Noname manuscript No.
(will be inserted by the editor)

Evolution-In-Materio: Solving Computational Problems Using Carbon Nanotube-Polymer Composites

Maktuba Mohid · Julian F. Miller · Simon L. Harding · Gunnar Tufte · Mark K. Massey · Michael C. Petty

Received: 31st January 2015 / Accepted: date

Abstract Evolution-in-materio uses evolutionary algorithms to exploit properties of materials to solve computational problems without requiring a detailed understanding of such properties. We show that using a purpose-built hardware platform called Mecobo, it is possible to solve computational problems by evolving voltages and signals applied to an electrode array covered with a carbon-nanotube-polymer composite. We demonstrate for the *first time* that this methodology can be applied to function optimization and also to the tone discriminator problem (TDP). For function optimization, we evaluate the approach on a suite of optimization benchmarks and obtain results that in some cases come very close to the global optimum or are comparable with those obtained using well-known software-based evolutionary approach. We also obtain good results in comparison with prior work on the tone discriminator problem. In the case of the TDP we also investigated the relative merits of different mixtures of materials and organizations of electrode array.

Keywords evolutionary algorithm · evolution-in-materio · material computation · evolvable hardware · function optimization · tone discriminator

Maktuba Mohid
Department of Electronics, University of York, York, UK.
Tel.: +4401904 322370
Fax: +4401904 323433
E-mail: mm1159@york.ac.uk

Julian F. Miller
Department of Electronics, University of York, York, UK.

Simon L. Harding
Department of Electronics, University of York, York, UK.

Gunnar Tufte
Department of Computer and Information Science, Norwegian University of Science and Technology, 7491 Trondheim, Norway.

Mark K. Massey
School of Engineering and Computing Sciences and Centre for Molecular and Nanoscale Electronics, Durham University, UK.

Michael C. Petty
School of Engineering and Computing Sciences and Centre for Molecular and Nanoscale Electronics, Durham University, UK.

1 Introduction

Natural evolution can be looked at as an algorithm which exploits the physical properties of materials. Evolution-in-materio (EIM) aims to mimic this by manipulating physical systems using computer controlled evolution (CCE) [8–10, 16]. In this paper we are using EIM to solve computational problems. It is important to note that one of unique features of EIM is that it aims to exploit physical processes that a designer may either be unaware of, or not know how to utilize. This is discussed in more detail in a recent review of EIM [17].

EIM was inspired by the work of Adrian Thompson who investigated whether it was possible for unconstrained evolution to evolve working electronic circuits using a silicon chip called a Field Programmable Gate Array (FPGA). He evolved a so-called *tone discriminator*, a digital circuit that could discriminate between 1kHz or 10kHz signal [20]. When the evolved circuit was analysed, Thompson discovered that artificial evolution had exploited physical properties of the chip. Despite considerable analysis Thompson and Layzell were unable to pinpoint what exactly was going on in the evolved circuits [21].

In [16] it was argued that materials with a rich physics might be more evolvable than those with an impoverished physics, such as silicon chips [1]. This inspired an attempt to see if computer-controlled evolution could utilize the physical properties of liquid crystal (LCD) to help solve a number of computational problems. The first demonstration showed that it was relatively easy to evolve a tone-discriminator in liquid crystal [6]. This was followed up with work showing that both two-input logic gates and a robot controller could be accomplished using computer controlled evolution of electrical configurations of liquid crystal [7,9]. Natural materials have many physical properties and internal processes that may be able to be exploited. Indeed, the history of evolvable hardware research has many examples where unusual and incidental physical aspects were incorporated in evolved solutions. Thompson's work demonstrated the exploitation of subtle physical properties of an FPGA chip [21] and Layzell discovered evolved circuits that depended on the presence of a high impedance oscilloscope [13] and evolved oscillators that were using incidental radio signals [2]. This is suggestive that physical systems are useful in genotype-phenotype mappings as they afford many ways of providing exploitable effects that can contribute to improved fitness. Kirschner and Gerhart defined evolvability (in a biological context)as the "the capacity to generate heritable selectable phenotypic variation"[12] and this results in an ability"(i) to reduce the potential lethality of mutations and (ii) to reduce the number of mutations needed to produce phenotypically novel traits". This supports the view that including physical matter in the phenotype increases the chance of mutation introducing novelty and therefore enhancing evolvability.

Both the FPGA and LCD are devices designed for other uses and it remains unclear whether these materials are particularly suitable for evolution-in-materio [17]. In addition, computational problems previously solved using these approaches have not been standard computational benchmarks.

In this paper, we describe the use of a purpose built platform called Mecobo that facilitates computer controlled evolution of a material [14]. The Mecobo platform has been developed within an EU funded research project called NASCENCE [3]. The computational material we have used in this investigation is a mixture of single-walled carbon nanotubes and a polymer. This new platform allows a variety of materials to be investigated in custom designed electrode arrays, using a variety of electrical signals and inputs. In other recent work, the new approach has been applied to solving traveling salesman problems [4], and classification problems [19]. Here, we apply the technique, for the first time to the benchmark problem of function optimization. We also apply Mecobo with the electrode-array to the tone-discriminator problem using same pairs of frequencies that Harding et al. used in the experiments with a liquid crystal display.

Evolutionary computation has been widely used to solve complex multi-modal optimization functions. Many of these functions have been specifically designed to be difficult and deceptive landscapes for search algorithms. The genotype representations that are used in evolutionary algorithms that attempt to solve these kinds of problems are direct. Usually it is a vector whose elements represent the independent variables of the optimization function. In the work we report here, we use a complex genotype-phenotype mapping in which configurations of materials are manipulated and measured outputs can be mapped to the domain vector of these optimization functions. The thinking behind this is that such complex-genotype-phenotype in general may provide more evolvable mappings than highly constrained simulated representations [1].

We show that using the Mecobo platform it is indeed possible to evolve solutions to benchmark function optimization problems. This is the first time EIM has been used to solve function optimization problems. At this stage, our aim is not to claim EIM as a particularly competitive method for solving function optimization problems, we are simply trying to apply EIM to standard computational benchmark problems so that we have a yardstick to assess various aspects of EIM using the Mecobo platform and to reveal the potential that EIM offers. When tackling a computational problem using a material system obvious questions emerge. For instance, what type of signals are appropriate? What materials give the best results. We also investigate the tone discriminator problem using many pairs of square waves of different frequencies. We have used the same pairs of frequencies that Harding et al. used in their experiments [6]. Our results indicate that single-walled carbon nanotube (SWCNT)/polymer substrate gives better results than with a LCD. We also repeated the tone discriminator experiment with different mixtures of material and different organizations of electrode array. In this way we can examine which mixtures of material and electrode array design provide better solution. Using materials in the genotype-phenotype map has, at present, some drawbacks. The main one is that it is slow (see later) this means that we can only feasibly evaluate relatively few potential solutions. However, it is a new approach to the solution of computational problems and as the technology is developed it could of-

[1] Digital chips are designed to emulate, as far as possible, a device that operates using Boolean algebra.

fer advantages over conventional computational methods [17].

To fairly assess the feasibility of using the EIM for function optimization we compared its performance with a software-based evolutionary technique using the same number of function evaluations. To do this we have compared its performance with that of Cartesian Genetic Programming (CGP) on the same set of optimization benchmarks. We do not claim that CGP is competitive with the state-of-the-art in function optimisation however, published results have merely shown that it is a reasonably effective technique for function optimization [18]. In addition, it has been used in as close as manner as possible to the way the material sample has been used. Like CGP, the dimensionality of the evolved vectors is not directly related to the dimensionality of the domain of the optimisation function.

The organization of the paper is as follows. In Section 2 we give a conceptual overview of EIM. We describe the Mecobo EIM hardware platform in Section 3. The preparation and composition of the physical computational material is described in Section 4. Section 5 describes the function optimization problem. The way we have used the Mecobo platform for function optimization is described in Section 6. We describe the function optimization experimental results and analysis in Section 7. Section 8 describes the tone discriminator problem and the way we have used the Mecobo platform to solve it. We describe our tone discriminator experiments and analysis of the results in Section 8.5. Finally, we conclude and offer suggestions for further investigation in Section 9.

