Alexander N. Prokopenya
Miroslaw Jakubiak (Eds.)

Computer Algebra Systems in Teaching and Research

Volume V

Siedlce University of Natural Sciences and Humanities
SIEDLCE 2015

VOLUME EDITORS

Alexander N. Prokopenya (Warsaw University of Life Sciences – SGGW, Poland)

Miroslaw Jakubiak (University of Natural Sciences and Humanities in Siedlce, Poland)

Cover designed by Paweł Trojanowski

ISSN 2300-7397

ISBN 978-83-7051-779-3

Uniwersytet Przyrodniczo-Humanistyczny w Siedlcach

Instytut Matematyki i Fizyki

08-110 Siedlce, ul. 3-Maja 54,

tel. +48 25 643 1003, e-mail: imif@uph.edu.pl

Contents

I. Mathematical Modeling and Differential Equations

On Programming Interactive Graphic Applications in *Mathematica* System

Valery Taranchuk [1)], Viktoria Kulinkovich [2)]

[1,2)] Belarusian State University
4, Nezavisimosti Ave. 220030 Minsk, Belarus
taranchuk@bsu.by
kulinkovichva@gmail.com

Abstract. *The paper presents key code structure of software modules, functions and options of Mathematica language, that should be used when creating a freely distributed interactive graphic applications in CDF format.*

1 Introduction

Paper "On the Preparation and Distribution of Interactive Graphic Applications Using *Mathematica*" provides recommendations for creating, maintaining and freely distributing interactive electronic educational resources using Wolfram *Mathematica*, computable document format (CDF), modules from Wolfram Demonstrations Project collection. It lists software modules that can be used in educational process to increase course success. This paper presents and explains key code structure of software modules from recommended Wolfram Demonstrations projects [1]. It also describes functions and options of *Mathematica* language that should be used when creating interactive applications in CDF format [2].

2 Programming Interactive Applications

Interactive electronic resources are used during all stages of learning process in Computer Graphics course. Program module components listed and dissected below are used during the development of electronic interactive resources distributed to students studying on the topic of "Geometric transformations in 2D and 3D. Matrix representation, composition of 3D transformations" and "Linear Filters". In addition to documents with explanations and illustrations of the algorithms, we provide students with software modules Understanding3DRotation+.cdf and ConvolutionLinearFiltering+.cdf, adapted from original ones [3, 4] (with Russian terminology, changed initial camera angles and zoom, added comments).

Understanding3DRotation+ control panel, and annotation of demonstration scene elements are shown in figures 1-3; output frames in figures 2, 3 include fragments of control panel with geometric and view point variables set. Left side of figure 1 shows control panel elements, right side – demonstration scene.

