On some technical issues and challenges in development and implementation of Autonomous Design System's (ADS) functionality

Dmitri Loginov

***Abstract*—** In this paper technical issues and challenges regarding implementation of ADS functionality on AutoCAD platform are discussed. The results of the experiments for adaptation of the synergetic computer algorithm and its basic functionality to AutoCAD environment are presented. Technical aspects of the possible implementation of this algorithm in ADS system, as well as vectors' elements combination/permutation routine performance are analyzed. A brief philosophical outlook on the problem of modeling the engineering creativity in ADS/CAD systems was given.

***Keywords*—** Self-organization, Synergetic computer, ADS, CAD, Pattern recognition, Synergetic neural networks, HVAC

I. Introduction

THE main characteristic of Autonomous Design System (ADS) is the ability to process complex non-routine tasks of the engineering design domain. In our approach the principles of synergetics and synergetic computer are used as a main tool in providing such functionality.

The concept of the synergetic computer was introduced by Hermann Haken in late eighties of the last century and it is based on the profound analogy between pattern formation and pattern recognition. Thus, the mathematical theory of synergetics was possible to use in derivation of the basic equations of synergetic computer. Synergetics (H. Haken's interpretation) can be considered as one of the modern, most promising research programs. It is oriented towards the search for common patterns of evolution and self-organization of complex systems of any kind, regardless of the concrete nature of their elements or subsystems (see e.g. [1], [2]).

In this paper the technical details of the theory of the synergetic computer and its possible realization on CAD (Computer Aided Design) and ADS platforms are discussed.

While the algorithm implemented is based largely on H. Haken models (for other synergetics-based models see [3]), the significant differences to the basic characteristics of the classical model were introduced and successfully tested in one of the most popular CAD environment. For the first time ever the synergetic computer was implemented on AutoCAD platform. The presented research constitutes the next intermediate step to the development and research of the fully functional Autonomous Design System (ADS).

II. Standard Haken Model

In this section a short overview of the mathematical background of the synergetic computer concept is presented. For in-depth discussion of the standard model see [4].

The basic dynamic equation of the synergetic computer or synergetic neural network is as follows:

$$\dot{q} = \sum_{k} \lambda_k v_k (v_k^+ q) - B \sum_{k' \neq k} (v_{k'}^+ q)^2 (v_k^+ q) v_k \\ -C(q^+ q) q + F(t), \tag{1}$$

where q is the state vector of a test (input) pattern with initial value q_0, λ_k is attention parameter, v_k is the prototype pattern vector, v_k^+ is the adjoint vector of v_k, which obeys the orthonormality relation

$$(v_k^+ v_{k'}) = \delta_{kk'}. \tag{2}$$

B, C are positive constants and $F(t)$ is fluctuating forces, which may drive the system out from its equilibrium state. Expression $v_k \cdot v_k^+$ acts as a matrix. This matrix has occurred in number of other publications and is called the learning matrix. The term

$$\xi_k = (v_k^+ q) \tag{3}$$

is called the order parameter. The equation (1) describes the dynamics, which pulls the test pattern $q(t)$ into one of the prototype patterns v_{k_0}, namely the one to which $q(0)$ was closest. This means the pattern is being recognized by the system.

The corresponding dynamic equation of order parameters reads:

$$\dot{\xi}_k = \lambda_k \xi_k - B\sum_{k'\neq k} \xi_{k'}^2 \xi_k - C\sum_{k'=1}^{M} \xi_{k'}^2 \xi_k . \tag{4}$$

The order parameters obey the initial condition

$$\xi_k(0) = (v_k^+ q(0))$$

by which the initial values of order parameters in the evolution series are determined.

The equations (1) and (4) could be derived from corresponding potential function equations (5) and (6). That is

$$\dot{q} = -\frac{\partial V}{\partial q^+},\ \dot{q}^+ = -\frac{\partial V}{\partial q},$$

$$V = -\frac{1}{2}\sum_{k=1}^{M} \lambda_k (v_k^+ q)^2 + \frac{1}{4}B\sum_{k\neq k'} (v_k^+ q)^2 (v_{k'}^+ q)^2 + \frac{1}{4}C(q^+ q). \tag{5}$$

And

$$\dot{\xi}_k = -\frac{\partial \tilde{V}}{\partial \xi_k},$$

$$\tilde{V} = -\frac{1}{2}\sum_{k=1}^{M} \lambda_k \xi_k^2 + \frac{1}{4}B\sum_{k'\neq k} \xi_{k'}^2 \xi_k^2 + \frac{1}{4}C(\sum_{k'=1}^{M} \xi_{k'}^2)^2. \tag{6}$$

The potential function is used to represent the potential field in the space of k order parameters in which the fictitious particle, representing the dynamics of the test pattern or the corresponding order parameter, moves. The example of the potential V is shown on Fig.1.

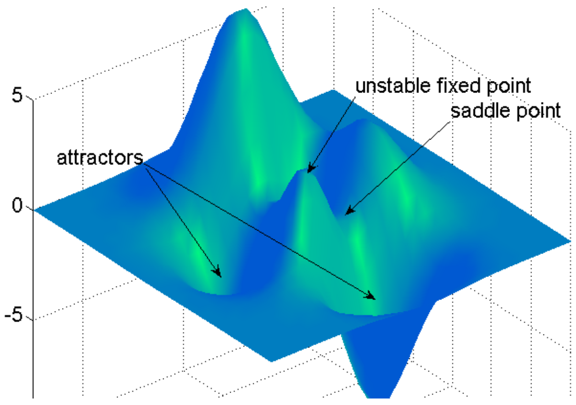


Fig.1 Example of potential function

In this plot the attractors are clearly visible. The attractors are the stable fixed points, which are represented by a bottom of each valley. The top of each mountain is an unstable fixed point. All points in the landscape from which the particle can roll down to the same attractors form the basin of attraction. Points of minimal height on ridges are saddle points.

