Chapter 8
Visualization by Coding: Drawing Simple Shapes and Forms in Various Programming Languages

Anna Ursyn
University of Northern Colorado, USA

Mehrgan Mostowfi
University of Northern Colorado, USA

ABSTRACT

The authors present image transformations that allow for checking and better understanding the graphical capacities of various languages and the place of programming in artistic production. The goal of this tutorial-style chapter is to introduce curious artists to basic programming concepts such as variables, arrays, loops, condition structures, classes, and functions through coding visualization of a simple 2-dimensional shape, a horse rider. At the same time, the goal is to make computer scientists more comfortable with the visual ways of dealing with concepts and objects to be programmed. How more complicated digital art can be produced through transformation using programming is also briefly discussed. Pointers to more advanced artistic techniques are also given. These techniques use transformations of code-driven lines to turn them into sculptures, photo silkscreens, photolithographs, knitted fractals, etc.

INTRODUCTION

The gap between art and programming seems to be a pressing issue. In particular, industries where the combination of these capabilities is beneficial, such as the game development industry, are in need of a new role called a "technical artist." We believe that exposing artists to the fundamentals of programming through a familiar concept (an artistic shape) can be a first step to bridge this code/art divide. In this chapter, we examine ways to use various programming languages in order to set up an environment for a programmer and explain routines and available shortcuts. By using the same image outcome, we intend to show options beyond the graphical capacities of various languages. The perspective taken in this chapter is based on an artist's experience, where coding is a part of artistic creation. It is aimed to explore how art and programming might complement one

DOI: 10.4018/978-1-4666-8142-2.ch008

another and how we can enhance the aesthetic quality of electronic image production through putting these links into action. The image of a horse rider was drawn and then programmed by one of the authors (Ursyn) in Fortran, now an obsolete language. This image resulted from recording points on a grid paper after a pencil drawing, which was thus converted to a setup of points as x, y, z, or –z. It was written almost 30 years ago, and presently it was translated into several modern languages: Python, C++, and Java by the second author. Further discussion is about the role of programming for data presentation in artistic input (Ursyn, 2013) and possible approaches to creating and designing that relate to working with programs.

There is a growing need and interest for understanding code and its role, which stems from the current programming options for choosing languages, libraries, solutions, shortcuts, open sources, apps, and conditions, as well as the transparency of code in visual editors (for example, Dreamweaver, HTML, and CSS). Computing languages are embedded in 3D editors (for example Maya, Blender, and Python). The idea of translation has deeper roots than we often are willing to acknowledge. It is grounded in symbolic systems, both related to languages and to abstract messages. The history of communication can be seen connected to the Rosetta stone (a stele – a stone slab containing three scripts of the same decree: ancient Egyptian hieroglyphs, Demotic script, and Ancient Greek script) and to various efforts to transfer meaningful tales such as Beowulf, the Old English epic poem consisting of thousands of alliterative and quite complicated lines. Perhaps the current STEM to STEAM (Science, Technology, Engineering, and Math to Science, Technology, Engineering, Art, and Math) movement has hopes related to this concept.

The visual aspect in our technology-based life is based on programming. The open source becomes a token of our daily availability. The 3D printing technology is one if the fastest developing and sought after technologies. For example, a 3D printing Startup Company Lulzbot (https://www.lulzbot.com/) in Loveland, Colorado offers codes and wireframes of machine's physical parts to anyone interested, and invites people to modify whatever they'd like according to their needs. However, there is still a gap between disciplines. The concepts of visual computing or aesthetic computing are unknown to many. There is a shortage of programming teachers, and not enough courses offered to young learners.

The place for programming in data presentation for artistic production is a recurring theme both in computing and arts criticism. However, while working around merging programming with graphics, we can see people not connecting the two. The most brilliant programmers often consider coding not related to visuals and see no need for it. On the other hand, the most brilliant visual people shy away from anything that would be based on code and give it a blank look. There is a visible merger though, and there is a need for it. Translations into various languages aim at setting up an environment for a coder and explaining routines and available shortcuts. By using the same image outcome, we intend to show options beyond graphical capacities of various languages. The power of this chapter's approach is beyond the fact that the same image can be seen translated from one language to another. A person willing to think about the place of programming in artistic production can see the visual differences and comprehend this issue. One can see the big picture by applying the power of the invaluable tool of comparison and contrast. Also, those willing to experiment should be able to actually compile the program in more than one language and see particular outcomes. Undoubtedly, the complexity of this task and the prerequisites it requires could be overwhelming to many. Nevertheless, this chapter gives some important outlooks on this topic and possible approaches to creating and designing while working with programs; the interested users could hopefully see the idea holistically and will