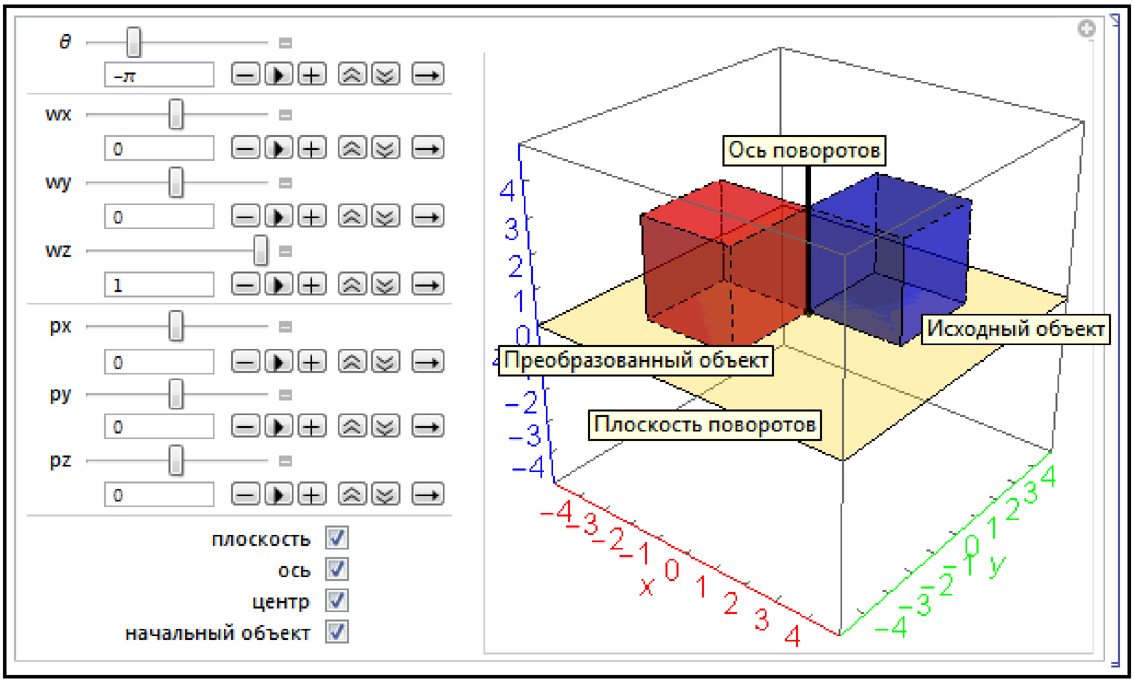

Figure 1: Control panel, resulting frame of the demonstration scene with labeled objects.

The demonstration scene includes: bounding box with labels, ticks, and tick labels (axes X, Y, Z and their labels are red, green, and blue accordingly), initial and transformed objects, plane and rotation axis. In the given example the size of the scene is $10 \times 10 \times 10$, origin of the coordinates is in the center of the scene, view point (in standard units specific to the scene size, that equals 10 for given example) – $\{1.4, -3.1, 1.3\}$, initial object – cube with size $3 \times 3 \times 3$. Figure 1 illustrates rotation around axis Z by 180°, wz = 1, all other variables are set to zero.

On figures 1-3 modifying wx, wy, wz variables will result in rotation, modifying px, py, pz – in translation along respective axis, modifying θ – specifies final rotation angle of the initial object about defined axis (both translations and rotations are taken into account).

Figure 2 illustrates translation transformations, two fragments in the upper row – translation along the X axis with px = -0.5 (on the left) and -1.5 (on the right), in the lower row – in addition to px = -1.5 along the Y axis two variations py = -0.5 and -1.5; in the series of those examples wz = 1, $\theta = 180°$, all other variables are set to zero.

It is important to note, that clipping algorithms are executed automatically during image rendering, thus for px = -1.5 py = 0 only right side of the cube is displayed, for px = -1.5 py = -1.5 – both left and front parts are clipped.

Software module allows moving and rotating the resulting scene, changing the frame size. Every action (rotation, movement) can be performed by setting the value in the input field or by moving the slider. User can also render a video that will update the view by automatically changing any of the above parameters; video output controls allow to adjust the speed and direction of playback, including the option of playing video frame by frame. Increment steps for the parameter can be defined in code, otherwise default value will be used; range boundaries are required. Module contains comments for all core functions and options used in the code, so that students can apply their changes, reuse code style and structure, exercise using other graphical primitives and shapes. For example, comments that explain how to use Graphics3D function to build and output graphics, as an exercise suggest swapping Cuboid primitive used as a source object with Cylinder, Sphere, Cone, etc.

