Rapport de Projet - OELA

2021-2022

Projet S4

On est là alors

Alexandre Devaux-Riviere Kaël Facon

Grégoire Vest Mathis Rabouille

Table des matières

1 Introduction

1.1 Présentation du groupe

Notre groupe est composé de 4 ex-lettons ayant dü rentrer en France suite au début du conflit entre la Russie et l'Ukraine. Nous nous sommes rassemblés au sein du groupe "On est là alors" car nous avions des idées en commun, nous voulions tous les quatre faire un logiciel en rapport avec le dessin et l'édition d'images. C'est donc dans cette optique que nous avons pensé à Decolor notre projet de logiciel de dessin et d'édition d'images.

1.2 On est là alors

Ce nom de groupe est au départ une blague que nous faisions entre nous. En effet, après être revenus en France il nous est naturellement venu le fait de dire "On est là" de manière sarcastique pour essayer de relativiser l'échec de notre semestre à l'étranger. Lors de la décision du nom de groupe "On est là" nous est alors apparu comme évident comme nom de groupe. Nous l'avons par la suite amélioré en "On est là alors" pour signifier que, quitte à être de retour en France, **alors** autant travailler et faire un bon projet de S4.

1.3 Membres du groupe

- Alexandre Devaux-Riviere (Chef de groupe)
- Kaël Facon
- Grégoire Vest
- Mathis Rabouille

1.4 Choix du logo

Nous avons opté pour un logo simple, discret mais aussi à la fois efficace et moderne regroupant l'ensemble des lettres des initiales de notre nom de groupe.

FIGURE 1 – Logo "On est là alors"

2 Présentation du projet - Decolor

2.1 Idée principale de Decolor

2.2 Etat de l'art

Énormément de logiciels existent aujourd'hui pour dessiner et/ou modifier des images. Notre but est de créer une application de dessin donc nous allons nous inspirer des logiciels les plus connus pour voir ce qui ressort le plus souvent.

Le logiciel le plus connu est Paint, qui possède une interface et des outils très simple d'utilisation. Au début nous allons beaucoup nous inspirer de cet environnement de travail car il y a toutes les bases dont nous auront besoin : interface de la feuille, interface utilisateur, pinceau, gomme, couleurs, outil sélection, enregistrement de fichier...

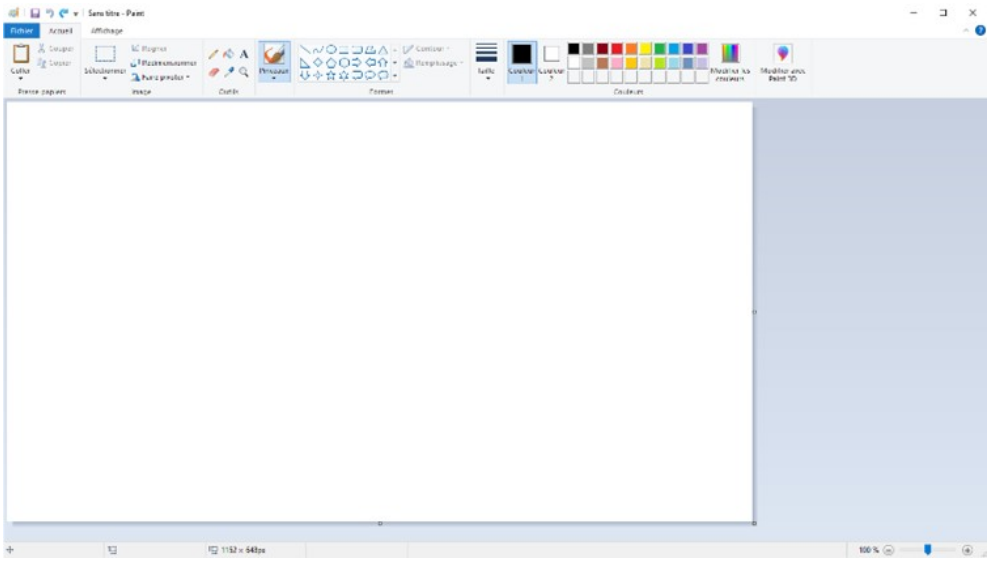


FIGURE 2 – Interface Paint

Ensuite, pour ce qui est des logiciels de dessin, nous avons l'embarras du choix : Photoshop, Gimp, Clip Studio Paint, Medibang Paint Pro, Krita, Procreate... On se concentrera évidemment plutôt sur les logiciels designés pour le dessin que sur ceux créés à la base pour la retouche d'image (par exemple Gimp est plus souvent utilisé pour les images que pour le dessin).

Ce qui est intéressant dans notre idée de projet, c'est que nous pourront ajouter des outils si nous voulons avancer encore plus notre projet. Voici donc quelques inspirations d'outils utilisés habituellement sur les logiciels de dessins et que nous pourront ajouter si le projet avance bien :

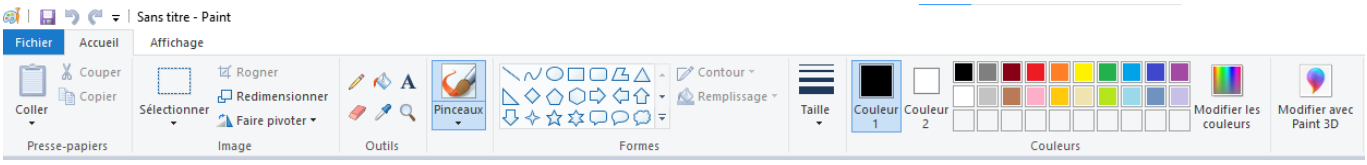


FIGURE 3 – Outils Paint

— **Outil sélection et déplacement** : un classique dans les logiciels de dessin/retouche d'images. Cet outil permet de déplacer/agrandir/tourner un certain nombre de pixels.

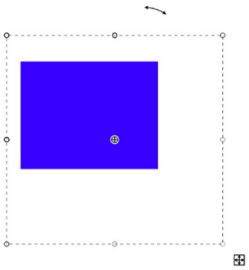


FIGURE 4 – Outil sélection sur Paint.net

— **Ajout de calques** : outil indispensable des logiciels de dessin pour facilement créer des choses plus complexes.

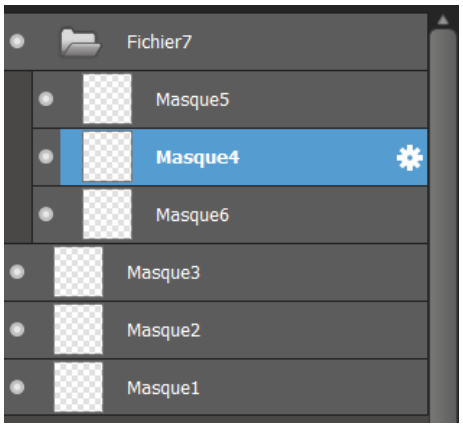


FIGURE 5 – Les calques sur Medibang

— **Choisir la couleur** : plutôt que de proposer des couleurs de base, faire en sorte que l'utilisateur puisse choisir lui-même sa couleur avec un code hexadécimal, un code RGB ou autre.

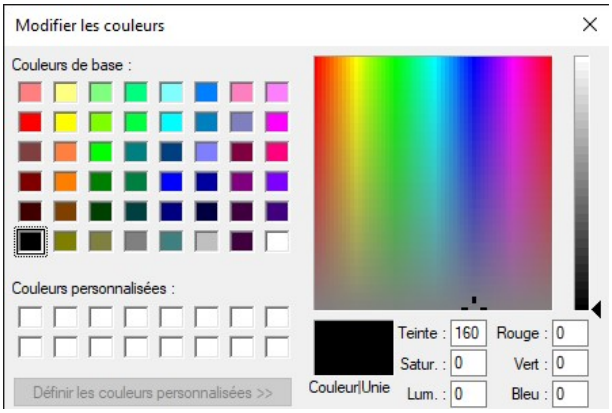


FIGURE 6 – Sélection de couleur sur Paint

— **Des pinceaux différents** : au début nous allons implémenter le pinceau de base comme celui de Paint, mais nous pourrons ensuite ajouter d'autres pinceaux pour facilement varier les possibilités de dessin.

FIGURE 7 – Une toute petite portion des pinceaux disponibles sur Clip Studio Paint

Bien sûr, ces outils sont déjà très complexes à coder alors que les outils de base devraient déjà nous prendre tout le temps du projet. Rien que pour le pinceau, il faut coder l'encrage, la couleur, la taille, l'opacité...

Cependant cela montre que même si énormément de logiciels de dessin existent, il est facile de cibler un type de dessinateur en fonction des outils que nous voudrons implémenter/perfectionner et de l'ergonomie du logiciel que nous allons créer. Par exemple, un logiciel « simple » comme Paint permet à n'importe qui d'apprendre à l'utiliser rapidement mais ne permet pas de faire des dessins complexes.

2.3 Logo

FIGURE 8 – Prototype numéro 1

FIGURE 9 – Prototype numéro 2

Notre logo n'est encore qu'un prototype et peut, de ce fait, complètement changer mais il concilie jusqu'ici parfaitement la thématique de l'application que nous vous présentons ci-dessous.

2.4 Origine / Ressources

Chaque membre du groupe étant très familier avec les bibliothèques SDL et GTK du fait de notre récent travail sur un OCR résolvant des sudokus sur des images, nous avons décidé de réaliser une application graphiquement similaire avec le projet de S3 tout en nous donnant l'opportunité d'agir sur des images de façon plus poussée. Notre application utilisera GTK comme bibliothèque graphique pour notre interface, l'espace de travail sur images ainsi que les interactions sur celle-ci. Cela nous permettra d'implémenter par exemple des boutons pour appeler certaines fonctions de modification ou montrer une image.

La bibliothèque SDL sera utilisée pour la manipulation d'images. Cela nous permettra de récupérer une image sous forme de "surface" que l'on pourra stocker en mémoire de manière à pouvoir la modifier à notre guise (nous avons pu manipuler ces bibliothèques lors de notre projet précédent, ce qui pourra nous être très utile). L'application choisie est inspirée de plusieurs applications d'édition d'images et de dessin telles que Paint (windows seulement), Photoshop (payant), GIMP (difficile à prendre en main).

Le type d'application choisie n'a pas été direct, nous nous sommes d'abord rassemblés et avons débattu sur plusieurs idées et thèmes ayant du potentiel (notament un logiciel comme géogébra ou un password manager).

2.5 Récapitulatif des bibliothèques

2.5.1 Simple DirectMedia Layer (SDL) et (SDLgfx)

SDL :

Simple DirectMedia Layer est une bibliothèque de développement multiplateforme libre conçue pour fournir un accès de bas niveau au matériel graphique. Elle nous permet de travailler sur la manipulation d'images. Nous avons déjà pu utiliser cette bibliothèque lors de différents TPs ou encore lors de notre projet de troisième semestre qui consistait à résoudre un sudoku en faisant un OCR.

SDLgfx :

Le code de la bibliothèque SDLgfx fournit des routines de dessin de base et ajoute des fonctions utiles pour le zoom images et fait des traitements d'image de base sur des tableaux d'octets.

2.5.2 GIMP Toolkit - GTK

GTK est une boîte à outils multiplateforme gratuite et libre pour créer des interfaces utilisateur graphiques. Elle nous permettra de pouvoir réaliser une application simple et facile d'utilisation. Nous avons jugé que GTK serait l'outil parfait pour mener à bien nos ambitions.

2.6 But, intérêt algorithmique, principe

L'application (donc nommée Decolor) sera principalement orientée vers un logiciel très simple d'utilisation permettant à l'utilisateur de dessiner ou modifier une image afin de laisser libre cours à sa créativité. L'interface graphique de l'application, pour la partie outils, aura une fonction qui permettra de charger une image à partir d'un explorateur de fichier et de faire des modifications sur cette dernière ou de la manipuler. Par la suite, cette image pourra être sauvegardée dans les fichiers locaux de l'ordinateur ou écrasée (au choix). Nous allons opter pour une interface sobre, ergonomique et surtout facile d'utilisation. L'intérêt algorithmique de notre projet se tourne vers l'imagerie notamment vers le traitement et l'édition d'images comprenant la modification et le stockage de pixels.

Ouverture de l'application par défaut :

Le principe de l'application est de mettre l'utilisateur directement sur une surface blanche, sur laquelle il pourra dessiner avec les différents outils présents, mais il pourra également éditer une image qu'il chargera au préalable.

Ce projet nous permettra donc d'acquérir de l'expérience que ce soit en programmation, en design ou encore en communication (rendus en latex + site de présentation).

3 Fonctionnement

A l'instar des logiciels dont nous nous sommes inspirés, notre logiciel permettra d'exécuter des modifications basiques de l'image. Par exemple dessiner à l'aide d'un pinceau ou appliquer des filtres sur l'image. Toutes les modifications appliquées à l'image seront affichées en temps réel dans l'interface graphique. Voici quelques exemples supplémentaires d'outils que nous pourrions implémenter dans notre projet.

3.1 Édition de la structure / format de l'image

3.1.1 Rotation de l'image

Parmi les opérations les plus basiques, la rotation de l'image sera probablement une des premières à être implémentée. Il sera possible de faire tourner l'image d'un angle précis souhaité.

3.1.2 Rogner l'image

L'utilisateur pourra rapidement modifier la forme d'une image en la rognant pour lui donner la forme spécifique qu'il souhaite avoir. L'image est automatiquement découpée pour remplir la géométrie de la forme sélectionnée tout en conservant ses proportions.
Lorsque vous souhaitez rogner une image, des traits épais noirs apparaissent aux coins de l'image et au milieu des différents côtés. Le rendu de la nouvelle image sera les pixels qui seront contenus dans les traits noirs.

3.1.3 Agrandir et rétrécir

Lorsque l'utilisateur souhaite agrandir une partie de son dessin, il aura à sa disposition un outil permettant d'agrandir ou de rétrécir les images. La complexité dans la réalisation de cette option sera sûrement de garder la bonne proportion de l'image sans la déformer sauf si l'utilisateur choisit d'étirer l'image dans un quelconque sens.

3.1.4 Transparence / Filtres

La **transparence** nous sera essentielle quant à la création de différents calques, sans cette dernière nous ne pourrions pas superposer les masques créant une seule et unique image. La gestion de la transparence nous contraint cependant sur les formats d'images que nous pouvons utiliser. En effet, certains formats comme le .jpeg ne comprennent pas la variable *alpha* correspondant à l'indice d'opacité d'un pixel alors que d'autres comme le format .png l'incluent.

Les **filtres** s'appliqueront sur l'entièreté des pixels d'un calque. Ces derniers permettront de modifier de manière homogène l'image. Les filtres les plus courants sont ceux de *Noir&Blanc, Sepia, Inversion de couleur, Contraste, Luminosité, Saturation, Netteté*. Nous essayerons donc d'implémenter le maximum de ces filtres afin d'avoir un logiciel complet et utile, certains étant plus durs à réaliser que d'autres.

FIGURE 10 – Noir&Blanc / Sépia / Inversion de couleur

FIGURE 11 – Contraste / Luminosité / Saturation

FIGURE 12 – Netteté augmentée

3.2 Édition des couleurs de l'image

3.2.1 Palette de couleurs

Nous allons commencer par implémenter une large palette de couleurs qui sera la base des outils d'édition qui suivent permettant à l'utilisateur d'avoir de larges possibilités dans le choix de couleurs (RGB) à partir des valeurs de rouge, vert et bleu (par défaut la couleur noire sera activée).

3.2.2 Pipette

L'outil pipette est un outil très pratique et simple d'utilisation. Il s'agit seulement de récupérer une couleur présente sur un endroit voulu de l'environnement dans lequel il peut fonctionner. Le prochain outil sélectionné (que nous détaillons juste après) portera ainsi cette couleur précisément. La couleur sera, de plus, enregistrée dans la palette de couleur.

3.2.3 Crayon

L'outil crayon servira à l'utilisateur de se mettre en mode dessin. Lorsqu'il appuiera sur le clique gauche de sa souris il pourra faire des traits de la couleur de son choix, sélectionnable à partir de la palette de couleurs. Il pourra de plus changer à tout moment l'épaisseur de son crayon, donc l'épaisseur du trait.

3.2.4 Seau

Le seau permettra le remplissage d'une zone de couleur homogène avec la couleur sélectionnée au préalable sur la palette.

3.2.5 Forme 2D

Si le temps restant nous le permet, nous souhaiterions implémenter un outil permettant de créer des formes géométrique simples telles qu'un cercle, un carré, un rectangle. Ces formes seront créées à l'aide d'algorithmes simples, qui à partir des coordonnées du pointeur, créeront la forme géométrique d'une taille variable (selon le curseur).

3.2.6 Zone de texte

Nous pourrions également ajouter un outil "zone de texte", un grand classique des logiciels de type Paint. Il permettrait de sélectionner une zone pour ensuite écrire dedans avec toutes les options habituelles, le choix de la police, de la taille ou l'italique et le gras.

