Recent Advances in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics Volume II: Applications

Editors

Krassimir T. Atanassov
Władysław Homenda
Olgierd Hryniewicz
Janusz Kacprzyk
Maciej Krawczak
Zbigniew Nahorski
Eulalia Szmidt
Sławomir Zadrożny

SRI PAS  **IBS PAN**

Recent Advances in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics
Volume II: Applications

Systems Research Institute
Polish Academy of Sciences

Recent Advances in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics
Volume II: Applications

Editors

Krassimir T. Atanassov
Władysław Homenda
Olgierd Hryniewicz
Janusz Kacprzyk
Maciej Krawczak
Zbigniew Nahorski
Eulalia Szmidt
Sławomir Zadrożny

IBS PAN  **SRI PAS**

Systems Research Institute
Polish Academy of Sciences
Newelska 6, 01-447 Warsaw, Poland
www.ibspan.waw.pl

ISBN 9788389475367

Simulation of a group of flying objects over artificial terrain

Bohdan S. Butkiewicz
address (Warsaw University of Technology),
Nowowiejska 15/19, 00-665 Warsaw, Poland
b.butkiewicz@ii.pw.edu.pl

Abstract

The aim of this paper is to present a simulation of autonomous mobile agents forming a group of objects led by the leader in 3D scenario. The approach is concentrate on good realization of the group behaviors. A fuzzy controller based on fuzzy logic and fuzzy sets theory determines movement of the leader. A mathematical model for an autonomous agent is introduced and its motion in the three-dimensional environment. The simulation of the group and leader's behaviors is shown, using an object oriented Java program, presenting both single and coexisting steering behaviors. A terrain is created based on the model of height map. The effectiveness of implemented solutions is discussed.

Keywords: steering behaviors, flying vehicles, 3D scenario, terrain generation, fuzzy control.

1 Introduction

The idea of Steering Behaviors [11] [12] [13] was introduced by Craig Reynolds in 1987. His computer-based model was a way to simulate the coordinated movements seen in flocks of birds or fish. The original flocking model was based on three distinct behaviors. The *Separation* behavior was used to prevent situations where members of the swarm were crowding each other. *Alignment* was used to

Recent Advances in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics. Volume II: Applications *(K.T. Atanassow, W. Homenda, O. Hryniewicz, J. Kacprzyk, M. Krawczak, Z. Nahorski, E. Szmidt, S. Zadrożny, Eds.), IBS PAN - SRI PAS, Warsaw, 2010.*

steer all elements of the flock in a common direction. The third behavior, *Cohesion*, steered a single member to the average position of nearby flock mates. During following years many researchers developed this idea. New behaviors were proposed [12]: *Wander*, *Seek*, *Flee*, *Pursuit*, *Evasion*, *Arrival*, *Obstacle avoidance*, *Path following*, *Wall following*, *Containment*, *Collision avoidance*, *Cohesion*, *Leader following*, *Hide target following*, etc. Some computer realizations can be found in [14] [15] [3] [18]. In other side, new environments are developed; especially methods for terrain creation are introduced in computer graphics [7].

First part of the paper concerns the creation of 3D scenario. Terrain generation algorithm applies height map method. It uses isocline lines similarly as for conventional maps. The OpenGL library [16] [19] and GUI platform [17] is used to building 3D view of the terrain. Second part of the paper describes creation of the flying vehicles, its attributes, simulation of behaviors, and testing the program.

2 Creation of virtual environment

The problem of terrain generation is closely related to game theory and practice [4] [5] [8] [9] [10], military applications (unmanned flying vehicles, missiles or submarines), television and cinematography, where million agents preserve its individuality and separation, but forming a crowd. Such movies as Lord of the Rings, The Two Towers, and The Narnia Chronicles are produced with big help of computer simulation in virtual environment.

There are many methods to generate terrain in 3D environment. The most popular is based on height data, i.e. isoclines; the lines used for geometric terrain data and terrain texture maps. Others are based on fractal methods [8]. Such algorithm as Fault Algorithm and Mid Point Displacement Algorithm (see [6]) are popular. As it was mentioned above, an array of height data is applied here to generate terrain data. Between isoclines there are intermediate areas where the height is change moderately from one level to other. The area in height map is filled by gray scale texture. The height map is a graphic file. The examples of such objects are shown in Fig. 1. An example of the terrain generated is shown.

