FUNCTIONAL REPRESENTATION IN COMPUTER GRAPHICS

Alexander P. Penev, Dimcho S. Dimov, Dobromir P. Kralchev

This paper presents some basic concepts of F-Rep and R-Functions used in computer graphics. Here we made short comparison of B-Rep and F-Rep. We also discuss advantages and some important development aspects of so-called hybrid systems. We consider goals, directions, and applications of the developed, by the authors, experimental open hybrid system OpenF.

1. Introduction. One of the most frequently used representation schemes in geometric modelling and computer graphics is the Boundary Representation (B-Rep). A reason is existence of hardware graphic accelerators that make fast visualization of B-Rep-based geometric models (scene).

Research work in a representation scheme, named Functional Representation (F-Rep) [1], started growing and developing intensively during the last years. F-Rep is based on a scene models described with the help of implicit functions, R-functions, etc. In theory, it is not so new and could be found in Rvachev's works [6], [7], and other.

However, lately availability of more possibilities of applications is appreciated and their speed is no so important. F-Rep is one of the most powerful representation schemes in computer graphics. It offers easy integration possibilities of other representation schemes. Sometimes F-Rep is much more effective and inexpensive than B-Rep. These specificities make it more attractive for using in future applications. The only fact (for now), which stops F-Rep wide spread is lack of specialized hardware for fast visualization of F-Rep-based models. However, with progress of computer hardware and graphic accelerators, this may be changed soon [2] and open the door to wider application of F-Rep.

2. F-Rep. Functional representation (F-Rep) is a representation scheme that is used for describing geometric objects (solids). The F-Rep [1] defines a geometric object by a single real continuous function f with parameter one point from Euclidean space, defined as:

$$f : X \longrightarrow \mathrm{R},\ X \in E^n \tag{1}$$

The function (1) may define geometric information $G = (\{f\}, \varnothing, \varnothing)$ and it induces point set $S_G = \{X \in E^n \mid f(X) \geq 0\}$.

Another way to interpret value of function f is as signed distance from point X to surface of solid S_G. These functions are so-called signed distance functions. Using these functions is more restricted requirement, but it gives some advantages of modelling with such functions.

Often F-Rep also means embedding of other schemes and methods (CSG, volumetric objects, blobby objects, sweeping, parametric models, etc.), as parts of the F-Rep scheme.

The geometric operations in F-Rep are defined analytically. For example, set-theoretic operations are implemented with using of so-called R-functions [7], [1] (see (2)). Other known operations are [1]: blending, offsetting, Cartesian product, metamorphosis, bijective and linear mapping, projection, etc. Base relations in F-Rep are point membership, inclusion, existence of intersection, etc. More complex methods, used in F-Rep, are reconstruction of solids from cross-sections or from their point set, collision detection, converting B-Rep and CSG into F-Rep (by R-functions), converting F-Rep into B-Rep (so-called polygonization) for visualization goals (marching cubes, marching triangles, adaptive polygonization, particle systems polygonization, for example).

Visualization algorithms are two classes: Polygonization based and Ray-tracing based. The first class converts F-Rep into B-Rep, which is visualized by B-Rep approaches (Z-Buffer, for example). Polygonization is also used for other goals not only for visualization. The second class makes visualization directly from observer's viewpoint, without intermediate conversion. Basic operation in these algorithms is finding nearest point in some direction (intersection of a ray with a model). There exists a fast finding method, when functions are normalized.

A fundamental advantage of F-Rep is its openness (extensibility) in direction of adding new primitives, operations, and relations. Large advantage is also easy implementation of nonlinear transformations and other complicated operations. For example (hard to be made in B-Rep) operation metamorphosis (morphing one solid into another solid), in F-Rep there is a simple solution $f_m(t) \equiv t \cdot f_1 + (1-t) \cdot f_2, t \in [0,1]$, where f_1 and f_2 are functional representations of two solids, t is parameter determining intermediate solid (morphing phase). However, the use of so-called R-functions leads to bigger power of F-Rep.

3. R-Functions. R-functions [7] are real functions of real variables which inherit some properties of logical functions (binary or ternary logic are used). A real function of real variables is called an R-function if it can change its sign if and only if at least one of its arguments has changed its sign.