The stable fixed points are at $q = v_k$, i.e. at the prototype patterns, and there are no other stable fixed points. The stable fixed points are equally characterized by $\xi_k = 1$, all other ξ's = 0.

The Haken's classical model is built up upon a number of assumptions. The most important of which are as follows:

- all attention parameters are equal and positive (i.e. balanced attention parameters)

$$\lambda_k = \lambda > 0, \quad \lambda = C \tag{7}$$

- the number of patterns is smaller than or equal to the number of features

$$M \le N \tag{8}$$

- vectors v_k are subject to the condition

$$\sum_k v_k = 0 \tag{9}$$

- the following normalization holds

$$(v_k^T v_k) \equiv \sum_{j=1}^{N} v_{kj}^2 = 1 \tag{10}$$

- the number of features per pattern should be the same for all prototype and test vectors (equality of vectors' meaningful dimensions). That is, for each

$$v_1(k), v_2(l)...v_n(m) \;;$$
$$k = l = ... = m, \tag{11}$$

where k, l,…,m are vectors' meaningful dimensions.

As vectors v_k are not necessarily orthogonal to each other, we need to construct the adjoint vectors, which may be formed as superpositions of the transposed vectors v_k^T :

$$v_k^+ = \sum_{k'} a_{kk'} v_{k'}^T. \tag{12}$$

The coefficients $a_{kk'}$ must be determined to satisfy the orthogonality condition (2). This may be done by multiplying (12) by $v_{k'}$ and interpreting $a_{kk'}$ and scalar products $(v_{k'}^T v_k)$ as elements of the corresponding matrices A and W

$$A = (a_{kk'})$$
$$W = [(v_{k'}^T v_{k'})].$$

Equation (12) then can be written in the form

$$I = AW \tag{13}$$

and can be solved formally by $A = W^{-1}$.

A. Synergetic neural network

Synergetic computer may be realized by artificial neural networks, which act in a fully parallel manner. The resulting system is then called Synergetic Neural Network or SNN. SNN may be realized e.g. as one- or three-layer network. By using order parameter concept the network can be considerably simplified. As order parameters are defined by (3), and satisfy

$$\dot{\xi}_k = \xi_k(\lambda - D + B\xi_k^2), \tag{14}$$

where

$$D = (B + C)\sum_{k'} \xi_{k'}^2, \tag{15}$$

then, for a three layer network, we may use order parameters' as neurons' representation in the network's second layer. The input layer is represented by input (test) pattern vectors $q_j(0)$, and if SNN has to act as an associative memory, the third layer should consist of

$$q_j(t) = \sum_k \xi_k(t) v_{kj}, \tag{16}$$

where q_j is the activity of the cell j at the output layer, ξ_k is the final state of order parameter cell layer with $\xi_k = 1$ for $k = k_0$ (i.e. the pattern has been recognized) and $\xi_k = 0$ otherwise. The network may be further simplified by introducing a common reservoir D as in (14). In this way the number of connections may be further reduced.

B. Benefits of the synergetic computer (SNN) approach

The classical model has a number of advantages over traditional neural computers (networks). That is, compared to e.g. Hopfield Neural Network (HNN) it has following advantages.

1. The model training time is short.
2. The space complexity is low, for SNN it is np, where n is the number of features, p is the number of patterns and $p \ll n$, while for HNN it is n^2 .
3. The processor time complexity for the recognition process is also low. For SNN it requires p^2 multiplications and p additions and for HNN n^2 multiplications, n additions.
4. There are no so-called *pseudo-states*. This is most important property. It may be proved that besides the prototype vectors there are no other attractors.

In HNN, the system has the following potential functions. In the case of discrete HNN:

$$V = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} w_{ij} v_i v_j + \sum_{i=1}^{N} \theta_i v_i$$

And in the case of continuous HNN:

$$V = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} w_{ij} v_i v_j - \sum_{i=1}^{N} v_i I_i + \sum_{i=1}^{N} \frac{1}{\tau}\int_0^{v_i} g^{-1}(t)dt$$

This potential function can not guarantee that all of the attractors are actually the desired ones. In the construction of HNN, no matter how carefully to learn and adjust the weights w_{ij} and threshold value θ_i , it is still difficult to avoid/control the generation of pseudo-state. In Haken model, considering the system's dynamic behavior, the precise control of the potential function (energy function) in the energy potential field and not the type of connection of neurons nor the non-linear mappings of them, allow thereby to eliminate the pseudo-states.

5. The association effect: all of the prototype patterns can be clearly and equally identified.

C. *Limitations of the synergetic computer*

The limitations of the synergetic computer model, as it often happens, are extensions of its advantages. Namely, the most prominent disadvantage worth noticing is that the system always selects the winning pattern out from the patterns presented, even if they are not actually the right ones. This occurs due to the fact that the biggest order parameter formed initially as a dot product among all available vectors will always win the competition. Therefore, if the vectors' set does not contain the right prototype pattern, it will not be recognized by the system. It is true, however that the system will recognize test pattern (vector) that is the most close to the prototype pattern (vector).

Some problems with correct identification may occur when the vectors have different order of elements (features). This is because the inner (dot) product value's magnitude depends on the order of elements in the sequence (vector). Such identification difficulties may occur in a case of different elements combinations' testing e.g. in points/distances permutation example (see onward in section IV).

