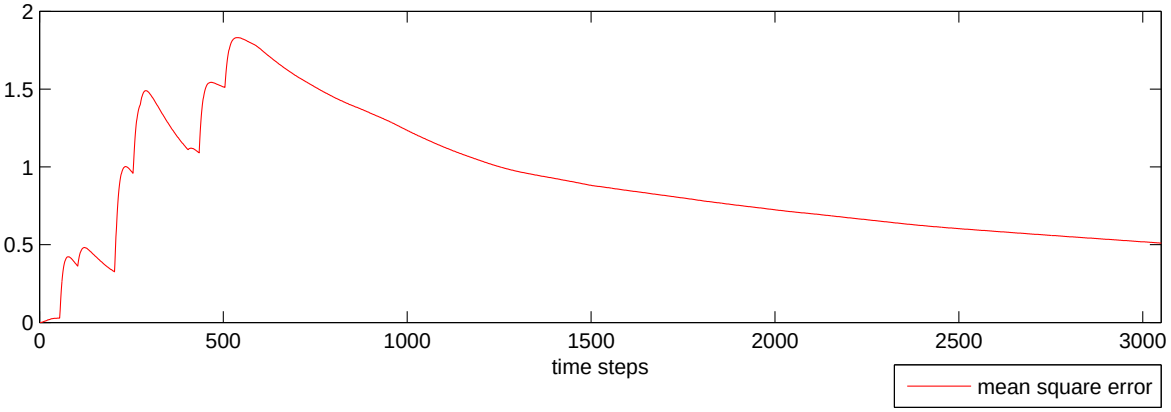

Figure 5.10: *Mean squared error*

Chapter 6

Thesis Contributions and Conclusion

This thesis presents the formal description and implementation of the ensemble module. The ensemble module combines several models of simple dynamical systems and the evolution algorithm. The module has multiple parameters that are altered by evolution algorithm in order to continuously improve the approximation of the blackbox system. The module is connectable into artificial hybrid neural network that represents the architecture of the intelligent autonomous agent. Due to suggestion of my supervisor, I decided to implement the module as node in Nengo API instead of ROS module. This decision was made because of reduntant and complicated communication between ROS nodes. Except this simple modification, the thesis is in full accordance with the thesis assignment. The ensemble module is conectable into framework of hybrid artificial neural network. The ensemble module can not be used to approximate any blackbox system. On the other hand it could be used to continuously approximate certain systems that we have some a priori information. Using this information in order to enhance the rules of the optimization procedure would result in better approximation. The ensemble module could be employed as subsystem in the architecture of the autonomous agent but only to approximate certain temporal dynamics. The future work could include adding more base models of dynamical system and employing a better strategy for combining of models.

Bibliography

A. Terry Bahill, F. S. (2009). Comparison of dynamic system modeling methods, *Systems Engineering, Wiley Periodicals* **Vol 12, No. 3**.

Agoston E. Eiben, a. a. (1994). *Genetic algorithms with multi-parent recombination*, Springer.

Agoston E. Eiben, J. S. (2008). *Introduction to Evolutionary Computing*, Springer.

Alpaydin, E. (2010). *Introduction To Machine Learning, Second Edition*, MIT Press.

Ashlock, D. (2004). *Evolutionary Computation for Modeling and Optimization*, Springer.

Backlund, A. (2000). The definition of system, *Kybernetes* **Vol. 29**: 444 – 451.

Bobal, V. (2009). *Identifikace systemu*, UTB, Zlín.

Boyd, S. (2012). Introduction to linear dynamical systems.
URL: *http://www.stanford.edu/class/ee263/lectures.html*

David Opitz, R. M. (1999). Popular ensemble methods: An empirical study, *Journal of Artifcial Intelligence Research* **11**: 169–198.

Eric B. Baum, Dan Boneh, C. G. (1995). On genetic algorithms, *COLT '95 Proceedings of the eighth annual conference on Computational learning theory* pp. 230–239.

Giorgio Valentini, F. M. (2002). Ensembles of learning machines.

Jiri Roubal, P. H. (2008). *Zaklady regulacni techniky v prikladech*, Praha, CVUT.

Kashiwagi, H. (2009). Nonparametric system identification, *CONTROL SYSTEMS, ROBOTICS, AND AUTOMATION* **VI**: 70–87.

Kittler, J. (1998). Combining classifiers: A theoretical framework, *Pattern Analysis & Applic.* **1**: 18–27.

Ljung, L. (2008). Perspectives on system identification, *Int J Neural Syst* **9**(2): 129–151.

Milan Vrožina, Zora Jančíková, J. D. (2007). Identifikace systÉmŮ.

Noskievic, P. (1999). *Modelovani a identifikace systemu*, Montanex.

Pajonk, O. (2009). *Overview of System Identification with Focus on Inverse Modeling : Literature Review*, Institute of Scientific Computing Carl-Friedrich-Gauß-Fakultat Technische Universitat Braunschweig.

Polikar, R. (2006). Ensemble based systems in decision making, *IEEE CIRCUITS AND SYSTEMS MAGAZINE* .

Priemer, R. (1991). *Introductory Signal Processing*, World Scientific.

Rokach, L. (2009). Ensemble-based classifiers, *Artif Intell Rev* **22**: 1–39.

Rowell, D. (2002). State-space representation of lti systems.
URL: *http://web.mit.edu/2.14/www/Handouts/StateSpace.pdf*

Russell, Stuart J.; Norvig, P. (2010). *Artificial Intelligence: A Modern Approach (3nd ed.)*, Pearson Education.

Streichert, F. (2012). Introduction to evolutionary algorithms.

Tomassini, M. (2012). Evolutionary algorithms.

Vítků, J. (2013). Towards automated design of complex modular systems inspired by nature.

Vítků, J., N. P. (2012). New hybrid decision-making mechanisms in the field of artificial life, *Kognice a umělý život* **XII**(ISBN: 978-80-86742-34-2): pp.254–263.

Włodzimierz Greblicki, M. P. (2008). *Nonparametric system identification*, Cambridge University Press.