Umair Ali Khan, Alireza Fasih, Kyandoghere Kyamakya, Jean Chamberlain Chedjou

Transportation Informatics Group, Alpen Adria University, Klagenfurt, Austria.

Genetic Algorithm Based Template Optimization for a Vision System: Obstacle Detection

Abstract *- A simulator is developed for training and optimizing the templates for cellular neural networks for obstacle detection. The simulator uses the Genetic Algorithm (GA) for training the cellular neural network. The traditional method of genetic algorithm involves creating an initial population of random solutions (chromosomes) in binary format, the so called chromosomes encoding. But this approach of genetic algorithm defines the chromosomes in the form of real numbers, thus eliminating the need of encoding and decoding of the chromosomes. The results differ, by no means, with those of the traditional methods. The method was used for obstacle detection for autonomous vehicles giving two stereo images of a sequence as inputs. The output results for various different image processing tasks are also presented.*

Keywords*: Optimization, Genetic Algorithm, Chromosomes encoding, Decoding, Obstacle detection.*

1. Introduction

The problem of obstacle detection for the vehicles driving without driver assistance is one of the major challenges in the field of robotics and machine vision. A robust mechanism depicting the most complicated and accurate vision system of human being needs to be sorted out properly. The problem of identifying changing environment of the roads, detecting the potential obstacles and avoiding them are tremendous tasks in the field of machine vision. The basic aim of obstacle detection is to extract feature points in images and removing all the other image contents. The most important factor which is always needed to be fine tuned is the speed. This process needs to be accurate and should be carried out with a very fast speed.

The common approaches in this context use analytical and statistical methods like motion estimation or generation of maps. One of these methods involves feature extraction, subsequent displacement vector estimation and a robust estimation of the motion parameters. Since this procedure is composed of several processing steps, the error propagation of the successive steps often leads to inaccurate results [1].

Using Cellular Neural Networks (CNN), direct obstacle detection can be performed which eliminates the above mentioned problem. The parallel computation paradigm of CNN provides a fast processing mechanism. Presenting two stereo images of a sequence to CNN, to highlight the 3D objects as potential obstacles in the image, provides a fast and robust mechanism for obstacle detection.

For obstacle detection using CNN, there is a need of training CNN for highlighting the 3D objects in the image. The training process includes parameter optimization for CNN. This approach has also been used in [1] which uses the so-called *iterative annealing* [2] method for parameters optimization. For carrying out this task, a CNN with 5-by-5 neighborhood and polynomial cell coupling of degree 3 was used. One of the drawbacks in iterative annealing is the possibility of trapping into local minima and ending with an incorrect solution.

The approach presented above is discussed in this paper by carrying out the same task using a 3-by-3 CNN. For template/ parameter optimization, we used genetic algorithm. Genetic algorithm is a learning algorithm based on the mechanism of natural selection and genetics, which has proved to be effective in a number of applications [3]. Suitable selections of its operators, avoids the algorithm to fall into local minima.

The common approach of genetic algorithm involves creating an initial population of binary numbers that represent the possible solutions. The search evolves with these initial population members (called chromosomes) and manipulates them in order to achieve accurate or optimal solution. The population of binary numbers needs repeated encoding and decoding process. Also the sizes of chromosomes are very large and vary proportionally with the problem variables.

We have used a real coded approach of genetic algorithm that exploits an initial population of real numbers rather than binary numbers. This eliminates the need of repeated encoding and decoding of chromosomes and adds to the efficiency and speed of the algorithm. The approach was not only used for finding obstacles in the images, but also for other image processing tasks e.g. thresholding, noise removal, filling etc and found to produce good results.

This paper discusses a brief introduction of Cellular Neural Networks and genetic algorithm as a good candidate for CNN parameter optimization. The obstacle detection using CNN based on the real coded approach of genetic algorithm is considered as well. Various output results for different image processing tasks are also presented.

2. Cellular Neural Network

Cellular Neural Network (CNN) was introduced by Leon O. Chua and Ling Yang in 1988. It is a massive parallel processing paradigm which combines some of the features of Cellular Automata (Discrete states, concept of neighborhood) [4] and Artificial Neural Networks (simple processing elements, continuous states and parallel computation) [5].

