β-TEK Software

Final Report For:

Virtual Loom

Submitted By:

Joshua O. Lee
November 24, 2012

β-TEK Software

Joshua O. Lee
B – TEK Software
125 Nowhere Road
Somewhere, MT 55555
(406)111-2222

Spinners and Weavers Guild of Montana
123 Any Street
AnyTown, MT 12345

November 24, 2012

To Spinners and Weavers Guild of Montana:

Enclosed is the final report in response to your request for a weaving software that your organization is wanting to sell to support your group. I hope you will find that this software, currently under the working name of Virtual Loom, not only meets all of your desired specifications, but exceeds your expectations.

The final report for the Virtual Loom includes the following:

- Introduction
- Review of Specifications
- Design Approach
- Debugging Results
- Conclusion

It has been a pleasure working with you and your group. If you have any questions, concerns, or would like to discuss this report in further detail, please do not hesitate to contact me. I look forward to working with you and your group again in the future.

Sincerely,

Joshua O. Lee

β-TEK Software

Table Of Contents

β-TEK Software

Introduction

The main purpose of this project was to provide an intuitive, easy to use weaving software, similar to more complex software like DB-Weave, for sale to the public. Virtual Loom is designed to allow a user to allow users to quickly and simply create weaving patterns, and preview what it would look like if it was woven without having to learn some sort of pseudo programing language. The Virtual Loom uses a simple and easy to understand user interface that any user, with basic understanding of how to operate a computer, can easily learn and use.

Review of Specifications

We at B-Tek feel the need to provide you and your group with high quality software that not only fully meets your needs and specifications but also exceeds them. Outlined below is the specifications that this project must meet.

Virtual Loom must:

- Have an intuitive and easy to use interface
- Must be quick and easy to learn
- Must be able to work without having to use or learn a pseudo programing language

β-TEK Software

Design Approach

Basic Overview of Design:

The Virtual Loom can be divided into three major components: Pattern Creation and Manipulation, Weaving, and File IO. Each of these components will be explained in further detail in individual sections below. The Virtual Loom consists of two separate forms. First, a main form (**figure 1-1**) for creating, manipulating, saving, opening, and weaving the pattern the user creates. Second, a dialog form (**figure 2**) used by the user to specify the patterns dimensions and the starting color of the wefts and warps when creating a pattern. When the user wishes to create a pattern, all the user needs to do is press the "new pattern" button or select new from the drop down menu. This action opens up the pattern setup dialog where the user will set the patterns dimensions and default colors for the warp and weft threads. When all the parameters are set and the user presses the "create" button; a new pattern class will be created with these parameters and displayed in the pattern window (**figure 1-2**) of the main form. This pattern is fully interactive and allows the user to change the heddle position for that pass just by clicking where the warps and wefts intersect (**figure 1-3**). If the user wishes to change the color of one of the warp or weft threads in the pattern they may select the thread number, type and it new color using the controls in the color group box (**figure 1-4**). Once the user likes the pattern they created all they need to do is click the "weave pattern" button. This button will simulate having the pattern woven onto fabric with the dimensions of 3in by 6in with a thread diameter of 1/16in. While this size may seem small it gives the best resolution for viewing the pattern that was created. The simulated "fabric" will then be displayed in the fabric window on the fabric tab (**figure 1-5**). If the user desires they may go back to the pattern window by clicking the pattern tab and then make alterations to the existing pattern then reweave it if desired. After the user is fully satisfied they may save their pattern as a Virtual Loom Pattern (.vlp) file or as a .jpg image to share, or they may print out a hard copy of their pattern. The woven "fabric" may also be saved as an image or printed if desired.