MapCreator algorithm [3] is developed for terrain simulation in 3D space with Cartesian system of axis (x, y, z). The point $(0, 0, 0)$ is in the middle of screen. It is typical for OpenGL library. The procedure allows introducing different shape of the terrain. The terrain objects are introduced separately layer by layer. Successive layers can have elliptical, circular, rectangular, and polygonal shapes with different height. It is possible also to introduce walls using pen and line options. The layers can have variable height. Pen options allow creating mountains with different shapes introducing layers as closed lines.

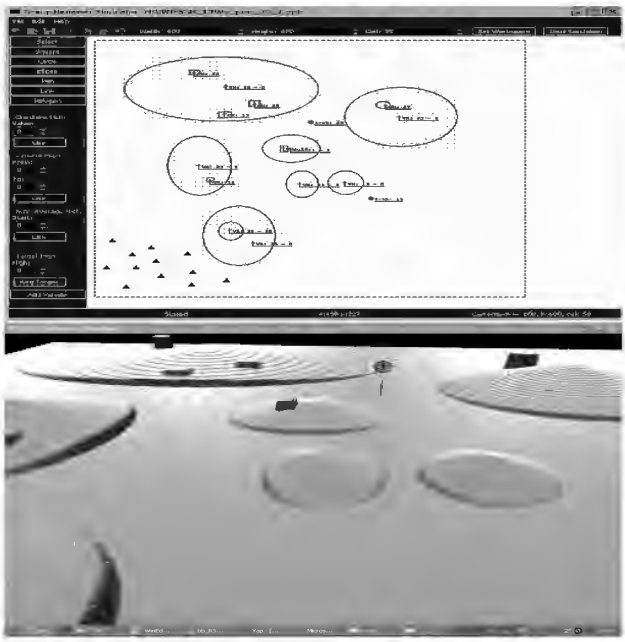

Figure 1: Example of height map and fragment of the terrain.

3 Movement of single vehicle

A member of the group moves over the terrain. This movement must fulfill many limitations. First, the vehicle must avoid obstacles and cannot enter in collision with others vehicles. Second, it must follow the leader and cannot fly away outside the group. If it happens it ought to join the group. In order to attain such behaviors, many changes of velocity are necessary. Moreover, good simulation of vehicle inertia is needed.

The inertia can be simulated by two methods. First method uses physical equations describing relation between force applied to the vehicle and its mass and acceleration. If rotation around longitudinal axis is taken in consideration then movement of vehicle is complicated. The distance is calculated by integration of velocity. It can to cause an error in vehicle's position. Second method, more popular, is based on slow change of velocity. If new velocity, resulting from rules of behavior, is calculated then it cannot be achieved immediately but the change is divided into parts. This method is applied here. Let actual velocity be described by vector $v_1 = [R_1, \Phi_1, \Theta_1]$, where R_1, Φ_1, Θ_1 are coordinates used by OpenGL Library. Let new desired vector of velocity be denoted by $v_2 = [R_2, \Phi_2, \Theta_2]$.

Then actual velocity is changed step-by-step using formula

$$v_{act} = [R_1 + i\frac{R_2 - R_1}{N}, \Phi_1 + i\frac{\Phi_2 - \Phi_1}{N}, \Theta_1 + i\frac{\Theta_2 - \Theta_1}{N}] \qquad (1)$$

where i denotes subsequent step and N is total number of steps during changing of velocity. In the simulation program $N = 10$ is used as default number. During simulation the image of the vehicle is rotated in such way to direct the "nose" of vehicle in accordance with actual velocity.

Any vehicle is represented in computer memory as separate thread. The treads are synchronized. In every moment only one thread is executed, others are sleeping. Any vehicle must interact with environment. All functions related to OpenGL Library are executed in main thread of *Simulation* program, among others a function drawing 3D environment. When drawing function is sleeping the thread of vehicle is awaken and it begins investigate the environment searching for possibility of collision. Partial modification of velocity to avoid collision with terrain obstacles is the result of searching. Other components of velocity are calculated as results from rules of behavior. The algorithm for obstacle avoidance is shown in Fig. 2.

