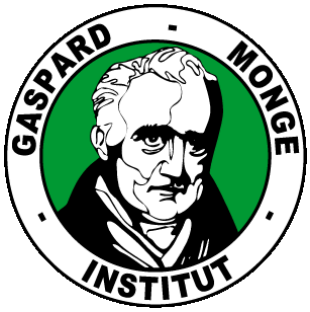

UNIVERSITÉ
PARIS-EST
MARNE-LA-VALLÉE

Création et Affrontement de personnages articulés personnalisables

Kévin TRANCHO

Licence Informatique 3ème année

Projet Tuteuré encadré par Cyril NICAUD

Année 2016-2017

Table des matières

Remerciements

Je tiens tout d'abord à remercier Monsieur Cyril Nicaud, enseignant-chercheur à l'université Paris-Est Marne-La-Vallée, qui a encadré ce projet tuteuré, mais aussi pour sa disponibilité pour le suivi du projet et pour nos échanges autours de démonstrations.

J'aimerai aussi remercier Claire David, responsable de la formation de 3ème année de Licence Informatique à l'université Paris-Est Marne-La-Vallée pour m'avoir mis en contact avec Monsieur Cyril Nicaud.

Je remercie également Victor Veillerette, ami et étudiant dans ma promotion pour avoir essayé le logiciel en tant que testeur et pour m'avoir soutenu depuis le début.

De plus je voulais aussi remercier ma famille, en particulier mes parents, Jason Hugel pour son soutien depuis le début, mes amis qui ont aussi été présents.

Chapitre 1

Introduction

1.1 Contexte

Dans le cadre de l'Unité d'Enseignement projet tuteuré qui est proposée sur le 5ème semestre de 3ème année de Licence Informatique, j'ai eu l'idée de ce projet tuteuré.

J'ai donc été mis en contact avec Cyril Nicaud par l'intermédiaire de Claire David pour mettre en forme l'idée et commencer le projet.

L'objectif était de créer un éditeur et gestionnaire de personnages inspiré de Blender (Logiciel d'infographie 3D).

FIGURE 1.1 – *Logo de Blender*

1.2 Le Sujet

1.2.1 Idée de départ - Premier intitulé

L'utilisateur pourra créer un personnage ou une créature de toutes pièces puis le faire combattre avec d'autres, une sorte de Street fighter personnalisable.
C'est-à-dire que le personnage pourra par exemple avoir une forme humaine ou autre créature (un python par exemple).
Les personnages pourront être stockés sous forme de fichier et servir de modèles prédéfinis (exemple humain).
Tout personnage sera articulé par un squelette, c'est-à-dire un ensemble d'os/membres

(structure sous forme d'arbre) auxquels sont associés des formes.
L'utilisateur disposera d'un éditeur de mouvements (rotations des articulations à temps donnés) pour définir les différentes attaques du personnage, ses déplacements ou autres.
Une fois que le personnage possède une représentation graphique, des articulations et des mouvements de base, il pourra être utilisé pour affronter d'autres personnages.
Dans un permier temps les adversaires seront gérés par une intelligence artificielle ou à la rigueur par un second joueur au clavier.

Fonctionnalités de base :

— Editeur de personnages (apparence)

— Editeur de mouvements (actions)

— Utilisation du personnage en jeu

On pourra toujours pousser le projet un peu plus loin avec des améliorations.

1.2.2 Idée plus claire et structurée

On divise le sujet en 4 grandes parties :

Etape 1 : l'armature/sequelette

— Mise en place d'une structure pour représenter l'armature (arbre)
— Fonctions de création, libération (gestion de la mémoire)
— Fonctions d'édition
— Ajout et Suppression
— Edition des valeurs de la structure
— Déplacer une extremité (absolu et relatif), rotation (absolue et relative)
— Création d'un éditeur d'armature
— Visualisation d'une armature
— Gestion de la sélection d'un os
— Edition de l'armature à la souris : ajout, suppression, déplacements
— Définition des limites d'angles (il existe une zone de minimum 30 que l'os ne peut pas franchir en rotation)
— Sauvegarde et chargement d'une armature (.armature)

Etape 2 : l'apparence/le personnage

— Ajout d'une forme à une articulation sélectionnée
— Gestion de la sélection d'un cercle, forme ou ensemble de formes
— Edition d'une forme : ajout de formes en tant qu'enfants, suppression de formes
— Opérations de base sur les cercles/formes/ensembles de formes
— Déplacement/Translation
— Agrandissement

— Rotation
— Duplication
— Gestion de transfert de forme entre articulation (utile pour une duplication par exemple)
— Sauvegarde et chargement de l'apparence d'un personnage (armature et formes) (.forme)
— Exportation et Importation de formes
— Calcul d'images pour les ensembles de formes pour fluidifier l'édition et préparer la gestion de mouvements

Etape 3 : les mouvements/actions

— Editeur de rotation des articulations
— Gestion des contraintes d'angles
— Création d'une timeline et déplacement d'un curseur sur celle-ci
— Possibilité de définir l'enclechement d'une rotation pour une articulation à un temps donné
— Visualiser l'animation de l'action
— Création d'actions de base prédéfinies
— Déplacement en avant
— Déplacement en arrière
— Saut
— Accroupissement
— Garde
— Création d'actions personnalisées
— Configuration des touches pour les actions
— Sauvegarde et chargement d'un personnage (armature, apparence, actions) (.perso)
— Vérification de cohérence d'un personnage pour le combat
— Chaque os semble recouvert par une forme apparente
— Le personnage est de taille raisonnable

Etape 4 : gestion d'un combat

— Visualisation du personnage dans une zone de combat
— Ajout des actions pour animer le personnage dans la zone et le diriger au clavier
— Gestion de modification des actions selon la vivacité des articulations
— Ajout d'un personnage adverse non contrôlé
— Ajout de quelques contrôles basiques au clavier pour servir de personnage test
— Gestion de collision et d'impacts entre personnages
— Gestion de perte de vivacité des membres lors d'impacts selon puissance
— Définition de règles d'arrêt de combat
— Création d'une petite intelligence artificielle à affronter

Etape 5 : améliorations (si le temps le permet)

— Actions de saisie en combat
— Améliorer l'intelligence artificielle

— Différents niveaux d'intelligence artificielle
— Intelligence artificielle qui apprend de ses échecs et ses réussites en combat ou selon la manière de jouer du joueur
— Petit langage permettant au joueur de coder une intelligence artificielle (et s'il le veut, la laisser jouer à sa place)
— Création d'un système de parcours graphique de fichiers pour la sauvegarde et le chargement
— Ajout d'armes et objets divers
— Ajout de sorts ou pouvoirs surnaturels
— Autres au fur et à mesure du projet

1.2.3 Contraintes

Contraintes matérielles

L'executable doit être compilable et executable sur une machine Linux 64 bits.
Le projet sera écrit en C ansi en utilisant la bibliothèque graphique SDL.
Le projet devra compiler avec -Wall et sera écrit au mieux dans la norme ANSI.