2 Conceptual Overview Of Evolution-In-Materio

EIM is a hybrid system involving both a physical material and a digital computer. In the physical domain there is a material to which physical signals can be applied or measured. These signals are either input signals, output signals or configuration instructions. A computer controls the application of physical inputs applied to the material, the reading of physical signals from the material and the application to the material of other physical inputs known as physical configurations. A genotype of numerical data is held on the computer and is transformed into configuration instructions. The genotypes are subject to an evolutionary algorithm. Physical output signals are read from the material and converted to output data in the computer. A fitness value is obtained from the output data and supplied as a fitness of a genotype to the evolutionary algorithm

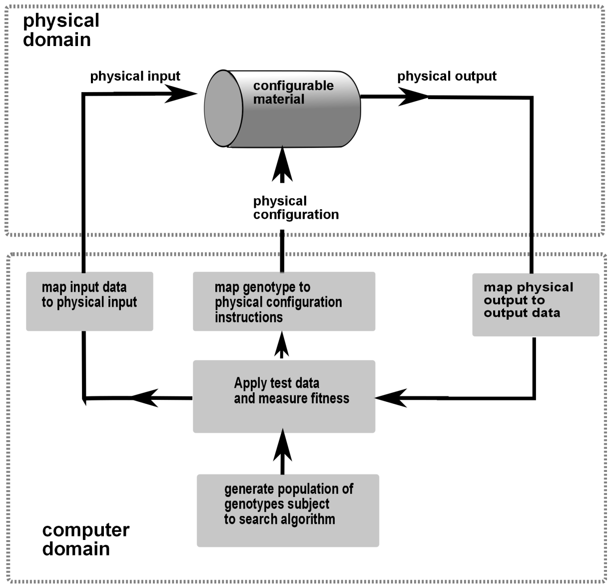


Fig. 1: Concept of evolution-in-materio [17].

[17]. The conceptual overview of EIM is shown in Figure 1.

Not all materials may be suitable for EIM. Miller and Downing suggested some guidelines for choosing materials. The material needs to be reconfigurable, i.e., it can be evolved over many configurations to get desired response. It is important for a physical material to be able to be "reset" in some way before applying new input signals on it, otherwise it might preserve some memory and might give fitness scores that are dependent on the past behaviour. Preferably the material should be physically configured using small voltage and be manipulable at a molecular level [16, 17].

3 Mecobo: An Evolution-In-Materio Hardware Platform

The Mecobo platform is designed to interface a large variety of materials. The hardware allows for the possibility to map input, output and configuration terminals, signal properties and output monitoring capabilities in arbitrary ways. The platform's software component, i.e. EA and software stack, is as important as the hardware. Mecobo includes a flexible software platform including hardware drivers, support of multiple programming languages and a possibility to connect to hardware over the internet [14].

It is important to appreciate that in EIM the computational substrate is piece of material for which the appropriate physical variables to be manipulated by evolution may be poorly understood (see Fig 1). This

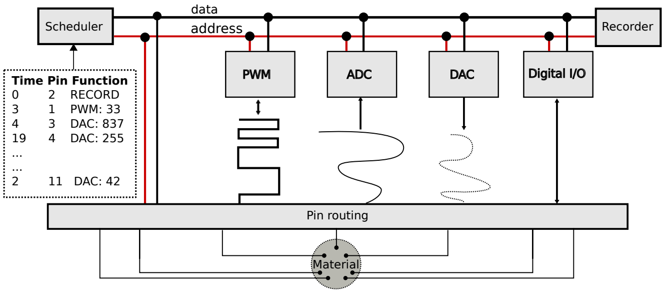


Fig. 2: Overview of the complete system.

means that the selection of signal types, i.e. inputs, outputs and configuration data, assignment to I/O ports might easily not cause sufficient response from the material. Thus interactions with the materials should be as unconstrained as possible. This means that any I/O port should be allowed by the hardware to accept any signal type. In addition, the signal properties, e.g. voltage/current levels, AC, DC, pulse or frequency, should be allowed to be chosen during evolution. The Mecobo hardware interface is designed to handle all these features. Many computational problems require input data so the interface in the Mecobo platform has been designed to allow user-defined external input data signals.

Figure 2 shows an overview of the hardware interface. In the figure an example set up is shown in the dotted box. The example genome defines pin 2 to be the output terminal, pin 1 to be the data input and pin 3 - 12 to be configuration signals. The architecture is controlled by a scheduler controlling the following modules: Digital I/O can output digital signals and sample responses. Analogue output signals can be produced by the DAC module. The DAC can be configured to output static voltages or any arbitrary time dependent waveform. Sampling of analogue waveforms from the material is performed by the ADC. Pulse Width Modulated (PWM) signals are produced by the PWM module.

The system's scheduler can set up the system to apply and sample signals statically or produce time scheduled configurations of stimuli/response. The recorder stores samples, digital discrete values, time dependent bit strings, sampled analogue discrete values or time dependent analogue waveforms. Note that the recorder can include any combination of these signals.

In the interface all signals pass a crossbar, i.e. pin routing. Pin routing is placed between the signal generator modules and the sampling buffer (PWM, ADC, DAC, Digital I/O and Recorder) making it possible to configure any terminal of a material to be input, output or receive configuration signals.

The material signal interface presented in Figure 2 is very flexible. It not only allows the possibility to evolve the I/O terminal placement but also a large variety of configuration signals are available to support materials with different sensitivity, from static signals to time dependent digital functions. At present, the response from materials can be sampled as purely static digital signals, digital pulse trains. A later version of Mecobo (version 3.5) will allow the direct input and output of analogue signals. Further the scheduler can schedule time slots for different stimuli when time dependent functions are targeted or to compensate for configuration delay, i.e. when materials need time to settle before a reliable computation can be observed.

3.1 Hardware Implementation

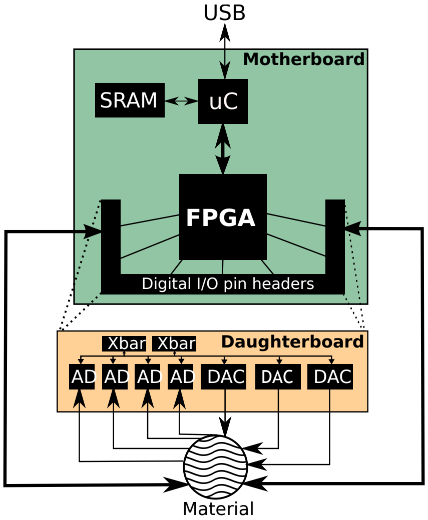
The hardware implementation of the interface is shown as a block diagram in Figure 3(a). Mecobo is designed as a PCB with an FPGA as the main component. The system shown in Figure 2 is part of the FPGA design together with communication modules interfacing a micro controller and shared memory. As shown in Figure 3(a) the digital and analogue designs are split into two. All analogue components are placed on a daughter board; such as crossbar switches and analogue-digital converters. This allows the redesign of the analogue part of the system without changing the digital part of the motherboard. The system shown in Figure 3(a) is an example of the current system. The micro controller stands as a communication interface between the FPGA and the external USB port.

Figure 3(b) shows the motherboard with the Xilinx LX45 FPGA, Silicon Labs ARM based EFM32GG990 micro controller connected to a 12 terminal material sample.

The Mecobo 3.0 hardware platform, that we used only allows only two types of inputs to the material: constant voltage (0V or 3.5V) or a square wave signal. However, different characteristics or input parameters associated with these inputs can be chosen. These input parameters are described in Table 1.

The start time and end time of each input signal determines how long an input is applied. Mecobo 3.0 only samples using digital voltage thresholds, hence the output from the material is interpreted as strictly high or low, (i.e. 0 or 1).

Also, in the case that an electrode is chosen to be read, a user-defined output sampling frequency determines the buffer size of output samples. If the output frequency is F_{out}, and the start time, $Time_{start}$ and end time, $Time_{end}$, are measured in milliseconds then the buffer size is Buf_{size} is given by:



(a) Mecobo block diagram.