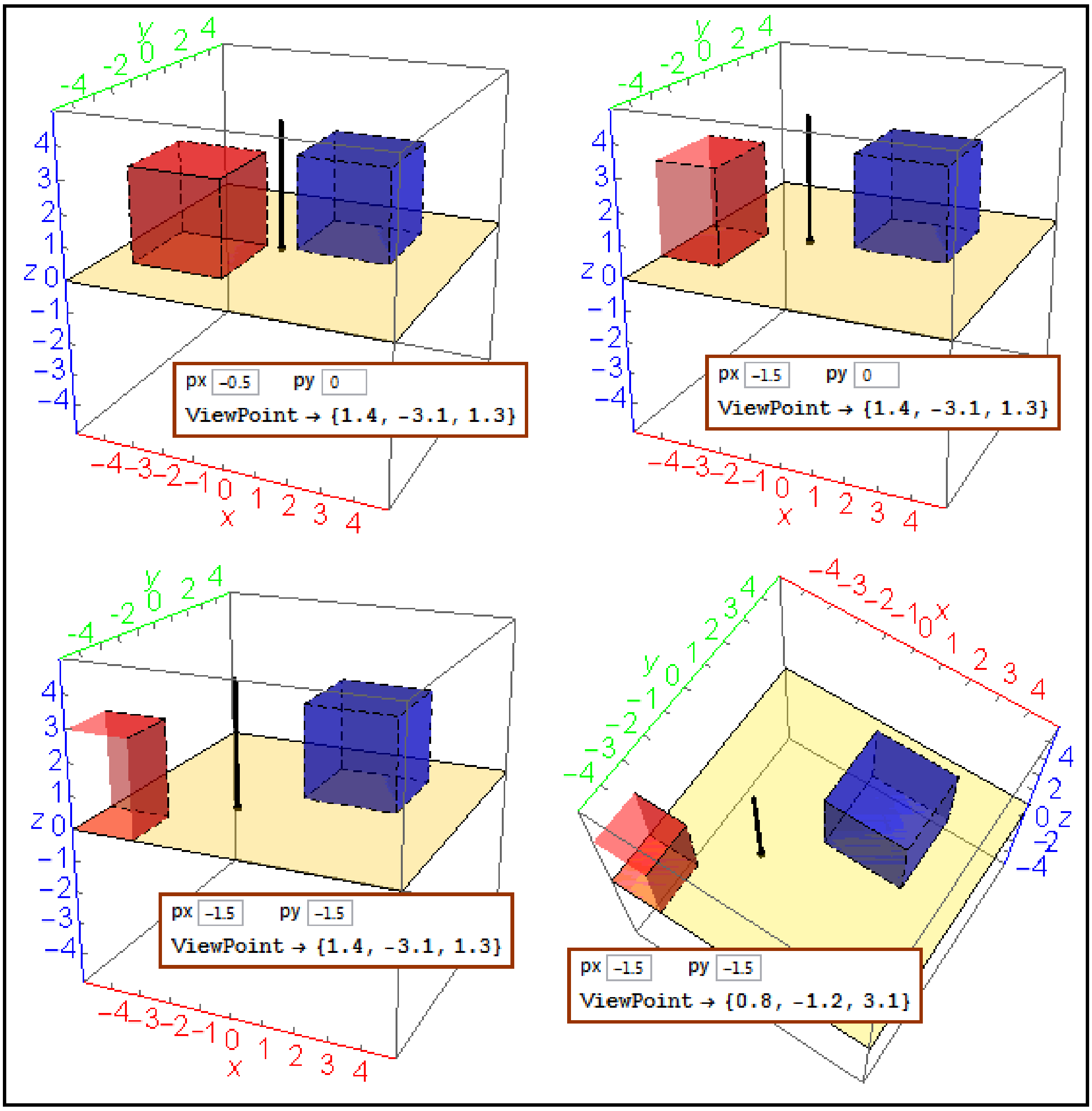

Figure 2: Fragments of application windows with the results of applying translation transformations.