The vector of actual velocity is normalizing to unit length. The searching ray with length d is sent according to actual direction of velocity. The length of rayon is enlarged in consecutive steps of the algorithm. If an obstacle is detected then distance to the obstacle is saved and used for modification of velocity. The algorithm of detection is complicated. The terrain is simulated using net of triangles. It is verified if the end of searching ray lays in any triangle of surface. If it is true, then normal vector to the terrain surface in this place, called flank vector, is calculated as mean value of normal vectors taken et 4 corners of terrain net placed near to the end of the ray. The vector is applied to modify actual velocity. New velocity is calculated as sum of actual vector and mirror vector to ray vector (see Fig 3). Similar procedure is used for *Flee* behavior, when vehicle flees an enemy.

In the paper is applied a modification of obstacle avoidance method, where bunch of 5 rays are used. Central ray is parallel to actual velocity and 4 rays are somewhat declined in 4 directions. New velocity is calculated as weighted mean value.

4 Modeling of behavior

Many studies are reported for group dynamics concerning individual behaviors, self-organization, and distributed autonomous system. The researchers studied how individual members behave in a group accompanied by a leader [1] [2] [11]

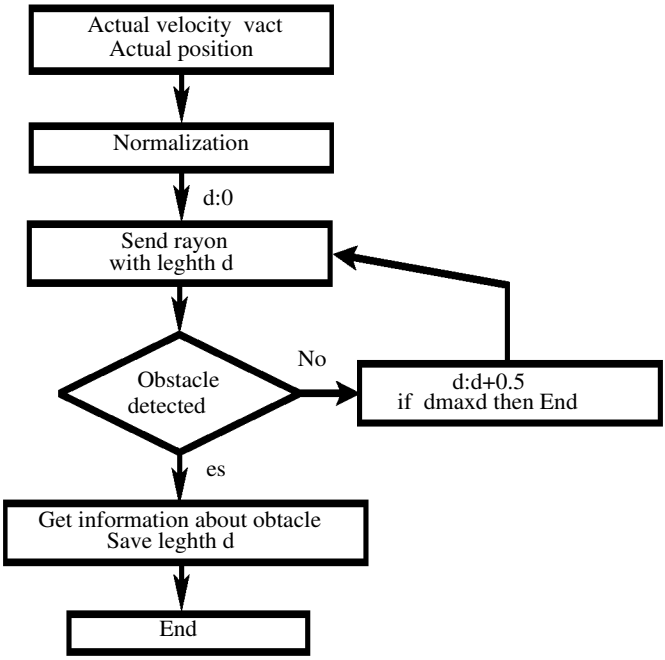

Figure 2: The algorithm for obstacle avoidance.

[12]. Most research has focused on techniques for modeling individual behavior of flock members inspired by Reynolds's boids [11]. Boids exhibit so-called emergent behavior in which characters only react to immediate events. Since Reynolds's paper in 1987, there have been a number of impressive papers on the use of behavioral models to generate computer animation.

In our investigations procedure of terrain generation was joined with main behavioral models as: seek and flee, obstacle avoidance, separation, alignment, cohesion, wander, arrive to target. Detailed description of these behaviors can be found in [12]. Herein the realization of some behaviors is shown only. As it was mentioned central and additional four beams looks for obstacles or other neighbor vehicles. Any beam measures a distance to nearest obstacle or vehicle on its way. Consider for example behavior *Separation*. Let our vehicle be surrounded only by other vehicles. For each of them the distance $d_i = pos_i - pos_0$ is measured, where pos_i is a position of the neighbor i and pos_0 is position of our vehicle. Now, new velocity v_{new} is calculated in order to avoid collision of vehicles. Resulting new

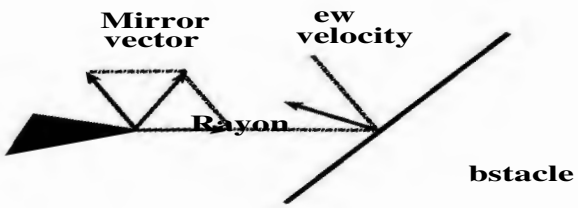

Figure 3: The method for obstacle avoidance.

velocity is calculated as weighted sum

$$\overrightarrow{v_{new}} = \sum_{i=1}^{n} \overrightarrow{d_i} * r_0/(T * r_i) \tag{2}$$

where r_0 is a constant, minimal distance taking in consideration for separation case, and T is time, step of simulation. The maximal area, where separation algorithm is active is limited by an ellipsoid. The value $T = 1$ is put for simulation. It is assumed that at most after 30 steps the collision between vehicles can be avoid. Maximal velocity is introduced to do the simulation more real. Separation behavior is shown in Fig. 4. Simultaneously with change of velocity, rotation of vehicle image is done. OpenGL library is used for it.

