VIRTUAL MOBILE ROBOT SOFA-2009
FOR COMPUTATIONAL ROBOTICS RESEARCH

Alexander M. Astapkovitch, PhD

Head of the Student Design Center, Saint-Petersburg State University of Aerospace Instrumentation,
Saint-Petersburg, Russia

Abstracts

The paper presents mathematical model of differential drive mobile robot SOFA-2009, developed as a benchmark model to provide possibility to compare control algorism of different group of researchers.

Model SOFA-2009 is formed with seven ordinary differential equations and includes dynamics and kinematics of the two wheel robot, simple models of wheels, gears and motors.

Example of the using SOFA-2009 model for supervised learning of the neuron net control system is discussed. Sample of the simulations with Mathcad 14 is presented. Robot behavior reaction, described as "rotation for prescribed angle and have a zero angle velocity at final point" is used for explanations of the proposed approach.

I. INTRODUCTION

The main goal of the article is helping to young researches to jump in to Computational Robotics field and namely to neuron net control system leaning.

This work is a part of student research project Phoenix-3 started on April 2008 in State University of Aerospace Instrumentation (Saint-Petersburg) under patronage of Student Design Center. Research activities of Phoenix research group is concerned of "Teaching by Showing" approach for multi channel real time control systems, based on artificial neural net [1 – 3].

History of student design project set Phoenix-X is reflected with fig. 1. Projects had designed by different student teams as a part of activity of Student Design Center of State University of Aerospace Instrumentation (Saint-Petersburg). It can be seen from fig. 1 that there are two lines in Phoenix-X activity: real robot designing and virtual reality robots.

During Phoenix-1 phase one step learning procedure was tested. The procedure used least square method in combination with Tickhnov regularization procedure [4] of cost function to get a correct solution for weight set for neural control net with one step learning procedure

$$\mathbf{w} = (\mathbf{S}^T\,\mathbf{S} + \gamma\,\mathbf{E})^{-1}\,\mathbf{S}^T\,\mathbf{Ua}, \qquad (1)$$

where **w** – is the neutron net weight vector for given control channel , **S –** is rectangular matrix of inputs, formed from sensor measurements, **Ua -** is vector of actor signals for a given control channel, formed by operator during showing phase and γ is regularization coefficient.

This approach was implemented for fast synthesis of neuron net controller for following the white stripe mission with real robot in real environment (robot Phoenix-1 and Phoenix-2 projects) [1, 2].

Because mobile robots are complex electromechanical devices the real experiments are very expensive in sense of time and stuff. Due to this reason during Phoenix-2 project a virtual environment and simplified robot model were developed to generate on board cameras images.

Phoenix-3 project is the third step of Phoenix student research group that consists from the hardware development and the background theory activities. In accordance with the project legend autonomous robot Phoenix-3 has to patrol the yard looking for open fire. If one will be found robot has to stop flame with on board equipment. Still, the strategic goal is a developing of understanding of the learning strategy for the control system on base of a neuron net.

There are two branches of the Phoenix-3 project, namely, robot development and mathematical simulation. Mathematical simulation has as the result mathematical model SOFA-2009, that pretend to be a benchmark model for different research group.

SOFA-2009 is as simple as possible model of two wheel differential drive robot. The model has the most significant features of the real robot. Phoenix-3 was used as prototype. Model is formed with seven ordinary differential equations and includes dynamics and kinematics of the robot, simple models of wheels, gears and motors.

This paper concerns of the mathematical simulation of control algorithm and learning methodology of control system on base of neuron net. One step learning methodology was tested for virtual mobile robot SOFA-2009 with Mathcad 14 and brief description is presented in this article.

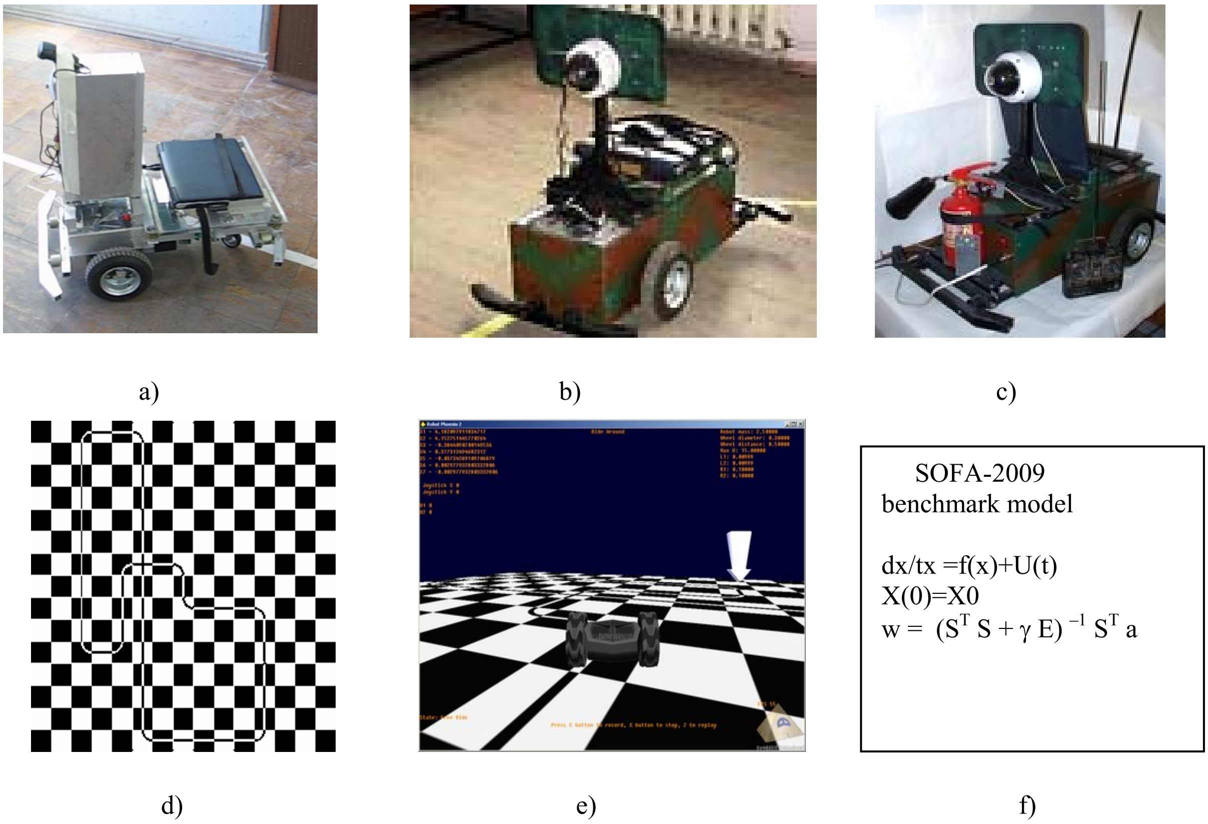


Fig. 1. Robot Phoenix-X family
a) Robot Phoenix-1, b) Robot Phoenix-2, c) Robot Phoenix-2,
d) Virtual environment, e) Virtual robot Phoenix-2V, e) Model SOFA-2009.

