In the software: the sensed values of a and da are specified through an input file “infile1-SenData.txt” as shown in Table 1. The input/output membership functions are specified through the input file “infile2-IOMFs.txt”, refer to Table 2. The knowledge base is specified through another input file “infile3-KB.txt”, this is shown in Table 3. The generated output is stored into an output file “fles-outfile.txt”, this file is shown in Table 4.

While operating in real-time the knowledge base and the MFs are merged into the FLES-source code. Sensed input values and the generated outputs are mapped to i/o-ports of the microcontroller being used. Depending on the problem the input, the output, and the knowledge base modules can be specified accordingly. It includes the name and units of input and output variables; the IMFs and the number of IMFs, OMFs and the number OMFs, and finally the KB.

The detailed results of execution tasks IE1 through IE5 of the inference engine, specified in section IV, are shown in Table 4. This can be disabled in on-line mode. In this table, the initial sections A, B, and C will display input data to the FLC. This includes the sensor data of the input variables; the input/output membership functions; and the knowledge base of the problem. The task here is to build an FLC to balance an inverted pendulum problem. It is a two-input and one-output problem. The input variable values for the FLC are: the pendulum’s angular displacement (angle a) is -12.0 degrees away from the center, and rate of angular displacement (derivative of angle da) is -3.0 degrees/sec. Input variables in1 and in2 both have 7 membership functions. Minimum and maximum values are -54 to +54 with 18-units between the vertices with units of degrees for in1 and degrees/sec for in2. The output variable mc has 7-MFs with minimum and maximum of -18 ma and +18 ma with 6 ma separation. The mc linguistic values ranging from negative-large (nl) to positive-large (pl). The knowledge base of the IP-problem is specified by 13-rules denoted as R0-to-R12. These are logically ANDed-rules.

The 5-step procedure by which the inference engine generates the control signal from the sensor input data is shown in section-D of Table 4. The five steps are denoted as: IE1 through IE5.

```
The IE1-Trace: In IE1 the input variables
are fuzzified: for in1 = -12 degrees, the
fuzzified values are: (12/18)-ns or
(6/18)-zr; and for in2 = -3 degrees/sec,
the fuzzified values are: (3/18)-ns or
(15/18)-zr.
```

```
The IE2-trace: In IE2 active rule set Ra
is determined: the activated rules are R7
and R8. Ra = {R7, R8}.
```

```
The IE3-trace: In IE3 effective input
membership functions eIMF-Rai for each
active rule Rai is determined:
eIMF-R7 = (12/18)-ns;
eIMF-R8 = (6/18)-zr;
```

```
The IE4-trace: In IE4 Rule to be fired is
determined:
Rule fired = max(eIMFs) = eIMF-R7 =
(12/18);
Rule fired is R7
```

```
The IE5-trace: In IE5 find output – motor
current mc:
output = eIMF * eOMF = (12/18) * ps
       = (12/18) * 6.0 = 4.0 mille-amps
```

VI. CONCLUSIONS

This paper presents embeddable FLC-software-code that can be downloaded into microcontrollers to build autonomous mobile-robots. We have used this software to implement two fuzzy logic systems, one is an estimation problems and the other a control problem. The estimation problem is an application one that is used for precise estimation of the battery’s state of charge SoC [5]. For this we have used Handyboard that supports interactive C. This required partial modification C++ into interactive-C. Recently we have implemented the same application on Arduino-Due microcontroller. The other is a control application that we have implemented is to balance an inverted pendulum. This is again implemented on a Arduino-Due microcontroller. This is an autonomous robot. The current microcontrollers are very powerful yet very inexpensive (under $50). With your own embeddable source-code, such as the one presented in this paper, one can build extremely complex yet inexpensive mobile-FLC.

REFERENCES

[1] Kim C. Ng, A Neuro-Fuzzy Controller for Mobile Robot Navigation and Multirobot Convoying, IEEE Transactions on Systems, Man, and Cybernetics – part B, VOL. 28, NO. 6, Pg. 829-840, 1998.