β-TEK Software

Pattern Creation and Manipulation:

While all three major parts of the Virtual Loom, Pattern Creation and Manipulation, Weaving, and File IO, are handled almost exclusively by the PatternClass, pattern creation and manipulation is the PatternClass's main role and makes up about 80% of its code. Pattern creation and manipulation also makes use of two different data structures, thread and intersection, to store pattern data, and an enum call thread type which is used to identify if a thread is a warp or weft thread. Creation of a new pattern is handled by one of the three constructors the PatternClass uses. Once a new pattern class is initiated the constructors calls private subroutines to calculate the dimensions for the warps and wefts. This information is stored into one of two thread arrays depending if the thread is a warp or weft. After all the warp and weft dimensions are calculated their intersections are calculated and stored into an intersection array. These intersections (**figure 3**) act as “hit boxes” for the pattern and is used to determine what threads are to be manipulated in the pattern. Once the new instance of the pattern class is initialized the drawPattern() function is called. This function returns a bitmap image representing the pattern to be drawn in the pattern window. When a user clicks on the pattern window the update() function is called with the mouse's current position is sent as arguments. Update() then determines using the intersections array what “hit box” the mouse was in when it was clicked. Update() then makes the necessary changes to the intersections, warps, and wefts arrays, redraws the image with the new parameters, then returns the new bitmap to be displayed. To change a thread's color changeThreadColor() is called with the threads type, number (or index), and new color passed as arguments. ChangeThreadColor() then replaces the color in the color property of the thread stored in the wefts or warps array with the new color. The pattern is then redrawn and the updated bitmap is returned to once again be displayed.

Weaving:

Weaving is handled by the weave() function in the PatternClass. When this function is called the fabrics window's dimensions in pixels, the fabrics dimensions in inches, and the thread diameter are passed as arguments. From this information “sections” are calculated. These “sections” are the width and height of the weft and warp threads and would be where these threads intersect. Before a section is drawn it checks the intersection array for the heddle's position. If the position is up it gets the color of the warp thread of that section and sets that sections color to the warp's color. If the position is down it gets the color of the weft thread of that section and sets that sections color to the weft's color. This process is repeated till the fabric is fully drawn (**figure 1-5**) based on the specified dimensions. A bitmap is then returned for display in the fabric window.

β-TEK Software

File IO:

File IO for .vlp files are handled by two functions and a constructor within the PatternClass. The save() function saves the current pattern to a specified .vlp file. The save() function's only argument that can be passed to it is the IO.Stream connected to the specified .vlp file, and all information is saved in a binary format with the following structure:

Size of the wefts array,
Size of the warps array.
Size of the intersections array,

The properties for each element in the wefts array in the following format:
X_Start,
X_End,
Y_Start,
Y_End,
Color as 32bit ARGB

The properties for each element in the warps array in the following format:
X_Start,
X_End,
Y_Start,
Y_End,
Color as 32bit ARGB

The properties for each element in the intersections array in the following format:
X_Start,
X_End,
Y_Start,
Y_End,
Weft Thread,
Warp Thread,
Heddle Position

When save() returns, a Boolean value is returned. True is for success and false is for failure of saving the file.

Opening a file is only achieved through a constructor meaning a new instance of PatternClass will have to be created to load a file. Unlike the other two constructors this constructor does not take the pattern's dimensions as arguments. Instead it only takes the IO.Stream connected to the specified .vlp file and the dimensions of the pattern window in pixels. When called this constructor creates a new bitmap based the pattern window's dimensions then it calls the private function LoadPattern(). This function then attempts to load the .vlp pattern then returns true or false depending on its success or failure. Since constructors can not return values to check if the pattern was successfully loaded the ErrorCheck property is called after initialization of the new pattern. If false is returned then no error occurred but if true is returned then an error occurred while loading the file. If true is returned detailed information on the error can be found in the InternalError property.

Debugging Results

Based on current tests the virtual loom is running with out any major program breaking bugs. Even with our best efforts though, there is one minor bugs left in the system. During testing it has been noticed that when some patterns are woven they will become distorted. This only seems to occur with patterns larger than 4x4 threads and is only really noticeable with complex patterns that are not the same in length and width. We currently believe this is caused by errors in the algorithm that converts the dimensions from inches to pixels. These algorithms have caused us much trouble through the course of this project and is the reason the constructor that accepts the dimensions of the pattern in inches is not and should not be used at the moment. The only other problem present in the program is the user interface not responding quickly to changes in the pattern. This is because whenever a change is made to the pattern, the entire bitmap is redrawn pixel by pixel. This is considered less of a bug and more of a consequence of design. We a currently working hard to rectify these problems and will provide you and your group with updated software as soon as possible. If any one using this software finds any new bugs please contact us imminently so we may begin fixing these problems.

