A virtual environment for solving plane geometry problems

Marco Ronchetti
(ronchet@science.unitn.it)
Facoltà di Scienze, Dipartimento di Informatica e Studi Aziendali
Università di Trento, 38100 Trento, Italy

1. Preface

My interest in the usage of computers for didactic purposes dates back to discussions I had with Marko Jaric when we envisioned a project for writing an interactive book for teaching nonperiodic structures and quasicrystals. We never actually started the enterprise (life is always complex), but a few years later, when Marko had already passed away, I began a project about using computers for teaching elementary geometry. The work is somehow inspired by the old discussions we had in Santa Cruz.

2. Introduction

2.1 The panorama

In a world that is changing at an impressive pace, education is bound to become a lifelong activity. The way we provide education also needs to change, to lower its costs and increase the efficiency of the educational process.
It is commonly believed that by exploiting multimedia and computer networks it will be possible to provide some form of education in an efficient and effective way. In fact, recent advances in hardware technology have delivered to the final user an impressive amount of computational power while dropping the cost of personal computers to the level of commodities like HiFi's or large TV sets. The net effect is that by now many families can afford to give to high school kids the computational power that only a few years ago could only be delivered by a supercomputer. Such huge power can be used to exploit the communicative power of multimedia. Moreover, in a few years Internet or its evolution will become almost as common as telephones.
However, it is not clear how in practice the educational process can profit of the new media. Much research and imagination is needed to find a convincing answer. Since the sixties the concept of computers taking the role of teachers (or at least helping them) has been with us, but in spite of the frequent change of name of the discipline (from Computer Aided Instruction to Computer Based Education, to Intelligent Tutor Systems etc.) embarrassingly little has been achieved.

Education is about transferring knowledge (to know), abilities (to know how to do) and behavior (to know how to be). It is mostly achieved in a person to person interaction, helped by the delivery of a body of knowledge packaged in an organized way (books) and by (individual or group) activities which are focussed to anticipate and/or reproduce situations which might arise in "the real world" (e.g. laboratory activity).
Focus on the knowledge to be transferred has lead to the creation of hyperbooks and to the construction of extensive Virtual Libraries. The process is in progress at a quick

A virtual environment for solving plane geometry problems

Marco Ronchetti
(ronchet@science.unitn.it)
Facoltà di Scienze, Dipartimento di Informatica e Studi Aziendali
Università di Trento, 38100 Trento, Italy

1. Preface

My interest in the usage of computers for didactic purposes dates back to discussions I had with Marko Jaric when we envisioned a project for writing an interactive book for teaching nonperiodic structures and quasicrystals. We never actually started the enterprise (life is always complex), but a few years later, when Marko had already passed away, I began a project about using computers for teaching elementary geometry. The work is somehow inspired by the old discussions we had in Santa Cruz.

2. Introduction

2.1 The panorama

In a world that is changing at an impressive pace, education is bound to become a lifelong activity. The way we provide education also needs to change, to lower its costs and increase the efficiency of the educational process.
It is commonly believed that by exploiting multimedia and computer networks it will be possible to provide some form of education in an efficient and effective way. In fact, recent advances in hardware technology have delivered to the final user an impressive amount of computational power while dropping the cost of personal computers to the level of commodities like HiFi's or large TV sets. The net effect is that by now many families can afford to give to high school kids the computational power that only a few years ago could only be delivered by a supercomputer. Such huge power can be used to exploit the communicative power of multimedia. Moreover, in a few years Internet or its evolution will become almost as common as telephones.
However, it is not clear how in practice the educational process can profit of the new media. Much research and imagination is needed to find a convincing answer. Since the sixties the concept of computers taking the role of teachers (or at least helping them) has been with us, but in spite of the frequent change of name of the discipline (from Computer Aided Instruction to Computer Based Education, to Intelligent Tutor Systems etc.) embarrassingly little has been achieved.

