EECS 3461: Assignment 3

XiaoTong Wu (215072309),

Matthew Longphee (214976096),

Ibrahim Suedan (215140569)

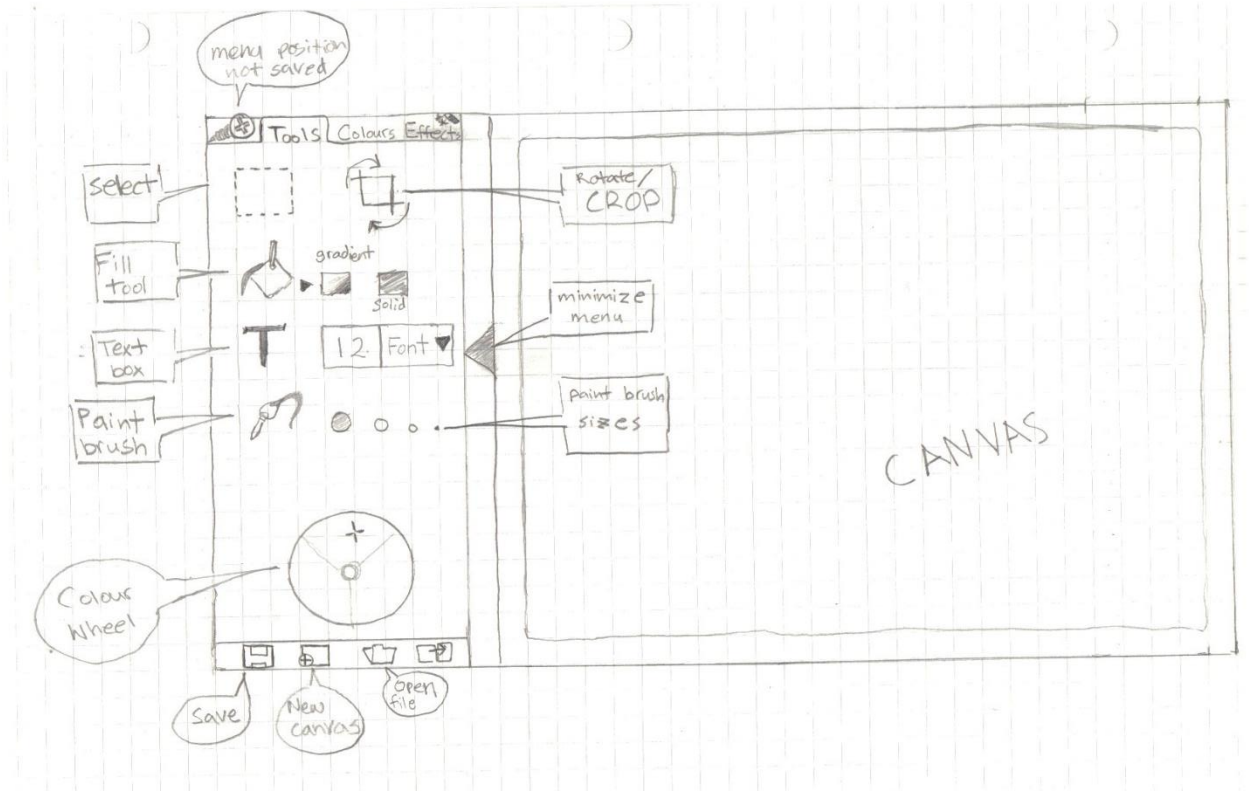
Table of Contents

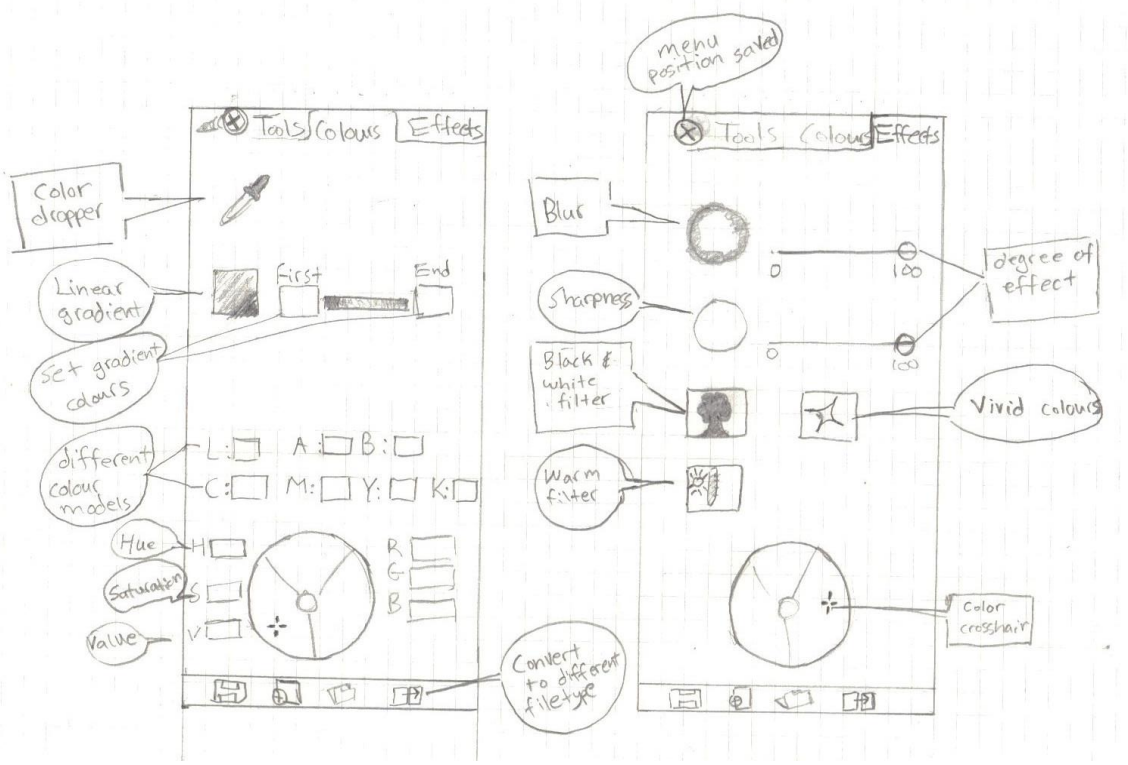
Initial Design

The initial design was done as a paper prototype for the sake of simplicity and time. It's much easier to draw out ideas on paper than it is to find clipart approximations on the internet. It's much easier to make adjustments as well.

The initial design was inspired by mobile applications' simplistic and sleek User Interfaces. The side menu especially was designed to come out of the side and be able to be minimized as can be found in many mobile applications. In those applications' cases, it is designed that way because phone screens need to save a lot of space.

The color wheel was designed to always be in the bottom corner of the screen, atop all menus, so that it could be quickly accessed when using the canvas. This was designed so in order to mimic how classical painters can easily pick different colors with a physical palette in real life. This was reimagined in the redesign as the quick panel.





