APPLICATIONS OF ARTIFICIAL INTELLIGENCE IN AREAS CONTROL ENGINEERING

Faculty of Electrical Engineering, University of Sarajevo

MÄLARDALEN UNIVERSITY SWEDEN

Red. prof. dr. Zikrija Avdagić, dipl.ing.el.

V. asis. mr Dušanka Bošković dipl.ing.el.

MoEE Student Belma Čardaković

Associate Professor dr Damir Isovic

Professor dr. Lars Asplund

Ph.D student Aida Causevic

LINNEAUS – PALME EXCHANGE PROGRAMME

Cooperation in the framework of a joint Linnaeus-Palme exchange programme

This project with its objective- exchange for teaching staff and students at university level stimulated cooperation based on mutual benefit between Malardalen University and University of Sarajevo.

Within this project were conducted two master thesis (1. Fuzzy controller optimization using genetic algorithm and 2. Fuzzy system modeling using ANFIS GUI tool), and one research project based on ROSA Real Time Operating System.

Project timetable:

2009 - 2011

Theoretical research

Field: AI - Artificial Intelligence

Field: RTCS - Real Time Computer Systems

Field: CE - Control Engineering

Elvira Laković, MoEE

Abstract

Fuzzy logic is well known and widely used today in control systems. Neural networks are also used in a wide variety of systems. Combination of these two different approaches should have many beneficial properties for solving many different problems, but still this combination is not used as much as it could be used. In this master thesis, possibilities for application of neuro-fuzzy controller for aircraft landing problem are explored. Simplified model of aircraft landing problem is used. For system simulation, MATLAB's Fuzzy Logic Toolbox and GUI are utilized.

Problem description

The speed of airplane landing is proportional to the square of height. Airplane is rapidly decreasing altitude, and landing very carefully in order to avoid damage. We will ignore the external influences (wind, etc.). We can identify two variables, which we take as an input to the system, those are height (h) and vertical velocity (v) of the airplane. The system output is the force f, that causes change in airplane altitude and velocity. The mass m, moving with velocity v has the impulse p = mv. If there is no activity of external forces, the mass will continue to move in the same direction, with the same velocity v. If the force f is active during the time interval Δt, the change of velocity will be equal: $\Delta v = f \cdot \frac{\Delta t}{m}$

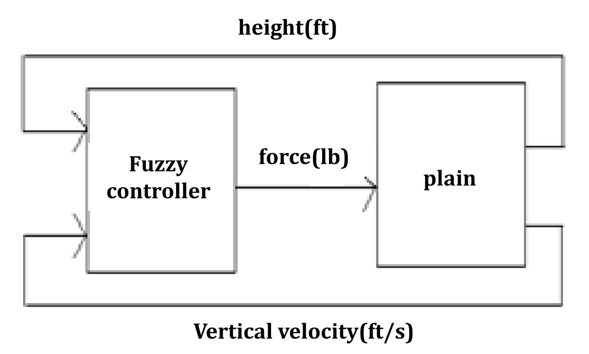
$$\Delta v = f \cdot \frac{\Delta t}{m} = [lb] \cdot \frac{[s]}{[lb\frac{s^2}{ft}]} = [\frac{ft}{s}]$$

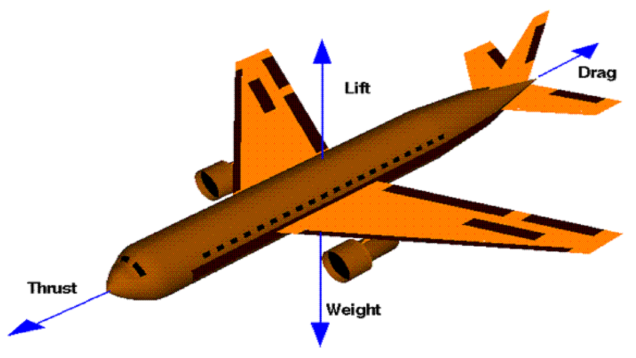
Assuming that Δt = 1.0 (s) and m = 1.0 (lb s^2/ft), we can say that Δv = f, or that the change of velocity is proportional to the applied force.