Education is about transferring knowledge (to know), abilities (to know how to do) and behavior (to know how to be). It is mostly achieved in a person to person interaction, helped by the delivery of a body of knowledge packaged in an organized way (books) and by (individual or group) activities which are focussed to anticipate and/or reproduce situations which might arise in "the real world" (e.g. laboratory activity).
Focus on the knowledge to be transferred has lead to the creation of hyperbooks and to the construction of extensive Virtual Libraries. The process is in progress at a quick

pace, but much has to be done, especially for enhancing the retrieval of meaningful information.
However "the next wave of educational support ... moves beyond providing access to more information to providing support for the process of engaging in the solution of complex problems as a collaborative endeavor" [Watt 98].
As a matter of fact, the exploitation of the new technology is aimed at enhancing education on at least three more facets:

- Creation of collaborative environment to facilitate the interaction among students.
- Enhancement of the interaction between student and a teacher/tutor/mentor;
- Creation of virtual environments for safe, inexpensive experimentation;

Collaborative environment and interaction between student and teacher can profit of telecommunication technology, which can be synchronous (like videoconference) or asynchronous (e-mail, notes delivered as Web pages, shared bulletin boards or ad-hoc collaborative tools). However the big challenge here is probably to understand how the communication through the media differs from a face to face interaction, and to find a methodology which effectively uses the available technology.
Virtual environments are in use since many years in dangerous or costly activities, and have a formidable success in the form of videogames, much less for high school education, although simulation activities and virtual labs start now to be rather common. Of course it would also be nice to be able to synthesize a tutor which monitors the activity of the student, and helps her/him when difficulty arise. However, most of the efforts aimed at producing intelligent behavior from machines have failed, and the ambitious dreams of Artificial Intelligence have not survived into the 90'.
Is spite of the limited successes obtained, it is possible to focus on simple, limited domains and to produce systems which can exhibit at least some degree of intelligent behavior.

2.2 Goals of the project

I envisioned a project for teaching elementary geometry using a virtual environment that embodies some amount of intelligence. The final tool will allow a high school student to read a problem regarding plane geometry and to use the computer to:

- Draw the elements which define the problem (points, lines, arcs, angles, polygons),
- Express the relations among elements contained in the hypothesis,
- State the thesis,
- Build a demonstration by expressing facts about the elements.

The project has several stages, the first of which has been completed in a prototypal way [Tave99]. The first generation tool "knows" theorems about plane geometry, and is therefore able to check the validity of the assertions made by the student, rejecting them if they are false. The student is therefore helped building a demonstration, since false assertions are caught early.

The second stage, now in progress [Brug99], develops an automatic theorem prover that can construct a demonstration as soon as the thesis is expressed. If the student is in trouble and does not know how to proceed, s/he can ask for help: the tool extracts from its demonstration some suggestion that can help the student proceed.

The third phase, currently at a project stage, recognizes that often the main difficulty for the student is to correctly understand the statement of a problem and to build a correct drawing for it. We plan to use an existing natural language interpreter to build a representation of the problem so as to be able to validate the drawing made by the student, and in case it is incorrect to provide some feedback.

Of course the system must be easy to use, and the user interface must be simple and powerful. The last challenge would be to dress the system so as to make it appealing like a game (maybe also incorporating facilities for collaborating with other students).

3. Geometry theorem proving

3.1 The background

Programs that prove geometry problems date back to the early years of computer science, when in the early spring of 1959 the program "Geometry Machine" by Gelernter [Gele59] proved its first geometry theorem mechanically using an axiomatic approach. Since then many research groups worked on the problem: a bibliography of the most important and significant papers on geometry theorem proving runs for twelve pages [Wang94].
An axiomatic approach however makes proving and discovering non-trivial theorems difficult, because the problem of a too large searching space makes the method highly inefficient. In recent years the availability of larger memories and faster CPU's has revitalized the assiomatic approach: Quaife used the automated reasoning program OTTER to prove a large number of theorems in plane geometry [Quai92].

A symbolic manipulation approach was followed by Cerutti and Davies [Ceru] in 1969, but a breakthrough in geometry theorem proving was obtained by Wu ten years later [Wu78], using an elegant algebraic approach. Wu observed that most geometric relations can be expressed by means of polynomial equations. Therefore, proving most geometric theorems can be reduced to manipulating the corresponding algebraic relations. A similar approach based on the application of Gröbner bases was first applied by Buchberger [Buch85] and has often been used since. The main problem with the algebraic approaches is that the constructed proofs are not readable and interpretable: the prover therefore becomes a sort of black box, which outputs a boolean value.
An approach based on the resolution of logical formulae is not as "black", but certainly at least a "gray" box: an output of hundreds of logical clauses can hardly be used to "understand" a proof.