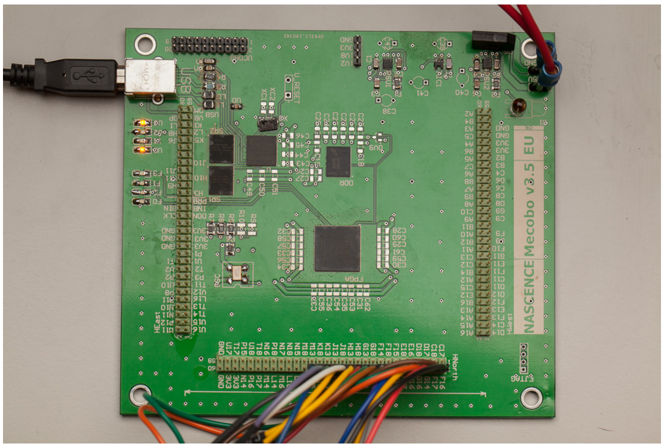

(b) Picture of Mecobo.

Fig. 3: Hardware interface implementation overview.

Table 1: Adjustable Mecobo input parameters.

Parameter Name	Description	Note
Amplitude	0 or 1 corresponding to 0V or 3.5V	wave signal amplitude must be 1
Frequency	Frequency of square wave signal	Irrelevant if fixed voltage input
Cycle Time	Percentage of period for which square wave signal is 1	Irrelevant if fixed voltage input
Phase	Phase of square wave signal	Irrelevant if fixed voltage input
Start time	Start time of applying voltage to electrodes	Measured in milliseconds.
End time	End time of applying voltage to electrodes	Measured in milliseconds.

$$Buf_{size} = F_{out}(Time_{end} - Time_{start})/1000 \qquad (1)$$

However, in practice due to pin latency, the real buffer size is generally smaller.

4 The Computational Material

The experimental material consists of SWCNT mixed with polymethyl methacrylate (PMMA) or polybutyl methacrylate (PBMA) and dissolved in anisole (methoxybenzene) [2]. The sample is baked causing the anisole to evaporate. This results in material which is mixture of SWCNT and PMMA/ PBMA. Different mixtures of material have been used in experiments.

Carbon nanotubes are conducting or semi-conducting and role of the PMMA/ PBMA is to introduce insulating regions within the nanotube network, to create non-linear current versus voltage characteristics. The idea is that this might show some interesting computational behavior. Another benefit of the polymer is to help with dispersion of the nanotubes in solution. The preparation of experimental material is given below:

- 20 μL of material are dispensed onto the electrode array;
- This is dried to leave a "thick film"

The experimental material is placed in the middle of a plate of the electrode array. Two different arrangements of electrode array in slides have been used in the experiments. In one arrangement, a single electrode array is placed on the slide. This is prepared by placing one drop of the experimental material in the middle of the slide. Twelve gold contacts arranged on one side connect with a circular arrangement of twelve electrodes which are connected directly with the material sample. This electrode arrangement is shown in Fig. 4 (Slide A). In another arrangement, two electrode arrays are placed in each slide. One drop of experimental material is placed in the middle of each electrode array. Sixteen gold contacts (eight on each side) are connected to grid electrode arrays under each sample. This electrode arrangement is shown in Fig. 5 (Slide B). The contacts are connected directly with the Mecobo board

[2] Mark K. Massey and Michael C. Petty prepared the materials used as substrates and the electrode masks for our experiments.

Fig. 4: Slide A: Single twelve electrode array (circular) and one material sample.

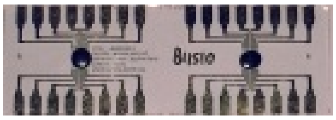


Fig. 5: Slide B: Two sixteen electrode arrays (grid) each with material sample

Table 2: Description of materials used in experiments. Slide electrode arrangements are of type A or B. See Figures 4 and 5 respectively.

Slide Number	Electrode arrangement	Mixture of Material (% wgt. of SWCNT to PMMA/PBMA)
1	B	1.0, PBMA
2	A	1.0, PBMA
3	A	1.0, PMMA
4	A	0.71, PMMA
5	A	0.50, PMMA
6	A	0.10, PMMA
7	A	0.05, PMMA
8	A	0.02, PMMA
9	A	0.01, PMMA
10	A	0.0, PMMA

via a suitable connector. The materials that have been used are described in Table 2.

5 Function Optimization

Benchmark function optimization problem are functions, $f(x_i)$ of a number (n) of real-valued variables, where $i = 1, 2, \ldots n$. The aim is to obtain the values of x_i which cause $f(x_i)$ to be a minimum. In evolutionary computation many complex, multi-modal functions have been designed whose minima are known, but are challenging functions to minimise using search algorithms. The functions are listed below. In the experiments, we have chosen 21/23 benchmark functions (Function 1- 18, 21 - 23) from [23] and 2/23 functions (Function 19 - 20) from [24].

$f_1(x) = \sum_{i=1}^{d} {x_i}^2$
$f_2(x) = \sum_{i=1}^{d} |x_i| + \prod_{i=1}^{d} |x_i|$
$f_3(x) = \sum_{i=1}^{d} (\sum_{j=1}^{i} x_j)^2$
$f_4(x) = max_i\{|x_i|, 1 \leq i < d\}$
$f_5(x) = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
$f_6(x) = \sum_{i=1}^{d} (\lfloor x_i + 0.5 \rfloor)^2$
$f_7(x) = \sum_{i=1}^{d} ix_i^4 + random[0, 1)$
$f_8(x) = \sum_{i=1}^{d} -x_i \sin(\sqrt{|x_i|})$
$f_9(x) = \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i) + 10]$
$f_{10}(x) = -20\exp(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}) - \exp(\frac{1}{d}\sum_{i=1}^{d} \cos(2\pi x_i))$
$+ 20 + e$
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos(\frac{x_i}{\sqrt{i}}) + 1$
$f_{12}(x) =$
$\frac{\pi}{d}\left\{10\sin^2(\pi y_i) + \sum_{i=1}^{d-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_d - 1)^2\right\}$
$+ \sum_{i=1}^{d} u(x_i, 10, 100, 4)\ ;\ \ y_i = 1 + \frac{1}{4}(x_i + 1)$

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \leq x_i \leq a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$$

$f_{13}(x) = \frac{1}{10}\left\{\sin^2(3\pi x_i) + \sum_{i=1}^{d-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})]\right\}$
$+ \frac{1}{10}(x_d - 1)^2[1 + \sin^2(2\pi x_d)] + \sum_{i=1}^{d} u(x_i, 5, 100, 4)$

$f_{14}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right]^{-1}$ where
$a_{1j} = 16(j - 3 - 5\lfloor j/5.1 \rfloor) \quad a_{2j} = 16(\lfloor j/5.1 \rfloor - 2)$

$f_{15}(x) = \sum_{i=1}^{i=11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$
where,

$$a_i = \begin{pmatrix} 0.1957 \\ 0.1947 \\ 0.1735 \\ 0.16 \\ 0.0844 \\ 0.0627 \\ 0.0456 \\ 0.0342 \\ 0.0323 \\ 0.0235 \\ 0.0246 \end{pmatrix}^T, b_i = \begin{pmatrix} 0.25 \\ 0.5 \\ 1 \\ 2 \\ 4 \\ 6 \\ 8 \\ 10 \\ 12 \\ 14 \\ 16 \end{pmatrix}^T$$

$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$
$f_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10(1 - \frac{1}{8\pi})\cos x_1 + 10$

$f_{18}(x) =$
$\left[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right]$
$\times \left[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$

$f_{19}(x), f_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{d} a_{ij}(x_j - p_{ij})^2\right]$
where,
$c_i = \begin{pmatrix} 1 & 1.2 & 3 & 3.2 \end{pmatrix}$ and for f_{19}, $d = 3$ and

$$a_{ij} = \begin{pmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix} \quad p_{ij} = \begin{pmatrix} 0.3689 & 0.1170 & 0.2673 \\ 0.4699 & 0.4387 & 0.7470 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.038150 & 0.5743 & 0.8828 \end{pmatrix}$$

while for f_{20}, $d = 6$ and

$$a_{ij} = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}$$

$$p_{ij} = \begin{pmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.991 \\ 0.2348 & 0.1415 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{pmatrix}$$

$f_{21}(x), f_{22}(x), f_{23}(x) = -\sum_{i=1}^{m}\left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$
with m =5,7,10 for f_{21}, f_{22} and f_{23}, respectively, and,

$$c_i = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.5 \end{pmatrix}^T, a_{ij} = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 1 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{pmatrix}$$

The dimension, d and the intervals over which the benchmark optimization functions are defined are seen in Table 3. The optima are shown with the results in Table 6.