III. Synergetic Computer on AutoCAD

Implementation of the synergetic computer core functionality on AutoCAD platform is the intermediate step towards the realization of SNN in real life design process as a main tool in ADS modeling. ADS is defined as an advance CAD system, which *has AI functionality and particularly the functionality to solve the creative tasks* of the engineering design process. ADS is opposed to the conventional CAD systems, (see e.g. [5]) which normally automate routine parts of the design process and generally have no AI capabilities. In this section some technical questions of implementation of synergetic computer basic functionality are discussed.

To this end, the objective was established to create an AutoCAD application that can recognize a number of simple geometric structures. At first the MATLAB prototype was created in order to test the basic functionality of the model and then the algorithm was implemented in AutoCAD environment. For the sake of simplicity of the presentation, in AutoCAD environment only three different patterns were implemented. Actually, the number of patterns tested (in MATLAB) was bigger and the noisy patterns were elaborated as well.

The prototype patterns for our case were chosen among AutoCAD (Acad) polygon entities (more specifically, these constitute of polyline objects in Acad database), namely, the triangle, square and hexagon. As Acad is a vector graphics software, we had to invent the way of representing our prototype vectors properly. We had chosen to code vectors' elements as a relative measure between entities' endpoints i.e. the distances between polygons' vertices, as shown on Fig. 2.

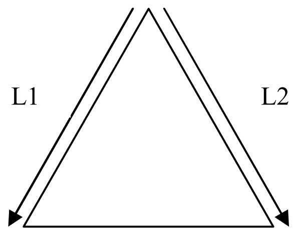


Fig. 2 Representation of prototype vectors' elements in AutoCAD graphics system.

Thus, in a case of triangle the raw prototype vector is as simple as $v_k = (L_1, L_2)$. We have now three state vectors to recognize (shown as raw vectors):

$$\begin{aligned} v_0 &= (L_{01}, L_{02}, L_{03}) \\ v_1 &= (L_{11}, L_{12}) \\ v_2 &= (L_{21}, L_{22}, L_{23}, L_{24}, L_{25}). \end{aligned} \tag{17}$$

From (14) we may deduce a discrete equation for the order parameter evolution

$$\begin{aligned} &\xi_k(n+1) - \xi_k(n) \\ &= \gamma(\lambda_k - D + B\xi_k^2(n))\xi_k(n), \end{aligned} \tag{18}$$

where γ is the iteration speed and term D is according to (15). The corresponding evolution for the case of three order parameter is shown on Fig. 3.

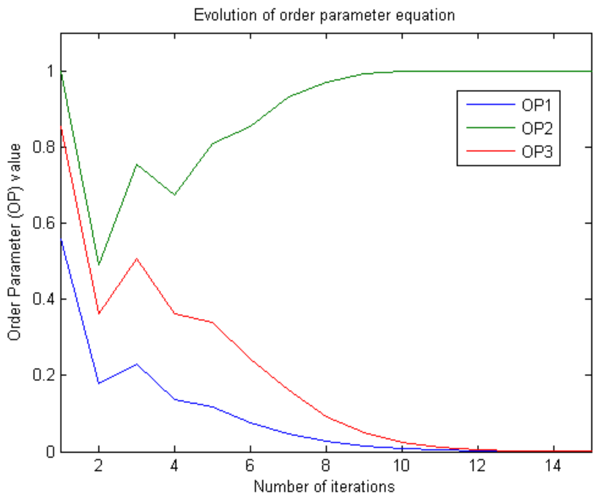


Fig. 3 Evolution of order parameter equations

From Fig. 3 is seen that the system converges after twelve steps, i.e. the winning order parameter becomes $\xi_1 = 1$ and other order parameters $\xi_2 = \xi_0 = 0$. Thus, the prototype pattern that corresponds to ξ_1 will be recognized. Note that due to the fact that the biggest initial order parameter $\xi_k(0)$ will always win the competition (in the case of balanced λ_k), the iteration (18) may be omitted and the resulted system remarkably simplified. The learning process will be then restricted to satisfying (10) and solving (13). If, however, we are dealing with normalized vectors, finding of adjoint vectors means just transposing and the whole dynamics reduces to forming the inner products $(v_k^T q_0)$, which further reduces the complexity of numerical computations.

The user interface of the resulted system is shown on Fig. 4.

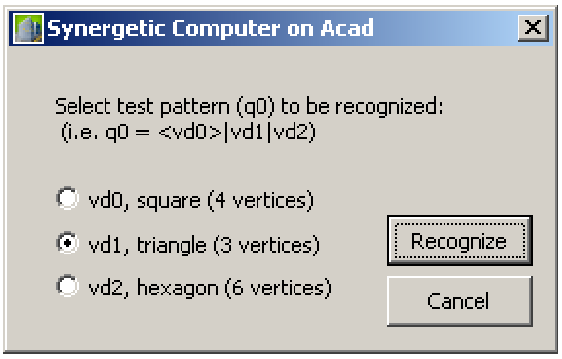

Synergetic Computer on Acad
Select test pattern (q0) to be recognized:
(i.e. q0 = <vd0>|vd1|vd2)
vd0, square (4 vertices)
vd1, triangle (3 vertices)
vd2, hexagon (6 vertices)
Recognize
Cancel