II. VIRTUAL ROBOT SOFA – 2009

Main goal of SOFA is introduction to robotic society the benchmark model than can be used to investigate neuron net control system learning.

The model has to be as simple as possible and of autonomous differential drive wheel robot is used as legend. In this part project PHOENIX-3 activity is close to research of group that uses model of the LEGO differential drive robot [5]. To save resources this group uses a simplified robot and PID based controller. The mathematical model LEGO robot was developed that was verified on real LEGO robot behavior.

SOFA is a virtual two wheels differential drive mobile robot. Every wheel is a rigid body disk with diameter $D_{w.}$ Distance between wheels is L_r and motors, gears, battery and control system are inside this space. Each wheel is driven independently with own motor. Motors have identical electric and mechanical parameters.

Forward/back motion is produced by both wheels been driven at the same rate. Turning right is achieved by driving the left wheel (wheel number 2) at a higher rate the right wheel (wheel number 1) and vice versa for turning left.

SOFA kinematics model is presented in fig. 2. At any given moment of the time SOFA moving is an arc with center at the $R_0(t)$. The position of the robot center is described with vector R_c (t) and angle φ(t), that is angle displacement of robot axe from initial inclement from axe x.

Robot movement is described with three ordinary differential equations concluded from basic assumption of SOFA model:

$$\varphi(t+\Delta t) - \varphi(t) = \Delta\varphi \text{ (for left wheel)} =$$
$$=\varphi \text{ (for right wheel)} \qquad (1)$$
$$R_0\,(t) = R_0\,(t+\Delta t) \qquad (2)$$

Whole model is described with system of 7 ordinary differential equations (3) – (9).

Gear for every wheel is described with equations (6) and (7) with free parameters – k_{12}, k_{21}. Model takes into account moment of inertia of wheel J_r. Coefficients k_{11}, k_{21} were introduced to provide flexibility to SOFA model.

Two channel control system and robot motors are described with equations (8) – (9), that are the same as in the work [5]. Motor currents are controlled with voltages $U_1(t)$ and $U_2(t)$ and affected by motor resistance Rm, motor inductance Lm and wheel angle velocity w_1 (w_2). Model implements the limit on robot accumulator voltage Vmax. It is supposed also that every motor is controlled with bridges and can rotate to forward-back direction.

Control system model depends on research task. In hand made control (operator control) the voltages graphs has to be described and implemented to model.

$$\begin{cases} \left.\dfrac{dR_c}{dt}\right|_x = \dfrac{D_w(\omega_1+\omega_2)}{4}\left[-\sin\varphi\right] & (3)\\ \left.\dfrac{dR_c}{dt}\right|_y = \dfrac{D_w(\omega_1+\omega_2)}{4}\left[\cos\varphi\right] & (4)\\ \dfrac{d\varphi}{dt} = \dfrac{D_w}{2Lr}\cdot(\omega_2-\omega_1) & (5)\\ \dfrac{d\omega_1}{dt} = \dfrac{-k_{11}}{J_r}\omega_1 + \dfrac{k_{12}}{J_r}I_1 & (6)\\ \dfrac{d\omega_2}{dt} = \dfrac{-k_{22}}{J_r}\omega_2 + \dfrac{k_{21}}{J_r}I_2 & (7)\\ \dfrac{dI_1}{dt} = -\dfrac{R_m}{L_m}I_1 - \dfrac{k_{13}}{L_m}\omega_1 + \dfrac{1}{L_m}U_1(t) & (8)\\ \dfrac{dI_2}{dt} = -\dfrac{R_m}{L_m}I_2 - \dfrac{k_{23}}{L_m}\omega_2 + \dfrac{1}{L_m}U_2(t), & (9) \end{cases}$$

To simulate robot movement Cauchy problem can be formulated for (3) – (9). In vector form system (3) – (9) can be rewritten as

dX(t)/dt = f(x) + U(t,x), (10)

X(0) =X0, (11)

where vector equation (10) presents ODE (3) – (9) and (11) is a initial vector value.

Model SOFA-2009 is defined with parameters set

$D_r = 0.3$; $L_r = 0.5$; $J_r = 0.25$;
$k_{11} = k_{22} = 75$; $k_{12} = k_{21} = 10$;
Rm = 0.1; Lm = 0.01; $k_{13} = k_{23} = 1.5$;
Vmax=12. (12)

Vmax is the maximal absolute value for accumulator voltage.

SOFA-2009 model was investigated with Mathcad 14. The ODE solver function Rkadapt (.,.,) was used for robot dynamic simulation. It consists from two examples: motion directly forward and turn around left wheel.

To move forward a maximal voltage VMAX=12 V was applied to left and right robot motors. To rotate to left voltage VMAX=12 V was applied just to right robot motor.

Results of the test simulation are presented with fig. 3. As it can be dedicated from presented graphs, SOFA-2009 dynamics features:

- robot can move forward with maximal speed as 0.8 m/sec;
- robot can rotate around one wheel with angular maximal speed 1/4 π rad/sec.

