

Université Paris 8

Master Arts

Mention : *Arts Plastiques et Art Contemporain*

Spécialité : *Arts et Technologies de l'Image Virtuelle*

Appréhender les personnages en 3D : quelques principes
fondamentaux du rig

Rodolphe Zirah

Mémoire de Master 2

2013- 2014

Résumé

Le *rig* est un enjeu majeur de la production des films d'animation en images de synthèse d'aujourd'hui, tout en étant caché dans l'ombre du travail de l'animateur. Pourtant, pour qu'une animation soit réussie, il faut que le *rig* soit bien pensé. Il faut donc créer son *rig* en pensant à rendre le travail des animateurs plus simple et intuitif, et certains choix du rigueur peuvent influencer fortement sur la qualité de l'animation et sur le temps de production.

Cette étude se découpe en trois parties. Nous commencerons par un état de l'art bref qui nous montrera que le *rig* existe bien avant la 3D, avec les créatures en volume de toutes l'histoire du cinéma d'animation. Nous discuterons ensuite de technique de *rig* à proprement parler. Nous donnerons, dans une deuxième partie, les principes fondamentaux nécessaires à la bonne compréhension et à la bonne réalisation d'un *rig*, en général. Puis nous détaillerons, dans une troisième partie, des cas particuliers de *rig* - personnage mécanique et *rig* facial- à partir d'étude de productions. On utilisera Maya (2013) comme outil principal.

Abstract

Nowadays, character rigging is one of major issue of animated films, while hidden in the shadow of animation work. Yet, for a movie to be successful it is necessary that the rig is well thought out. We must therefore keeping in mind that rig is made to make the animator's job easier and intuitive, and some choices made by riggers may strongly influence the quality of the animation and production.

This study is divided into three parts. First by establishing a state of the art and studying the first creatures in history of animation, we will see that the rig is well before 3d. Then we discuss about rigging technics. In the second part we will talk about fundamental principles to make a rig. Then in a third part will explore particular case of production. We use Autodesk Maya as main tool.

Sommaire

Introduction	p.1
I/ Etat de l'art	p.2
II/ Principes fondamentaux, les enjeux d'un bon rig	p.7
2.1 Les hiérarchies, la notion de parentage	p.7
2.2 Le skinning	p.8
2.3 Orientation des joints	p.12
2.4 Gimbal lock et rotate order	p.15
2.5 Le fk/ik	p.16
2.6 La création de contrôler	p.19
2.7 Le squash and stretch.....	p.23
2.8 Le twist.....	p.28
2.9 La main	p.31
2.10 Finaliser un rig	p.33
III/ Etude de cas	p.34
3.1 Reebot	p.35
3.1.1 Rig d'un personnage mécanique	p.37
Roues et amortisseurs	p.37
Ribbon spine	p.41
Préservation du volume	p.44
Le corps, les hélices la courroie	p.46
Les bras coulissants	p.47

3.1.2	Intégrer son rig au sein d'un pipeline de film d'animation	p.48
3.2	Elaïa : rig facial d'un personnage cartoon	p.50
	Stylisé ou réaliste	p.51
	Une bonne topologie	p.53
	Expression faciale	p.54
	Placer les joints et les contrôleurs	p.59
	Création des expressions faciales, « skinning » et « set driven keys »	p.62
	Conclusion	p.64
	Bibliographie / Webographie	p.65

Introduction

En informatique graphique le *rig* est un ensemble de joints et de contrôleurs qui permettent de mettre en pose un personnage. En comparaison à l'animation traditionnelle 2D où l'animateur est le dessinateur, dans l'animation 3D, le personnage créé par le *character designer* et le modelleur doit passer d'abord par les mains du *rigger* avant de rejoindre celles de l'animateur afin que tous les systèmes qui permettront de l'animer soient mis en place. Il faut donc créer son *rig* en pensant à rendre le travail des animateurs plus simple et intuitif. Car si le *rig* est mal fait, l'animateur risque de «se battre» contre lui, dans le sens où il peut être plus laborieux pour lui de positionner l'objet qu'il souhaite animer dans l'espace si les axes de rotation, de translation ou de *scale* ne sont pas correctement placés ou orientés. Cette différence entre un bon et un mauvais *rig* peut influencer de manière considérable sur le temps que l'animateur va mettre pour effectuer son action et donc influencer aussi fortement sur le temps de production.

Depuis les premiers développements des logiciels 3D cette discipline du *rig* n'a jamais cessé d'évoluer et de nombreuses techniques sont apparues, mais on a pu voir des concepts similaires bien avant la 3D, dès lors qu'on a pensé faire de l'animation en volume, ou *stop motion*. Tout d'abord je présenterai un état de l'art général, qui nous donnera quelques références clefs.

Nous discuterons ensuite de technique de *rig* à proprement parler. Il est difficile de parler de la réalisation d'un *rig* sans connaître au préalable les concepts de base. C'est pourquoi il nous faudra d'abord discuter des principes fondamentaux du *rig*, pour aborder par la suite les façons de penser la réalisation d'un *rig*. Un certain nombre de questions se posent. Tout d'abord Que vais-je *rigger* ? S'agit-il d'un personnage, ou, par opposition, d'un objet ou d'une machine ? Qu'est-ce qu'un joint, comment l'orienter ? Qu'est-ce que le *skinning* ? Etc. Nous n'allons pas présenter la totalité des techniques, mais aborder l'exemple d'un bras qui nous permettra de comprendre un certain nombre de techniques qui pourront ensuite s'appliquer à tous types de *rig*. Le logiciel Maya (2013) sera mon outil principal.

Je reviendrai ensuite sur différentes méthodes que j'ai pu employer pendant mes projets pour présenter des cas particuliers de *rig*. La dernière partie reviendra sur deux projets de film d'animation assez différents, l'un traitera du *rig* d'un personnage mécanique et l'autre, à l'opposé, traitera de la partie la plus organique qui soit : le visage.

I/ Etat de l'art

Tout au long de l'histoire de l'animation furent mises en place diverses techniques qui ont permis de donner vie à des personnages, animaux et autres créatures imaginaires. Il s'agit pour moi de les situer et d'expliquer les techniques qui me semblent les plus importantes et qui ont amené à celles utilisées aujourd'hui dans l'animation 3D. Souvent marquée par l'apparition d'un film de court, moyen, ou long métrage, l'histoire des techniques de l'animation est très vaste et les artistes du monde de l'audiovisuel y ont été très prolifiques. C'est pourquoi je ne vais pas dresser toute cette histoire bien que passionnante, mais plutôt axer mes recherches sur comment animer des modèles en volumes. Car on a pensé le *rig* bien avant que la 3D existe.

En effet les systèmes élaborés dans les films de marionnettes du polonais Ladislav Alexandrowitch Starewitch, qui ont par ailleurs marqués fortement l'histoire des films d'animations de marionnettes, se rapprochent à bien des égards des systèmes mis en place dans les outils contemporains d'informatique graphique. Ses personnages sont dotés d'armatures en bois, de chevilles et de vis pour les articulations, les visages fabriqués en peau peuvent être déformés pour produire différentes expressions et donner plus de réalisme à ses personnages.



Ladislav Starewitch, *Le Roman de Renard* (1911)

Les armatures en fil de fer de l'américain Willis O'Brien datent environ de la même époque, et servent à mettre en pose ses créatures, qui inspireront tous les futurs réalisateurs de films fantastiques. Elles deviendront des armatures en bois recouvertes de pâte à modeler pour représenter la peau, comme dans le film *Ghost of Slumber Mountain* (1918) où il met en scène son premier dinosaure. Il développera pour *The Lost World* (1925) un procédé de fabrication de marionnettes toujours utilisé aujourd'hui. En s'inspirant d'un squelette de dinosaure, il fera usiner des os en métal, munis de billes d'acier à chaque extrémité pour pouvoir les articuler. La peau est alors constituée de couches de latex, et certains dinosaures seront même dotés de ballons à l'intérieur de leur cage thoracique pour donner l'illusion de la respiration. Ce système d'ossature très sophistiqué se rapproche par certains points du travail du *rigger*. C'est grâce au succès de ce film, et au réalisme de ses créatures articulées, que Willis O'Brien pourra produire huit ans plus tard son film le plus marquant *King Kong* (1933).



Willis O'Brien, *The Lost World* (1925)

Ray Harryhausen, digne successeur d'O'Brien, est un personnage marquant de l'histoire des effets spéciaux car il invente la *Dynamation*, principe qui lui permettra d'intégrer ses marionnettes dans des images en prises de vues réelles, plutôt que de les capter dans des maquettes, image par image. Il est alors le pionnier du *compositing* et de l'intégration. Je ne rentrerai pas dans les détails de ces techniques car ce n'est pas mon sujet, mais il fit aussi évoluer de manière remarquable ses marionnettes. Il mit au point des techniques que l'on peut retrouver dans les logiciels modernes. Celle qui pour moi est marquante est un système qui ressemble à celui d'O'Brien : on retrouve les squelettes en acier mais Harryhausen a rajouté des pièces en caoutchouc attachées aux os pour reproduire les muscles. Les mouvements de la peau sur les muscles permettront ainsi d'obtenir des déformations réalistes au niveau des articulations. Problématique commune aux *rigger* d'aujourd'hui.



Squelette de *King Kong* (1933)

Il m'est impossible de terminer cette partie historique sans parler de Phil Tippett. Fortement inspiré par Willis O'Brien et Ray Harryhausen, il travaille pour Georges Lucas au sein d'ILM où il développera le *go-motion*. Ce système a été une avancée considérable du stop motion car il permit d'obtenir du flou de bougé, il fut développé notamment pour le film *The Empire Strikes Back* deuxième volet de la saga *Star Wars* en 1980. Phil Tippett sera aussi un pionnier de

l'image de synthèse, avec le film *Jurassic Park*. Son studio d'effets spéciaux fut le premier à abandonner le système pour passer sur des Apple II, il marquera alors dès 1993 le passage des techniques traditionnelles aux images virtuelles.

Nous arrivons aux premiers pas du *rig* appliqué aux images de synthèse, ou plutôt à parler d'animation. L'animation du corps humain a été un domaine de recherche majeur dès les premières images de synthèse. Plusieurs chercheurs de l'université de l'Ohio ont mené des recherches sur l'animation du corps humain dès le début des années 80, et ils ont présenté l'avancée de leurs recherches avec des films au SIGGRAPH. *Walking skeleton*¹ montre un squelette humain brut, pas habillé, pour étudier la marche, les sauts, etc. L'animation est encore difficile. En 1985, on a pu aussi voir à Motion studies², qui présente des recherches qui vont être menées sur plusieurs années, pour réaliser l'animation de personnages de type humain et d'animaux, pour donner lieu à un film, *Eurythmy*³, où l'on voit à la fois des hommes courir, danser, des animaux de type chiens courir, ainsi qu'un vol d'oiseaux géré par intelligence artificielle. Ces travaux d'animation formeront la base du logiciel Character studio, puis seront intégrées dans 3DS.

Le premier personnage à être présent dans film 3D est André, un groom au teint verdâtre qui fait sa bat avec une abeille. *The Adventures of André and Wally B.* sort en 1984 et c'est le premier court métrage de Pixar, animé par John Lasseter. C'est encore Lasseter qui dessinera et animera le premier personnage en 3D animé dans un long métrage, il s'agit du chevalier de verre dans *Young Sherlock Holmes* (1985). On voit que les techniques sont encore balbutiantes, les personnages ont du mal à bouger, ils sont en verre, ou en plastique (*Ton Toy*, Pixar, 1988) pour contrebalancer leur manque de souplesse. Pour pallier aux contraintes liées à l'animation, on a vu souvent des personnages géométriques, très stylisés, sans textures et sans ombres, comme en France où la première série de la société Mac Guff présente *La vie des bêtes* (1986) en caméra subjective, sans qu'on ne voit jamais aucun animal ; ou dans la série *Les fables géométriques* (Fantôme - 1989) où les membres des personnages ne sont pas raccordés entre eux.

¹ *Walking skeleton*, David Zeltzer, Ohio State University, 1985.

² *Motion studies*, Susan Amkraut, Michael girard, george karl, Ohio State University, 1985.

³ *Eurythmy*, Susan Amkraut, Michael girard, george karl Ohio State University, 1989.



Young Sherlock Holmes (1985)

Petit à petit les techniques de base ont été développées, puis améliorées, et aujourd'hui, on peut trouver des papiers techniques sur toutes les problématiques du *rig* : techniques modulaires⁴, autorig⁵, le rig dans les jeux vidéo⁶, etc. La majorité des références que j'utilise pour travailler sont néanmoins des tutoriaux, il en existe énormément, et c'est sur des tutoriaux que je vais m'appuyer la plupart du temps. Comme celui de Aaron Holly, « Monkey Body Setup : Rigging for feature animation »⁷, qui est certainement le plus connu et le plus indispensable.

4 J. Smith & J. White, « BlockParty: modular rigging encoded in a geometric volume » ILM, Proceeding ACM SIGGRAPH Sketches, n°115, 2006.

5 Pantuwong, N. & Sugimoto, M. « A fully automatic rigging algorithm for 3D character animation » dans ACM Press, 2011.


6 McLaughlin, T., Cutler, L. & Coleman, D. « Character rigging, deformations, and simulations in film and game production ». dans 1–18 (ACM Press, 2011)

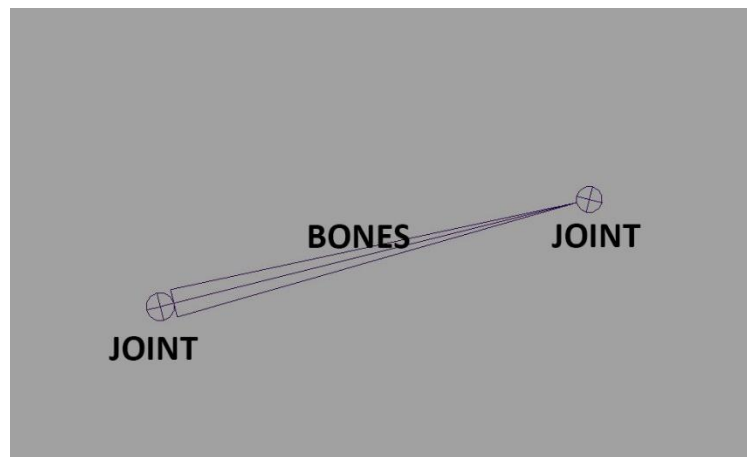
⁷ Aaron Holly, « Monkey Body Setup rigging for feature animation », Fahrenheit.

II/ Principes fondamentaux du rig

A partir d'un exemple assez complexe, celui du bras, nous allons aborder dans cette partie les concepts fondamentaux liés *rig* : le parentage, le *skinning*, l'orientation des joints, les *gimbal lock* et le *rotate order*, le FK-IK, la création de contrôleurs, le *squash and stretch*, le *twist*, et pour terminer le bras on riggera la main.

2.1 Les hiérarchies et la notion de parentage

Qu'est-ce qu'un joint, et comment le créer ? L'outil principal du rigger développé dans Maya est le joint, il permet de créer des hiérarchies d'objets et leurs parentages, qui vont ensuite permettre de créer le squelette et les articulations de notre objet, ici un bras. Dans maya la création des joints se fait avec le *Joint Tool* , les joints se créent de manière hiérarchique, si un joint est parenté sous un autre joint, ces deux joints seront alors connectés par un os (*bone*).



A noter que les *bones* ne sont qu'une représentation graphique des connections entre les joints, ce ne sont pas des objets Maya. Il n'est donc pas possible de manipuler un *bone*. Ce mode de visualisation graphique des connections nous permet de comprendre de façon imagée que les joints représentent les articulations de notre personnage et que ces articulations sont connectées par des os, comme cela serait le cas chez l'être humain. Il sera donc question de placer des joints

à l'intérieur d'un modèle 3D comme s'il s'agissait d'un squelette, de manière à ce que chaque joints correctement placés et orientés permette le mouvement de l'articulation souhaitée.

Une chaîne de joints (ou squelette), est toujours constituée d'un joint racine « root joint », et de « joints enfants ». Il est important de bien comprendre cette notion de parentage : l'objet racine est positionné par rapport au monde, ses coordonnées seront toujours relatives au centre du monde, alors que les objets parentés eux prennent leurs coordonnées par rapport à la position de l'objet à l'intérieur duquel il se trouve. On parlera alors de position locale.

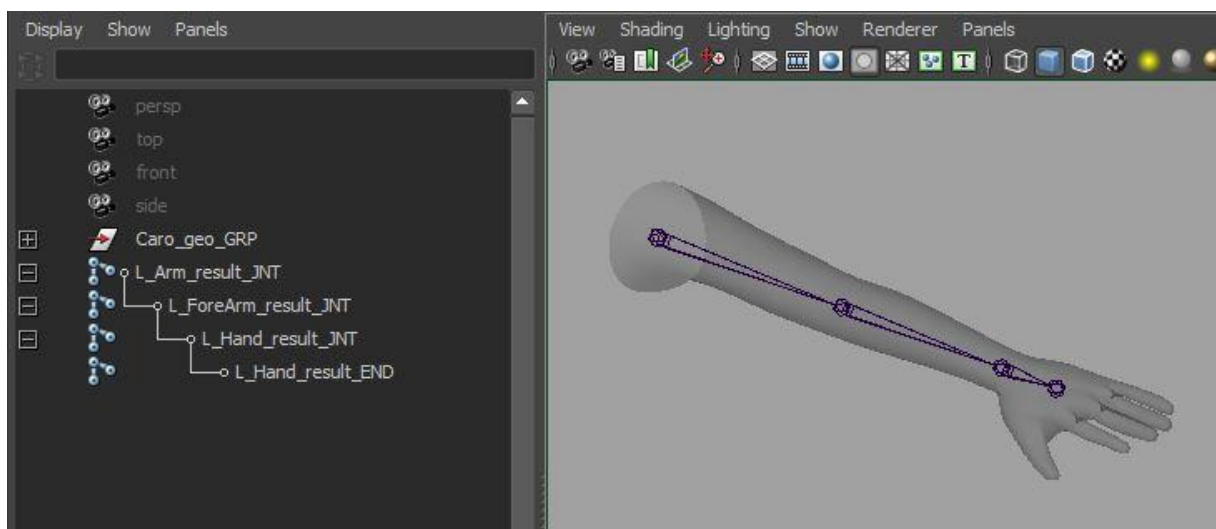


Fig. 1. Exemple de hiérarchie et de nomenclature pour le rig d'un bras

2.2 Le skinning

Maintenant que nous savons créer des joints, l'étape suivante est de savoir comment contrôler notre géométrie et comment l'attacher aux joints. Le skinning vient du terme anglais « skin », il consiste à attacher la géométrie aux joints comme s'il s'agissait d'une peau. Dans Maya la géométrie est attachée aux joints avec le node « skinCluster ». Chaque vertex est liés aux joints par une valeur, on peut se représenter cette valeur comme une contrainte de position. Plus la valeur est grande, plus le vertex est emporté par le joint. Nous verrons plus en profondeur à quoi correspondent ses valeurs dans la suite de ce chapitre.

Tip : Pour créer un node « skin cluster », sélectionner tous les joints qui doivent influencer notre peau puis la géométrie, dans le menu animation aller dans l'onglet Skin > Bind Skin > Smooth Bind .

Pour bien comprendre ce qu'est le skinning il est important de revoir les options que nous propose Maya pour le « Smooth Bind. » :

- **Bind to** : Par défaut Maya attache la géométrie sur tous les joints de notre hiérarchie, pour cette option je préfère utiliser « selected joint », j'obtiens plus de contrôle en étant sûr que les influences ne s'appliquent qu'aux joints que j'ai sélectionné.
- **Bind method** : avec l'option par défaut « Closest Distance » les valeurs d'influences sont déterminées par la distance à laquelle les vertex se trouvent par rapport aux joints. Cependant la seconde option « closest in hierarchy » apporte une condition supplémentaire, elle permet à Maya de prendre en compte la hiérarchie de la chaîne de joint (en plus de cette distance du vertex au joint). Elle évite ainsi qu'une autre chaîne de joints, qui serait placée à côté de celle que nous voulons skinner, ne vienne influencer la géométrie.
- **Skinning method** : deux modes de skinning existent, « linéaire » et « dual-quaternion ». Le mode dual-quaternion a été introduit dans Maya 2010 pour palier à un problème de déformation de la peau lors du calcul des rotations linéaires notamment quand on tourne les joints sur leur axe principal, ce qu'on appelle « twist » (on le verra plus bas). Je ne rentrerai pas dans les détails mais de manière générale il faut voir les quaternions comme une autre méthode mathématique pour calculer les rotations dans un espace 3D, qui permet le twist.

J'utilise la majeure partie du temps le mode linéaire, car je préfère gérer le twist moi-même. Le dual-quaternion peut être pratique en additif mais pour le reste je trouve qu'on n'a pas assez de contrôle. Pour l'utiliser en mode additif il existe le mode « Weight Blended » (qui pourrait être utile pour skinner une épaule par exemple). Il est à noter

que si vous travailler sur un projet de jeux vidéo, la plupart des moteurs de jeux ne prennent pas encore en compte les quaternions.

- **Normalize weights** : c'est le mode de normalisation des poids d'influence. Des valeurs normalisées sont toutes comprises entre 0 et 1, et la somme de ces valeurs est toujours égale à 1. De cette manière on conserve la proportionnalité des valeurs. On peut demander à Maya de normaliser les valeurs en choisissant l'option « Classic Linear ». Une autre option existe « Post », elle permet d'ajouter des valeurs après que les valeurs aient été normalisées, les valeurs pourront alors dépasser 1. Cette autre option à mon sens n'est pas pratique car il est très difficile de comprendre des valeurs d'influences non normalisées. Dans le souci d'un meilleur contrôle sur ces valeurs je préfère donc toujours utiliser le mode « Classic Linear ».
- **Max influences** : Correspond au nombre maximum de joints autorisés à influencer une géométrie. Sa valeur par défaut est 5, ce qui est préconisé dans la documentation Maya pour obtenir les meilleurs résultats. D'une manière générale, quand on prend un personnage qui est composé d'un grand nombre de joints, certains tutoriaux préconisent de garder une valeur assez basse (3 ou 4), pour réduire les zones d'influences par joints et ainsi obtenir un meilleur « skinning » par défaut. Ceci est vrai, en baissant cette valeur le résultat est un peu meilleur, mais de mon point de vue le skinning par défaut de Maya n'est jamais très propre. Je préfère maintenir le « Max influence » à une valeur plus élevée, de 6 à 8, voire plus en fonction du nombre de joints et je vais expliquer pourquoi. Lorsque l'on assigne une valeur basse, le résultat par défaut est donc meilleur, mais on s'expose à un autre problème : lorsque l'on souhaite revenir sur notre skinning et attribuer des valeurs à un joint *à posteriori* (c'est à dire qui n'était pas pris en compte au départ), Maya, qui va essayer de conserver le nombre maximum de joints influençant mes vertex va retirer des valeurs à un joint sans que je puisse contrôler ce choix. Je verrai alors « sauter » mes vertex de manière indésirable. Ce problème est très fréquent lorsque l'on fait ses premier skinning, et a de quoi nous rendre fou si l'on ne comprend pas ce qui est en train de se passer. C'est pourquoi maintenir un « Max influence » élevé, et donc un nombre de joints élevé, évitera que ces sauts de vertex se produisent.