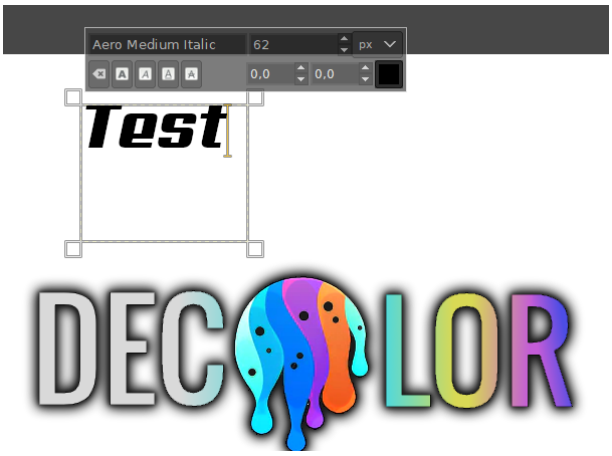


FIGURE 13 – Zone de texte

4 Aspects Techniques et Méthodologiques

4.1 Moyens matériels

Nous allons réaliser ce projet durant le semestre aménagé pour les étudiants revenant des pays baltes, nous serons donc en distanciel. Les méthodes de travail seront donc légèrement différentes et le matériel que nous pourrons utiliser également. Cela ne nous empêchera pas de nous donner à fond dans ce projet en parallèle des cours que nous aurons en distanciel. Nous disposerons tout de même de nombreux logiciels, grâce à EPITA, qui pourront nous servir durant le développement de ce projet.

4.2 Moyens intellectuels

Nous avons à disposition Internet qui est une grande bibliothèque contenant les informations et les ressources nécessaires nous expliquant comment utiliser les logiciels mis à disposition ou développer dans certains langages à travers les différents tutoriels, manuels numériques sur les logiciels et forums d'entraide existant. Nous aurons également la possibilité de recevoir de l'aide de la part des professeurs de l'EPITA, des ASMs et des étudiants des années supérieures de l'école, qui ont tous plus d'expérience que nous.

5 Découpage du projet le long du semestre

5.1 Tableau de répartition des tâches

	Alexandre	Kaël	Grégoire	Mathis
GTK - Interface		**Supp**		**Resp**
Interface graphique globale		X		*
Implémentation des outils	*	X	*	*
Chargement et enregistrement des fichiers				X
Implémentation de l'environnement de travail	*	*		X
Fenêtres informartives		*		X
SDL	**CoResp**		**CoResp**	
Création des outils	X		*	
Création des formes 2D	X		*	
Gestion des images	*		X	
Edition de la structure des images	X	*	*	
Application de filtres		*	X	*
Structure du Projet	**Supp**	**Resp**		
Ergonomie de l'application	*	X		
Compilation des fichiers (Makefile, headers...)	X	*	*	*
Communication	**Resp**	**Supp**		
Site Web	X	*		
Rendus Overleaf	X	*	*	*

Légende :

X = Responsable, * = Suppléant

5.2 Prévisions et accomplissement par soutenance

Soutenance / Tâche	Prévisions S. 1	Accompli S. 1	Prévisions S. 2	Accompli S. 2
GTK - Interface	70	80	100	110
SDL	50	70	100	100
Structure du Projet	60	70	100	100
Communication	80	80	100	100

TABLE 1 – Tableau des soutenances (en %)

6 Avancement Chronologique (Soutenance 1) - Projet Decolor

6.1 Structures LIFO / FIFO

Afin de mieux comprendre l'implémentation algorithmique de certains outils et de certaines fonctionnalités, nous introduisons ici les structures qui nous ont permis de les réaliser.

6.1.1 Outils - (FIFO)

Afin de manipuler certains outils comme celui du seau avec son parcours largeur de l'image, il a fallu créer une structure FIFO (une queue) que nous avons choisi circulaire et sans sentinelle pour une meilleure optimisation (liste chaînée circulaire).

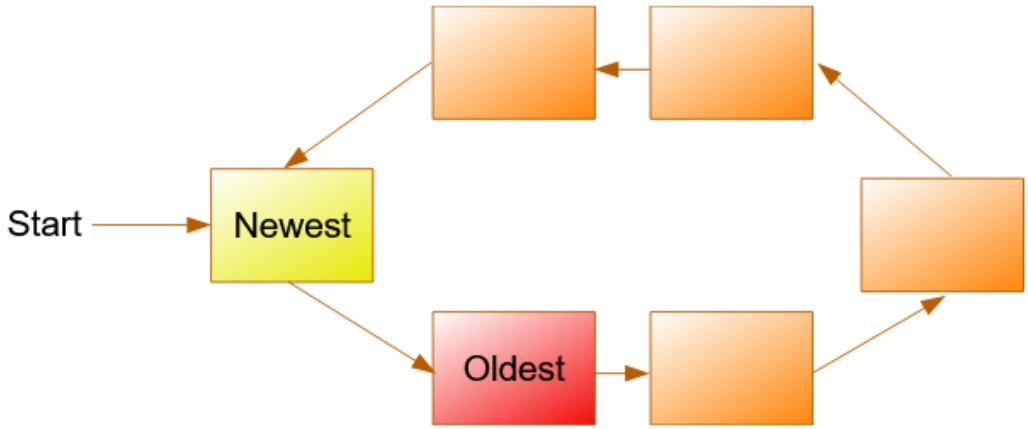


FIGURE 14 – Schéma de la structure FIFO choisie

6.1.2 Retour en arrière / Retour en avant - (FIFO + LIFO)

Dans le but de rajouter la fonctionnalité majeure pour notre logiciel que représente l'option de retour en avant / retour en arrière, il fallu créer une structure spéciale. En effet, le problème était de stocker dans une pile les modifications d'une image au fur et à mesure en enlevant les éléments du fond de la pile afin de limiter l'enregistrement des modifications (à 15 dans notre cas).

Nous avons de ce fait imaginé et implémenté la fusion d'une pile et d'une queue (FIFO + LIFO). De plus, du fait de l'interconnexion des options retour en arrière / avant, il fallait donc deux piles de même nature faites pour coïncider de tel sorte qu'un élément dépilé devait se retrouver dans la pile communiquante et être affiché.

Afin de simplifier l'explication de cette option, voici des schémas de la structure et des interactions que l'on peut y retrouver ("élément actuel" représente l'élément (image) affiché sur l'interface et "Data Element" représente un pointeur SDL_Surface) :

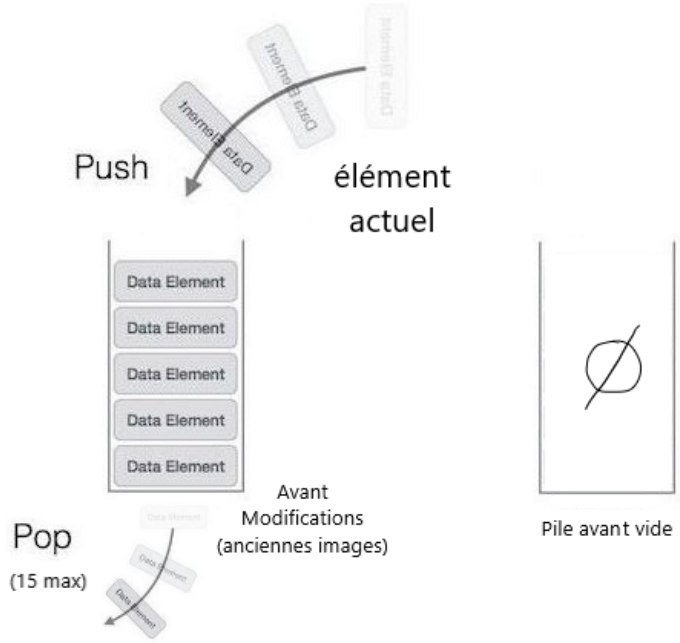


FIGURE 15 – Empilation des modifications de l'image

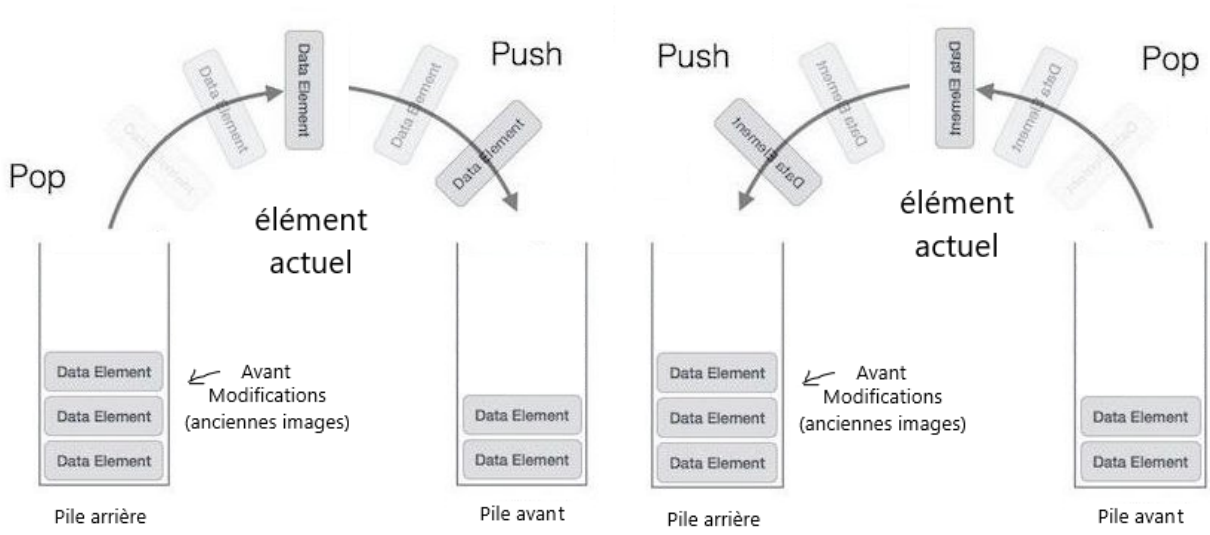


FIGURE 16 – Schéma du Retour en arrière (ctrl + Z) et Retour en avant (ctrl + Y)

Ainsi, pour résumer, à chaque modification de l'image, on empile celle-ci dans la pile arrière et on vide la pile avant. Lorsque l'on retourne en arrière, l'ancienne image s'il y en a une (celle en haut de la pile arrière) est affichée et l'image sur laquelle on était passe dans la pile avant. Lorsque l'on retourne en arrière, l'image actuelle est empilée dans la pile arrière et on affiche l'image dépilée de la pile avant.

De cette façon le retour en arrière et avant est optimisé et fonctionne parfaitement comme le fait un logiciel de type Paint3D.

6.2 Edition d'image - SDL

6.2.1 Algorithmes derrière les outils

Le crayon :

La création du crayon passe par trois étapes et possède plusieurs options (la couleur et la taille). Pour recréer un tracé de crayon, il y a, tout d'abord, une fonction qui s'occupe de dessiner un point de la couleur et de la taille sélectionnée en forme de losange.

Dans un second temps, une fonction s'occupe de dessiner un segment, continuité de points créés par la première fonction, entre deux coordonnées de l'image grâce à l'algorithme de Bresenham.

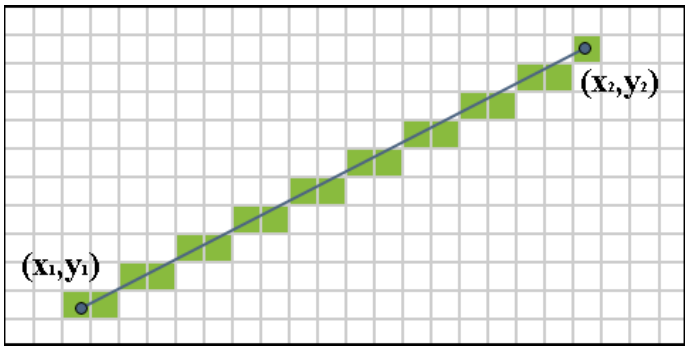


FIGURE 17 – Algorithme de Bresenham - Traçage d'un segment

La troisième partie du crayon se trouve au niveau de l'interface GTK avec une fonction qui, en continu, enregistre les coordonnées d'un clic maintenu sur la page et est expliqué dans les parties de ce rapport concernant notre interface.

L'ensemble de ces fonctions contribue à créer un pinceau plutôt fluide et précis.

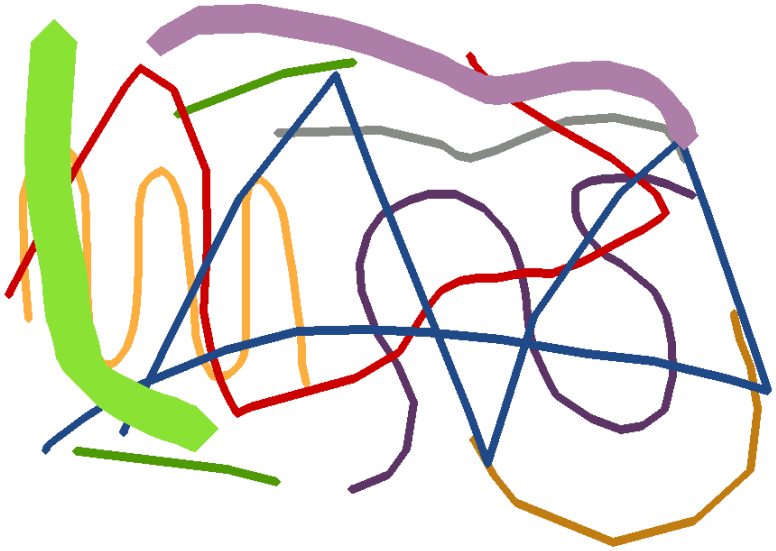


FIGURE 18 – Crayon Fonctionnel avec option couleur et taille

Le seau :

Le seau, outil emblématique d'un logiciel d'édition d'image, se fait par un parcours largeur du graphe qu'est l'image (pointeur SDL_Surface). Ce parcours utilise une structure FIFO comme introduite plus tôt. L'endroit du clic sur l'interface est envoyé dans la fonction du seau, et la couleur du pixel correspondant à ces coordonnées est enregistrée. Par la suite, l'ensemble des pixels aux alentours qui sont de la même couleur que celle enregistrée rentrent dans le parcours largeur et sont modifiés par la suite.

Le seau possède aussi un seuil de sensibilité afin d'inclure plus ou moins de variance de couleur de pixels à changer.

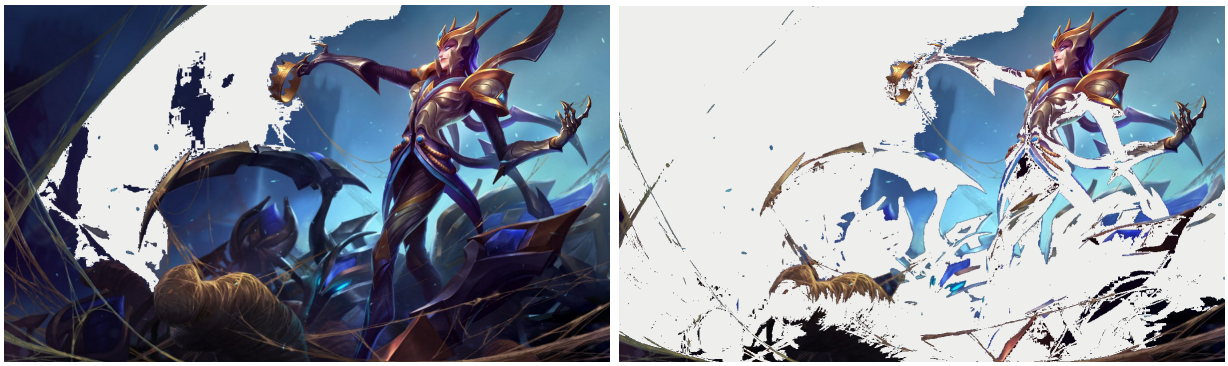


FIGURE 19 – Application du seau sur une image avec un seuil de 10% et 30%

FIGURE 20 – Application du seau sur une image avec un seuil de 60% et 90%

FIGURE 21 – Application du seau sur une image avec un seuil de 100%

La pipette :

Nous n'avons pas eu à implémenter la pipette. En effet, elle est déjà disponible directement sur l'interface avec GTK. Plus d'informations sont donnés dans la suite de notre rapport.

La gomme blanche :

Outil essentiel pour dessiner, la gomme utilise exactement le même algorithme que celui du crayon sauf qu'à la différence de celui-ci, la couleur sera toujours le blanc !

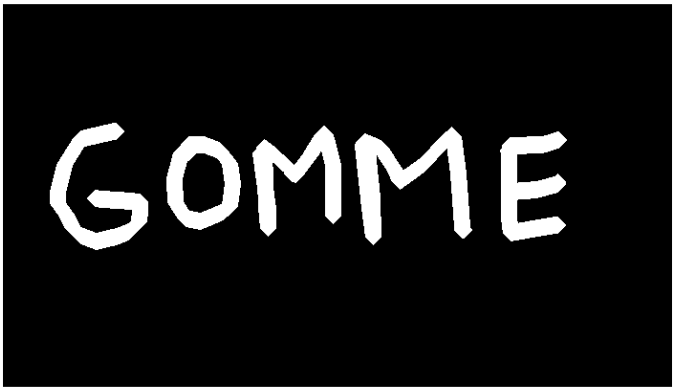


FIGURE 22 – Gomme fonctionnelle avec taille réglable

La gomme qui restore l'image :

Cette gomme reprend le fonctionnement de l'autre gomme sauf qu'à la place de placer du blanc sur les pixels des traits, on place les pixels aux même coordonnées de l'image précédemment chargée (blanc par défaut). Grâce à cet outil il est beaucoup plus simple de dessiner sans se préoccuper des potentielles erreurs de dépassement sur l'image modifiée.