FIGURE 1.2 – *Logo de SDL*

1.3 Intérêt

1.3.1 Besoins fonctionnels

D'un point de vue utilisateur

L'intérêt du projet est de permettre à l'utilisateur de créer un personnage depuis 0 et pouvoir définir aussi bien la manière dont il sera articulé, que son apparence ou encore les différentes actions et animations possibles.
Une fois un personnage créé, il pourra affronter d'autres personnages dans une arène.

D'un point de vue technique

Le projet demande de pouvoir créer un squelette pour un personnage, donc la gestion d'un arbre, pour lequel on va gérer les rotations des articulations.

Pour la partie forme, ce qui est intéressant c'est de gérer un arbre pour lequel chaque noeud est representé par un cercle et relier ces cercles de manière à donner une apparence cohérente au personnage.

Pour le mode combat, on demande de gérer les collisions, rapports à la gravité pour

des personnages dont on ne connait pas la forme des os, ou forme globale que l'utilisateur peut lui donner.

Chapitre 2

Organisation du projet

2.1 Organisation du travail

Un rendez-vous mensuel était organisé pour voir l'avancée du projet et discuter de la suite du travail à faire.
Pour la répartition des tâches, le projet a été réalisé seul.

2.2 Choix des outils de développement

Le langage C a été choisi car c'était le plus adapté à la tâche, c'est-à-dire qu'étant compilé en langage machine, il permet de bonnes performances pour un grand nombre de calculs d'affichage et autres calculs de sélections par exemple.
De plus c'est aussi le langage avec lequel j'avais le plus d'aisance.
Le calcul, la génération et rotation d'images étant au coeur du projet, nous avons choisi SDL qui garanti des performances acceptables pour ce genre de tâche.
L'inconvénient du choix de ce langage et de cette bibliothèque graphique est le fait que le programmeur doit gérer tout ce qui relève de la mémoire et de l'algorithmique ce qui peut devenir chronophage ou peu performant si mal géré.

2.3 Planning prévisionnel

En première idée, il avait été défini de suivre au mieux cette répartition du temps :
— Etape 1 (armature) : du 01 octobre au 20 octobre
— Etape 2 (formes) : du 20 octobre au 10 novembre
— Etape 3 (mouvements) : du 10 novembre au 30 novembre
— Etape 4 (combat) : du 30 novembre au 20 decembre
— Etape 5 (améliorations) : dès la fin de l'Etape 4

Chapitre 3

Analyse du projet

3.1 Première lancée

Dans un premier temps, je me suis lancé directement dans l'écriture de code du 02 au 10 octobre.
C'est-à-dire environ 2 500 lignes dans lesquelles j'ai attaqué la création des os et directement des formes.
L'approche a été de commencer par beaucoup de tests unitaires.
Sans suivre les différentes étapes proposées.
Le projet a été recommencé de 0 avec une meilleure stratégie.

3.2 Les structures de données

3.2.1 Armature

L'armature est un arbre binaire de type fils gauche - frère droit.
Il existe des fonctions génériques pour les ajouts et suppressions détaillées plus bas.

Cette représentation en mémoire permet d'avoir les fils les uns à la suite des autres.
Ce qui est utile pour être vu comme une liste chaînée triée où les os derrière leur père sont trouvés en premiers.

Pour cet arbre les coordonnées de la racine sont les coordonnées absolues du point.
Pour chaque fils, les coordonnées sont relatives à son père.

Pour chaque noeud on attribue un identifiant binaire, c'est-à-dire le chemin pour y accéder depuis la racine.
0 pour passer par le frère.
1 pour passer par le fils.
Cela permet d'optimiser la recherche d'un Os dans l'armature et donc optimiser le calcul des coordonnées de l'os sans parcourir tout l'arbre.

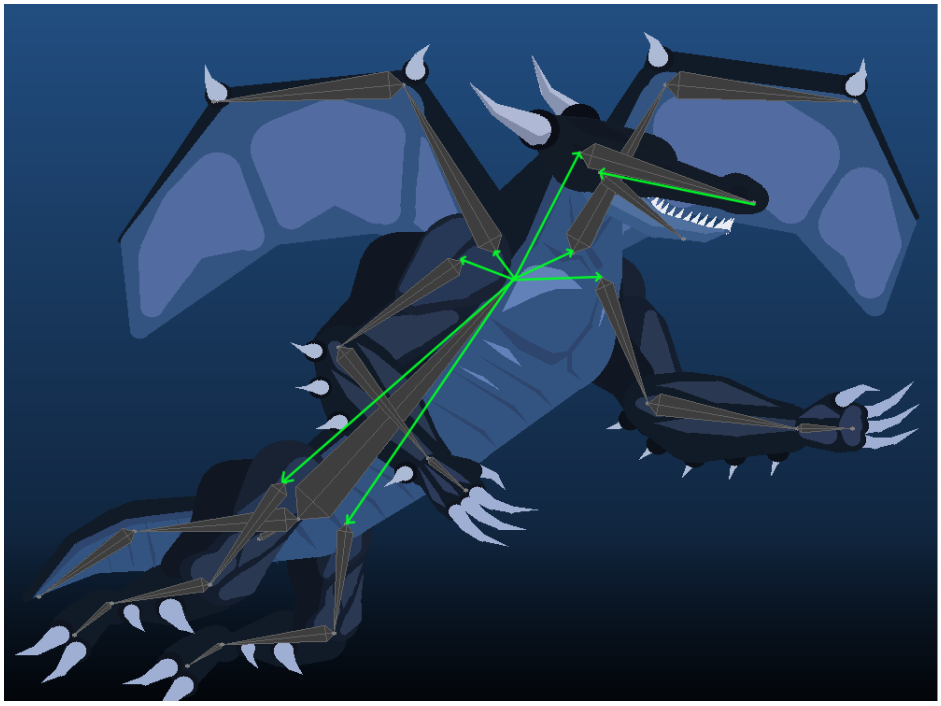

FIGURE 3.1 – *Coordonnées relatives des Os enfants*

3.2.2 Formes

Les formes sont gérées sous un ensemble de formes.
Chaque Os possède un ensemble de formes.
Une forme est un arbre binaire de type fils gauche - frère droit.
Pour ne pas stocker la couleur pour chaque noeud on utilise une structure intermédiaire.
Une forme est composée d'un cercle dont le centre est relatif à l'os et non à son père dans l'abre.
Le but de la structure forme est de pouvoir calculer une image pour les rotations sans avoir à faire tourner chaque élement de l'ensemble de formes.
Pour donner une cohérence à la forme, on calcule les tangentes externes entre deux cercles.
En trouvant les points aux cercles qui permettent de tracer les tangentes externes on peut dessiner un polygone qui relie les deux cercles.
Pour cela nous utilisons une solution du problème qui se traduit par une fonction évaluable en temps constant. (Idée mathématique menant à la résolution du problème en Annexe)
Chaque forme devient à l'oeil un arbre de cercles dont un polygone relie chaque fils à son père avec ce mécanisme qui donne une cohérence à la forme.