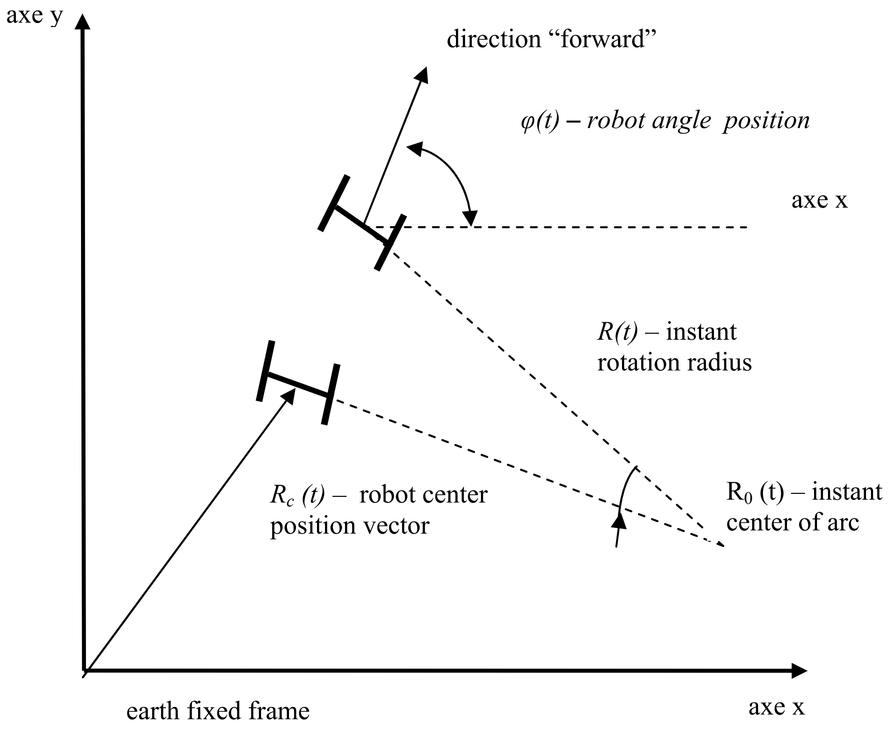
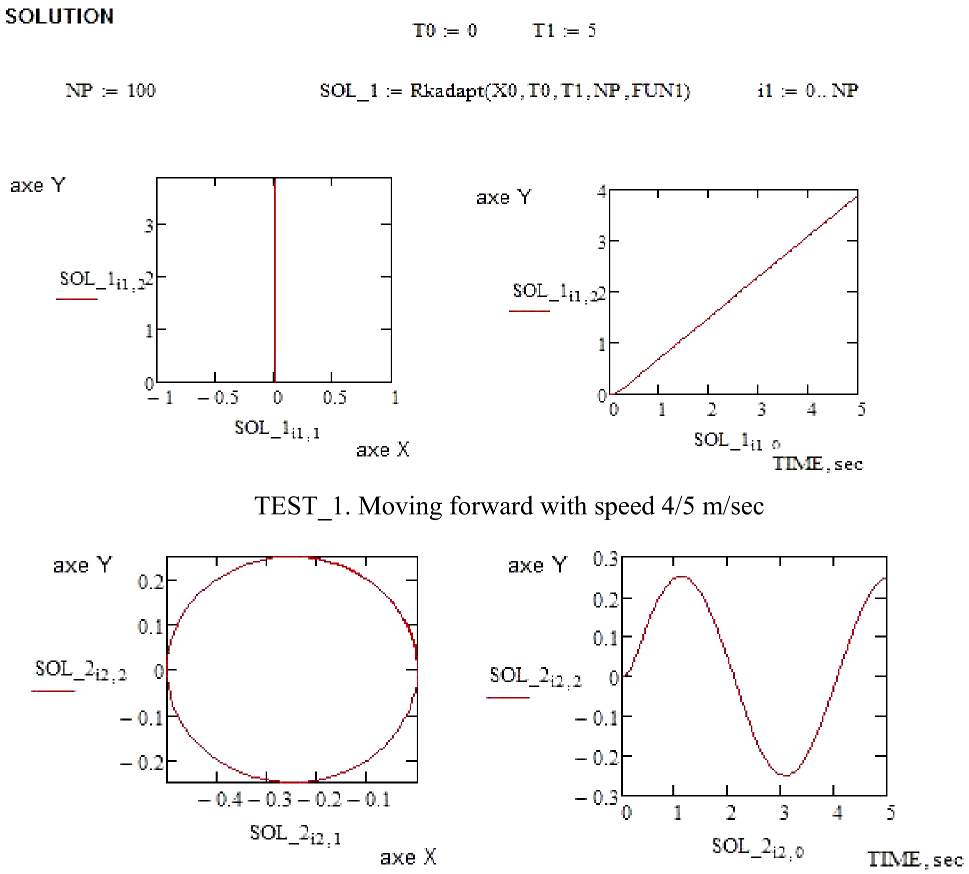

axe y
direction “forward”
φ(t) – robot angle position
axe x
R(t) – instant rotation radius
R_c (t) – robot center position vector
R_0 (t) – instant center of arc
earth fixed frame
axe x

Fig. 2. Kinematics and dynamics models of virtual robot SOFA



```
SOLUTION
                                   T0 := 0        T1 := 5

     NP := 100            SOL_1 := Rkadapt(X0, T0, T1, NP, FUN1)        i1 := 0.. NP
```

TEST_1. Moving forward with speed 4/5 m/sec

TEST 2. Rotation around left wheel with ang. speed π/4 r/s

Fig. 3 SOFA-2009 model testing

III. NEURON NET CONTROL SYSTEM FOR ROBOT SOFA-2009

As it was pointed above, SOFA-2009 model was developed as the part of the student research project Phoenix-3. This model now is using for researching of the "supervised learning" approach to robot control system. Mathcad 14 was used as a tool due to it's friendly and fast user interface, that is a very important for educational purposes.

Mathcad 14 code is presented in Appendix 1. This example is a simple enough for the understanding of the proposed approach.

Robot behavior reaction, described as "rotation for prescribed angle and have a zero angle velocity at final point" will be used below for explanation.

Total scheme of the developed Mathcad Code is presented in fig. 3. Experiment with virtual robot includes at least three steps:

– sample generating and neuron net control system learning ;

– simulation of the robot dynamics with "learned "neuron net control system;

– research experiments.

The same robot model, namely SOFA-2009 was used for all three steps. It has to be outlined out that different control systems are used during first two steps. The Coushy problem solver function Rkadapt (.,.,..) was used to simulate robot movement and get solution table.

During sample generating control voltage graphs for every motor, that will be used for learning, has to be known. This graphs researcher has to dedicate from his experience and solved problem. In our case sample "rotate to left on π/4 in place and zero velocity at final position" was developed to estimates weights for left and right motor neuron reaction. To reach this behavior the left motor and the right one has to rotate in opposite directions. For as large angle as π/4 it is reasonable to use maximum voltage VMAX. Different signs voltages have to be implied to every motor simultaneously. To reach the goal the pulse length parameters were dedicated by the preliminary set of the simulations. This learning sample is presented in fig. 6

In details, one step learning procedure for control system on base of neural net was described [2]. Briefly, supervised learning paradigm can be interpreted as the nonlinear curve fitting problem described as system of linear equation with rectangular matrix

$$\left|\begin{matrix} \mathbf{S_1(T_1)\ S_2(T_1)\ ..\ S_n(T_1)} \\ \mathbf{S_1(T_2)\ S_2(T_2)\ ..\ S_n(T_2)} \\ \\ \mathbf{S_1(T_p)\ S_2(T_p)\ ..\ S_n(T_p)} \end{matrix}\right| * \left|\begin{matrix} \mathbf{w_1} \\ \mathbf{w_2} \\ \\ \mathbf{w_n} \end{matrix}\right| = \left|\begin{matrix} \mathbf{A_1(T_1)} \\ \mathbf{A_1(T_2)} \\ \\ \mathbf{A_1(T_p)} \end{matrix}\right| \quad \mathbf{(13)}$$

$$w = (S^TS+\gamma\ E)^{-1}\ S^T\ a,$$

where $S_i(T_k)$ – value of i-sensor during T_k moment of time, $A_m(T_k)$ – value of control voltage on m actor during T_k moment of time, w_i – weight of the i-sensor.