FIGURE 23 – Gomme d'annulation fonctionnelle avec taille réglable

6.2.2 Formes 2D

Le segment :

Pour réaliser ce segment, on reprend un algorithme précédemment utilisé pour le crayon sauf qu'à défaut de tracer des segments en continu, seulement un segment entre deux coordonnées sera tracé sur l'image grâce à l'algorithme de Bresenham précédemment expliqué (coordonnées du début du clic et du relâchement du clic).

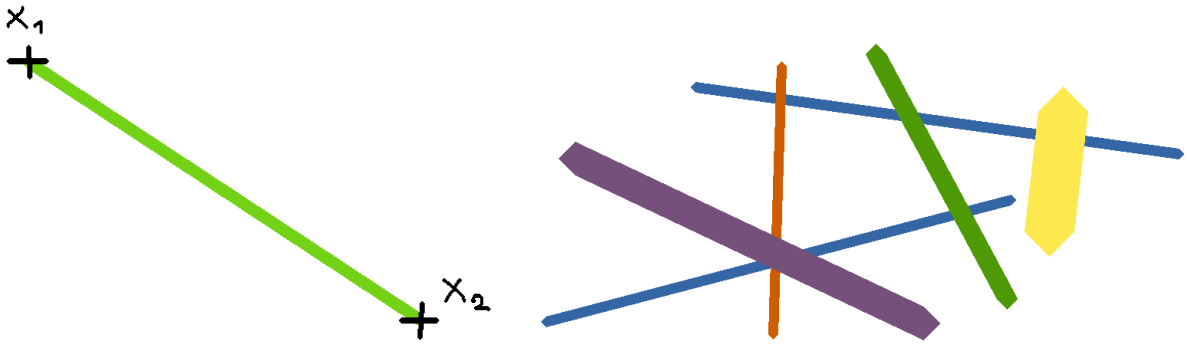


FIGURE 24 – Forme Segment avec couleur et épaisseur modifiable (Exemples X1 vers X2)

Le rectangle :

Afin de concevoir le rectangle vide, les coordonnées au clic puis au relâchement du clic sont enregistrées et un rectangle qui est intérieur à ces deux coordonnées est créé. Cela se fait grâce à l'aide d'une fonction de tracés verticaux et d'une fonction de tracés horizontaux appelées deux fois respectivement pour former les quatre cotés de la forme. De cette façon, l'utilisation de cet outil se fait de façon plus précise. La taille et la couleur sont aussi des options modifiables.

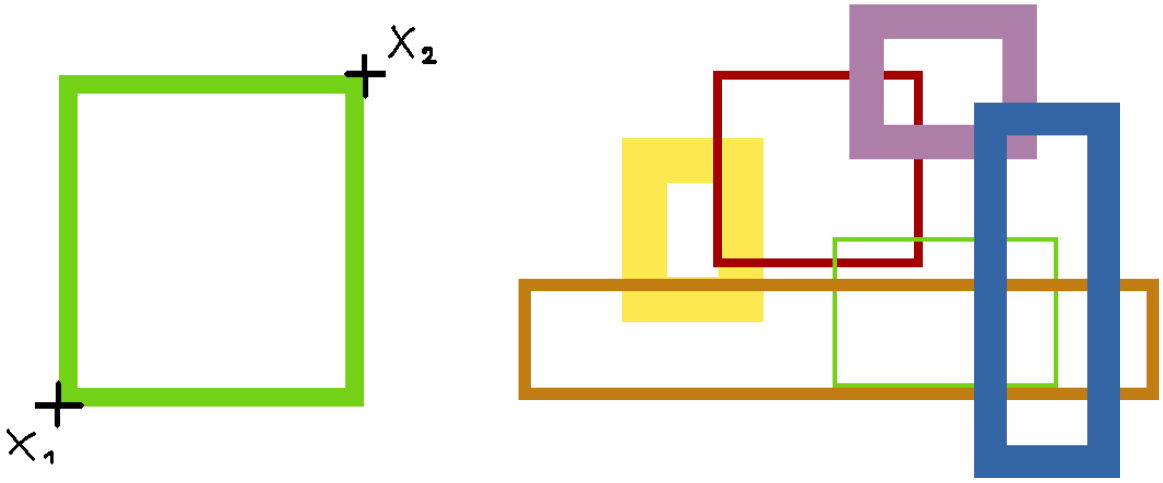


FIGURE 25 – Forme Rectangulaire avec couleur et épaisseur modifiable (Exemples X1 vers X2)

Le triangle :

Pour la conception du triangle vide, les coordonnées au clic puis au relâchement du clic sont enregistrées et un triangle qui est intérieur à ces deux coordonnées est créé. Certains calculs sont faits pour obtenir les coordonnées des sommets de celui-ci en fonction du sens des deux coordonnées X1 et X2 pour finalement obtenir les points A et B. Par la suite, trois segments sont formés entre X1 et A, A et B, ainsi que B et X1. De cette façon, l'utilisation de cet outil se fait de façon plus précise. La taille et la couleur sont aussi des options modifiables.

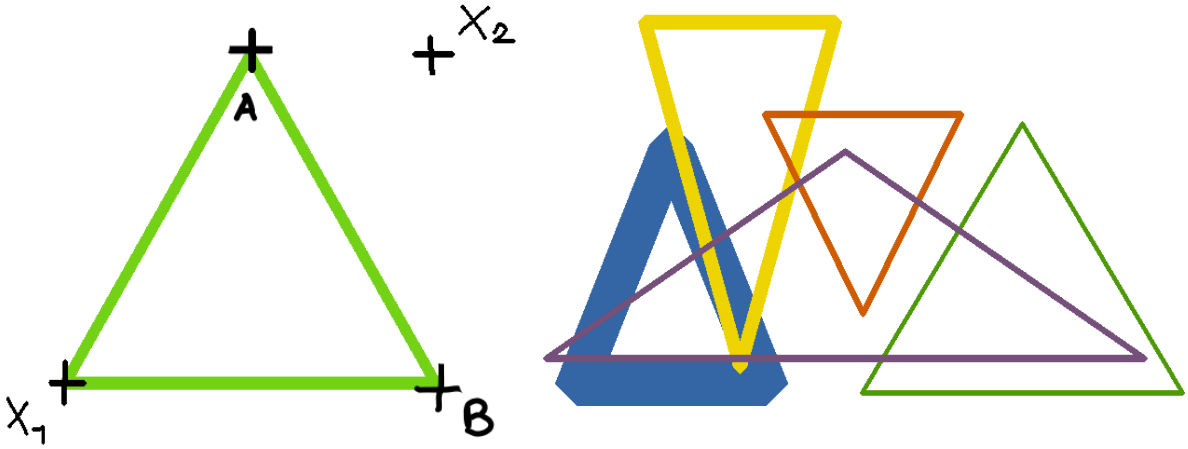


FIGURE 26 – Forme Triangulaire avec couleur et épaisseur modifiable (Exemples X1 vers X2)

Le cercle :

Pour la conception du cercle vide, les coordonnées au clic puis au relâchement du clic sont enregistrées et un cercle qui est intérieur à ces deux coordonnées est créé (de centre X1 et de rayon X1 - X2). Par la suite, huit arcs de cercles sont formés par symétrie centrale (X1) et se lient pour former le cercle entier grâce à l'algorithme du cercle de Bresenham puis remplacé par l'**algorithme d'Andres** à la soutenance 2. De cette façon, l'utilisation de cet outil se fait de façon plus précise et optimisée. La taille et la couleur sont aussi des options modifiables.

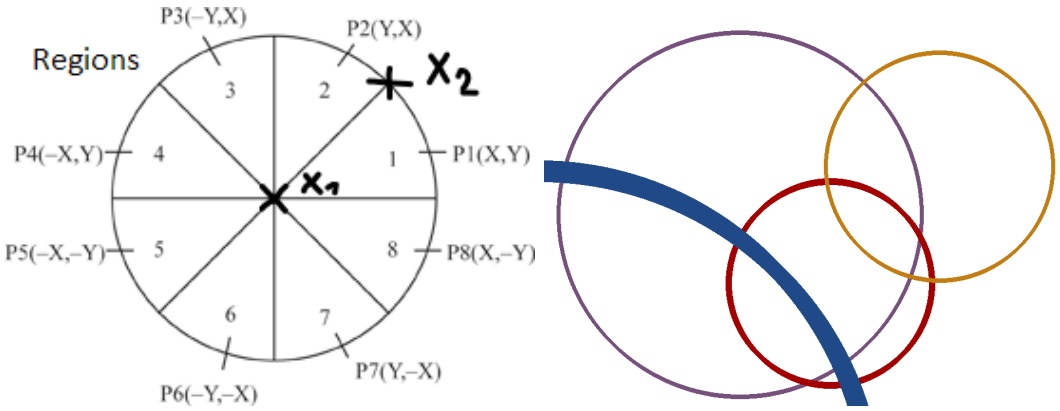


FIGURE 27 – Forme Circulaire de Bresenham avec couleur et épaisseur modifiable (Exemples X1 vers X2)

Formes 2D remplies

Les formes existent aussi de façon « remplies » mais les fonctions ne sont pas encore optimisées donc on ne peut pas encore accéder à cela depuis l'interface.

6.2.3 Les premiers filtres

Une des fonctionnalités « basiques » d'un logiciel de dessin/retouche photo est d'avoir quelques filtres simples pour changer les couleurs/tons de l'image rapidement et efficacement. Nous avons donc implémenté quelques filtres qui permettent de modifier les images que l'utilisateur a créé :

— **Filtre de nuance de gris** : le plus basique existant, les pixels de l'image passent tous par la formule « 0.3*r + 0.59*g + 0.11*b » pour avoir une image qui reste dans les mêmes tons mais en gris.

FIGURE 28 – Image de base

FIGURE 29 – Filtre gris

— **Filtre négatif** : les couleurs deviennent leur opposé. Il suffit pour cela, pour chaque pixel, de modifier sa couleur en les soustrayant à 255 (la valeur maximale).

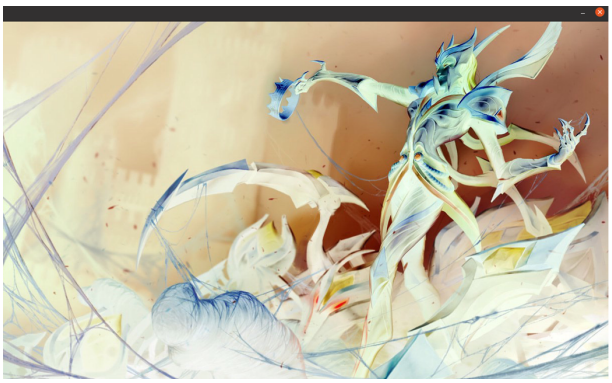


FIGURE 30 – Filtre négatif

— **Filtre de couleur customisable** : comme nous n'avions pas trop d'idée de filtre à faire sans que ça nous prenne trop de temps, nous avons créé un filtre avec plusieurs paramètres pour customiser le choix du rendu. Ce filtre prend en paramètre : la surface évidemment, un booléen constant pour chaque composante de la couleur pour savoir celles qu'on gardera, une valeur entre 0 et 255 pour savoir à quel seuil les composantes qu'on ne garde pas devront être. Par exemple, sur cette image on garde le rouge et on met le vert et le bleu à 0 :

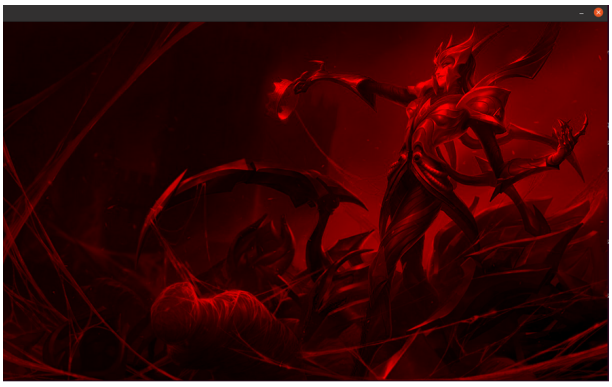


FIGURE 31 – Filtre rouge

Sur celle-ci, on garde le vert et le bleu et on fixe le rouge à 100 :

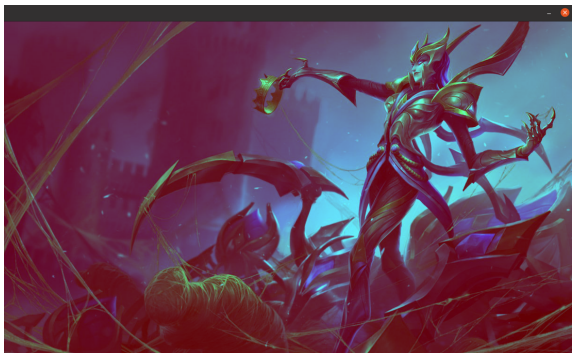


FIGURE 32 – Filtre "bleu-vert"

Et, évidemment, on peut cumuler ces filtres basiques pour faire des filtres plus complexes.

6.2.4 Transformation sur l'image

On a pu implémenter quelques transformations simples sur les images. Ces fonctionnalités ne sont pas encore toutes disponibles sur l'interface mais elles existent cependant dans notre code. On peut compter :

— **Le rognage** : l'utilisateur devrait choisir le x et le y du départ et de l'arrivée, pour effectuer le rognage sur toute la zone en dehors.

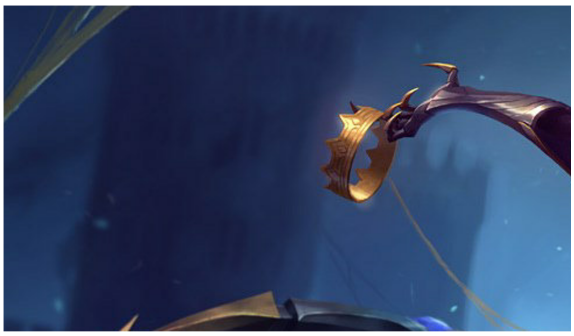


FIGURE 33 – Image rognée

— **L'inversion** : l'utilisateur pourra aussi choisir de faire une symétrie axiale de l'image (plus communément appelé « outil miroir ») horizontalement ou verticalement.

FIGURE 34 – Image inversée

— **La rotation** : L'utilisateur pourra simplement effectuer la symétrie centrale en choisissant l'angle qu'il voudra.

FIGURE 35 – Image tournée de 3 degrés

— **L'agrandissement/rétrécissement** : L'utilisateur pourra agrandir ou rétrécir l'image (n'a pas été retenu pour être inclus dans l'interface).

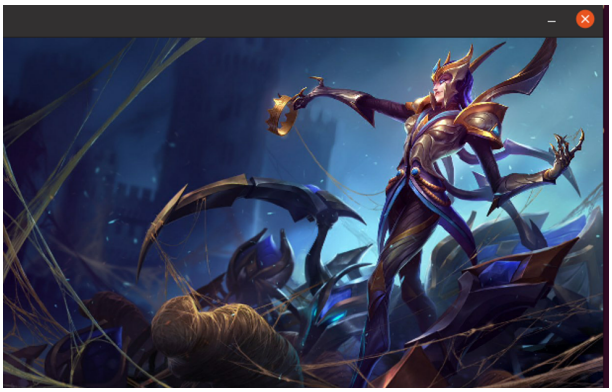


FIGURE 36 – Image rétrécie (elle est plus pixelisée)

6.3 Premières Interfaces - GTK

Premiers prototypes :

Nous avons commencé le travail à partir de rien, nous avions besoin d'un début d'interface pour implémenter les premières choses basiques du logiciel comme le fait de pouvoir charger une image et de la sauvegarder. Nous avons donc utilisé gtk 3.0 pour créer une simple interface répondant aux problème que nous avions. Le fait d'avoir travaillé sur gtk lors du projet d'OCR du semestre dernier m'a beaucoup servi dans ce cas. Je savais pertinemment que cette interface était simple voir même trop simple mais je savais aussi que nous allions en changer pour une version finale plus complète.

FIGURE 37 – Premier prototype d'interface

Ensuite, lorsque nous avons eu quelques outils de base prêts, Kaël a créé l'architecture de l'interface à l'aide de Glade, logiciel que nous avions découvert lors d'un TP de programmation du semestre dernier. Une fois ceci fait nous avons pu commencer à structurer le code de l'interface où nous avons commencé par lier les éléments de l'interface aux fonctions que les autres membres avait créés sur SDL.