CNN is an n-dimensional array of mainly identical systems, called cells [6]. What distinguishes CNN from traditional Artificial Neural Networks is the locality of connections. Unlike artificial neural networks, every cell in cellular neural network communicates directly to its nearest neighbors only. The locality of coupling adds the amazingly enhanced processing speed in cellular neural networks. Each cell is made up of a linear capacitor, a non linear Voltage-controlled current source and a few resistive linear circuit elements, as shown in Fig.1 [7].

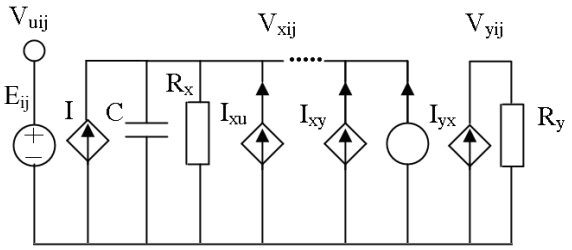


Fig.1 Basic architecture of a CNN cell

Where C is a linear capacitor; R_x and R_y are linear resistors;

I is an independent voltage source; I_{xu} and I_{xy} are linear voltage controlled current sources with the characteristics $I_{xy}(i,j;k,l)=A(i,j;k,l)u_{ykl}$ and $I_{xu}(i,j;k,l)=B(i,j;k,l)u_{ukl}$ for all $C(i,j)\in N(i,j)$; I_{yx} is a piecewise-linear voltage-controlled current source.

Applying Kirchhoff's current law (KCL) and Kirchhoff's voltage law (KVL), the following state equation of CNN can be derived.

$$\dot{x}_{ij} = -x_{ij} + \sum_{C(k,l)\in N(i,j)} A(i,j;k,l)y_{kl} + \sum_{C(k,l)\in N(i,j)} B(i,j;k,l)u_{kl} + I \quad (1)$$

Where x_{ij} is the state of the cell $C(i,j)$. $A(i,j;k,l)$ and $B(i,j;k,l)$ are *feedback* and *control* templates respectively for all cells $C(k,l)$ in the neighborhood $N(i,j)$ of cell $C(i,j)$. The output equation is given as:

$$y_{ij} = \frac{1}{2}(|x_{ij}+1| - |x_{ij}-1|) \quad (2)$$

Feedback template (A), *control template (B)* and *Bias (I)* work as CNN instructions and determine its output for a given input.

3. Genetic Algorithm

Genetic algorithm was invented by John Holland in 1960s [8]. It is an effective method for determining the parameters for CNN. This method is inspired by the mechanism of natural selection and genetics. It has been effectively used for solving difficult search, optimization and machine-learning problems. It works by creating genotypes (set of chromosomes) that represent the possible randomly chosen solutions. The search evolves to improve the quality of chromosomes until the best chromosome is found that represents the optimal solution. The process of evolution occurs in the form of generations and in each generation better chromosomes are sorted out.

The parameters of genetic algorithm that play an important role in the process of evolution and finding the best solution are *initial population, selection*, *Reproduction*, *crossover*, *mutation,* and *fitness function*.[9]

3.1. Initial Population

An initial random population of the chromosomes is generated. Each chromosome represents a possible problem solution. All chromosomes are composed of a fixed number of genes.

The approach we used in genetic algorithm represents the chromosomes in the form of real numbers instead of binary digits. This creates chromosomes of relatively smaller sizes and the repeated operations of encoding and decoding are eliminated. For a 3 x 3 CNN, total numbers of genes contained in a chromosome are 19. 9 genes are for the control template; 9 for the feedback template and 1 for bias. Similarly, if used for 5 x 5 CNN, the total number of genes in a chromosome would be 51.

3.2. Selection

As the process of evolution creates new generations, and every new generation contains better chromosomes than the previous one, the process of selecting better chromosomes for next generation is very important. The selection process does not consider the chromosomes merely on the basis of fitness. This process is rather random, through which the parents for the new generations are selected randomly based on their fitness. The method of selection used in our genetic algorithm approach is based on selecting parents from half of the population. After evaluating the fitness of all the chromosomes, the population is sorted in descending order. The best chromosomes occupy higher locations in the list. Every time the two chromosomes are selected randomly for crossover and mutation from half of the population. In order to increase the probability of selecting good parents, the parents can be selected from a specified fraction of the half of the population. For example, the first parent can be selected from half of the population and the second can be selected from 30% of the half of the population from the pool of much better parents.