On vient de présenter toutes les options du skin cluster. Mais comment régler les valeurs d'influences des joints sur la géométrie, et comment visualiser ces valeurs ?

La première façon (Fig.2) consiste à sélectionner les vertex de la géométrie et d'aller voir les valeurs de l'attribut « smooth skin » dans la fenêtre « Component Editor » dans cette fenêtre on peut voir sous forme de tableau les valeurs d'influence de chaque joint sur chaque vertex, ainsi que le total des valeurs par vertex.

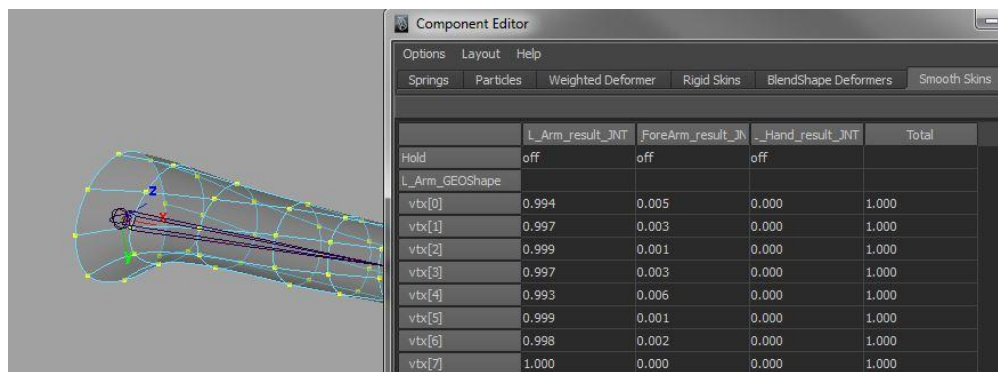



Fig.2.

La seconde façon (Fig.3.) pour visualiser et régler les valeurs d'influence est d'utiliser le « paint skin weights tool » . Cet outil nous permet de peindre directement sur la géométrie la valeur d'influence des joints et de visualiser par un dégradé cette carte d'influence. Je trouve cet outil plus visuel, et il permet de régler plus facilement les valeurs d'influences. À mon avis, manipuler les valeurs de chaque vertex dans le « Component Editor » c'est manipuler une suite de nombres pour sculpter notre géométrie, ce qui me paraît être une approche complexe, moins graphique, et donc moins pratique pour moi. Peindre les influences directement sur la géométrie est une approche bien plus artistique, bien plus proche de la matière (c'est comme si tu sculptais directement la géométrie).

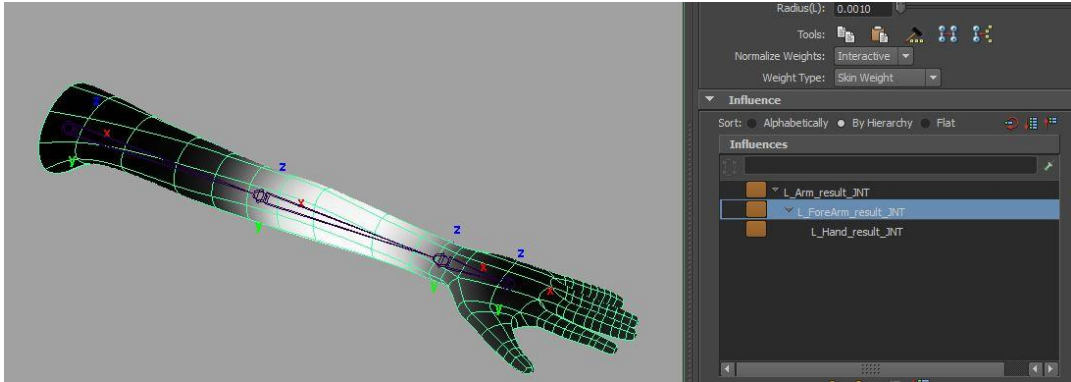


Fig.3.

Tip : utiliser le mode « Classic Linear », garder un « Max influence » élevé et « Maintain max influence » coché, retravailler son skin avec le « paint skin weights tool » et ensuite si l'on souhaite éliminer les petites valeurs parasites utiliser la fonction Maya « Prune Small Weight ».

2.3 Orientation des joints

Tip : dans Maya seuls le mode « gimbal » pour l'outil de rotation, et le mode « objet » pour l'outil de translation vous permettront de faire des rotations et des translations de manière correcte selon l'axe des joints.

Il faut toujours garder en tête que le rig est amené à être manipuler :un joint mal orienté peut ainsi entrainer des rotations sur plusieurs axes, ce qui peut rendre le travail de l'animateur très difficile. Il sera beaucoup plus simple pour lui de contrôler la rotation des articulations de son rig dans ces courbes d'animations sur un seul axe de rotation (Fig.4.).

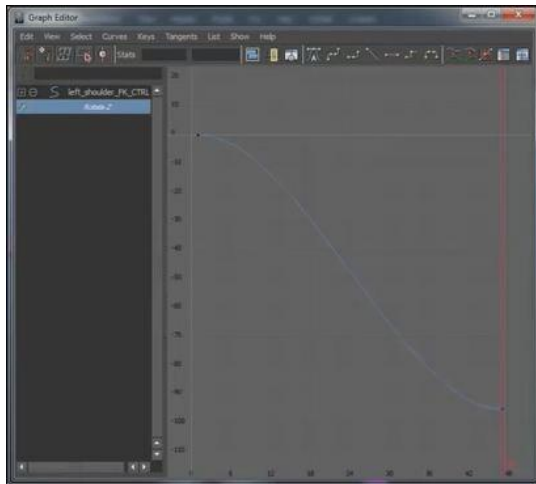
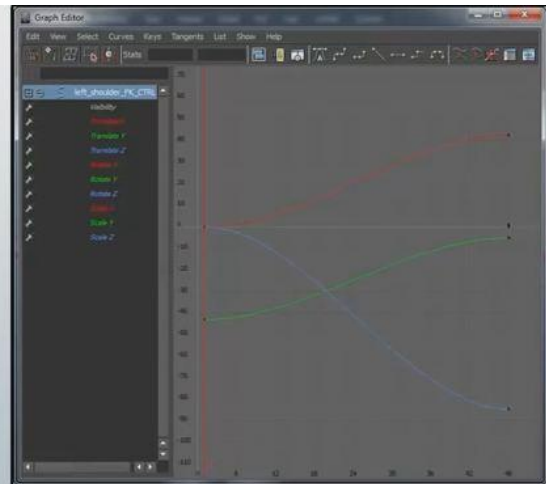


Fig.4. Rotation sur axe simple



Rotation sur axes multiples

Il est aussi important que les valeurs de rotation de chaque joint soient cohérentes par rapport au mouvement que l'on souhaite obtenir au final. Il est préférable de conserver des valeurs positives dans le sens où l'articulation va plier naturellement. Préserver ces valeurs positives permettra de conserver des valeurs logiques lors de l'animation.

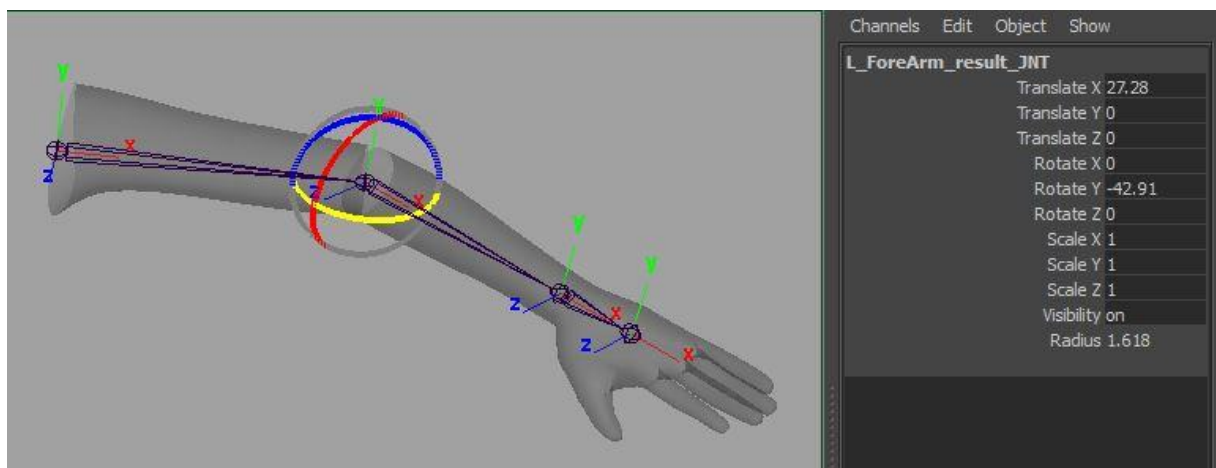


Fig.5. Exemple d'un bras, les valeurs rotate Y sont négatives dans le sens où le bras plie, les joints sont donc mal orientés

Dans cet exemple (Fig5.), on voit que les joints du bras sont mal orientés. Mais comment les orienter? Afin d'obtenir une rotation en Y positive, il suffit de faire une rotation à 180 degrés sur l'axe des X dans l'attribut « joint orient » du premier joint de notre chaîne. Les joints enfants suivront eux l'orientation de leur parent. Cet attribut permet d'orienter les joints sans pour autant leur donner de valeur dans les axes de rotation « rotate ».

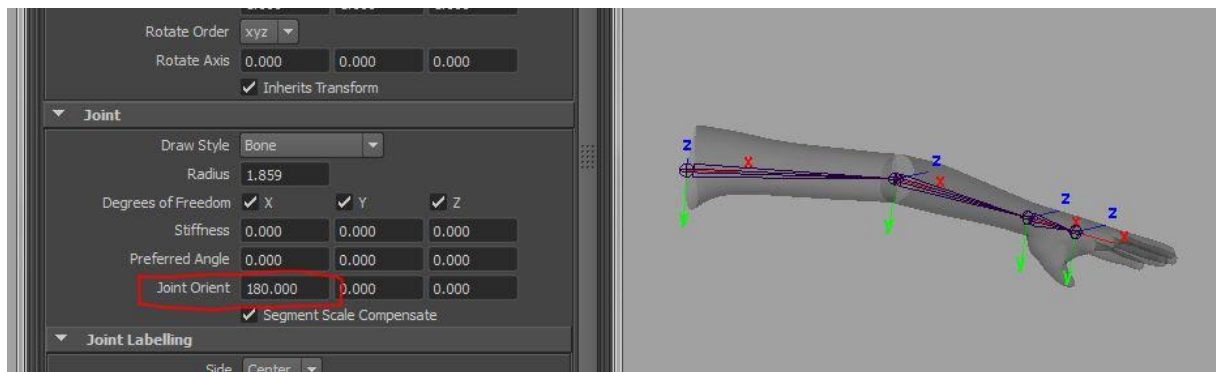


Fig.6. L'attribut « joint orient » permet de ré-orienter les joints sans leurs donner de valeur dans les axes de rotations

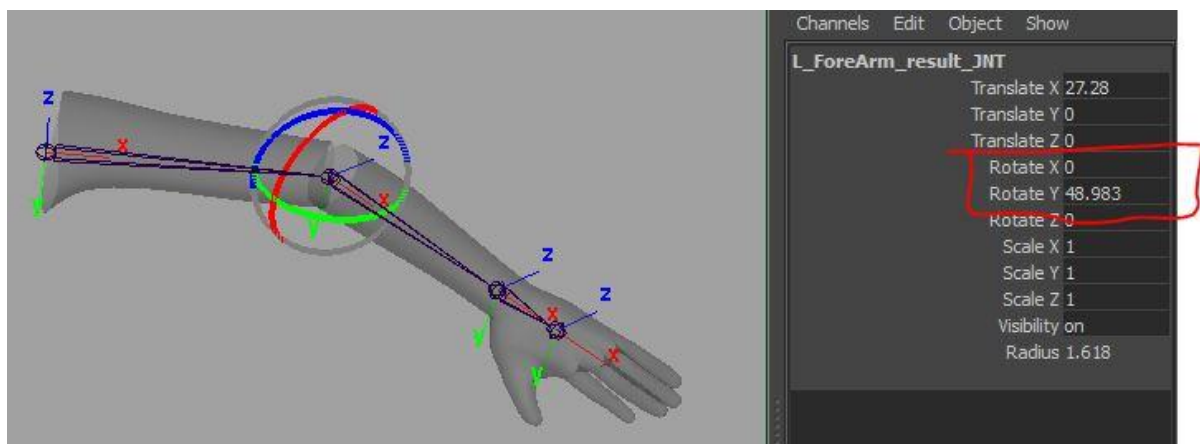


Fig.7. Les joints sont convenablement orientés, le plis du coude se fait sur le rotate Y et est positif.
Si on abaisse l'épaule la valeur en rotate Z est aussi positive

De manière générale j'utilise toujours l'attribut « Joint Orient » pour orienter les joints, il existe cependant d'autre méthodes : il y a l'outil « Orient Joint » de Maya, ou encore celui de Michael Comet, qui est un très bon outil *cometJointOrient.mel*⁸.

⁸ Michael Comet, cometJointOrient.mel, [en ligne], <http://www.comet-cartoons.com/melscript.php> (consulté en mai 2014)

Tip : pour afficher les axes de rotation locaux des joints, sélectionner le joint puis Display > Transform Display > Local Rotation Axes.

2.4 Le gimbal lock et le rotate order

Nous allons être confronté à un autre problème, le « gimbal lock ». Il apparaît quand deux axes de rotation se superposent. Le mode gimbal de l'outil de rotation de Maya est l'unique mode de rotation qui permette d'afficher la superposition des axes de rotation. Quand deux axes sont superposés, nous confondons les valeurs de ces deux axes qui résulteront du même mouvement, nous perdons donc de ce fait une possibilité de mouvement.

Dans Maya les rotations locales se calculent les unes par rapport aux autres, il faut donc définir une ordre de priorité des axes de rotation : le « rotate order ». Pour éviter la superposition de deux axes, il faut penser à conserver une logique de mouvement dans les rotations. Comment fonctionne cet ordre de rotation ? Pour le comprendre il faut lire le « rotate order » à l'envers : quand mon « rotate order » est XYZ, l'axe Z sera prioritaire sur les deux autres axes, ensuite viendra le Y puis le X (l'axe X n'emporte donc aucun axe). Prenons les trois articulations de notre bras comme exemple :

- L'épaule : imaginons notre personnage en pose T. Pour une animation de marche, l'épaule devra d'abord se baisser pour arriver en pose A, puis donner la possibilité au bras de se balancer d'avant en arrière, puis elle doit aussi pouvoir tourner sur son axe principal. Selon la manière dont les joints sont orientés, l'axe Z monte et baisse le bras, l'axe Y le balance d'avant en arrière, et le X peut bouger sur son axe principal. Pour ce type d'animation mon rotate order sera de ce fait XYZ.

Cependant il faut faire attention, l'épaule est une articulation qui peut tourner à 360°, il n'y aura donc pas de rotate order idéal qui me permettra d'effectuer toutes les actions possibles, ce sera à l'animateur d'ajuster son rotate order en fonction des actions qu'il souhaite réaliser. Il est donc important dans notre rig de prévoir cette éventualité.

*Tip : Le rotate order est un attribut qui est keyable mais qui est caché. Il est possible de l'afficher en allant dans le **menu de la « Channel Box » Edit > Channel Control**, par ce moyen il est aussi possible d'afficher et de débloquent des attributs que l'on aurait par mégarde caché ou bloqué.*

- Le coude doit d'abord pouvoir plier, puis monter ou descendre, et ensuite tourner sur son axe principal. L'axe Y plie le bras, l'axe Z le fait monter de bas en haut, et le X tourner sur son axe, YZX, soit un « rotate order » XZY.
- Le poignet doit pouvoir d'abord pivoter sur son axe principal X, puis se casser Y, tout en gardant dans tous les cas la possibilité de monter et descendre Z. Ce qui donne XYZ soit un « rotate order » ZYX.

2.5 Le FK / IK

Dans le cas d'un cycle de marche par exemple, où le bras se balance naturellement d'avant en arrière. Une chaîne de joints classique créée et orientée comme nous l'avons vu précédemment est animable en FK « Forward Kinematics ». Elle reproduit de manière parfaite le mouvement de balancier. Dans notre exemple du bras, la position de la main sera définie par les rotations du bras, puis de l'avant-bras puis de la main. Le mouvement s'effectue donc du haut (l'épaule) vers le bas (la main).

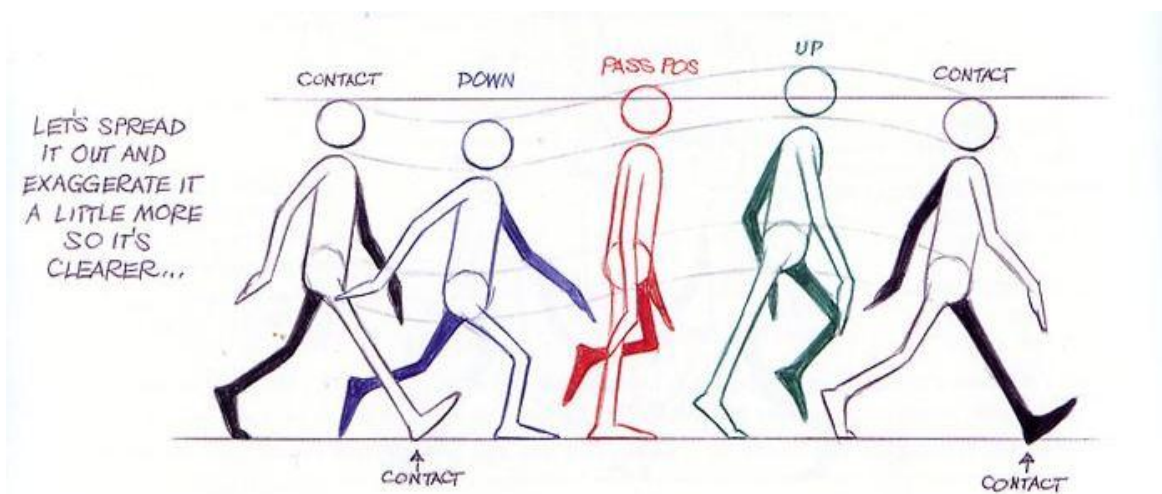


Fig.8

Mais ce type d'animation FK peut se montrer particulièrement complexe lorsque notre personnage se trouve en position d'appui sur le sol. Sur ce cycle de marche par exemple (Fig.8.) il serait possible, mais très difficile de faire l'animation de la jambe en FK. Si on devait, à chaque repositionnement du bassin, replacer le pied sur le sol par le biais de trois articulations différentes : la rotation de la hanche, du genou et du pied, le processus serait très laborieux.

C'est pourquoi un autre système est développé dans Maya : les IK « Inverse Kinematics ». Contrairement à une chaîne classique en FK, les IK utilisent la position du dernier élément de la chaîne de joints pour déterminer la position des joints entre la base et la fin de la chaîne. Dans Maya les IK sont créés à l'aide de l'outil « IK Handle Tool ». Si on poursuit avec notre cycle de marche, il faudrait donc d'abord définir une base (la hanche) puis une fin (la cheville) avec l'outil pour qu'il crée un « ik handle ». Celui-ci nous permettra de manipuler notre chaîne de joints IK et de l'animer du bas vers le haut. Ainsi le contact au sol est préservé si l'on bouge le bassin, et il suffira de positionner la cheville ou le bassin pour positionner toute la jambe. Ce système sera aussi très utile lorsque le personnage est en appui sur les bras, par exemple s'il fait des pompes ou s'il garde les mains posées sur une table.

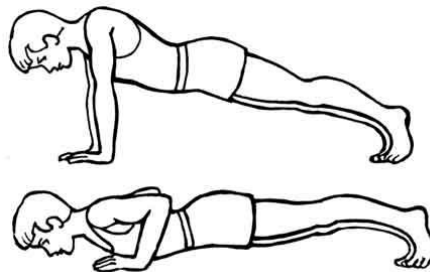


Fig.9.

Revenons-en à notre bras. Il est question d'intégrer ces deux types de contrôle, FK et IK, au rig de notre bras (à noter que pour la jambe la manière de faire est identique). Pour y arriver j'utilise un système avec trois chaînes de joints. Une chaîne FK, une chaîne IK, et une chaîne « result ». La chaîne « result » est celle qui emporte la géométrie par « skinning », sa position est influencée par les deux autres chaînes, elle peut prendre la position de la chaîne FK ou de la chaîne IK en fonction des besoins d'animation.

Pour passer de l'une à l'autre, il faut faire en sorte que les valeurs de rotation de la chaîne « result » soit le résultat d'une moyenne entre les valeurs de rotation des chaînes IK et FK. On appelle ça l'« Ik Blend » (Fig.10). Pour calculer cette valeur moyenne, j'utilise le node « blendColors » de Maya. Avant d'utiliser le blendColors, je dois dupliquer ma chaîne de joints « result » deux fois, et renomme la première « IK », et la seconde « FK ». Ensuite je créé trois nodes « blendColors », un pour chacune de mes articulations, et je connecte dans l'entrée « color1 » l'attribut « rotate » de ma chaîne IK et dans l'entrée « color2 » l'attribut « rotate » de ma chaîne FK. Je récupère la valeur de sortie « output » et la connecte au « rotate » de ma chaîne « result ». L'attribut « blender » de mes nodes « blendColors » contrôlera donc le passage de mon bras du FK vers l' IK.

