Handling Paraconsistency and Paracompleteness in Robotics

Jair M. Abe  
Graduate Program in Production Engineering  
Paulista University  
São Paulo, Brazil  
jairabe@uol.com.br

Kazumi Nakamatsu  
University of Hyogo, Hyogo, Japan  
nakamatu@shse.u-hyogo.ac.jp

Seiki Akama  
C-Republic, Tokyo, Japan  
akama@jcom.home.ne.jp

Alireza Ahrary  
Sojo University  
Kumamoto, Japan  
ahrary@cis.sojo-u.ac.jp

Abstract — Automation, and Robotics have experienced incredible improvements over the past decades. Due to the high sophistication more and more elaborated logical tool became necessary to overcome what classical logic offers. In this work, it is shown how a new class of non-classical logics, namely paraconsistent annotated evidential logic $E_\tau$, can be used to deal situations where the presence of conflicting, diffuse and paracomplete must be treated in a non-trivial way. Our purpose is to show its usefulness through examples that have been successfully applied.

Keywords — paraconsistent logic, automation, robotics, annotated logics

I. INTRODUCTION

In this expository work, we present an application of hardware-based on a new class of non-classical logics, namely the so-called paraconsistent annotated logics. We focus on a particular annotated logics; the paraconsistent annotated evidential logic $E_\tau$ [1], [3].

Why of the consideration of the use of concepts of logic $E_\tau$? With the advancement of technology, automation, and robotics, it has become one of the areas of most significant interest and the sophistication of details has motivated the search for ever more elaborate tools. For all this to have an adequate and safe and robust operation, it is necessary to study its fundamentals.

Theoretical bases should be carefully considered, and more general and formal tools should be investigated. Some concepts that have attracted the attention of specialists in recent times are handling of conflicting signals (for example, in a multi-agent system the study of conflicts can have different meanings; Fuzzy logic has been widely used for this purpose, has been used in control systems, etc.

The Fuzzy logic has been widely used for this purpose, has been used in control systems, etc. The last concept that we would like to mention is the one of paracompleteness.

Depending on the application can vary of meaning: in robotic implementations (for example in the use of sensors) can be interpreted like dual concept of the inconsistency (see, eg [2]) in information system can be interpreted as lack of information.

Both the concepts of inconsistency and that of paracompleteness cannot be directly into existing systems, as far as we know; hence the importance of paraconsistent and paracomplete logics that can directly address such constrictions and their physical implementation is direct without the aid of extra-logical devices. The para-control

Logical circuits and programs can be designed based on the Para-analyzer algorithm [11].

A hardware or software built by using the Para-analyzer, to treat logical signals according to the structure of the paraconsistent annotated logic $E_\tau$, is a logic controller that we call Para-control [14]. A software based on the Para-analyzer can be built by using usual the programming languages such as C, C++, Pascal, VB, etc. In the present section, we discuss hardware systems based on the Para-control.

The Para-control is built via "electronic" implementation of the Para-analyzer. The circuit makes the comparison between the values found and the external adjustments to define the region of the lattice that represents the resulting output logical state.

The circuit that we are discussing was built with discrete devices, and it has the following characteristics:

a) Electric voltage levels represent the favorable evidence degree and contrary evidence degree from 0 to 5 Volts DC.

b) The equations for determining the certainty degrees and the contradiction degrees are similar to configurations of operational amplifiers of type 741.

c) For external adjustments, the upper limit control values are represented by positive voltage and the lower limit control values are represented by a negative voltage.

d) The resulting output logical state is represented by a high level of voltage at a specific point in the output. This signal works as an indicator flag of the resulting logical state after the paraconsistent analysis. There are 12 outputs represented by a word of 12 digits.

e) The certainty degree and the contradiction degree are presented at the outputs as analog control.

f) The inversion of the certainty degree obtains a logical negation of the states. Therefore we realize the operator NOT as an inverting amplifier.

The block diagram of the para-control is as follows:
The Para-control is a hybrid circuit, i.e., has both analog and digital components. The analog part of the circuit produces two output analog values representing the certainty degree and contradiction degree. The digital part of the circuit compares control values limit and the contradiction and certainty degrees. The Fig. 2 shows some logical values related to the values of electric voltage used in the Para-control circuit.