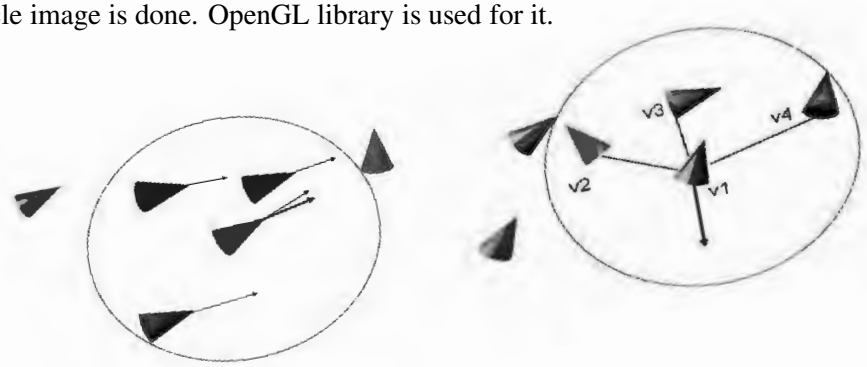

Figure 4: The behaviors *Alignment* (left) and *Separation* (right).

If any square distance is lower then threshold then necessary corrections is made. Basing on this distance matrix, a repulse force is calculated for any of agents, placed in the neighborhood circle. The forces calculated for any of agents pair are normalized by coefficient 1/dij. The repulse forces are summed for any agent and new desired force is calculated.

A flock has always a leader. Behavior *Leader following* is applied to all agents in the group. It allows maintaining appropriate distance between agents and leader

and does not allow disturbing the leader in leading (see Fig. 5. The leader looks for the target. Two strategies are possible. The leader knows where is the target or it must look for a target. In second case behavior called *Wander* is used.

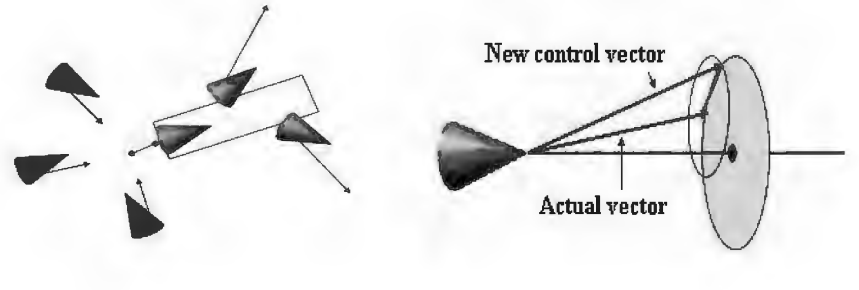

Figure 5: The behaviors *Leader following* (left) and *Wander* (right).

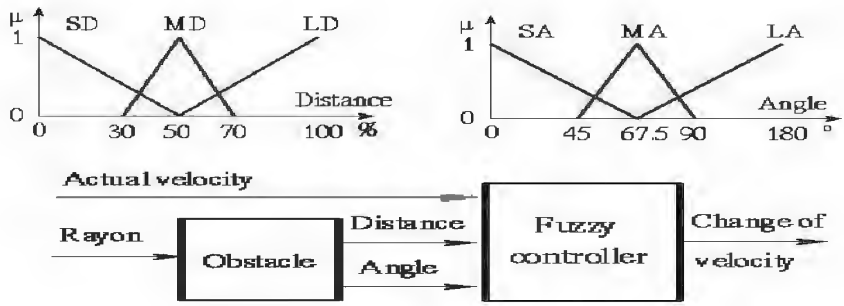

Figure 6: The *Obstacle avoidance* strategy.

Many of behaviors described above are in conflict. Thus, very important thing is to establish the priority. Fig. 7 shows the hierarchy of priorities. The resulting velocity is calculated as weighted mean, but it is possible to use fuzzy reasoning for this purpose.