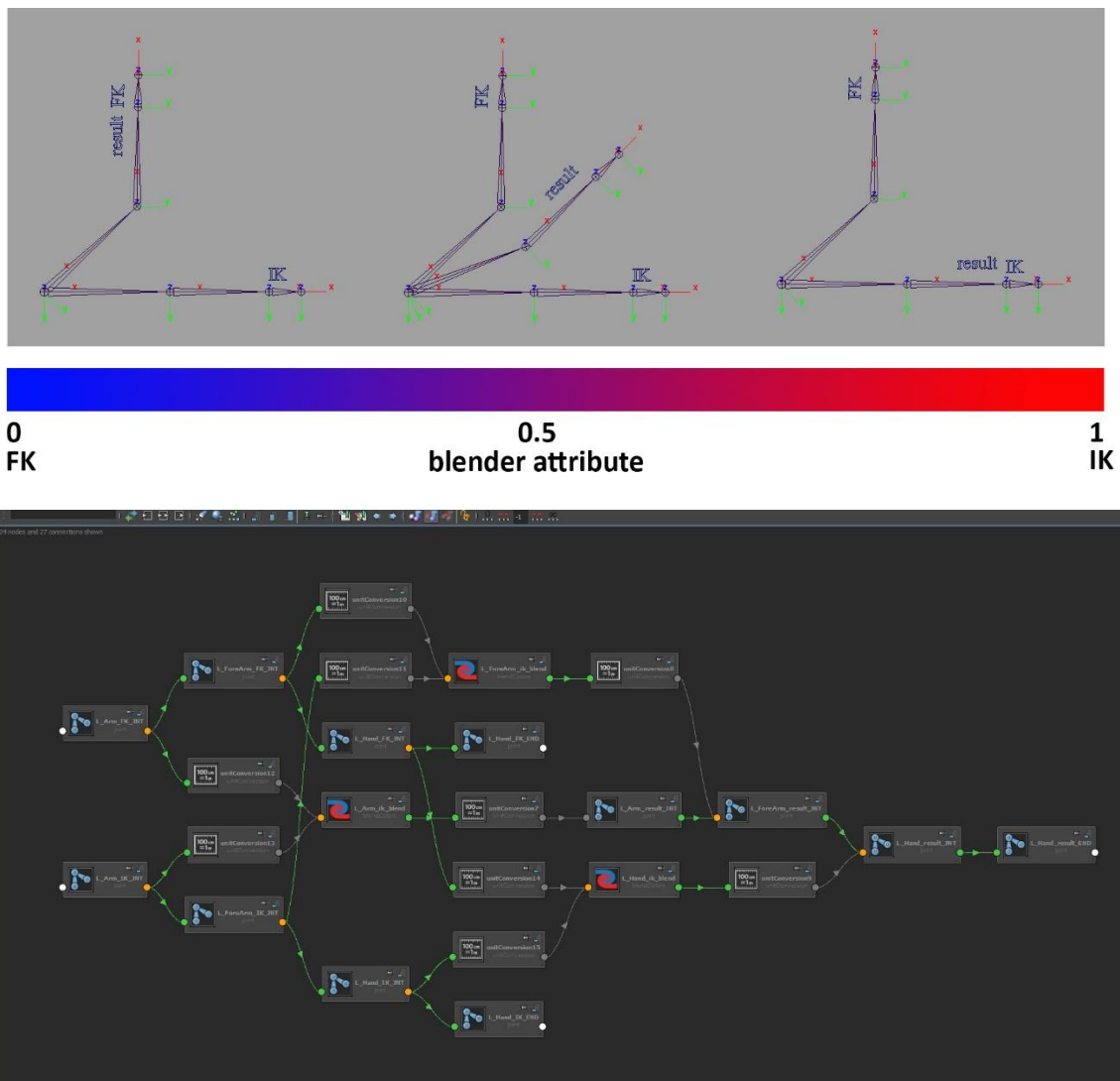


Fig.10.

Ensuite il faut appliquer les deux « Ik Handle » sur notre chaîne IK. Le premier entre le bras et la main en utilisant l’option « Ik RP solver ». Ce paramètre permet d’utiliser une contrainte « pole vector » pour orienter le coude. Le deuxième entre la main et le fin de la main en utilisant cette fois l’option « Ik SC solver » pour pouvoir orienter la main en mode IK. Créer un locator au niveau du coude puis appliquer la contrainte « pole vector » à l’« ik handle » du bras, de cette manière le locator dirigera l’orientation du coude.

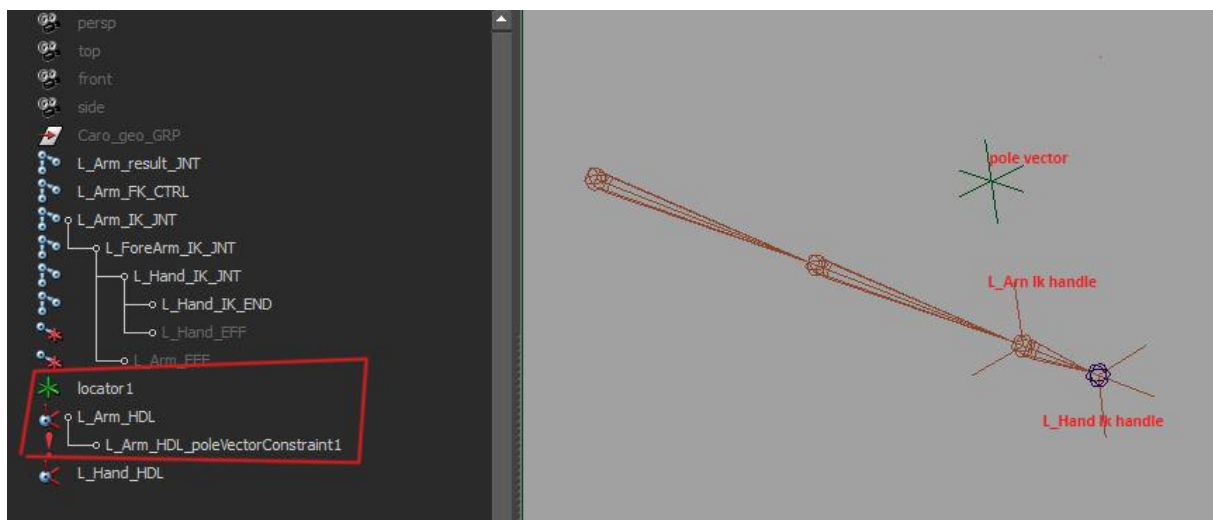



Fig.11.

2.6 La création de contrôleurs

Dans Maya les contrôleurs sont fait avec des « courbes ». Le « curve tool »  de Maya permet de créer ces formes vectorielles en 3D, qui une fois parentées ou contraintes aux joints vont nous servir à animer notre personnage. Ces contrôleurs permettent aux animateurs de manipuler notre rig dans l’espace et d’animer les différentes hiérarchies de joints qui contrôlent notre géométrie par skinning. Il serait possible d’animer directement les joints mais il n’est pas souhaitable que les animateurs puissent accéder directement aux valeurs des joints car en les modifiant ils pourraient du coup casser le rig. Il faut donc, pour que notre rig reste stable tout en long du travail d’animation, créer un contrôleur pour chaque articulation et chaque hiérarchie de joints. Les valeurs d’un contrôleur doivent toujours être égales à zéro, pour deux raisons : il faut qu’à tout moment il soit possible de remettre le personnage à sa pose initiale en remettant

toutes les valeurs de ses contrôleurs à zéro ; il est aussi plus simple pour les animateurs de comprendre leurs courbes d'animation si les valeurs démarrent à zéro (et comme vu précédemment si les contrôleurs ont des valeurs positives et négatives cohérentes).

Pour créer ces contrôleurs , il y a deux possibilités :

- je peux créer une curve et la contraindre directement à l'objet que je souhaite animer. Il s'agira de créer entre la curve et l'objet une contrainte de type parent, orientation ou de position. Ainsi la curve dirige mon objet. Si je contrains le contrôleur à mon objet et que mon objet ne se trouve pas au centre du monde, il faudra que je positionne le contrôleur à l'endroit où se trouve l'objet. Ainsi ses valeurs de position ne seront pas nulles, je pourrai alors faire un « Freeze Transform » pour mettre toute ses valeurs à 0. Mais fonctionner de cette manière pose un problème : il sera impossible de retrouver la position de l'objet par rapport au monde si on fait un « Freeze Transform ». C'est pourquoi, pour créer un contrôleur de ce type, il est conseillé de directement créer la curve au centre du monde et de la grouper. Ensuite il suffit de placer le groupe à la position de l'objet que l'on souhaite contrôler, de cette manière les valeurs du groupe correspondront à la position du contrôleur dans l'espace du monde et le contrôleur conservera des valeurs locales nulles.
- L'autre possibilité est de parenter le shapeNode de la curve directement sous le joint, pour que le joint devienne lui-même un contrôleur. Cette technique est possible seulement sur des contrôleurs qui dirigent des rotations, les valeurs de translation d'un joint ne pouvant être remise à zéro en utilisant la fonction « Freeze Transform ». Dans Maya il n'est possible de parenter les shapeNodes qu'en utilisant le langage MEL ou python.

Tip: Pour parenter une shape, utiliser la commande MEL : parent -r -s;

Ces deux méthodes sont utiles, et correspondent à un usage bien précis. Pour diriger ma chaîne de joint FK je préfère donc utiliser la seconde méthode qui est de parenté la shape de la curve sous le joint car en FK mes joints ne sont dirigés que par des rotations. En fonctionnant de cette manière, mes contrôleurs prendront la position exacte de mes joints, leur hiérarchie, ainsi que

leurs orientations. Si on avait utilisé l'autre méthode il aurait fallu recréer une hiérarchie entre les contrôleurs, identique à celle des joints mais aussi re-configurer leur « rotate order ».

Pour l'exemple de notre bras, je crée donc un contrôleur pour chaque joint de ma chaîne FK en parentant les « shapes » des « curves » sous les joints. Puis un contrôleur pour diriger l'« ik handle » de ma chaîne IK, un contrôleur pour son « pôle vector », en utilisant la méthode de la « curve et du groupe », et enfin un contrôleur « settings » qui portera l'attribut « FK/IK blend » qui nous permettra de passer du mode FK au mode IK. Il est possible d'ajouter des attribut aux objet en utilisant le menu de la « *Channel Box* » *Edit* > *Add Attribute* ou encore depuis la barre de menu générale *Modify*> *Add Attribute*.

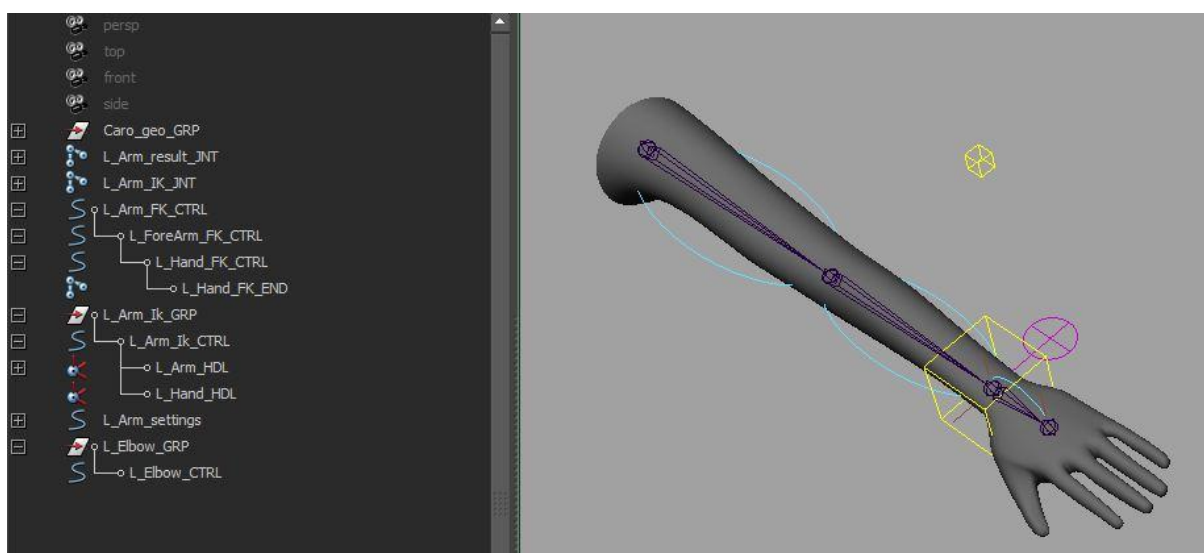


Fig.12

Maintenant que les contrôleurs FK et IK sont créés, il faut parenter les « ik handles » sous le contrôleur IK de la main pour qu'il emporte le bras et qu'il puisse orienter la main. On connecte ensuite l'attribut « Fk / Ik Blend » créé sur le contrôleur « setting » à l'attribut « blender » des nodes « blendColors » que nous avons créé pour passer d'un mode à l'autre.

Chacun de ces contrôleurs portera une couleur distincte qui permettra à l'animateur de comprendre quel type de contrôleur il est en train de manipuler. Pour attribuer des couleurs au curve il a aussi plusieurs méthodes : la première est de regrouper les différents contrôleurs dans des calques et d'attribuer une couleur à chaque calque ; la seconde méthode est de sélectionner

la « shape » de la curve et, dans l' « *Attribute Editor* » > *Object Display* > *Drawing Overrides*, cocher **Enable Overrides** et choisir la couleur. Je préfère toujours utiliser cette seconde méthode par souci de propreté car les calques passent dans les références de fichiers. En production si chaque graphiste décide de créer ses propres calques, quand on arrive en fin de chaîne, la scène peut alors contenir une multitude de calques qui ne servent plus à rien et il sera impossible de les supprimer, puisque nous sommes en référence. Nous verrons par la suite l'importance de travailler en référence.

Pour finaliser le bras, il ne reste plus qu'à gérer la visibilité de ses contrôleurs. En effet s'ils sont tous affichés en même temps cela prête à confusion, et l'animateur arrivera difficilement à savoir si le contrôleur se trouve en mode IK ou FK. Pour gérer l'affichage des contrôleurs, j'utilise la valeur de l'attribut « Fk / Ik Blend ». Je groupe mes contrôleurs FK et IK chacun dans un groupe distinct. Si l'attribut « Fk / Ik Blend » est à 0 je mets la visibilité de mon groupe IK à 0 ; s'il est à 1 j'affiche mon groupe IK et je masque mon groupe FK. Ainsi, les deux types de contrôleurs s'affichent séparément en fonction du mode choisi. Il y a comme d'habitude plusieurs façon de procéder mais j'ai choisi pour cet exemple d'utiliser les « set driven key ».

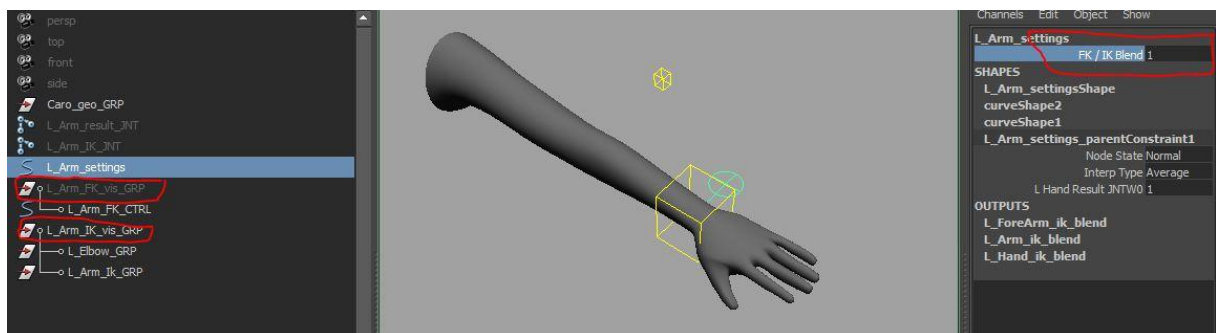


Fig.13

2.7 La compression et l'étirement ou « squash and stretch »

A ce niveau du rig du bras, il est important de revenir sur quelques principes d'animation, qui vont influencer notre rig, comme le « squash and stretch ». Pourquoi faut-il que le personnage puisse s'étirer ou se compresser ? Dans quel cas est-il amené à se déformer ?

- Le volume

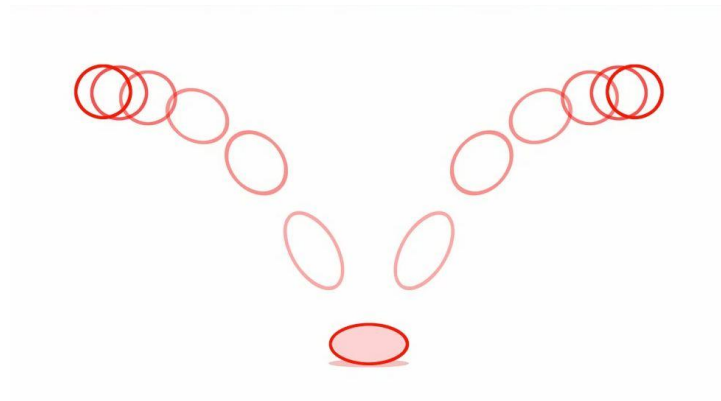


Fig.14

Pour animer le rebond d'une balle et lui donner du dynamisme, on peut voir sur ce dessin une variation du volume de la balle : elle s'étire au moment où elle descend, elle se comprime au moment de l'impact, et elle s'étire à nouveau au moment de sa remontée.

- Le retard et le dépassement

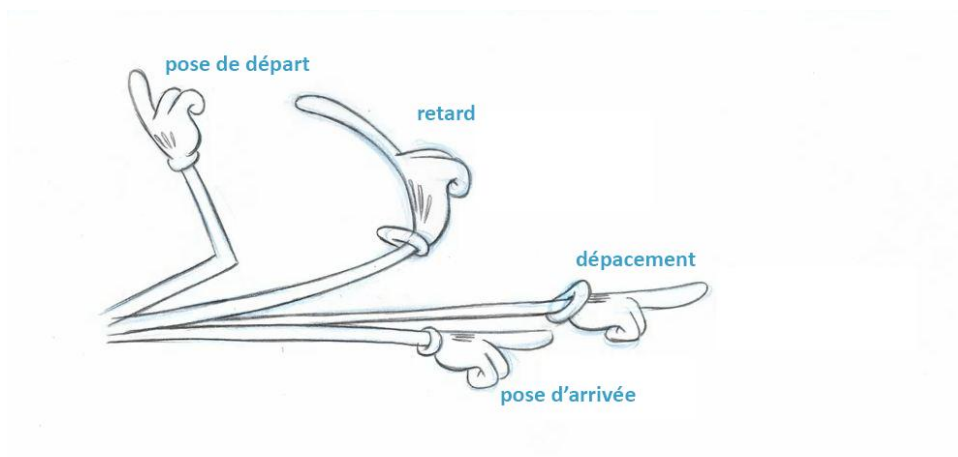


Fig.15

Dans cette animation de bras, afin d'amplifier et de donner du dynamisme au mouvement, deux poses intermédiaires sont créées : une pose de retard où l'on peut voir le bras et la main se déformer de manière complètement irréaliste, et une pose de dépassement où le bras s'étire.

Il faut donc bien comprendre que dans le dessin animé ou les films d'animation 3D, les animateurs ne cherchent pas à conserver des proportions ou des mouvements anatomiquement possibles dans leurs poses. L'exagération du mouvement qu'ils produisent aura souvent un effet immédiat sur le caractère comique mais aussi sur la compréhension de l'action par le spectateur. De plus, nous sommes confrontés au fait qu'un modèle 3D est très rigide au départ. Pour lui donner vie, il va donc falloir lui donner la capacité de souplesse. Richard Williams, réalisateur de l'animation du film « Qui veut la peau de Roger Rabbit » expose très bien l'importance de la souplesse dans son livre, *Techniques d'animation, pour le dessin animé, l'animation 3D, et le jeu vidéo*⁹.

Comment faire pour donner cette souplesse ? On poursuit avec l'exemple du bras en lui donnant la possibilité de s'étirer et de se compresser pour favoriser le travail d'animation. Nous avons trois chaînes de joints, il faudra permettre à ces trois chaînes de « stretch ».

- Commençons par la chaîne FK : j'ajoute un attribut « length » avec un minimum à 0 et une valeur par défaut à 1 aux deux contrôleurs FK, bras et avant-bras (on laisse de côté la main cette fois-ci). On fait ensuite un « set driven key » entre l'attribut « length » du bras et le translate X de l'avant-bras. Quand la longueur du bras est égale à 1 et l'attribut translate X est égal à sa valeur initiale on crée la première clé. Puis on met l'attribut translate X à 0 et l'attribut « length » à 0, et on crée la deuxième clé.

⁹ Richard Williams, *Techniques d'animation : Pour le dessin animé, l'animation 3D et le jeu vidéo*, Eyrolles, 2011.

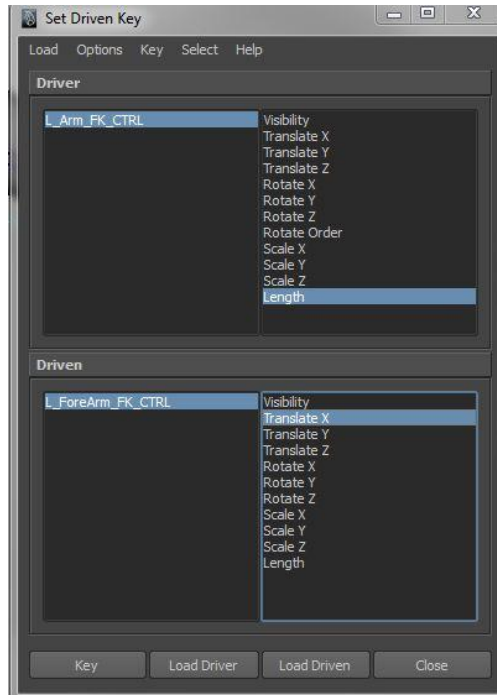


Fig.16

Ensuite je vérifie bien dans mon « graph editor » que mes courbes sont linéaires et que mon « post infinity » est aussi en linéaire. De cette manière, le bras pourra s'étirer à l'infini.