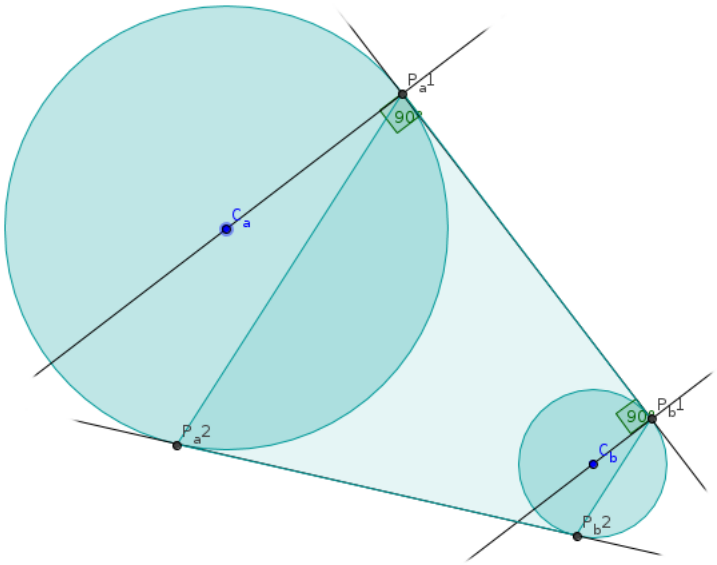

FIGURE 3.2 – *Tangentes externes entre deux cercles*

FIGURE 3.3 – *Exemple de sélection d'une forme et affichage des cercles*

3.2.3 Actions

Pour les actions, l'élément atomique est un temps, un temps auquel on attribue un angle d'arrivée et qui connait les temps précédents et suivants.
Un mouvement sur un Os est défini comme une liste de temps, donc un enchaînement d'angles atteints.

Une action est un ensemble de mouvements, ici implémenté comme une liste chaînée dans le but de faire des ajouts et suppressions de mouvements en plus petite complexité algorithmique.
Pour jouer un mouvement, on joue chaque mouvement les uns à la suite des autres.

FIGURE 3.4 – *Les temps d'actions sur la timeline en bas pour un Os sélectionné*

3.2.4 HitBox

Les HitBox sont représentées en mémoire comme un arbre binaire de type fils gauche - frère droit.
Elles sont calculées automatiquement à partir de l'os et de sa forme, c'est une simplification de la forme.
C'est-à-dire que pour le calcul de la HitBox :

- La HitBox devient les extrémités de l'Os
- La HitBox s'agrandit en poussant sur les différentes formes de l'Os
- Si en poussant une composante est déjà inclue dans une autre on ne l'ajoute pas
- On définit un seuil pour lequel deux cercles inclus à plus ou moins ce seuil ou de rayon inférieur sont supprimées car non représentatives de l'aspect général

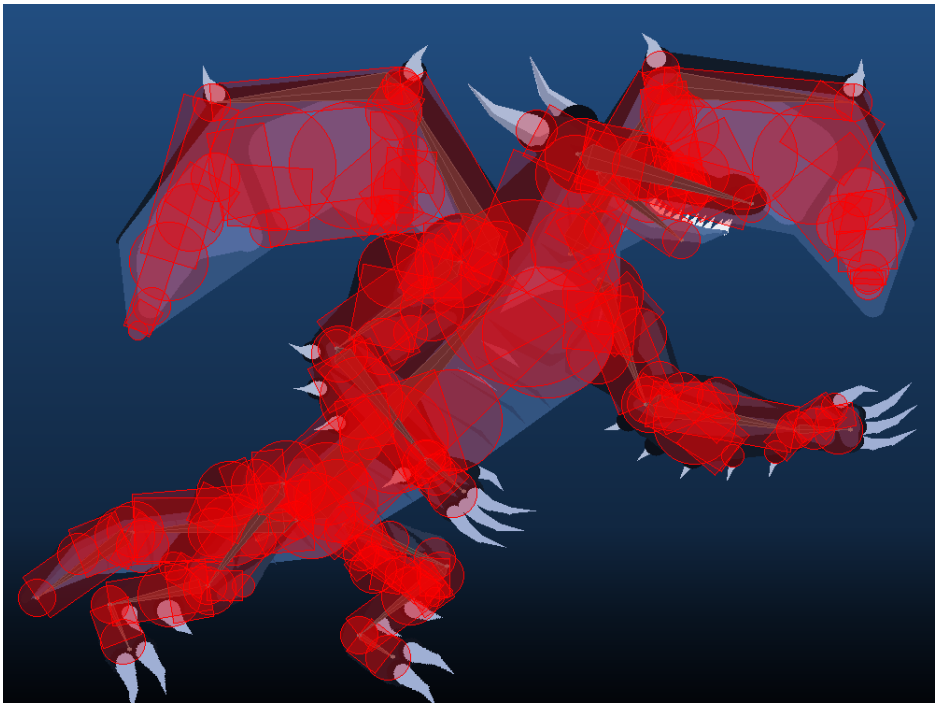

FIGURE 3.5 – *Exemple de HitBox calculée automatiquement*

FIGURE 3.6 – *Second exemple de HitBox*

3.2.5 Personnage

Un personnage est la structure générale qui est representée par une Armature (arbre binaire) et par un ensemble d'actions (ensemble de listes chaînées).

3.2.6 Deplacements

On définit 3 types d'actions pour les déplacements :

Mouvement Nul

Le mouvement Nul est un vecteur qui n'engendre à lui seul aucune translation du personnage, caractérisé par un vecteur Nul.
Ce type de mouvement est réservé pour les coups portés à un autre adversaire.
Il peut être combiné avec un mouvement Statique, c'est-à-dire qu'on peut lancer un déplacement linéaire vers une direction et donner un coup en même temps ou après lancement.

Mouvement Statique ou Linéaire

Le mouvement Statique ou Linéaire est un mouvement dont le vecteur de déplacement du personnage est constant.
C'est-à-dire que c'est un translation qui s'effectue vers la droite ou exclusif vers la gauche.
Seul il n'engendre aucune attaque de combat, mais peut être combiné par une action d'attaque.
On peut interrompre un mouvement Statique avec une autre action.

Mouvement Continu ou Dynamique

Le mouvement Continu ou Dynamique est un mouvement dont le vecteur varie au cours du temps et demande une condition d'arrêt.
Il existe deux moyens de créer un mouvement continu :

- Une translation vers le haut (saut ou bond si orienté)
- Une glissade (translation vers le bas)

En cas de mouvement opposé à la gravité, on a besoin de définir l'arrêt du mouvement continu comme le fait de toucher le sol.
Un mouvement vers le haut seul est combinable avec un mouvement Statique et en cas de collision au sol rend la main au mouvement Statique.
En cas de mouvement orienté vers le bas (glissade), si activé pendant un mouvement Statique, les mouvements se combinent et le personnage glisse vers la direction définie en mouvement Statique.
Si le mouvement vers le bas n'est pas combiné avec un mouvement Statique au déclenchement ou par composition il sera équivalent à un mouvement Nul.