![Para-control circuit diagram](image)

**Fig. 1.** Representation in blocks of the Para-control circuit

**TABLE I. LOGICAL LEVELS AND AN ELECTRIC VOLTAGE (NEGATIVE VALUES)**

<table>
<thead>
<tr>
<th>EV</th>
<th>5.0</th>
<th>4.5</th>
<th>4.0</th>
<th>3.5</th>
<th>3.0</th>
<th>2.5</th>
<th>2.0</th>
<th>1.5</th>
<th>1.0</th>
<th>0.5</th>
<th>0.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
<td>1.0</td>
</tr>
<tr>
<td>UE</td>
<td>1.0</td>
<td>0.9</td>
<td>0.8</td>
<td>0.7</td>
<td>0.6</td>
<td>0.5</td>
<td>0.4</td>
<td>0.3</td>
<td>0.2</td>
<td>0.1</td>
<td>0.0</td>
</tr>
<tr>
<td>CD</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
<td>1.0</td>
</tr>
<tr>
<td>UD</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
<td>1.0</td>
</tr>
</tbody>
</table>

EV = Electrical voltage  
FE = Favorable Evidence  
UF = Unfavorable Degree  
CE = Certainty Degree  
UD = Uncertainty Degree

**TABLE II. LOGICAL LEVELS AND AN ELECTRIC VOLTAGE (POSITIVE VALUES)**

<table>
<thead>
<tr>
<th>EV</th>
<th>0.0</th>
<th>0.5</th>
<th>1.0</th>
<th>1.5</th>
<th>2.0</th>
<th>2.5</th>
<th>3.0</th>
<th>3.5</th>
<th>4.0</th>
<th>4.5</th>
<th>5.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
<td>1.0</td>
</tr>
<tr>
<td>UE</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
<td>1.0</td>
</tr>
<tr>
<td>CD</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
<td>1.0</td>
</tr>
<tr>
<td>UD</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
<td>1.0</td>
</tr>
</tbody>
</table>

EV = Electrical voltage  
FE = Favorable Evidence  
UF = Unfavorable Degree  
CE = Certainty Degree  
UD = Uncertainty Degree

The circuit analog detector of the extreme states is the first part of the Para-control. It receives two inputs: favorable evidence degree and contrary evidence degree that range from 0 to 5 Volts. The operational amplifiers A1 and A7 are linked in configuration sum/inverter, and they obtain the equation of the contradiction degree: \( G_{Ct} = \mu + \lambda - 1 \), where the value 1 of the equation is associated with voltage +5 Volts. The amplifier A2 inverts the signal of the degree of contrary evidence degree \( \lambda \) and, in A8 it is added to the favorable evidence degree \( \mu \) obtaining the certainty degree signal in the output. These two amplifiers accomplish the equation of the certainty degree: \( G_c = \mu + (-\lambda) = \mu - \lambda \).

The amplifier A9 inverts the favorable evidence degree to enable it to compare with A13 and A12. In the amplifier A13, the comparison is made with the upper certainty control value, which can be adjusted externally by the potentiometer \( V_{ccs}(V) \). In the amplifier A12, the comparison is made with the lower certainty control value that externally can be adjusted through the potentiometer \( V_{cci}(F) \).

The amplifiers A11 and A10 compare the contradiction degree with the externally adjusted values through the potentiometers \( V_{cii}(\perp) \) (lower value of uncertainty control) and \( V_{cii}(\top) \) (upper value of uncertainty control). The amplifiers A3 and A4 verify if the certainty degree and contradiction degree are positive, respectively.

The amplifiers A5 and A6 are two comparators that detect if the favorable evidence and contrary evidence degrees are greater or equal to \( \frac{1}{2} \), whose correspondent voltage value is 2.5 Volts. The output signals of the amplifiers A3, A4, A5, and A6, together with the output signals of the extreme values, serve for logical detection of the non-extreme states.

The output signals of the circuit are adjusted through Zener diodes and resistors, for a voltage of +5 Volts when they are considered to be at logical level 1. When they are considered to be at logical level 0, they will have voltage of 0 Volts.
These voltage values allow the detecting circuit of non-extreme states to be designed with logical gates, as well as to transform the Para-control circuit so that it is compatible with circuits of the TTL (Transistor-Transistor-Logic) family. The analog detector circuit of the extreme states is designed for a symmetrical range of ±12 Volts.

The second part of the circuit Para-control is digital. In this circuit, the logic detection of 8 non-extreme states is carried out. When a non-extreme state is detected, a voltage of +5 Volts appears in the corresponding terminal, as presented in the Fig. 9.

The circuit is designed with usual logical gates OR and AND. The inputs are received from the analog detector circuit of extreme states presented in the Fig. 2.

The signals are combined to present in the output the logical states according to the divisions considered in the lattice previously seen.

Therefore, the main purpose of the logical circuit detector of the non-extreme states is to present a high voltage (+5 Volts) when one of the non-extreme states of the lattice is detected.