3.3. Reproduction

The reproduction phase involves generating new population for the next generation from the selected parents by applying two genetic operators *crossover* and *mutation* on the selected parents. These two processes result in the next generation population of chromosomes that is different from the previous generation.

In our approach, after parents selection, crossover, mutation and fitness evaluation, the two childs can go to the next generation only when the fitness of each one of them is greater than the worst child. The first child is compared with the worst parent and if its fitness is greater than the worst parent, it is replaced by the worst parent and the population is sorted in descending order. The same is done with the second child.

3.4. Crossover

After selecting parents from the population by any suitable mechanism, the genetic algorithm operator *crossover* is applied on the selected parents. Crossover breeds the selected parents to produce new childs for the next generation. For the reproduction phase and to produce the next generation, breeding the parents to produce new childs is necessary, otherwise the evolution process cannot proceed to better solution. Crossover can (but not every time) produce childs that have better fitness than the parents. We use 2-point crossover in which two crossing sites are selected in the parent chromosomes randomly and the genes between the crossing sites are interchanged between the two parents.

3.5. Mutation

Mutation is a genetic operator that maintains genetic diversity from one generation of a population to the next. The purpose of mutation is to prevent trapping into local minima.

In our approach of genetic algorithm that uses real number chromosomes, any arbitrary number (gene) in a randomly selected parent is changed by a randomly selected number that falls in the interval to which all the chromosomes genes belong.

3.6. Fitness Function

Fitness function plays an important role in determining the exact solution. It determines the fitness for every chromosome in every generation by comparing it with the original solution. The exact solution can be reached when an exact (or near to exact) match is found between a chromosome and the original solution. The search may also finish when a specified number of populations/iterations has been completed. The fitness function used to compare every output image with the target image in our genetic algorithm approach is based on Euclidean distance between the two images. It first calculates a cost which is a measure to which extent two images differ from each other.

This cost value is then mapped to a fitness value which represents the fitness of the output image. Throughout the evolution process, the genetic process aims to minimize the cost function and increase the fitness value.

$$Cost(i,j) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}\left(I(i,j) - T(i,j)\right)^2}{4MN} \quad (3)$$

$$Fitness = 1 - Cost \quad (4)$$

In Eq.3, I represent $M\ X\ N$ output image from the CNN and T represents $M\ X\ N$ target or reference image. Euclidean distance is useful to find the pixel wise difference between the input and the target image. The denominator *4MN* is used for normalizing the cost between 0 and 1. The input image is normalized in the range [-1, +1] before being fed into the CNN.

4. Obstacle Detection

Collision prevention for autonomously navigating moving vehicles driving without driver assistance needs a robust prediction of potential objects. The basic aim of obstacle detection is to extract feature points in images.
Two images of a synthetical image sequence are presented in Fig.2, showing a ride over a textured plane on which three dimensional objects are located, which has been recorded by a moving camera. As in real traffic scenes, the motion direction and the viewing direction are identical. The goal is to find a CNN that is able to extract the three dimensional objects by presenting two images of such a sequence [1]. The task can be performed by removing all the details inside the image except the 3D objects that represent the obstacles.

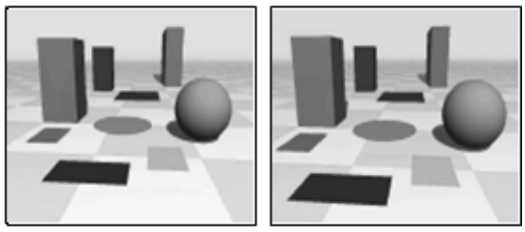
The approach used in [1] performs the edge extraction of the two images and then thresholding as shown in Fig.3. After that the two thresholded images are presented as input to CNN for training along with another target image. For the sake of comparison, we have used the same images.

We found that using suitable CNN parameters, a direct thresholding of the image can also remove the background and highlight only the foreground objects. Direct thresholding for textured plane removal is shown in Fig.4. The next step is to remove the objects present on the plane and to extract only the 3D objects in the image that represent the obstacles. This is done by presenting two thresholded images of this sequence to CNN along with a reference or target image.

Fig.5 shows the initial condition, the input and the target images applied to the CNN. A target image is constructed by removing all the plane objects and by leaving only those above the plane. The targeting is achieved just after 170 iterations producing following CNN parameters.