FIGURE 38 – Prototype de l'interface graphique actuelle par Kaël et Alexandre

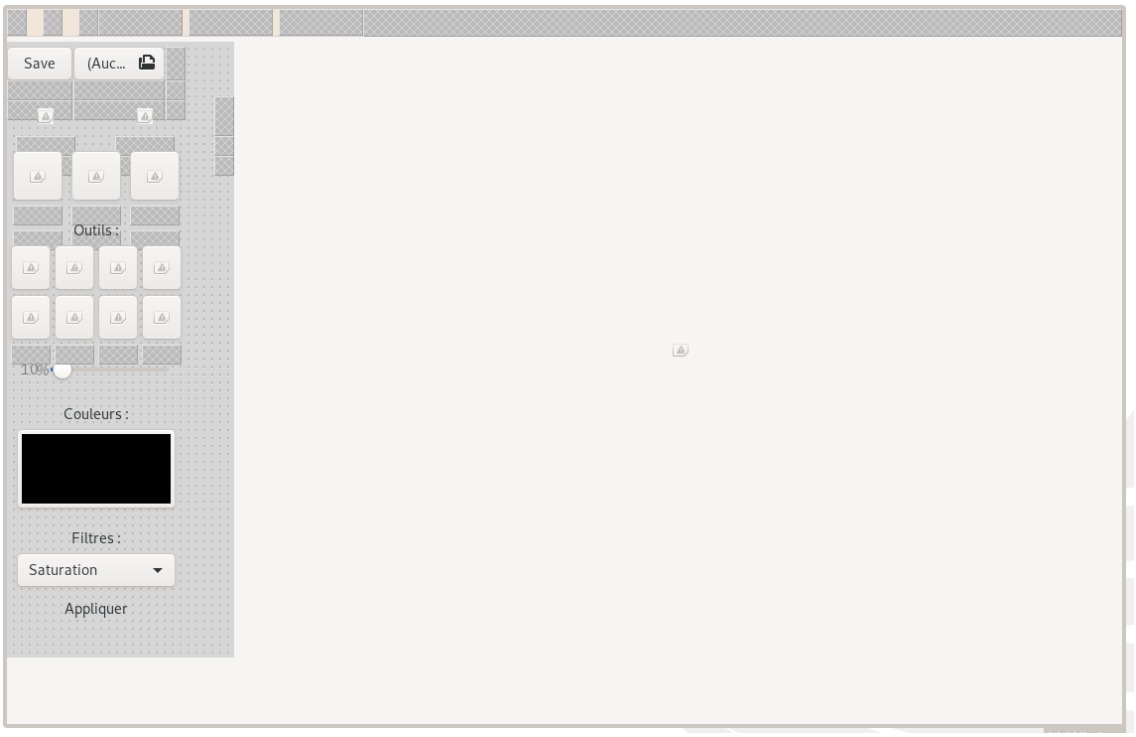


FIGURE 39 – Prototype de l'interface actuelle sur Glade

6.4 Ergonomie de l'interface

Notre interface est pensée pour être ergonomique à souhait !
En effet ce qui est le plus accessible se trouve au milieu de l'écran c'est pourquoi nous avons décidé d'y placer l'outil de sélection de couleur qui est, pour sûr, un élément majeure de n'importe quelle application de dessin ! Ce dernier nous permet également de séparer les outils classiques et les filtres, de manière à bien distinguer les deux !

Nous avons regroupé les boutons utilitaires permettant la gestion de l'image, les boutons de "Sauvegarde" et de "Chargement", les boutons "Retour arrière" et de "Retour avant" et les boutons "Zone de Sélection", "Zone de Texte" et "Rogner".

Situé en dessous de ces outils de gestion nous avons mis les outils de dessin classique ("Pinceau", "Seau", "Gomme") sur la première ligne et les outils géométriques ("Segment", "Rectangle", "Triangle", "Cercle") sur la seconde ligne.

On retrouve en dessous du bouton sélection de couleur un panneau déroulant correspondant à la zone de sélection des différents filtres. Une fois le filtre sélectionner il suffit d'appuyer sur le bouton "Appliquer", un jeu d'enfant !

6.5 Icônes de l'interface

Pour illustrer les différents outils de l'application il fallait créer des icônes les représentants. Pour rappel voici les différents outils pour l'instant présents dans l'application :

- Pinceau
- Seau
- Gomme
- Gomme d'annulation
- Segment
- Rectangle
- Triangle
- Cercle
- Rogner
- Zone de Sélection
- Zone de Texte
- Retour arrière/ Retour avant

Nous avons décidé de créer les icônes par nous même. Cela représentait deux avantages certains, éviter des problèmes de droit d'auteur tout en nous permettant d'avoir une unicité dans le style des icônes ! Pour le style des icônes nous avons opté pour des icônes de dimensions de 20x20 pixels sur fonds transparents, toutes avec un même schéma de couleur : l'outil en NoirBlanc et ce que l'outil nous permet de faire sur la zone de dessin en dégradé de couleur (rouge et bleu).

6.5.1 Implémentation des outils dans l'interface

Nous avons d'abord commencer par implémenter les 8 outils de bases dans l'interface, pour ce faire il nous a fallu trouver un moyen de gérer le clic et les coordonnées de la souris, nous savions que cela serait nécessaire, j'avais donc déjà créé un prototype de fonction s'occupant du clic simple. Maintenant que je savais ce dont j'avais besoin, nous avons créé une fonction qui met a jour en temps réel la position de la souris et l'enregistre à l'appui et au relâchement du clic. Ceci fait nous pouvions implémenter les outils, certains ayant simplement besoin des coordonnées de l'appui du clic par exemple le seau et certains ayant besoin des coordonnées de l'appui et du relâchement comme les formes.

FIGURE 40 – Outils

La deuxième chose que nous avons décidé d'implémenter est le bouton de sélection des couleurs et également le sélecteur permettant de choisir l'épaisseur des outils. Pour ce faire nous avons utilisé deux variables globales tenue à jour par des fonctions liées aux signaux respectifs du bouton et du sélecteur.

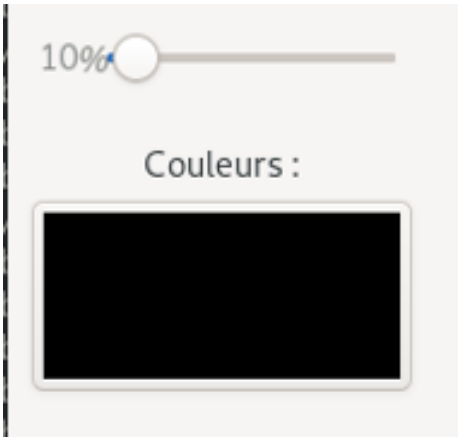


FIGURE 41 – Les outil en question

Le plus dur a été d'implémenter le pinceau à l'aide de la fonction traçant des segments entre deux points car nous n'avions aucune idée de comment faire. Pour implémenter cet outil j'ai d'abord eu beaucoup de mal et nous avons essayé de suivre des tutoriels mais finalement une idée simple et efficace nous est venue, simplement garder en mémoire la dernière position de la souris et la relier à la nouvelle par un segment lorsque l'outil du pinceau est choisi et que le clic est enfoncé. Implémenter ceci n'a pas été très compliqué mais l'outil du pinceau reste améliorable.

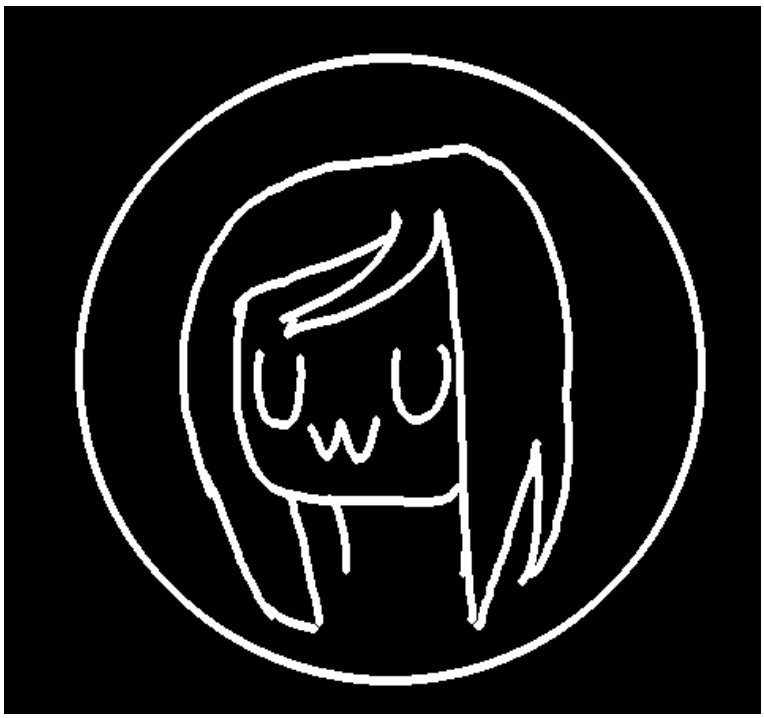


FIGURE 42 – Exemple des possibilités du pinceau

Pour savoir à tout moment quel outil est sélectionné nous avons décidé d'utiliser une variable globale qui serait modifiée à l'appui des boutons de sélection des outils.

6.5.2 Mise à jour de l'affichage

Nous avons eu aussi beaucoup de problèmes pour l'affichage, nous voulions d'abord utiliser une gtk image et une SDL Surface reliées par la fonction update qui utiliserais un fichier temporaire pour passer de gtk image à SDL Surface pour y appliquer les modifications souhaitées puis de nouveau faire la conversion dans le sens inverse pour mettre à jour l'affichage mais avec cette solution il était impossible de gérer la souris et ses déplacements et clics. Pour remédier à ça nous avons utilisé un autre module de gtk, la DrawingArea avec celui-ci nous avons pu efficacement gérer la souris à l'aide d'une Event Box gérant les signaux liés au déplacements et au clic de la souris. Il nous fallait donc juste une nouvelle fonction de mise à jour de l'affichage et pour ce faire j'ai remplacé celle associée au signal «draw» qui est appelée à chaque fois que la zone de dessin est affichée, ou que nous la déclenchons.

Dans cette fonction qui est une de celle qui m'a pris le plus de temps j'ai gardé le principe de fichier temporaire mais pour simplifier les choses nous avons fait le choix de créer une SDL Surface au lancement du logiciel que nous gardons jusqu'à sa fermeture. Toutes les modifications créées par les différents outils sont fait sur cette SDL Surface et ensuite nous appelons la fonction de mise a jour de l'affichage qui va faire le parallèle entre SDL et GTK grâce a un fichier temporaire unique.

6.6 Redimension de la fenêtre et centrage de l'image

Afin de rendre l'utilisation de l'application plus agréable nous avons fait en sorte que les dimensions et la disposition de la fenêtre s'adapte en fonction de l'image téléchargée.

Pour cela nous agrandissons la taille de la fenêtre si l'image téléchargée est plus grande que les dimensions 1280x850 pixels.

A l'inverse si l'image est plus petite que les dimensions précédemment citées, l'image sera recentrée au milieu de la zone de dessin.

6.6.1 Conclusion temporaire sur l'interface

Au final, pour cette première nous avions une interface qui permettait de bien utiliser les outils développés par les autres membres du groupe. Malgré quelques problèmes rencontrés lors du développement, comme celui de la gestion de la souris, du choix entre GTK Image et GTK Drawing Area, tous les problèmes on été réglés rapidement et de façon efficace.

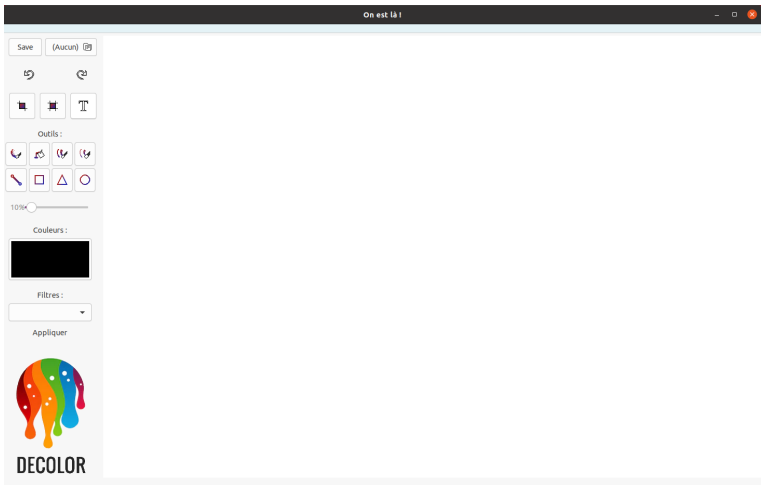


FIGURE 43 – Interface actuelle du logiciel

7 Avancement Chronologique (Soutenance 2) - Projet Decolor

7.1 Interface et ergonomie

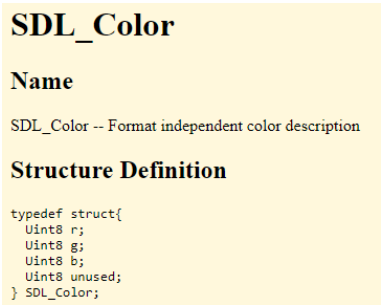
7.1.1 Raccourcis claviers

Dans le but de peaufiner l'ergonomie du logiciel, nous avons réfléchi à ce que nous pourrions ajouter. Nous avons donc décidé qu'à l'image des logiciels classiques de ce genre nous allions implémenter des raccourcis clavier pour pouvoir annuler et de nouveau appliquer des modification faite plus rapidement. Pour ce faire nous avons utilisé GTK qui possède une gestion du clavier intégrée, il existe un événement qui, lorsqu'il est déclenché, permet de récupérer l'identifiant de la touche pressée. Nous utilisons également une autre fonctionnalité de GTK : les masques qui nous permettent entre autre de gérer la souris mais également certaines touches du clavier spécifiques comme shift ou ctrl. Cela nous permet donc de détecter l'appui de n'importe quelle touche et de savoir si l'une des touches spéciales est appuyée en même temps, si c'est le cas alors on appelle la fonction associée au raccourci.

Cette fonctionnalité n'était pas parmi les plus urgentes à implémenter mais le logiciel étant dans un stade assez satisfaisant et le temps restant étant plutôt court nous avons décidé de commencer par ça au niveau de l'ergonomie. Désormais l'interface est assez complète et nous ne pouvons pas nous permettre de la surcharger de boutons et de fonctionnalités, pour cette raison les raccourcis clavier peuvent en effet s'avérer primordiaux. Tout les logiciels de dessins basent une énorme partie de leur ergonomie d'utilisation sur les raccourcis clavier

7.1.2 Fonctionnalités Abandonnées

Nous avions en tête d'ajouter la gestion de l'opacité au projet malheureusement à cause de certains choix que nous avions fait à la première soutenance. Cela était trop compliqué pour être faisable sans tout recommencer et nous n'avions pas du tout le temps de l'implémenter entièrement. En effet, plus tôt dans le projet nous avions choisi de garder SDL pour nos modifications d'image et que nous ne sommes pas passés à cairo pour interagir directement avec la zone de dessin. Notre version de SDL ne nous permettant pas de gérer la composante alpha sur nos couleurs nous a totalement empêché de développer cette fonctionnalité.



SDL_Color

Name

SDL_Color -- Format independent color description

Structure Definition

```
typedef struct{
  Uint8 r;
  Uint8 g;
  Uint8 b;
  Uint8 unused;
} SDL_Color;
```

FIGURE 44 – Structure SDL Color dans notre version

7.1.3 Nouvelles fonctionnalités

Avec l'arrivée de nombreuses nouvelles fonctionnalités concernant le déplacement de l'image, ça rotation et aussi le redimensionnement de la zone de dessin il a fallu réfléchir a comment nous pouvions intégrer toutes ces fonctionnalités sans surcharger l'interface et la rendre illisible ou difficile a utiliser. Nous avons

donc après mures réflexions fini avec une interface assez chargée mais tout a fait lisible et utilisable simplement.

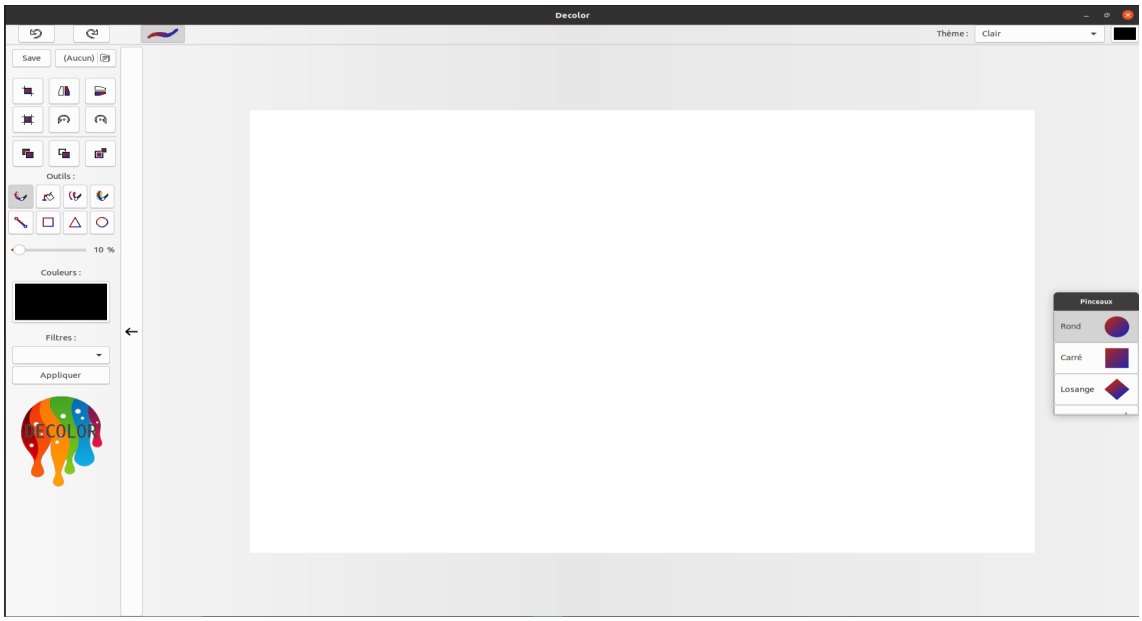


FIGURE 45 – Interface finale

Nous avons commencé par placer les nouveaux boutons dans l'interface pour accueillir les nouvelles fonctions. Nous les avons placés arbitrairement au dessus des huit premier outils.