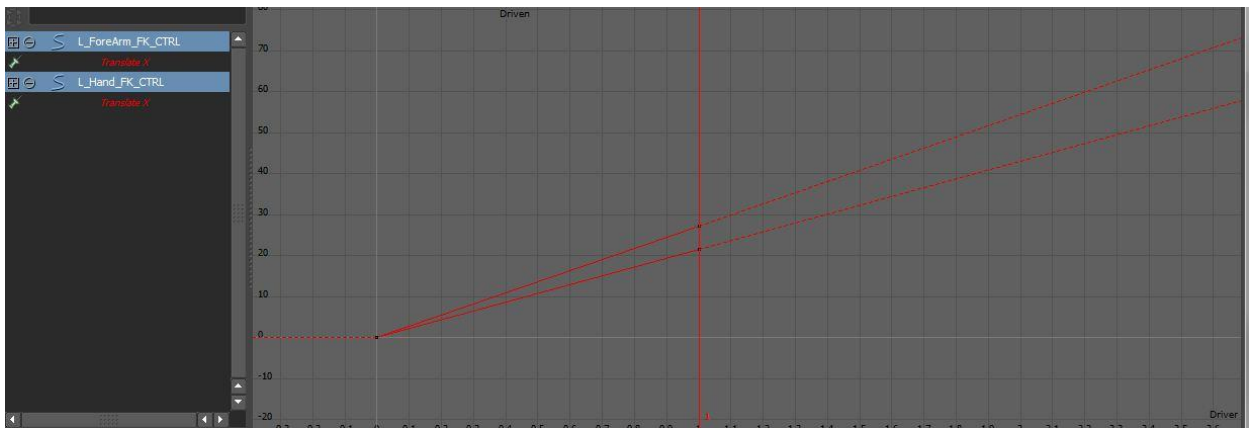


Fig.17.

Comme nous l'avons fait pour l'IK blend et les rotations, il faut maintenant ajouter la possibilité à la chaîne « result » d'être aussi conduite par la chaîne IK et FK en translation. Pour cela je répète la même opération que pour les rotations, je crée deux nouveaux nodes « blendColors » et je connecte les translations de mes chaîne FK et IK à ma chaîne « result » (Fig.18.).

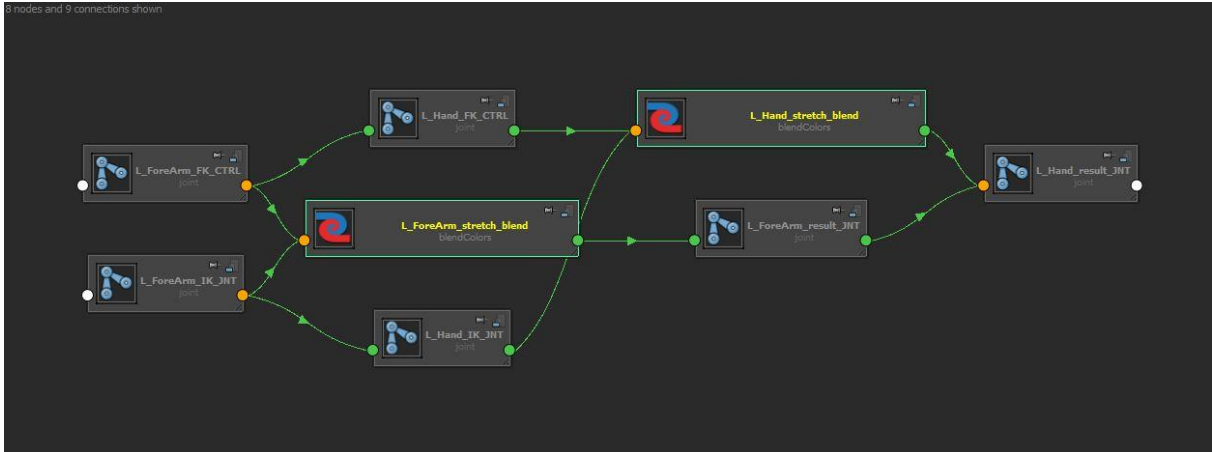


Fig.18.

Pour la chaîne IK c'est un peu différent, car c'est le déplacement du contrôleur IK qui déterminera le « stretch » du bras. Je ne peux pas utiliser les valeurs de translate de mon contrôleur pour driver mon stretch car en fonction du positionnement du bras dans l'espace ce ne sera pas toujours le même axe qui sera responsable du « stretch ». Il faut alors utiliser le « distance tool » qui va créer un « distance node » qui nous renverra en permanence la longueur du bras, quelle que soit sa position dans l'espace. Je crée les deux locators qui serviront au « distance node », un au niveau du bras et l'autre au niveau du contrôleur IK ; je parente ce deuxième locator sous ce contrôleur, ainsi ma distance est évaluée en fonction du déplacement de mon contrôleur.

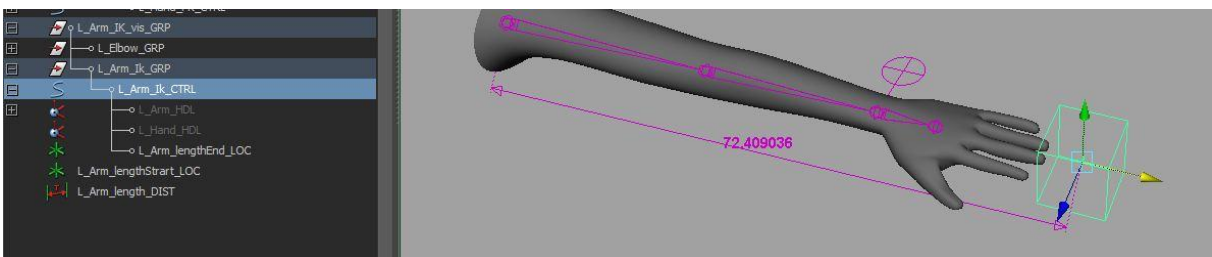


Fig.19.

On a maintenant une distance cohérente, que l'on peut utiliser pour réaliser le stretch IK. Nous allons devoir faire une autre série de « set driven keys », le « driver » sera donc l'attribut « distance » contenu dans la shape de mon « distance node ». La longueur totale de mon bras sera égale au translate X du joint IK de l'avant-bras plus le translate X du joint IK de la main.

Je ne veux évidemment pas que le bras se comprime (lorsque la distance est inférieure à la longueur totale de mon bras) mais conserver le fait que le bras puisse se plier. Contrairement à la stretch FK, je ne calculerai donc pas le stretch IK à partir de 0, sinon mon bras s'écraserait. Considérant que la zone de 0 à la longueur totale du bras correspond au moment où le bras plie, je crée mes clés depuis la longueur totale jusqu'à 2 fois cette longueur. Je n'ai plus qu'à mettre mes courbes d'animation du joint IK de l'avant-bras et du joint IK de la main en linéaire puis à régler le « post infinity » en linéaire lui aussi pour que le stretch s'applique indéfiniment.

J'ai écrit un script MEL qui permet de réaliser rapidement les étapes que nous avons décrites : il va s'occuper de créer les « driven keys » et de calculer les valeurs translate X de nos joints IK.

```
string $driver = "mondistancenodeShape.distance";
float $longueurHautBras = `getAttr avant-bras_IK_JNT.translateX`;
float $longueurAvantBras = `getAttr main_IK_JNT.translateX`;
float $longueurTotale = $longueurHautBras + $longueurAvantBras;

setDrivenKeyframe -currentDriver $driver -driverValue $longueurTotale -
attribute "translateX" -value $longueurHautBras avant-bras_IK_JNT;


setDrivenKeyframe -currentDriver $driver -driverValue ($longueurTotale
*2) -attribute "translateX" -value ($longueurHautBras*2)
avant-bras_IK_JNT;

setDrivenKeyframe -currentDriver $driver -driverValue $longueurTotale-
attribute "translateX" -value $longueurAvantBras main_IK_JNT;

setDrivenKeyframe -currentDriver $driver -driverValue ($longueurTotale
*2) -attribute "translateX" -value ($longueurAvantBras*2) main_IK_JNT;
```

2.8 Le twist

Le « twist » permet une répartition correcte de la rotation, par exemple du poignet sur l'avant-bras, et permet une meilleure déformation de la peau et évite l'effet « papier de bonbon ». Comme pour le FK/IK, le twist est présent sur plusieurs parties d'un rig : par exemple pour un personnage humanoïde, il sera nécessairement sur la colonne et les bras. Nous allons donc poursuivre notre exercice en réalisant un twist sur l'avant-bras, mais la méthode que l'on va suivre serait exactement la même sur la colonne vertébrale (avec quelques spécificités tout de même).

Tout d'abord pour effectuer le twist j'ai besoin d'une nouvelle chaîne de joints dans mon avant-bras. Je duplique donc les joints qui composent l'avant-bras, puis avec le « Insert Joint Tool »  je vais insérer quatre autres joints entre le début et la fin de cette chaîne que je pense évidemment à renommer correctement (on indique quatre joints car ça répartie correctement le twist).

Tip : pour déplacer un joint sans déplacer ses enfants, utiliser la touche « inser » du clavier.

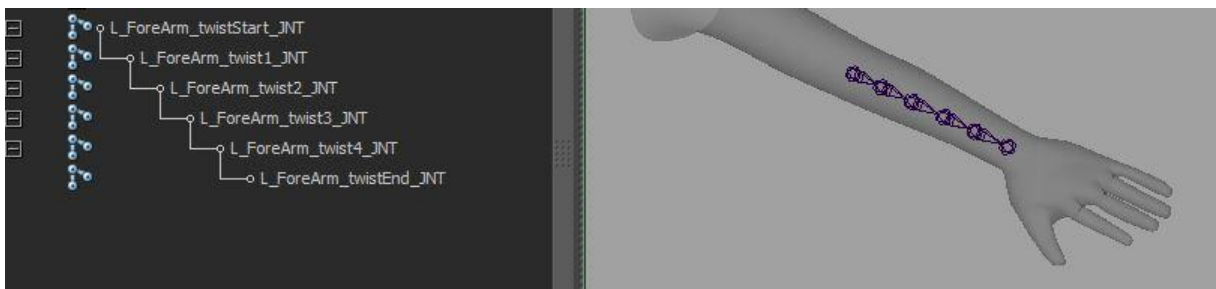



Fig.20.

Ensuite, je vais créer un « IK Spline Handle », à la différence de l'« IK Handle », il va utiliser une « curve » pour diriger les joints. C'est en déplaçant les points de cette curve que je pourrai obtenir le contrôle que je souhaite avoir sur la chaîne de twist. Je crée donc une curve en mode linéaire entre les joints du début et de fin de ma chaîne. Ensuite, pour créer l'« IK Spline Handle » avec l'« IK Spline Handle Tool » , je sélectionne d'abord les joints puis la curve. Par défaut l'« IK Spline Handle Tool » crée une curve. Je ne souhaite pas qu'il le fasse car je

veux qu'il utilise celle que j'ai créé au préalable : je décoche donc dans les options de l'outil « Auto create curve ».

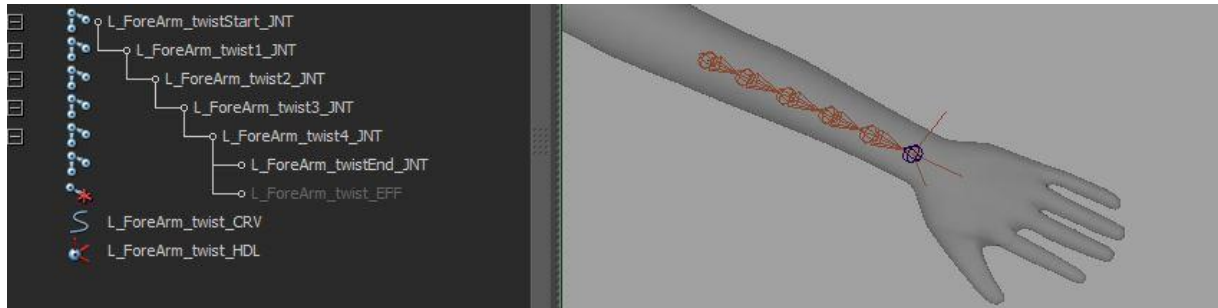


Fig.21.

Pour contrôler la curve et donc mes joints de twist, je vais créer deux nouveaux joints, des « bind joints ». Je les nomme de cette manière car ma curve va être skinner à ces joints. En déplaçant ces « bind joints » je déplacerai la curve. Je vais donc dupliquer le premier et le dernier joints de ma chaine et les renommer, puis skinner ma curve à ces joints en utilisant un « Max Influence » de 2.

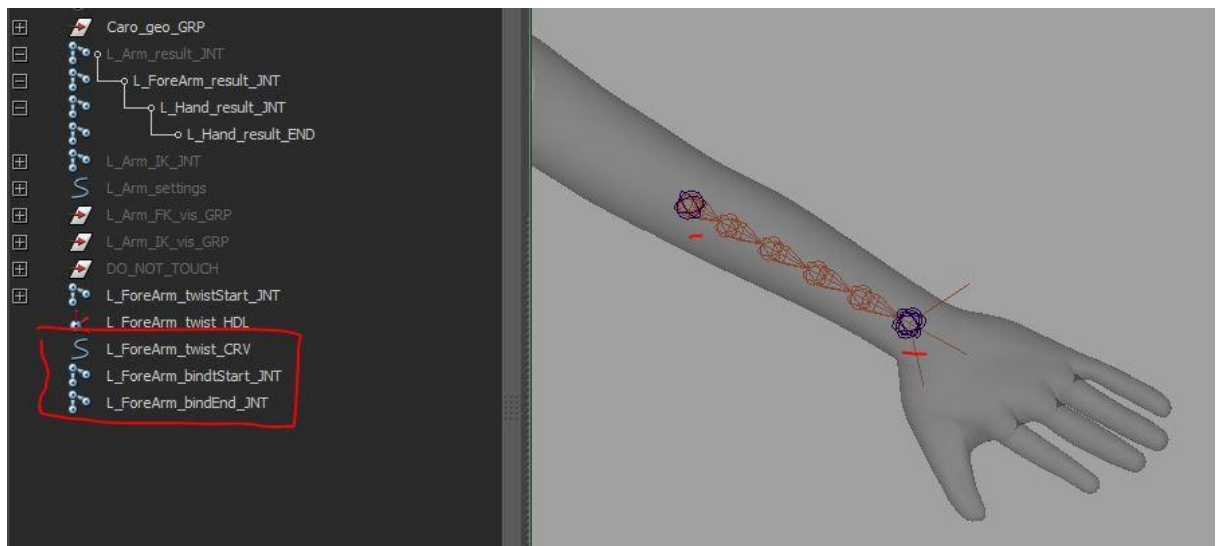


Fig.22.

Mes deux « bind joints » contrôlent maintenant ma chaine twist. Il me reste encore à activer le twist de mon « ik handle », et dire à Maya que mes « bind joints » contrôlent ce twist (Fig23.). Puis j'applique une contrainte « parent » entre mes « bind joints » et les joints de la chiane

« result » correspondant. En conséquence ma chaîne de twist va maintenant suivre le déplacement du bras.

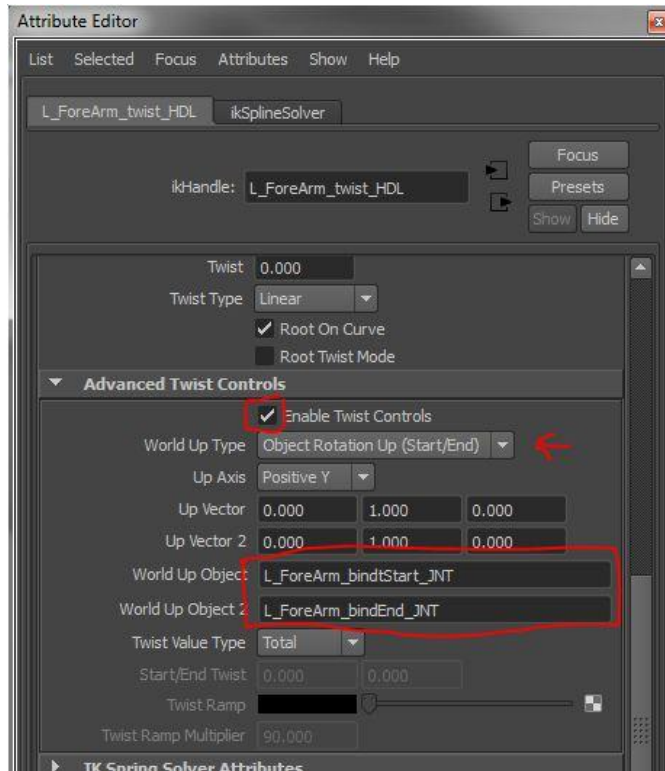


Fig.23.

Il reste une dernière chose à faire. Vu que nous avons donné la possibilité de « stretch » au bras, et qu'il est lié à notre chaîne de twist, il va falloir donner cette même faculté à notre twist. Pour cela, on utilise la courbe de l'« IK Spline Handle » pour connaître la longueur de notre chaîne, et l'attribut scale X pour en augmenter ou en diminuer sa longueur.

Pour connaître la longueur d'une courbe dans Maya on utilise le nœud « Curve Info ». L'attribut « Arc length » de ce nœud nous renverra la dimension de la courbe. Je connecte donc l'attribut « world space » à l'entrée « input curve » du nœud « Curve Info ». Pour que la longueur de la courbe contrôle le scale X des joints il faut d'abord que je normalise la longueur c'est-à-dire qu'elle soit comprise entre 0 et 1. Pour y arriver, je crée un nœud « Multiply Divide » et je divise la valeur de la longueur contenue dans l'attribut « Arc Length » par sa propre valeur. Il reste donc à connecter la sortie du nœud « Multiply Divide » au scale X de mes joints de twist pour que ma chaîne stretch (Fig.24).

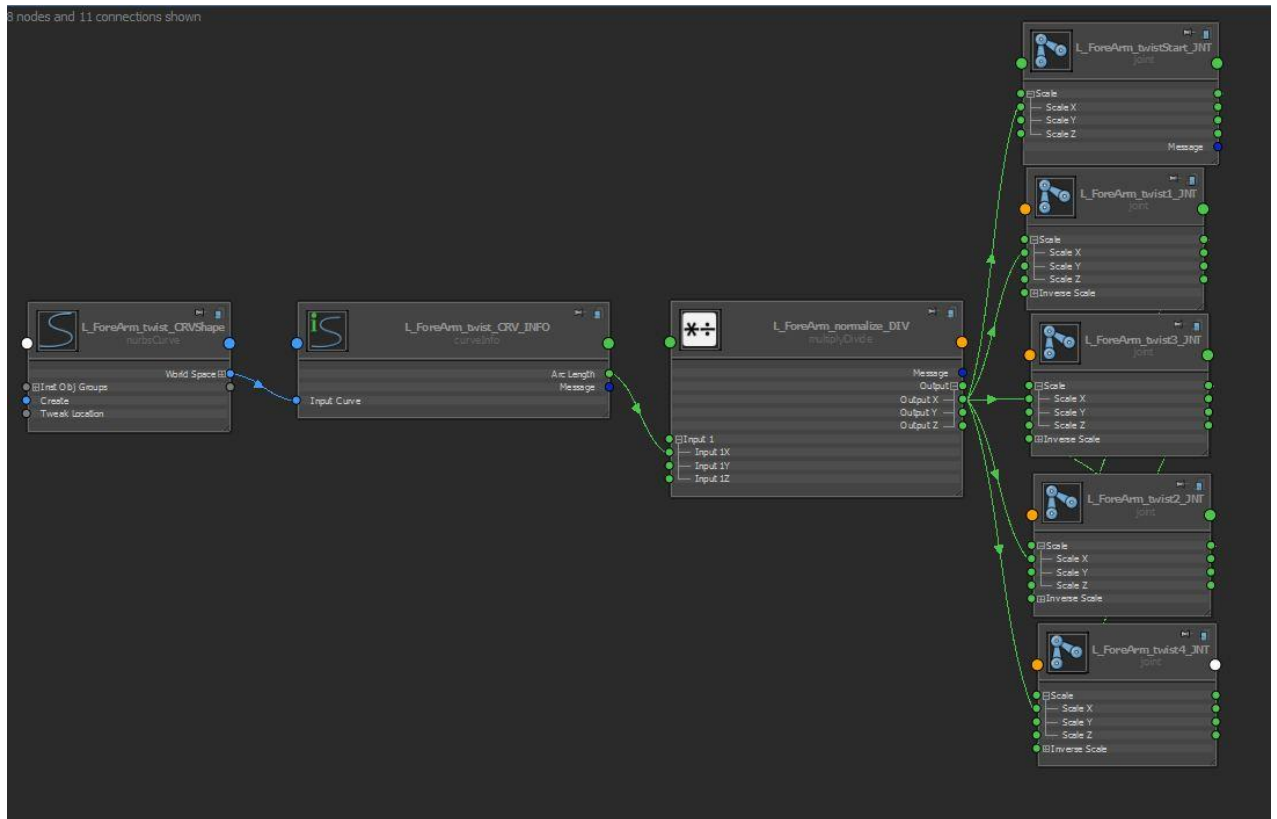


Fig.24.

2.9 La main

Pour finaliser entièrement notre exemple du bras il reste à voir une dernière partie, la main. Cette partie est intéressante car elle va nous permettre de comprendre comment donner à l'aide d'attributs complémentaires un maximum de contrôle à l'animateur tout en conservant les principes d'animations FK que nous avons vu précédemment. La principale difficulté pour mettre en pose la main est la dimension des doigts ainsi que leur nombre d'articulations. Il faut donc donner la possibilité aux animateurs de les contrôler plus facilement. Pour cela, je vais faire une chaîne qui est contrôlée d'une part en FK mais aussi par un système d'attributs personnalisés qui permettront de contrôler plusieurs doigts à la fois en une seule valeur.

Je commence par créer un doigt, avec ses contrôleurs FK, puis je vais insérer dans la hiérarchie d'autres joints « orient ». Ces joints « orient » me serviront alors d'offset, et porteront les animations commandées par les attributs personnalisés que je vais créer sur un autre contrôleur (Fig.25.).