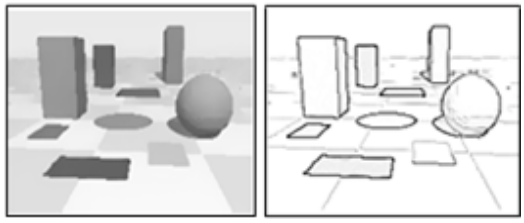
$$A = \begin{pmatrix} 0.5 & 1 & 3.2 \\ -3 & 4.8 & 0.3 \\ 1.4 & 5 & 1.7 \end{pmatrix}, B = \begin{pmatrix} 4.5 & 3 & -0.1 \\ 4.4 & 4.5 & 1 \\ -1 & 3.2 & 4.6 \end{pmatrix}, I = -3.8 \quad (5)$$

Fig.6 shows the output of the simulator. It can be seen that all the plane textures are removed and only the objects above the plane remain. The final output image does not contain the lower edges of the objects since there is no way to discriminate the lower edges of the objects and the edge pixel of texture on the plane.

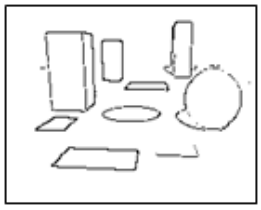


(a) (b)

Fig.2 Two images of a synthetically generated image sequence

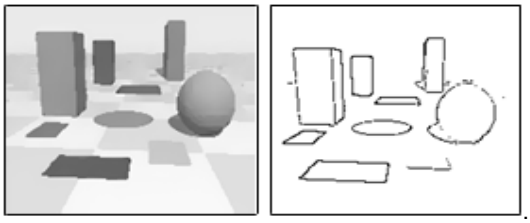


(a) (b)



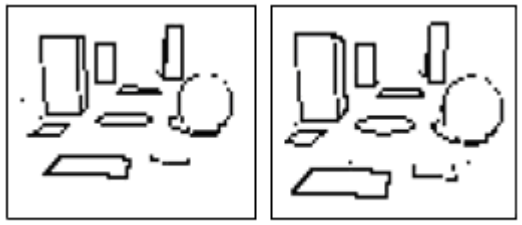
(c)

Fig.3 (a), (b), (c); input image, edge extracted image and thresholded image respectively

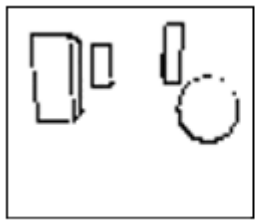


(a) (b)

Fig.4 (a) Input image and (b) thresholded image having no textured plane



(a) (b)



(c)

Fig.5 (a) initial condition image (b) input image (c) target image

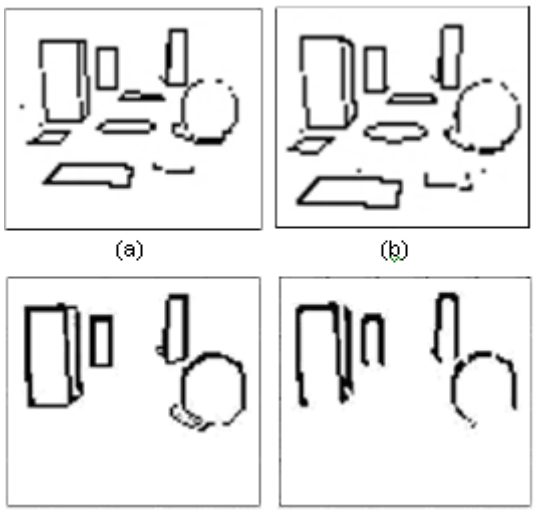


Fig.6 (a), (b) Two input images of a sequence (c) Target Image (d) CNN generated output image.

Other images of the sequence were also tested in the same way. Fig -7 shows the results of two more images of such sequence.