R-functions allow us to create a function for an almost arbitrary shape easily, in the same way as we write a logical expression. For example conjunction $X \wedge Y$ is (so-called) the logical friend of R-function (2).

There are some systems of R-functions with different properties applicable for goals of geometric modelling and computer graphics. Most often used are (2), (3) and (4) respectively for intersection/conjunction, union/ disjunction, and complement/negation.

(2) $$f_1 \wedge_a f_2 \equiv \frac{1}{1+a} \cdot \left(f_1 + f_2 - \sqrt{f_1^2 + f_2^2 - 2a \cdot f_1 \cdot f_2} \right)$$

(3) $$f_1 \vee_a f_2 \equiv \frac{1}{1+a} \cdot \left(f_1 + f_2 + \sqrt{f_1^2 + f_2^2 - 2a \cdot f_1 \cdot f_2} \right)$$

(4) $$\neg f \equiv -f$$

In practice are used particular cases $a = 1$ (respectively $\min(f_1, f_2)$ for intersection and $\max(f_1, f_2)$ for union) and $a = 0$ respectively $f_1 + f_2 - \sqrt{f_1^2 + f_2^2}$ and $f_1 + f_2 + \sqrt{f_1^2 + f_2^2}$. There are also functions with C^m discontinuity, for example (5) is used instead (2).

(5) $$f_1 \wedge_a^m f_2 \equiv \frac{1}{1+a} \cdot \left(f_1 + f_2 - \sqrt{f_1^2 + f_2^2 - 2a \cdot f_1 \cdot f_2} \right) \cdot \left(f_1^2 + f_2^2 \right)^{\frac{m}{2}}$$

Difference between these functions is only discontinuity (not so important for visualization) and speed of calculation, of course.

The basic that R-functions give to computer graphics and F-Rep in particular, is possibility to compose practical arbitrary solids (functions) based on more simple already constructed functions or primitives as spheres, cylinders, cones and so on.

4. B-Rep Vs. F-Rep. Both representation schemes (F-Rep and B-Rep) have pros and cons. Of course, sometimes some cons may be comparatively or controversial (depending on the goals). Some of them are:

- B-Rep's pros: have hardware visualization, simplicity, well-known, etc;
- B-Rep's cons: not so open, only boundary of solid is described, more quality of solids want more information (more triangles, for example), etc;
- F-Rep's pros: open, extensible, nonlinear transforms, whole solid is described, volumetric textures/properties, etc;
- F-Rep's cons: there is not hardware visualization, require more complex calculations, etc.

Pros and cons of both may be used efficiently in so-called hybrid representation schemes (hybrid systems).

5. Hybrid systems. Depending on the goals, we use a given model. Its may be more or less exact/accurate in some of its properties. In practice, it is convenient to use more than one model for a solid (with different precision, for example). What will be used at given moment from some algorithm, depends on need of algorithm or on user settings. An example for this is computer games, where models with different level-of-details (LoD) are used for visualization of scene elements with different distance to player (for better speed). In F-Rep also is possible similar approach. First, model can be calculated with different accuracy of arithmetic (double, single, or integer precision). Second, model can be approximated with different function set, to calculations are fast or accurate.

Everything mentioned up to this moment concerns the case when we have different models in one representation scheme (F-Rep for example). Similarly, if it is needed, we convert a model into another representation scheme (depending on the goals). For example, we polygonize F-Rep solid and receive B-Rep solid/object (list of triangles, for example), next we use them (for visualization), and if this object is not more necessary then we destroy it. If memory is enough then we may save this model for following use (something like high-level cache). If later we need identical B-Rep object for the same F-Rep solid (with the same precision of approximation, etc) then we do not need to approximate (polygonize) existing object – we simply use cache. This approach gives to system more power and more possibilities.

6. System "Open F". For research purposes of F-Rep possibilities and relationship between F-Rep and other representation schemes, we develop an object-oriented system of software libraries and tools named OpenF. Its goals, architecture, characteristics, stages of development and possible application are described detailed in [4]. This library is developed as open source and freeware under CC license (Creative Commons).