Conclusion

As can be seen the Virtual Loom meets all of your groups specifications. It uses an intuitive and easy to understand user interface that any one, with basic computer operation knowledge, will be able to learn quickly. It also allows user to manipulate the pattern directly using graphical objects eliminating a need to learn some sort of pseudo programing language to use it. I would like to thank the Spinners and Weavers Guild of Montana for allowing us to work on such an interesting project for them. I look forward to conducting further business with the Spinners and Weavers Guild of Montana in the future.

Appendix

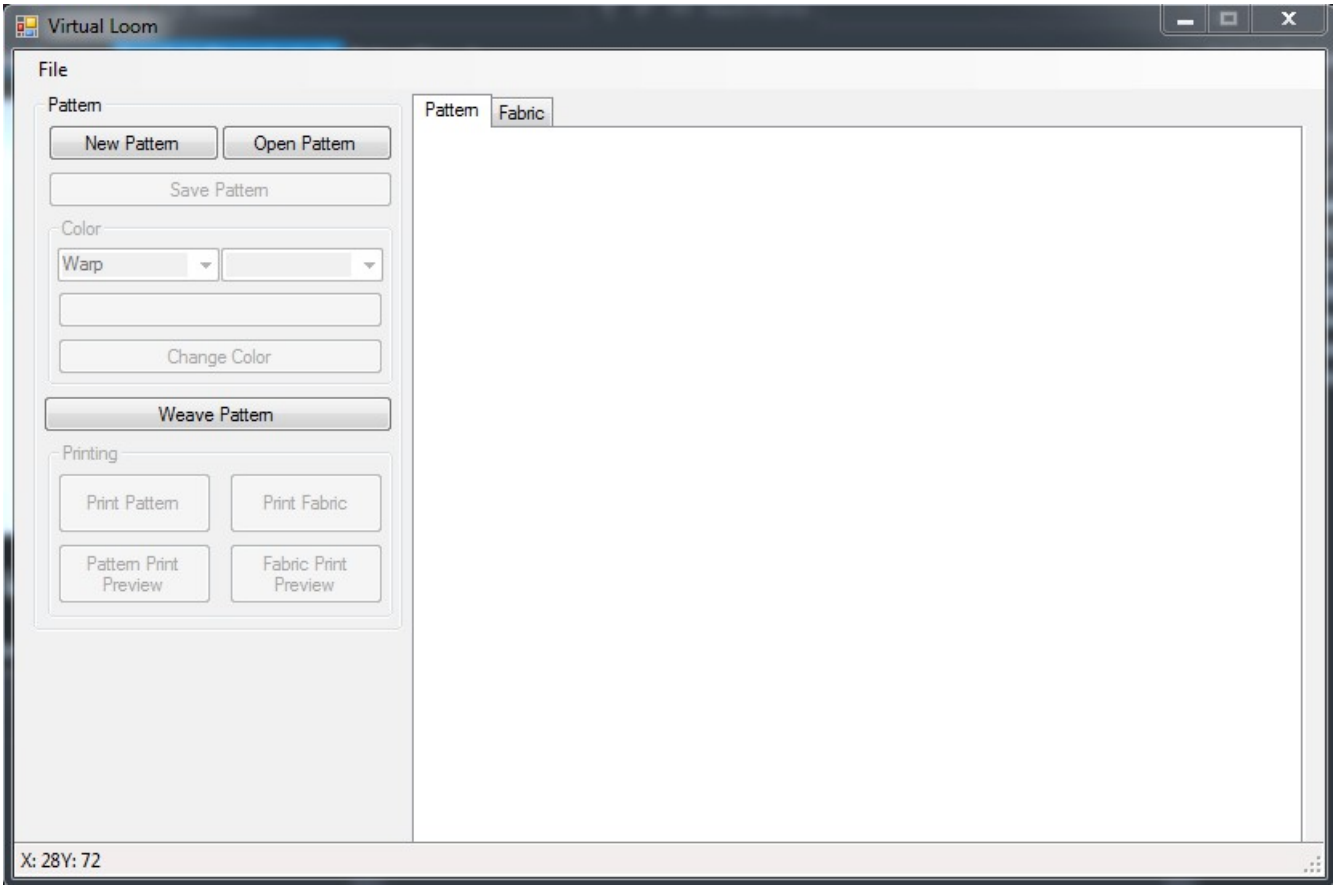

Figure 1-1: The main form

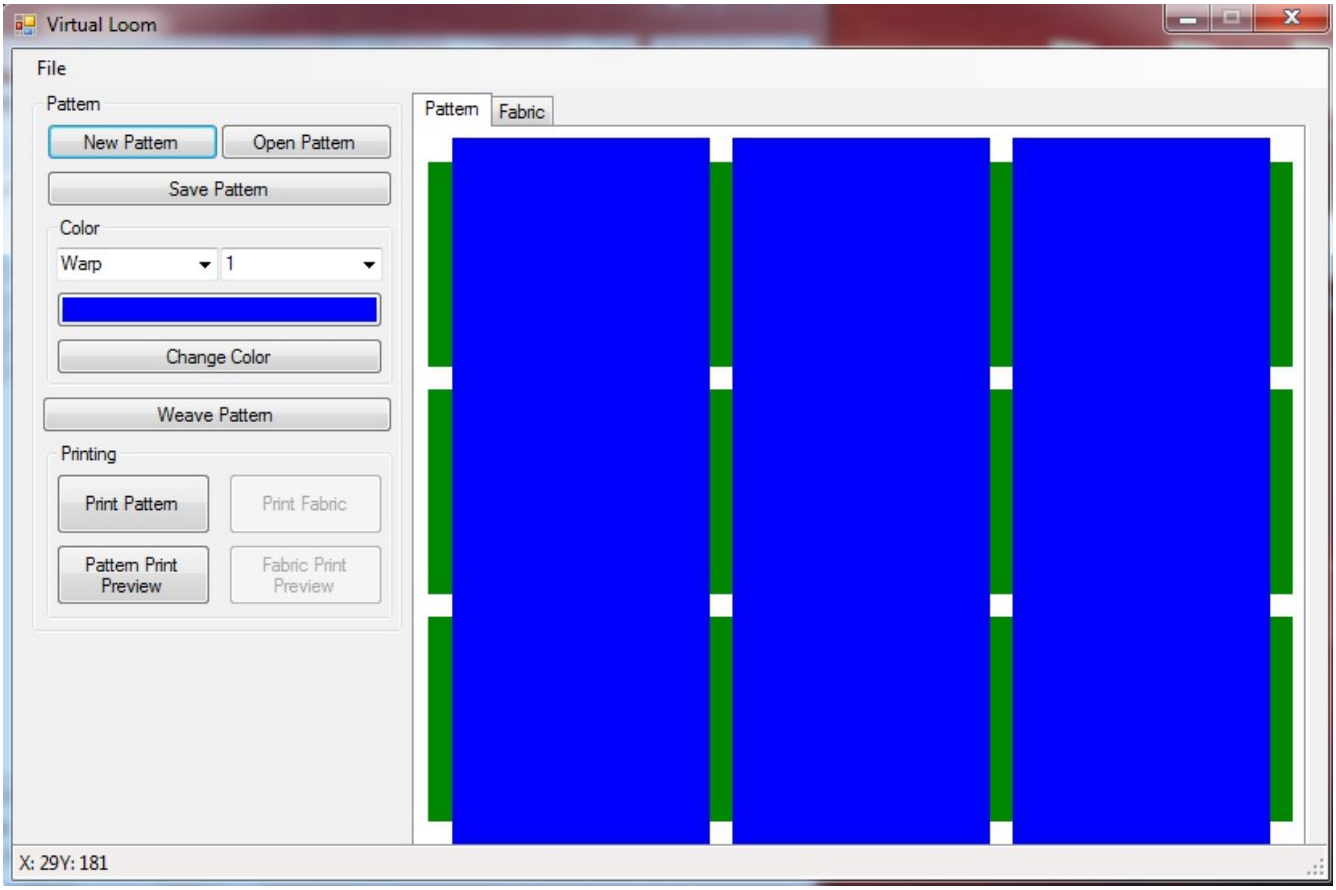