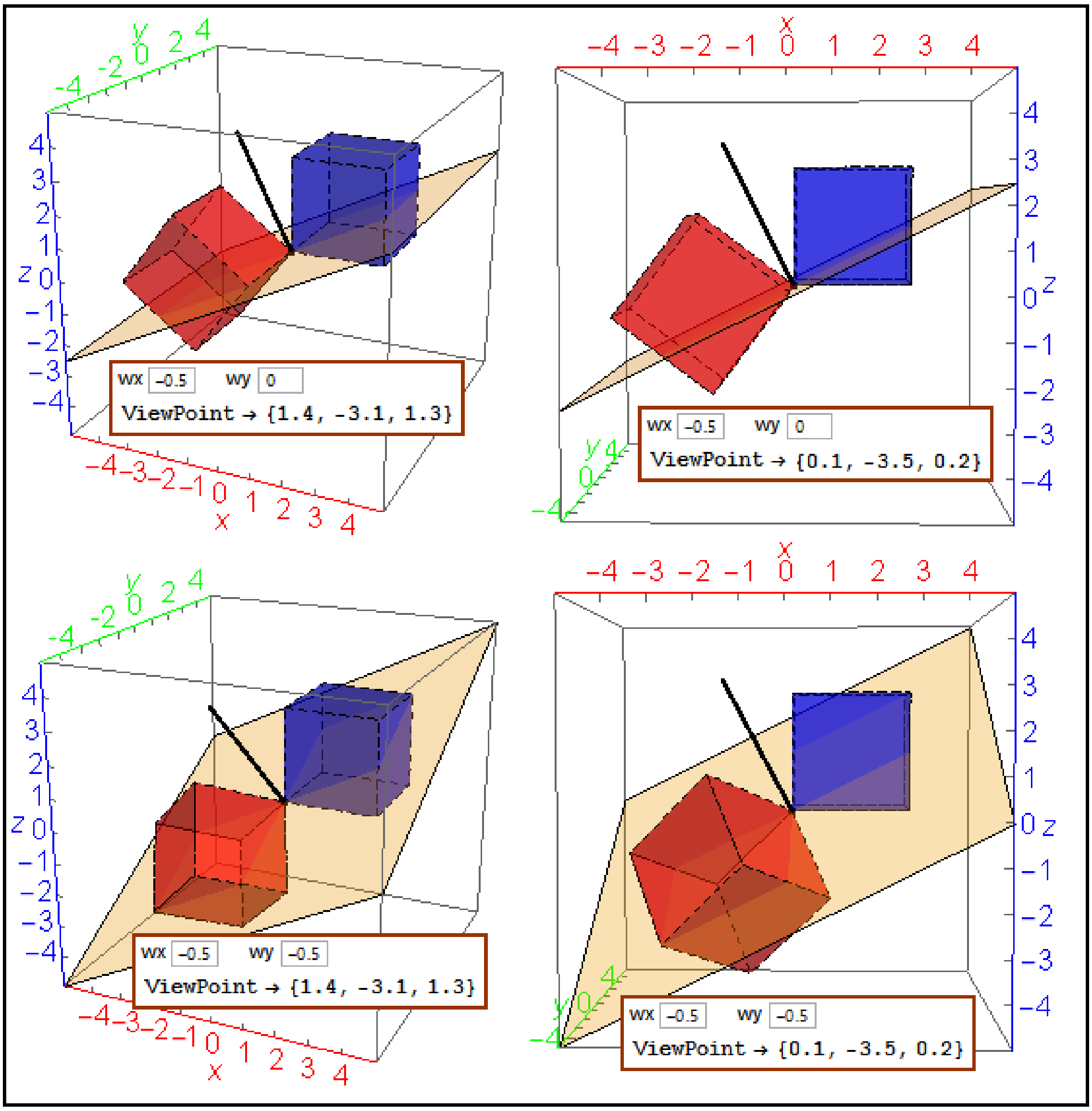

Figure 3: Fragments of application windows with the results of applying rotation transformations.

Functions and options of the source code. Exercises include explanations of following view configurations:

- text format (FormatType, BaseStyle, Style, TraditionalForm, StandardForm, FontFamily, FontSlant, FontSize, AxesStyle, LabelStyle, TicksStyle, GridLinesStyle, MeshStyle, BoundaryStyle, FillingStyle, ClippingStyle);
- line thickness and type (Thickness, AbsoluteThickness, Dashed, Dotted, DotDashed, Thick, Thin);
- color modes, transparency and lightning (Colors, GrayLevel, RGBColor, CMYKColor, LABColor, ColorFunction , ColorFunctionScaling, Opacity, Lighting,

Specularity);

- options to define output view (ImageSize, PlotRegion, PlotRange, PlotRange-Clipping, AspectRatio, BoxRatios, Scaled, BoxStyle).

The module uses RotationTransform, TranslationTransform transformation functions. Additional explanations and exercises are provided for AffineTransform, GeometricTransformation functions.