Other approaches express statements on geometric entities such as distances, vectors, areas and volumes rather than point coordinates, and use algebraic manipulation on them to produce shorter and more readable demonstrations. This so-called coordinate-free method is usable for some classes of geometry problems and started with the work by Crapo [Crap87].

3.2 Our approach

Our original aim was to re-use whenever possible the available knowledge, and the relative software. Unfortunately, after testing a few prover available on the net, we realized that all proofs produced by the above mentioned methods are unusable to guide a high school student in her/his discovery process. On the other hand, we do not need terribly efficient algorithms that are able to prove very complex problems. We therefore decided to develop our own prover, the requirement being that its "reasoning" should be as close as possible to that of a human being.

Our first choice was never to rely on numerical values (for coordinates, length, area etc.) since we want our demonstrations not to depend on the particular instance chosen. We wanted to recognize configurations, in which one of the known theorems can be applied; apply the theorem and add its consequences among the known facts. By iteration then one would proves the thesis, or reaches a dead state in which no new information can be added. This limits the scope to the problems which can be solved without making geometrical constructions beyond what is suggested by the thesis: it is not a too severe limitation as long as only high school problems are involved.

The second choice was to approach the problem in an Object-oriented way. We chose Java as implementation language for several reasons:

- Java is a clean Object Oriented language (unlike C++ or, worse, Visual Basic)
- Java has a nice set of graphic classes
- Java is portable on any platform (although most of the world is running Wintel PC's, this is anyway a plus).

We developed a hierarchy of classes representing geometric primitives. These are called G_Point, G_Line, G_HalfLine, G_Segment, G_Angle, G_Triangle. They are all subclasses of an abstract class called G_Object which implements some common methods (e.g. setName) and defines an interface for some other method (e.g. draw) so as to force the implementation by the subclasses. G_Objects have a graphic representation so that it is possible to draw them on screen, but their geometric coordinates are in no way used in the demonstration. One of our final goals is to be able to show the student that if the original figure is stretched (but the hypothesis constraints are respected) the demonstration still holds.
Relationships among instances of these classes represent properties. Properties of G_Objects can be Reflexive, Symmetric and Transitive (RST-Properties: e.g. equal amplitude of G_Angles) or Symmetric (S- Properties: e.g. two G_Lines can be orthogonal to each other). The implementation of RST-Properties and S-Properties as sets and relations among sets allows to some degree a quick and simple discovery of new facts, as we shall discuss in the next two paragraphs.

3.3 The RST-Properties

Let us consider the RST-Properties which can be interesting (e.g. parallelism of straight lines, equal length for segments, etc). Each RST-Property defines an equivalence class, which we will call a G_Set.
When a G_Object is instantiated, a new G_Set is associated to each RST-property available for that G_Object. For instance, when a line is created, the G_Set of the lines parallel to it will be created. The G_Set is initially populated only with the G_Object (i.e. a line is parallel to itself).
When two G_Object are proved to satisfy an RST-property, the corresponding G_Sets are merged. For instance, let us assume that we already know that line r1 is parallel to line r2: r1 and r2 therefore share a common G_Set. Let's assume there also is a third line r3 about which we do not know anything. We can represent the configuration as:

Parallel{r1,r2}   Parallel{r3}

meaning: a G_Set relative to the Parallel RST-Property contains r1 and r2, and a separate G_Set relative to the Parallel RST-Property contains r3.
Now, let's imagine that r3 is proved to be parallel to r1: we must therefore merge the G_Set of r1 and the G_Set of r3. The resulting G_Set contains r1, r2, and r3;

Parallel{r1,r2,r3}

We therefore now know that r3 is parallel to r2. A possibly large number of logical inferences are done in one single step.