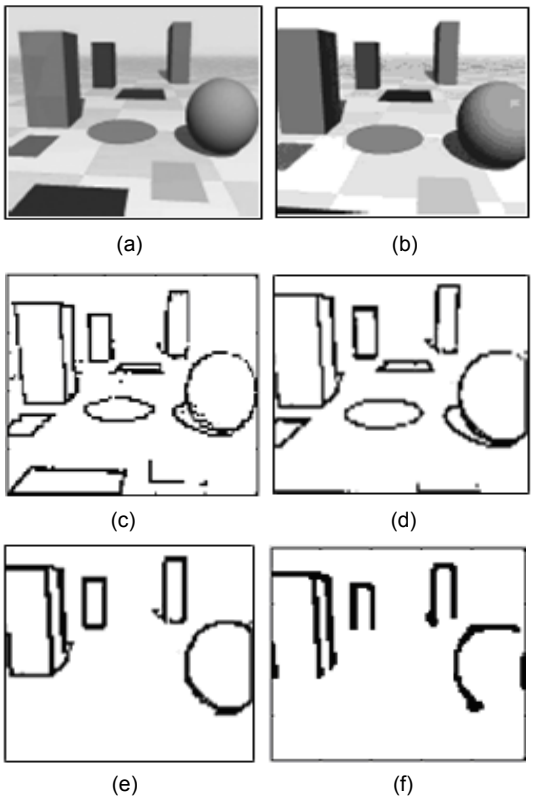


Fig.7 Obstacle detection from two other images of the sequence; (a), (b) Two input images of the sequence; (c) , (d) Thresholded input images to CNN; (e) target Image; (f) CNN generated output.

5. Test Results

We tested the real coded approach of genetic algorithm for a number of images. Some sample results are shown in Fig.8-11. In these figures, Fig (a) represents the input image, (b) the target image and (c) the CNN generated output. The CNN generated parameters are also given with corresponding results.

Fig.8 shows separation of a rectangular part from a square as specified in the target image.

(6) $$A=\begin{pmatrix}4.6 & 3.7 & 3.2\\ 3.3 & 4.5 & -0.8\\ 2.1 & 0.5 & -1.5\end{pmatrix}, B=\begin{pmatrix}3.3 & 2.5 & 3.6\\ -0.6 & 1.3 & 2.4\\ 4.2 & 0.8 & -2.7\end{pmatrix}, I=4.6$$

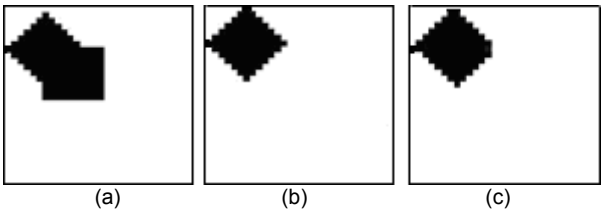


Fig.8 Removing the rectangle part from the figure

As it appears in Fig.9 (a), a noisy image is considered as input and a target is provided for noise removal. Fig.9(c) shows the output without noise.

(7) $$A=\begin{pmatrix}-1.92 & 3.28 & -1.68\\ 2.48 & 4 & 2.16\\ 0 & 3.44 & 0\end{pmatrix}, B=\begin{pmatrix}1.6 & 4 & 2.96\\ 1.92 & 1.68 & -0.08\\ 0.72 & 3.44 & -1.92\end{pmatrix}, I=3.36$$

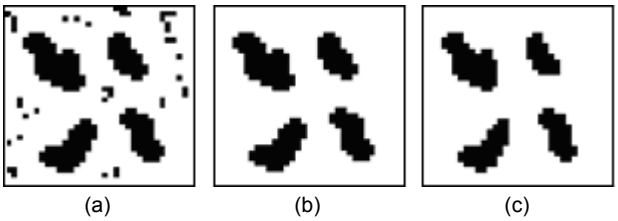


Fig.9 Removing noise from the image

An image having a tube is fed to the CNN, as shown in Fig.10(a). The aim is to fill the tube with a given dotted pixel. Fig.10(c) shows the output as specified by target.