Testing: Subjective Evaluation

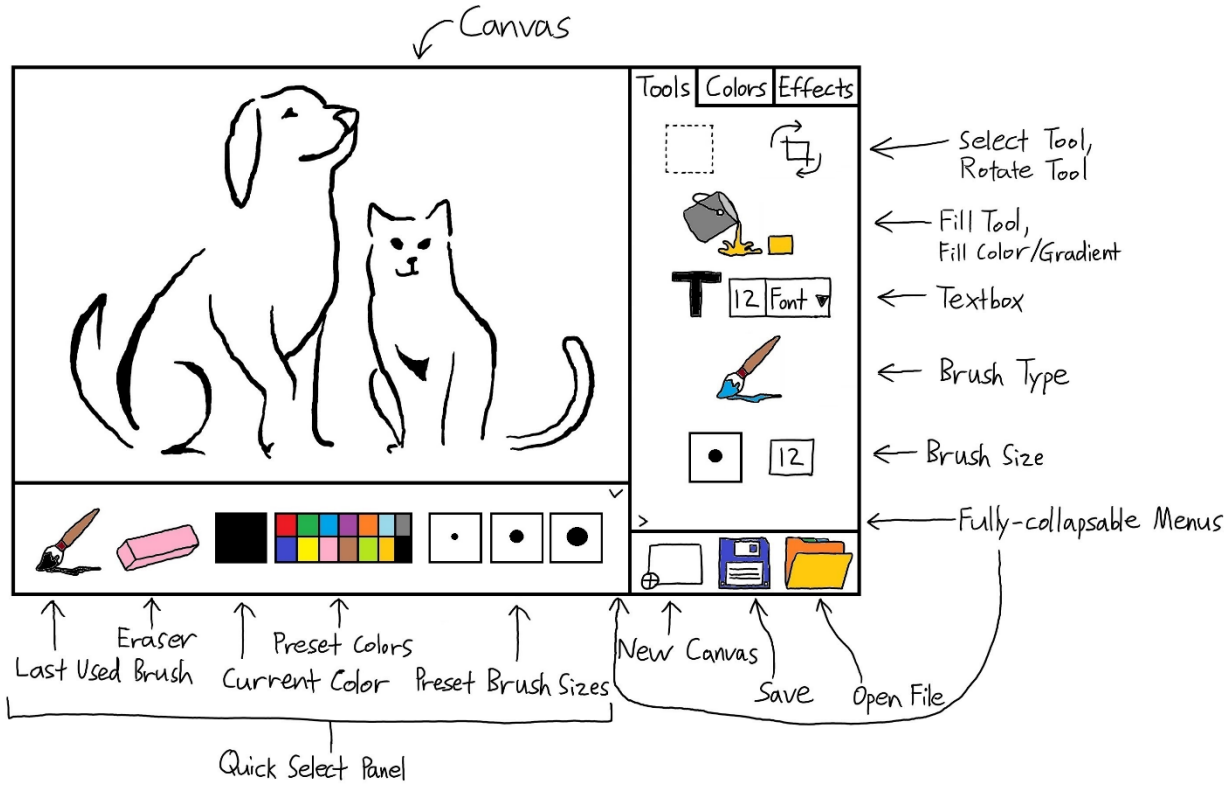
We carried out a casual subjective evaluation for our first iteration of the program based on our **initial design** between our friends on campus. Evaluators received a sheet of paper with simple photo-editing tasks (detailed in the **use cases** in assignment 2) which can all be done with the tools implemented in the program. The evaluators were asked to complete these tasks without any help. Observations of their mood, feelings and verbal responses were recorded during the test.

Evaluator	**Significant Quote During the Test**	**Time to Complete**	**Comments After the Test**	**Overall Mood**
Eugene Schulze	"Where is the brush size?"	0:57	"The menu is too crowded and too many options."	Critical
Thomas Irwin	"Crap, I messed up. "	------ [evaluator gave up]	"Why's there no eraser???" "I shouldn't have to reset the entire board!"	Frustrated
Jana Wong	"What does this button do?"	1:31	"This is why I stick with Microsoft Paint."	Confused
Randal Gonzalez	"Ez"	0:34	"Noice, so how much are you guys selling it for?"	Content
Casey Mathis	"L-A-B C-M-Y-K? What are all these letters?"	0:53	"All I know is R-G-B."	Confused

From the lectures, we remembered that users often try to find the easiest (may not be most efficient) way to complete a task when using a Human-Computer Interface, so we made a few adjustments in the **final design** based on the feedback we received.