By choosing G_Set to be a subclass of G_Object, we allow recursivity. That means that also a G_Set (being a G_Object) can have a G_Set associated to an RST-property. As an example let's consider the similarity of triangles. A G_Triangle (subclass of G_Object) has a G_Set linked to the property "Congruent". The G_set has another G_Set linked to the property "Similar". Let us now we consider three triangles A, B and C, and let A be congruent to B, and C similar to B. There is another triangle D about which we do not know anything. The resulting structure can be represented as:

Similar{Congruent {A, B}, Congruent {C}}　　　Similar{Congruent {D}}

By stating that C is congruent to D, we unify the G_Set Congruent relative to C and D, and the resulting structure becomes:

Similar{Congruent{A, B}, Congruent{C,D}}

From the resulting representation it can be easily seen that now D is known to be similar to A.

3.4 The S-Properties

An S-Property is connected to an RST-property. For instance, S-Property "orthogonality" is linked to RST-property "parallelism", i.e. it is a relationship between two G_Sets. Therefore, if we know that r1 is parallel to r2, and r3 is orthogonal to r4, the resulting structure can be represented as:

Orthogonal[Parallel{r1, r2}, Parallel{ }]　　　Orthogonal[Parallel{r3}, Parallel{r4 }]

When we now state that r3 is orthogonal to r1, the resulting structure becomes:

Orthogonal[Parallel{r1, r2, r4}, Parallel{r3 }]

The system therefore has "discovered" that r4 is parallel to r2.

3.5 Sum and product of geometric Objects

Another class of relations is relative to the sum of certain properties (e.g. length), and to their product (e.g. area). Again, our approach is based on sets, since we never make use of numerical coordinates. However, nested relations (like in the case of segments A, B, C, D, E with A=B+C and C=D+E) can be expensive if treated in a straightforward way, requiring costly exploration of tree structures. We decided to implement a system that allows a more efficient representation and identification. G_Objects are associated to a unique ID, which is obtained setting a single bit in a field of many zeros. The field must contain at least as many bits, as many objects are present, and therefore can be rather long. In our example,

A {00001}
B {00010}
C {00100}

D {01000}
E {10000}

The bit sequences can be represented in a more compact form as integer numbers: A {1}, B {2}, C {4}, D {8}, E {16}. Of course we need very long integers, but this is not a problem since they can be simulated by hand, or one can use special Java classes from the package Java.math.

Now, each G_Object has at least one representation, but can have more than one of them. For instance, since A=B+C we choose to associate to A two representations:

A=B+C => A {00001, 00110} => A {1, 6}

The second representation shows the sum. When we discover that C=D+E, we update the representation of C, which becomes

C=D+E => C {00100, 11000} => C {4, 24}

and scan all the available representations of other elements. In particular, since A=B+C we want to introduce an additional representation of A as A=B+D+E.

A{00001, 00110, 11010} =>A{1, 6, 26}

Recognizing that the bit sequences can be mapped into integers, all operations become quicker and easier than one might think. In fact, when we know that A=B+C and we want to express that C=D+E (and therefore A=B+D+E), all we need to do is:
- verify that a possible representation of A contains C, which is easy and quick since the answer is true if the bitwise operation (A/x) AND r(C/C) gives r(C/C);
- subtract r(C/C) from R(A/x), and add to the result r(C/D+E).

Here A/x means some representation of A, C/C means the basic representation of C and C/D+E means the representation of C as sum of D+E. In our example the steps are:
- 6 AND 4 = 4 (a representation of A contains C)
- Build a new representation of A as 6-4+24=26, which is what we expected.

The representation can be updated as soon as the new information is available (eager approach) or when it is needed (lazy approach). We are still in the process of evaluating which solution is more efficient for our needs.

A problem arises when one wants to sum two (or more) occurrences of the same object, since this would break the convention that sequences containing one single non-zero bit are to be interpreted as basic elements. For instance, if we tried to assert that A=C+C we would end in trouble, because C+C yields the basic representation of D (4+4=8). We fix this by associating a flag to each basic representation, signaling that the representation is actually basic, or that it has to be considered as composite. When we detect a collision (i.e. the result of a sum hits an already existing basic representation), we rearrange the representations so as to leave space to the non-basic representation and we use a slightly modified algorithm and take multeplicity into account.

The same apparatus can be extended to treat products among G_Objects.