The four signals of the extreme states $\top = $ Inconsistent, $t = $ True, $f = $ False, and $\bot = $ Paracomplete are inputs to the logical gates NOR1 and NOR2.

The logical gates NOR1, NOR2, and AND1, are interconnected in such manner that a high level in any one of the four received signals inhibits any output signals of the non-extreme states.

The signals positive $G_0$, positive $G_e$, $\mu_1$ greater than $\frac{1}{2}$ and $\mu_2$ greater than $\frac{1}{2}$ are connected by the logic gates such that the only output signal represents the detected non-extreme logical state.

In fact, the Para-control works as a personal decision-making machine. In industrial applications, the values of the certainty and contradiction degrees will define the conclusion of the paraconsistent analysis giving an adequate answer for the control system.

Depending on the project, the answer of the paraconsistent analysis can be used as a feedback to manipulate the contradictions requiring new information or then merely to help the system to make decisions based on certainty values and of contradiction values.

The Para-control circuit was used as a useful controller of the robot Emmy, where outputs of two ultrasound sensors represent the favorable evidence degrees and the contrary evidence degrees.

The robot Emmy is the first paraconsistent robot ever, and its control hardware employs only paraconsistent logic [2].

Fig. 3. Logical circuit for detecting non-extreme states.

II. ARDUINO CONTROLLER

This paragraph presents the Para-control implemented in Arduino.

```cpp
float C1 = 0.5, C2 = -0.5, C3 = 0.5, C4 = -0.5, Gin, Gce;
String Output;
int S, time = 3000;
float Prev_Val_Mi = 0.0, Prev_Val_La = 0.0;

void setup()
{
    DDRB = DDRB | B00001111;
    lcd.begin(16, 2);
    lcd.setCursor(2, 0);
    lcd.print("PARAANALIZE");
    Serial.begin(9600);
    Serial.println("PARAANALIZE");
    delay(1000);
}

void loop()
{
    int Mi_Sensor_Val, La_Sensor_Val;
    float Pres_Val_Mi, Pres_Val_La;
    do
    {
        delay(time);
        Mi_Sensor_Val = analogRead(A0);
        La_Sensor_Val = analogRead(A1);
        Pres_Val_Mi = Mi_Sensor_Val * (5.0/1023.0);
        Pres_Val_La = La_Sensor_Val * (5.0/1023.0);
        while ((Pres_Val_Mi == Prev_Val_Mi) & (Pres_Val_La == Prev.Val_La));
```
if (((Pres_Val_Mi >= 0) && (Pres_Val_Mi <= 1)) && ((Pres_Val_La >= 0) && (Pres_Val_La <= 1)))
{
    Gce = Pres_Val_Mi - Pres_Val_La;
    Gin = Pres_Val_Mi + Pres_Val_La - 1;
    if (Gce >= C1)
    {
        Output = "V";
        PORTB = B11110001;
    }
    else if (Gce <= C2)
    {
        Output = "F";
        PORTB = B11110010;
    }
    else if (Gin >= C3)
    {
        Output = "I";
        PORTB = B11110011;
    }
    else if (Gin <= C4)
    {
        Output = "P";
        PORTB = B11110100;
    }
    else if (((Gce >= 0) && (Gce < C1)) && ((Gin >= 0) && (Gin < C3)))
    {
        if (Gce >= Gin)
        {
            Output = "Qv->T";
            PORTB = B11110101;
        }
        else
        {
            Output = "Qi->V";
            PORTB = B11110110;
        }
    }
    else if (((Gce >= 0) && (Gce < C1)) && ((Gin > C4) && (Gin <= 0)))
    {
        if (abs(Gce)) >= Gin)
        {
            Output = "Qf->I";
            PORTB = B11111011;
        }
        else
        {
            Output = "Qp->F";
            PORTB = B11111010;
        }
    }
    else if (((Gce > C2) && (Gce <= 0)) && ((Gin > C4) && (Gin <= 0)))
    {
        if (abs(Gce)) >= (abs(Gin))
        {
            Serial.println("Qf->P");
            PORTB = B11110101;
        }
        else
        {
            Serial.println("Qp->F");
            PORTB = B11110110;
        }
    }
    else if (((Gce > C2) && (Gce <= 0)) && ((Gin > C4) && (Gin <= 0)))
    {
        if (abs(Gce)) >= (abs(Gin))
        {
            Serial.println("Values Out of Range");
            printSerial(Pres_Val_Mi, Pres_Val_La, Gce, Gin, Output);
            printLCD(Pres_Val_Mi, Pres_Val_La, Gce, Gin, Output);
            Prev_Val_Mi = Pres_Val_Mi;
            Prev_Val_La = Pres_Val_La;
            void print serial(float Pres_MI, float Pres_LA, float GCE, float GIN, String Output)
        }
        else
        {
            Serial.println("Unknown State");
            PORTB = B11111111;
        }
    }
    else
    {
        Serial.println("Values Out of Range");
        printSerial(Pres_Val_Mi, Pres_Val_La, Gce, Gin, Output);
        printLCD(Pres_Val_Mi, Pres_Val_La, Gce, Gin, Output);
        Prev_Val_Mi = Pres_Val_Mi;
        Prev_Val_La = Pres_Val_La;
    }
    Serial.println(""");
    Serial.println("Mi = ");
    Serial.println(Pres_MI);
    Serial.println(" La = ");
    Serial.println(Pres_LA);
    Serial.println(" Gce = ");
    Serial.println(GCE);
    Serial.println(" Gin = ");
    Serial.println(GIN);
    Serial.println(" S = ");
    Serial.println(Output);
III. ROBOT EMMY II