In our case two channel control system is used and system (13) has to be solved twice to get set of weights for left motor and right one control neuron. To get stable (correct) solution Tikhonov regularization method, has to be used. This procedure can be interpreted as a linear programming problem of the cost function

$$\min_{w} \mathbf{F(w) = (Sw - Ua, Sw - Ua)} + \gamma\ \mathbf{(w,w)}, \quad (14)$$

with independent variable vector **w.**

This problem has analytical solution expressed and weight vector for every channel control neuron can be calculated with

$$\mathbf{w = (S^T\ S + \gamma\ E)^{-1}\ S^T\ Ua} \qquad \mathbf{(15),}$$

where **Nsen** – number of the sensors, **Np** – number of the time discretization pointing in learning sample, **w** – weight vector with dimension **Nsen**, **S –** rectangular **Nsen*Np** matrix of the sensor measurements, **Ua – a Np** vector of actor signals, formed from learning sample,γ is the regularization coefficient and a **E** – square unity matrix **Nsen*Nsen.**

Example of Mathcad 14 code is presented in Appendix 1. In thepresented example just three sensor are used: sensor difference between goal and instant angle, angle velocity sensor and sensor of the difference between angle velocity at final position and instant angle velocity. The same sensor measurement set at any given time slice are used by the both control channel to calculate voltage. System is symmetrical and the weights for different control channels have to be the same, except the sign, that can be used as an additional test.

Structure of control system neutron net is rather simple and presented in fig. 5. Some simulation results from learning example test are presented in fig. 6. In this example neuron net control system was trained to rotate on π/4 in place. As the results weights vectors are

- for left channel $W^t = [\ 17.535\ \ 0.5, -0.5]$;
- for right channel $W^t = [\ -17.535 - 0.5, 0.5]$.

For better understanding some remarks have to be made. In our case second and third sensor are the same due to sample is supposed that final angular velocity is zero. So, weights for this sensor have to be equal due to used regularization procedure. Accept this, due to symmetry of control channels and selected learning sample weight vector for left and right channel differs just with sign.

Fig. 6,c) presents results of the check of the training results. Sensor matrix from sample is multiplayed by weight vector for every channel. It can be seen that voltages from neuron net controller is like to sample. But also, it can be seen, that neuron control net can exceed maximal accumulator voltage.

To limit maximal output voltage actor layer neuron was introduced to system. This neuron limits output voltage. Two parameters Vmax and Vmin are used for this purposes. Fig. 6,d) presents simulation of robot under control of the neuron net without limits on voltage and with limits VMAX=12V, VLOW=1 V.

But the most interest is the estimation of the quality of control of learned control system for final angle much exceeds that one, used for learning. Fig. 7 presents results of the simulation for turn left on angle **π** for the same as for learning sample initial position and final velocity vectors. Two cases were simulated – with or without limits on voltage.

As it can be seen, the learned neuron system solves the task for both cases and robot reaches the prescribed final point in phase plane. For case with imposed limits on maximal used voltage rotation is accelerated during initial phase and is slowed down near the final point. The most part of the trajectory robot rotates with maximal speed.

Some important and not trivial conclusion can be made from this simulation. Sample control voltage did not exceed the Vmax, but this fact is not reflected in the parameters of learned control system. The reason is that the approximation problem was formulated as linear programming problem without any limits on variables.

To take into account such type implicit limits approximation task has to be formulated as linear programming problem with limits on variables and Lagrange method can be used to get a solution in form of one step learning procedure. For more complex models and explicit limits this problem is open and need further investigation.

IV. CONCLUSION

Presented results are just a part of the student research project Phoenix-3. More details about whole project activities and Mathcade 14 code examples with SOFA-2009 model are presented in site guap.ru > student design center > project Phonix-3 and can be used free any interested person.

REFERENCES

[1] Astapkovitch A.M. One step learning procedure for neural net control system. Proc. International forum “Information systems. Problems, perspectives, innovation approaches” , Vol.2, p.3-9, SUAI, Saint-Petersburg, 2007.

[2] Д.А.Астапкович, А.А.Гончаров, А.С.Дмитриев, А.В.Михеев. Проект “Робот Феникс-1”. Сб. докладов пятьдесят девятой международной студенческой научно-технической конференции ГУАП., Ч. I, Технические науки, 18-22 апреля 2006, г. Санкт-Петербург, с.211-216.

[3] Industrial aplication of neural networks , edited by Lakhmi C.Jain, V.Rao Vemuri CRC Press LLC, Boca Raton, London, New York, Washington, D.C ,1999.

[4] Тихонов А.Н, Арсенин В.Я. Методы решения некорректных задач, М., Наука, главная редакция физико-математической литературы, 1979, с. 286.

[5] Kevin J Worrall, Euan W McGookin A mathematical model of a LEGO differential drive robot. Department of Electronics and Electrical Engineering University of Glasgow Department of Aerospace Engineering University of Glasgow