3.6 The Theorems

The basic theorems needed (e.g. angle-side-angle) are explicitly coded into routines that receive as input some G_Object. Preconditions of the theorem are evaluated on the G_Objects, and if they are satisfied the consequences (postconditions) of the theorem are applied to the elements.

In the first version of our system, the program applies the theorem suggested by the student. If the preconditions are false, the "move" is rejected and the student is warned, otherwise the postconditions modify the internal data structure, i.e. enrich the knowledge about the problem.

In the second version, the program takes the initiative of trying to apply the known theorems to the available elements, and iterates till either the thesis has been proved or a fixed point is reached (i.e. application of known theorems does not produce changes in the data structure).

Which theorems should be implemented? A critical balance must be achieved between a large number of specialized theorems (which might require fewer iterations, but in each iteration one would have to test many theorems) and a small number of general theorem (less powerful, but with quicker iterations). We think the answer should be in the hand of the final user. In fact, by using specialized theorems that are not known to the student, the system might well find a proof, but the proof would not be understandable for the student. On the other hand, too little knowledge of basic theorems would (in the best case) lead to long and boring demonstrations. So we believe that the student (or the teacher) should have some degree of freedom in deciding which theorems to "enable" as possible building blocks for the demonstration.

Moreover, during the automatic discovery process the system will in general prove a set of accessory facts, which are not relevant for reaching the thesis. A backward inspection from the thesis toward the hypothesis will allow dropping the unneeded assertions, producing a cleaner proof.

4. Conclusions

We tried a preliminary version of the prover, which was indeed able to find the solution of some simple problem in a few seconds. The listing of the steps performed in searching the solution was short and understandable, as we desired.

The next steps are to consolidate the results, studying how the system scales with problem complexity. We also intend to get a friendlier user interface. Finally we hope to test soon the program "on the field" by putting it to work in a classroom.

As we mentioned earlier, the third phase also contemplates the integration of a natural language engine for starting from the written problem rather than from its graphical representation.

As a final remark, we remember that our final goal was not to produce a more efficient theorem prover, but a more useful one. We do not need a tool that is able to discover new theorems in geometry, but rather an instrument which is innovative in the use of computers in a classroom, and most importantly actually usable and useful for the students.

5. References

[Brug99] Novella Brugnolli, Tesi di Laurea in Matematica, Università di Trento 1999

[Buch85] B.Buchberger, " Gröbner bases, an algorithmic method in polynomial ideal theory", Multidimensional Systems Theory (N.K.Bose ed.), D. Reidel Publishing, Dordrecht-Boston pp 184-232

[Crap87] H.Crapo "Computer-Aided Geometry Reasoning", Proc. INRIA Workshop (INRIA Sophia-Antipolis, June 22-26,1987) INRIA Rocquencourt, France

[Ceru 69] E.Cerutti and P.J.Davis "Formac meets Pappus: some observation on elemenatry analytic geometry by computer", Amer.Math.Mnthly, 76, 895-905 (1969)

[Gele59] H.Gelernter, "Realization of a Geometry Proving Machine", Proc.Int.Conf. on Information Processing (Unesco, Paris, June 1959) pp.273-282

[Quai92] A.Quaife, "Automatic Development of Fundamental Mathematical Theories", Kluwer Academic Publisher, 1992

[Tave99] Paolo Tavernini, Tesi di Laurea in Matematica, Università di Trento 1999

[Wang94] D.Wang, "Medlar II - Esprit BRP6471 - Deliverable DII.4.1P. can be found at http://medlar.doc.ic.ac.uk/lar/subtaskII41.html

[Watt98] C.Watters, M.Conley and C.Alexander, "The digital Agora: using technology for learning in the social sciences", Commun. ACM 41 (Jan 1998) 50-57