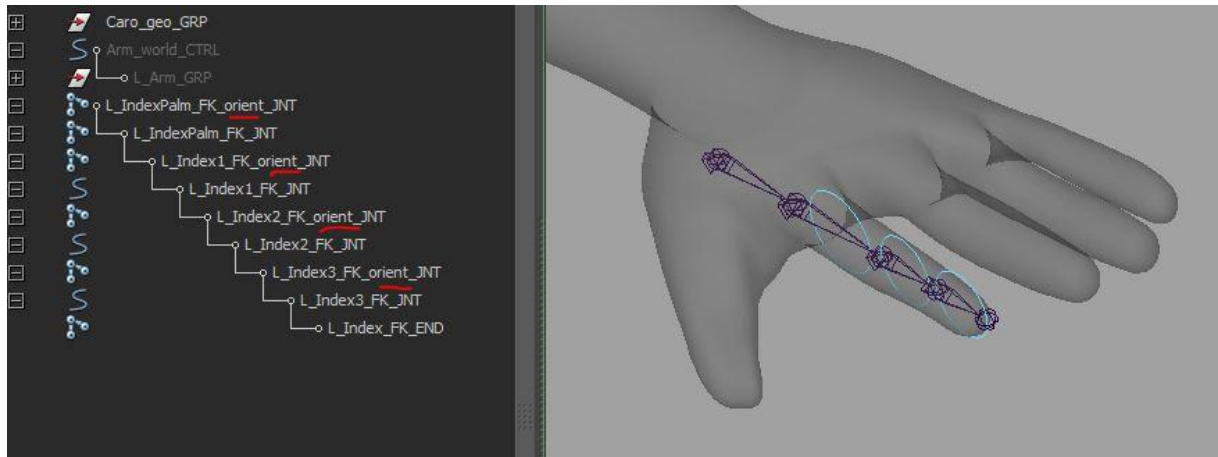


Fig.25.

Par exemple je vais créer deux attributs qui prendront comme valeur un minimum de -10 et un maximum de 10 (évidemment il est possible d'en créer beaucoup d'autres selon les besoins des animateurs) :

- Curl qui va permettre au doigt de se plier
- Scratch qui va permettre au doigt de gratter

Avec mes deux nouveaux attributs je vais contrôler la rotation de mes joints « orient » en utilisant des « set driven key ». Il faut faire la même chose pour tous les doigts de la main en pensant à bien orienter les joints.

Je vais aussi créer un autre contrôleur général qui contrôlera la main entière sur lequel je vais ajouter ces attributs:

- Fist, le poing
- Spread, qui permet au doigt de s'écarter
- Relax, qui mettra la main dans une pose relâcher

Ces attributs dirigeront les joints « orient » qui contrôlent la main.

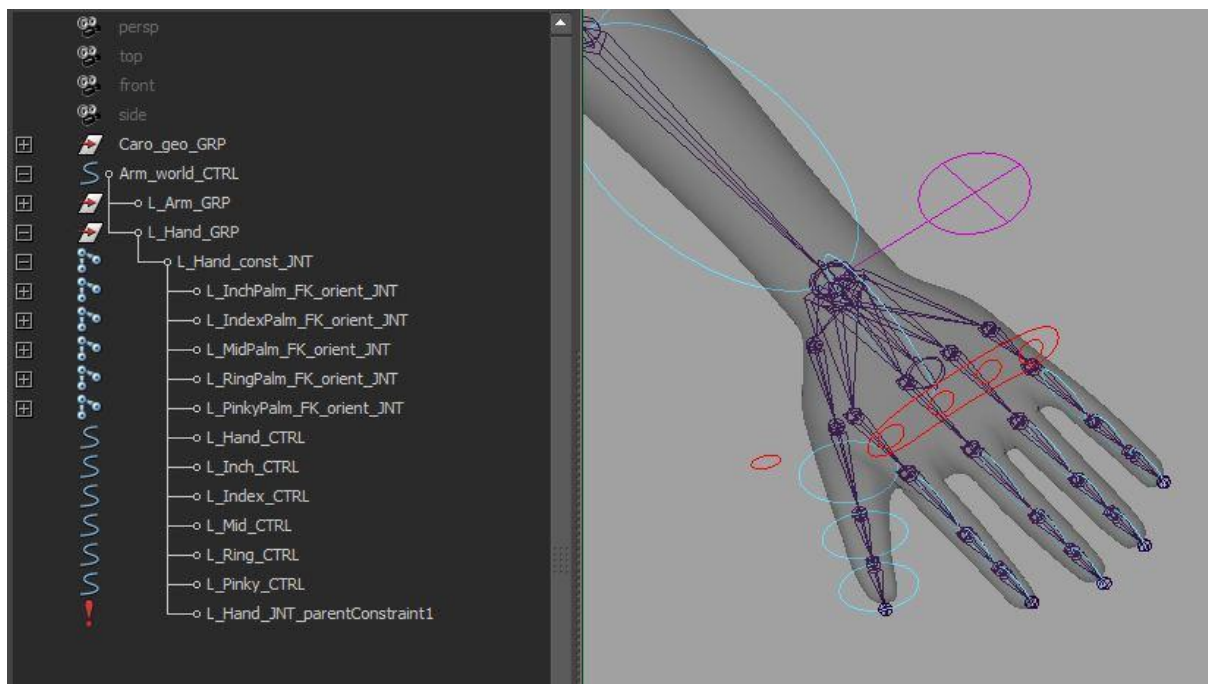


Fig.26.

Une fois les contrôles de la main terminés, je crée un joint auquel je parente toute les hiérarchies de mes doigts, puis je contrains « parent » ce joint au joint de la main de la chaîne « result » de mon bras. De cette manière la main et le bras restent des parties distinctes dans mon rig. Chacune des parties de mon rig est dans son propre groupe, cette manière de concevoir le rig. Il s'agit en partie d'une approche modulaire : chaque partie est isolée et facile à debugger, tout en étant réutilisable sur un autre rig. Il suffira en effet d'exporter chaque groupe dans une scène différente et de réimporter le module que l'on a besoin d'utiliser pour un autre personnage.

2.10 Finaliser un rig

Dans ce court chapitre je reviens sur un ensemble de règles qui si elles sont respectées rendront la vie des animateurs beaucoup plus sereine. Nous avons déjà abordé certaines de ces règles mais il est important de les lister, de manière à vérifier qu'on les respecte toutes avant de livrer son rig aux animateur .

- Les contrôleurs doivent être facilement préhensibles et utilisables par les animateurs
- Les contrôleurs doivent avoir des valeurs à zéro

- Les axes de rotation doivent être cohérents par rapport au mouvement que l'on souhaite exécuter (les joints bien orientés)
- Les « rotate orders » doivent être réglés de manière à permettre le bon mouvement de nos articulations
- Penser à locker et/ou cacher les attributs ou nodes que l'animateur ne doit pas modifier
- Le rig doit être convenablement organisé et nommé de manière à être facilement débuggé et à être réutilisable dans d'autre rig

Grâce à cet exemple du bras, nous avons pu aborder un certain nombre de techniques applicables à l'ensemble des rig que nous pourrions être amené à réaliser, puisque ces techniques peuvent s'appliquer au reste du corps. Toutefois, il n'y a pas une unique manière de faire un rig. Aussi, j'espère que chacun pourra s'approprier ses techniques de base, les tester, et développer sa propre façon de faire.

Pour poursuivre notre étude et aborder d'autres techniques plus spécifiques, je vais m'appuyer sur quelques-uns des projets que j'ai pu réaliser cette année.

III/ Etude de cas

Le premier cas d'étude que je veux vous présenter est *Reebot*. Il s'agit d'un film promotionnel d'1 min 30 réalisé par Frédéric Mayer qui met en scène un petit robot en 3D intégré dans l'univers de la société filmée en prise de vues réelles. J'ai notamment réalisé le rig de ce petit robot (j'ai aussi fait le rendu mais ça c'est une autre histoire). Je vais à la fois essayer de revenir sur les aspect du rig qui ont bien fonctionné mais aussi sur les difficultés que j'ai pu rencontrer. Comme il s'agissait d'un vrai premier projet en entreprise, réalisé au contact de professionnels, cela me permettra aussi de revenir sur des aspects liés à la mise en place du pipeline pour un projet de film d'animation. Ensuite je parlerai d'Elaïa, projet de film en full 3D sur lequel nous avons travaillé avec Steve Beaucamp. La partie la plus complexe du rig de cette petite fille est certainement son visage, et comme ce sont des techniques que nous n'avons pas encore abordé je me concentrerai sur l'analyse du rig facial en vous présentant les méthodes que j'ai privilégiées.

3.1 Reebot

Comme on le fait avant de commencer n'importe quel rig, on va commencer par une courte analyse du personnage de *Reebot*. C'est en l'observant, et en cernant le personnage, sa modélisation mais aussi son caractère qu'on pourra imaginer sa manière de bouger, de se déplacer et se déformer. Cela nous permettra ensuite de faire des choix sur les techniques à envisager pour réaliser son rig. On pourrait dire que les caractéristiques physiques de *Reebot* sont plutôt mécaniques, à voir sa modélisation : c'est un petit robot construit avec des objets de récupération, des pièces électroniques, des morceaux de clavier d'ordinateur, différentes pièces comme des vis, des pistons, des morceaux de circuit imprimé etc...



Fig.27.

Il a deux roues pour se déplacer, il va donc rouler, ces deux roues sont fixées sur des pièces de métal triangulaires qui elles-mêmes ont la possibilité de tourner indépendamment des roues.

Ces pièces triangulaires sont connectées au reste du corps par des amortisseurs en forme de soufflets à l'avant et d'autres ressemblant à des antennes télescopiques à l'arrière. Ces amortisseurs devront pouvoir s'étirer et se comprimer afin de donner plus de liberté aux animateurs. Ensuite vient le corps, étant fait d'une coque métallique qui ne se déforme pas, il n'y a pas vraiment de difficultés, il suffira d'être sûr de bien placer son centre de gravité. Il y a cependant quelques éléments annexes à rigger, une hélice et une courroie sur le côté gauche. Il y a ensuite ses bras et ses mains, qui sont construits comme ceux des humains. Ceux-ci agiront donc comme les bras d'un humain, il faudra un contrôle FK et IK. Mais ils ont quand même une particularité : ils coulissent le long du corps. Les mains elles auront deux types de contrôle, le FK mais aussi des attributs supplémentaires pour plier les doigts. Ensuite il y a le cou, fait de trois pistons il forme une partie mécanique assez complexe, il faudra pouvoir contrôler les rotations à la base ainsi que des rotations et des translations pour le haut du cou qui permettront l'étirement des pistons. La tête elle est assez simple, il faudra bien placer son pivot, elle comporte une hélice, une petite barre métallique qui est située à l'arrière de la tête et des yeux en diaphragme. Ces yeux sont la seule partie du visage de Reebot à représenter une expression faciale et à pouvoir s'animer, ils vont donc exprimer toutes les émotions.

Je peux déjà tirer quelques conclusions. Une particularité de ce personnage est qu'il est mécanique, il n'y a donc pas de déformation de la peau, la majeure partie des pièces qui le composent sont rigides et ne se déforment pas, il n'y a donc pas de skinning complexe à effectuer. Par contre il risque d'être très lourd pour l'animation car il est composé de 2298 pièces et de presque 500 000 polygones. Il faut envisager la mise en place d'un système de proxy, pour que les animateurs puissent animer en temps réel dans Maya, ce qui implique de créer une géométrie beaucoup plus légère, qui ne sera pas la géométrie finale lors du rendu mais qui permettra aux animateurs de visualiser les mouvements et actions. Il faut donc prévoir qu'ils puissent passer de la géométrie haute résolution à une géométrie base résolution facilement.

Si l'on récapitule, j'ai défini cinq parties :

- les roues et amortisseurs (équivalents de ses jambes)
- le corps, hélice, petite courroie
- les bras (coulissants)
- le cou (pistons)

- la tête

3.1.1 Rig d'un personnage mécanique

Je ne vais pas couvrir toutes les parties de ce rig, cette démonstration pourrait très vite devenir ennuyeuse et répétitive puisque nous avons déjà abordé les techniques principales pour effectuer un rig . Je vais plutôt revenir sur les parties qui me semblent être les plus intéressantes et auxquelles j'ai porté le plus d'attention pendant la production, et ainsi partager mon expérience.

Les roues et amortisseurs

Je reviens d'abord sur cette partie car c'est sans doute celle qui m'a posée le plus de problèmes. Pour obtenir le double axe de rotation au niveau des roues, celui de la roue et celui du triangle de direction, j'ai commencé par créer deux joints. Pour positionner le premier joint correctement sur l'axe de la roue j'ai sélectionné un « loop » de vertex et créé un « cluster » ce qui me donne le centre de la roue, puis j'ai « snapper » le joint sur le « cluster ». J'ai ensuite créé les deux contrôleurs de ces joints en parentant une shape sous chaque joints. Après, j'ai créé un contrôleur principal pour la roue, qui permet le positionnement de la roue au niveau du sol, il sert aussi de pivot pour les différentes inclinaisons possibles de la roue. J'ai parenté les deux contrôleurs de rotation de la roue sous ce contrôleur principal. Puis j'ai « smooth bind » la géométrie de ma roue avec une « Max Influence » à 1 pour être sûr qu'un seul joint affectera ma roue. Puis j'ai fait de même pour le triangle de direction. J'ai choisi de faire le « bind » au fur et à mesure pour bien voir l'évolution du rig et contrôler mes axes de rotation. Ensuite j'ai « locké » les axes rotation qui ne me serviront pas sur les contrôleurs, je ne préfère jamais les cacher avant d'avoir entièrement terminé le rig. Puis j'ai colorisé mes contrôleurs.

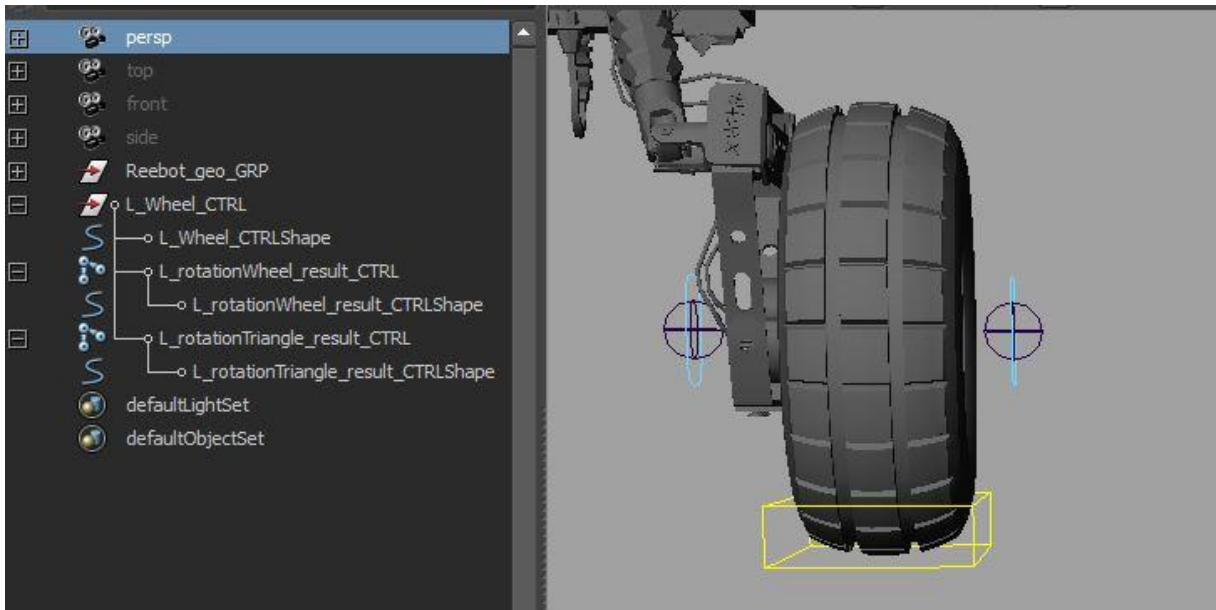


Fig.28.

J'ai ensuite simulé la pression du pneu grâce à une lattice, et un cluster. Lorsque le contrôleur de la roue descend à -1 en Y, le cluster à l'inverse monte de 1 en Y. J'ai effectué cette opération en utilisant un « set driven key », il faut bien vérifier que les clés d'animations sont en linéaire. J'ai aussi donné la possibilité à l'animateur d'activer ou non cette fonction car dans certain cas il pourrait être amené à descendre la roue plus bas que 0 sans vouloir que le pneu s'écrase. J'ai donc créé un attribut booléen sur le contrôleur principal de la roue « pressure », ainsi qu'un « condition node » qui vérifie si cette attribut est on ou off.

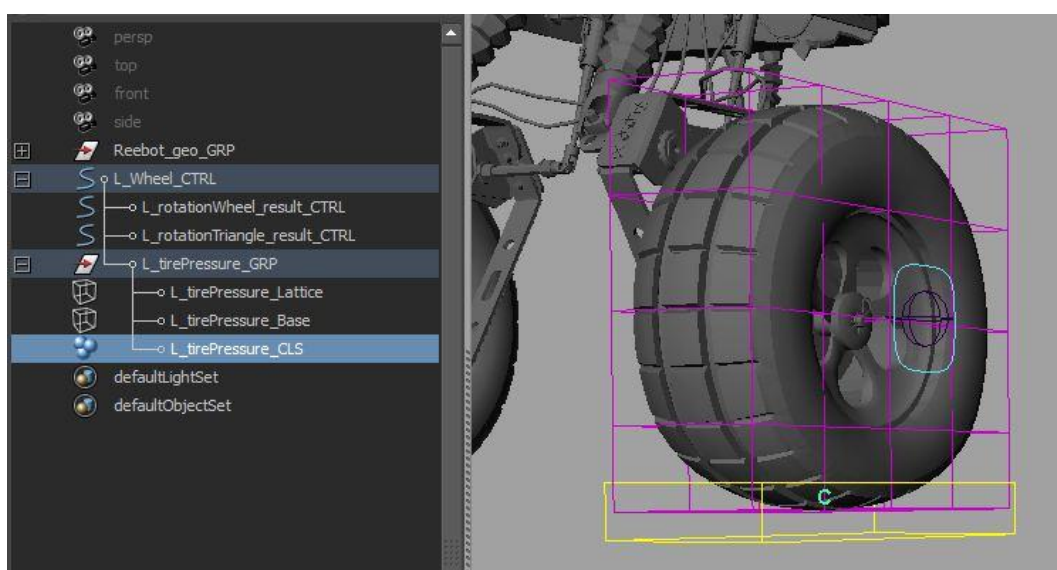


Fig.29.

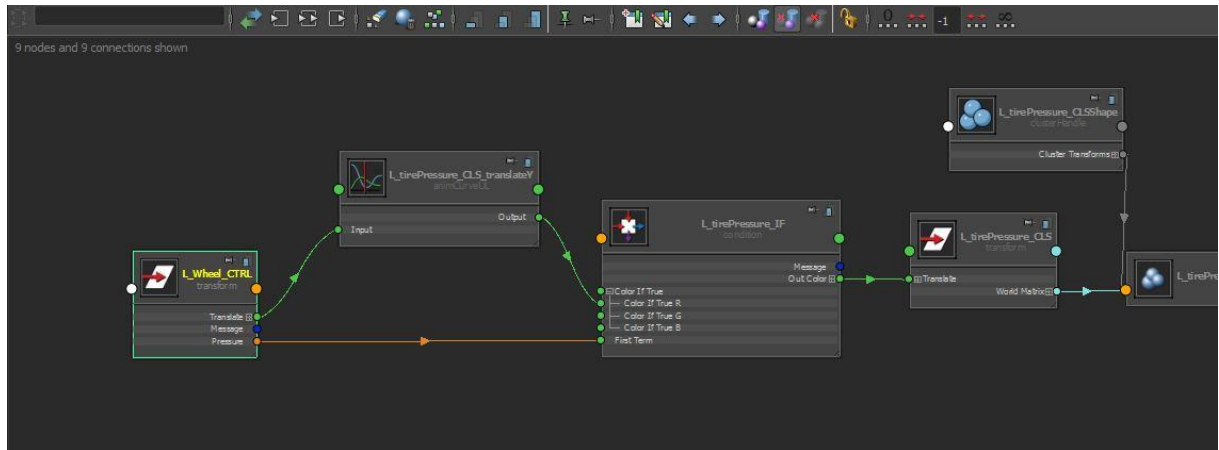


Fig.30.

On continue avec les amortisseurs. Sur ce projet j'ai décidé d'utiliser la « ribbon spine » car le système classique de « squash and stretch » qui utilise l'« IK spine handle » m'a posé problème. En effet il empêche de contrôler avec précision le bout de la chaîne, vu que j'ai travaillé sur des pièces mécaniques les légers décalages en bout de chaîne disloquaient complètement les pièces, ce qui ne convenait pas du tout.

La « ribbon spine » présenté dans plusieurs tutoriaux¹⁰ m'a permis de corriger ce problème. Elle n'a pas été trop complexe à mettre en place. Comme j'ai déjà insisté sur le fait de penser son rig de manière modulaire il m'a suffi de la créer une fois puis de l'importer dans ma scène autant de fois que nécessaire. Elle m'a permis de faire les amortisseur mais aussi le rig de tous les pistons du cou de Reebot. Pour finaliser la partie basse de Reebot, j'ai donc contraint les extrémités des « ribbon spine » aux contrôleurs du triangle de direction (Fig.31.).

¹⁰ Fahrenheit – rigging for feature animation, „Monkey Body Setup
 Digital Tutors - Advanced Character Rigging in Maya 2013
 Digital Tutors - Rigging Quadrupeds in Maya

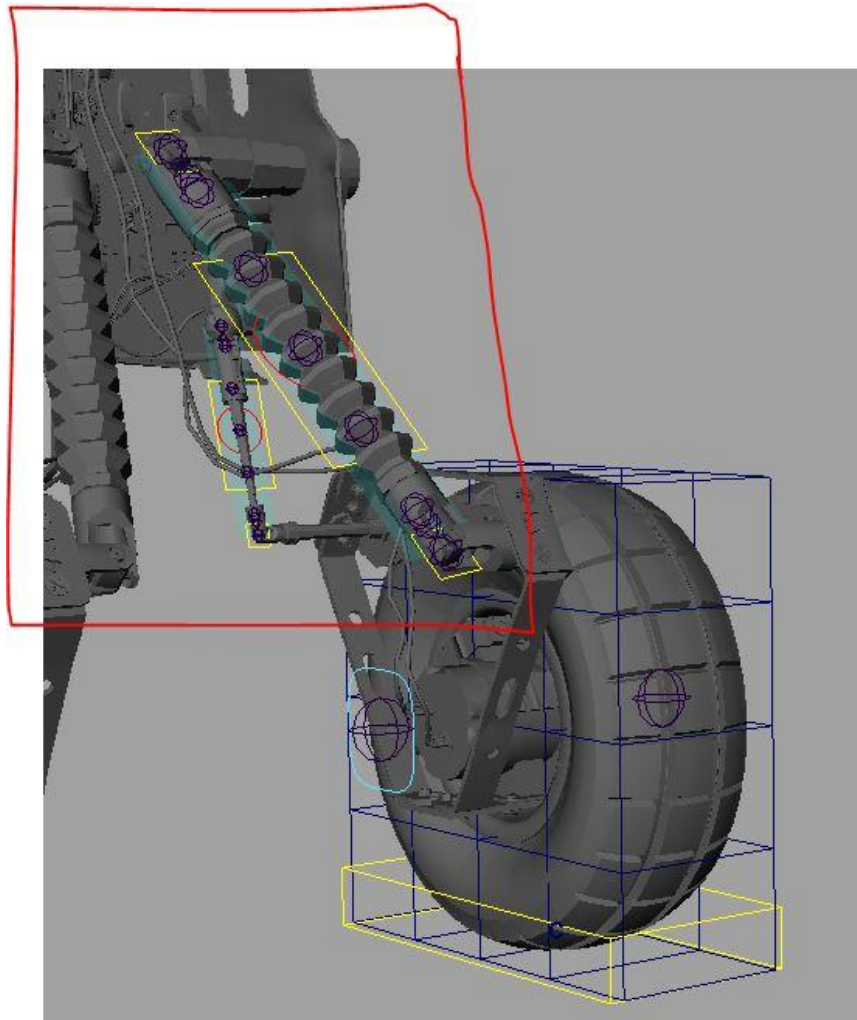


Fig.31.