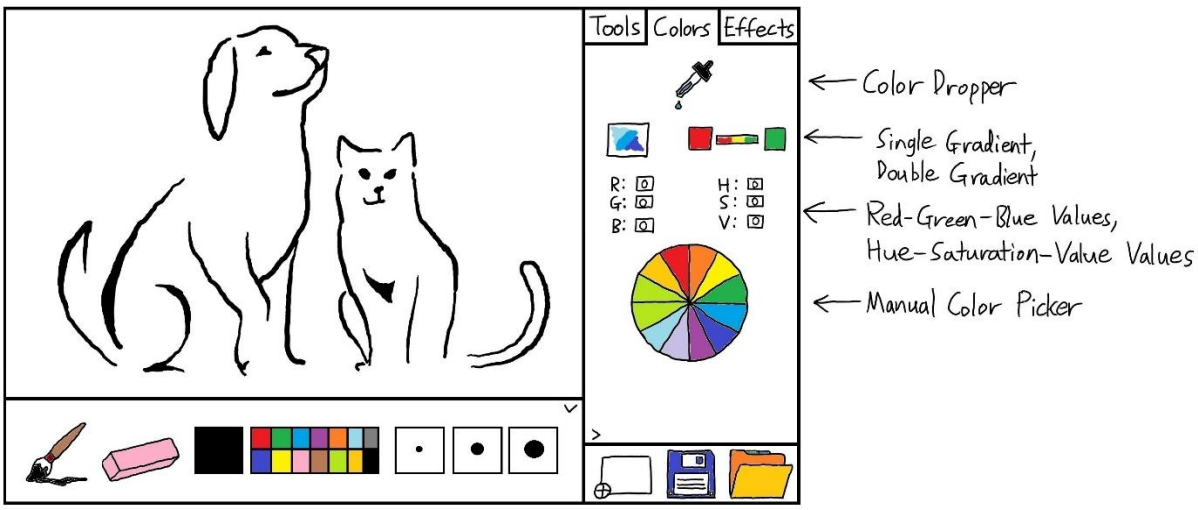
Final Design

After a couple of iterations, we made a few changes to our initial design.

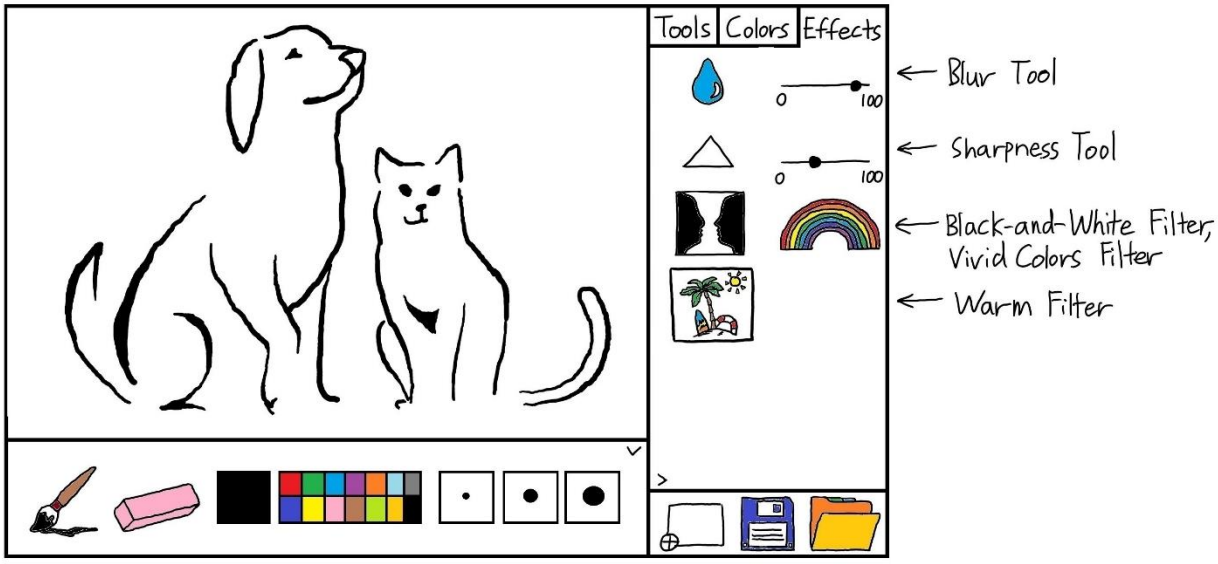


First, we moved the toolbelt menu to the right. Most people read/view from left to right. Thus, the components on the left comes first and is viewed as more important. In this case, the canvas.