3.2.7 Arbre générique fils gauche - frère droit

Etant un mécanisme utilisé par l'Armature, les formes et la HitBox, ce mécanisme a été rendu générique pour l'ajout et la suppression.
Le mécanisme générique nécessite de recevoir une structure compatible, c'est-à-dire que

cette structure doit fournir la position de ses champs dans la structure pour être interprétée comme un arbre binaire fils gauche - frère droit.

Mecanisme d'ajout

— S'il n'y a pas de noeud on le crée
— Sinon si le noeud n'a pas de fils, on lui ajoute un fils
— Sinon On ajoute un frère au frère de ses enfants n'ayant pas de frère (dernier frère de son enfant)

Mecanisme de suppression

On peut équiper la fonction de suppression d'une fonction générique indiquant quel chemin prendre (cas du chemin binaire de l'Armature).
On recherche le noeud à supprimer, une fois trouvé :

— Si pas d'enfant, son frère prend sa place
— Sinon s'il n'a pas de frère son fils prend sa place
— Sinon le frère le plus éloigné de son fils prend le frère du noeud comme frère puis le fils du noeud prend sa place

Chapitre 4

Developpement

4.1 Bilan général

Après avoir recommencé de 0 :
Au moment de l'écriture du rapport, le projet est composé de plus de 17 500 lignes de code source C.
C'est-à-dire plus de 800 fonctions (externes et locales confondues).
En local, des archives des différentes versions sont tenues à jour.
Pas de fuites mémoire apparente de la responsabilité du projet.
Fuites mémoires connues de SDL :

- 504 blocs non free sur ma machine (donné par valgrind ./executable -no-action qui vérifie les fuites mémoires dues à SDL uniquement)
- 1 blocs non free si utilisation de filledPolygonRGBA (505 blocs non free constants dans ce cas)

Pas de cas de crash connu référencé pour le moment.
Beaucoup d'erreurs d'initialisation indiquées par valgrind en particulier lors de l'utilisation de Image_print_rotated_on_surface.

4.2 Etape 1 : Armature

Du 12 octobre au 14 novembre.

4.2.1 Mise en place

- Mise en place d'une structure globale permettant de gérer la fenêtre avec SDL dans l'ensemble du projet : module Window
- Création du mécanisme d'Amature, c'est-à-dire création d'un os avec un début et une fin
- Affichage d'une représentation graphique de la structure d'Os sous forme d'une ligne de de deux points

4.2.2 Début de l'édition d'une armature

- Calcul de distance à un point dans l'Armature

— Selection des extrémités d'un Os
— Translations sur les extrémités de l'Os sélectionné
— Ajout et Supression d'Os
— Sélection d'un Os entier (distance à un segment)

4.2.3 Plus de fonctionnalités

— Possibilité de recoller un Os à son père
— Ajout de la zone d'angle et de son édition (Zone qui limite les rotations d'un Os plus tard)
— Modification de l'aspect graphique des Os pour une apparence proche de ceux de Blender
— Ajout de Menus génériques
— Ajout d'entrées de texte
— Ajout d'un explorateur de fichiers (proposé en Etape 5 mais utile pour les sauvegardes et chargements)
— Ajout de sélection multiple des Os
— Ajout de miroir selon l'axe des abscisses

4.2.4 Ajouts après fin de l'étape

— Changer le père d'un Os (Pour éviter de tout recommencer depuis le début ou recréer toute une partie)
— Ajout d'un père qui pousse depuis l'extremité de départ d'un Os
— Définition d'ordre d'affichage des Os entre eux et par rapport à leur père

4.3 Etape 2 : Formes

Du 14 novembre au 08 janvier.

4.3.1 Mise en place

— Ajout d'un ensemble de Formes à chaque Os
— Sélection de cercles
— Dessin des polygones formés par les tangentes externes entre les cercles
— Sélection de différents niveaux : cercle, forme, ensemble de formes
— Définition d'une couleur pour une forme

4.3.2 Plus de fonctionnalités

— Changer l'os dont dépend l'ensemble de formes
— Dupliquer un cercle, une forme ou un ensemble de formes
— Possibilité d'agrandir ou faire pivoter les formes et ensembles de formes
— Création d'une liste de formes pour sauvegarder et importer des motifs créés
— Possibilité de changer l'ordre d'affichage des formes dans l'ensemble de formes

4.4 Etape 3 : Actions

Du 08 janvier au 05 février.

4.4.1 Mise en place

— Ajout d'une nouvelle méthode d'affichage et gestion des Os en fonction d'un angle définissant leur rotation
— Sélection d'un Os en prenant en compte les mécanismes de rotations
— Rotation dirigeable à la souris
— Mise en place d'une timeline pour visualiser les différents temps et se déplacer un curseur sur l'animation
— Ajout de temps pour lequel l'Os est à un angle donné
— Jouer les animations, rotation des Os en simultané
— Définition d'une liste d'actions
— Possibilité de définir un angle de départ différent de celui de l'édition de l'Armature
— Création d'un vecteur définissant le déplacement du personnage

4.4.2 Après fin de l'étape

— Possibilité de définir le vecteur de déplacement
— Ajout et Suppression d'actions
— Renommer une action
— réinitialiser l'angle de départ

4.4.3 Pas encore implémentés au temps de l'écriture du rapport

— Agrandissement de la taille du temps d'une rotation (Définition début, fin et déplacement)
— Définition de touches personnalisées pour les actions (Nécessaire pour le mode 2 joueurs)

4.5 Etape 4 : Combat

Du 05 février au 03 mars.

4.5.1 Gestionnaire de Personnage

— Renversement d'un personnage par miroir
— Modification de la taille d'un personnage
— Actions enclenchées par les touches
— Ajout d'actions de déplacements droite/gauche saut
— Possibilité de combiner les déplacements
— Ajout de déplacements vers le bas pour les glissades
— Génération de HitBox automatiquement
— Gestion des dimensions d'un personnage par ses HitBox et adhérence au sol

— Ajout de menu de sélection de deux personnages
— Gestion des personnages dans une zone de combat

4.5.2 Pas encore implémentés au temps de l'écriture du rapport

— Gestion de collision des HitBox entre deux personnages (gestion des impacts)
— Gestion de la vivacité des Os
— Définition du résultat d'un coup porté
— Gestion d'une petite Intelligence artificielle

4.6 Erreurs et bugs connus

4.6.1 Erreur valgrind sur la rotation d'image en combat

valgrind détecte une erreur de valeurs non initialisées lors de l'affichage d'une image après rotation dans le module Image.
Erreur dans la fonction Image_print_rotated_on_surface détectée à l'appel de SDL_LowerBlit dans la fonction SDL_BlitSurface.
Erreur répétée à chaque affichage de l'image d'un Os, donc en très grande quantité lors d'un combat.
Ne semble pas pour autant à première vue poser de problème à l'execution.

4.6.2 Editeur de formes scroll dans la liste de Formes

Si l'utilisateur garde une forme sélectionnée et effectue un scroll dans la liste de formes, les actions de changement de la taille des cercles et celle du scroll se font en même temps.