[2] Dimiter Driankov and Alessandro S affiotti, Fuzzy Logic Techniques for Autonomous Vehicle Navigation, A Springer-Verlag, 2001.

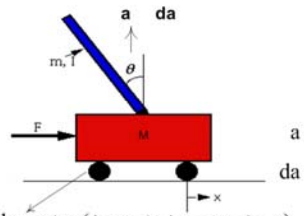
[3] Chuen Lee, "Fuzzy Logic in Control Systems: Fuzzy logic Controller – Part I", IEEE International Conference on Systems, Man, and Cybernetics, 1990, Vol: 20, No. 2, Pages: 404-418.

[4] Chuen Lee, "Fuzzy Logic in Control Systems: Fuzzy logic Controller – Part Part II", IEEE International Conference on Systems, Man, and Cybernetics, 1990, Vol: 20, No. 2, Pages: 419-435

[5] Maila, Sreelatha (G. N. Reddy), “Embedded-System Controlled Fuzzy Logic Expert System to Estimate Battery-SOC, Summer II, Electrical Engineering, Lamar University, 2008.

Fuzzy-Logic Expert Systems FLES: Inference Engine IE: Case Study

Overall operations by the Inference Engine IE in the RT-FLES



dc-motor (+mc => +x; -mc=> -x)

Figure 3. Balancing an Inverted Pendulum.

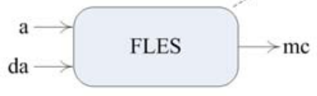
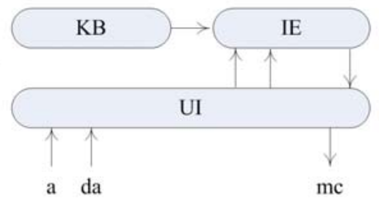


Figure 4a. Fuzzy Logic Expert System FLES with i/o.



FLES: Fuzzy Logic Expert System; KB: Knowledge Base; IE: Inference Engine; UI: User Interface

Figure 4b. Fuzzy Logic Expert System FLES with i/o.

Note: da = dθ/dt is computed: "θ" is sampled into the FLES at a constant interval of around 1000-samples/sec (or at a sampling fs frequency of 1 kHz) from a photo-sensor. That is, dt = 1/fs = 1 msec. $d\theta_n/dt = (\theta_n - \theta_{n-1})$ / 1 msec; For θ_n = 25.02 degrees, θ_{n-1} = 25.00 degrees; dθ = (25.02 – 25.0*) = 0.02 degrees; da = dθ/dt = 0.02/1 ms = 20 deg/sec.

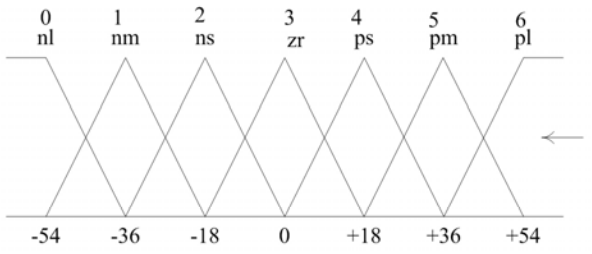


Figure 5. Fuzzy Logic Input Membership Functions IMFs for the input variables: a and da.

Input variable 1: a; Units: degrees; Min / Max: -54 / +54

Input variable 2: da; Units: degrees/sec; Min / Max: -54 / +54

```
Table 1. Sensor Input Data.

// infile1-SenData.txt
// Input values
-12.0    : degrees
-3.0     : degrees/sec
```

Output variable: mc

Units: milli_amps

Min/Max: -18/18

Member Ship Functions:

0 nl	-18
1 nm	-12
2 ns	-6
3 zero	0
4 ps	6
5 pm	12
6 pl	18

Note::
a => angle, θ, theta, angular displacement;
da => derivative of the angle, dθ/dt, θ'