Ribbon spine

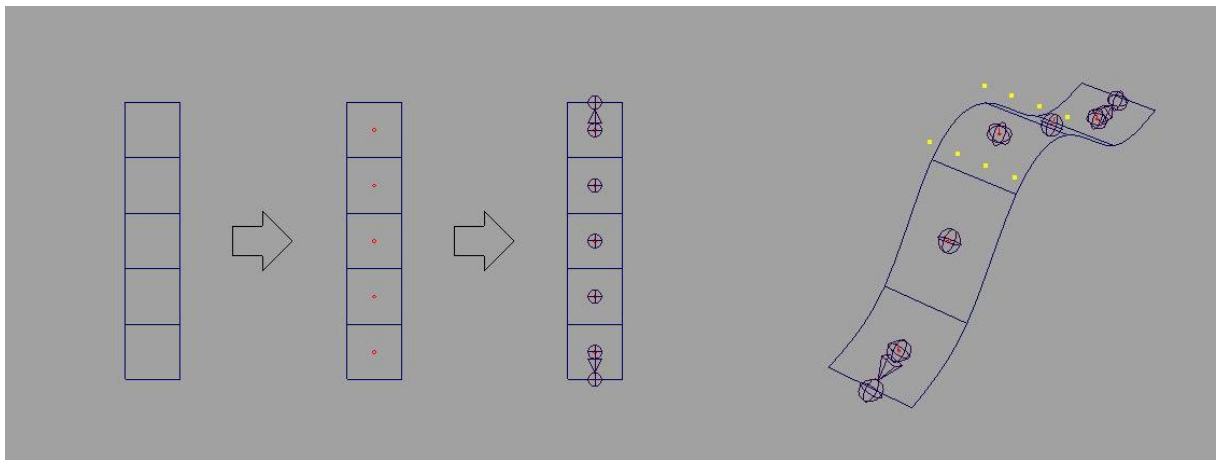


Fig.32.

Le principe de la « ribbon spine » est qu'elle utilise des « follicles », éléments qui servent à la base à attacher les cheveux (donc une curve) sur une surface. Ces follicles prennent leurs positions par rapport aux « uvs » de la géométrie. Si l'on crée une surface, puis un « hair système » sur cette surface et qu'on en conserve uniquement les follicles, que l'on parente un joint sous chaque follicle, on obtient un système qui peut permettre le « squash and stretch » et le « twist ». En déplaçant les points de la surface on manipule le positionnement des joints. Cependant il n'est pas du tout pratique de manipuler des points, pour les manipuler il va falloir créer un système de contrôle annexe.

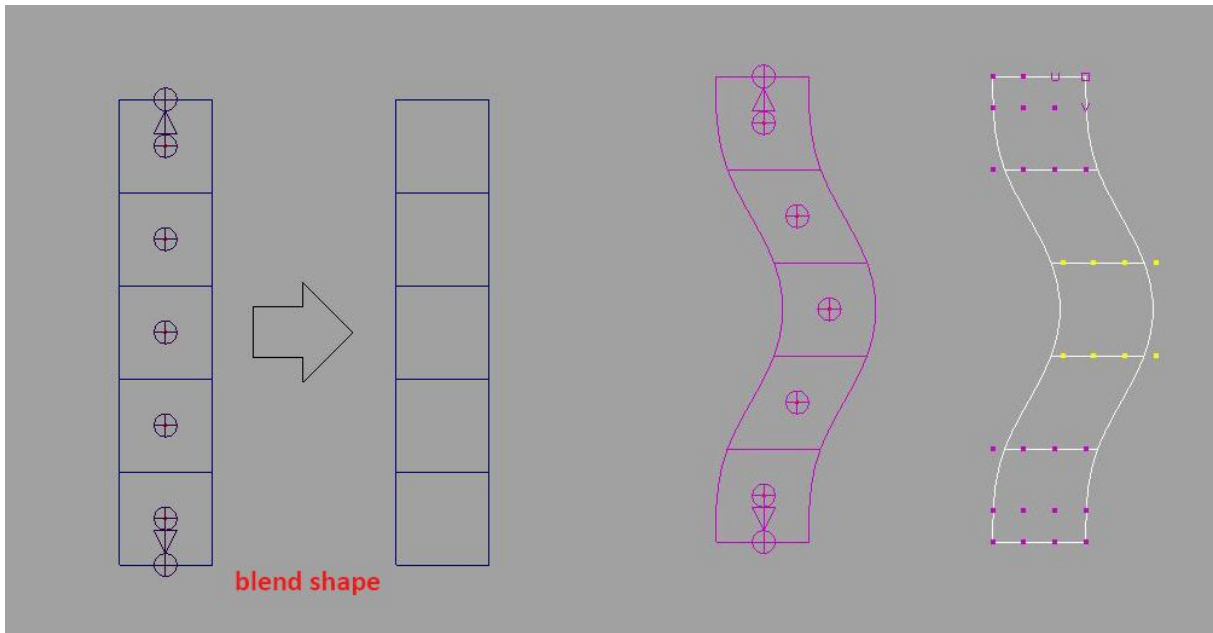


Fig.33.

Pour contrôler ma « ribbon » je vais d’abord utiliser un « blend shape ». Ensuite je vais créer une « curve » en mode 2, qui va me donner un « curve » avec trois vertex qui correspondront au trois contrôleurs que je souhaite créer. Et je vais appliquer à la surface du « blend shape » un autre déformeur le « wire deformer », qui consiste à déformer une géométrie grâce à une « curve », ainsi ce ne sont plus les points de ma surface mais les points de la « curve » qui contrôlent ma « ribbon ».

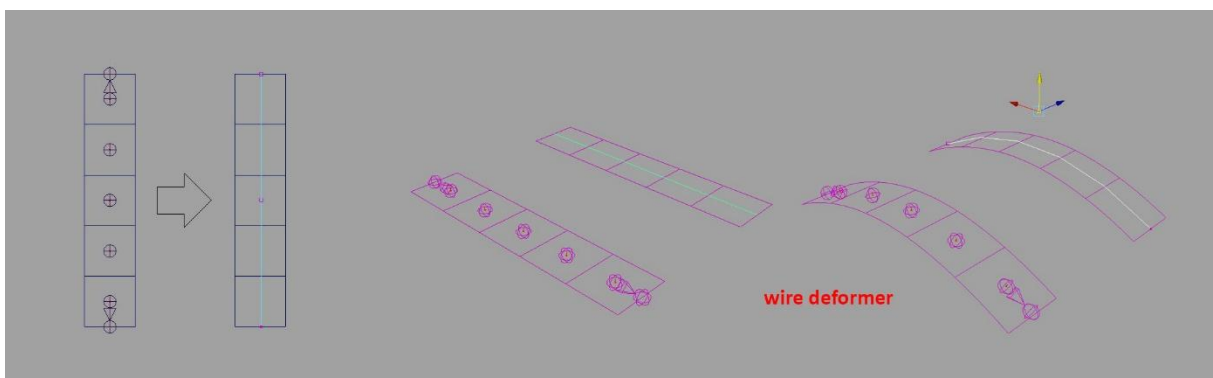


Fig.34.

Puis je vais attacher trois « cluster » à ma curve : le premier se fait en sélectionnant deux points, un sur l’extrémité et l’autre au centre ; le deuxième se fait en sélectionnant juste le point du centre ; le troisième se fait en sélectionnant les deux derniers points (la seconde extrémité et le

centre). Il faudra que je replace les pivots de mes « clusters » et leur origine pour qu'ils coïncident avec les deux extrémités de ma « ribbon ». Et pour que la déformation se face de manière linéaire j'ajuste dans le « Component Editor » les poids d'influences des « clusters » de chaque extrémité sur le vertex central de ma « curve ».

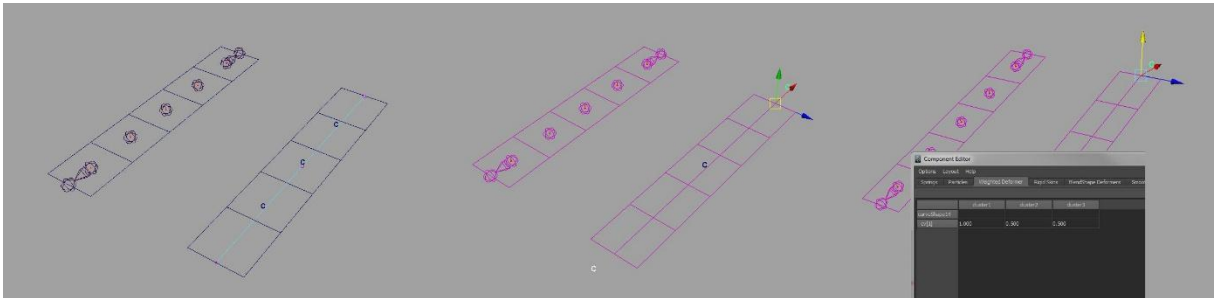


Fig.35.

Mes « clusters » sont alors bien réglés et ma ribbon se déforme correctement. Il me reste à créer des contrôleurs sous forme de curve car je ne souhaite pas qu'il soit possible de toucher directement aux clusters.

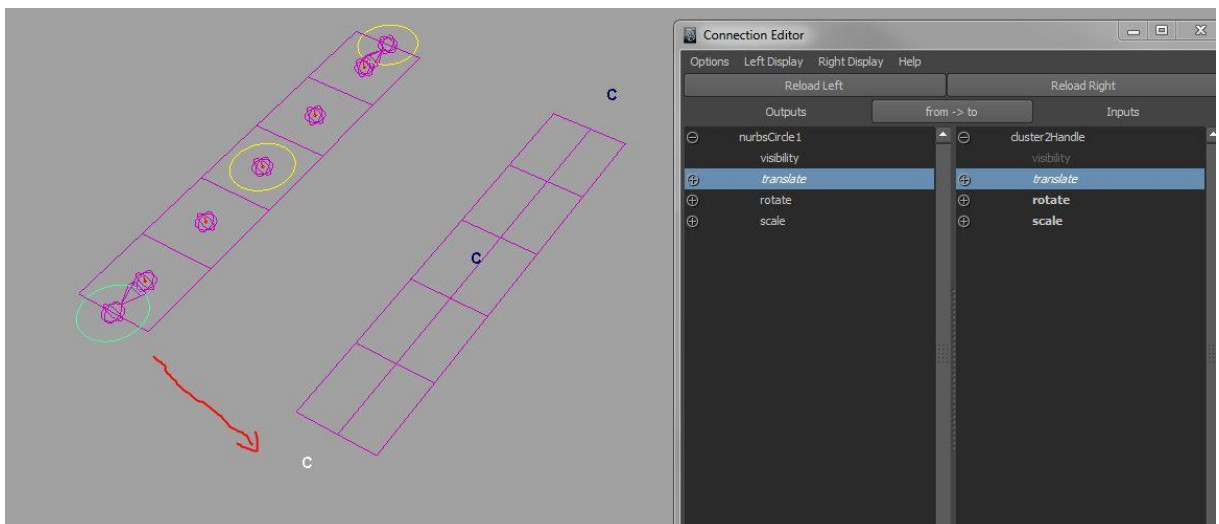


Fig.36.

Je créé donc un contrôleur pour chaque cluster et je connecte leur valeur de translation en connexion directe. Ma ribbon est alors bien contrôlée par mes nouveaux contrôleurs.

Mes contrôleurs ne sont pas capables de faire « twister » ma ribbon. Pour y arriver je vais ajouter un nouveau déformeur, le « twist deformer ». Il me suffira alors de connecter le rotate X du contrôleur du haut de ma « ribbon » à l'attribut « start angle » de mon node « twist », et celui du contrôleur du bas à l'attribut « end angle » de mon node « twist ».

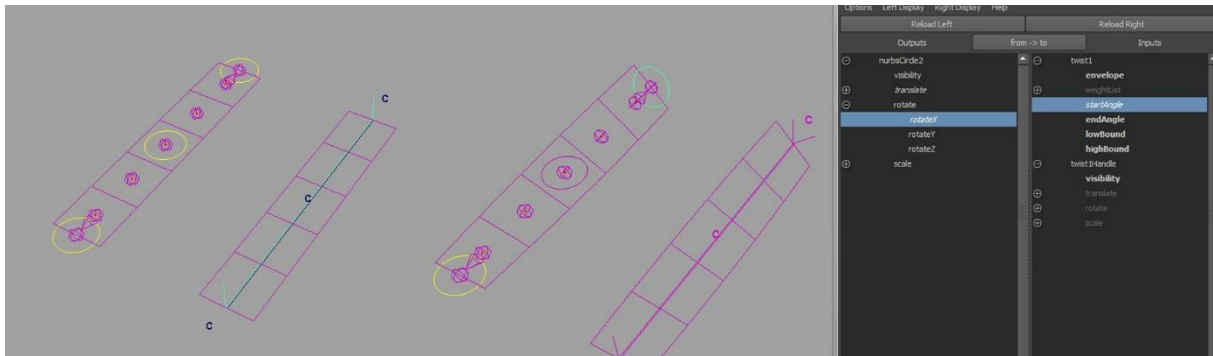


Fig.37.

La préservation du volume

Bien que la préservation du volume n'ait pas été nécessaire sur ce projet, j'ai mis en place ce système sur ma « ribbon spine » pour prévoir le cas où je la réutiliserais pour une colonne vertébrale, cette capacité serait alors très utile. En quoi consiste la préservation du volume ? En fonction de la longueur de ma ribbon je vais effectuer un scale sur mes joints pour déformer la géométrie, et ainsi garder son volume. Ce qui provoquera un effet d'écrasement et d'étirement.

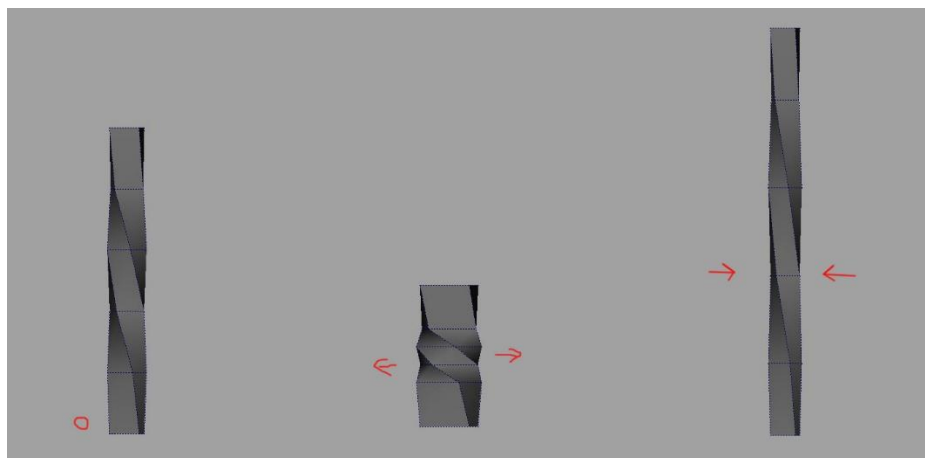


Fig.38.

Pour conserver le volume j'applique donc la formule suivante : si X est l'axe dans lequel ma « ribbon » stretch, la valeur de scale Y et scale Z est égale à 1 divisé par la racine carrée de scale X ¹¹. Pour effectuer ma préservation de volume dans Maya, j'ai besoin d'abord de connaître la longueur de la « ribbon », vu que j'utilise une « curve » comme « defomer » je peux connaître la longueur de cette « curve » grâce au node « curve info ». Puis une fois cette longueur connue, il me faut donc trois nodes « multiply divide » pour effectuer l'opération. Le premier me permet de normaliser la longueur en divisant sa valeur par elle-même. Le second calcule la racine carrée de cette longueur normalisée en utilisant sa fonction « power ». Enfin le troisième sert à diviser 1 par le résultat de la racine carrée de la longueur. Une fois ce résultat obtenu je le connecte au scaleY et scaleZ des joints de ma « ribbon ».

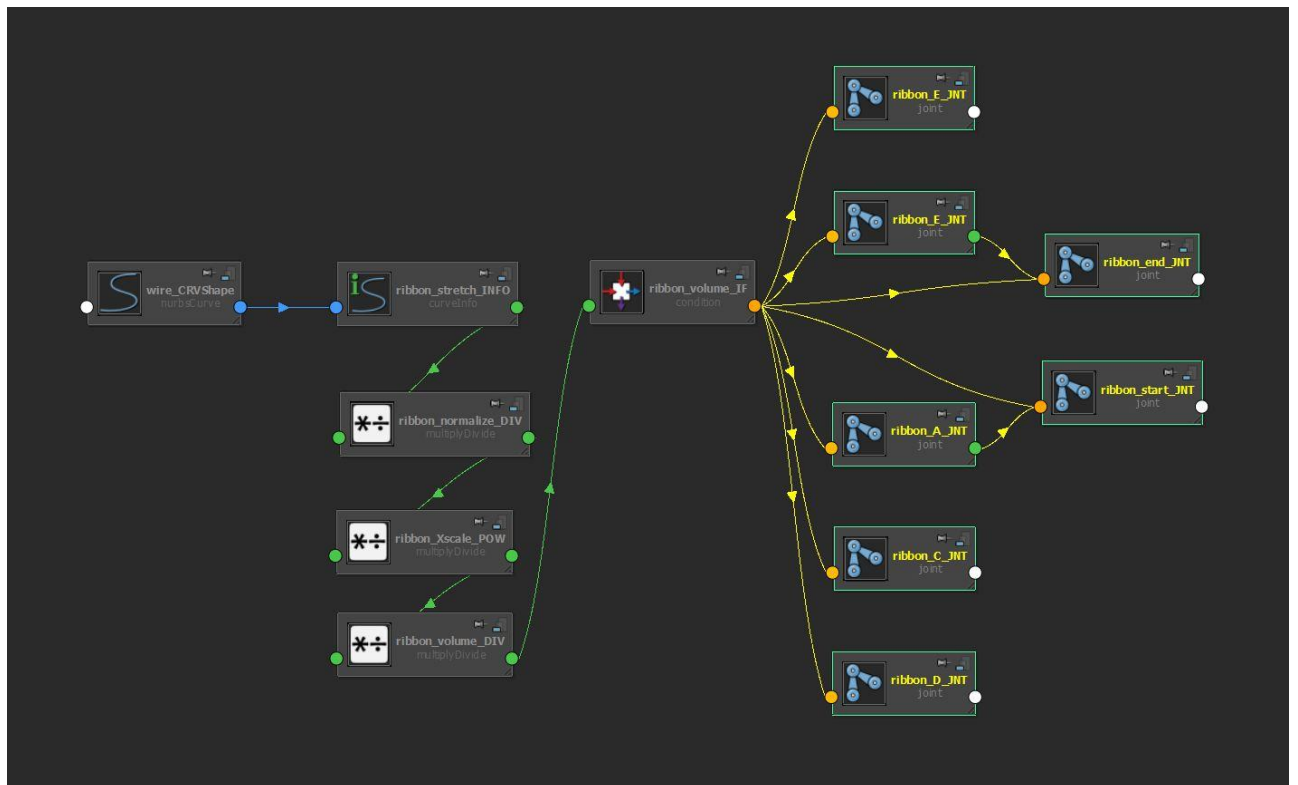


Fig.39.

¹¹ Maya Learning Channel, Creating a Character Rig - Part 5: Torso squash and stretch, [en ligne], https://www.youtube.com/watch?v=lp-5PD3aNIg&list=PL8hZ6hQCGHMXXKqaX9Og4Ow52jsU_Y5veH&index=6, consulté en mai 2014.

Le corps, les hélices et la courroie

Pour le corps, j'ai d'abord placé le « root joint », le centre de gravité de Reebot et j'ai créé un contrôleur pour ce joint. Je n'ai pas parenté une « shape » car ce contrôleur doit aussi permettre des translations, je l'ai donc placé dans un groupe. Ensuite j'ai créé deux contrôleurs, un relatif au centre du monde et un local, à l'intérieur duquel je place mes différents modules. Sur le contrôleur du « root » j'ai créé un attribut « Reebot Start » et un « Speed ». Quand l'attribut « Reebot Start » sera « on » les hélices et la courroie gauche se mettront en marche. L'attribut « Speed », correspondra à la vitesse de rotation des différents éléments.