Fig. 4 GUI of SNN application

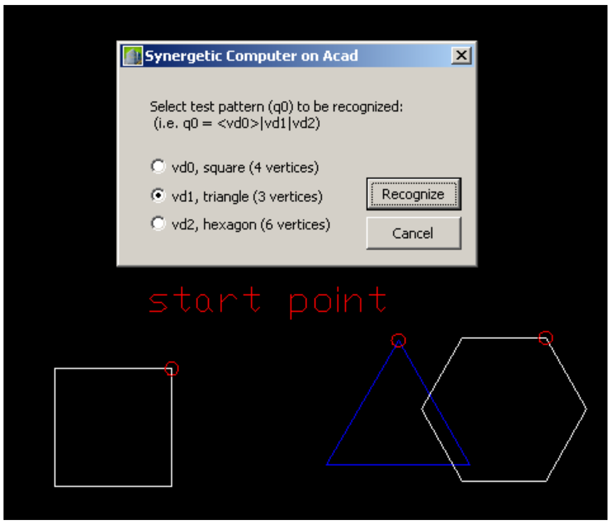
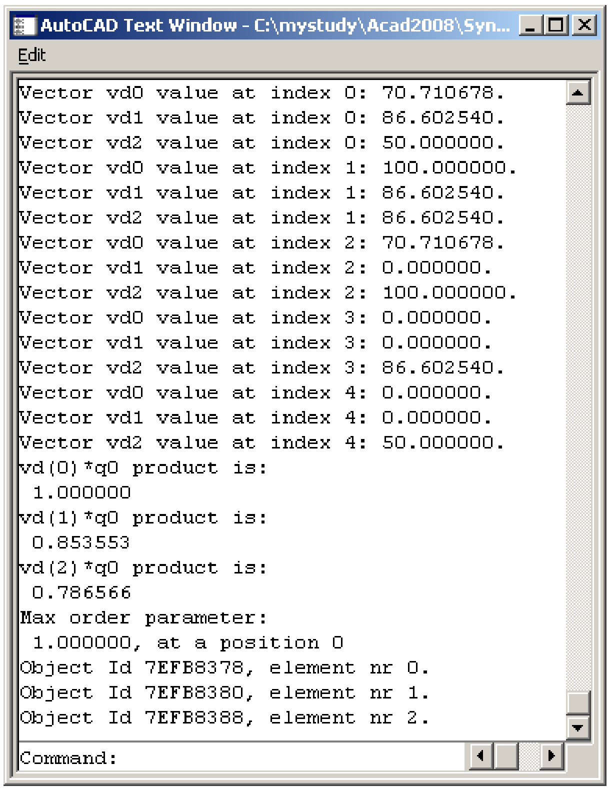

Synergetic Computer on Acad
Select test pattern (q0) to be recognized:
(i.e. q0 = <vd0>|vd1|vd2)
vd0, square (4 vertices)
vd1, triangle (3 vertices)
vd2, hexagon (6 vertices)
Recognize
Cancel
start point

Fig. 5 AutoCAD graphics screen, the identified pattern is selected

The user may choose the pattern desired by selecting respective radio button, then by pressing the action button the system performs the recognition process and selects the identified polygon by changing its color property (to blue), see Fig. 5.

The location and position of the test patterns has no effect on the recognition results. The polygons may be rotated, or, even overlapped on each other, the system still successfully identifies the figures.

On Fig. 6 the Acad text window with implemented system's output messages is shown.



```
Vector vd0 value at index 0: 70.710678.
Vector vd1 value at index 0: 86.602540.
Vector vd2 value at index 0: 50.000000.
Vector vd0 value at index 1: 100.000000.
Vector vd1 value at index 1: 86.602540.
Vector vd2 value at index 1: 86.602540.
Vector vd0 value at index 2: 70.710678.
Vector vd1 value at index 2: 0.000000.
Vector vd2 value at index 2: 100.000000.
Vector vd0 value at index 3: 0.000000.
Vector vd1 value at index 3: 0.000000.
Vector vd2 value at index 3: 86.602540.
Vector vd0 value at index 4: 0.000000.
Vector vd1 value at index 4: 0.000000.
Vector vd2 value at index 4: 50.000000.
vd(0)*q0 product is:
 1.000000
vd(1)*q0 product is:
 0.853553
vd(2)*q0 product is:
 0.786566
Max order parameter:
 1.000000, at a position 0
Object Id 7EFB8378, element nr 0.
Object Id 7EFB8380, element nr 1.
Object Id 7EFB8388, element nr 2.

Command:
```

Fig. 6 AutoCAD text window, program control messages

A. *Differences from standard model*

The model differs from classic Haken representations by the following points: (8), (9) and (11). More specifically, in our Acad model the number of patterns allowed to be bigger or equal to the number of features $M \geq N$. Additionally, numerical simulations have shown that the model works well in situation where $M \gg N$.

We have omitted the condition (9) in our model, as tests have proved it to be redundant.

The size of the prototype vectors is different in our implementation, thus the (11) is not satisfied. However, the model still performs well. Here, of course, it is the number of meaningful dimensions that is important. For a system to be solvable, the trailing zeros should be added to vectors of different size:

$$v_0 = (L_{01}, L_{02}, L_{03}, 0, 0)$$
$$v_1 = (L_{11}, L_{12}, 0, 0, 0)$$
$$v_2 = (L_{21}, L_{22}, L_{23}, L_{24}, L_{25}).$$

All these modifications, while simplifying the system and allowing for a greater flexibility, do not degrade the model's performance nor obscure general properties of SNN. It is worth noting, that the recognition rate of the model so far was 100%. This could be explained e.g. by the fact that only the noiseless patterns were used for the test vectors.