Figure 6. Output Membership Functions OMFs for the output variable mc.

```
Table 2. Input/Output Membership Functions IOMFs.

// infile2-IOMFs.txt
// Input Membership Functions IMFs: min, max, n
-54.0  : in1min a: degrees
+54.0  : in1max a: degrees
 7     : in1num
-54.0  : in2min da: degrees/sec
+54.0  : in2max da: degrees/sec
 7     : in2num
-18.0  : in3min mc: milli-amps
+18.0  : in3max mc: milli-amps
 7     : in3num
```

```
FLES Knowledge-Base: Rule-Set
R0: IF a IS zr AND da IS nm THEN mc IS pm
R1: IF a IS ps AND da IS ns THEN mc IS ps
R2: IF a IS zr AND da IS ps THEN mc IS ns
R3: IF a IS zr AND da IS pm THEN mc IS nm
R4: IF a IS zr AND da IS pl THEN mc IS nl
R5: IF a IS nl AND da IS zr THEN mc IS pl
R6: IF a IS nm AND da IS zr THEN mc IS ps
R7: IF a IS ns AND da IS zr THEN mc IS ps
R8: IF a IS zr AND da IS zr THEN mc IS zr
R9: IF a IS ps AND da IS zr THEN mc IS ns
R10:IF a IS pm AND da IS zr THEN mc IS nm
R11:IF a IS pl AND da IS zr THEN mc IS nl
R12:IF a IS ns AND da IS ps THEN mc IS ps
```

Figure 7. The FLES Knowledge Base KB for the IP-Problem

```
Table 3. The Knowledge Base for the IP-Problem.

// infile3-KB.txt
// Knowledge base for the fuzzy logic expert system
// Example: infile3KB.txt for: 13 rules, 2-inputs, 1-output
// MFs: 0-nl; 1-nm; 2-ns; 3-zr; 4-ps; 5-pm; 6-pl
// Rule: 0  1  2  3  4  5  6  7  8  9  10 11 12
//       ----------------------------------------------
         zr ps zr zr zr nl nm ns zr ps pm pl ns :kbIn1 MFs
         nm ns ps pm pl zr zr zr zr zr zr zr ps :kbIn2 MFs
         pm ps ns nm nl pl ps ps zr ns nm nl ps :kbOut MFs
```