Figures 4-6 show control panel and output view of **ConvolutionLinearFiltering+ module**. Control panel includes the following elements: sliders (to set the matrix filter); buttons (to select one of the standard filters); output view includes two images: original image (on the left) and processed image (on the right). The user can set an arbitrary matrix as a filter or select one of the standard filters (including filter with randomly generated values).

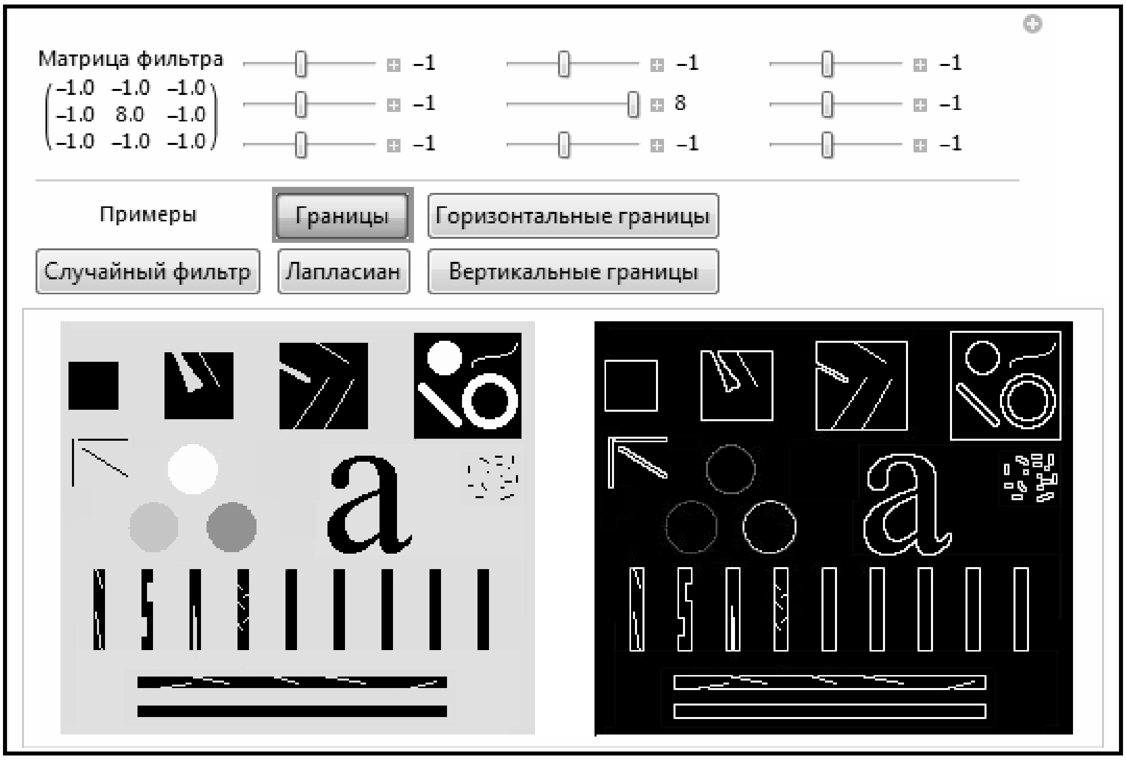

Figure 4: Control panel and output view of the module used for edge detection filter demonstration.

Figures 4 and 5 illustrate the processing with standard linear filters; matrix is defined in the source code. Figure 6 demonstrates processing with linear filter that is set by the user. The matrix filter can be entered manually or by moving the slider. The resulting matrix is shown in the upper left corner.