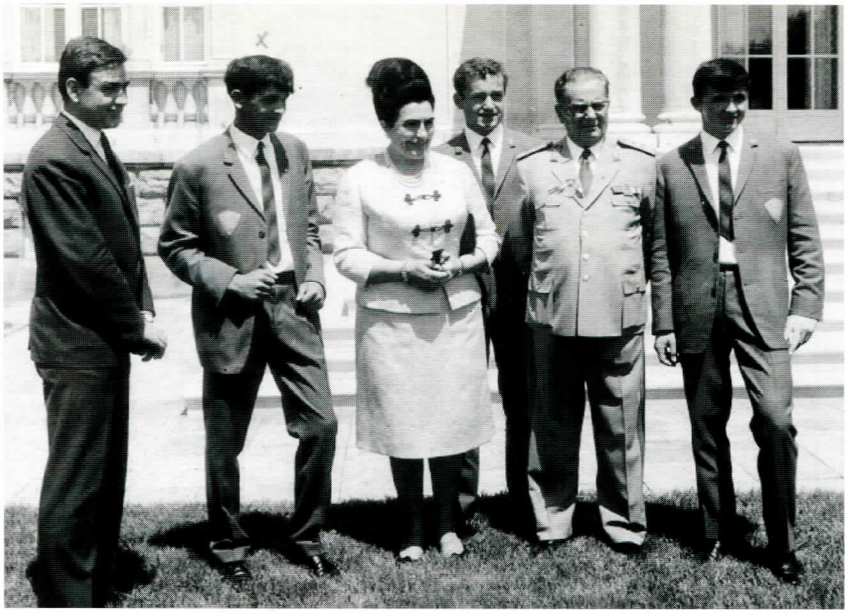
[Wu78] W.T.Wu, "On the decision problem and the mechanization of theorem proving in elementary geometry" Sci Sinica 21, 159-172 (1978)

VIRTUELNA SREDINA ZA REŠAVANJE GEOMETRIJSKIH PROBLEMA U RAVNI

Marko Ronketi

Interesovanje za korišćenje računara u didaktičke svrhe potiče još od diskusija sa Markom Jarićem, kada smo razmatrali projekt pisanja interaktivne knjige za proučavanje neperiodičnih struktura i kvazikristala. Realizaciju poduhvata nikada nismo započeli (život je komplikovan), međutim nekoliko godina kasnije, kada je Marko već preminuo, pokrenuo sam projekt korišćenja računara za učenje elementarnih pojmova geometrije. Na neki način ovaj je rad inspirisan davnim diskusijama koje smo imali u Santa Kruzu.

На насловној сӣрани "ФРОНТА, 1970. године
On the cover page of "FRONT", 1970.



У друштву председника Тита 1970. године
In the company of President Tito, 1970.

Говор испред гимназије "Маршал Тито", 1990. године,
за 20. годишњицу матуре
Addressing the Academy "Marshal Tito" in 1990, for the 20th high school reunion

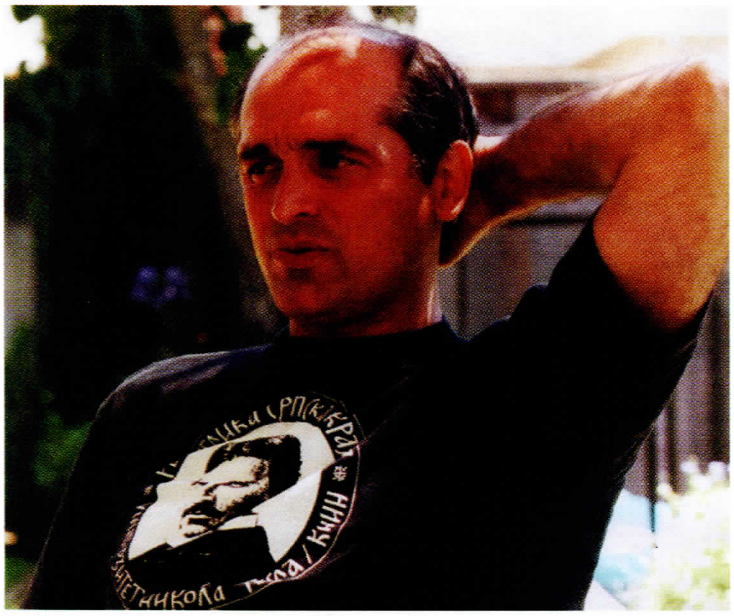
Црква и црквени центар "Riverside" (десно) и Грантов Гроб (лево) у Њујорку. Иза Грантовог гроба је студентски дом "International House" у коме је Марко живео, а горе лево у позадини је "City College of New York" у коме је студирао 1974-1978. Riverside Church and the Interchurch Center (right) and Grant's Tomb (left) in NYC. Behind the Tomb is "International House" in which Marko lived, and back in the upper left is "City College of New York" in which Marko studied in 1974-1978.