Chapitre 5

Manuel d'utilisation

Pour chaque éditeur, il est possible d'accéder à un menu de fonctionnalités par l'intermédiaire de la touche **Espace**.

5.1 Editeur d'Armature

5.1.1 Changement d'éditeur

A partir de l'éditeur d'armature :
— On peut retourner au menu par la flèche directionnelle gauche
— On peut passer à l'éditeur de Formes par la flèche directionnelle droite

5.1.2 Sélections

La sélection des os se fait à la souris avec le clic **droit**.
Il est possible de faire 3 types de sélections :
— Sélection de l'extrémité de départ
— Sélection de l'extrémité en fin
— Sélection de tout l'os
S'il faut choisir entre la fin de l'os parent et le début d'un os enfant, la sélection porte sur la fin de l'os parent.
Il est possible de sélectionner plusieurs os en même temps, pour cela on combine le maintient de la touche **Ctrl** et du clic **droit**.
Il existe un ordre de sélection qui est important pour certaines actions comme le changement de père, la défintion de miroirs par exemple.
Le **premier** Os sélectionné est en **jaune**, ses **suivants** sont en **orange**.
On peut changer le premier os sélectionné en sélectionnant un os en **orange** qui deviendra l'os **jaune**.

FIGURE 5.1 – *Sélection multiple*

5.1.3 Translations

Une translation s'effectue par le maintient du clic **gauche**, c'est-à-dire qu'elle commence au clic et se termine au relachement.
Sélectionner le début de l'os ne fera se déplacer que ce point.
Sélectionner la fin de l'os fera se déplacer le point ainsi que les enfants de l'os.
Sélectionner un os implique la sélection de début de fin d'un os.
Il est possible de combiner le déplacement avec l'utilisation de **Ctrl** et **Shift**, c'est-à-dire :

- Shift : déplace la sélection de manière indépendante des relations de parenté
- Ctrl : déplace la sélection en gardant fixe les parties des enfants non directement reliées à la sélection
- Ctrl + Shift : déplace la sélection sans appliquer l'effet miroir s'il existe

5.1.4 Ajouts et suppressions

Ajouts

On peut ajouter un os par le menu avec **Espace** ou par raccourci clavier **E**.
A la création l'extrémité de fin de l'os est sélectionnée et on commence en mode translation, il suffira de cliquer **clic gauche** pour valider les coordonnées du nouvel os.
L'os ajouté depuis un autre os devient l'enfant de cet os.
Il est possible de créer par l'ajout un parent à un os en sélectionnant le début de l'os.

Suppressions

Il est possible de supprimer un os depuis le menu avec **Espace** ou par raccourci clavier **Suppr**.
Si on supprime un os qui possède des fils, ses fils deviennent fils de son père.
On ne peut pas supprimer l'os racine.

5.1.5 Os fils et son père

Relier

Si un os a été déplacé et n'est plus lié à son père, il est possible de le relier à nouveau. Action **relier** possible depuis le menu par **Espace** ou par raccourci clavier **P**.

- Sans modificateur : Pas de déformation, le départ de l'os revient à la fin de son père
- Ctrl : On fixe la fin de l'os et seul le départ de l'os revient à la fin de son père
- Shift : Pas de déformation, le départ de l'os va aux mêmes coordonnées que le départ de son père
- Ctrl + Shift : On fixe la fin de l'os et seul le départ de l'os va au départ de son père

Changer de père

Changer de père veut dire que l'os dépendra d'un autre os qui deviendra son nouveau père.
Un os ne peut pas prendre comme père l'un de ses descendants.
Action **reparenter** depuis le menu ou par **R**.

5.1.6 Miroir

Un os miroir est un os qui imite les translations de son miroir en ordonnées et fais l'opposé des translations en abscisses.
Le miroir est valable uniquement pour l'éditeur d'Armature.
Pour créer un miroir il existe plusieurs méthodes pour lequelles on lance l'action **miroir** depuis le menu ou par **M**.

- Selection unique et l'os possède un miroir on casse le lien
- Sélection unique et il existe un miroir pour le père de l'os, alors le miroir créé sera le fils du miroir du père
- Sélection unique et pas de miroir pour le père, alors le miroir de l'os sera un frère créé
- Sélection multiple et l'os possède un miroir, on casse le miroir et on recrée un miroir entre les deux premiers os sélectionnés (si ce n'est pas l'ancien miroir)
- Sélection multiple on crée un lien de miroir entre les deux os

FIGURE 5.2 – *Ailes en miroir*

5.1.7 Ordre des Os

Il existe une relation d'ordre entre les os qui permet de faire qu'un os puisse être devant ou derrière son père.
Cet ordre fait aussi que deux frère peuvent passer devant ou derrière l'autre.
Pour faire avancer un os, on utilise **Avancer** depuis le menu ou la touche directionnelle **flèche haut**.
Pour faire reculer un os, on utilise **Reculer** ou la touche **flèche bas**.

FIGURE 5.3 – *Ordre des Os par rapport à leur père*

5.1.8 Zone Angle

La zone d'angle est la zone par laquelle l'os ne pourra pas passer en rotation lors d'un mouvement.

On peut la définir par **zone angle** dans le menu ou par la touche **G**.
Une fois en mode zone angle, on peut déplacer la zone avec **clic gauche** et valider par **clic droit**.
Si on se place en face du cercle on fait bouger toute la zone en conservant sa taille.
Sinon on sélectionne l'extrémité dont on est le plus proche pour réduire ou augmenter la zone.
La zone reste comprise entre 30 et 330 degrés.

FIGURE 5.4 – *Zone angle*

5.2 Editeur de Formes

5.2.1 Changement d'éditeur

— Gauche : pour retourner à l'éditeur d'Armature
— Droite : pour passer à l'éditeur de Mouvements

5.2.2 Sélections

Une sélection s'effectue par un clic **droit**.
Il existe 3 types de sélections :

Sélection d'ensemble de formes

Cette sélection s'effectue sur les os du personnage.
Cette sélection est prioritaire si aucune sélection n'existait avant.
Ici on contrôle toutes les formes de l'os en même temps.
Attention si plusieurs formes sont regroupées ensemble.

FIGURE 5.5 – *Sélection Ensemble de formes*

Sélection d'une forme

Ici on sélectionne toute une forme, c'est-à-dire un ensemble de cercles de couleur unie.
Cette sélection est prioritaire dans un os s'il existait une sélection d'ensemble de formes.
On contrôle l'ensemble de cercles indépendamment des autres formes de l'os.

FIGURE 5.6 – *Sélection d'une forme*

Sélection d'un cercle

C'est la sélection la plus atomique de l'éditeur de formes.
Ici on contrôle uniquement un cercle.
Cette sélection est prioritaire dans une forme sélectionnée auparavant.

FIGURE 5.7 – *Sélection d'un cercle*

5.2.3 Translations - Rotations - Angrandissements

Pour effectuer une translation, même mécanime que pour l'éditeur d'Armature, on la gère au clic **gauche**.
Pour les cas de rotations et d'agrandissements, il est possible de les gérer depuis le menu, cependant les raccourcis claviers **R** pour rotate et **S** pour scale seront plus pratiques.
On peut à la fois combiner rotation et agrandissements.