Table 3: Benchmark optimization functions: dimensions and intervals.

f_i	Interval	f_i	Interval
1	$[-100,100]^{30}$	2	$[-10,10]^{30}$
3	$[-100,100]^{30}$	4	$[-100,100]^{30}$
5	$[-30,30]^{30}$	6	$[-100,100]^{30}$
7	$[-1.28,1.28]^{30}$	8	$[-500,500]^{30}$
9	$[-5.12,5.12]^{30}$	10	$[-32,32]^{30}$
11	$[-600,600]^{30}$	12	$[-50,50]^{30}$
13	$[-50,50]^{30}$	14	$[-65.536,65.536]^{2}$
15	$[-5,5]^{4}$	16	$[-5,5]^{2}$
17	$[-5,10]\times[0.15]$	18	$[-2,2]^{2}$
19	$[0,1]^{4}$	20	$[0,1]^{6}$
21	$[0,10]^{4}$	22	$[0,10]^{4}$
23	$[0,10]^{4}$		

Table 4: Description of genotype.

Gene Symbol	**Signal applied to, or read from i^{th} chromosome and j^{th} electrode**	**Allowed values**
$p_{i,j}$	Which electrode is used	$0, 1, 2 \ldots 11$
$s_{i,j}$	Type	0 (constant) or 1(square-wave)
$a_{i,j}$	Amplitude	0 , 1
$f_{i,j}$	Frequency	500 ,501 . . . 10K
$ph_{i,j}$	Phase	$1, 2 \ldots 10$
$c_{i,j}$	Cycle	$0, 1, \ldots 100$

6 Optimizing Functions Using Evolution-In-Materio

6.1 Methodology

The experiments were performed with slide 4 (see Table 2) with an electrode array having twelve electrodes. This sample was chosen somewhat nominally but partly because its weight of carbon nanotubes as a fraction of polymer was mid range in the material samples.

In the experiment, reported here, one electrode has been used as output and the remaining electrodes have been used as configuration voltages. No inputs were needed. The configuration voltages affect the electrical behaviour of the carbon nanotube-polymer material and the interaction induces certain voltages on the output electrode. It is this unknown mapping that is being exploited by computer-controlled evolution.

We read a series of output values (0 or 1) from a buffer of samples taken from a single electrode. These values were used to define the value of a variable 'x_i' in function optimization problem (see Section 5). As the optimization functions have more than one dimension, more than one output from the device is needed. To deal with this, a split genotype technique has been used. In this technique, we used a genotype consisting of multiple chromosomes, each of which was applied to eleven electrodes. On each application of a chromosome we read the binary values in an output buffer of samples from the remaining electrode. For instance, for a 30 variable optimization problem we used 30 chromosomes. Each chromosome defined which electrode would be read and which electrodes would receive the configuration data (square waves or constant voltage). There can be many other ways of using the twelve electrodes. We could have read two electrodes and applied evolved configuration data to the remaining ten and other choices are possible. Examining other choices remains for future work.

Using the Mecobo platform we can control the time (in milliseconds) that a signal is applied to the material (see Section 3). Here, we accumulated output values in a buffer for 128 milliseconds. The number of samples of output buffer can be controlled by start time, end time and frequency of output electrode. In experiment, we used a 25KHz buffer sampling frequency.

6.2 Genotype Representation

Each chromosome used $n_e = 12$ electrodes at a time. Associated with each electrode there were six genes which either define which electrode was used as an output, or characteristics of the input applied to the electrode: signal type, amplitude, frequency, phase, cycle (see Section 3). So each chromosome required a total of 72 genes and a genotype of 30 chromosomes required a total of 2106 (72x30) genes. Mutational offspring were created from a parent genotype by mutating a single gene (i.e one gene of 2106). The values that genes can take are shown in Table 4. The chromosome index, i takes values $0, 1, \ldots d-1$, where d is the number of dimensions of the function optimization problem and the electrode index, j takes values $0, 1, \ldots n_e - 1$.
The i^{th} chromosome, C_i is defined by:

$$C_i = p_{i,0}s_{i,0}a_{i,0}f_{i,0}ph_{i,0}c_{i,0} \ldots p_{i,11}s_{i,11}a_{i,11}f_{i,11}ph_{i,11}c_{i,11}$$

The genotype for a d dimensional problem is a collection of d chromosomes: $C_0C_1 \ldots C_{d-1}$

For solution with 1 output and 11 configuration voltages, the last 6 gene values of i^{th} chromosome are related to the output. These are: $p_{i,11}s_{i,11}a_{i,11}f_{i,11}ph_{i,11}c_{i,11}$

In these output genes, only the first $p_{i,11}$ has any effect, the remainder are redundant. The gene $p_{i,11}$ decides which electrode will be used for the output of the device. Thus, mutations in this gene can choose a different electrode to be used as an output.

6.3 Output Mapping

To determine a real-valued output from a collection of ones in an output buffer it was decided to use the fraction of ones. However, initial findings revealed that the output buffer never contained more than 40% ones. As a result, before the function optimization experiment, an initial evolutionary investigation was performed to discover the typical contents of an output buffer under various conditions. The fraction of number of ones in the output buffer was calculated to obtain the values of the variable required to optimize functions. However, because the buffer contained a maximum of 40% ones, the fraction of ones was multiplied by 2.5 so that a real-valued output would take values between 0 and 1. We denote this value by q.

In the initial investigation, evolutionary runs were carried out to find the electrode configurations (which electrode is used as output or configuration voltage, signal type, amplitude, phase, cycle, frequency of configuration voltages) that gave different percentages of ones in the output buffer. The different percentages were 0%, 10%, 20%, 30% and 40%. The evolved electrode configurations that gave these percentages were used to seed the initial populations for the evolutionary runs for the function optimization problems.

These real values determined from the fraction of ones in the output buffer were linearly mapped to the ranges that particular variables were allowed to take in various optimization functions. This was done as follows. Let, max_i and min_i be the ranges allowed for a variable, x_i in a function optimization problem. Then the equation used for calculating the linearly mapped output value, x_i is given by:

$$x_i = min_i + (max_i - min_i)q \tag{2}$$

The linearly mapped output values x_i were determined corresponding to each chromosome to obtain the measured vector minimizing the optimization function.

7 Experiments

Twenty-three benchmark functions of function optimization problem were investigated. A $1+\lambda-ES$, evolutionary algorithm with $\lambda = 4$ was used [15] and run for 5000 generations. The $1+\lambda-ES$ evolutionary algorithm has a population size of $1+\lambda$ and selects the genotype with the best fitness to be the parent of the new population. The remaining members of the population are formed by mutating the parent. The experiment was performed over 10 independent runs for each benchmark function. Only 10 runs were undertaken as it took over 7 days for these experiments. Different functions took different time due to different number of dimensions. The elapsed time increased with the number of dimensions.

7.1 CGP Experimental Details

To evaluate the effectiveness of the EIM method for solving function optimization problems we compared results with Cartesian Genetic Programming using a $1 + 4$ evolutionary algorithm over the same number of generations [3].