Figure 1-2: The main form with the default pattern created

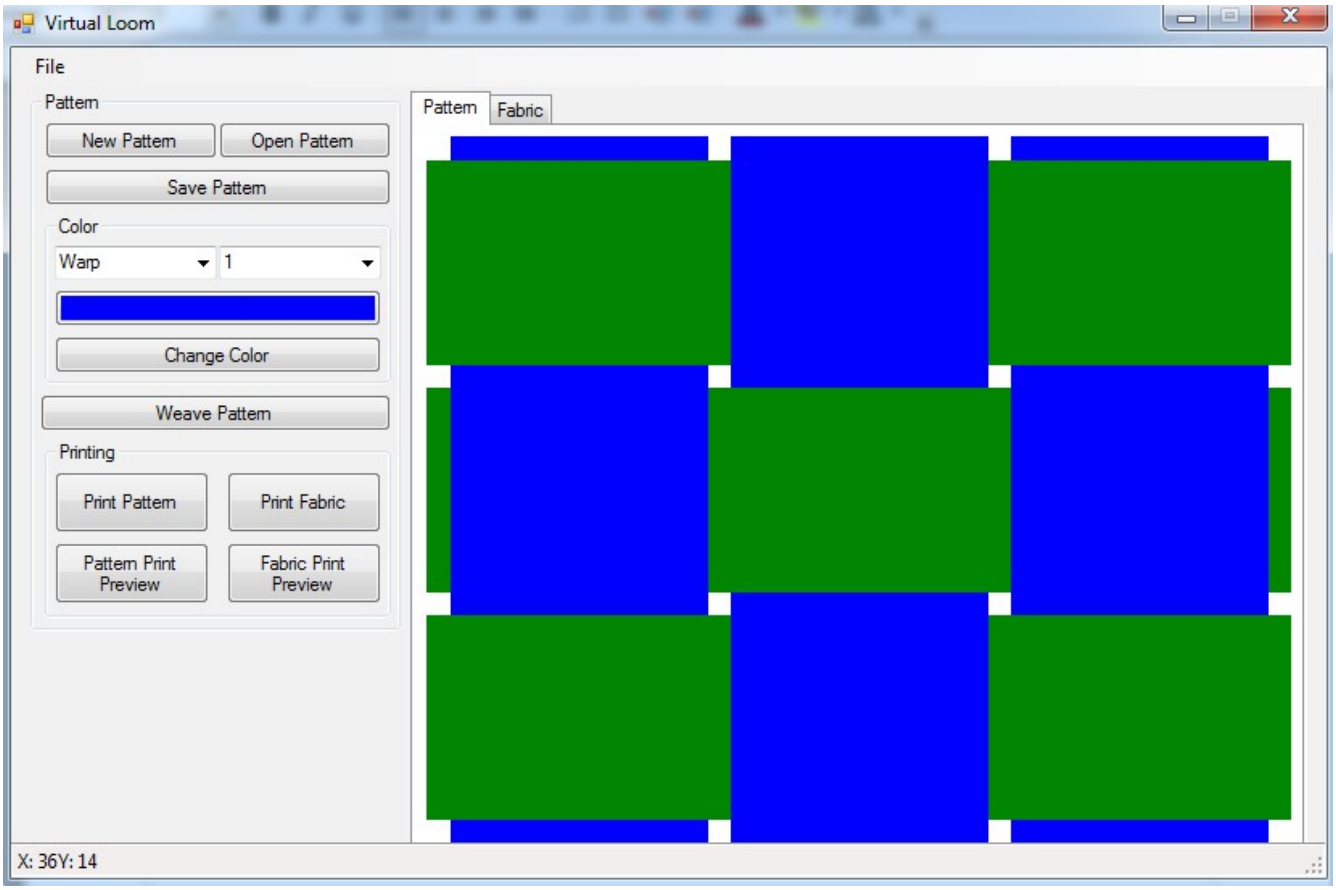

Figure 1-3: A pattern created by a user from the default pattern

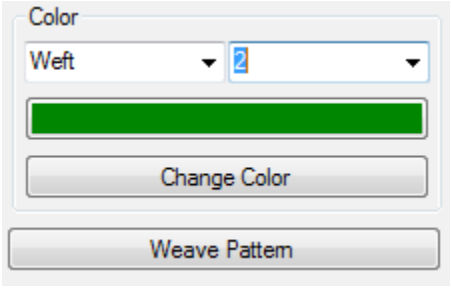

Figure 1-4: Color group box with controls

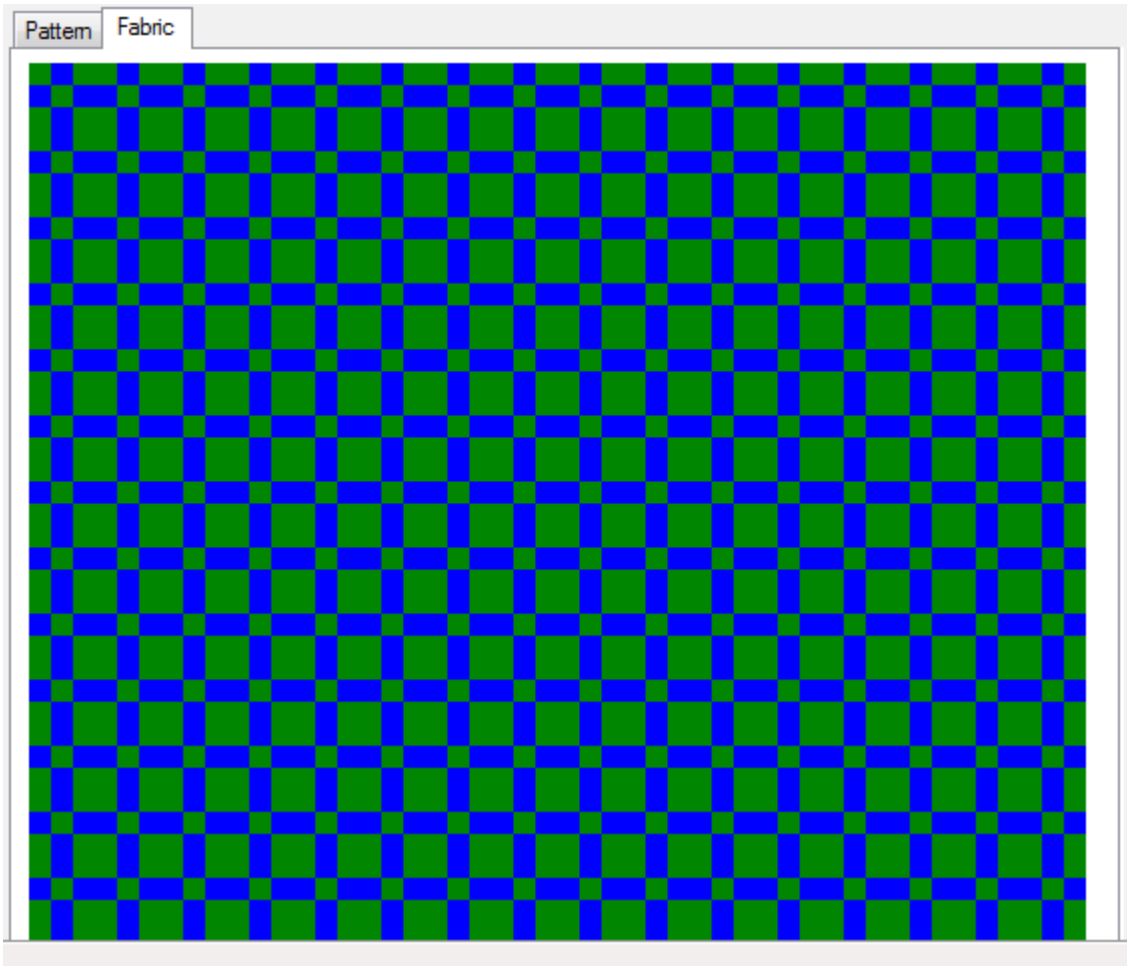