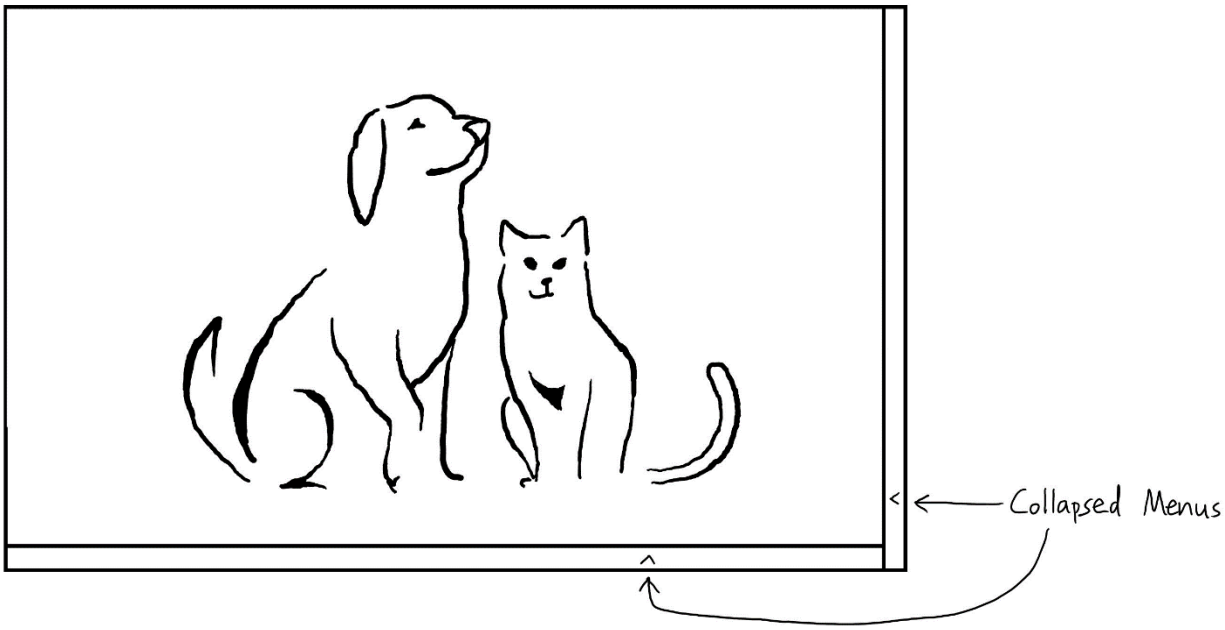
Second, we added a quick select panel at the bottom for easy access to the most frequently used features without pulling up the complicated toolbelt menu (can be hidden).



Third, we removed unpopular color models such as L-A-B, C-M-Y-K in favor of R-G-B for simplicity.



Fourth, we altered some icon images for clarity.



Both panels/menus are fully collapsible.

Implementation Overview

- Our program was implemented in Java Swing using IntelliJ IDEA and Eclipse.

Development Phase 1

Features	Implementation
Canvas	A JPanel that modifies the inherited paintComponent method to draw the image and user modifications.
Empty menus and tabs	Collapsable JPanels (toggle with shortcut keys “1” and “2”)
New Canvas	A JButton that clears the Canvas.
Save File	A JButton that takes a snapshot of the Canvas and saves it as “Saved_iImage.jpg” in the current directory.
Open File	Opens up the default JFileChooser dialog for the user to select an image file. Gives an pop-up error if the file is not an image file, but does not crash the program.

Development Phase 2

Phase 2 of the implementation involved creating the quick select panel and the color tab.

The bottom panel near the screen is where the quick select panel is located. The quick select panel is used to hold the most frequently used features to allow the user to access these tools quickly and easily. The first feature of the quick section is the paintbrush feature which when clicking it switches to the paintbrush. The next feature is the eraser feature, which allows the user to eraser anything drawn on the canvas. The way it was implemented was quite simple, since the canvas is already white, when switching to the eraser it switches to the white color paint brush allowing the user to erase the canvas. Next was the current color indicator, which shows the current color you are using. The is done by creating a circle using the graphics component in java and changing the color of that circle to whatever color the user changes the paint brush to. Next is the preset color selection. This was done by creating a square button and coloring it. By pressing the button, it changes the brush color. Lastly, the preset brush size selection was also a button with an image on it. When pressed, using javas' graphics2D it sets the stroke size to a larger or smaller size.