5 Simulation

The algorithms were written using C++ and Visual Studio environment. Many trials were done with crisp and fuzzy version of the algorithms. Some examples are shown here.

Fig. 8 (left) presents forming the group and leader following (right). Behaviors *Obstacle avoidance*, *Alignment*, and *Cohesion* are included. It can be seen that

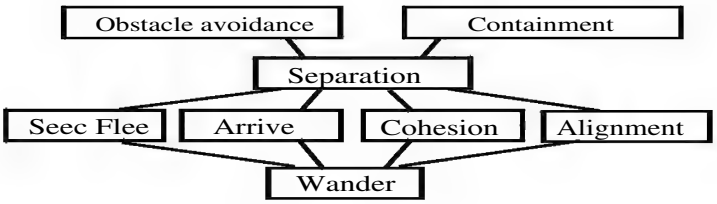

Figure 7: Priority of behaviors.

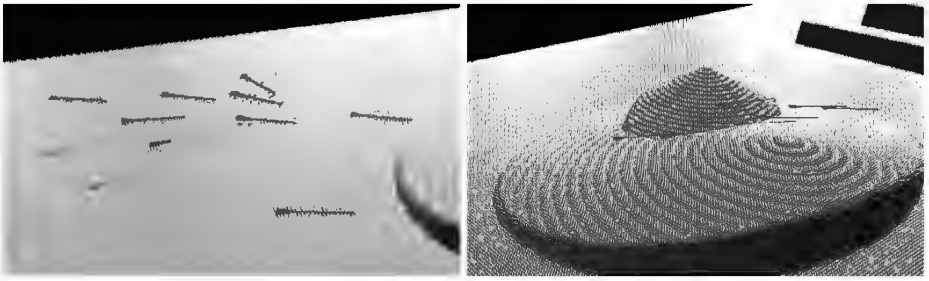

Figure 8: Leader forms the group which follows the leader.

vehicles follow the leader avoiding obstacles, fly in the same direction, they are separated, and distance between them is sufficient.

Next example, Fig. 9 (left), shows obstacle avoidance, where it is difficult. The group must be dispersed, but after it joins together. Fig. 9 (right) shows the arrival to target, where obstacle avoidance, alignment and cohesion are included. In this case it can be seen that the group is compact, separation is small, alignment is very good.

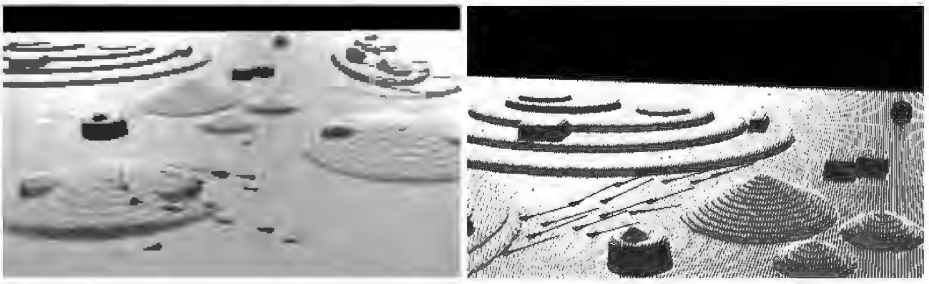

Figure 9: Behaviors: *Obstacle avoidance* (left) and *Arrive to target* (right).

6 Conclusions

The results obtained show that the best is simple version of algorithms, where fuzzy logic is applied for reasoning. The movement of the group was faster; there are no collisions between vehicles and obstacles. If more complicated behaviors are included as *Containment*, *Neighborhood* and sometimes *Cohesion* it arrives that a vehicle touch or pass slightly by an obstacle. It is a problem of behavior priority between *Obstacle avoidance* and *Containment* or *Cohesion* and *Alignment*, where the same priority levels are supposed. Therefore, form my experience the version with *Obstacle avoidance*, *Separation*, *Alignment*, *Seek* or *Flee*, and *Wander* gives good results. The behavior *Arrive to target*, when the movement is slowing near the target can has smaller priority, similar as *Wander*. In further realizations, it is intended to improve the terrain generation algorithm in order to obtain smoother view.

References

[1] Arkin R. (1992) Behavior-based Robot Navigation in Extended Domains. Journal of Adaptive Behavior, 1(2) 201–225.