B. *Tools and technologies used*

AutoCAD Architecture 2008 as a main framework for the model, VC++ 8.0, ObjectARX 2007, Eigen3, MATLAB 7.0.

C. *Further research*

The directions for further research connected with this section discussion are outlined below.

In our model the balanced attention parameters were used (according to (7)) and $\lambda_k = C = B = 1$. It is possible to use the unbalanced attention parameter technique e.g. $\lambda_1 = 0,6$ and $\lambda_2 = 0,4$. In that case the biggest value will influence the evolution of the order parameters and it is possible for initially smaller order parameter eventually to win the competition. Attention parameters thus could be used as an additional instrument to guide the selection of order parameters in situations where selection criteria based solely on ξ_k values are not sufficient. Those situations are likely to arise when dealing with more complicated patterns and process objectives, as e.g. in ADS structures or, just as simple, as in treatment of vectors of different size, like (17). The implementation of SNN in ADS, as well as the treatment of noisy patterns, is a subject for a further research.

IV. Possible Implementation in ADS

Of course, the implementation of the synergetic computer for the recognition of simple AutoCAD entities is not the aim of its own. Instead we want to apply these principles to the useful recognition scenarios e.g. as a component of ADS system. Although there are plenty of different application possibilities that could be elaborated, let us research the one from HVAC (Heating Ventilation Air Conditioning) engineering domain. More specifically, we want to automate the process of building's outer peripheries (e.g. outer walls) data acquisition from AutoCAD environment. To this end, we will use the Heatloss original software developed in author's earlier research as a framework for software agents testing and implementation on AutoCAD platform.

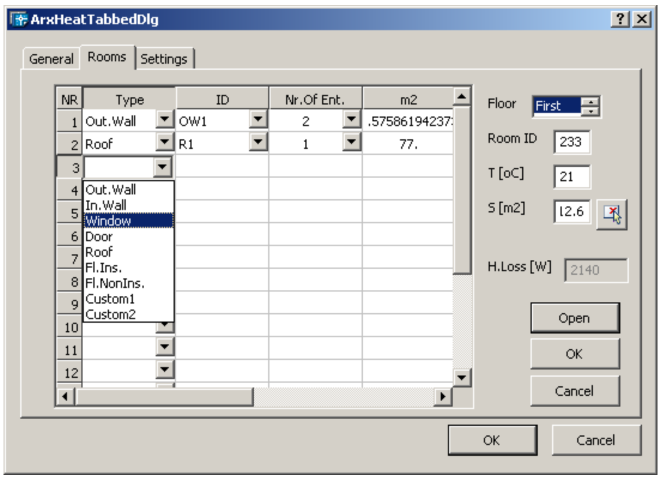
The process of the selection of outer walls of the building clearly falls into the engineer's creative activity class, if we consider this process as a dynamic synergetic system. It is quite easy for the human designer visually identify the outer walls from other geometry on the graphic screen. For the computer, however, it is not an easy task, if we treat all graphical data equally in the sense of human visual perception. This is our task to treat the underlying vector entities as patterns, such that we could apply the principles of the self-organization theory and use the synergetic tools described above. Note that the same task may be solved by "traditional" cybernetic approach methods e.g. by introducing some additional metadata to the graphical objects in order to make it possible for a straightforward computational identification. Such parametrical approach is very common nowadays in information systems design; in CAD domain it is used e.g. in BIM (Building Information Modeling) applications.

In the following subsections we briefly explain the Heatloss software's related existing functionality and describe its possible improvements by exploiting synergetic computer properties/advantages. We will also discuss the issues of optimal permutation selection realization of test patterns.

A. *Additional functionality for HeatLoss software*

The HeatLoss software is ObjectARX module that automates the calculation of the building's heat losses. Below is the brief explanation of related GUI.

The Rooms tab is a main working UI of the program. It has a grid control, which is similar to a spreadsheet by its functionality. The grid control used in this program is very powerful custom control. It has a rich set of different features; most of them is not used currently in HeatLoss, but are planned to the future releases. In the next version, for example, we plan to add a drag-and-drop capability to the grid.



NR	Type	ID	Nr.Of Ent.	m2
1	Out.Wall	OW1	2	.57586194237
2	Roof	R1	1	77.
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				

Floor: First
Room ID: 233
T [oC]: 21
S [m2]: 12.6
H.Loss [W]: 2140

Fig. 7 HeatLoss application rooms peripheries window (Rooms tab)

User selects type of the periphery, its ID and program brings his immediately to the AutoCAD drawing screen (DWG) where he picks characteristic points. Next the program computes area of the periphery and heat loss of the room, based on the data in the Settings tab and room's inner temperature and displays UI back to the user. The process repeats. Of course it is possible to modify data in the grid and in the tab after initial calculation is done. User may change, for instance, number of entities, area of the periphery, U value, inner temperature etc and program updates the heat loss value accordingly.

We want to add to this program the additional functionality, i.e. the ability for the system to select the outer periphery (outer wall) on its own, using synergetic computer approach. We shown earlier that such ability transform the conventional CAD system into ADS system.

This can be achieved by the following. When user selects the ID of the type of the periphery chosen, the system instead of the prompting the user to manually select the periphery (wall) length perform the identification process based on the prototype patterns stored. Upon the identification, the system acquires technical parameters of the room identified and calculates the heat loss of this room.

One way to compare room's characteristic vectors is to use all possible combinations (more specifically, permutations) of the representing distances (see Fig. 8) and to form corresponding test vectors. The basic principle is the same as described in section III.