New variable values are calculated as follows:

$$v_{i+1} = v_i + f_i \qquad i = 0, 1, 2, \ldots, n$$

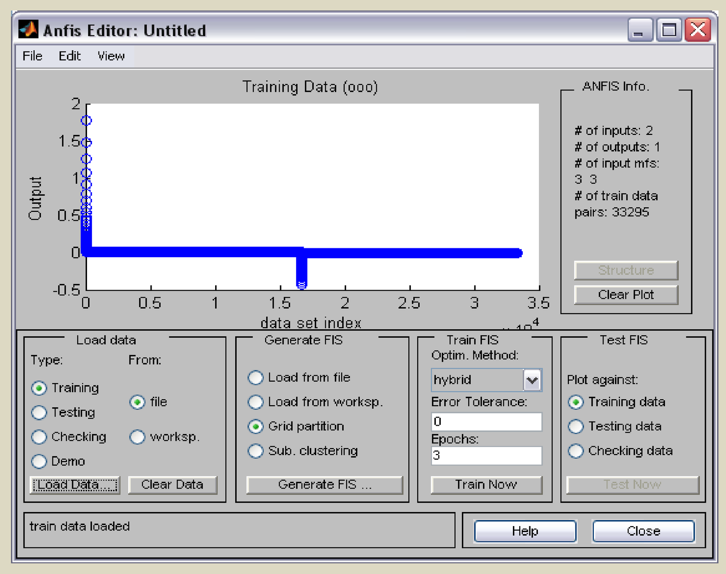
$$h_{i+1} = h_i + v_i \cdot \Delta t$$



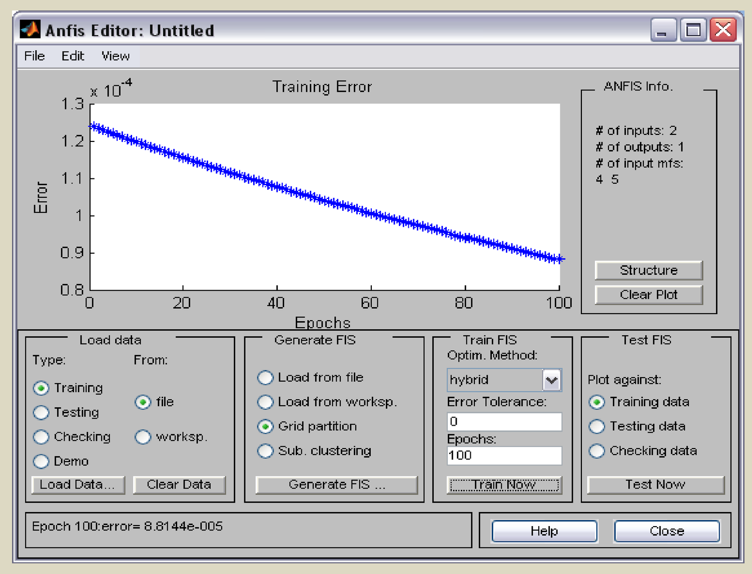


Four main forces acting on an airplane during flight

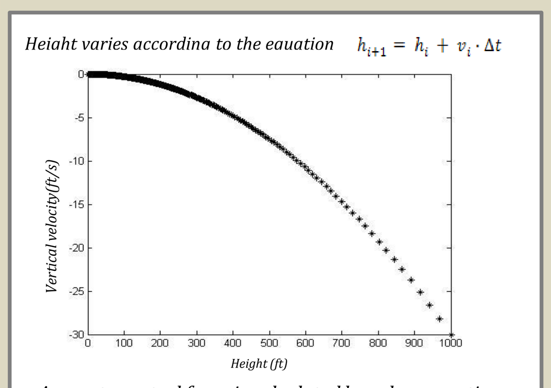
Inputs as linguistic variables and member functions in FIS structure