[2] Blumberg B., Galyean T. (1995) Multi-Level Direction of Autonomous Creature for Real-Time Virtual Environments. Computer Graphics Proceedings, SIGGRAPH 95.

[3] Cieċiara D. (2010) Modeling of behavior of object group (in polish). Warsaw Univ. of Technology, BS diploma work.

[4] Davison A. (2005) Introduction to Java 3D. Computer Engineering Lab IV, pp. 240–302.

[5] Davison A. (2005) Killer Game Programming in Java. O'Reilly.

[6] Fernandes A. R.: Terrain Tutorial.
http://www.lighthouse3d.com/ opengl/terrain/

[7] Karlsson B. (2005) Beyond the C++ Standard Library: An Introduction to Boost. Addison Wesley

[8] Martz P.: Generating Random Fractal Terrain.
http://www.game-programmer.com

[9] Millington I., Funge J. (2009) Artificial Intelligence for Games, 2-nd ed. Morgan Kaufmann Pub., Amsterdam.

[10] Motekew K. (2009) Modeling, Simulation, & Control with Java3D Visualization. http://vehsim.sourceforge.net/#jv.

[11] Reynolds C. W.: Flocks, herds, and schools: A distributed behavioral model. Computer Graphics, SIGGRAPH Conf. Proc. 25–34 (1987)

[12] Reynolds C. W.: Steering Behaviors For Autonomous Characters. Conf. Proc. Game Developers Conference, 763–782 (1999)

[13] Reynolds C. W. (2000) Interaction with Groups of Autonomous Characters. Proc. Game Developer Conference, San Francisco, pp. 449–460.

[14] Sauer J.: Steering behavior. http://fbim.fh–regensburg.de/∼saj39122 /feisch/Diplomarbeit/theory_eng/steering_eng.html

[15] Schnellhammer C., Feilkas T.: Steering behaviors. www.steering behaviors.de.

[16] Seddon C. (2005) OpenGL Game Development. Wordware Publishing, Inc.

[17] Smart J., Hock K. (2006) Cross - Platform GUI Programming with wxWidgets. Parentice Hall.

[18] Soleniec E. (2010) Fuzzy controller implementation in the steering of a group of autonomous agents (in polish). MS dissertation, Warsaw Univ. Technology.

[19] Wright R.S. Jr., Lipchak B., Haemel N. (2007) OpenGL Super Bible, 4th Edition. Addison Wesley.

The papers presented in this Volume 2 constitute a collection of contributions, both of a foundational and applied type, by both well-known experts and young researchers in various fields of broadly perceived intelligent systems.
It may be viewed as a result of fruitful discussions held during the Ninth International Workshop on Intuitionistic Fuzzy Sets and Generalized Nets (IWIFSGN-2010) organized in Warsaw on October 8, 2010 by the Systems Research Institute, Polish Academy of Sciences, in Warsaw, Poland, Institute of Biophysics and Biomedical Engineering, Bulgarian Academy of Sciences in Sofia, Bulgaria, and WIT - Warsaw School of Information Technology in Warsaw, Poland, and co-organized by: the Matej Bel University, Banska Bystrica, Slovakia, Universidad Publica de Navarra, Pamplona, Spain, Universidade de Tras-Os-Montes e Alto Douro, Vila Real, Portugal, and the University of Westminster, Harrow, UK:

Http://www.ibspan.waw.pl/ifs2010

The consecutive International Workshops on Intuitionistic Fuzzy Sets and Generalized Nets (IWIFSGNs) have been meant to provide a forum for the presentation of new results and for scientific discussion on new developments in foundations and applications of intuitionistic fuzzy sets and generalized nets pioneered by Professor Krassimir T. Atanassov. Other topics related to broadly perceived representation and processing of uncertain and imprecise information and intelligent systems have also been included. The Ninth International Workshop on Intuitionistic Fuzzy Sets and Generalized Nets (IWIFSGN-2010) is a continuation of this undertaking, and provides many new ideas and results in the areas concerned.

We hope that a collection of main contributions presented at the Workshop, completed with many papers by leading experts who have not been able to participate, will provide a source of much needed information on recent trends in the topics considered.

ISBN-13 9788389475367
ISBN 838947536-7