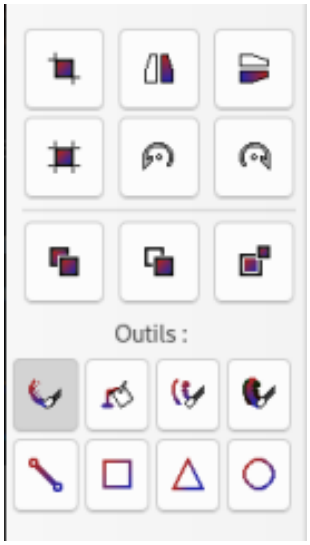


FIGURE 46 – Nouveaux boutons

Pour certaines des fonctionnalités il a également fallu implémenter de nouvelles fenêtre surgissante (pop-up). Dans le cas de la fonction de redimensionnement nous avions besoin d'une fenêtre pour récupérer les nouvelles dimensions de l'image auprès de l'utilisateur de façon efficace. Pour créer cette fenêtre pop-up nous avons utilisé un GTK Dialog, il nous permet de lancer une fenêtre customisable avec deux zones utilisables, une zone de contenu où l'on peut mettre n'importe quel type d'élément cela peut aller des

boutons aux entrées de texte ou simplement à du texte pour afficher un message et une zone d'action qui ne peut contenir que des boutons majoritairement utilisés pour valider un choix ou fermer la fenêtre en annulant l'action. La fenêtre basique créée pour la fonction de redimensionnement a, plus tard, été largement améliorée et de nouvelles fonctionnalités lui ont été ajoutées, comme le choix du placement de l'ancienne image dans la nouvelle zone agrandie.

FIGURE 47 – Fenêtre surgissante de redimensionnement

Pour rendre l'utilisation plus facile nous avons également corrigé quelques problèmes dont le fait qu'avant il fallait forcément rajouter l'extension .bmp au nom du fichier que nous étions entrain de sauvegarder, aujourd'hui l'extension est forcément présente au moment de la sauvegarde. Au final nous pensons avoir réussi à faire une interface assez aérée et agréable à utiliser mais jusqu'à maintenant nous avons parlé que du coté ergonomique de l'interface mais une autre majeure partie du travail a été concentrée sur l'aspect et le visuel de l'interface.

7.1.4 Apparence et thème CSS

L'ajout de ces nouvelles fonctionnalités a entraîné la création de nouveaux boutons et donc de nouvelles icônes !
Pour les nouvelles icônes nous sommes restés sur le même style graphique afin de garder une unicité dans l'application. Les icônes ont donc un format de 20x20 pixels sur fond transparents, toutes avec un même schéma de couleur : ce qui touche à l'image en dégradé de couleur (rouge et bleu) et le reste en noir et blanc.

Voici une liste exhaustive des nouveaux ajouts :

- Copier
- Coller
- Couper
- Flèche de rotation droite
- Flèche de rotation gauche
- Cacher/Montrer
- Pinceaux :
  - Carré
  - Etoile
  - Losange
  - Manuscrit
  - Rond
  - Trait horizontale
  - Trait verticale
- Position :
  - Nord-Ouest
  - Nord
  - Nord-Est
  - Ouest
  - Centre
  - Est
  - Sud-Ouest
  - Sud
  - Sud-Est

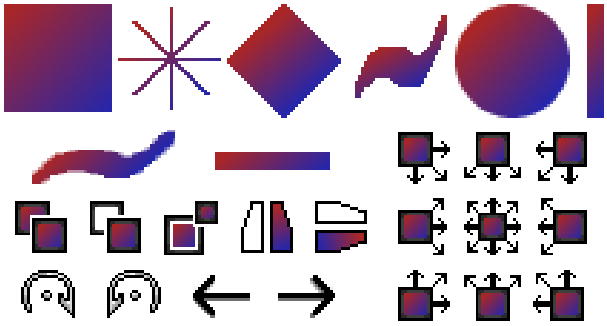


FIGURE 48 – Nouvelles icônes

-CSS

Pour la beauté de l'application et le confort de l'utilisateur nous avons créé plusieurs thèmes de couleurs. Nous avons donc créé 4 thèmes de base qui sont classiques mais néanmoins indispensables !
Et également 2 autres thèmes personnalisables par l'utilisateur dans l'application !

-Thème de base

Pour les thèmes de base nous retrouvons donc 4 thèmes ; Clair, Sombre, Clair contrasté, Sombre contrasté.
Le thème **Clair** est directement inspiré du thème clair par défaut des applications gtk.
Nous avons donc essayé de reproduire ce thème de base le plus fidèlement !

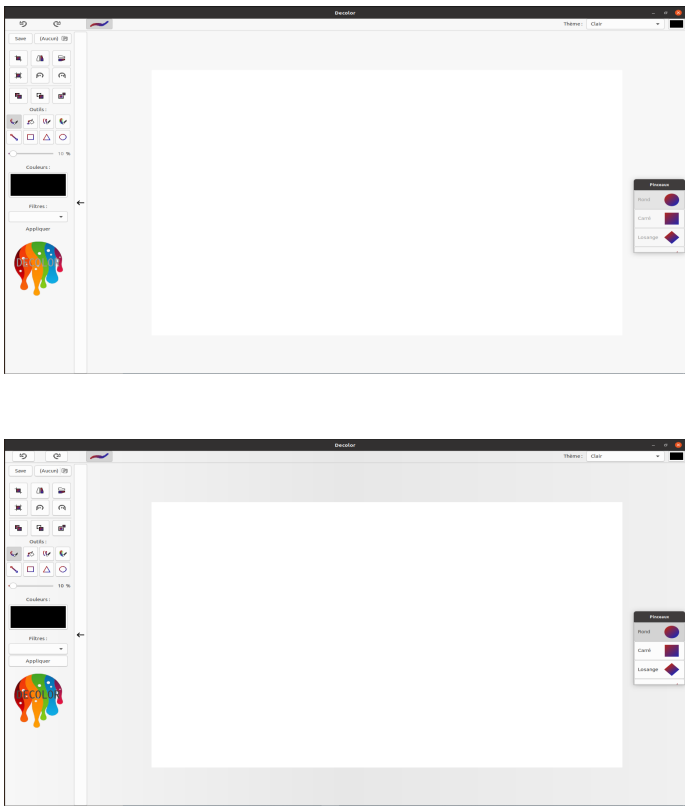


FIGURE 49 – Thème par défaut au dessus // Thème reproduit en dessous

Le thème **Sombre** est tout simplement à l'opposé du thème clair !

FIGURE 50 – Interface avec le thème sombre

Les thèmes **Contrastés** sont semblables aux thèmes normaux à la différence près que les différences entre les couleurs sont plus marquées, les boutons ressortent donc mieux avec des contours plus nettes.

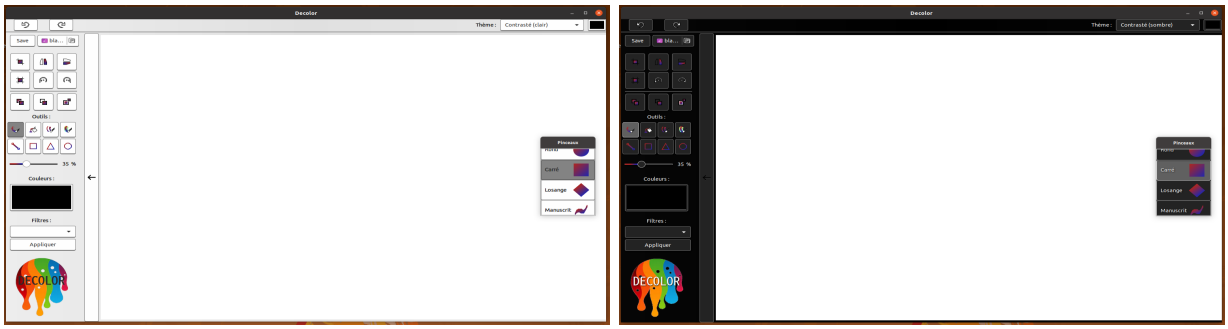


FIGURE 51 – Thème contrasté clair à gauche // Thème contrasté sombre à droite

-Thème personnalisé

Pour les thèmes personnalisés, c'est un poil plus complexe !
En effet, le principe est donc d'afficher un thème correspondant à la couleur sélectionnée par l'utilisateur ! Il faut donc dans un premier temps déterminer si la couleur choisie correspond à une couleur claire ou sombre. Pour cela il suffit de calculer son niveau de gris, de ce dernier nous pourrons déterminer quel fichier CSS utiliser et quels calculs appliquer.

Prenons un **vert clair**, pour exemple la couleur de code hexadécimal **#81F584**.
Son niveau de gris est de **192** (#C5) cette couleur est au dessus du palier fixé, nous allons donc modifier et charger le fichier CSS correspondant aux couleurs claires.

Nous allons donc calculer 4 déclinaisons de la couleur, une couleur plus claire et 3 plus sombres, ainsi qu'un niveau de gris pour le texte :

Couleur = **#81F584** px `clair/bg_color.png`
Couleur plus claire = Couleur * 110 % + 10 = **#97FF9B** px `clair/fg_color.png`
Couleur plus sombre1 = Couleur * 90 % = **#74DC76** px `clair/dark_fg_color.png`
Couleur plus sombre2 = Couleur * 80 % = **#67C469** px `clair/dark_bg_color.png`
Couleur plus sombre3 = Couleur * 75 % = **#60B763** px `clair/bf_last_color.png`
Couleur texte = Niveau_de_gris(Couleur * 25 % - 10) = **#272727** px `clair/text_color.png`

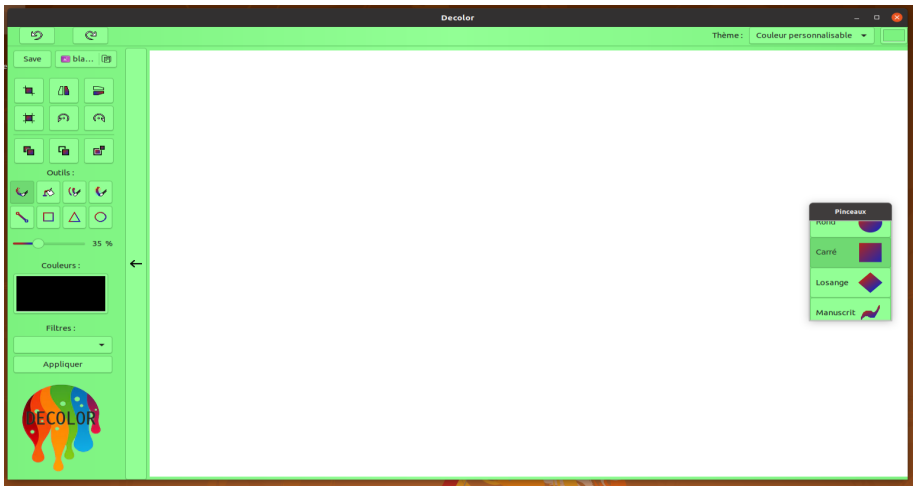


FIGURE 52 – Thème coloré clair

Prenons un **bleu foncé**, pour exemple la couleur de code hexadécimal **#091891**.
Son niveau de gris est de **33** (#21) cette couleur est en dessous du palier fixé, nous allons donc modifier et charger le fichier CSS correspondant aux couleurs sombres.

Nous allons donc calculer 4 déclinaisons de la couleur, une couleur plus sombre et 3 plus claires, ainsi qu'un niveau de gris pour le texte :

Couleur = **#091891** px `sombre/bg_color.png`
Couleur plus sombre = Couleur * 80 % = **#071274** px `sombre/dark_bg_color.png`
Couleur plus claire1 = Couleur * 300 % + 30 = **#3966FF** px `sombre/bf_last_color.png`
Couleur plus claire2 = Couleur *250 % + 15 = **#254BFF** px `sombre/fg_color.png`
Couleur plus claire3 = Couleur *110 % + 50 = **#3B4CD1** px `sombre/dark_fg_color.png`
Couleur texte = Niveau_de_gris(Couleur * 350 % + 140) = **#FEFEFE** px `sombre/text_color.png`

FIGURE 53 – Thème coloré sombre

L'utilisateur peut sélectionner les différents thèmes dans onglet se situant en haut à droite de l'interface, on y retrouve également une zone pour choisir la couleur du thème personnalisé.

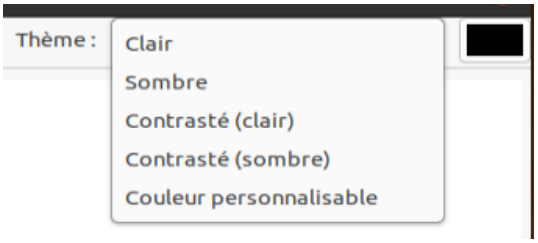


FIGURE 54 – Zone de sélection du thème de l'interface

Pour aérer et embellir l'interface nous avons également rajouter un bouton permettant de rétracter la barre d'outils.
Quand bien même cette dernière est fermée, les options de retour en arrière, de retour en avant ainsi que de sélection des formes des pinceaux restent disponibles. Cela évite à l'utilisateur de devoir ouvrir et refermer l'onglet des outils trop souvent !

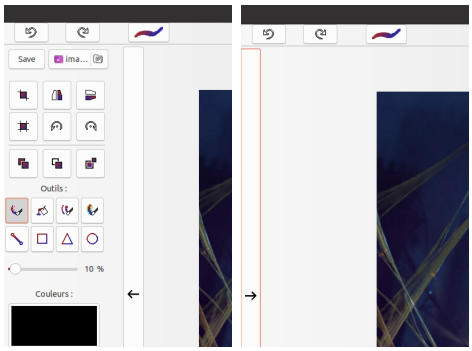


FIGURE 55 – Barre d'outils affichée // Barre d'outils rétractée

Par ailleurs le centrage de l'image a été retravaillé afin de prendre en compte plus de cas notamment celui du plein écran et de la rétractation de la barre d'outils

7.2 Nouveaux outils et modification des anciens

7.2.1 Le crayon

Tout d'abord, nous avons ajouté des crayons différents. Pour la 1ere soutenance, le crayon avait une forme de carré pas très belle à voir, alors nous avons d'abord modifié cette forme en disque puis nous avons créé beaucoup d'autres crayon que l'utilisateur peut sélectionner. On compte parmi ces crayons : le rond, le carré, le losange, le manuscrit (un crayon en forme de segment un peu penché pour la calligraphie), l'étoile, le trait vertical et le trait horizontal. Ces choix de crayon modifient aussi les formes !

Évidemment l'utilisation de chaque crayon varie en fonction de la taille et de la couleur sélectionnée.

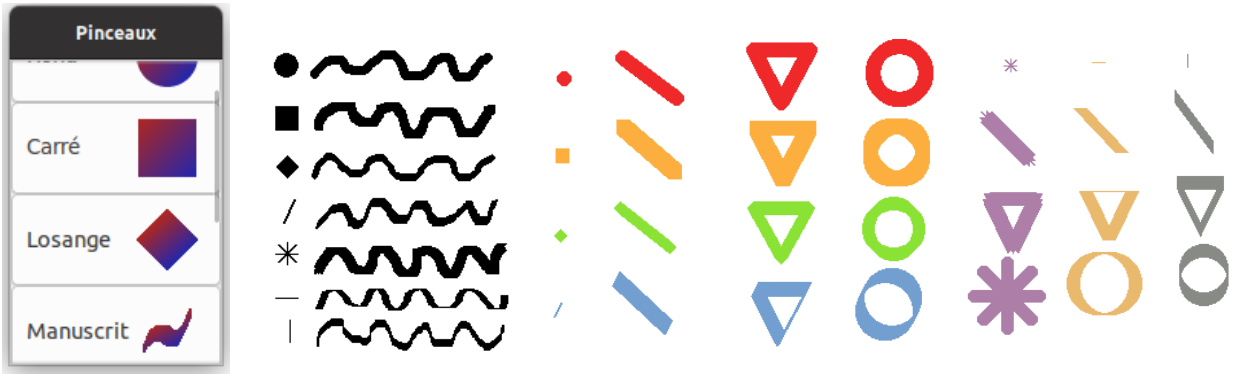


FIGURE 56 – Types de pinceaux, leurs tracés, et les différentes formes qu'ils génèrent

7.2.2 Les outils de transformations de l'image

Nous avions déjà codé l'inversion et le zoom pour la première soutenance mais ils n'étaient pas encore disponibles sur l'interface. Nous avons donc ajouté cet outil à l'interface en enlevant le zoom qui rendait l'image trop pixélisée et donc qui n'apportait aucune utilité au logiciel.