CGP is a graph-based form of genetic programming [15]. The genotypes encode directed acyclic graphs and the genes are integers that represent where nodes get their data, what operations nodes perform on the data, and where the output data required by the user is to be obtained. Five constant inputs (terminals) are generated randomly in the interval [-1, 1] at the start of each evolutionary run. The function set chosen for this study is defined over the real-valued interval [-1.0, 1.0] and is shown in Table 5. CGP creates a graph of mathematical operations that maps the random constants (inputs) to a vector of real-values that define the domain of the benchmark optimisation problem. Likewise the material maps evolved inputs into output values that are used to form a vector used to define the domain vector of the optimisation function. The key similarity here is that both techniques *build* a real-valued vector rather than evolving directly.

The number of outputs is $n_o = d$, where d is the dimensionality of the optimization problem. Since the terminals and functions all return numbers in the interval [-1, 1] the program outputs, q_i also have values defined in this range. However, as noted previously, the optimization functions are defined over a variety of intervals. Thus the program outputs, q_i, need to be mapped to the intervals defined in the optimization problem, x_i [18]. Equation 3 gives the mapping.

$$x_i = \frac{max_i - min_i}{2} q_i + \frac{max_i + min_i}{2}. \tag{3}$$

We used *three* mutation parameters. A probability of mutating connections, μ_c, functions, μ_f and outputs, μ_o. In all experiments $\mu_c = 0.01$, $\mu_f = 0.03$, and $\mu_o = 0.04$.

[3] In both cases of experimental material and CGP, offspring replaced parents if their fitness was greater than *or equal to* the parent.

Table 5: Node function gene values and their definition.

Value	Definition
0	$\sqrt{\lvert z_0 \rvert}$
1	${z_0}^2$
2	${z_0}^3$
3	$(2exp(z_0+1)-e^2-1)/(e^2-1)$
4	$\sin(z_0)$
5	$\cos(z_0)$
6	$\lvert z_0 \rvert^{\lvert z_1 \rvert}$
7	$\sqrt{({z_0}^2+{z_1}^2)/2}$
8	$(z_0+z_1)/2$
9	$(z_0-z_1)/2$
10	$z_0 z_1$
11	**if** $\lvert z_1 \rvert < 10^{-10}$ **then** 1 **else if** $\lvert z_1 \rvert > \lvert z_0 \rvert$ **then** z_0/z_1 **else** z_1/z_0
12	**if** $z_0 > z_1$ **then** $z_2/2$ **else** $1 - z_2/2$

We chose a linear CGP geometry by setting the number of rows, $n_r = 1$ and the number of columns, $n_c = 100$ with nodes being allowed to connect to any previous node.

7.2 Analysis of Results

The results of the two sets of experiments are compared using the non-parametric two sided Mann-Whitney U-test and also the two-sample Kolmogorov-Smirnov (KS) test [11]. We also computed the effect size statistic[22]. A U or KS test value of < 0.05 indicates that the difference between two dataset is statistically significant. The effect size, A value shows the importance of this difference considering the spread of the data; with values $A < 0.56$ showing small importance, $0.56 <= A < 0.64$ medium importance and $A >= 0.64$ large importance. Therefore if a comparison between results is shown to be statistically significant with a medium or large effect size, then we can be reasonably sure that any difference is not due to under sampling.

The experiments show that in 7/23 functions the best results with the experimental material are equal to optimum results and in case of 11/23 functions the best results are very close to optimum results. In 4 cases the average results with the experimental material are equal to optimum results and in 11 cases the average results are within 4% of the optimum results. In 10/23 functions the best results of experimental material are better than or equal to the best results of CGP. In case of 6/23 functions the average results of experimental material are statistically better or at least equal to the average results of CGP. For details see Table 6.

It should be noted that CGP has been compared [4] with Differential Evolution (DE), Particle Swarm Optimization (PSO), Evolutionary Algorithm (SEA). Comparisons showed that in 15/20 benchmarks CGP is same or better than DE, in 19/20 cases CGP is same or better than PSO or SEA [18].

8 Discriminating Tones

The tone discriminator is a device which takes two different (different frequencies) signals as inputs and returns a different response for each of the input signals. It was first described in the work of Thompson [20], and later on Harding and Miller applied evolution-in-materio to solve tone discriminator problems using an LCD by many pairs of frequencies [6, 5]. The same pairs of input frequencies have been used here to compare with the results of Harding et al.'s tone discriminator experiments. It should be noted that in the tone discriminator used by Thompson and Harding et al. a single output was used and the idea was that the voltage should go high when a high frequency square wave was presented and low when a low frequency signal was presented. In our experiments we used two outputs to decide whether an incident square wave signal was the higher or lower of the two frequencies presented.

8.1 Methodology

Nine different sets (sets A-I) of tone discriminator experiments have been performed. All of these experiments were performed with Mecobo 3.0. In the first set of tone discriminator experiments (A), possible the same pairs of frequencies as in [5] were investigated using slide 4 (according to Table 2). Mecobo does not support frequencies lower than 500 Hz, so no frequency lower than 500 Hz was tried in any of the tone discriminator experiments here. Harding et al. tried 114 pairs of frequencies ranging from 500 Hz to 4500 Hz. The experiments of set A used 119 pairs of frequencies, of these 114 pairs are the same as used by Harding et al. However, in experiments B-H, only 45 pairs have been used in order to reduce the number of experiments. The 45 pairs were selected in a way so that they cover most of the parts of the full distribution of 119 pairs. All experiments were performed using the evolutionary process.

The experiments of set A were performed to compare the performance of a mixture of single-walled carbon nanotubes and a polymer against the performance

[4] Based on average results over 30 independent runs, and 500000 evaluations for each run.

Table 6: Comparative results of experimental material with CGP on 23 benchmark optimization functions. The best and average results for both CGP and experimental material are computed from 10 independent evolutionary runs. All results are for 5000 generations. The "Res." column shows whether the results of experimental material is equal to or close to optimum or not. '✓' indicates the result is equal to or close to optimum and 'X' indicates the result is not close to optimum. The first results of this column are according to the best results of experimental material and the second results are according to the average results of experimental material. The "Com. Res." column shows the comparison between the best and average results of the experimental material with CGP. The '+' indicates the result with the material is equal or better than the result of CGP and '-' indicates the result of experimental material is worse. The first result of this column shows comparison of best results and second result of this column shows comparison of average results. "U-Test", "KS-Test" and "Effect Size" columns show results of statistical significance test (L = large, M = medium, S = small). The statistical significance tests are performed over the results of all 10 runs of all 23 functions. '✓' of "U-Test", "KS-Test" columns indicates that the difference between the two datasets is statistically significant and 'X' indicates that the difference is not statistically significant. 'n/a' of "U-Test" column indicates that the test is not useful for the datasets. In these cases all evolutionary runs in both cases (in-materio and CGP) converged to the known global optimum.

No.	Expected Output	Best Results of Experimental Material	Average Results of Experimental Material	Best Results of CGP	Average Results of CGP	Res.	Com. Res.	U-Test (<0.05)	KS-Test (<0.05)	Effect Size
1	0	1.547937E-05	3.261765E-05	0	0	✓ ✓	- -	✓	✓	L
2	0	1.013977E-02	2.372435E-02	0	0	✓ ✓	- -	✓	✓	L
3	0	127.902	1614.87	3.938	3.064647E+03	X X	- +	X	X	L
4	0	2.038043E-01	5.600358E-01	0	0	✓ ✓	- -	✓	✓	L
5	0	1.136536E-01	3.871166E-01	27.690461	37.668384	✓ ✓	+ +	✓	✓	L
6	0	0	0	0	0	✓ ✓	+ +	n/a	X	S
7	0	3.813224E-03	1.071498E-02	1.398843E-03	1.082964E-02	✓ ✓	- +	X	X	S
8	-12569.487	-12451.028	-12255.608	-12569.450	-12569.330	✓ ✓	- -	✓	✓	L
9	0	2.992018	5.634806	0	0	X X	- -	✓	✓	L
10	0	1.166181E-02	3.490709E-02	1.439820E-16	1.439820E-16	✓ ✓	- -	✓	✓	L
11	0	3.395043E-03	8.345206E-02	0	0	✓ ✓	- -	✓	✓	L
12	0	1.047303E-01	1.151369E-01	3.072479E-01	6.513926E-01	✓ ✓	+ +	✓	✓	L
13	0	4.336251E-05	1.587237E-04	7.415736E-06	1.176390E-03	✓ ✓	- +	X	X	S
14	0.9980038	0.9980038	0.9980038	0.9980038	0.9980038	✓ ✓	+ +	n/a	X	S
15	0.0003075	3.084965E-04	3.430931E-04	4.437455E-04	1.059346E-03	✓ ✓	+ +	✓	✓	L
16	-1.0316284	-1.0316284	-1.0316229	-1.0316284	-1.0307720	✓ ✓	+ +	X	X	M
17	0.397887	0.397887	0.397916	0.397887	0.397994	✓ ✓	+ +	X	X	M
18	3.0	3.0	21.906419	3.0	3.0	✓ X	+ -	✓	X	L
19	-3.86	-3.86	-3.86	-3.86	-3.86	✓ ✓	+ +	n/a	X	S
20	-3.32	-3.32	-3.32	-3.32	-3.286059	✓ ✓	+ +	n/a	X	S
21	-10.153196	-5.100772	-5.100771	-10.153199	-8.636950	X X	- -	✓	✓	L
22	-10.402819	-5.128822	-5.128822	-10.402864	-8.820119	X X	- -	✓	✓	L
23	-10.536284	-5.175645	-5.175644	-10.536290	-9.463113	X X	- -	✓	✓	L