(8) $$A=\begin{pmatrix}0.9 & 4.6 & -1.7\\ 3.7 & 4.1 & 0\\ 2.6 & 1.2 & -2\end{pmatrix}, B=\begin{pmatrix}-2.1 & 1 & -0.7\\ 1.4 & 1 & 4.7\\ -2.7 & 3.4 & -3.7\end{pmatrix}, I=-4.5$$

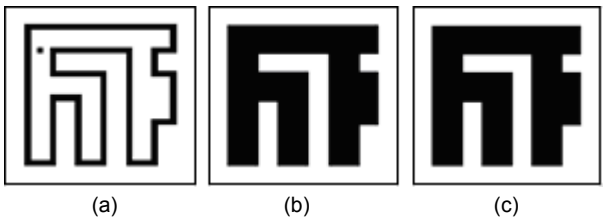
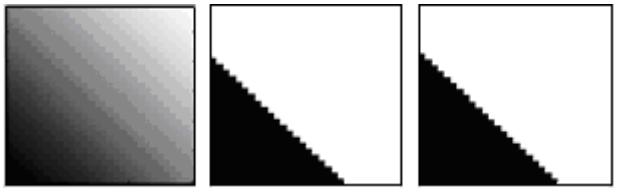


Fig.10 Filling a long tube with a dotted pixel

In Fig.11, a changing gradient image was provided as input. The aim is thresholding upto a specified limit as shown in the target image. Fig.11(c) shows the output image for the following set of parameters.

(9) $$A=\begin{pmatrix}2.7 & 2.6 & -2.8\\ 4 & 4 & 1.2\\ -3.3 & 3.9 & 1.2\end{pmatrix}, B=\begin{pmatrix}4.2 & -2.4 & -1.4\\ -3.5 & 1.8 & 4.1\\ -2 & 4.3 & 1.2\end{pmatrix}, I=4.6$$



(a) (b) (c)

Fig.11 Thresholding to a specified limit

6. Conclusion

Obstacle detection was performed by a sequence of stereo images using Cellular Neural Networks. The CNN parameters were determined by genetic algorithm based on real number chromosomes. Using real number chromosomes, repeated encoding and decoding of the chromosomes is not required. Unlike binary chromosomes, relatively smaller chromosomes are produced. This approach was successfully implemented for obstacle detection and also it was found to produce good results for many other image processing tasks. In future, we aim to implement this approach on hardware level for enhancing efficiency and speed.

REFERENCES

[1] D. Feiden and R. Tetzlaff, *"Cellular Neural Networks for Motion Estimation and Obstacle Detection," Advances in Radio Science (2003)* 1:143-147.

[2] D. Feiden and R. Tetzlaff, *"Feature Extraction in Motion Estimation with Cellular Neural Networks using Iterative Annealing," ECCTD'01-European Conference on Circuit Theory and Design,* August 28-31, 2001, Espoo, Finland.

[3] O.N. Ucan, et al., *"Extraction Of Facial Features Using Genetic Cellular Neural Networks," Journal of Electrical & Electronics*, vol.2, No.2, year 2002.

[4] S. Wolfram, *"Theory and applications of cellular automata," Advanced Series on Complex Systems, Singapore: World Scientific Publication,* 1986.

[5] Leon O.Chua and Tamas Roska, *"The CNN Paradigm", IEEE Transactions on Circuits and Systems – I: Fundamental Theory and Applications*, vol. 40, No. 3, March 1993.

[6] M.H. Hassoun, "*Fundamentals of artificial neural networks*," MIT press, 1995.

[7] Leon O. Chua and Ling Yang, *"Cellular Neural Networks: Theory," IEEE Transactions on Circuits and Systems*, vol. 35, No. 10, October, 1988.

[8] M. Mitchell, "*An introduction to genetic algorithms"*, Bradford Books, 1996.

[9] Tibor Kozek, Tamas Roska, Leon O.Chua, *"Genetic Algorithm for CNN Template Learning," IEEE Transcations on Circuits and Systems–I: Fundamental Theory and Applications.* Vol. 40, NO. 6, June 1993.

Authors

Umair Ali Khan, Research Assistant, Transportation Informatics Group, Alpen Adria University, Klagenfurt, Austria, Email: ukhan@edu.uni-klu.ac.at; Alireza Fasih, Research Assistant Transportation Informatics Group, Alpen Adria University, Klagenfurt, Austria, Email: alireza.fasih@uni-klu.ac.at ; Prof. Dr.-Ing. Kyandoghere Kyamakya, Head Transportation Informatics Group, Alpen Adria University, Klagenfurt, Austria, Email: kyandoghere.kyamakya@uni-klu.ac.at ; Dr.-Ing. Jean Chamberlain Chedjou, Transportation Informatic Group, Alpen Adria University, Klagenfurt, Austria, Email: jean.chedjou@uni-klu.ac.at