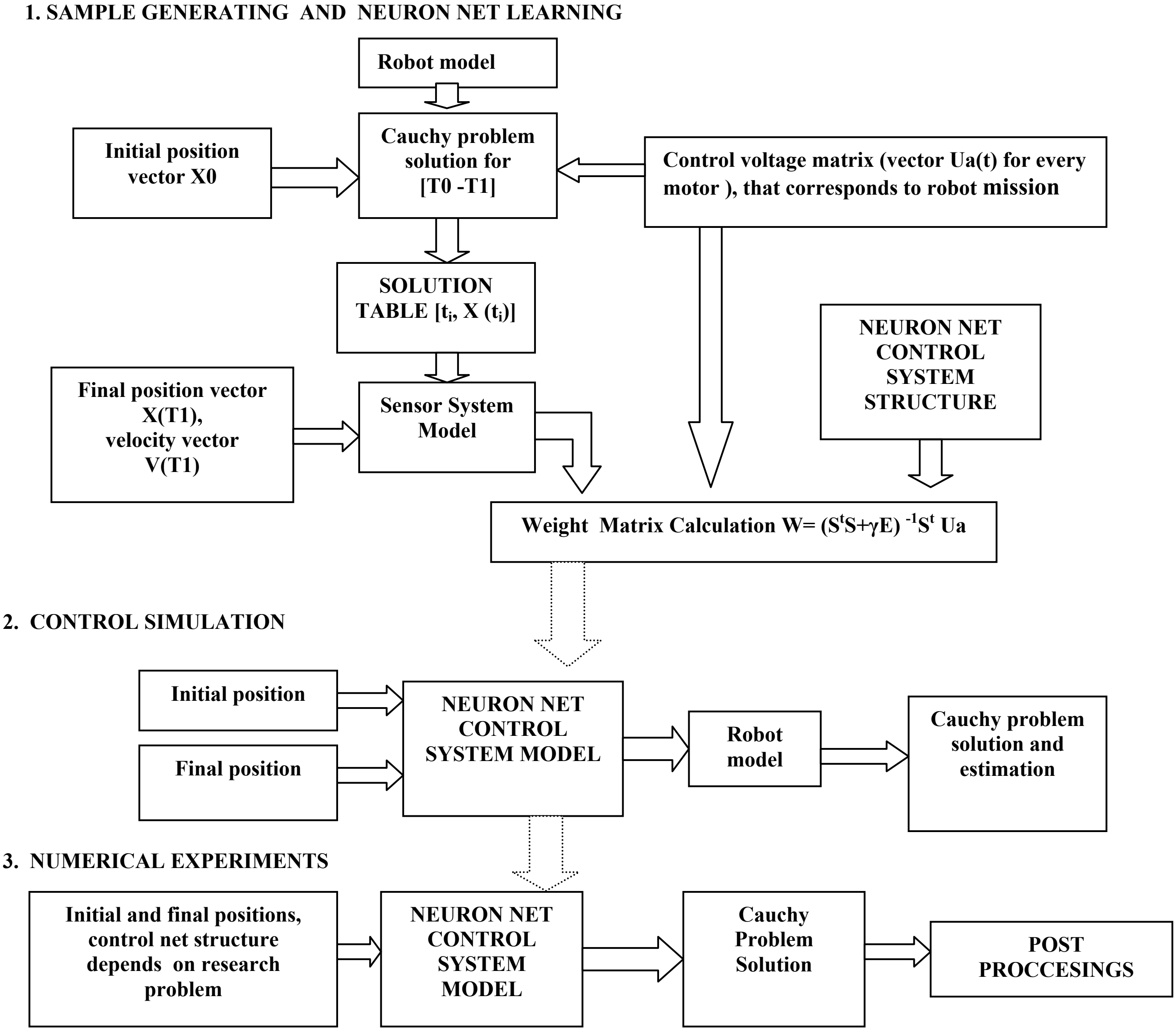

1. SAMPLE GENERATING AND NEURON NET LEARNING
Robot model
Initial position vector X0
Cauchy problem solution for [T0 -T1]
Control voltage matrix (vector Ua(t) for every motor), that corresponds to robot mission
SOLUTION TABLE [t$_i$, X (t$_i$)]
NEURON NET CONTROL SYSTEM STRUCTURE
Final position vector X(T1), velocity vector V(T1)
Sensor System Model
Weight Matrix Calculation W= (S^{t}S+γE) $^{-1}$S^t Ua
2. CONTROL SIMULATION
Initial position
Final position
NEURON NET CONTROL SYSTEM MODEL
Robot model
Cauchy problem solution and estimation
3. NUMERICAL EXPERIMENTS
Initial and final positions, control net structure depends on research problem
NEURON NET CONTROL SYSTEM MODEL
Cauchy Problem Solution
POST PROCCESINGS

Fig. 4. Suprvised learning methodology

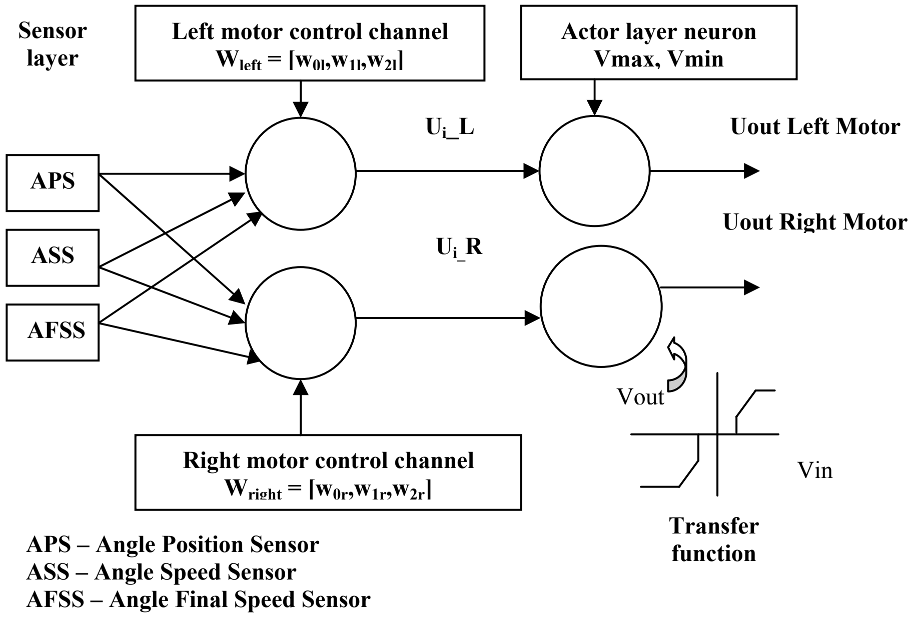

Sensor layer
Left motor control channel W$_{left}$ = [w$_{0l}$,w$_{1l}$,w$_{2l}$]
Actor layer neuron Vmax, Vmin
U$_i$_L
Uout Left Motor
APS
ASS
AFSS
Uout Right Motor
U$_i$_R
Vout
Vin
Right motor control channel W$_{right}$ = [w$_{0r}$,w$_{1r}$,w$_{2r}$]
Transfer function
APS – Angle Position Sensor
ASS – Angle Speed Sensor
AFSS – Angle Final Speed Sensor