FIGURE 57 – Boutons Inversion

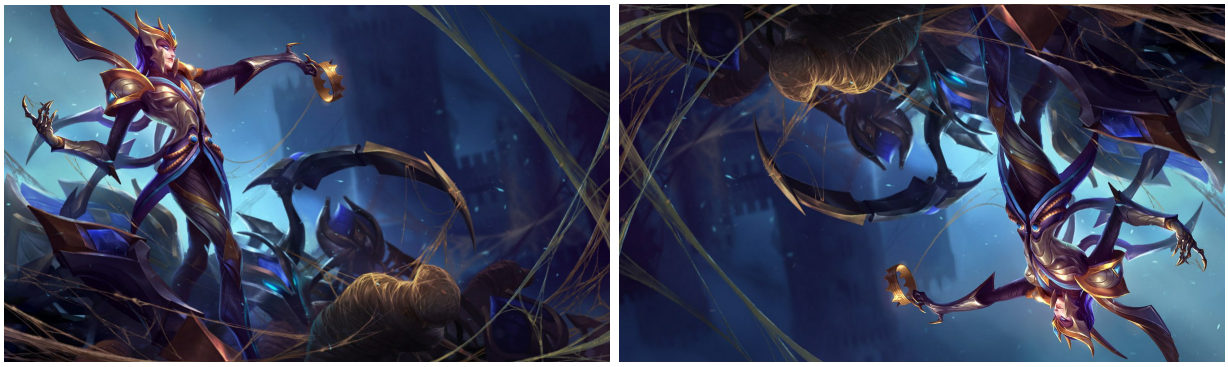


FIGURE 58 – Inversion de l'image horizontalement et verticalement

Nous avons ensuite ajouté quelques outils pratiques :

— **Rotations** :

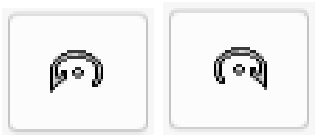


FIGURE 59 – Boutons Rotation

Pour cette fonctionnalité, nous voulions tout d'abord utiliser la fonction d'une librairie directement disponible (rotozoomSurface), sauf qu'il s'est avéré qu'elle mettait un cadre noir non voulu aux bords de l'image. Le choix a donc été fait de créer nos deux propres fonctions pour tourner une image de 90 degrés vers la gauche et 90 degrés vers la droite. Des boutons ont ensuite été ajoutés sur l'interface pour pouvoir accéder à ces deux nouvelles fonctionnalités.

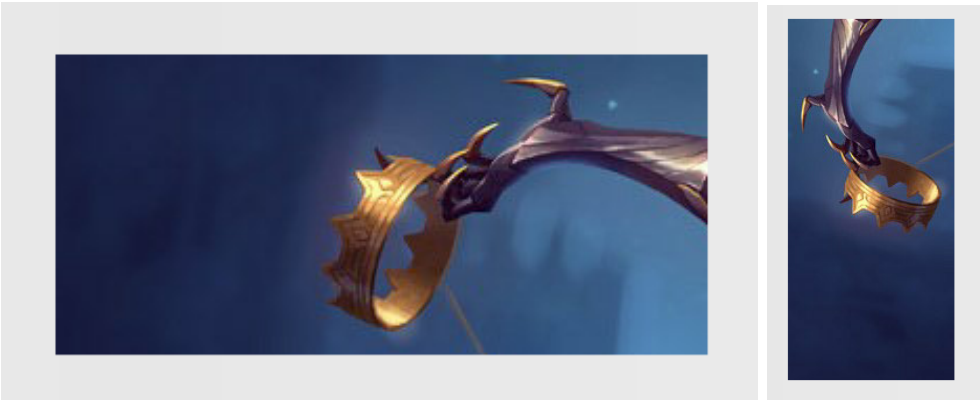


FIGURE 60 – Rotation de 90 degrés vers la gauche (Résultat rétrécit pour rentrer dans la page)

— **Rogner l'image (Crop)** :

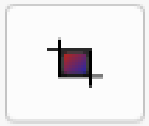


FIGURE 61 – Bouton Rogner

Afin d'intégrer cette nouvelle fonctionnalité clé de notre projet, nous avons utilisé les coordonnées de la souris au clic puis au relâchement du clic avec une vue en temps réel du cadre qui sera la nouvelle image avec un rectangle qui apparaît en tant que pré-visualisation. Une fois le clic relâché, une fonction s'occupe de récupérer des coordonnées valides pour l'image (les coordonnées pourraient être négatives ou dépasser le cadre de l'image) qui sont passés par la suite à notre fonction qui copie la partie de l'image sélectionnée dans une nouvelle image qui est ensuite affichée.

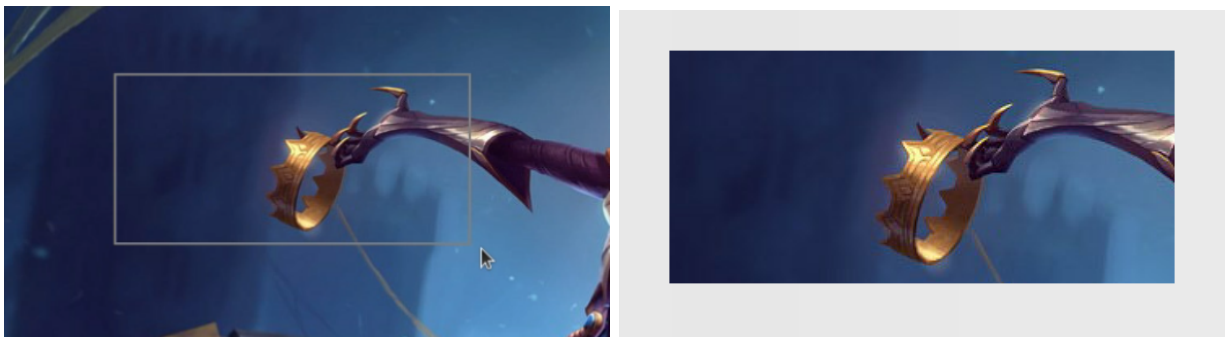


FIGURE 62 – Rétrécissement de l'image en fonction de la zone sélectionnée sur l'image de départ

— **Redimensionner l'image** :

FIGURE 63 – Bouton Redimensionner

Cette nouvelle fonctionnalité a été un ajout capital pour le logiciel. Une fois l'outil cliqué, comme précédemment expliqué dans la partie GTK, une fenêtre surgissante (pop-up) s'ouvre nous demandant les dimensions et le sens de de la nouvelle image voulue. La fonction liée à cette fonctionnalité comporte donc 9 cas possibles. Il est aussi important de préciser que l'on peut, si souhaité, étendre seulement la hauteur et/ou la largeur de l'image comme rétrécir la hauteur et/ou la largeur de l'image en même temps.

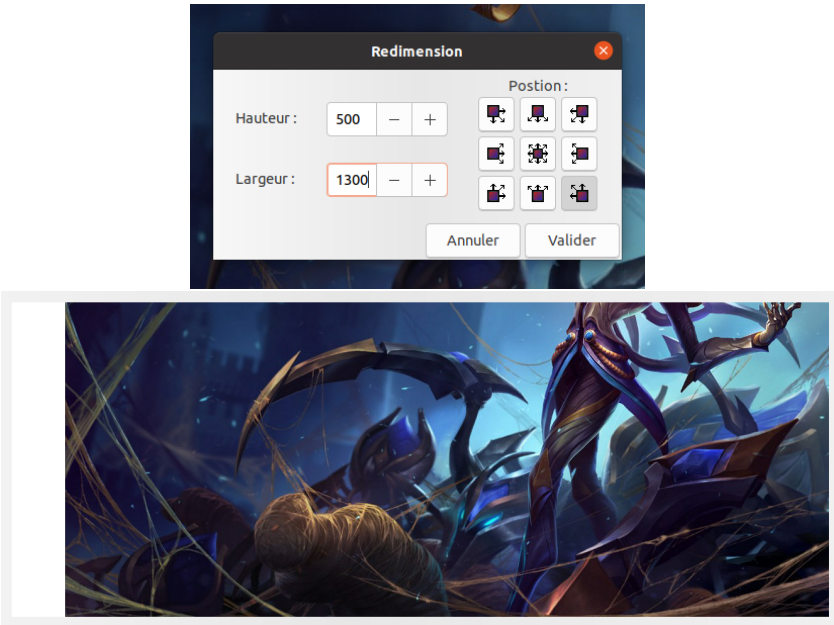


FIGURE 64 – Paramètres du redimensionnement de l'image voulu et résultat obtenu (Rétrécissement de la hauteur + agrandissement de la largeur + centrage de l'image de départ en bas à droite de la nouvelle)

— **Copier** :

FIGURE 65 – Bouton Copie

Pour continuer, nous avons ajouté un bouton afin de copier une zone sélectionnée de l'image. Une fois l'outil cliqué, nous devons juste sélectionner une partie de l'image que nous pouvons voir à l'aide de la pré-visualisation de la zone voulue. Par la suite une copie de cette zone sera enregistrée dans le logiciel et permettra à l'outil "Coller" de fonctionner correctement.

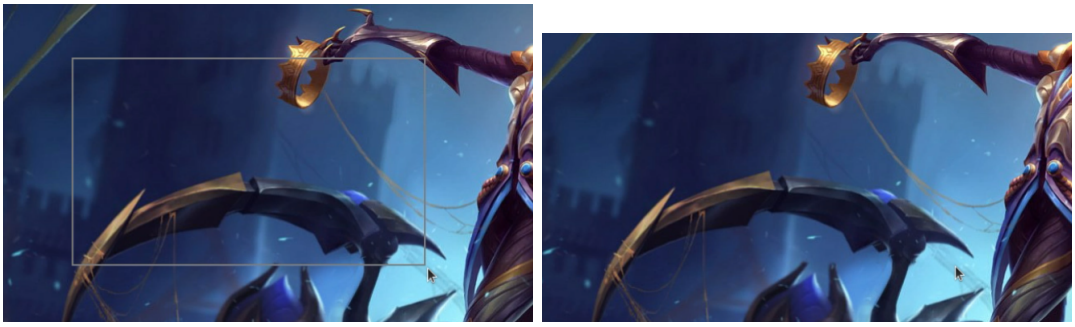


FIGURE 66 – La zone sélectionnée est copiée et aucun changement sur l'image de base n'a lieu

— **Couper** :

FIGURE 67 – Bouton Couper

Ce nouvel outil fonctionne en utilisant le même principe que l'outil "Copier". Cependant, à la différence de celui-ci, la zone sélectionnée est supprimée, c'est à dire qu'elle est effacée (remplacée par du blanc). L'exemple ci-dessous est utile pour mieux comprendre :

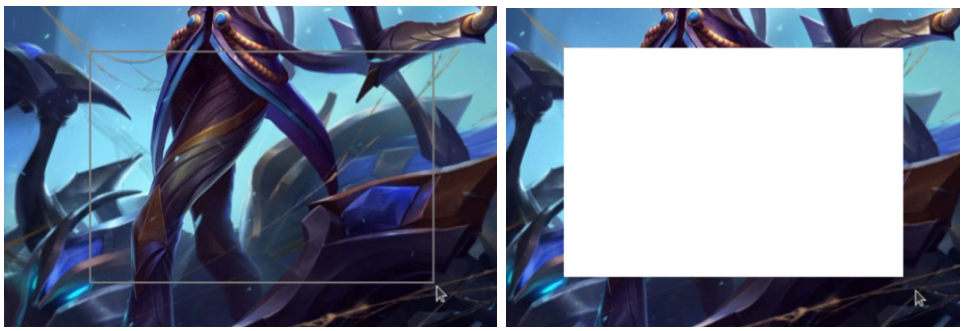


FIGURE 68 – La zone sélectionnée est copiée et est effacée sur l'image

— **Coller** :

FIGURE 69 – Bouton Coller

Cette nouvelle fonctionnalité est un complément des outils "Copier" et "Couper" qui lui permettent de fonctionner correctement. L'image qui est copiée et donc enregistrée dans le logiciel après l'utilisation de ces deux outils peut être posée sur l'image à n'importe quel moment et par 2 moyens différents. Tout d'abord, on retrouve le moyen du bouton qui appelle une fonction pour coller la copie n'importe où sur l'image à chaque pression du clic puis le second moyen consiste en l'appui en simultané des touches "ctrl" et "v" qui collera l'image copiée à l'endroit où se trouve la souris sur l'image.

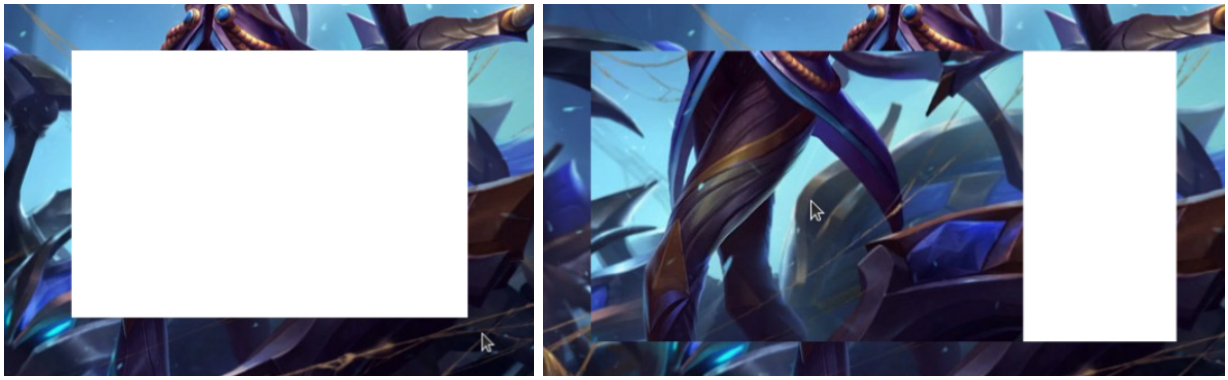


FIGURE 70 – On reprend l'image précédemment coupée, et on colle à coté (à gauche) ce qui a été copié

7.2.3 Les filtres

Nous avons ajouté beaucoup de filtres car nous nous sommes rendu compte que c'est une façon simple et efficace d'améliorer notre logiciel. Cela ajoute un aspect "modification d'image" au logiciel car bien évidemment l'utilisateur peut superposer les filtres pour les combiner. Nous avions codé au début 3 filtres : nuance de gris, négatif et un filtre "factice" qui permettait de créer d'autres filtres sans avoir besoin de faire de calculs. Puis nous nous sommes justement plus penchés dans les quelques calculs mathématiques sur les pixels que nous devions faire pour avoir des filtres jolis. Nous avons pu créer :

— **Le filtre luminosité** : il suffit d'ajouter un valeur n positive ou négative à chaque composante de chaque pixel (sans que ça ne dépasse 0 ou 255) pour rendre l'image plus claire ou plus sombre.

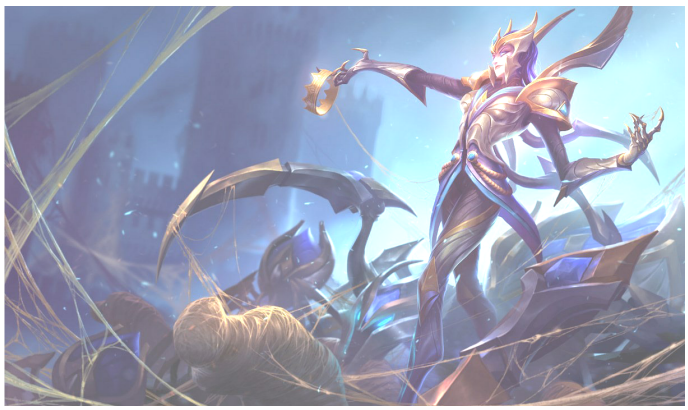


FIGURE 71 – Filtre luminosité avec n = 50

— **Le filtre contraste** : pour ce filtre, il faut rendre les pixels clairs encore plus clairs et les pixels sombres encore plus sombres en fonction d'un facteur donné. On applique donc une fonction qui se découpe en 2 parties : soit x la composante du pixel et n un facteur donné,

$$x < \frac{255}{2}, f(x) = \frac{255}{2} * (\frac{2x}{255})^n$$

.

$$x > \frac{255}{2}, f(x) = 255 - \frac{255}{2} * (\frac{2(255-x)}{255})^n$$

FIGURE 72 – Filtre contraste avec n = 3

— **Le filtre de couleur** : il suffit de faire la nuance de gris du pixel puis de la multiplier au pourcentage de chaque composante du pixel. C'est à dire que pour la composante rouge d'un pixel par exemple, soit rx, gx, bx les composantes d'une couleur x, alors la composante deviendra :
(r / 255 * gris, g / 255 * gris, b / 255 * gris) avec gris = 0.3*rx + 0.59*gx + 0.11*bx

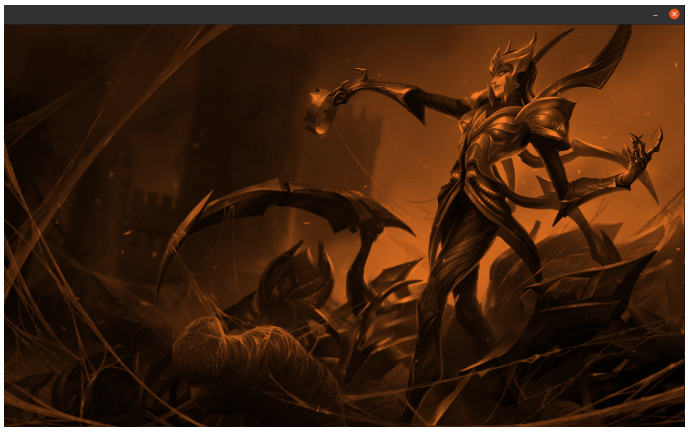


FIGURE 73 – Filtre obtenu en sélectionnant du orange

— **Le filtre flou** : nous avons dû utiliser le flou gaussien pour ce filtre. Le principe est simple, il faut, pour chaque pixel, faire la moyenne de tous les pixels à n distance de lui.