The color tab is located on the second tab and is shown on the panel on the right size of the screen. The color tab is used to deal with the color aspect of the application with regards to changing the color to a custom color and using the color dropper tool. The first feature is the color dropper tool. The tool is used to highlight a color and allow that user to change the color to that color. This would have been implemented by when the user presses the icon, it would take a screenshot on the screen. It then would map the colors out to certain coordinates on that screenshot. When the user clicks on the screen, it would take those coordinates and find out the color and then change the color accordingly. Lastly, we have the custom color picker, which is a circle of red, green and blue all mixed together. The user can click on the color wheel, allowing them to move a pointer within the wheel. The user then can select a color by click again on the wheel on their desired color. It will then change the current color and the paintbrush color that that color. This was done by creating a color wheel class and a color model class. The color wheel class creates the wheel with java graphics as well the pointer/selector. The color model class allows you to access what color the user is selecting with a few get methods as well as allow you to set a color as well.

Development Phase 3

Features	Implementation
Tools panel	A JTabbedPane that contains a JPanel, which in turn contains most of the canvas drawing features
Effects panel	A JTabbedPane that contains a JPanel, which in turn contains most of the image manipulation features
Rotate	A JButton that turns the degree of whatever image is on the screen by 15 degree increments
Brush size	A JSlider that allows you to choose the stroke width. The current brushstroke width is shown in a JLabel

TextBox	A 'library' or new class, InvisibleTextField was going to be used for this implementation, but it could not be gotten to work in time
Blur	A JSlider was used to control the blur of an image between values 0 and 9 (1 to 10).*
Sharpness	JSlider that could not be implemented in time. (Its implementation turned out to be nontrivial)
Black and White	A JButton that turns colors into grayscale by getting the average RGB value for each respective pixel
Warm	A JButton that decreases by a percentage, the Blue and Green of each pixel in the image
Vivid	A JButton that changes the hue of a loaded image by 5% increments

* In order to make applying blurs very quick, an array of size 10 of whichever image is loaded onto the canvas is saved. Blur effects of varying degrees – from 1 to 10 – are applied to the image and stored into the array at its matching index. Therefore, changing the blur slider actually only changes the index of which image the user is accessing.

Testing: Diagnostic Evaluation

A diagnostic evaluation was performed at the end of our development cycle based on our **final design**, involving the participation of 3 users. We tried our best to find users whom the application was intended for. Based on our user profile in assignment 2, two of the users we got to participate fell in to the casual/non-experienced and one of them fell into the experienced category.

Our evaluation consisted of three user-based evaluation techniques: observational methods, experimental methods and questionnaires.

For the observational method, what we did is before we did the experimental method, we had the user freely use the program without any guidance. As they did that, we viewed as the used the program taking notes along the way. We did a direct observation by being present to view the user, use our program. The first thing every user did was try to draw on the canvas. Next, they tried to change the color in the preset color selection. We noticed that the non-photoshop users didn't use any of the filters and effects to a great degree.

The experimental method consisted of our 4 uses cases mentioned in assignment 2. Now as we were short for time and underestimated the difficulty of adding a crop and textbox feature, we did not add those features to our program. Instead we asked the user to add a light red color with a brush size of 6 on the picture, slightly sharpen the image and to add a warmth filter. So, the use cases now are:

1. Choose the photo to be edited
2. Add a light red color with a brush size of 6 on the picture
3. Slightly sharpen the image
4. Add a warmth filter to the image
5. Export the image to an appropriate image format

Our finding with this evaluation were quite good. The casual/non-experienced users were able to do steps 1 and 2 quite well. For steps 3 and 4 the users were a little confused on which effects were blur, sharpness and warmth. After some experimentation, they quickly realized the intuitive meaning of the icons. Lastly, the users had no problem with saving and exporting the image. The experienced user liked the overall simplicity and layout of the application. The user was able to do all the uses cases well but did mention that the sharpen and warmth button could be clearer. From this, we concluded that we should add tooltips along with the image icons to make the button even clearer for the user.

Lastly, the questionnaire consisted of 5 questions of how the user experience was. Each of the questionnaires had 5 choices from very not satisfied to very satisfied. The responses were widely favorable.