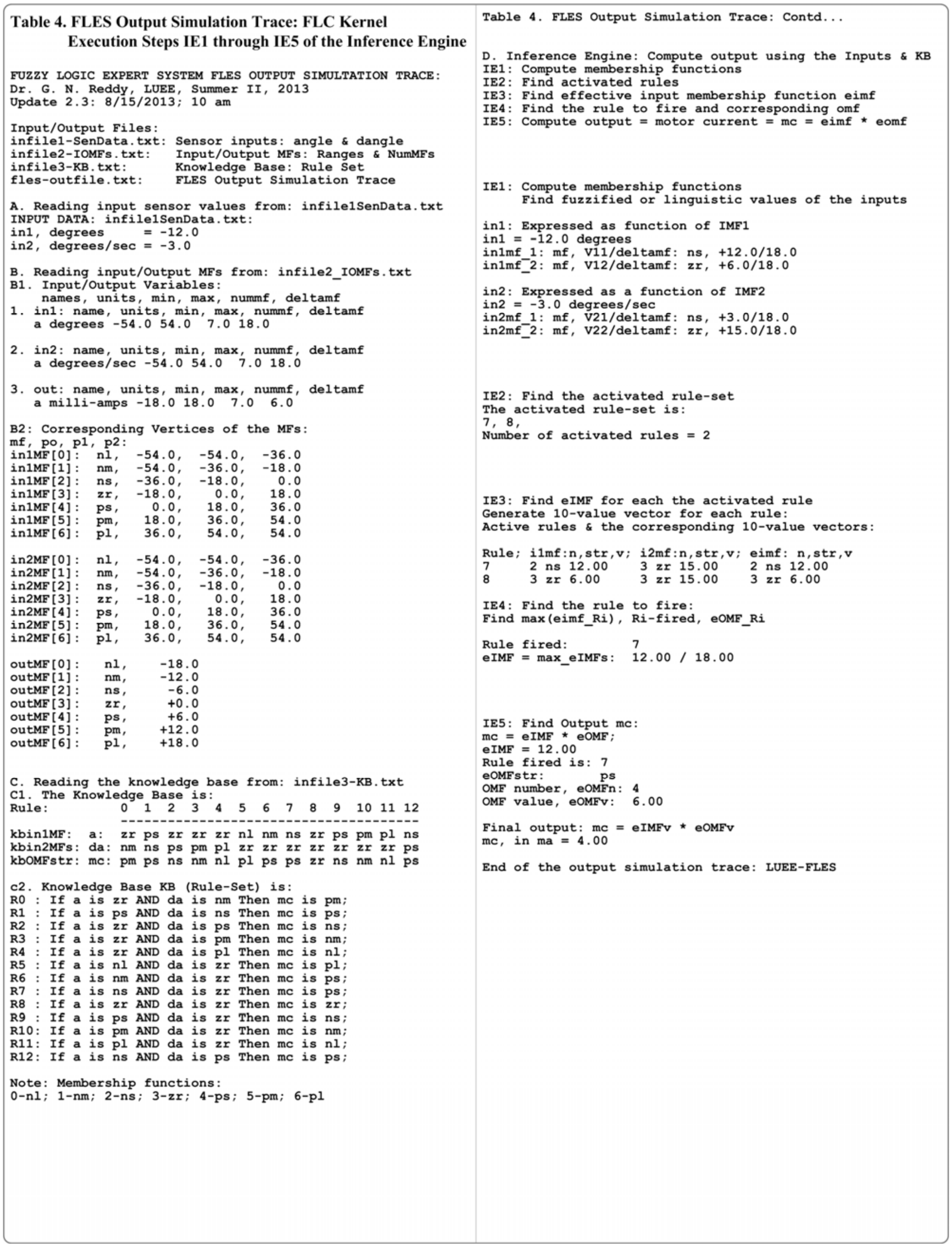


Table 4. FLES Output Simulation Trace: FLC Kernel Execution Steps IE1 through IE5 of the Inference Engine

```
FUZZY LOGIC EXPERT SYSTEM FLES OUTPUT SIMULTATION TRACE:
Dr. G. N. Reddy, LUEE, Summer II, 2013
Update 2.3: 8/15/2013; 10 am

Input/Output Files:
infile1-SenData.txt: Sensor inputs: angle & dangle
infile2-IOMFs.txt:   Input/Output MFs: Ranges & NumMFs
infile3-KB.txt:      Knowledge Base: Rule Set
fles-outfile.txt:    FLES Output Simulation Trace

A. Reading input sensor values from: infile1SenData.txt
INPUT DATA: infile1SenData.txt:
in1, degrees     = -12.0
in2, degrees/sec = -3.0

B. Reading input/Output MFs from: infile2_IOMFs.txt
B1. Input/Output Variables:
    names, units, min, max, nummf, deltamf
1. in1: name, units, min, max, nummf, deltamf
   a degrees -54.0 54.0  7.0 18.0

2. in2: name, units, min, max, nummf, deltamf
   a degrees/sec -54.0 54.0  7.0 18.0

3. out: name, units, min, max, nummf, deltamf
   a milli-amps -18.0 18.0  7.0  6.0

B2: Corresponding Vertices of the MFs:
mf, po, p1, p2:
in1MF[0]:  nl,  -54.0,  -54.0,  -36.0
in1MF[1]:  nm,  -54.0,  -36.0,  -18.0
in1MF[2]:  ns,  -36.0,  -18.0,    0.0
in1MF[3]:  zr,  -18.0,    0.0,   18.0
in1MF[4]:  ps,    0.0,   18.0,   36.0
in1MF[5]:  pm,   18.0,   36.0,   54.0
in1MF[6]:  pl,   36.0,   54.0,   54.0

in2MF[0]:  nl,  -54.0,  -54.0,  -36.0
in2MF[1]:  nm,  -54.0,  -36.0,  -18.0
in2MF[2]:  ns,  -36.0,  -18.0,    0.0
in2MF[3]:  zr,  -18.0,    0.0,   18.0
in2MF[4]:  ps,    0.0,   18.0,   36.0
in2MF[5]:  pm,   18.0,   36.0,   54.0
in2MF[6]:  pl,   36.0,   54.0,   54.0

outMF[0]:   nl,    -18.0
outMF[1]:   nm,    -12.0
outMF[2]:   ns,     -6.0
outMF[3]:   zr,     +0.0
outMF[4]:   ps,     +6.0
outMF[5]:   pm,    +12.0
outMF[6]:   pl,    +18.0


C. Reading the knowledge base from: infile3-KB.txt
C1. The Knowledge Base is:
Rule:          0  1  2  3  4  5  6  7  8  9  10 11 12
               --------------------------------------
kbin1MF:  a:   zr ps zr zr zr nl nm ns zr ps pm pl ns
kbin2MFs: da:  nm ns ps pm pl zr zr zr zr zr zr zr ps
kbOMFstr: mc:  pm ps ns nm nl pl ps ps zr ns nm nl ps