The complete training data set

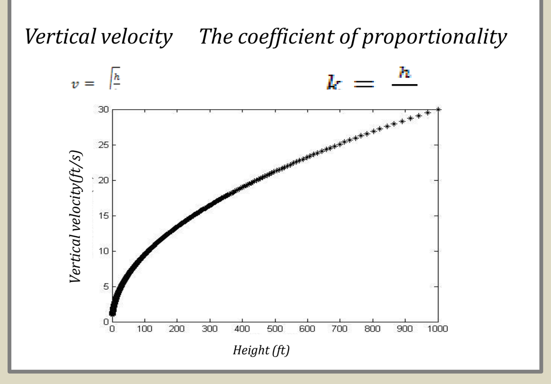


Training error in 100 epochs

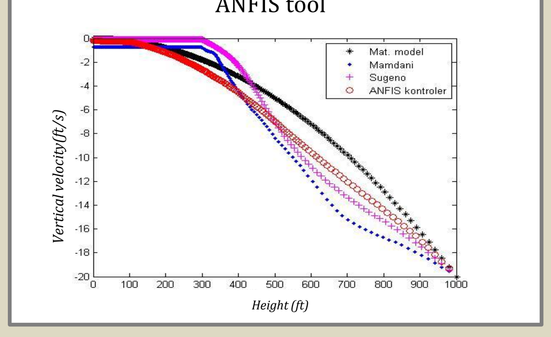


Accurate control force is calculated based on equations $f_i = v_{i+1} - v_i$

Data set for positive vertical velocity



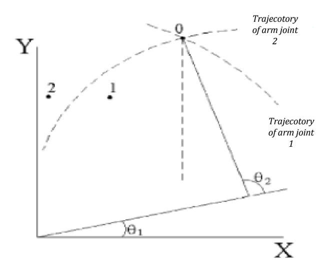
Comparison of mathematical model of fuzzy system, hand-modeled Mamdani and Sugeno controllers and controller modeled by ANFIS tool



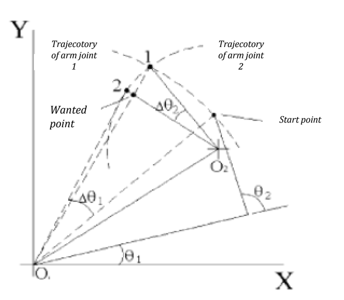
Damir Lotinac, MoEE

Abstract

This thesis analyses the possibility of application of genetic algorithm to fuzzy controller optimization. Fuzzy controller, for which the parameters are optimized, solves the problem of inverse kinematics for two segment robot arm. The implemented genetic algorithm is adequate for setting parameters of such a fuzzy controller. The evolution of membership functions was satisfactory. However, if we want a controller that will be a complete solution of such problems, we need to adjust rules as well as membership functions.