of an LCD, i.e. the experimental results of set A were compared with the results of Harding's experiments using 114 pairs of frequencies. In tone discriminator experiments using LCD, the number of runs was 5 and each input signal was sent to the material for 250 milliseconds, so for comparing the performances, the number of runs of each of the tone discriminator experiments was 5 and the input-output timing was 250 milliseconds. The experimental settings of all sets of tone discriminator experiments are described in Table 7 and the motives for performing the tone discriminator experiments are also described in Table 8.

All of these experiments were performed with electrode arrays having 12 electrodes. However, in the case of slide 1, one electrode array was used, where only 12 electrodes were used from the 16 electrodes of that electrode array, these were the middle 6 electrodes from each side of one sample.

For all experiments, one electrode was used for inputting the signal to be discriminated, two electrodes were used as outputs and nine electrodes were used as configuration inputs. Each chromosome defined which electrodes were outputs, inputs (received square waves) or received the configuration inputs (square waves or static voltages). The frequency applied to the input electrode was the input frequency to be discriminated. The cycle time of input square wave signal was set to 50% and its amplitude was set to one (i.e. 3.5 V).

8.2 Genotype Representation

Each chromosome used $n_e = 12$ electrodes at a time. The values that genes could take are shown in Table 4. The genotype representation for the experiments is almost identical to that used in the function optimisation experiments with the exception that phase was not used and only one chromosome was used (so i is 0 and can be ignored).

Mutated children were created from a parent genotype by mutating a single gene (i.e. one gene of 60).

Table 7: The experimental settings of all sets of tone discriminator experiments. All of these experiments were performed by Mecobo 3.0. The second column shows the number of pairs of frequencies on which the experiments were performed. The third column shows the slide number (according to Table 2). All of these experiments used 12 electrodes of the electrode array, a 25 KHz output sampling frequency and 250 milliseconds for the input-output timing. The number of runs was 5 and the number of generations was 500. However, the evolutionary run was terminated if fitness score gave 100% correct result.

Set	Number of pairs of frequencies	Slide
A	119	4
B	45	1
C	45	2
D	45	3
E	45	5
F	45	6
G	45	7
H	45	8
I	1	1

Table 8: The motives behind the experimental comparisons for the tone discriminator experiments. The first column shows the sets of experiments. The second column shows the motives.

Experiments	Motive
Set A	Comparison of results using a mixture of single-walled carbon nanotubes with a polymer against the results using an LCD.
Sets B, C	Comparison of results using different organisations of electrodes. (same mixtures of material, but organisations of electrodes are different)
Sets C, D	Comparison of results using different polymers (the same percentage of single-walled carbon nanotubes, but in different polymers).
Sets A, D-H	Comparisons of results using different percentages of single-walled carbon nanotubes in PMMA.

Table 9: Description of genotype for tone discriminator experiments. The "No. of gen. in each elec." column shows the number of genes associated with each electrode. The "Gen. ass. with each elec." column shows the genes that are associated with each electrode. The "Total no. of genes" column shows the total number of genes in each genotype. The "Genotype representation" column shows the representation of a genotype. The "Genes related to inputs" column shows the gene values of a genotype, that are related to inputs. The "Genes related to outputs" column shows the gene values of a genotype, that are related to outputs.

No. of gen. in each elec.	Gen. ass. with each elec.	Total no. of genes	Genotype representation	Genes related to inputs	Genes related to outputs
5	$p_j, s_j, a_j,$ f_j, c_j	12X5 =60	$p_0s_0a_0f_0c_0 \ldots$ $p_{11}s_{11}a_{11}f_{11}c_{11}$	First 5 genes: $p_0s_0a_0f_0c_0$	Last 10 genes: $p_{10}s_{10}a_{10}f_{10}c_{10}$ $p_{11}s_{11}a_{11}f_{11}c_{11}$

In the input and output genes, only the p_j (here the values of j are 0, 10, 11) has any effect, others do not have any effect. The gene p_j decides which electrode will be used for the inputs and outputs of the device. Thus, mutations in these genes can choose a different electrode to be used as an input or output.

8.3 Output Mapping

The output was determined using the average transition gap by examining the output buffers of output electrodes. Mecobo 3.0 can only recognise binary values, so the output buffers contain bitstrings. So, in tone discriminator experiments, the *transitions* from 0 to 1 in the output buffers were used to calculate the class that an instance belongs to. For each output buffer, the positions of transitions were recorded and the gaps between consecutive transitions were measured and an average calculated. A transition-based mapping was used as it is frequency related. Since instance data affects frequencies of applied signals, it seemed natural to use the method of reading output buffer bitstrings, which is itself frequency related. An example of average gap calculation for an output electrode is shown in Figure 6

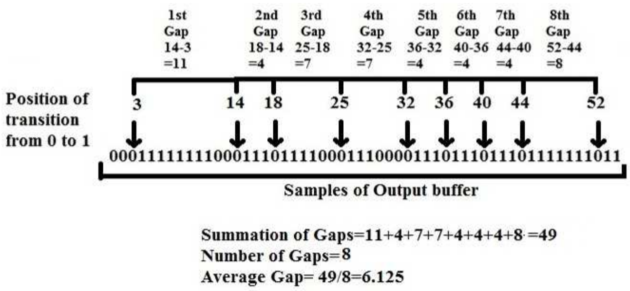


Fig. 6: An example of average transition gap calculation for an output electrode

The tone discriminator problem was interpreted as two-class problem. Each output was associated with a particular class. The class associated with an output electrode was determined by the output buffer with the *lower* average transition gap. If the contents of the buffer from the first output electrode had the lower average transition gap, it was designated to be class one (low frequency), otherwise it was designated to be class two (high frequency). So, the buffer contents from the first output electrode were expected to have the lower average transition gap only if the input frequency was low frequency.

Thus, if the tone discriminator works as desired, it would have class one when the first electrode buffer has the lower average transition gap whenever the input frequency is low. It would have class two when the first electrode buffer has the higher average transition gap at the time when input frequency is high.

It should be noted that the output was decided to be class one in the case that both output buffers had the same average transition gap.

8.4 Fitness Score

The fitness score was measured here using the similar method used by Harding and Miller in their evolution-in-materio experiments to solve tone discriminator problems [6, 5]. The fitness calculation for tone discriminator experiments is described as follows:

Let, S is the vector containing the input sample and L is the length of S. The elements of the two-component vector, O decides the output class at a given time. The value of S can be either HIGH or LOW. The i^{th} element of the input sample is $S[i]$ and output vector is $O[i]$. The elements of the two component vector, x can be decided by the Equation 4.

$$x(i) = \begin{cases} 1, \, S[i] = LOW \text{ and } O[i] = 1 \\ 1, \, S[i] = HIGH \text{ and } O[i] = 2 \\ 0, \, Otherwise \end{cases} \tag{4}$$

The fitness value is calculated using Equation 5.

$$fitness = 100 \frac{\sum_{i=1}^{L} x(i)}{L} \tag{5}$$

In the experiment, L=2, i.e. one input signal is low and another input signal is high.