5.2.4 Ajouts et suppressions

Les ajouts se font depuis le menu ou par le raccourci clavier **E** pour extrude.
Les suppressions se font depuis le menu ou par **Suppr**.

Sélection de l'Os

Depuis l'os on créera une nouvellement forme par sélection d'un cercle en mode transition.

Sélection d'une Forme

Depuis une forme, il faut se placer sur un cercle de la forme pour lancer l'extension à un autre cercle partie de la forme en mode transition.
Supprimer depuis celle sélectionne supprimer toute la forme.

Sélection d'un Cercle

Le mécanisme est le même que pour la sélection d'une forme sauf qu'ici on se reduit à un cercle.

5.2.5 Couleur

Il est possible d'accéder au sélectionneur de couleur depuis le menu ou par la touche **C**.
Il change la couleur de la forme ou du cercle sélectionné.
Il est possible de maintenir le clic en continu pour voir l'aspect qu'aurait la forme selon la couleur.
Pour valider la couleur on clique en dehors de la zone de sélection de couleur.
Pour créer un nouveau cercle d'une certaine couleur, il est possible de lancer le sélectionneur sur une forme possédant la couleur et valider la couleur.

FIGURE 5.8 – *Sélectionneur de couleur*

5.2.6 Duplication

Pour les cas de sélection de cercle ou de forme la duplication reste interne à l'os, on crée un nouvelle forme identique à celle sélectionnée.
Pour le cas de sélection d'un ensemble de formes, la duplication demande d'être apparentée à un os qui peut être celui depuis lequel on duplique, on se place dans une configuration de changement de père (section suivante).
La différence sera qu'un ensemble de forme non apparenté sera effacé.

5.2.7 Changement de père

Lors d'un changement de père seul la translation est autorisée.
On démarre l'action dans le menu ou avec la touche **P**.
Pour apparenter la sélection à un nouvel os il faut le sélectionner avec le clic **droit**, le clic gauche servant pour les translations.
Si erreur pour apparenter la sélection garde le même parent.
Permet en particulier de fusionner deux ensembles de formes.

5.2.8 Liste de formes

La liste de formes en haut à droite permet de garder en mémoire des formes dans le but de les réutiliser plus tard ou les avoir à disposition.
La liste de formes chargée par défaut est la dernière liste de formes utilisée à après arrêt du processus.
On peut lancer un menu sur la liste de formes pour la sauvegarder, en charger une nouvelle ou existante.
On peut ajouter une forme à la liste depuis le menu ou par la touche **A**.
Il est possible de scroller dans la liste.
Lorsque l'on clique sur une forme en se place dans un mode de duplication du nouvel

l'ensemble de formes créé et sélectionné.

FIGURE 5.9 – *Sélection multiple*

5.2.9 Ordre des Formes

Pour les formes il existe un ordre d'apparition des formes dans l'ensemble de formes.
On peut avancer une forme depuis le menu ou par la touche directionnelle **haut**.
On peut de même reculer une forme par le menu ou la touche **bas**.

5.3 Editeur de Mouvements

La touche **V** permet de voir ou non l'armature dans l'éditeur de mouvements.

5.3.1 Changement d'éditeur

— Gauche : pour revenir à l'éditeur de Formes
— Droite : pour essayer le personnage en simulation solo

5.3.2 Sélections et rotations

On peut sélectionner un os par clic **droit** et lui faire faire une rotation par maintient du clic **gauche**.

FIGURE 5.10 – *Sélection et rotation d'un Os*

5.3.3 Liste des actions

Pour accéder à la liste des actions on utilise la touche **M**.
Si aucune action n'est choisie on revient au cas de non gestion d'un animation.

5.3.4 L'angle de départ

Si aucune action n'est choisie, la modification de l'angle modifie l'angle de d'apart d'un os.
On peut réinitialiser cette modification par le menu si sélection d'un os sans être dans la gestion d'une action.

5.3.5 Gestion de la timeline et animation

Si on édite une action, une timeline apparaît en bas de la fenêtre, on peut y déplacer le curseur avec le maintient du clic **droit**.
Si un mouvement est en cours la zone durant laquelle il affecte l'os est en vert sinon en rouge.

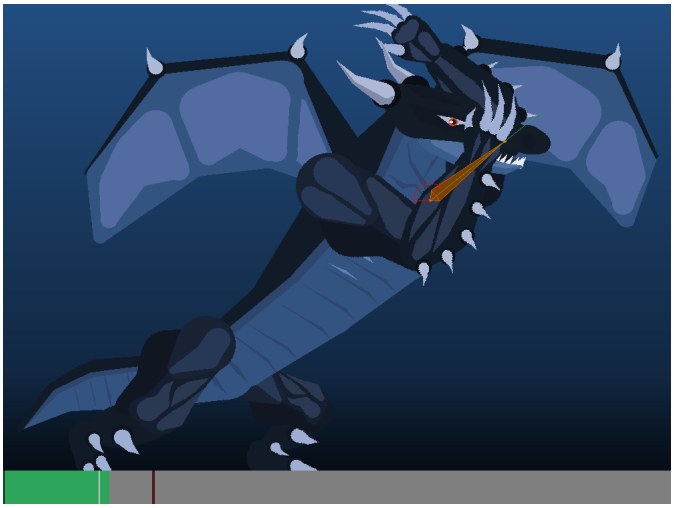

FIGURE 5.11 – *Edition des mouvements avec curseur sur la timeline*

5.3.6 Ajout et suppression de mouvements

Ajout

On peut ajouter un mouvement pour un os à un temps donné par le curseur sur la timeline soit par le menu avec **Ajouter mouvement** ou par la touche **A**.
Le mouvement est calculé en fonction des temps précédents pour essayer de faire au mieux pour se terminer et se placer à l'angle définit.
Le personnage possède cependant une limite de vitesse qui empêche aux mouvements d'être définis par l'utilisateur.

5.3.7 Définition du déplacement

Pour définir le déplacement il faut se trouver dans une action, sans os sélectionné et lancer le menu.
On peut supprimer un déplacement pour considérer que c'est une action de combat ou définir son vecteur par **Ajouter déplacement**.
Il existe plusieurs manière de considérer un déplacement :

- Déplacement vers gauche ou droit : déplacement simple linéaire
- Déplacement vers le haut : saut ou bond orienté dans une direction
- Déplacement vers le bas : possibilité de glisser ou glissade orientée

Les action de saut/bond et de glissade sont aussi vues comme des actions capables de blesser l'adversaire.