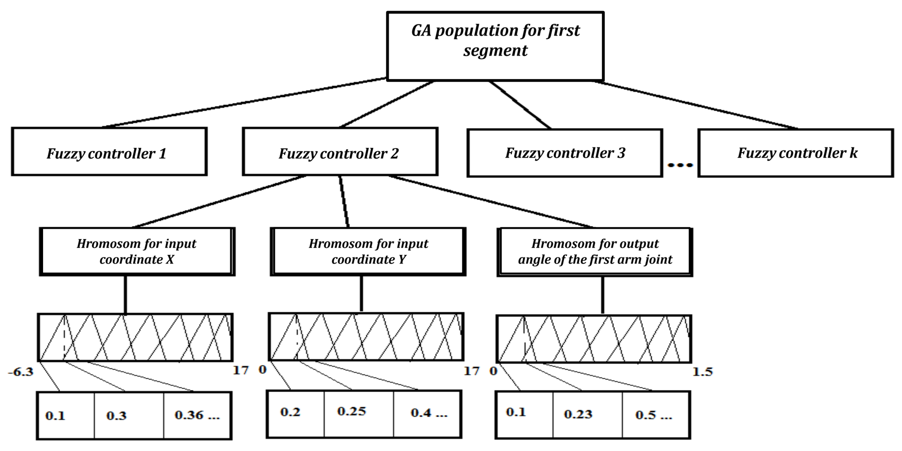


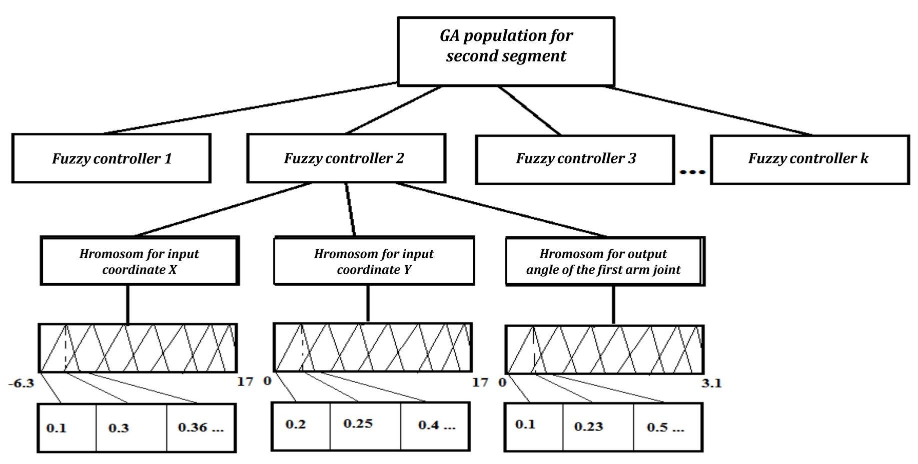
Direct kinematics – process of calculating the last point of the associated structure in space, when angles of all joints are known

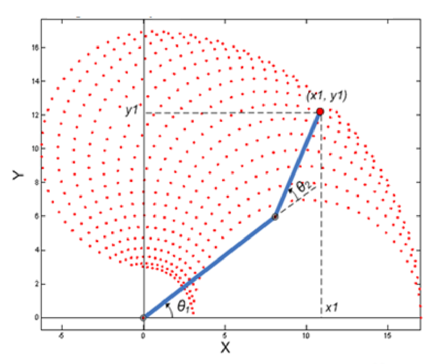


Inverse kinematics – process of setting angles of all joints, based on known endpoint of the associated structure

Fuzzy - Genetic system

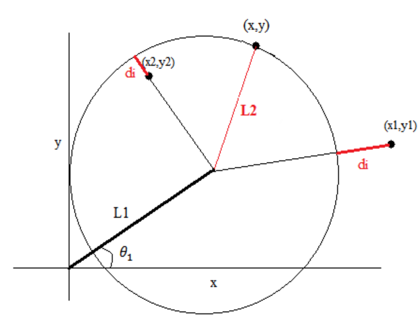




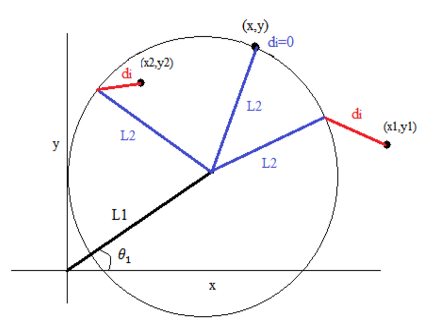


Fitness of controller: $f_j = \frac{n}{\sum_{i=1}^{n} d_i}$

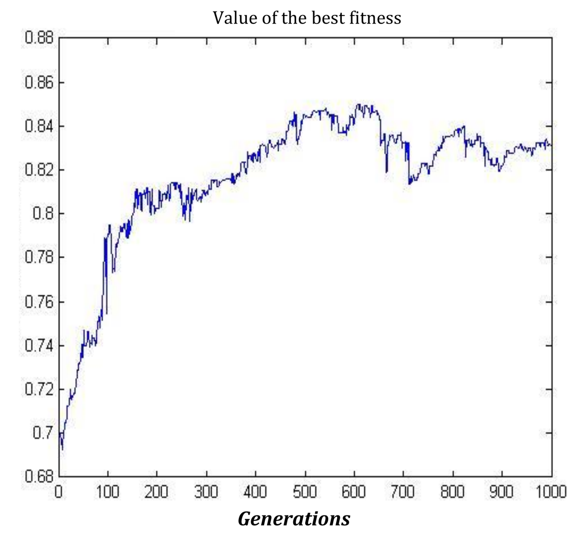
*Measuring the distance from the end of the **first segment***