8.5 Tone-discriminator Experiments and the Results

For each of the experiments, a (1+4)-evolutionary algorithm was used. In experiments, a child replaced the parent if its fitness was greater than or equal to the parent. The total time required for all 5 runs of 119 pairs of frequencies took almost 1 day.

An image is drawn for the first experiment in Figure 7 showing average number of generations to find solutions for 114 pairs of frequencies. The image is then compared with the results of Harding and Miller's experiments with an LCD. Different shades have been used to show average number of generations to find solutions (100% correct result). The shades are from black to red. Black is used for generation number 10 (and less than 10) and red is used for generation number 100 (and more than 100).

A tone discriminator experiment was performed with slide 10, i.e. material with only PMMA (0% single-walled carbon nanotubes) using one pair of input frequencies to investigate whether the single-walled carbon nanotubes are required in the material mixture for computation or not. It has been observed that no evolution took place with this material sample. The measured results from the output electrodes were always zero, and this result was observed over all 100 generations for one run. In further investigations, the results of tone discriminator experiments with the same percentage of single-walled carbon nanotubes (1.0%) (slide 2 and 3) in PBMA and in PMMA have also been compared to discover whether the type of polymer plays any role in computation or not using the results of experiments C and D. Statistical significance tests have

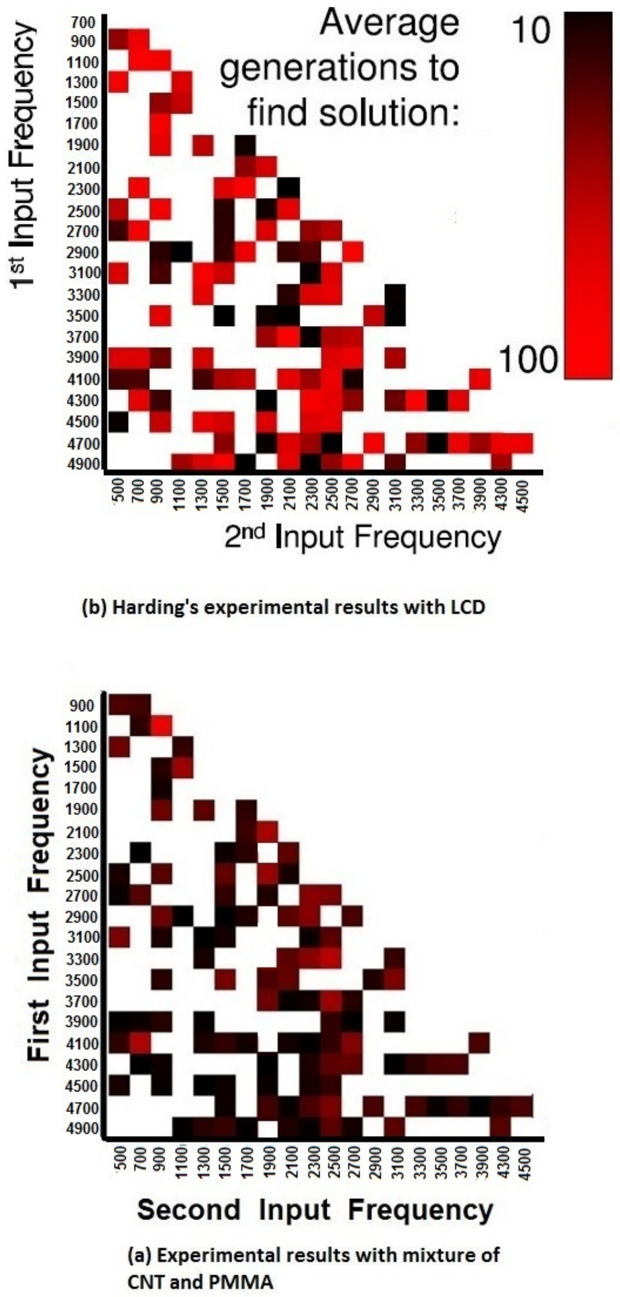


Fig. 7: Graph showing a comparison of results using a mixture of single-walled carbon nanotubes with PMMA against the results using an LCD (a) The experimental results with a mixture of single-walled carbon nanotubes and PMMA. (b) Harding et al.'s experimental results using an LCD [5].

been performed for the comparison using the U-test and KS-test [11]. The effect size statistic [22] has also been computed. The statistical significance tests have been performed over all pairs of frequencies using the average (averaged over 5 runs) number of generations to find 100% correct result for each of the pairs of frequencies. According to U-test and KS-test, the difference of the results is statistically not significant.

As experiments with 0% single-walled carbon nanotubes showed that no evolution happened further investigations were performed using slide 8 and 9 (with 0.02% and 0.01% single-walled carbon nanotubes in PMMA respectively). It has been found from the investigations that no evolution took place when material sample 9 (0.01% single-walled carbon nanotubes in PMMA) was tried and the output buffers contained only zeroes (this experiment was carried out with one pair of frequencies, the number of generations was 100 and the number of evolutionary runs was 1). When an experiment used slide 8 (i.e. with 0.02% single-walled carbon nanotubes in PMMA), the output buffers contained mixtures of 0 and 1 and evolution was possible. Thus single-walled carbon nanotubes are required in the material mixture to enable computation.

Experiments A, D-H were used to compare results obtained by different percentages of single-walled carbon nanotubes in PMMA. Statistical significance tests (U-test, KS-test and effect size) have been performed for comparing the results. The comparison results of tone discriminator experiments with different percentages of single-walled carbon nanotubes in PMMA (slides 3-8 according to Table 2) are shown in Table 10. The best mixture of single-walled carbon nanotubes in PMMA was the material on slide 5 (0.50% single-walled carbon nanotubes in PMMA). This was determined using the average number of generations to find 100% correct results in all 5 runs in the case of all 45 pairs of frequencies (experiments with slide 5 took the least number of generations (26.16) on average of obtaining 100% correct results). An image is shown for tone discriminator experiments with slide 5 in Figure 8 showing the average number of generations (averaged over 5 runs) to find solutions for 45 pairs of frequencies.

Further investigations (experiments B and C) were carried out to see whether different organisations of electrodes with the same mixture of material (1.0% single-walled carbon nanotubes in PBMA) (slide 1 and slide 2) play any role in computation. According to U-test and KS-test, the difference of the results is statistically not significant. The statistical significance tests have been performed over the average number of generations required to give 100% correct results in all 5 runs for 45 pairs of frequencies.

Table 10: Statistical significance tests on results of tone discriminator experiments with different percentages of single-walled carbon nanotubes in PMMA (slides 3-8 according to Table 2). The first column shows the pair of slides (see Table 2). "U-test", "KS-test" and "Effect size" columns show results of the statistical significance tests. The statistical significance tests have been performed over the average number of generations required to give 100% correct results in all 5 runs for 45 pairs of frequencies. '✓' in "U-test", "KS-test" columns indicates that the difference between the two data samples is statistically significant and 'X' indicates that the difference is not statistically significant. It should be noted that the average number of generations required to give 100% correct results in all 5 runs for 45 pairs of frequencies are 31.19, 34.56, 26.16, 28.80, 43.16 and 57.57 for slides 3, 4, 5, 6, 7 and 8 respectively.