c2. Knowledge Base KB (Rule-Set) is:
R0 : If a is zr AND da is nm Then mc is pm;
R1 : If a is ps AND da is ns Then mc is ps;
R2 : If a is zr AND da is ps Then mc is ns;
R3 : If a is zr AND da is pm Then mc is nm;
R4 : If a is zr AND da is pl Then mc is nl;
R5 : If a is nl AND da is zr Then mc is pl;
R6 : If a is nm AND da is zr Then mc is ps;
R7 : If a is ns AND da is zr Then mc is ps;
R8 : If a is zr AND da is zr Then mc is zr;
R9 : If a is ps AND da is zr Then mc is ns;
R10: If a is pm AND da is zr Then mc is nm;
R11: If a is pl AND da is zr Then mc is nl;
R12: If a is ns AND da is ps Then mc is ps;

Note: Membership functions:
0-nl; 1-nm; 2-ns; 3-zr; 4-ps; 5-pm; 6-pl
```

```
Table 4. FLES Output Simulation Trace: Contd...

D. Inference Engine: Compute output using the Inputs & KB
IE1: Compute membership functions
IE2: Find activated rules
IE3: Find effective input membership function eimf
IE4: Find the rule to fire and corresponding omf
IE5: Compute output = motor current = mc = eimf * eomf



IE1: Compute membership functions
     Find fuzzified or linguistic values of the inputs

in1: Expressed as function of IMF1
in1 = -12.0 degrees
in1mf_1: mf, V11/deltamf: ns, +12.0/18.0
in1mf_2: mf, V12/deltamf: zr, +6.0/18.0

in2: Expressed as a function of IMF2
in2 = -3.0 degrees/sec
in2mf_1: mf, V21/deltamf: ns, +3.0/18.0
in2mf_2: mf, V22/deltamf: zr, +15.0/18.0



IE2: Find the activated rule-set
The activated rule-set is:
7, 8,
Number of activated rules = 2



IE3: Find eIMF for each the activated rule
Generate 10-value vector for each rule:
Active rules & the corresponding 10-value vectors:

Rule; i1mf:n,str,v; i2mf:n,str,v; eimf: n,str,v
7     2 ns 12.00     3 zr 15.00    2 ns 12.00
8     3 zr 6.00      3 zr 15.00    3 zr 6.00

IE4: Find the rule to fire:
Find max(eimf_Ri), Ri-fired, eOMF_Ri

Rule fired:         7
eIMF = max_eIMFs:  12.00 / 18.00



IE5: Find Output mc:
mc = eIMF * eOMF;
eIMF = 12.00
Rule fired is: 7
eOMFstr:        ps
OMF number, eOMFn: 4
OMF value, eOMFv:  6.00

Final output: mc = eIMFv * eOMFv
mc, in ma = 4.00

End of the output simulation trace: LUEE-FLES
```