Fig.5. Neuron net structure

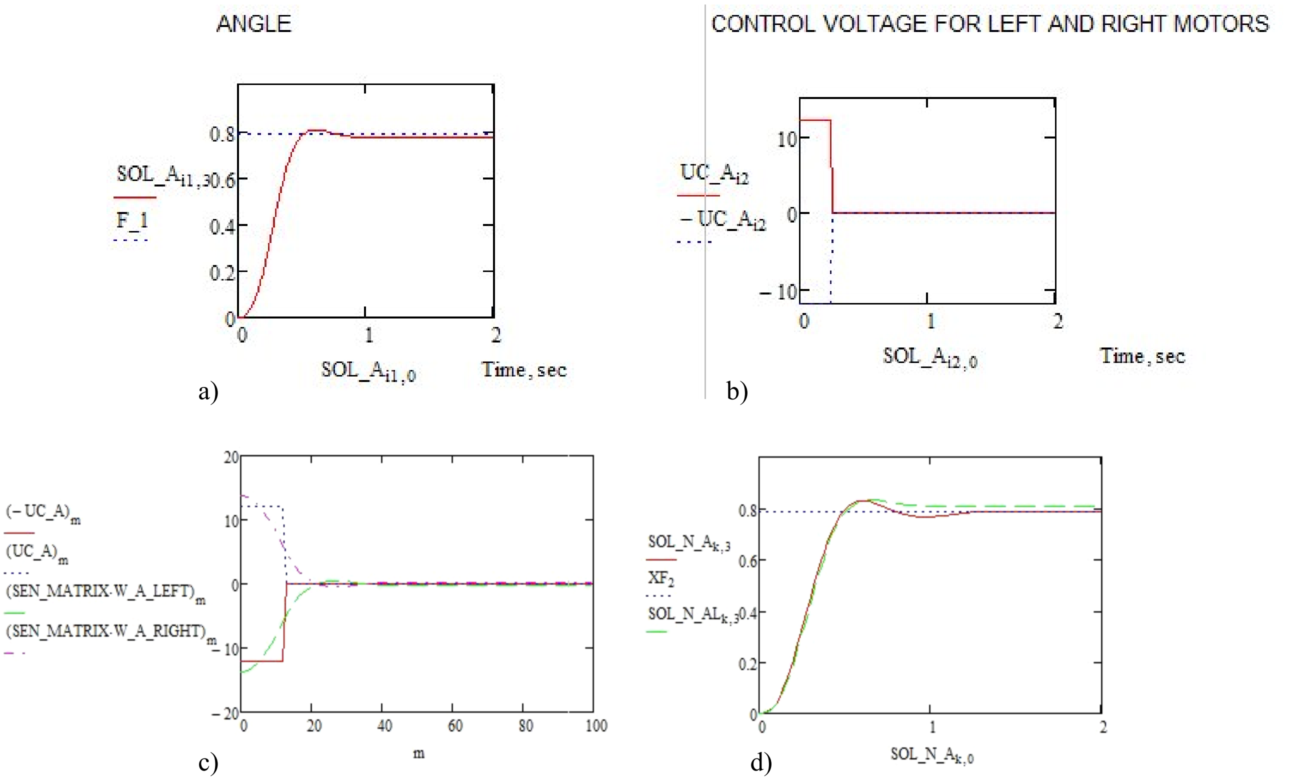


Fig. 6. Example of the supervised learning for π/4 rotate to left sample
a) sample rotation to left with maximal velocity; b) learning sample of control voltages for left and right motors; c) preliminary estimation of the training; d) estimation of the robot dynamics under neuron net control system for π/4 rotate to left for limited and unlimited actor voltage