Pair of slides	U-test	KS-test	Effect size
Slide 3-Slide 4	X	X	Small
Slide 3-Slide 5	X	X	Small
Slide 3-Slide 6	X	X	Small
Slide 3-Slide 7	✓	✓	Large
Slide 3-Slide 8	✓	✓	Large
Slide 4-Slide 5	X	✓	Medium
Slide 4-Slide 6	X	X	Medium
Slide 4-Slide 7	✓	✓	Medium
Slide 4-Slide 8	✓	✓	Large
Slide 5-Slide 6	X	X	Small
Slide 5-Slide 7	✓	✓	Large
Slide 5-Slide 8	✓	✓	Large
Slide 6-Slide 7	✓	✓	Large
Slide 6-Slide 8	✓	✓	Large
Slide 7-Slide 8	X	X	Medium

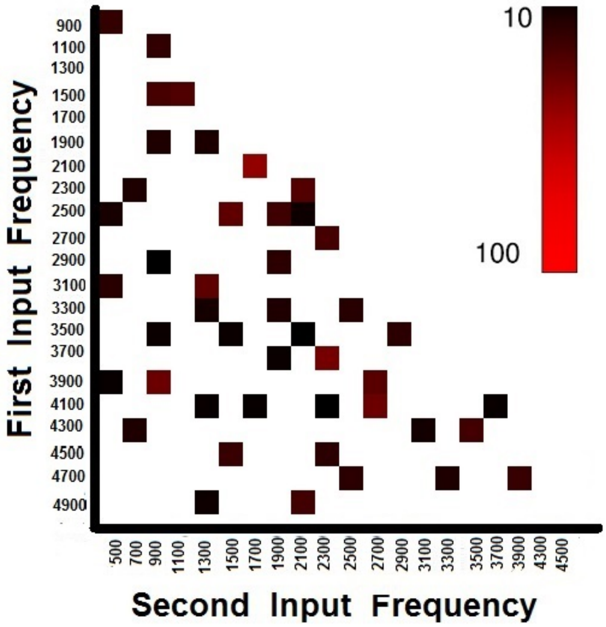


Fig. 8: Graph showing tone discriminator results for 45 pairs of frequencies with slide 5.

9 Conclusions And Future Outlook

Evolution-in-materio is hybrid of digital and analogue computing where digital computers are used to configure materials to carry out analogue computation. This holds the promise of developing entirely new computational devices. A purpose-built evolutionary platform called Mecobo, has been used to evolve configurations of a physical system to obtain the minima of complex, multi-modal mathematical functions. The material used is a mixture of single-walled carbon nanotubes and a polymer. The aim of the paper is not to show that the experimental results of solving function optimization problems using EIM is yet competitive with state-of-the-art function optimization algorithms but rather to start a new beginning in the world of computation. This is the first time it has been shown that such an approach can be used to solve well-known benchmark function optimization problems. In some cases, we found that we could find a solution closer to the global optimum using the EIM approach than an effective software-based evolutionary technique. In the second problem, i.e., the tone discriminator experiment, it has been found that the results are very good and competitive with the results obtained by past attempts using EIM and liquid crystal to solve the tone discriminator problem.

In other work using Mecobo it has been shown that digital logic functions can be implemented [14]. We have also obtained encouraging results on machine learning classification problems [19]. In these cases, in principle, a classifier can be implemented using an electrode array and a material sample on a microscope slide and some interfacing electronics. Such a system could act as a standalone device. This could have advantages of fast classification and low power compared with software based approaches.

Of course, there are many questions for the future. Does evolutionary computation in materio scale well on larger problem instances? What other classes of computational problems are solvable using this technique? What are the most suitable materials and signal types for evolution-in-materio? The Mecobo platform is currently under development and the next version will be able to allow the utilization of analogue voltages. This may make some types of computational problems more readily solved using evolution-in-materio.

Acknowledgements The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 317662.

References

1. Banzhaf, W., Beslon, G., Christensen, S., Foster, J., Képès, F., Lefort, V., Miller, J.F., Radman, M., Ramsden, J.: Guidelines: From artificial evolution to computational evolution: a research agenda. Nature Reviews Genetics **7**, 729–735 (2006)
2. Bird, J., Layzell, P.: An Evolved Radio and its Implications for Modelling the Evolution of Novel sensors. In: Proceedings of Congress on Evolutionary Computation, pp. 1836–1841 (2002)
3. Broersma, H., Gomez, F., Miller, J.F., Petty, M., Tufte, G.: NASCENCE Project: Nanoscale Engineering for Novel Computation Using Evolution. International Journal of Unconventional Computing **8**(4), 313–317 (2012)
4. Clegg, K.D., Miller, J.F., Massey, M.K., Petty, M.C.: Travelling salesman problem solved 'in materio' by evolved carbon nanotube device. In: Parallel Problem Solving from Nature - PPSN XIII - 13th International Conference, Proceedings, *LNCS*, vol. 8672, pp. 692–701. Springer (2014)
5. Harding, S.: Evolution in materio. Ph.D. thesis, University of York (2006)
6. Harding, S., Miller, J.F.: Evolution in materio: A tone discriminator in liquid crystal. In: In Proceedings of the Congress on Evolutionary Computation 2004 (CEC'2004), vol. 2, pp. 1800–1807 (2004)
7. Harding, S., Miller, J.F.: Evolution in materio : A real time robot controller in liquid crystal. In: Proceedings of NASA/DoD Conference on Evolvable Hardware, pp. 229–238 (2005)
8. Harding, S., Miller, J.F.: Evolution in materio. In: R.A. Meyers (ed.) Encyclopedia of Complexity and Systems Science, pp. 3220–3233. Springer (2009)
9. Harding, S.L., Miller, J.F.: Evolution in materio: Evolving logic gates in liquid crystal. International Journal of Unconventional Computing **3**(4), 243–257 (2007)
10. Harding, S.L., Miller, J.F., Rietman, E.A.: Evolution in materio: Exploiting the physics of materials for computation. International Journal of Unconventional Computing **4**(2), 155–194 (2008)
11. Hollander, M., Wolfe, D.: Nonparametric statistical methods. Wiley (1973)
12. Kirschner, M., Gerhart, J.: Evolvability. Proceedings of the National Academy of Science (USA) **95**, 8420–8427 (1998)
13. Layzell, P.: Hardware Evolution: On the Nature of Artificially Evolved Electronic Circuits. Ph.D. thesis, School of Cognitive and Computing Sciences, University of Sussex, UK (2001)
14. Lykkebø, O.R., Harding, S., Tufte, G., Miller, J.F.: Mecobo: A hardware and software platform for in materio evolution. In: O.H. Ibarra, L. Kari, S. Kopecki (eds.) Unconventional Computation and Natural Computation, LNCS, pp. 267–279. Springer International Publishing (2014)
15. Miller, J.F. (ed.): Cartesian Genetic Programming. Springer (2011)
16. Miller, J.F., Downing, K.: Evolution in materio: Looking beyond the silicon box. In: NASA/DOD Conference on Evolvable Hardware, pp. 167–176. IEEE Comp. Soc. Press (2002)
17. Miller, J.F., Harding, S.L., Tufte, G.: Evolution-in-materio: evolving computation in materials. Evolutionary Intelligence **7**, 49–67 (2014)
18. Miller, J.F., Mohid, M.: Function optimization using cartesian genetic programming. In: GECCO (Companion), pp. 147–148 (2013)
19. Mohid, M., Miller, J.F., Harding, S.L., Tufte, G., Lykkebø, O.R., Massey, M.K., Petty, M.C.: Evolution-in-materio: Solving machine learning classification problems using materials. In: Parallel Problem Solving from Nature - PPSN XIII - 13th International Conference, Proceedings, *LNCS*, vol. 8672, pp. 721–730. Springer (2014)
20. Thompson, A.: Hardware Evolution - Automatic Design of Electronic Circuits in Reconfigurable Hardware by Artificial Evolution. Springer (1998)
21. Thompson, A., Layzell, P.: Analysis of unconventional evolved electronics. Communications of the ACM **42**(4), 71–79 (1999)
22. Vargha, A., Delaney, H.D.: A critique and improvement of the cl common language effect size statistics of mcgraw and wong. Journal of Educational and Behavioral Statistics **25**(2), 101–132 (2000)
23. Vesterstrom, J., Thomsen, R.: A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In: Evolutionary Computation, 2004. CEC2004. Congress on, vol. 2, pp. 1980 – 1987 (2004)
24. Yao, X., Liu, Y.: Fast evolutionary programming. In: In L. J. Fogel et al. (Eds.), Proceedings of the 5th Annual Conference on Evolutionary Programming, pp. 451–460. MIT Press (1996)