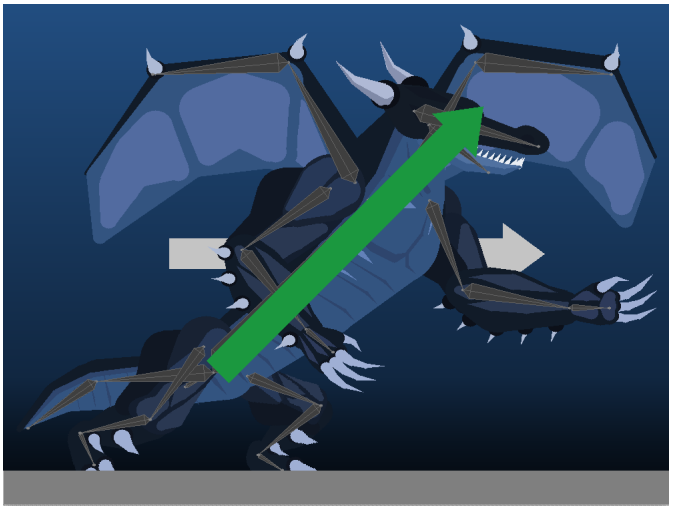

FIGURE 5.12 – *Définition d'un déplacement de bond vers l'avant*

5.3.8 Ajout et suppression d'actions

On peut ajouter un action en lançant le menu à partir du moment qu'aucun os n'est sélectionné.
Si on édite une action on peut choisir dans le menu de la supprimer.
Pendant l'édition d'une action on peut lancer le menu pour renommer l'action.

FIGURE 5.13 – *Ajout d'une action*

5.4 Combat et gestion du Personnage

5.4.1 Le menu principal

Depuis le menu principal on sélectionne deux personnages qu'on peut respectivement créer, charger ou éditer.
Le personnage adverse est inversé, c'est-à-dire qu'il intéragit dans le sens opposé au personnage du premier joueur.

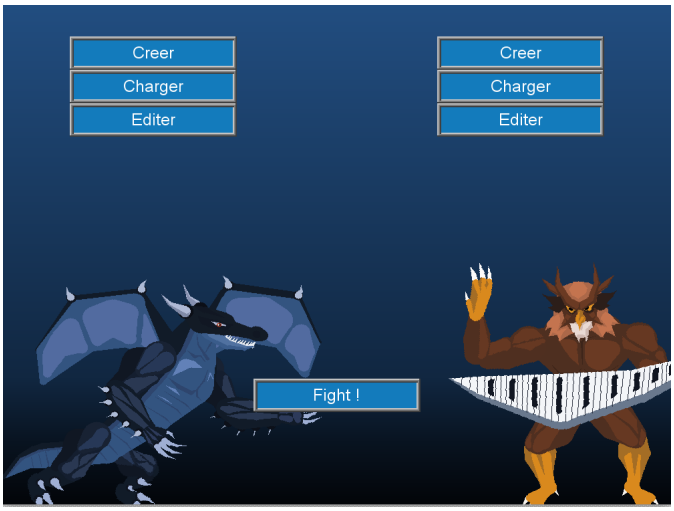

FIGURE 5.14 – *Menu principal*

5.4.2 Sauvegarde et chargement d'un personnage

Un petit outil est fourni pour aider à la sauvegarde et le chargement des personnages.
Il est possible d'utiliser un fichier en ouverture ou en écriture en cliquant dessus.
Ou on peut aussi l'entrer en texte dans l'input en haut de la fenêtre en cliquant dessus.

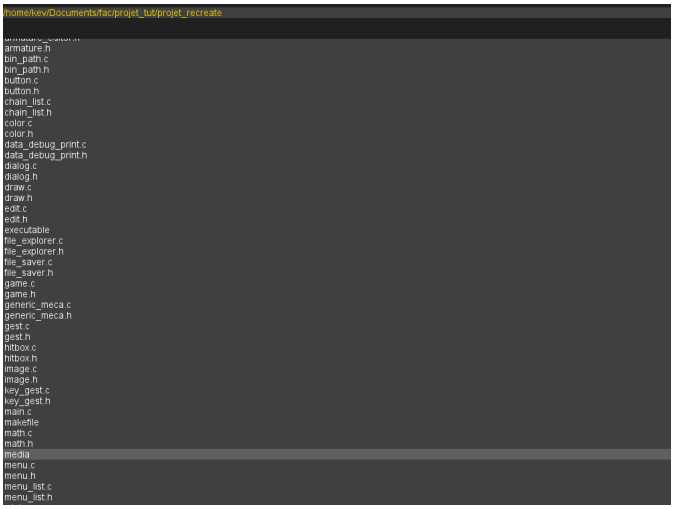

FIGURE 5.15 – *Sauvegarde d'un fichier*

5.4.3 Simulation d'un personnage

Une fois les formes, les animations, touches et déplacements définis pour un personnage on peut tester son fonctionnement dans un mode de simulation solo.
A droite après l'éditeur de mouvements, pour sortir du mode simulation on presse la touche **Echap**.
Ce mode permet de visualiser le comportement des animations en plein combat, la combinaison de déplacements et trouver une manière de jouer le personnage et le contrôler.

FIGURE 5.16 – *Test des actions du personnage en solo*

5.4.4 Combat

Une fois prêt au combat, il ne restera plus qu'à entrer dans l'arène et combattre un second joueur sur le clavier ou un ordinateur un peu délirant.

FIGURE 5.17 – *Combat entre deux personnages*

Chapitre 6

Bilan du projet

6.1 Perspectives

Le projet est encore améliorable, c'est-à-dire que des améliorations sont proposées en tant qu'étape 5.
On pourra aussi étudier à nouveau les mécanimes du jeu pour les optimiser et diminuer les temps de calculs :

- Améliorer l'affichage des images par rotations (Il est déjà fait en sorte de ne pas calculer deux images successives dont l'angle est proche à moins 1 degré)
- Améliorer la simplification des HitBox pour diminuer leur nombre tout en gardant une cohérence qui ne choque pas à l'oeil
- Rendre les éditeurs simples d'utilisation tout en gardant un grand nombre de fonctionnalités
- Gérer les retours arrière

6.2 Conclusion

En conclusion, le projet partait pour être ambitieux, long et demandait de bien tout gérer par le choix d'un langage de bas niveau qu'est le C.
Les éléments d'interface du projet comme les boutons, ou l'explorateur de fichiers manquent de design, mais ce n'était pas le coeur du projet.
Malgré des difficultés de gestion du temps pour finir le projet, en suivant la planification des différentes tâches et étapes, il a été assez simple d'avancer le projet et aboutir à un projet acceptable.

Chapitre 7

Annexes

7.1 Tangentes externes entre deux cercles

Précisons la résolution du problème des tangentes externes entre deux cercles et montrons que la complexité de la fonction résolvant le problème est de complexité constante.