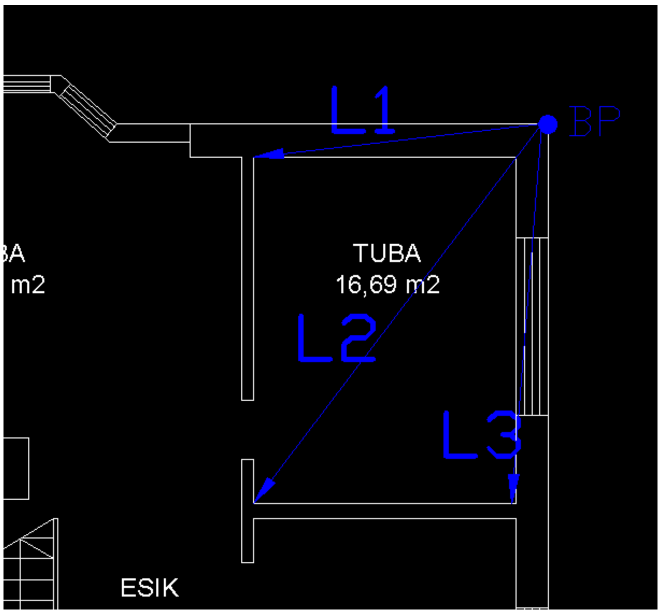


Fig. 8 Representation of the room geometry as test vector elements

Here in Fig. 8 shown three leaders L1, L2, L3 drawn from the base point BP. This combination corresponds to the prototype pattern vector being recognized. However, in order to get these points (distances) into comparison in AutoCAD, we have to select all line/polyline type entities in the vicinity of BP. Thus we will have eventually a number of different distances from BP to the corresponding entities' start/end points; then we have to permute these distances such as to get the right vector elements combination (e.g. L1, L2, L3). For example, if we have 42 distances (i.e. 42 possible elements of the test vector) and the prototype vector consisting of 3 elements, we will have as much as $C_k^r = \frac{k!}{(k-r)!} = \frac{42!}{(42-3)!} = 68880$ different permutations.

Therefore in order to preserve numerical computation efficiency it is very important what kind of permutation computation algorithm is chosen. In the next subsection we will discuss this issue in more detail.

B. Patterns' features selection optimization

As the computational speed in engineering applications is quite important [6], we do not want to degrade software performance in permutations module as well. This is particularly important for CAD and ADS systems. That is why ObjectARX (C++) technology is chosen for our systems' implementation. In comparison with other AutoCAD development technologies (VBA, .NET, AutoLisp/DCL see e.g. [7]) ObjectARX is the most powerful IDE creating the fastest and most compact ARX (DLL) modules available. The processing speed of the permutation algorithm depends on the functions and programming language constructs chosen for one particular application. Let us review some widely used C++ permutation algorithms/functions. We then compare them and select the most effective (optimal) routine for our system implementation. That is, we will perform the pattern selection process optimization for our ADS module.

Howard Hinnant [8] has performed the tests amongst most widely used permutation (combination) algorithms. Below is a brief review of the results. There were 3 different approaches, solutions A, B, and C.

Solution A.

The standard library has *std::next_permutation* and it is possible trivially build a next_k*_permutation from it* and a *next_combination* from that (Fig. 9):

```
template<class RandIt, class Compare>
bool next_k_permutation(RandIt first, RandIt
mid, RandIt last, Compare comp)
{
    std::sort(mid, last, std::tr1::bind(comp,
std::tr1::placeholders::_2
                                              ,
std::tr1::placeholders::_1));
    return std::next_permutation(first, last,
comp);
}
```

Fig. 9 Example code for solution A

The performance results of this solution are as follows:

```
N = 100, r = 5, visits = 75287520
    next_combination total = 4519.84 seconds
    next_combination per visit = 60034.3 ns
```

Fig. 10 The performance printout of solution A

Solution B.

This solution is developed by Hervé Brönnimann (called

N2639) and can be found at [9]. This proposal adds eight algorithms (*std::next_partial_permutation, next_combination, next_mapping, next_repeat_combination_counts*, their counterparts *std::prev_partial_permutation, std::prev_combination, std::prev_*-mapping, *std::prev_repeat_combination_counts*, with their overloads) to
the header <*algorithm*>, for enumerating permutations and combinations, with and without repetitions. They mirror and extend *std::next_permutation* and *std::prev_permutation*. For sizes known at compile-time, these algorithms can generally be simulated by a number of nested loops. The performance results of this solution are shown on Fig. 11.

```
N = 100, r = 5, visits = 75287520
    next_combination total = 6.42602 seconds
    next_combination per visit = 85.3531 ns
```

Fig. 11 The performance printout of solution B

Solution C.
Finally there is a solution C found here [10]. This solution has a different signature/style and is called *for_each_combination (for_each_permutation)*, and is used much like *std::for_each*. The driver code between the timer calls is as follows:

```
Clock::time_point t0 = Clock::now();
f = for_each_combination(v.begin(), r,
v.end(), f);
Clock::time_point t1 = Clock::now();
```

Fig. 12 The driver code between the timer calls of solution C

The performance results of this solution are shown on Fig. 13.

```
N = 100, r = 5, visits = 75287520
    for_each_combination = 0.498979 seconds
    for_each_combination per visit = 6.62765
ns
```

Fig. 13 The performance printout of solution C

Solution C is 12.9 times faster than solution B, and over 9000 times faster than solution A.

We consider this a relatively small problem: only 75 million visits. As the number of visits increases into the billions, the discrepancy in the performance between these algorithms continues to grow. Solution A is already unwieldy. Solution B eventually becomes unwieldy. Solution C is the highest performing algorithm to visit all combinations/permutations author aware of.