Figure 1-5: A woven pattern displayed in the fabric window

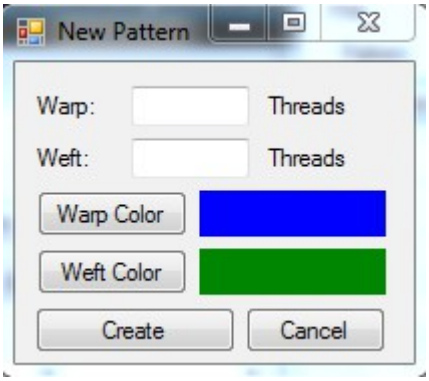

Figure 2: The pattern creation dialog box

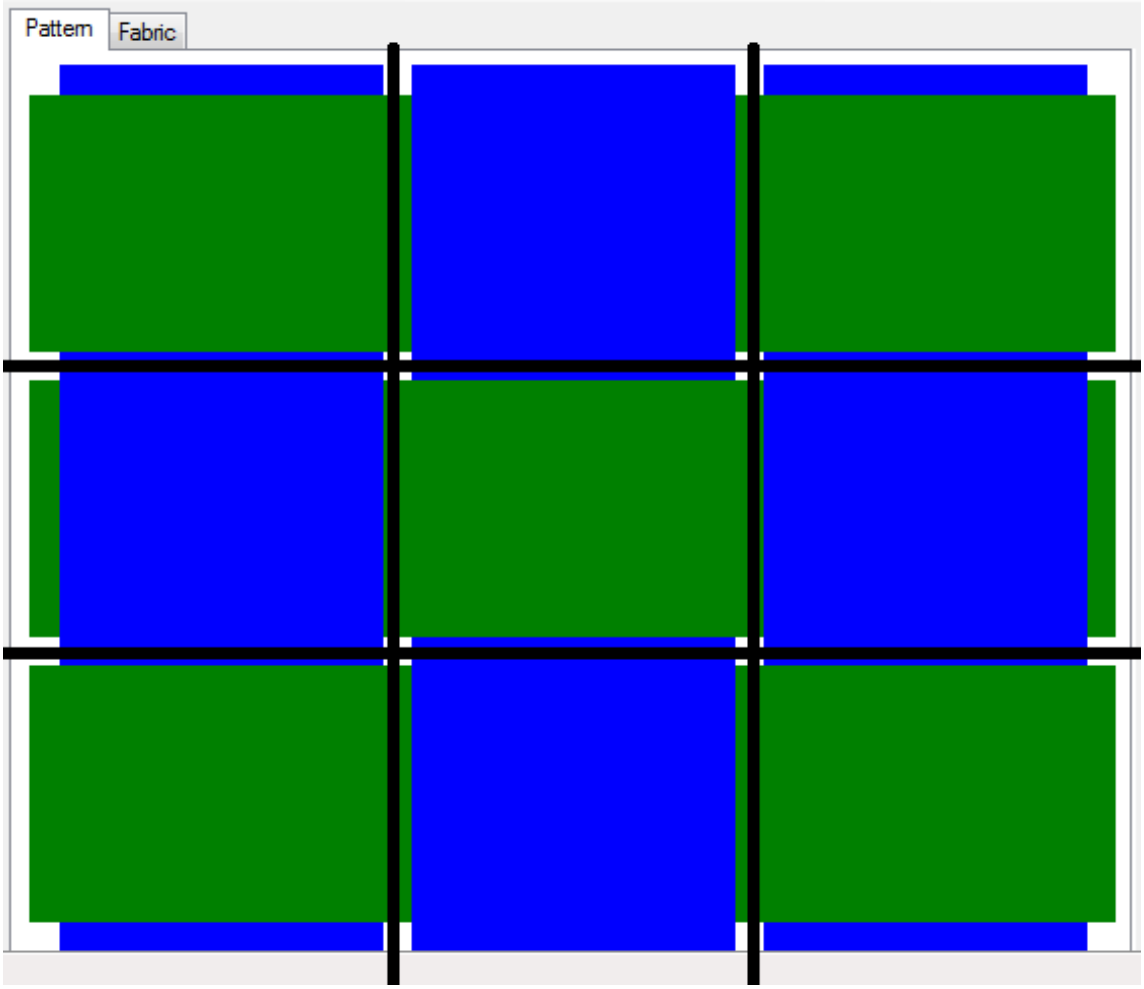

Figure 3: A pattern with the “hit boxes” displayed