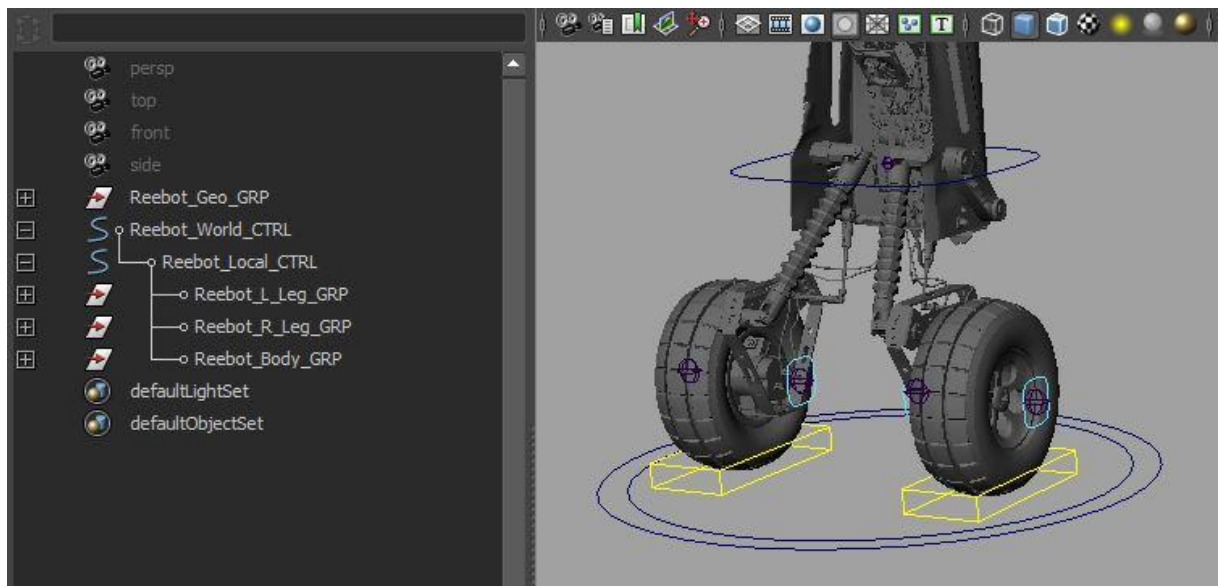


Fig.40.

Pour « rigger » la courroie, j'ai utilisé une curve et le « wire deformer ». J'ai pu remarquer que le « wire deformer » ne fonctionne qu'avec un mesh parfaitement rond au départ. J'ai donc dû recréer la modélisation de la courroie, pour qu'elle soit ronde et j'ai appliqué le deformer. Puis à l'aide des points de la curve j'ai replacé la courroie et lui est redonné sa forme initiale. Ensuite il suffit de faire une rotation de la géométrie et non de la curve pour quelle suive la forme de la curve. Pour l'hélice j'ai juste placé un joint au centre de la pale. Voici le node graph (Fig.41) que j'ai utilisé pour effectuer ces rotations, à noter que je préfère toujours passer par des nodes et pas par des expressions, les nodes sont plus rapides à être calculer dans Maya.

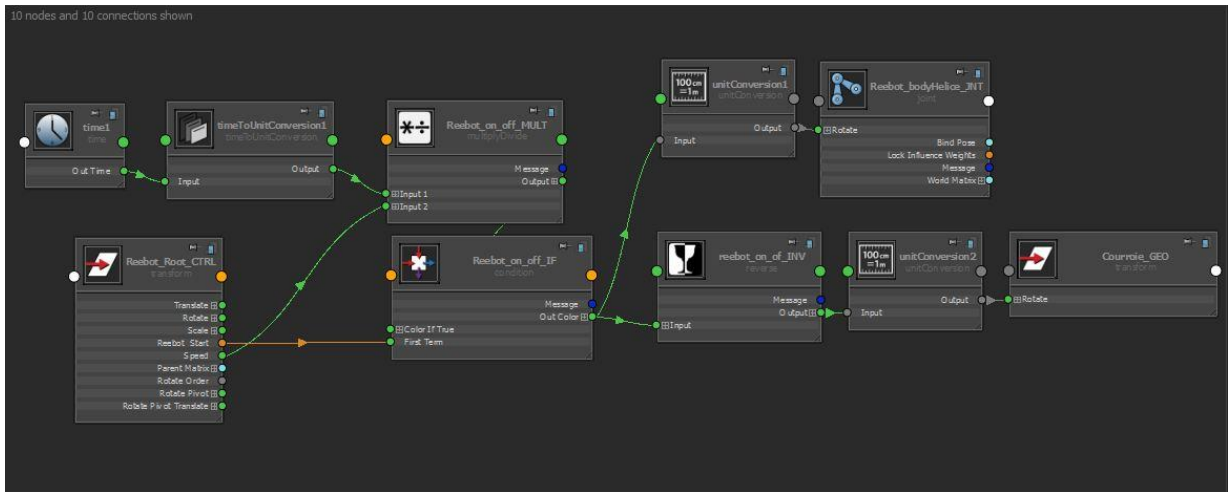


Fig.41.

Les bras coulissants

Les bras sont sans doute la seule partie de Reebot ressemblant de près ou de loin à un être humain. Je ne vais évidemment pas revenir sur les bras que nous avons déjà vu au début de ce mémoire, mais plutôt sur une particularité propre à Reebot, car ses bras peuvent coulisser le long de son corps. J'ai simplement parenté l'ensemble du bras sous un « locator », j'ai attaché le « locator » à la « curve » en utilisant un « motion path » dans Maya, *Animate > Motion Path > Attach To Motion Path*, puis j'ai créé un attribut « Arm Slide » sur l'un des contrôleurs du bras, et j'ai connecté en connexion directe cet attribut à l'attribut « U Value » du motion path.

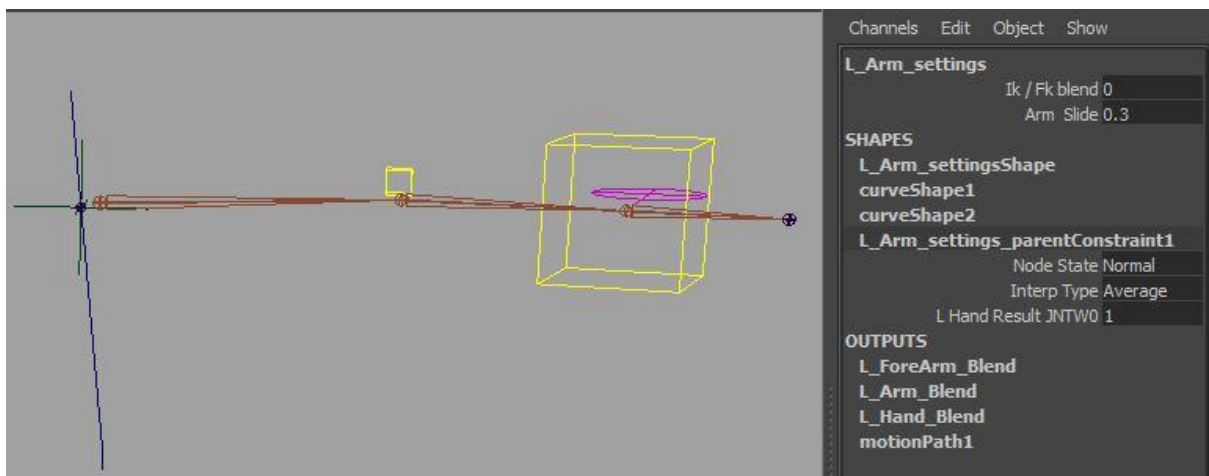


Fig.42.

3.1.2 Intégrer son rig au sein d'un pipeline de film d'animation

Pour bien comprendre le travail du rig il est aussi important de revenir sur certains aspect du pipeline. Comme ce film a été réalisé dans une petite structure, j'ai pu être présent depuis le rig jusqu'au rendu, et il a fallu bien penser, dès le début du projet, à la création d'un pipeline fonctionnel. Par le mot fonctionnel je veux dire qu'il est important que les équipes qui interviennent sur le projet aient la possibilité de le faire chacune en même temps, plus précisément que le modelleur puisse travailler pendant que de mon côté je commence le rig, afin que les animateurs aient au plus vite un rig fonctionnel, et que les graphistes du rendu puissent commencer à rendre les plans alors que l'animation n'est pas encore terminée.

Il faut d'abord penser à hiérarchiser nos scènes Maya. Vu qu'il y a de nombreux intervenants sur le projet, il est logique qu'il ne soit pas possible d'effectuer tout le travail, depuis la modélisation jusqu'au rendu, dans une même scène Maya. Il serait impossible de tous travailler sur le même fichier, et en plus il ne serait pas possible de faire évoluer le projet par parties distinctes. Pour rendre tout cela possible, il faut découper le projet en étape. Ma méthode est la suivante. D'abord je crée un dossier qui porte le nom de mon projet. Dans ce dossier je crée par exemple trois autres dossiers, un nommé « séquences », qui contiendra un dossier pour chacun de mes plans dans lesquels je crée un projet Maya ; un nommé « personnages » qui contiendra un projet Maya pour chaque personnage ; et un troisième nommé « décors » qui lui aussi contiendra un projet Maya par décors. De cette manière mon projet de film est correctement organisé et il sera simple de s'y retrouver. Chaque dossier « scene » correspondant à chacun des projets Maya contient donc plusieurs dossiers. S'il s'agit d'un personnage il contiendra par exemple un dossier « modélisation », un dossier « rig », et un dossier « shading », et chacun de ces dossiers contiendra les scènes Maya correspondantes. S'il s'agit d'une séquence, il contiendra un dossier « layout », un « animation », un « rendu » et souvent je préfère aussi créer un dossier « caméra » dans lequel j'exporte la camera finale de chaque plan.

Maintenant que nous avons compris l'importance de bien hiérarchiser nos scènes Maya, comment travailler avec ? Tout est donc décomposé en parties distinctes, modélisation, rig, shading etc... Mais comment récupérer ces étapes de travail et comment les intégrer au nôtre ? Il faut travailler en référence **File > Create Référence** car cette référence est actualisable, c'est à dire que si je suis en train de modifier mon rig, l'animateur pourra mettre à jour, actualiser le

rig sur lequel sa scène d'animation prend référence. Ainsi même si le rig est modifié l'animateur ne perdra pas son travail d'animation. Un autre point important est le cache d'animation. Une scène de rendu est en bout de chaîne de production, elle doit donc importer en référence beaucoup d'éléments venant d'autres scènes, comme par exemple l'animation du personnage. Cependant même si on utilise le système de référence, ces scènes peuvent devenir très lourdes, une fois tous les éléments importés. Pour le rendu il est préférable que le rig ne soit pas présent dans la scène, car il serait très lourd. Sur Reebot par exemple, les de 2000 nodes « Skin Cluster » ainsi que tous les nodes qui composent son rig alourdiraient considérablement la scène (j'avoue qu'il n'a pas été totalement optimisé au niveau du nombre de pièces !!). Imaginons qu'il y ait plusieurs personnages, la scène pourrait devenir encore plus difficile à utiliser. En plus, sur le coup d'une mauvaise manipulation, il serait possible lors du rendu de modifier l'animation. C'est pourquoi on utilise le cache d'animation.

*Tip : Pour faire un cache d'animation il faut sélectionner toute la géométrie que l'on souhaite mettre en cache, puis dans l'onglet **Animation – Geometry Cache – Export Cache***

Le cache d'animation est un fichier qui contient les coordonnées de chaque vertex de chacune des pièces sélectionnées. Plusieurs options sont possible, je précise juste qu'en général je coche « One file per frame » ce qui permet de ne pas devoir recalculer toute la séquence si je dois modifier le cache d'une partie de mon animation. Une fois ce cache calculé dans ma scène de rendu j'importe en référence la scène de Reebot avec ses « shaders » et j'applique le cache dessus. Ainsi si l'animation est modifiée, il suffira de recharger le nouveau cache. Ce cache d'animation sera en quelque sorte la référence de mon animation. A noter l'importance dans ce type de projet de rendre les décors et l'animation des personnages à part. Un autre conseil en tant que rigger, j'ai pu remarquer que les « uvs » ne se mettent pas à jour en référence un fois que les géométries sont skinnées, il est donc préférable de ne pas importer la géométrie en référence dans la scène de rig si les « uvs » ne sont pas terminés. Je sais bien que dans une production idéale cette remarque peut paraître stupide, mais dans les faits c'est souvent le cas et donc je préfère importer la géométrie directement dans ma scène de rig et corriger si besoin mes « uvs » à l'intérieur.

Pour en terminer avec ce projet je voulais revenir sur quelques méthodes qui m'ont permis de palier à différents problèmes rencontrés lors de cette production. Dans un premier temps, l'importance des noms d'objets et des noms des scènes. Si je fais une nouvelle version de mon

rig, je veux être sûr d'en conserver la dernière version fonctionnelle, je vais donc implémenter ma version et la renommer. Mais il faut faire attention à la façon dont on implémente les scènes. Si j'implémente *mascene_01.ma* en *mascene_02.ma*, vu que le « namespace » prend le nom de la scène appelée, le nom de mes objets va changer, ce qui va forcément créer des problèmes. Pour maintenir les noms toujours identiques malgré les changements de version il existe une méthode qui consiste à numéroter les versions toujours en début de nom, par exemple *001_mascene.ma*. Maya n'autorise pas qu'un nom commence par un nombre, il supprimera donc le numéro de version qui précède le nom de la scène, ainsi même si le numéro de version change, les noms des objets seront toujours identique

3.2 Elaïa : rig facial d'un personnage cartoon

Dans ce troisième chapitre je souhaite revenir sur un autre projet Elaïa film réalisé au cours de cette année dans le contexte d'un projet de trois semaines. Sur ce projet outre le rig du corps, l'enjeu principal fut de réaliser un rig facial. Je n'avais pas peur de rencontrer des difficultés techniques pour réaliser le rig du visage mais j'avais plus de difficultés à imaginer comment aborder le rig facial, je ne savais pas trop ce qu'il fallait faire en termes d'animation, de décomposition des expressions, etc.

Tout d'abord comme pour Reebot j'ai commencé par analyser ma modélisation pour répondre aux premières questions que je devais me poser en abordant ce rig facial. Quel type de rig facial je souhaite réaliser ? Quel sont les émotions qu'il va devoir exprimer ? Par quel moyen vais-je pouvoir faire passer les émotions de mon personnage ?

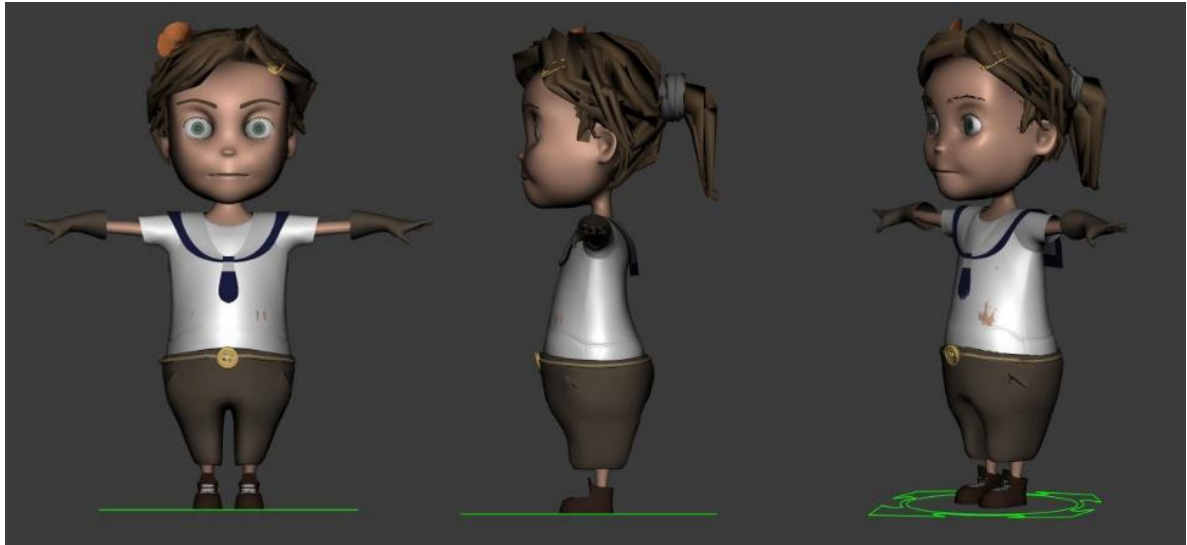


Fig.44.

Stylisé ou réaliste

Elaïa est une petite fille d'environ une dizaine d'année, elle est un personnage stylisé (ou cartoon), par opposition à un personnage dit réaliste : elle n'a pas de proportions réalistes et que le type d'animation recherché n'est pas non plus une animation vouée à faire croire à la réalité de la représentation humaine mais plutôt à une adaptation de celle-ci dans un but d'originalité et de créativité, car elle va prendre place d'un univers imaginaire et onirique. L'animation cartoon permet plus de dynamisme et de liberté dans les mouvements.

On n'apporte pas les mêmes résultats avec un personnage stylisé ou un personnage réaliste. Chacun va apporter ses particularités de modélisation et donc de rig. En même temps, ils sont tous les deux une adaptation, une copie d'une certaine réalité. Du coup, pour mon rig facial, quel référent dois-je choisir ? Est-ce que les différences entre stylisé et réaliste se retrouvent aussi dans l'animation du visage ?



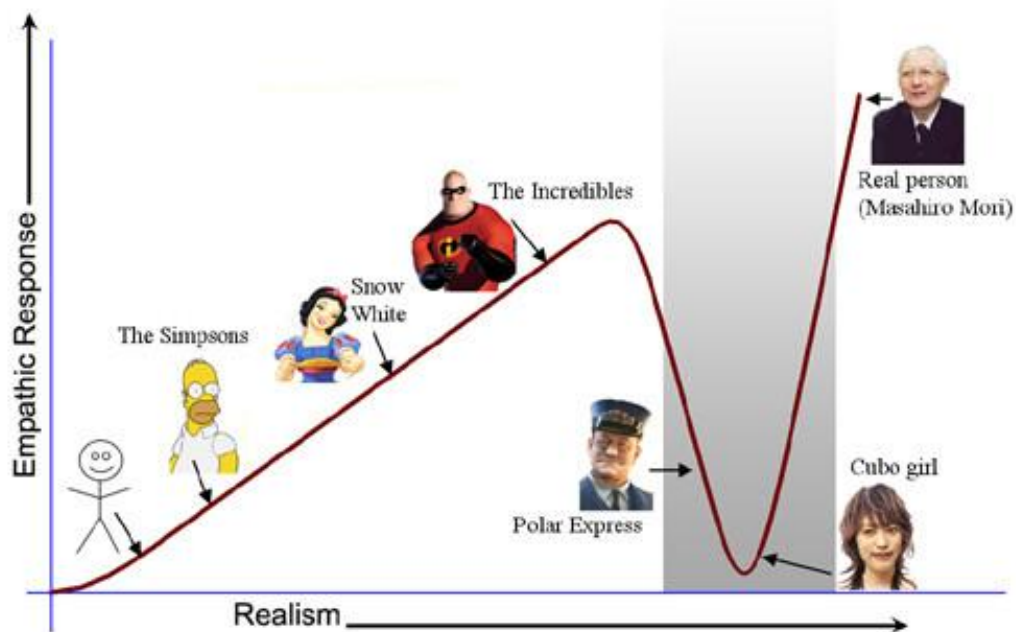
Dans cette série d'images on peut reconnaître un cheval, une jeune fille, et un éléphant. Même si on arrive à les nommer par ce que nous avons tous déjà vu un cheval blanc, une jeune fille blonde et un éléphant, on voit bien qu'ils ne sont pas représentés d'une manière réaliste, dans le sens où dans la réalité, il serait impossible de les rencontrer avec ces expressions : le même type de regard dans la construction, la recherche des lignes comme celle des sourcils, typiques des dessins animés qui cherchent à exacerber les expressions afin de faire passer le maximum d'émotions possibles dans un but narratif.



Ces personnages sont par contre réalisés de manière réaliste, même s'ils représentent des créatures imaginaires. S'ils cherchent à se rapprocher au plus proche de la réalité, c'est pour accroître la crédibilité du personnage même si le public sait pertinemment qu'il n'existe pas, en vrai. Du coup, les expressions du visage sont très proches des expressions humaines, le détail des yeux, les rides, le grain de la peau, sa transparence. Il arrive souvent que l'animation de ce type de personnages soit faite en « motion capture », comme ici.

Mais nous avons vu que si des films utilisent la mocap avec un rendu expressif, stylisé, cela ne fonctionne pas forcément (on se rappelle de *Polar Express*, 2004).

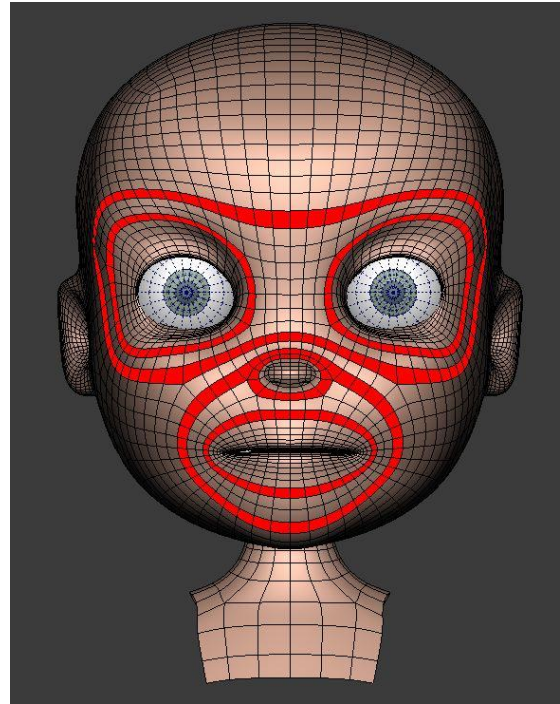
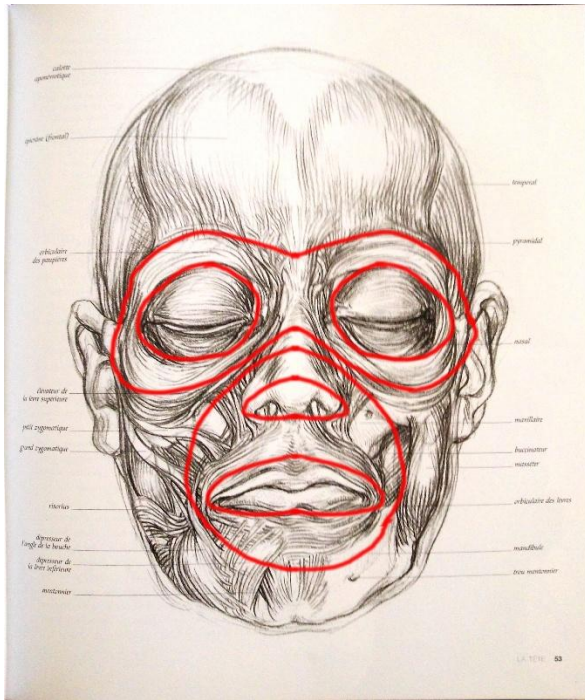
Cela peut s'expliquer par le phénomène d'« uncanny valley ». On n'a pu remarquer en effet que le trop plein de réalisme pour représenter des personnages qui se rapprochent trop de l'homme pouvait vite devenir troublant, et qu'on finisse par le rejeter ; tandis qu'un personnage stylisé et bien pensé laisse plus de place à la projection et à l'empathie.