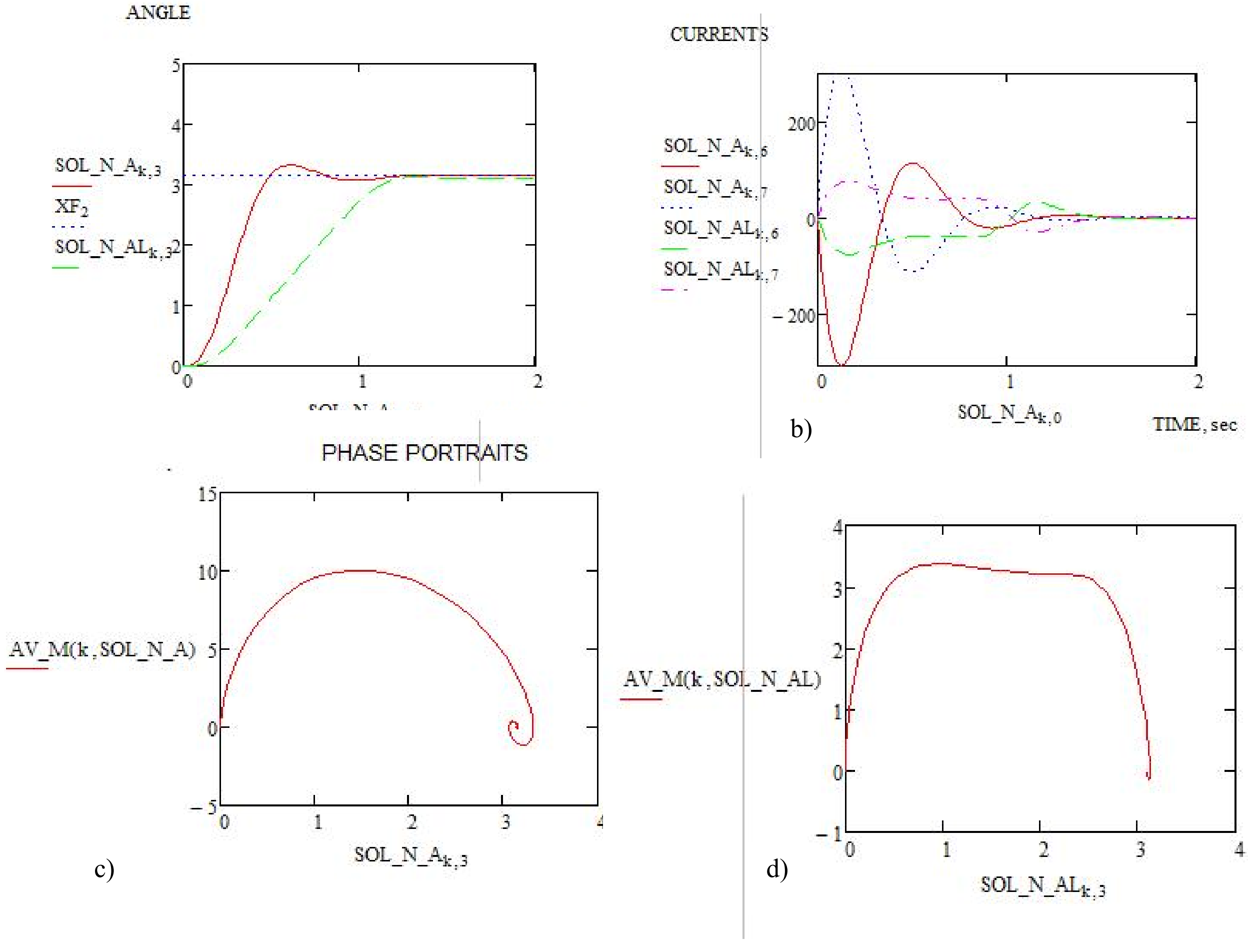


Fig. 7. Sample of autonomous operation for π rotate task
a) rotation to left with limited and unlimited Vmax; b) motor currents for limited and unlimited voltage;
c) phase portrait for unlimited case: start point (0,0),final (3.14,0); d) phase portrait for limited Vmax: start point (0,0),final (3.14,0)

VMR SOFA _2009 (Virtual Mobile Robot)

Problem description

SOFA-2009 PARAMETER LIST

Wheel diameter	$Dw := 0.3$
Distance between wheels	$Lr := 0.5$
Moment of inertia	$Jr := 25$
Left wheel gear parameters	$K11 := 75 \quad K12 := 10 \quad K13 := 1.5$
Right wheel gear parameters	$K22 := K11 \quad K21 := K12 \quad K23 := K13$
Motor resistance	$Rm := 0.1$
Motor inductance	$Lm := 0.01$
Maximal accumulator voltage	$VMAX := 12$

ROBOT INITIAL POSITION (VECTOR X0)

$$X0 := \begin{pmatrix} 0\\0\\0\\0\\0\\0\\0 \end{pmatrix}$$

ROBOT FINAL POSITION (VECTOR XF)

$$XF := \begin{pmatrix} 0\\0\\ \frac{\pi}{4}\\0\\0\\0\\0 \end{pmatrix}$$

ROBOT FINAL POSITION VELOSITY (VECTOR VF)

$$VF := \begin{pmatrix} 0\\0\\0\\0\\0\\0\\0 \end{pmatrix}$$

$F\_1 := XF_2$

Supervised learning to rotate with maximal angular speed

SAMPLE_1. **Rotation to left with maximal speed**

$$FUN(T,X) := \begin{bmatrix} \frac{(X_3+X_4)}{2}\cdot\frac{Dw}{2}\cdot(-\sin(X_2)) \\ \frac{(X_3+X_4)}{2}\cdot\frac{Dw}{2}\cdot(\cos(X_2)) \\ \frac{Dw}{2\cdot Lr}\cdot(X_4-X_3) \\ \frac{-K11}{Jr}\cdot X_3+\frac{K12}{Jr}\cdot X_5 \\ \frac{-K22}{Jr}\cdot X_4+\frac{K21}{Jr}\cdot X_6 \\ \frac{-Rm}{Lm}\cdot X_5-\frac{X_3}{Lm}\cdot K13 \\ \frac{-Rm}{Lm}\cdot X_6-\frac{X_4}{Lm}\cdot K23 \end{bmatrix}$$

"OPERATOR " CONTROL VOLTAGE

$D(X1,X2) := X1 - X2 \qquad K\_A := 0.375 \qquad EA := 0.01$

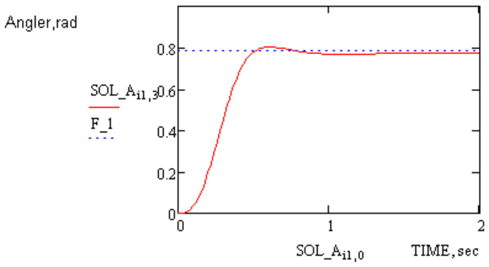
$UA\_A(F0,F,F1) := if(|D(F1,F)| < EA,0,if(|D(F,F0)| < |D(F1,F0)|\cdot K\_A,VMAX,0))$

$$U1(T,X) := \begin{pmatrix} 0\\0\\0\\0\\0\\ \frac{-UA\_A(X0_2,X_2,XF_2)}{Lm} \\ \frac{UA\_A(X0_2,X_2,XF_2)}{Lm} \end{pmatrix}$$

$FUN1(T,X) := FUN(T,X) + U1(T,X)$

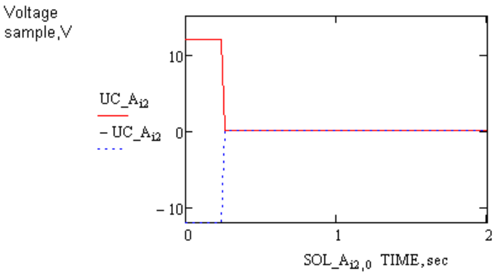
SOLUTION $\quad T0 := 0 \quad T1 := 2 \quad NP := 100 \qquad SOL\_A := Rkadapt(X0,T0,T1,NP,FUN1)$

$i1 := 0..NP$



CONTROL VOLTAGE SAMPLE FOR LEFT AND RIGHT MOTORS

$i2 := 0..NP \qquad UC\_A_{i2} := UA\_A(X0_2,SOL\_A_{i2,3},XF_2)$



ONE STEP LEARNING PROCEDURE

SENSOR MODELS	A -DISTANCE TO FINAL POSITION	INSTANT ANGEL_VELOSITY	DIFFERENCE BETWEEN INSTANT AND ANGLER _VELOCY AT FINAL POINT
MATRIX FORM	$DAF\_M(m,SOL) := (XF_2 - SOL_{m,3})$	$AV\_M(m,SOL) := \frac{Dw}{2\cdot Lr}\cdot(SOL_{m,5} - SOL_{m,4})$	$DVAF\_M(m,SOL) := VF_2 - AV\_M(m,SOL)$
VECTOR FORM	$DAF\_V(X,XF) := (XF_2 - X_2)$	$AV\_V(X) := \frac{Dw}{2\cdot Lr}\cdot(X_4 - X_3)$	$DVAF\_V(X,VF) := VF_2 - AV\_V(X)$

SENSOR MATRIX CALCULATION NUMBER OF SENSORS $NSEN := 3$

	Angler_to_final_ point	**Angler_velosity**	**Diff_for_final Angler_velosity**
$m := 0..NP$	$SEN\_MATRIX_{m,0} := DAF\_M(m,SOL\_A)$	$SEN\_MATRIX_{m,1} := AV\_M(m,SOL\_A)$	$SEN\_MATRIX_{m,2} := DVAF\_M(m,SOL\_A)$