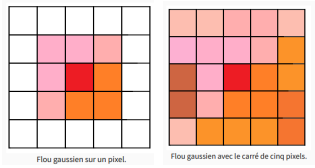


FIGURE 74 – Principe du flou gaussien sur un pixel puis sur cinq pixels

Si n est trop grand, le flou est évidemment très long à s'exécuter et on peut le remarquer sur n'importe quel logiciel, nous l'avons donc bloqué à 5.

FIGURE 75 – Filtre de flou gaussien avec n = 5

— **Le filtre Détourage** : nous avons ici aussi utilisé le principe du flou gaussien pour ce filtre. Le principe est simple, il faut, pour chaque pixel, réaliser le calcul suivant :

255 - | la valeur du pixel - la moyenne de tous les pixels à n distance de lui |.

Comme pour le flou, si n est trop grand le filtre est trop long à s'éxecuter donc nous l'avons aussi bloqué à 5.

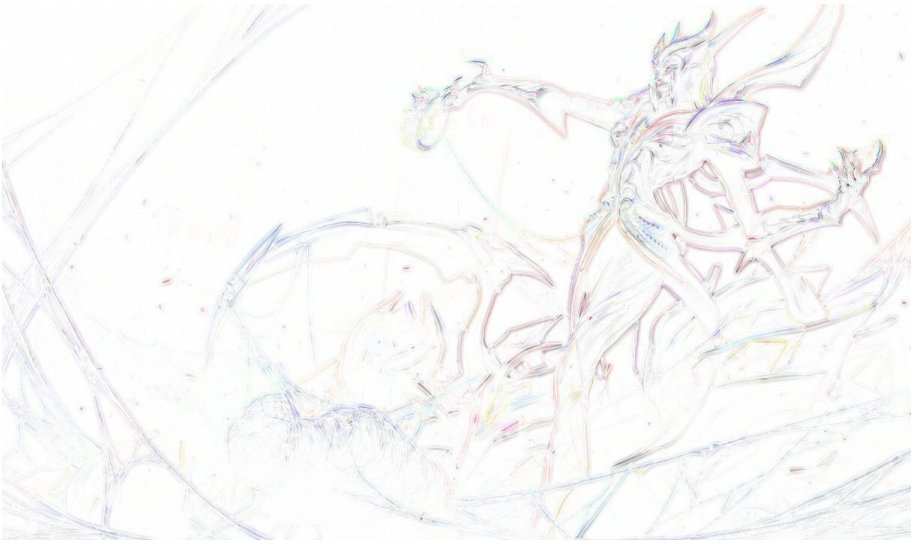


FIGURE 76 – Filtre de détourage avec n = 5

7.2.4 La pré-visualisation des formes et des outils

Afin de pouvoir mieux se repérer pour l'utilisation des formes 2D, nous avons mis en place un système qui montre en temps réel à l'utilisateur, tant qu'il maintient le clic de la souris, le rendu finale de l'image s'il relâchait le clic. Cet ajout est vraiment très utile notamment pour l'édition rapide et facile de l'image. Certains outils nécessitants la sélection d'une partie de l'image par l'utilisateur utilisent aussi la pré-visualisation qui prend la forme d'un fin rectangle gris. Avoir un aperçu de la zone sélectionnée permet d'éviter toute erreur et fait gagner en précision. On la retrouve pour les outils : "Rognage", "Copier" et "Couper".

FIGURE 77 – Pré-visualisation (ici pour poser un cercle)

7.2.5 Fonctionnalité abandonnée

Lors de la seconde soutenance, nous avons voulu implémenter une grosse fonctionnalité sur notre logiciel : **les calques**. Nous avons créé une structure de liste chaînée qui permettait de lier chaque calque avec le précédent et une sentinelle au début de la liste qui compte tous les calques créés. Nous avions ainsi une structure fonctionnelle qui permettait de créer les fonctions basiques du calque simplement.

Malheureusement nous nous sommes rapidement rendu compte d'un gros problème d'optimisation : en effet, à cause des bibliothèques que nous utilisons pour le projet, la preview de ce que l'utilisateur est en train de faire aurait été difficilement possible car il aurait fallu 3 surfaces au lieu d'une (une pour les calques derrière, une pour celui sur lequel l'utilisateur dessine et une devant) et le crayon n'étant déjà pas parfaitement fluide avec une seule surface, nous ne pouvions pas nous permette d'en ajouter 2 supplémentaires.

Nous avons donc abandonné cette partie du projet qui n'était de toute façon pas prévue à la base pour nous concentrer davantage sur l'amélioration des autres outils.

Nous avons aussi abandonné l'option de pouvoir **écrire du texte** sur nos images par manque de fonctionnalités sur SDL où nos images sont modifiées.

8 Site internet du logiciel

Le site Web de notre projet est avant tout le site Web de notre projet (OELA - Decolor). Ceux-ci sont respectivement représentés au travers de différents articles ainsi que liens de téléchargements et autres interactions dans le but d'attirer les visiteurs à télécharger l'application.

La structure du site Web est entièrement terminée mais sera mis à jour suite aux nouveaux ajouts.

Voici le lien pour aller le consulter, **`https://topagrume.github.io/decolor_web/accueil.html`**

Possédant déjà un compte *GitHub*, nous nous sommes tournés vers ce support pour héberger le site et le mettre à jour automatiquement au fil des nouvelles versions envoyées vers la sauvegarde distante. A l'aide de nos connaissances en HTML (pour la mise en forme des pages), CSS (pour le style et l'apparence des pages) et JavaScript (pour ce qui est du code à exécuter côté client surtout les effets dans notre cas), des forums en ligne ainsi que d'exemple d'organisation, de répartitions du texte, d'effets et d'autres outils disponibles librement sur Bootstrap et d'autres espaces de libre-échange, nous avons pu réaliser ce site.

Nous nous sommes basés sur le thème de notre logo Decolor pour l'apparence du site qui allie des couleurs vives.Le langage CSS n'étant pas toujours évident, cela est passé par beaucoup de tâtonnements, de découvertes puis aussi d'exploration de documents sur Internet, mais nous sommes aujourd'hui très satisfaits du résultat qui s'adapte de plus à de nombreuses tailles d'écran mis à part l'arbre d'historique de notre projet avec les écrans de téléphones. La navigation est alors agréable, intuitive et plus aisée.

8.1 La structure du site

Il a tout d'abord fallu réfléchir à la structure du site, son arborescence, l'organisation de chacune des pages, la répartition des images, du texte, des liens, d'un historique de projet, de texte et de statistiques. Nous avons décidé de choisir une structure simple composée d'un en-tête dans lequel on retrouve le nom de notre groupe, le nom et le lien vers les différentes pages du site : (Projet, Avancées, Documentation, Équipe) ainsi qu'un bouton « télécharger ». Puis dans la suite de la structure un corps de texte qui laisse libre recours à nos choix et explications et enfin un pied de page destiné à n'être qu'un simple décor.

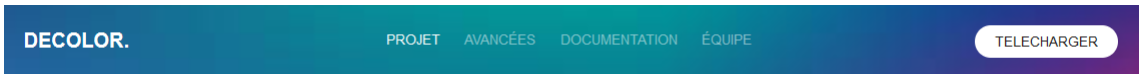


FIGURE 78 – Structure du site

8.2 Page d'accueil

Bien en évidence, nous avons opté pour directement placer un bouton afin de renvoyer vers un lien de téléchargement du logiciel en version normale/lite. Dans le corps de la page, nous avons opté pour une présentation brève de notre logiciel, et de ses qualités, répartie en différents onglets mis en parallèles avec des images représentatives, puis de nos valeurs en tant que développeur, une présentation brève de l'intérêt algorithmique et de son fonctionnement et enfin un petit paragraphe sur notre politique ainsi que le choix de notre Logo.

Plus bas, il nous a semblé important présenter les différents points d'avancement par des images représentatives explicites.

Enfin en pied de page, nous avons implémenté des statiques à titre purement informatif et esthétique sur le nombre de nos commit, d'ajouts réussies, d'étapes réussies et enfin le nombre d'heures qui ont été dédiées au projet dans son ensemble.

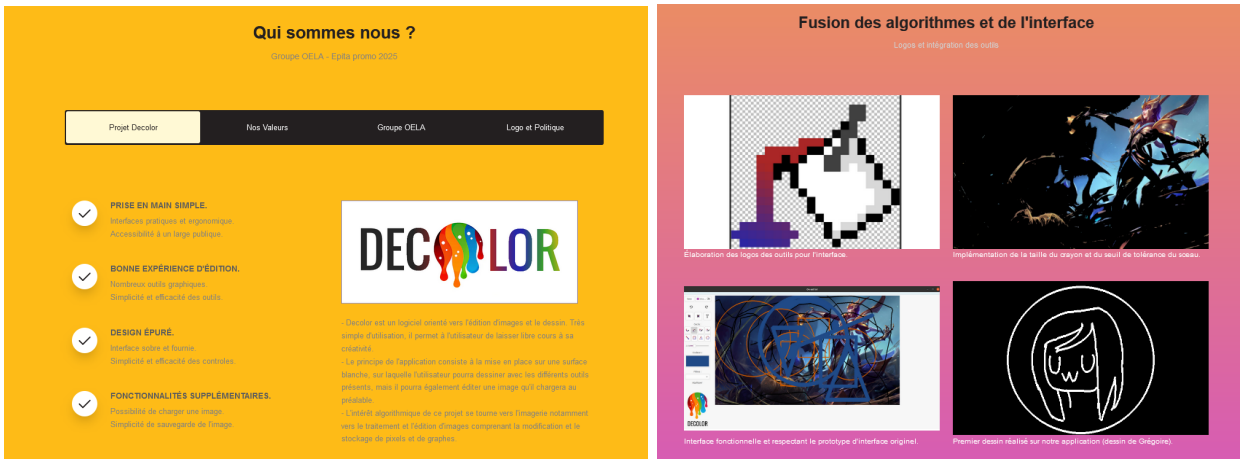


FIGURE 79 – Présentation du projet et des ajouts

8.3 Page d'avancements

Nous avons ici opté pour présenter les différentes étapes clés du projet passées tout au long de notre semestre (mars à juin) avec notamment de nombreux détails sur l'état de notre avancée, de nos choix, de nos réorientations, de notre productivité, de nos implémentations et étapes passées.

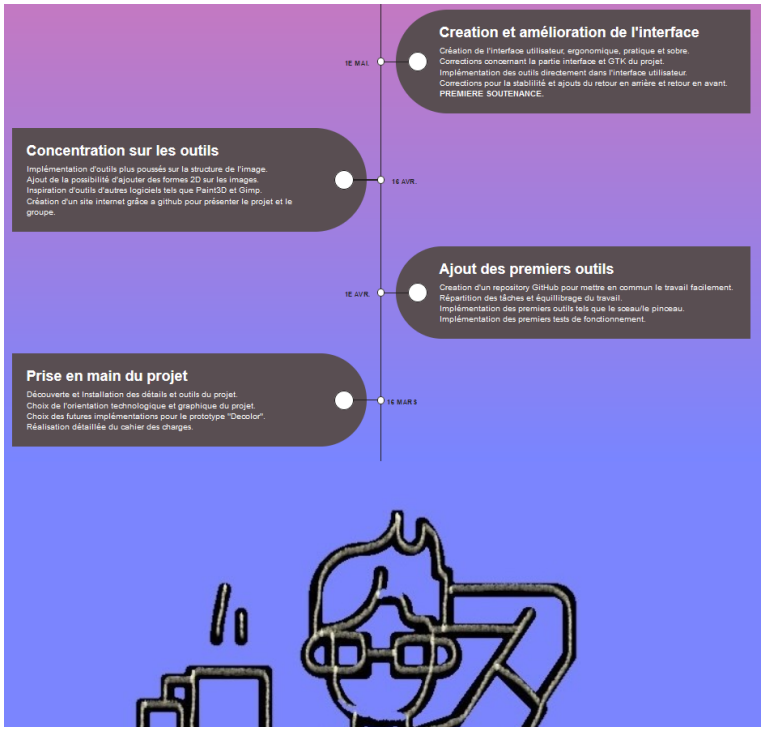


FIGURE 80 – Page d'avancements en fonction du calendrier

8.4 Page de documentation

Dans cette partie nous avons rassemblé l'ensemble des documents qui nous ont été demandés durant l'avancement de notre projet à savoir notre cahier des charges, les rapports de projet des première et

deuxième soutenances. Elles sont présentées avec leurs différentes caractéristiques avec un bouton pour les télécharger.

Dans la lignée des présentations, nous avons rassemblé l'ensemble des logiciels et des outils qui nous ont permis de faire aboutir notre projet.

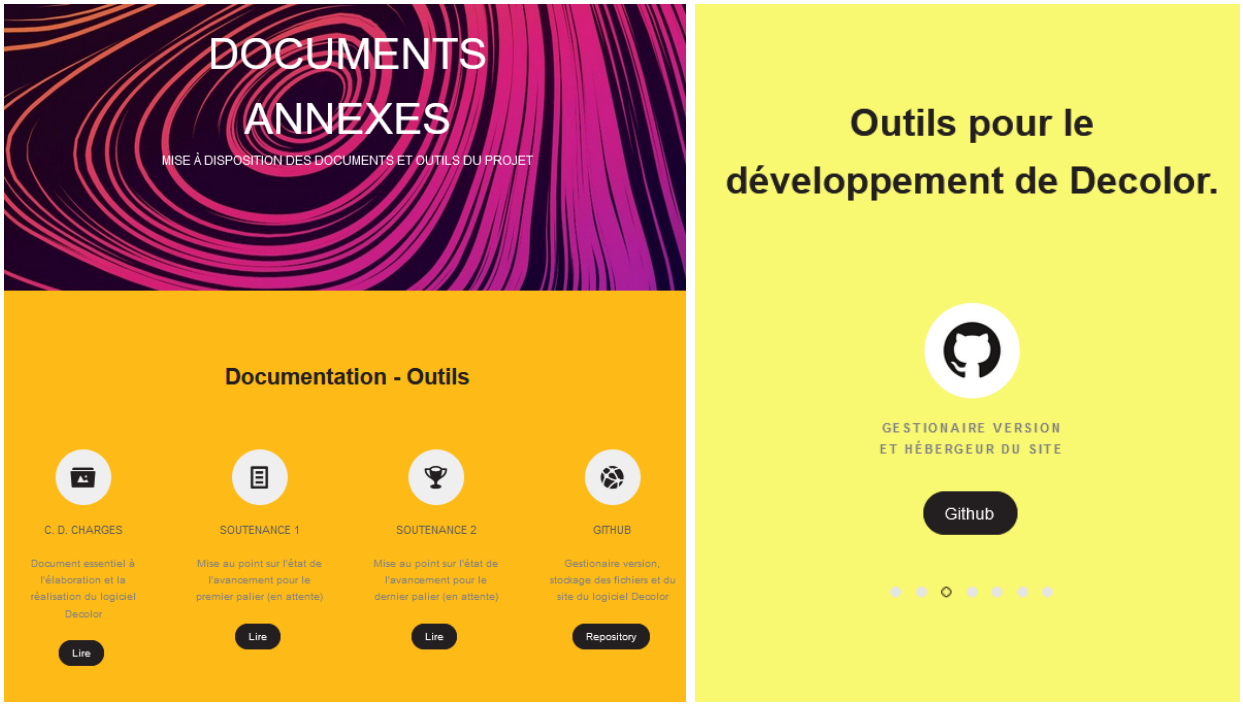


FIGURE 81 – Page documentations et outils

8.5 Page de l'équipe

Il nous a semblé important de présenter les différents membres du projet individuellement avec notamment leurs domaines de travail. Les liens vers le Twitter et le GitHub de chaque membre est renseigné sous chaque personne. Nous avons aussi simplement disposé quatre moyens pour pouvoir nous contacter ou qui concernent la partie communication de notre projet.

FIGURE 82 – Présentation de l'équipe

8.6 Page de téléchargement

Lorsque le bouton télécharger est activé, un ".tar.gz" correspondant à la version choisie de notre projet est téléchargé. Après l'avoir décompressé, il est fonctionnel. Une version du projet en version lite (sans fichiers de test/inutiles) et une version normale sont actuellement disponibles au téléchargement.