Thus we have to choose the approach of the solution C for our systems coding.

C. *Further research*

The actual implementation of the algorithms described in this section in ADS system is left for a further research. More specifically, this research is currently running and we obtained some preliminary results in the realization of solution C on AutoCAD in outer wall recognition routine. The description of the results of the experiments is the subject of further publications.

V. Some General Remarks on the Subject

In this section the general and somewhat philosophical remarks on the theory of ADS and modeling of the creative part of the design process are presented.

The ongoing research in AI domain and in the field of general technology shows that traditional methods of solving engineering problems based on formal logic and systematical approach shifts toward the new unrevealed, presently undocumented features of human mind and intelligence (more closely to the characteristics of self-organization?). There are neural networks, which try to copy the functionality of biological brain cells – neurons, fuzzy logic and modeling (for a contemporary research on fuzzy dynamic systems see e.g. [11]), expert systems, evolutionary programming/computing, knowledge-based systems, swarm and genetic algorithms and so on.

The routine parts of the engineering design process could be successfully modeled with the help of cybernetics. It is really the art of combinatorial manipulation and constructing to fulfill the goal, using the already known or novice technology, IT in this case. As it is based on cybernetics, it falls down to organizational theories, contrary to self-organization paradigm, and therefore is not connected with the subject of interest of this paper.

Let us take a look at the notions of organization and self-organization from the concept point of view. The concept of organization denotes the process that leads to the rise of goal-oriented structures due to conscious human goal-directed action or some external ordering influence, and the concept of self-organization would denote the process that leads to the rise of goal-oriented structures beyond conscious human goal directed action or some external ordering influence. Although the term "self-organization" is widely used (and more appropriate) in the field of synergetics, it has been utilized in cybernetics as well. In cybernetics, however, it has different meaning (from the philosophical point of view). In cybernetics and systems engineering self-organization is understood as an effect of an external ordering factor (e.g. self-organizing map in [12]). In synergetics self-organization is understood as the rise of harmonious behavior distinguished from man's intervention and from external (with regard to the system) ordering factors. External factors (e.g. strong non-equilibrium) are indispensable for self-organization, but only as conditions, not as ordering forces.

Hopefully, it is possible to imitate the creativity (at least to some degree) by means of synergetic modeling. Could we model the creative part of the (engineering) design process as well? To answer this question we must analyze the synergetic approach and compare it with traditional information technology modeling instruments e.g. with cybernetics.

In cybernetics as well as in synergetics the objective processes are modeled in order to control them. The cybernetic models make it possible for man to strive for the desirable results using the program created by him. The synergetic models take into account that the programs form in the course of self-organization [13].

All exact sciences (and also the traditional scientific cognition) are model-based. These are exact only within that model. Therefore it is not possible to explore/predict/study adequately the real world by means of “exact” sciences by definition. We can use exact sciences to explore models. CAD systems, ADS frameworks are examples of design systems’ models. Both cybernetics and synergetics are exact sciences as well. So we can use these disciplines only for the development and research of models of the underlying real world’s phenomena and not for the investigation of the real world itself. It must be underlined that in exact sciences the approach to the interaction between organization (management) and self-organization does not go (and due to the specificity of exact sciences must not go) farther from certain boundaries.

The limits mean that exact sciences in their models of influence upon self-organization give only such recommendations according to which the future state of an object of management is given from the outside. Exact sciences do not make any contribution to the opening of the creative potential of the elements of the system [13]. So we cannot use standalone synergetic methods (a kind of exact science) to explore the creative potential of the system (and self-organization). As the synergetics is exact science and is based on mathematics, it has known limitations in its capability to explore the real world. But still we can use it to *create the better models* of the real life systems, not to understand these systems completely. On the other hand, building more adequate models of the environment leads to a better understanding of the environment itself. And, therefore, may lead us to a new level of understanding, to help us form a new paradigm and from within it - to model even more precisely, closely to the real world.

Synergetics better than cybernetics models the processes of the real world which is ultimately the self-organizing system. So we can use principles of synergetics in conjunction with traditional computing technology to model some aspects of the real systems. It is worth showing how creativity is understood in synergetics. The meaning of the word creative is the unpredictability and unavoidability of the unknown. The creative chaos is the field of unknown and unpredictable chances. The meaning of the word is closely related to such concepts as non-equilibrium condition and conditions close to equilibrium.

Synergetics accentuates also one necessary condition of self-organization: the order arises from chaos only under the condition of *strong non-equilibrium*. It is necessary to distinguish strictly chaos under the conditions close to equilibrium (in which, generally speaking, self-organized structures can only decompose) from chaos under the strongly non-equilibrium conditions (in which composing of structures through self-organization can take place) [13]. The former type of chaos is non-creative, the latter is creative.

In engineering design process theory the meaning of the words “creative” (and “creativity”) is slightly different. Here the word “creative” denotes a non-routine part of the design process. Contrary to the routine procedures where inputs and outputs of the system are known or predictable, the creative part of the process deals with output data that is mainly unknown, although the field of possibilities (possible outputs, similar to synergetics theory) is generally defined. This is true in ordinary design scenarios where the ultimate goal of the design procedure is known. When the output data of the system is completely undefined and unknown, then we are dealing with the system that generates some new design information (i.e. invention mechanism). Note, that the input data in majority of cases is defined (both in ordinary design scenarios and in invention apparatus). The modeling of the technical invention processes is even more complicated (if not impossible) than imitating the creative part of the conventional design process (i.e. the process where the field of the possibilities of the output information is defined). There is a hope that using the methods of synergetics and the philosophy of self-organizing systems we can try to address the problems of modeling creative design in a more precise and better manner. The new science which accepts creativity based on chance and irreversibility in nature, and considers the fundamental indeterminacy of the whole history of nature and of human society should evolve to acknowledge the potential of this approach.