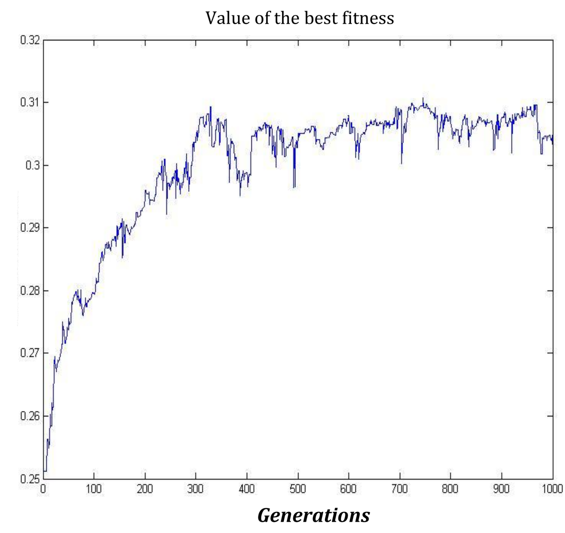
*Measuring the distance from the end of the **second segment***



Results - First segment



Results - Second segment



MoEE Student Irfan Šljivo

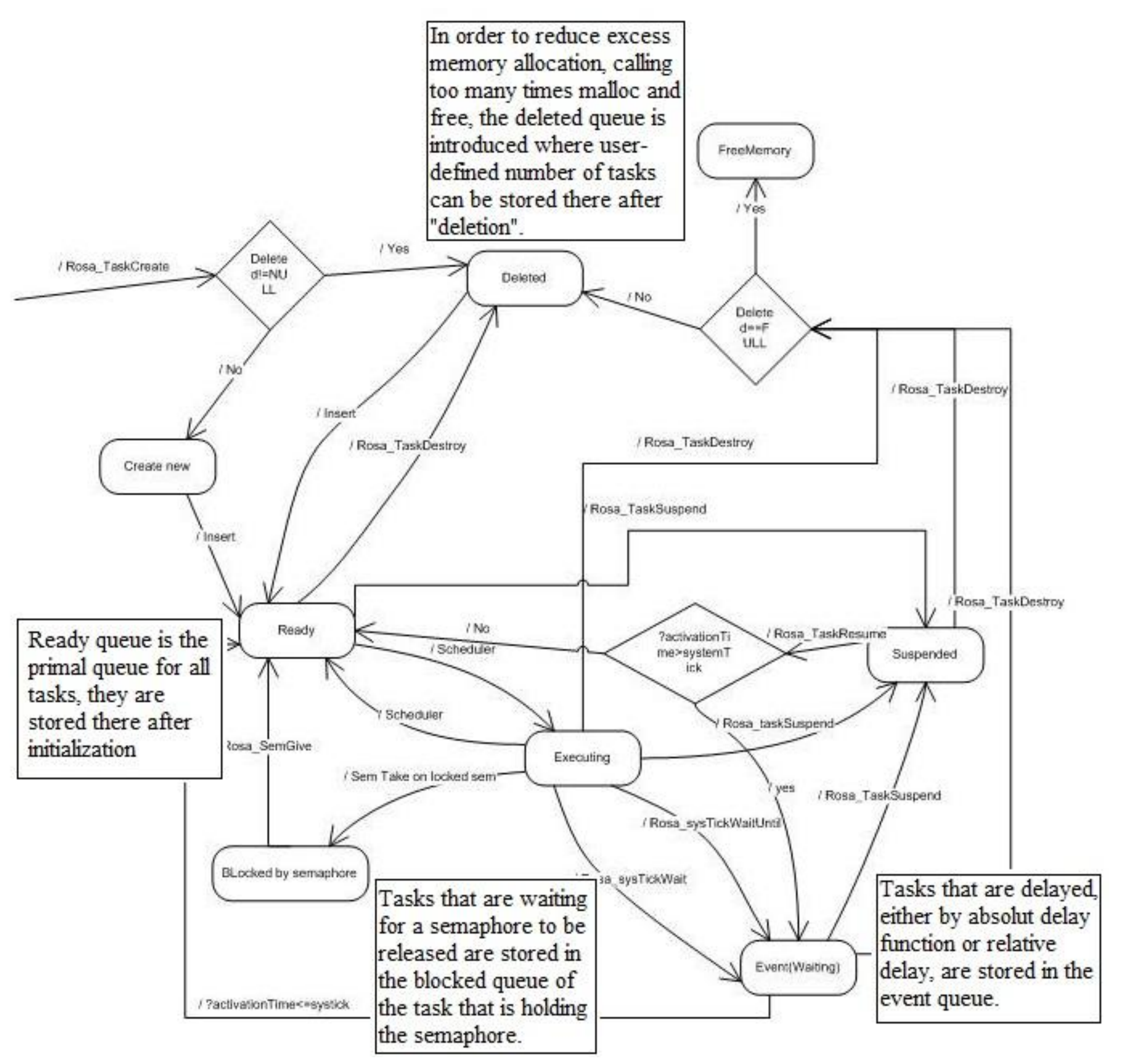
ROSA

ROSA is a Real Time Operating System for Atmel board EKV1100. The system is constructed with time slicing priority based scheduler, and includes the implementation of Priority Inheritance Protocol which makes it dynamic priority scheduler.

Besides these functionalities the system enables dynamic task creation and termination, it contains functions for delaying tasks, for both absolute and relative delay. Another feature of this RTOS is binary and mutex semaphores.

Tasks can be in one of six different states, those are: executing, ready, waiting for an event, suspended, blocked by semaphore or deleted. Each of these states is defined as a queue in form of either an circular double linked list or an circular single linked list.

The RTOS enables another feature called safe mode, where dynamic allocation is avoided. In this mode the maximum number of tasks is declared before runtime, and the static task stack size is chosen in order to allocate memory for all tasks. This feature eliminates usage of dynamic memory allocation which is a big problem for embedded systems due to memory fragmentation.



CODE SNIPPETS FROM ROSA KERNEL

```
void schedulerPreemptive(void)
{
if(TCBREADY_LIST!=NULL){
        //Find the next task to execute
        if(EXECTASK!=NULL)
        {
        if(TCBREADY_LIST->priority>=EXECTASK->priority )
        {
                ROSA_insertReady(&EXECTASK);
                EXECTASK=TCBREADY_LIST;
                EXECTASK->status=EXECUTINGTASK;