Following functions and options are used in this module (in the addition to the ones mentioned above):

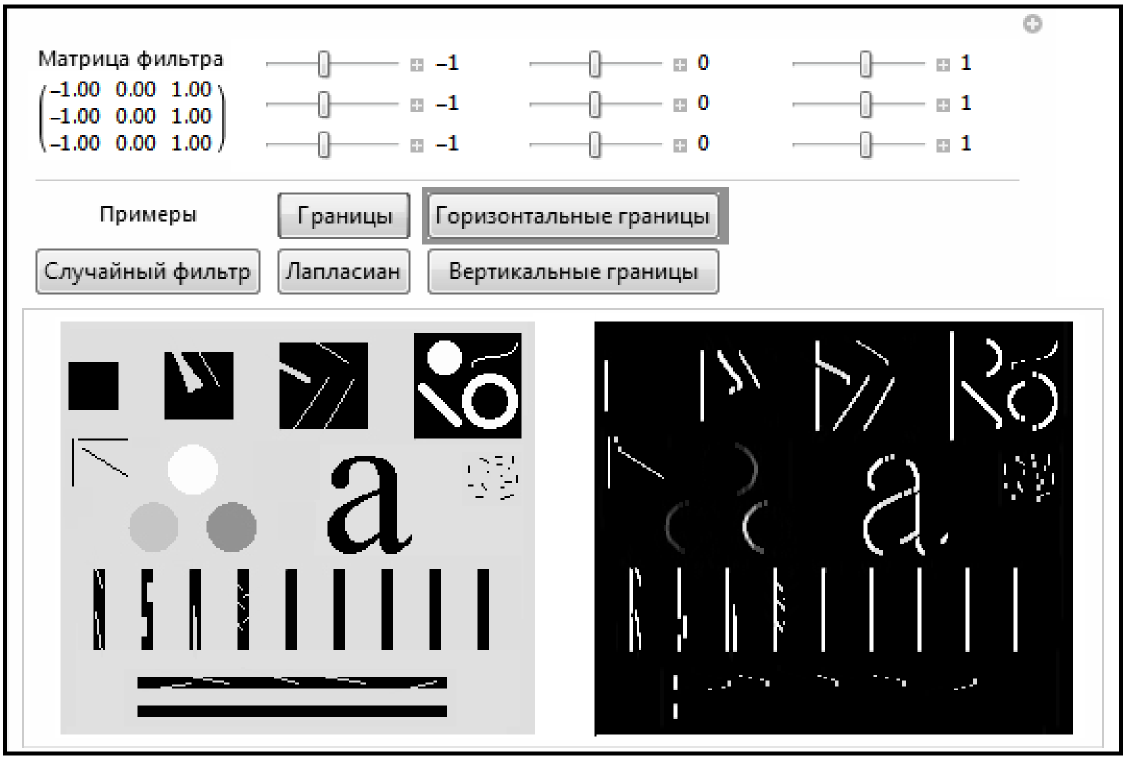

Figure 5: Control panel and output view of the module used for horizontal edge detection filter demonstration.

- functions for image processing (ImageConvolve, ImageAssemble);
- function for displaying elements in graphics grid(Grid, Row, Column);
- functions and options for working with real numbers (NumberForm, RandomReal, WorkingPrecision).

Configuring dynamic interactivity tools. Manipulate function is often used to support dynamic interactivity in *Mathematica* notebooks, dialog windows, controls to input parameters used in calculations (including symbolic computation) and defining and viewing plots. Manipulate module allows to create a variety of interactive tools for a given expression *expr* using provided arguments. Expression *expr* is interpreted in the most generic form, so it can be a list of elements that can include text, mathematical expressions, graphic functions, etc. Special emphasis is made on programming dynamic output and interactive tools; examples illustrate functions and options of dynamic computations, toggling indicators on and off, using flags, buttons, menus. Following components are explained: Manipulate, Dynamic, Initialization, Delimiter, PopupMenu, Checkbox, CheckboxBar, RadioButtonBar, SetterBar, TogglerBar, ControlType, Locator, Slider, Slider2D, ColorSlider, SaveDefinitions, AutorunSequencing.