Basically, we can consider a model as an idealized version of the real system. The model is always a simpler and more primitive than the real system. The traditional tool for creating engineering design models nowadays is a Computer Aided Design (CAD) system. For a creation of a new CAD system we use CAD programming. Thus, CAD programming is essentially construction of the model (computer program) for the model (CAD application) of a model (engineering design, project) of the system (e.g. engineering installation). Such models’ cascading occurs e.g. in a case when we are programming under some existing CAD platform, let’s say under AutoCAD. On this level of abstraction the model itself is very precise (it is nested into surrounding model etc.) and perfectly describable by mathematics.

The aim is to try to add to this model the properties/specifications of the self-organizing systems’ behaviors. The author does not really think that the model will be capable of substituting the engineer completely in the process of producing creative design. But there is a hope that the model built in the spirit of synergetics could facilitate the emergence of the elements of the creativity in engineering design in which the human participates as well. It is likely that these models in cooperation with the operator (engineer) can function more effectively in creating new designs.

VI. Conclusion

This paper describes the results of the synergetic computer implementation on AutoCAD platform.

A number of useful modifications to the standard model were committed, tested and successfully implemented in the form of AutoCAD application. This is the first documented usage of SNN on AutoCAD platform.

The presented research constitutes another step to the development and research of the fully functional Autonomous Design System (ADS).

Synergetic computer has one major advantage, compared to the traditional neural computers, namely, there are no so-called pseudo-states, into which the system could be trapped in. It may be proved that besides the prototype vectors there are no other attractors. In addition, the functionality of SNN, especially pattern recognition mechanism and treatment of ambiguous and noisy patterns closely resembles the functionality of biological neural systems, including human brain [14]. This point supports the whole philosophical study of the self-organization phenomenon and is the main reason for selecting synergetic computer approach for ADS implementation.

A closer look on the problem of adding synergetic ADS functionality to the existing HVAC CAD application was taken. The technical part of the implementation of C++ permutation functionality and its performance considerations as part of the currently developing ADS system was discussed.

We also gave a short philosophical outlook on the problem of modeling the engineering creativity in ADS.

References

[1] H. Haken, H. Knyazeva, “Arbitrariness in nature: synergetics and evolutionary laws of prohibition,” *Journal for General Philosophy of Science*, No.31, 2000, pp. 57–73.

[2] H. Haken, *Synergetics. Introduction and advanced topics*, Springer, 2004.

[3] Hermann Haken, *Brain Dynamics. An Introduction to Models and Simulations*, Second Edition, Springer, 2008

[4] H. Haken, *Synergetic Computers and Cognition. A Top-Down Approach to Neural Nets*, Springer, 2004.

[5] Lucia-Antoneta Chicos, Gheorghe Oancea, Camil Lancea, Daniel Bancila, “Software system of integrated and simultaneous engineering,” in Proc. 10th WSEAS International Conference on Applied Computer Science (ACS '10), Iwate Prefectural University, Japan, 2010, pp. 238–241.

[6] Cosmin Marian Poteras, Mihai Mocanu and Constantin Petrişors, “A Distributed Design for Computational Steering with High Availability of Data”, *International Journal Of Systems Applications, Engenering & Development, (NAUN),* Issue 1, Volume 6, 2012, pp. 52–61.

[7] Sever Alexandru HABA, Gheorghe OANCEA, Anisor NEDELCU, “Data Exchange Software Module between AutoCAD and CATIA Environments,” in Proc. 11th WSEAS International Conference on Systems Theory And Scientific Computation (ISTASC '11), Florence, Italy, August 23-25, 2011, ISBN: 978-1-61804-027-5, pp. 134–137.

[8] Howard Hinnant. (2012, May 10). Stackoverflow C++ discussion. [Online].Available: http://stackoverflow.com/questions/2211915/combination-and-permutation-in-c

[9] Hervé Brönnimann. (2012, May 10). *Algorithms for permutations and combinations, with and without repetitions*. [Online].Available: http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2008/n2639.pdf

[10] Howard Hinnant. (2012, May 12). *Combinations and Permutations*. [Online].Available: http://home.roadrunner.com/~hinnant/combinations.html

[11] Nikos Mastorakis, Olga Avramenko, “Fuzzy models of the dynamic systems for evolution of populations,” in Proc. 3rd WSEAS International Conference on Mathematical Biology and Ecology, Gold Coast, Queensland, Australia, 2007, pp. 27–32

[12] Supakit Siripanadorn, Wipawee Hattagam, Neung Teaumroong, “Anomaly detection using self-organizing map and wavelets in wireless sensor networks,” in Proc. 10th WSEAS International Conference on Applied Computer Science (ACS '10), Iwate Prefectural University, Japan, 2010, pp. 291–297.

[13] Leo Näpinen, “Philosophical Foundations of Synergetic Modelling,” *Proceedings of the Estonian Academy of Sciences. Humanities and Social Sciences*, Vol.4, No.42, 1993, pp. 378–390.

[14] D. Loginov, “Synergetic Modelling – Application Possibilities in Engineering Design,” in Proc. 10th WSEAS International Conference on System Science and Simulation in Engineering (ICOSSSE '10), Iwate Prefectural University, Japan, 2010, pp. 111–116.