In short, goals of OpenF are Geometric modelling (with more than one representation scheme), Visualization of models (with help of Ray-tracing and Polygonization), User interaction in distributed protected client-server environment, and Creation of wide-open system. Main characteristics are Openness (opportunity to be expanded in one or more directions), Hybridness (availability and possibility for co-working with more than one inner representation), Flexibility (easy adaptable in accordance with different applications), Distributiveness (simultaneous work of the system parts on different computer systems), and Multi-user (possibility for simultaneous work of many system users – independent or mutual work of more users). Goal of OpenF is to reach the

level of modern DBMS, but especially for geometric modelling. This includes transactional model, role-based security, distributed model, SQL-like query language, complex hierarchical models, etc.

The system architecture consists of three layers: core, expansion modules (plug-ins), and applications. Plug-ins can dynamically expand system with elements as sources, targets, storages, converters, communicators, and subsystems. Examples for sources are mouse, keyboard, files, functions (that produce information or events), as well as other logical input devices. Main task of sources is to produce information in the system. Similarly, storages are elements that save the information in system – models, intermediate results, caches, etc. They can be virtual/temporary (in memory) or permanent stored in file system backend or in database backend. Targets visualize information, converters – transform between representations, etc.

Each element of the system has levels of abstraction in the system: Logical level, Conceptual level and Physical level.

System creation should be by stages (system stage, application stage, and user stage). In addition, system must be independent of developers/teams, programming language, and operating system. Current main development language is C#. In the present development of OpenF is in system stage i.e. implementation and testing of core and some basic plug-ins.

Finally, the aimed system should be able to be used on the following purposes: Research, Application, and Education.

As example for similar existing system is HyperFun [5], but it has different goals, architecture, and it is only F-Rep centric. HyperFun have description language and visualization algorithms.

7. Conclusions. The advantages of F-Rep undoubtedly give occasion to consider that this scheme will be one of future fundamental schemes in computer graphics. When it is combined with existing widespread schemes, like B-Rep (for compatibility with existing B-Rep-based models), its applicability grows. Future research must be in the following directions: More detailed comparison of the quality and speed of F-Rep, Investigation of interaction between user and F-Rep-based systems, and Visualization of F-Rep models with the help of specialized hardware (Ray-tracing-based). In these directions system OpenF with its characteristics as openness and hybridness will help for further research and application of F-Rep in computer graphics.

REFERENCES

[1] Pasko A., Adzhiev V., Sourin A., Savchenko V., “Function Representation in Geometric Modeling: Concepts, Implementation and Applications”, The Visual Computer, vol. 11 no. 8, (1995), 429-446

[2] Schmittler J., Woop S., Wagner D., Slusallek P., “Real time Ray Tracing of Dynamic Scenes on an FPGA Chip”, ACM SIGGRAPH / Eurographics Conference on Graphics Hardware, (2004)
[3] Shapiro V. “Theory of R-functions and applications: a primer”, Cornell University, TR CPA88-3, (1988)
[4] Penev A., Dimov D., Kralchev D., “Open hybrid system for geometrical modeling”, 17th Int. conference SAER-2003, vol. 1, (2003), 131-135
[5] Adzhiev V., Cartwright R., Fausett E., Ossipov A., Pasko A., Savchenko V., “Hyper-Fun project: a framework for collaborative multidimensional FRep modeling”, Eurographics / ACM SIGGRAPH Workshop, (1999), 59-69
[6] Rvachev V. L. “Geometrical application of logic algebra”, Kiev, Technika, 1967, (in Russian)
[7] Rvachev V. L. “R-function theory and it applications”, Kiev, Naukova Dumka, 1982, (in Russian)

Alexander Penev
13 Lerin Str., ap. 18
4000 Plovdiv, Bulgaria
e-mail: apenev@pu.acad.bg

Dimcho Dimov
5 Poruchik V. Stefov Str.
4000 Plovdiv, Bulgaria
e-mail: dimcho_dimov@abv.bg

Dobromir Kralchev
3 Svoboda ave., ap. 15
4000 Plovdiv, Bulgaria
e-mail: dobromir_kralchev@abv.bg