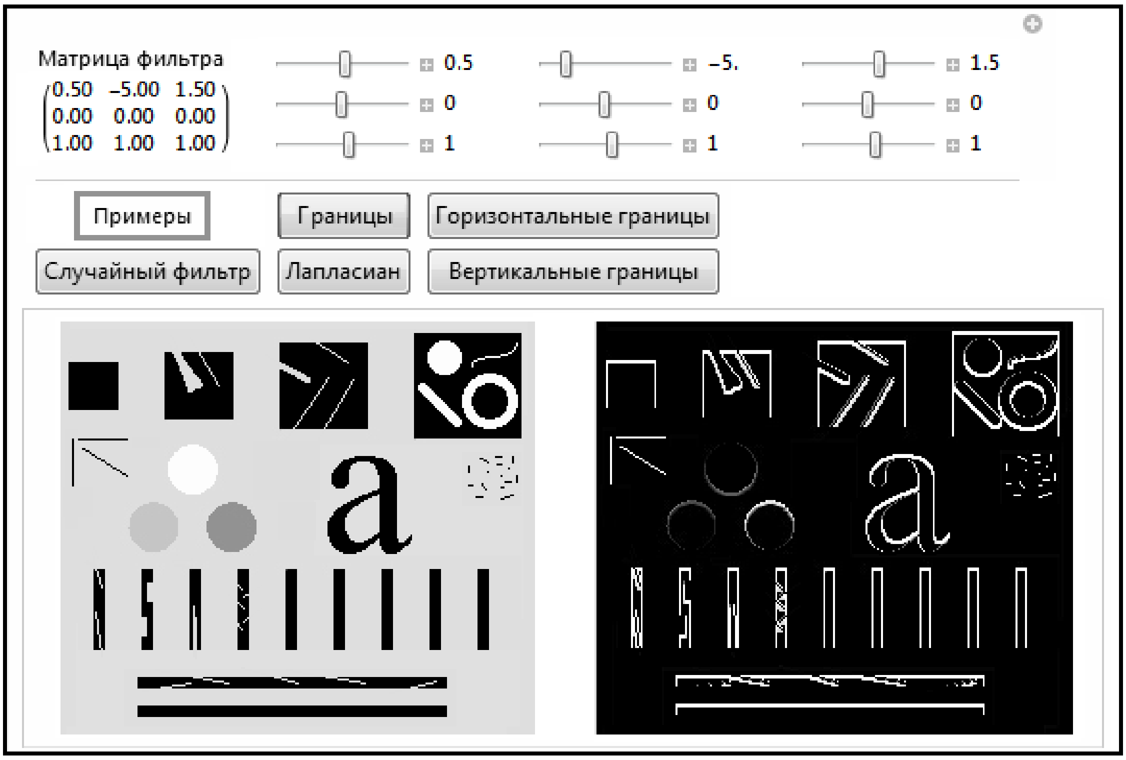

Figure 6: Control panel and output view of the module used for random filter demonstration.

Options for implementing dynamic interactivity in CDF documents. *Mathematica* notebooks that are intended to be saved and later viewed in CDF Player, should include initialization (Initialization : − >), or instructions to save animation settings (SaveDefinitions). Those steps are required, because CDF Player cannot load user data during execution. Only lists, bundles and sets that are part of the knowledge base are accessible. That means that all the data that Manipulate function uses should be integrated in the interactive elements of CDF file.

3 Conclusion

Recommended software modules from freely distributed Wolfram Demonstrations Project collection, reviewed functions and options of *Mathematica* language that enhance possibilities for creating and using interactive educational resources containing mathematical notation of any complexity and graphic illustrations of all types and categories. Listed solutions for distributing live interactive content, obtaining hard copies in any of the conventional styles. It is important to note that creating such interactive resources does not require programming experience. Additional information is available on BSU Department of Computer Applications and Systems web site www.cas.fpmi.bsu.by, CDF documents are available for review.

References

[1] *Wolfram Demonstrations Project*. [Electronic resource] / Access mode: http://demonstrations.wolfram.com/ - Access date: 24.06.2015

[2] *Computable Document Format CDF*. [Electronic resource] / Access mode: http://www.wolfram.com/cdf/ - Access date: 24.06.2015

[3] *Understanding 3D Rotation*. [Electronic resource] / Access mode: http://demonstrations.wolfram.com/Understanding3DRotation/ - Access date: 24.06.2015

[4] *Image Kernels and Convolution (Linear Filtering)*. [Electronic resource] / Access mode: http://demonstrations.wolfram.com/ImageKernelsAndConvolutionLinearFiltering/ - Access date: 24.06.2015