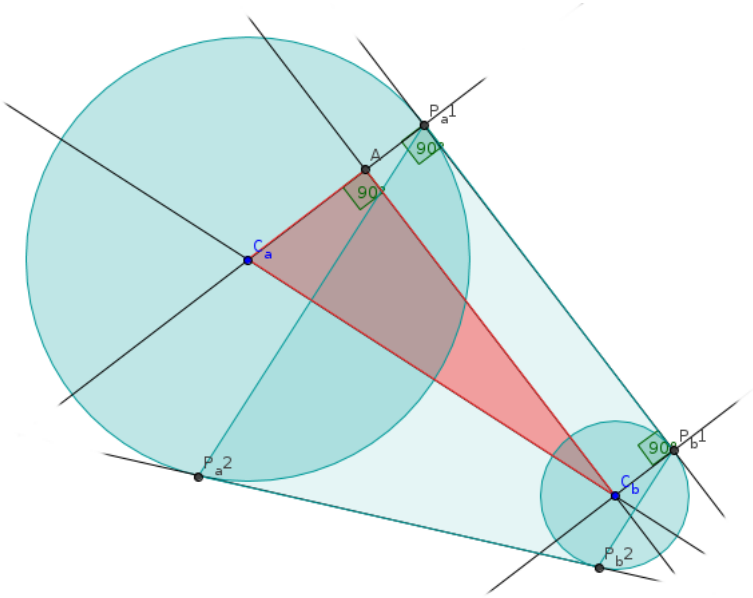

FIGURE 7.1 – *Tangentes externes entre deux cercles*

Soit le cercle de centre $C_\alpha \in \mathbb{R}^2$ et de rayon $R_\alpha \in \mathbb{R}^+$
Soit le cercle de centre $C_\beta \in \mathbb{R}^2$ et de rayon $R_\beta \in \mathbb{R}^+$
Posons les points $P_\alpha, P_\beta \in \mathbb{R}^2$ tels que $(P_\alpha P_\beta)$ soit une tangente externe entre les cercles définis précédemment.
Une telle droite existe si les cercles ne sont pas contenus l'un dans l'autre.
Dire que P_α appartient au premier cercle est équivalent à dire qu'il existe $\theta \in \mathbb{R}$ tel que :

$$P_\alpha = R_\alpha e^{i\theta} + C_\alpha$$

Or $(P_\alpha C_\alpha)//(P_\beta C_\beta)$ et une droite coupe deux droites parallèles en le même angle, alors :

$$P_\beta = R_\beta e^{i\theta} + C_\beta$$

Prenons le triangle rectangle en rouge sur notre figure, d'après le théorème de Pythagore :

$$d(C_\alpha, C_\beta)^2 = (R_\alpha - R_\beta)^2 + d(P_\alpha, P_\beta)^2$$

Or

$$d(P_\alpha, P_\beta)^2 = (P_{\alpha x} - P_{\beta_x})^2 + (P_{\alpha y} - P_{\beta_y})^2$$

$$= (R_\alpha cos(\theta) + C_{\alpha x} - R_\beta cos(\theta) - C_{\beta_x})^2 + (R_\alpha sin(\theta) + C_{\alpha y} - R_\beta sin(\theta) - C_{\beta_y})^2$$

$$= ((R_\alpha - R_\beta)cos(\theta) + (C_{\alpha x} - C_{\beta_x}))^2 + ((R_\alpha - R_\beta)sin(\theta) + (C_{\alpha y} - C_{\beta_y}))^2$$

$$\begin{aligned} &= (R_\alpha - R_\beta)^2 cos^2(\theta) + 2(R_\alpha - R_\beta)(C_{\alpha x} - C_{\beta_x})cos(\theta) + (C_{\alpha x} - C_{\beta_x})^2 \\ &+(R_\alpha - R_\beta)^2 sin^2(\theta) + 2(R_\alpha - R_\beta)(C_{\alpha y} - C_{\beta_y})sin(\theta) + (C_{\alpha y} - C_{\beta_y})^2 \end{aligned}$$

$$\begin{aligned} &= (R_\alpha - R_\beta)^2(cos^2(\theta) + sin^2(\theta)) + 2(R_\alpha - R_\beta)(C_{\alpha x} - C_{\beta_x})cos(\theta) + (C_{\alpha x} - C_{\beta_x})^2 \\ &+2(R_\alpha - R_\beta)(C_{\alpha y} - C_{\beta_y})sin(\theta) + (C_{\alpha y} - C_{\beta_y})^2 \end{aligned}$$

$$\begin{aligned} &= (R_\alpha - R_\beta)^2 + 2(R_\alpha - R_\beta)(C_{\alpha x} - C_{\beta_x})cos(\theta) + (C_{\alpha x} - C_{\beta_x})^2 \\ &+2(R_\alpha - R_\beta)(C_{\alpha y} - C_{\beta_y})sin(\theta) + (C_{\alpha y} - C_{\beta_y})^2 \end{aligned}$$

Or

$$d(C_\alpha, C_\beta)^2 = (R_\alpha - R_\beta)^2 + d(P_\alpha, P_\beta)^2$$

D'où

$$\begin{aligned} (C_{\alpha x} - C_{\beta_x})^2 + (C_{\alpha y} - C_{\beta_y})^2 &= (R_\alpha - R_\beta)^2 + (R_\alpha - R_\beta)^2 + 2(R_\alpha - R_\beta)(C_{\alpha x} - C_{\beta_x})cos(\theta) \\ &+(C_{\alpha x} - C_{\beta_x})^2 + 2(R_\alpha - R_\beta)(C_{\alpha y} - C_{\beta_y})sin(\theta) + (C_{\alpha y} - C_{\beta_y})^2 \end{aligned}$$

$$-2(R_\alpha - R_\beta)^2 = 2(R_\alpha - R_\beta)(C_{\alpha x} - C_{\beta_x})cos(\theta) + 2(R_\alpha - R_\beta)(C_{\alpha y} - C_{\beta_y})sin(\theta)$$

$$-(R_\alpha - R_\beta) = (C_{\alpha x} - C_{\beta_x})cos(\theta) + (C_{\alpha y} - C_{\beta_y})sin(\theta)$$

Posons :

$$t = tan(\frac{\theta}{2})$$

Résolvons une forme $asin(\theta) + bcos(\theta) = c$ où :

$$a = (C_{\alpha y} - C_{\beta_y})$$

$$b = (C_{\alpha x} - C_{\beta_x})$$

$$c = -(R_\alpha - R_\beta)$$

$$a\frac{2t}{1+t^2} + b\frac{1-t^2}{1+t^2} = c$$

$$2at + b(1-t^2) = (1+t^2)c$$

$$2at + b - bt^2 = c + ct^2$$

$$-(b+c)t^2 + 2at + b - c = 0$$

$$\Delta = 4a^2 + 4(b+c)(b-c) = 4(a^2 + b^2 - c^2)$$

Si $b \neq -c$

$$tan(\frac{\theta}{2}) = t = \frac{a \mp \sqrt{a^2 + b^2 - c^2}}{b+c}$$

$$\theta = 2tan^{-1}(\frac{a \mp \sqrt{a^2 + b^2 - c^2}}{b+c})$$

Si $b = -c$

$$-(b-b)t^2 + 2at + b + b = 0$$

Si $a \neq 0$

$$t = \frac{-b}{a}$$

$$\theta = 2tan^{-1}(\frac{-b}{a})$$

Si $a = 0$

$$bcos(\theta) = -b$$

$$\theta = \pi$$

On a donc trouvé l'angle θ permettant de déterminer les coordonnées des points P_α et P_β.
Les calculs mathématiques ci-dessus se font en complexité constante, on peut donc calculer les tangentes externes entre deux cercles en complexité constante.