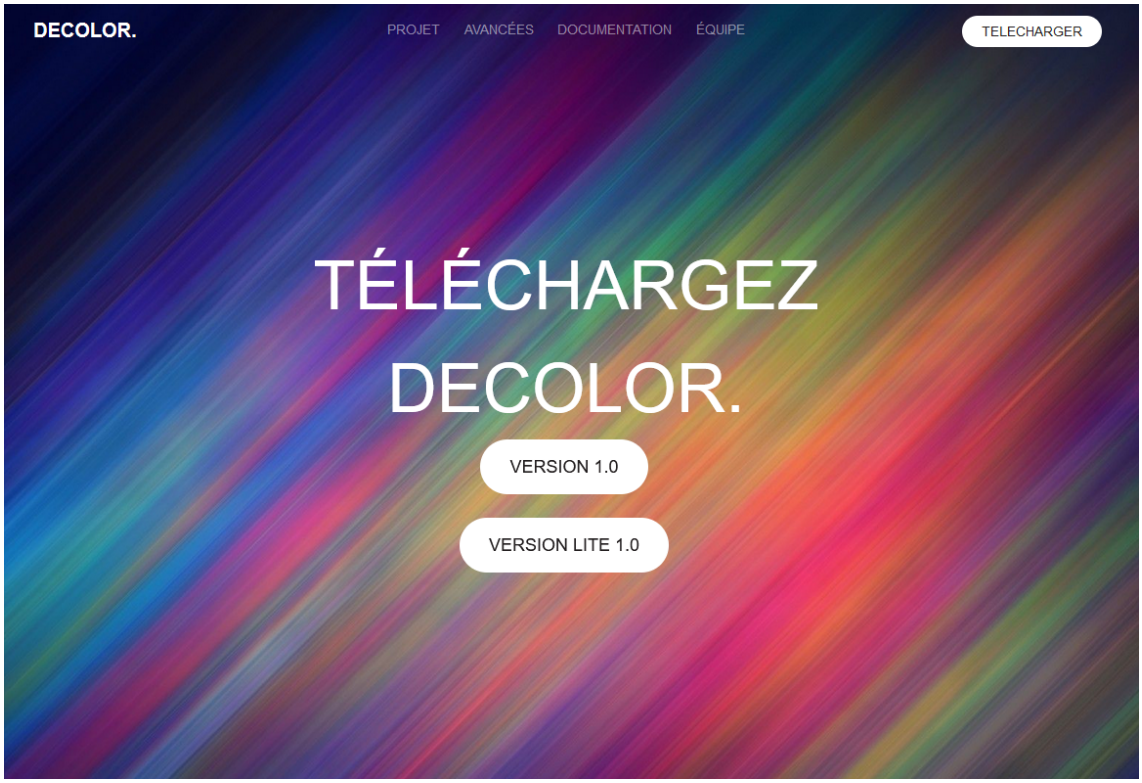


FIGURE 83 – Page de téléchargement

9 Développement sur les tâches réparties et la réalisation

9.1 alexandre.devaux-riviere

En tant que chef de projet, mon rôle a été de rester à l'écoute, faire tout mon possible pour épauler et guider mes camarades ainsi que maintenir une organisation efficace afin que le projet aboutisse et soit une totale réussite.

Sachant que j'ai des affinités avec la programmation et une connaissance de GTK plutôt limitée, j'ai donc principalement travaillé sur tout ce qui touche à l'image donc à ce qui impose l'utilisation de SDL, tout en gardant un oeil sur le reste. Je me suis beaucoup investi dans la création de la structure algorithmique de l'ensemble des outils d'édition de l'image et de la structure de celles-ci avec aussi l'implémentation de structures utiles au logiciel dans l'interface.

La cohésion du groupe, aidée par l'amitié qui nous liait déjà, a finalement fait de ce projet une réussite. Nous avons passé de longues soirées à travailler "tous ensemble" sur le projet, en échangeant avis et conseils. Le fait d'apprendre de nouvelles choses tout en créant a été d'une grande motivation durant tout ce projet. Ma méthode de développement s'est en partie basée sur mon expérience de travail en tant que

utilisateur de ce type de logiciel.

Personnellement, je considère ce projet comme une expérience très enrichissante que ce soit au niveau des notions de travail en groupe tout comme pour les connaissances en C que nous avons pu acquérir au cours de cette période.

9.2 kael.facon

Ce projet me paraissait être l'occasion parfaite pour consolider les bases que j'ai acquis en C durant le 3ème semestre. Il pouvait me permettre de mettre en oeuvre les connaissances que j'ai appris durant le 4ème semestre. Ayant travaillé avec les librairies GTK et SDL au travers du projet du 3ème semestre, j'ai pu me montrer utile pour tout ce qui était lié à l'interface et à la gestion des images.

Lors de la première soutenance je me suis concentré sur le fait de construire et d'implémenter une interface fonctionnelle et pratique, laissant certains autres aspect de côté. Mais lors de la seconde soutenance je me suis penché sur des pans que je ne connaissais pas de gtk et qui m'ont permis d'optimiser la place prise par les fonctionnalités. Ayant une interface plus aéré à ma disposition j'ai pu me concentrer sur son esthétique grâce notamment à l'ajout de multiples thèmes pour l'application.
Afin d'avoir un projet plus personnel j'ai également créé des icônes et différents logos afin de représenter le groupe, le projet mais également les options variées de l'application.

Personnellement j'ai été surpris par la vitesse à laquelle le projet s'est développé et l'ampleur qu'il a pris !
Grâce à un groupe motivé et inspiré pour travailler nous avons atteint nos objectifs et sommes même allés au delà ! Notre travail était coordonné et la communication était bonne !
Je pense que la cohésion d'équipe et la bonne répartition des tâches a aidé au développement rapide et efficace du projet ! Sans compter sur notre chef de projet qui était investit et savait où mettre les échéances afin que le projet avance sans discontinuité.

Pour faire simple, un projet qui se déroule sans accroc, ça fait du bien !

9.3 gregoire2.vest

De mon côté, j'ai adoré participer à ce projet car je suis un utilisateur de ce type de logiciel. J'ai donc pu découvrir toute l'algorithmique qui se cachait derrière ces logiciels. Même si je me suis simplement occupé de la partie SDL (c'est-à-dire les outils), j'ai observé comment fonctionnait toute l'interface que mes coéquipiers ont codé et cela m'a intéressé du début à la fin.

Pour ce qui est du code pur, j'ai apprécié coder avec du C en revenant de Lettonie car ça reste un langage très efficace et très populaire. J'ai pu apprendre beaucoup de chose tout en étant de plus en plus à l'aise avec le langage et c'est certain que ce projet m'aura beaucoup servi pour l'avenir.

J'ai aussi adoré le fait de travailler avec d'autres personnes que habituellement. Vu que je suis à Lyon et que les personnes du groupe dans lequel je suis habituellement étaient à l'étrange, j'ai pu découvrir une toute autre organisation différente de celle que j'avais l'habitude de voir donc cela m'a aussi apporté beaucoup de choses.

Et enfin, le plus sympathique durant ce projet était évidemment d'être dans un groupe avec une bonne ambiance et de voir notre logiciel bien avancer !

9.4 mathis.rabouille

Pour moi ce projet a été l'occasion de revenir a du C après mon retour de Lettonie. Il a été assez compliqué à appréhender car il s'agissait d'un thème assez libre. Mais c'est un projet qui fut très enrichissant.

Il nous a forcé à nous adapter a différents problèmes et à sortir de notre zone de confort. Nous avons eu un très bon travail d'équipe sur ce projet selon moi et la communication est passée facilement ce qui nous a permis d'avancer à un bon rythme. Il m'a permis de m'améliorer en C et dans l'utilisation d'outils comme GTK et je pense que cette expérience est très importante pour la suite. Ce projet a aussi été marqué par de nombreux choix que le groupe a du faire et nous nous sommes bien tenus à ces choix. Sans tous cela le travail aurais été beaucoup plus compliqué et nous n'aurions sûrement pas fini a temps. Malgré tout, le projet s'est très bien passé et nous en sommes tous fier.

10 Nous contacter

— Alexandre Devaux-Rivière : alexandre.devaux-riviere@epita.fr
— Kaël : kael.facon@epita.fr
— Grégoire: gregoire2.vest@epita.fr
— Mathis : mathis.rabouille@epita.fr

11 Annexes

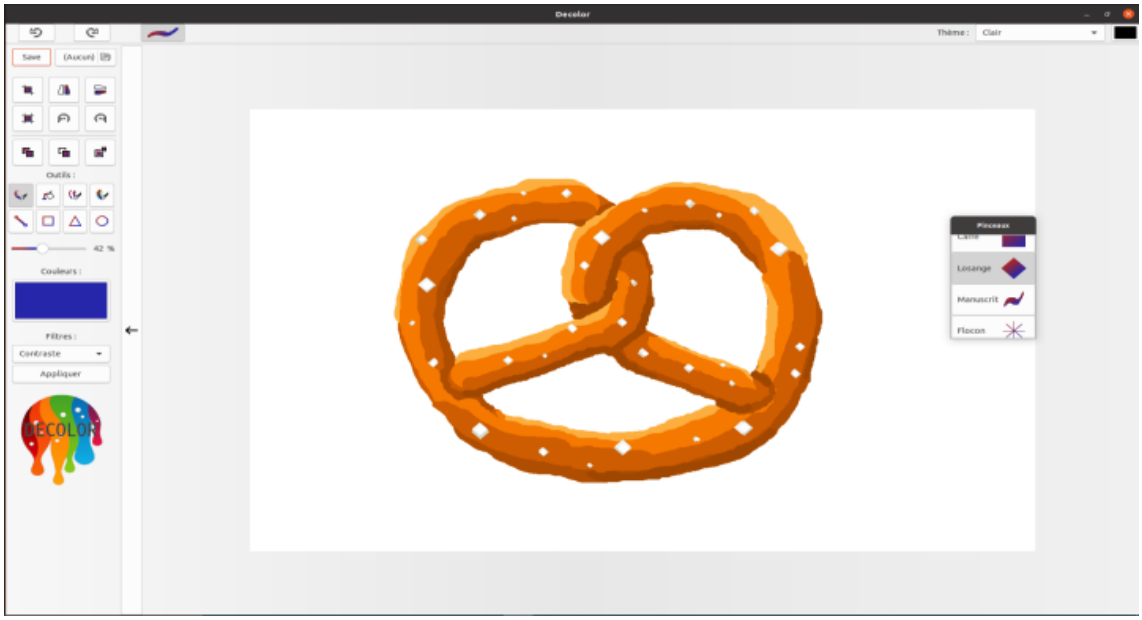


FIGURE 84 – Dessin de Kael intitulé : Pablo

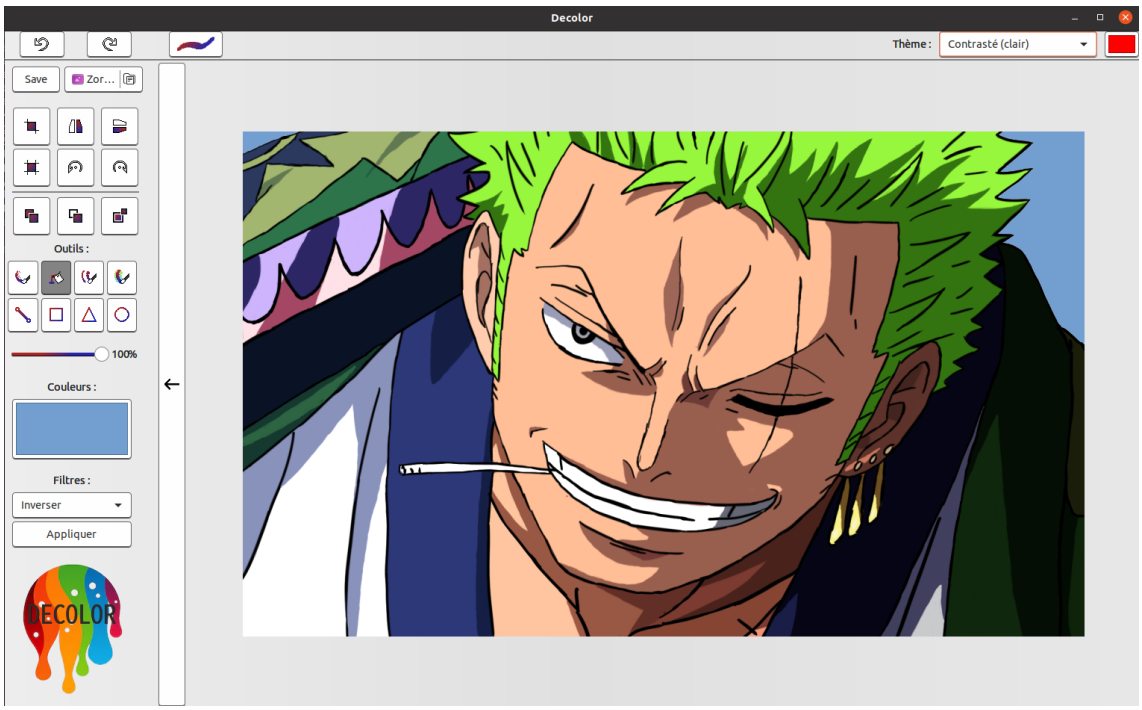


FIGURE 85 – Dessin de Mathis intitulé : Zoro le raciste

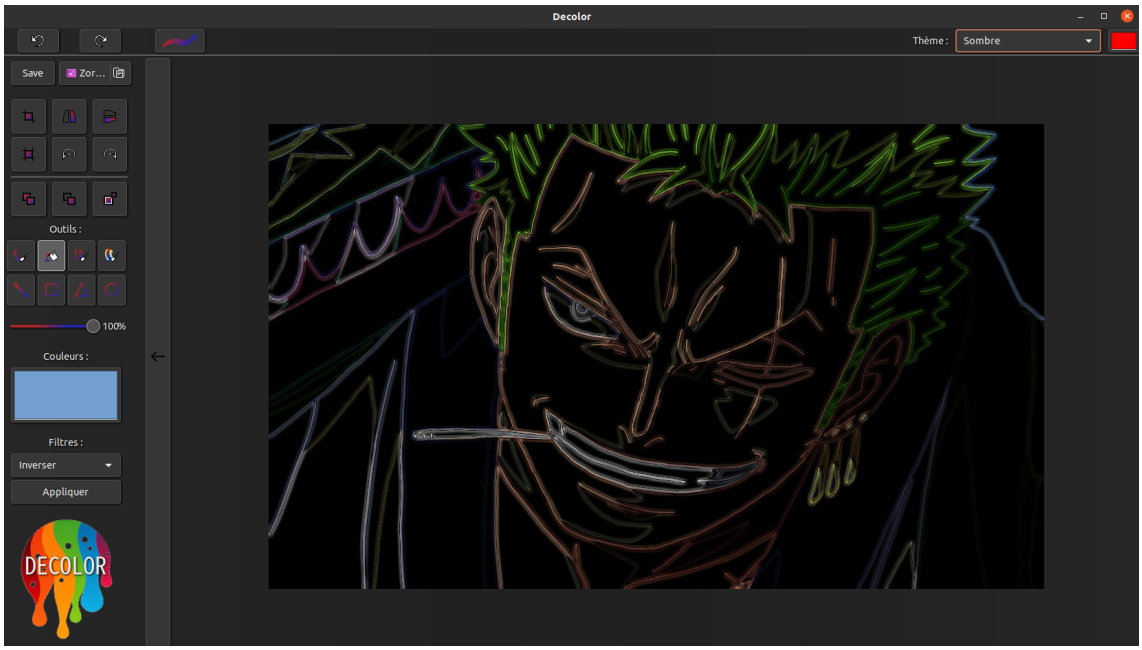


FIGURE 86 – Dessin de Mathis après utilisation de filtres

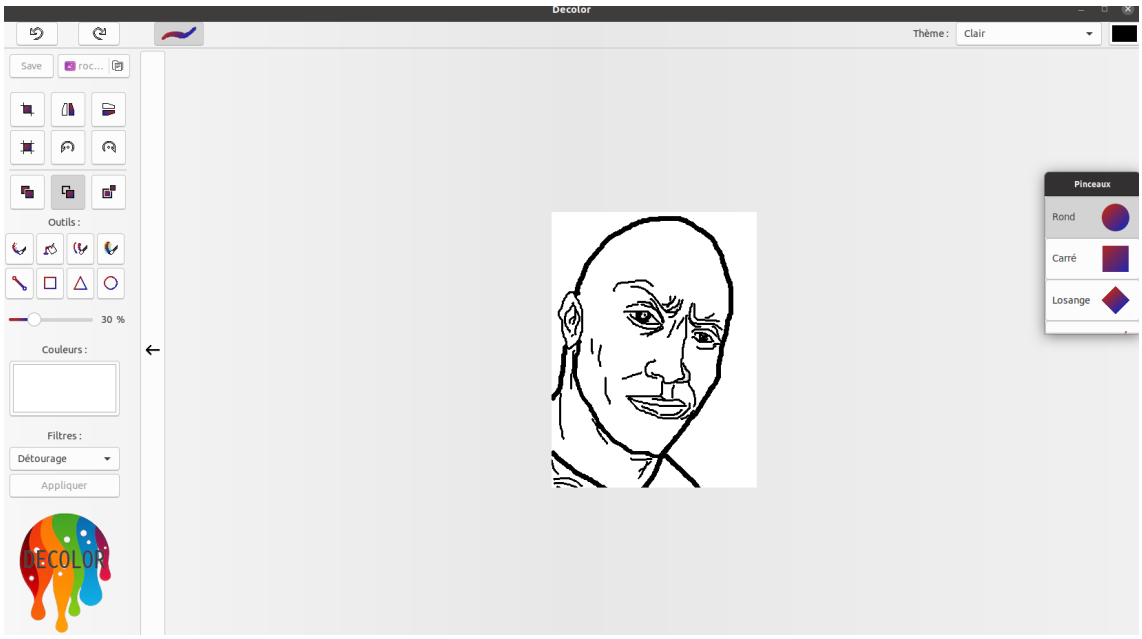


FIGURE 87 – Dessin d'Alexandre intitulé : Ze Roque

FIGURE 88 – Dessin de Grégoire intitulé : une tortue I guess