                removeFromReady(&TCBREADY_LIST);
}
        }else{
                EXECTASK=TCBREADY_LIST;
                EXECTASK->status=EXECUTINGTASK;
                removeFromReady(&TCBREADY_LIST);

        }
}}
```

This is the preemptive scheduler. It checks if there are tasks that should be executing and if there is such a task, then it swaps it with the currently executing one.

A task can be in only one state at a time. There is always one task in the system, in order for the system to keep running.

All queues, but suspended according to id, are organized like double linked lists and suspended by id is organized as a single linked list.

The part of the code in the function int ROSA_semGive(sem * semaphore); for releasing a semaphore, that is used for implementation of the Priority Inheritance Protocol. If the task that is releasing a semaphore has had its priority raised by higher priority task that has been blocked by that task, then its priority should be restored to original value, or a value of the highest priority task that is blocked by that task, in case when that task is holding more then one semaphore.

```
....
int maxPrioHold=0;

if(semaphore->owner->holdsSem!=NULL){
maxPrioHold=semaphore->owner->holdsSem->blockedTasks->priority;
sem *temporary=semaphore->owner->holdsSem;
while(temporary->next!=semaphore->owner->holdsSem)
{       if( maxPrioHold<temporary->next->blockedTasks->priority)
                maxPrioHold=temporary->next->blockedTasks->priority;
temporary=temporary->next;
}
}
if(maxPrioHold>semaphore->owner->origPriority)
        semaphore->owner->priority=maxPrioHold;
else
        semaphore->owner->priority=semaphore->owner->origPriority;
....
```