The Emmy II robot is an autonomous mobile robot able to avoid obstacles while it is moving in a non-structured environment [21], [22].

The control system of the Emmy II uses six logic states instead of 12 logic states used in the Para-analyzer algorithm.

Two sensors are responsible for verifying whether there is any obstacle in front of the robot or not. The signals generated by the sensors are sent to a microcontroller. These signals are used to determine the favorable evidence degree ($\mu$) and the contrary evidence degree ($\lambda$) of the formula “The front of the robot is free.” The favorable and contrary evidence degrees are used to determine the robot movements.

The favorable evidence degree is the signal generated by the sensor 1. The signal generated by the sensor two is considered the contrary evidence degree regarding the formula “Ahead there is no obstacle”. The favorable evidence degree accuses that it is low when there is an obstacle near sensor 1. When the obstacle is far from the sensor 1, the favorable evidence degree is high. Otherwise, when there is an obstacle near the sensor 2, the contrary evidence degree is high, and when there is an obstacle far from the sensor 2, the contrary evidence degree is low. The Emmy II controller decision of what movement the robot should perform is based on the lattice showed in Figure 4.

![Fig. 4. The lattice of Emmy II controller](image)

The decision for each logic state is the following:

- The robot goes ahead. DC motors 1 and 2 are supplied for spinning around forward.
- The robot goes back. DC motors 1 and 2 are supplied for spinning around backward.
- The robot turns right. Just DC motor 1 is supplied for spinning around forward.
- The robot turns left. Just DC motor 2 is supplied for spinning around backward.
- The robot turns right. Just DC motor 2 is supplied for spinning around backward.
- The robot turns left. Just DC motor 1 is supplied for spinning around backward.

Here is the justification for each decision:

- If the logical state is true (V), it means that the front of the robot is free. Therefore, it is permitted the robot go ahead.
- When the logical state is inconsistent (T), $\mu$ and $\lambda$ are high (i.e., they belong to the T state). It means that the sensor 1 is far from an obstacle and the sensor 2 is near to an obstacle, so the left side is freer than the right side. Then, the behavior should be to turn left by supplying only the DC motor 2 for spinning around forward and keeping the DC motor 1 stopped.
- When the Paracompleteness ($\bot$) is detected, $\mu$ and $\lambda$ are low. It means that the sensor 1 is near an obstacle and the sensor two is far from an obstacle, so the right side is freer than the left side. Then, the behavior should be to turn right by supplying only the DC motor 1 for spinning around forward and keeping the DC motor two stopped.
- In the false state (F) there are obstacles near the front of the robot. Therefore the robot should go back.
- In the QF→T state, the front of the robot is obstructed but the obstacle is not so near as in the false state, and the left side is a little bit more free than the right side. So, in this case, the robot should turn left by supplying only the DC motor 1 for spinning around backward and keeping the DC motor 2 stopped.
- In the QF→$\bot$ state, the front of the robot is obstructed but the obstacle is not so near as in the false state, and the right side is a little bit freer than the left side. So, in this case, the robot should turn right by supplying only the DC motor 2 for spinning around backward and keeping the DC motor one stopped.

The basic structure of the Emmy II robot is shown in figure 5.

![Fig. 5. The Emmy II robot basic structure](image)
CONCLUSIONS

This work shows at least in outline the usefulness of paraconsistent annotated evidential logic \( E \tau \) in Automation and Robotics. In fact, the controller based on logic \( E \tau \) allows dealing inconsistent states in an easy way, as well as the paracomplete state. Moreover, the electronic implementations are advantageous in comparison with the usual ones.

ACKNOWLEDGMENT

We express special acknowledgments to Dr. João I. da Silva Filho for useful comments.

REFERENCES