С одбране Марковог доктората, новембра 1977. године. Лево на слици је тадашњи југословенски студент, а садашњи познати физичар, Горан Сењановић (живи и ради у ITCP у Трсту). У Средини је Спента Вадиа, познати индијски физичар. Thesis defense, November 1977. To the left is Goran Senjanović, another Yugoslavian student at CCNY at the time (today at ICTP in Trieste). In the Middle is Spenta Wadia, a noted Indian physicist.

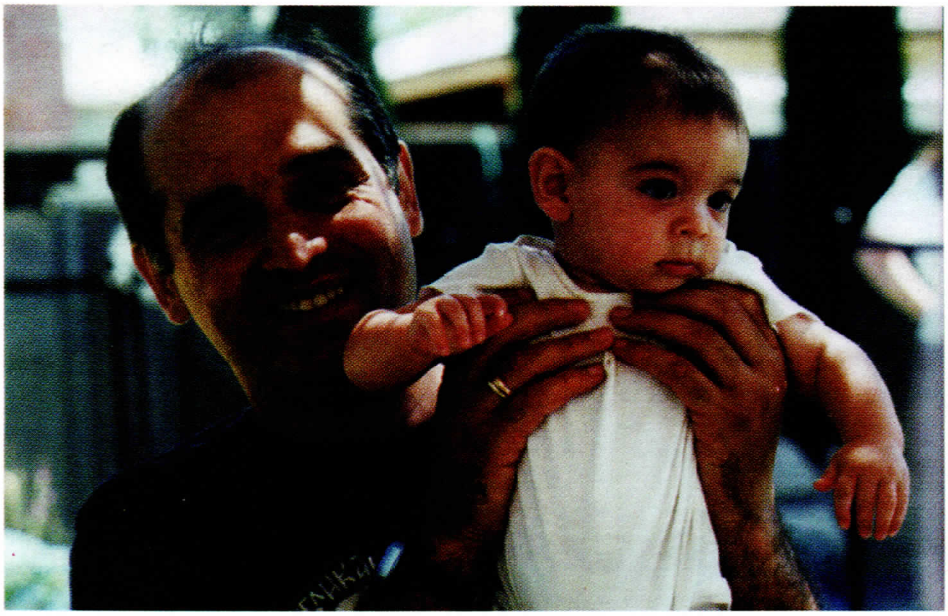
Прослава рођења сина Александра Шокорца 1976. године у студентском дому. У првом реду: Миливој Белић, Горан Сењановић, Жељко Антуновић, Александар Шокорац и Славица Јарић – прва Маркова супруга. Седе: Радмила Јевицки и Даница Сењановић. Девојчица је Наташа, Горанова и Даничина ћерка. Сликао Антал Јевицки.
Birth celebration of A. Šokorac's son, 1976 in I. House. In the first row: M. Belić, G. Senjanović, Ž. Antunović, A. Šokorac i S. Jarić – the first wife of Marko. Sitting: R. Jevicki and D. Senjanović. Little girl is Nataša, Goran's daughter. Picture taken by A. Jevicki

Три пријатеља још од студија у Београду и Њујорку, у кинеском ресторану у College Station-у, Тексас, 1990. година. Лево: Антал Јевицки, професор на Браун универзитету, у средини: Миливој Белић, професор у Институту за физику, Београд.
Three friends since the student days in Belgrade and NYC, in the Chinese restaurant in College Station, TX, 1990. Left: Antal Jevicki, Professor at Brown University, in the middle: Milivoj Belić, Professor at the Institute of Physics, Belgrade.

На циљу маратонске трке у Сан Франциску, 1990. године
At the finish of the San Francisco Marathon, 1990.



Опоравак после прве операције, у мајици Универзитета „Никола Тесла“,
Книн, 1996. године
Recovery after the first operation, in the T-shirit of the University „Nikola Tesla“,
Knin, 1996.



Са сином Војином, 1996. године
With son Vojin, 1996.