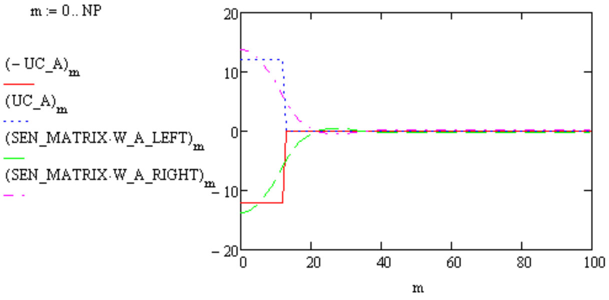
UNITY matrix $\quad k := 0..NSEN - 1 \quad j := 0..NSEN - 1 \quad E_{j,k} := 0 \quad E_{k,k} := 1$

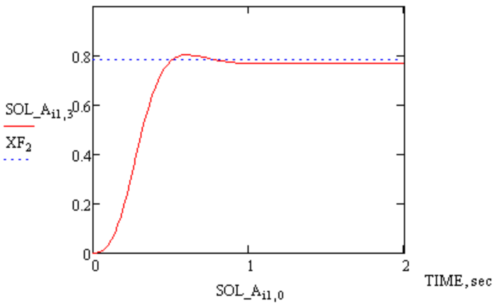
NEURON NETWORK WEIGHTS CALCULATION

$W\_A\_LEFT := \left(SEN\_MATRIX^{T} \cdot SEN\_MATRIX + 0.0001 \cdot E\right)^{-1} \cdot SEN\_MATRIX^{T} \cdot (-UC\_A)$ $W\_A\_LEFT^{T} = (-17.535 \;\; 0.5 \;\; -0.5)$

$W\_A\_RIGHT := \left(SEN\_MATRIX^{T} \cdot SEN\_MATRIX + 0.0001 \cdot E\right)^{-1} \cdot SEN\_MATRIX^{T} \cdot UC\_A$ $W\_A\_RIGHT^{T} = (17.535 \;\; -0.5 \;\; 0.5)$

$m := 0 .. NP$





ANN CONTROL SIMULATION

Artificial Neuron Net_CONTROL VOLTAGE FUNCTION

$UA\_N(X,XF,W) := DAF\_V(X,XF) \cdot W_0 + AV\_V(X) \cdot W_1 + DVAF\_V(X,VF) \cdot W_2$

ACTOR NEURON FUNCTION

```
NEU_A(VAL,LIM_U,LIM_L) := | x ← 0
                          | x ← LIM_U  if  VAL > LIM_U
                          | x ← -LIM_U  if  VAL < -LIM_U
                          | x ← VAL  if  LIM_L < VAL ≤ LIM_U
                          | x ← VAL  if  -LIM_U < VAL ≤ -LIM_L
                          | return  x
```

Final angler position is changed in comparison to learning sample $XF_2 := \pi$

NEURON NET CONTROL SYSTEM

$$X0 = \begin{pmatrix} 0\\0\\0\\0\\0\\0\\0 \end{pmatrix} \quad XF = \begin{pmatrix} 0\\0\\3.142\\0\\0\\0\\0 \end{pmatrix} \quad VF = \begin{pmatrix} 0\\0\\0\\0\\0\\0\\0 \end{pmatrix}$$

UNLIMETED VOLTAGE

$$UN(T,X) := \begin{pmatrix} 0\\0\\0\\0\\0\\ \dfrac{UA\_N(X,XF,W\_A\_LEFT)}{Lm}\\ \dfrac{UA\_N(X,XF,W\_A\_RIGHT)}{Lm} \end{pmatrix}$$

LIMETED VOLTAGE $VMIN := 3$

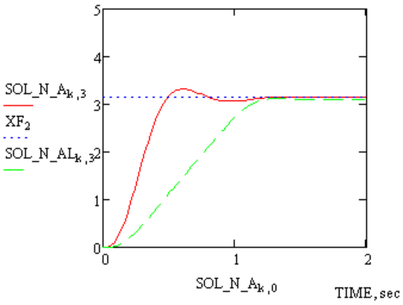
$$UNL(T,X) := \begin{pmatrix} 0\\0\\0\\0\\0\\ \dfrac{NEU\_A(UA\_N(X,XF,W\_A\_LEFT),VMAX,VMIN)}{Lm}\\ \dfrac{NEU\_A(UA\_N(X,XF,W\_A\_RIGHT),VMAX,VMIN)}{Lm} \end{pmatrix}$$

TEST SOLUTION $T0 := 0$ $T1 := 2$ $NP := 100$ $FUN\_NA(T,X) := FUN(T,X) + UN(T,X)$ $FUN\_NAL(T,X) := FUN(T,X) + UNL(T,X)$

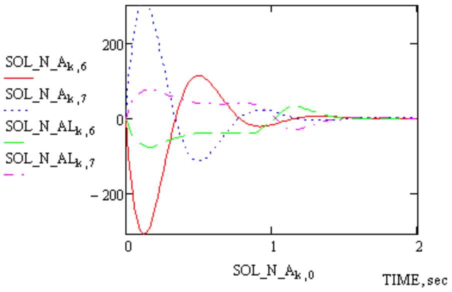
$SOL\_N\_A := Rkadapt(X0,T0,T1,NP,FUN\_NA)$ $SOL\_N\_AL := Rkadapt(X0,T0,T1,NP,FUN\_NAL)$

$k := 0 .. NP$

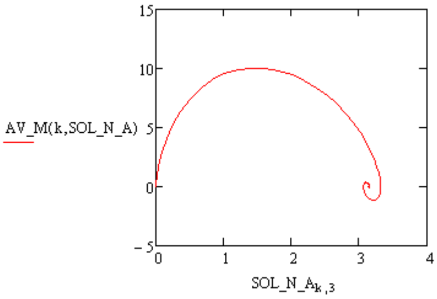
ANGLE

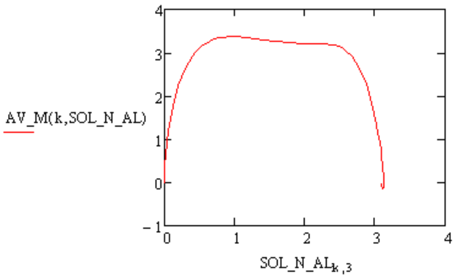


CURRENTS



PHASE PORTRAITS


AV_M(k,SOL_N_A)
$SOL\_N\_A_{k,3}$


AV_M(k,SOL_N_AL)
$SOL\_N\_AL_{k,3}$