Une bonne topologie

Tout d'abord la topologie c'est la manière dont sont organisés les polygones sur notre modèle. Pour permettre au modèle de se déformer convenablement ils doivent être organisés en « loops » bien précis. Le visage est la partie du corps humain qui se déforme le plus, il est constitué d'environ 40 muscles, tous utilisés pour créer nos diverses expressions faciales. Je ne vais pas entrer dans trop dans les détails de l'anatomie faciales¹², mais pour bien comprendre comment réaliser une bonne topologie pour la modélisation d'un visage il est essentiel de comprendre comment il est constitué. Les différents « loops » de notre modélisation devront permettre à notre visage d'exprimer la déclinaison des expressions. C'est pourquoi je m'appuie toujours sur une référence anatomique comme point de départ, même si on s'en détourne un peu à la fin pour notre personnage cartoon.

¹² Anatomie pour l'artiste, Sarah Simblet, Dessain et Toltra, 2001.



Les expressions faciales

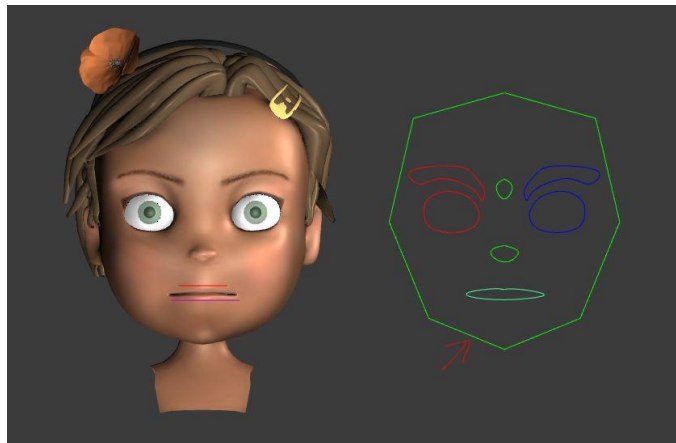
Dans un premier temps il est très intéressant de voir comment se forment les expressions du visage. En consultant des ouvrages dédiés aux artistes¹³ sur les expressions faciales ou encore sur l'anatomie et le positionnement des muscles du visage, j'ai d'abord pu constater que les expressions faciales se traduisaient souvent par le mouvement de quatre parties majeures, les sourcils, les yeux, le nez, et la bouche. Il m'a fallu dans un premier temps définir quelles sont les expressions que l'on peut faire avec son visage, et ce n'est pas évident car évidemment elles sont infinies. Cependant autre chose m'a paru marquant, tout homme est capable de discerner des sentiments comme la joie, la tristesse, la surprise, la peur, la colère, le dégoût ou encore le mépris. Cette capacité de discernement est innée et cependant il est difficile de dire avec précision, quelle partie du visage bouge et à quelle moment. La référence dans le domaine de la décomposition des émotions est le *facial action coding system (FACS)*¹⁴, c'est une méthode de description des mouvements du visage datant de 1968 dans laquelle sont découpés les

¹³ Expressions, a visual reference for artists, Mark Simon, 2005.

¹⁴ Facial Action Coding System, The manual, Paul Ekman, Wallace Friesen, Joseph Hager, CD Rom, HTML demonstration version (en ligne) <http://www.face-and-emotion.com/dataface/facs/manual/TitlePage.html>

mouvements du visage en unités d'action. Ce sont ces unités d'action qui m'ont particulièrement intéressées pour la réalisation de mon rig facial car elles m'ont permis de décomposer les expressions du visages en mouvements bien distincts.

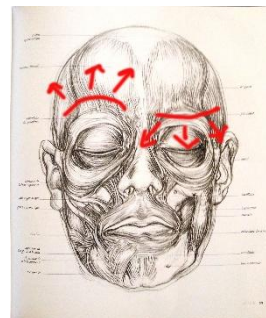
Chacun de ces mouvements, ou unité d'actions, sera donc une animation possible d'une partie de mon visage. J'ai opté de réaliser mon rig à l'aide de joints et de « set driven keys ». J'ai donc commencé par créer des contrôleurs pour chacune des parties de mon visage en dissociant droite et gauche afin de pouvoir donner de l'asymétrie à mon visage.



Sur chaque contrôleur, j'ai créé des attributs correspondants à mes différentes unités d'actions. J'ai commencé par les sourcils. D'après le FACS, mais aussi mon analyse de différentes expressions, il est clair que les sourcils peuvent principalement bouger de deux façons : monter et descendre, et se froncer soit bouger de l'intérieur vers l'extérieur. Je décide de découper le sourcil en trois parties, intérieur, milieu et extérieur pour contrôler au mieux mes expressions.

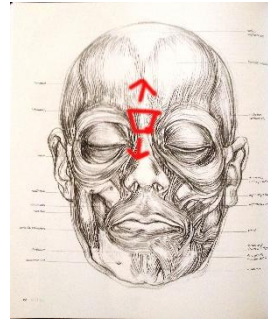
J'ai donc décidé de créer les attributs suivant :

- Brow Inner Up Down
- Brow Mid Up Down
- Brow Outer Up Down
- Brow Squeeze In Out



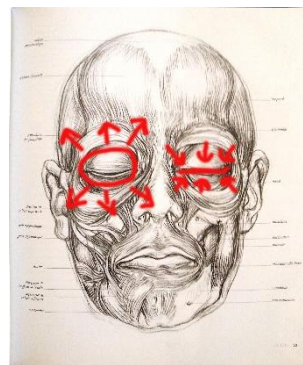
J'ai aussi créé un contrôleur pour la partie centrale entre les deux sourcils auquel j'ai ajouté deux attributs :

- Brow Up Down
- Brow In Out



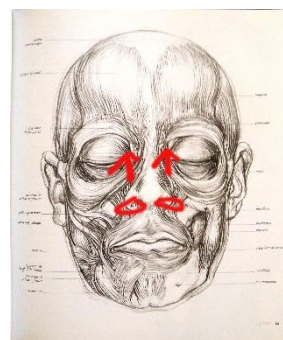
Ensuite pour les yeux je vais créer des attributs pour permettre aux paupières hautes et basses de se baisser et de se lever. Afin de donner le maximum de contrôle à l'animateur sur la forme de l'œil je vais aussi répartir le mouvement des paupières en trois parties Inner Mid Outer, je donne donc à mes deux contrôleurs (un pour chaque œil) ces attributs :

- Upper Inner Up Down
- Upper Mid Up Down
- Upper Outer Up Down
- Lower Inner Up Down
- Lower Mid Up Down
- Lower Outer Up Down

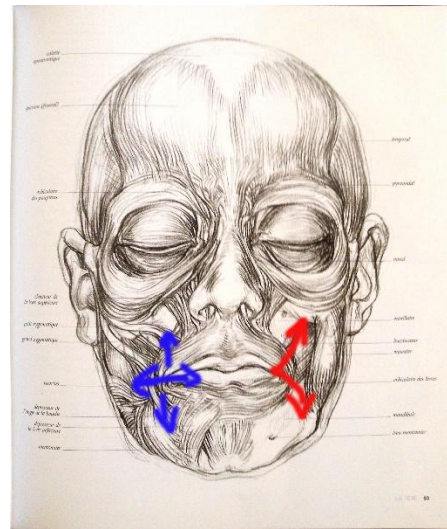
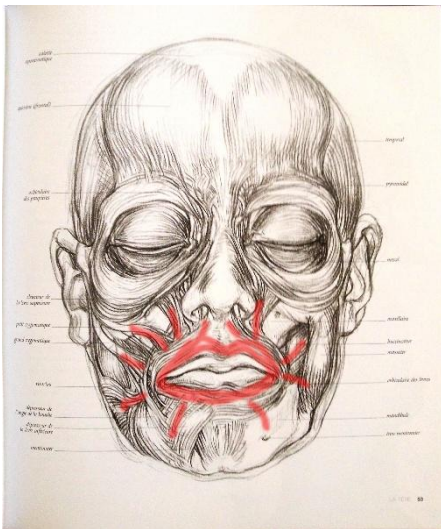


Pour le nez je vais reproduire deux mouvements, la monter du nez lorsque l'on est en colère ou agressif, puis l'écartement des narines. A la différence des contrôleurs précédents, vu que j'ai fait un seul contrôleur pour le nez je vais distinguer la gauche et la droite, donc pour mes deux mouvements je vais créer quatre attributs :

- L Flare
- R Flare
- L Sneer
- R Sneer



Maintenant la bouche. Celle-ci est bien plus complexe que toutes les parties du visage que nous avons faites précédemment. A la différence des sourcils ou du nez, la bouche n'est pas directement positionnée sur un os elle s'appuie sur les dents mais est raccordée par des muscles à notre crâne. Ces muscles sont disposés tout autour d'elle ce qui lui permet de bouger à peu près dans toutes les directions. De plus, les lèvres ont aussi des capacités de mouvements comme se pincer, avancer vers l'avant, ou rouler de l'arrière vers l'avant. Et évidemment l'os et les muscles de la mâchoire permettent d'ouvrir la bouche. Pour obtenir un sourire, ou une moue, je vais devoir cumuler deux attributs : si je donne la possibilité aux coins de la bouche de bouger de haut en bas et de gauche à droite, le cumul haut et gauche me donnera le sourire, bas et gauche me donnera la moue.



Je vais donc créer ces attributs pour la bouche, il faudra encore séparer gauche et droite :

- Jaw Open Close
- L Mouth Corner In Out
- R Mouth Corner In Out
- L Mouth Corner Up Down
- R Mouth Corner Up Down
- Mouth Up Down
- Mouth Left Right

Et pour le contrôle des lèvres, je sépare lèvre du haut et lèvre du bas :

- Upper Lip Roll In Out

- Upper Lip Up Down
- Upper Lip In Out
- Lower Lip Roll In Out
- Lower Lip Up Down
- Lower Lip In Out

Evidement il est possible de rajouter d'autres attributs comme par exemple la possibilité de monter la lèvre supérieure à trois endroits différents : gauche milieu et droite, alors que pour l'instant on a juste donné la possibilité d'animer la lèvre de haut ne bas sans distinction de côté. Mais tous ceux qu'on a défini pour l'instant suffisent pour commencer à positionner nos joints et les contrôleurs de ces joints pour réaliser nos expressions. D'autre part, tous les attribut que nous venons de voir doivent avoir un maximum de 10 et un minimum de -10, qui sera le « range of motion », la valeur d'échelle de nos mouvements, pour effectuer nos « set driven keys ».

Placer les joints et créer les contrôleurs

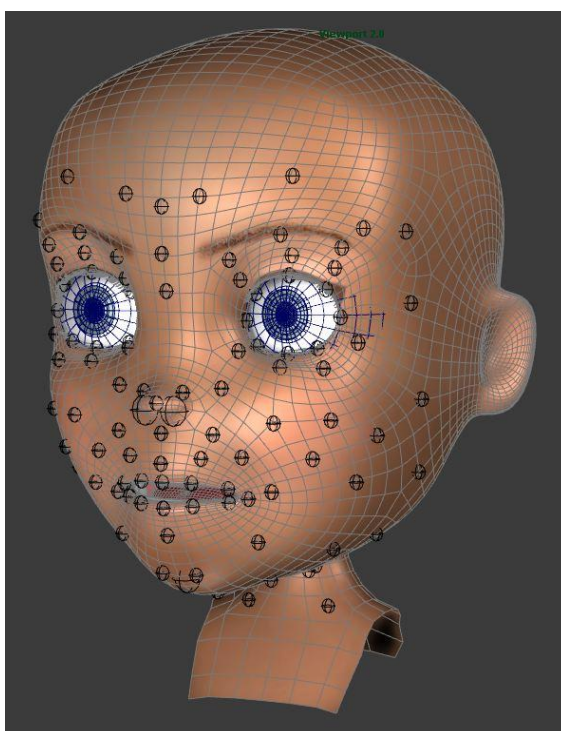
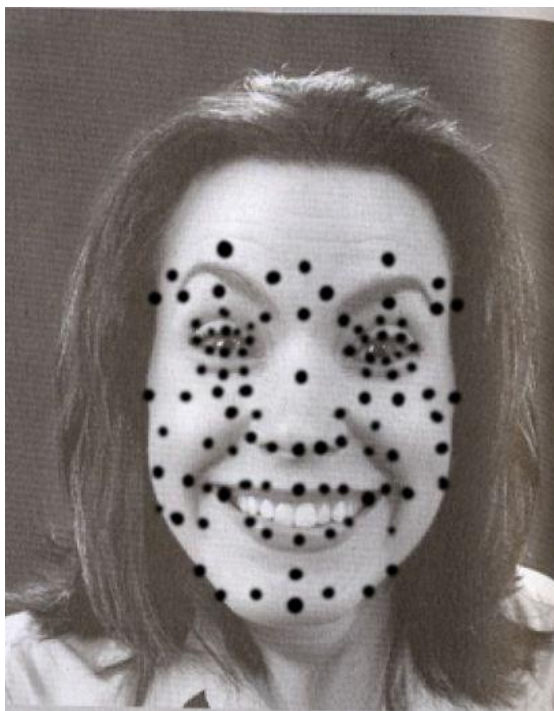
Placer les joints sur un visage peut paraître au départ un peu déconcertant. En effet le placement des joints aura beaucoup d'influence sur nos déformations et donc sur la réalisation de nos expressions faciales. Comme pour la topologie ou les attributs il est bon de commencer par un analyse de l'anatomie du visage.



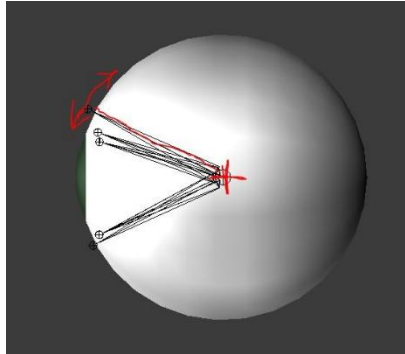
Je prends donc une expression très marquée comme celle de cette femme, afin de bien comprendre quelles déformations je serai amené à obtenir. Je commence donc par signifier les creux « creases »(en bleu), qui vont donc avoir besoin d'un joint pour se former. Puis ensuite je définis les bosses, les plis, ou plutôt les zones mobiles (en rouge) de notre visage qui formeront nos expressions, elles auront besoin de deux joints. Puis enfin les zones fixes, celles qui ne bougeront pas et qui maintiendront la forme de notre visage (en vert), ce sera donc les parties osseuses du visage, elles nécessiteront également un joint.



C'est lignes directrices me permettent alors de d'établir le positionnement des joints de mon visages.



Pour placer les joints des paupières, il faut placer un joint au centre de l'œil puis le dupliquer et amener ce deuxième joint jusqu'à la paupière où il servira à la « skinner ». De cette manière la paupière sera dirigée en rotation depuis le centre de l'œil, en suivant le premier joint.



Les joints sont maintenant placés, il faut cependant encore créer pour chaque joint un contrôleur. Les joints devront être contrôlés en translation et en rotation il faudra donc créer comme nous l'avons déjà vu dans le chapitre sur les contrôleurs, un « locator » à l'intérieur d'un group « offset », et contraindre ce « locator » aux joints en contrainte parent et scale. Une fois les joints convenablement renommés, je vais utiliser un script pour créer et renommer tous ces contrôleurs car la tâche pourrait être longue et fastidieuse.

Script Python :

```
import maya.cmds as cmds

sel = cmds.ls(sl=1)

for jnt in sel:
    locName = s.replace("_JNT", "_CTRL")
    loc = cmds.spaceLocator(n=locName)[0]
    group = cmds.group(loc, n=loc + "_GRP")
    cmds.delete(cmds.pointConstraint(jnt, group))
    cmds.delete(cmds.orientConstraint(jnt, group))
    cmds.makeIdentity(group, a=1, t=1, r=1)

    cmds.parentConstraint(loc, jnt, mo=1)
    cmds.scaleConstraint(loc, jnt)
```

Création des expressions faciales, « skinning, et « set driven keys »

Tout est donc en place pour commencer l'animation de mes expressions faciales. Le visage est séparé en deux parties symétriques, la droite et la gauche. On commence par réaliser la partie gauche. J'ai choisi d'utiliser les « driven key », elles sont particulièrement adaptées au rig facial en raison de deux traits caractéristiques des « driven keys ». La première est qu'il sera possible de diriger le même contrôleur de joints du visage avec plusieurs attributs. Un node de blend se créera automatiquement, si celui-ci comporte plusieurs animations. La deuxième est qu'en réglant le « post infinity » des courbes d'animation nous allons pouvoir obtenir des valeurs infinies pour pouvoir exagérer nos poses.

Il faudra donc sélectionner les « locators » les charger en « driven » du « set driven key » puis charger en « driver » un des contrôleurs de notre visage. Puis créer des clés à la valeur 0, 10 puis -10 pour chacun des contrôleurs et des attributs. Je recommande vivement pour « skinner » de créer une animation sur la « timeline ». Ce sera beaucoup plus simple de peindre notre skin en faisant des aller-retour entre -10 et 10. On s'est donc occupé de la partie gauche du visage, maintenant que faire avec la droite ? La question de faire un miroir des « set driven keys » d'un contrôleur à un autre, est assez complexe car il n'y a pas d'outil intégré à Maya pour le faire. Effectuer le miroir consiste donc appliquer les animations des « set driven keys » d'un contrôleur de la partie gauche sur un contrôleur de la partie droite, il va donc falloir le faire en plusieurs étapes. Premièrement il va falloir dupliquer les nodes d'animations du contrôleur gauche et les connecter à son équivalent du côté droit. Puis il faut inverser les translations et les rotations sur l'axe perpendiculaire à notre axe de symétrie pour obtenir l'effet miroir escompté. Enfin il va falloir modifier le « driver » de ces animations et donc attribuer mon « set driven key » à l'attribut qui se chargera du côté droit. Ce travail est très long, il existe heureusement des scripts pour le faire j'ai utilisé celui-ci : `copySDK.py`¹⁵.

¹⁵ IsoparmB, `copySDK.py`, <http://www.creativecrash.com/maya/script/copysdk-py>, [en ligne], consulté en mai 2014



Conclusion

L'étude de toutes ses techniques m'a permis de mieux maîtriser mes connaissances, car même si je suis amené à utiliser ces techniques tous les jours, les mettre à l'écrit n'a pas été une chose facile. Synthétiser l'ensemble des techniques dans ce mémoire m'a finalement permis d'en comprendre tous les principes fondamentaux. C'est souvent en revenant sur les bases que l'on s'aperçoit des choses que l'on ne connaît pas. Une fois cet ensemble de connaissances acquis, une autre réflexion s'engage. On peut alors se détacher de la technique, pour penser le *rig* d'une manière plus large : quelles sont les méthodes qui me permettront d'être efficace en production ? Quels sont les outils que je dois utiliser ou développer ? En avançant sur ce mémoire j'ai pu dégager plusieurs pistes. La première est l'approche modulaire : plusieurs papiers techniques¹⁶, traitent de ce sujet et de nouveaux auto-rig¹⁷ sont apparus en utilisant ce concept. J'ai moi-même compris en réalisant mes propres rig l'importance de cette approche, qui permet d'envisager un système de rig réutilisable et universel pour tout type de personnage. Ce mémoire m'a aussi donc clairement permis d'identifier les techniques qui conviendraient le mieux à réaliser à mon tour un concept de rig modulaire. Les techniques avancées que j'ai pu aborder dans ce mémoire comme la « ribbon spine » me donnent aussi des pistes sur les techniques envisageables dans mes productions futures.

¹⁶ Smith, J. & White, J. BlockParty: modular rigging encoded in a geometric volume. in 115 (ACM Press, 2006).

¹⁷ cr_ModularRiggingSystem, gratuit, en MEL, sur le site <http://creaturerigs.com>

Rapid Rig: Modular Auto-rig 1.0.9 MEL, <http://www.creativecrash.com/maya/script/-rapid-rig-modular-procedural-auto-rig-for-maya>

Bibliographie/ Webographie

Ouvrages

MILLER Erick, THURIOT Paul, Unay Jeff, Hyper-realistic Creature Creation, Alias learning tools, 2006.

MILLER Erick, More Hyper-Realistic Creature Creation, Autodesk 2009, sur Maya 2009.

SIMBLET Sarah, Anatomie pour l'artiste, Dessain et Toltra, 2001

SIMON Mark, Expressions, a visual reference for artists, , 2005.

SIMON Mark, Facial Expressions Babies to Teens, a visual reference for artists, 2008.

WILLIAMS Richard, Techniques d'animation. Pour le dessin animé, l'animation 3D et le jeu vidéo, 2001

Articles

Pantuwong, N. & Sugimoto, M. A fully automatic rigging algorithm for 3D character animation. in 1 (ACM Press, 2011).

Smith, J. & White, J. BlockParty: modular rigging encoded in a geometric volume. in 115 (ACM Press, 2006).

McLaughlin, T., Cutler, L. & Coleman, D. Character rigging, deformations, and simulations in film and game production. in 1–18 (ACM Press, 2011).

Borosán, P., Jin, M., DeCarlo, D., Gingold, Y. & Nealen, A. RigMesh: automatic rigging for part-based shape modeling and deformation. *ACM Transactions on Graphics* **31**, 1 (2012).

Bhatti, Z. & Shah, A. Widget based automated rigging of bipedal character with custom manipulators. in 337 (ACM Press, 2012).

Mémoires

Arnaout Tamouze - Comment représenter un monstre ? M2 ATI.

Antoine Brémont – Le rigging ou comment préparer un objet 3D à l’animation ? M2 ATI.

Manuel Alejandro d’Macedo- Rigging, à la recherche d’une cohérence de mouvement et d’expression, Mémoire M2 ATI.

Tutoriaux

RIGGING FOR FEATURE ANIMATION, Fahrenheit Monkey Body Setup

*Aaron Holly a travaillé comme Character Technical Director à Tippett Studio, Industrial Light & Magic, ESC Entertainment, DreamWorks Animation et Disney Feature Animation
Technique ancienne mais une grande partie toujours d’actualité notamment la rebon spine
Et les deux joints au niveau des genoux et des coudes pour conserver leurs volumes.*

Digital Tutors - Advanced Character Rigging in Maya 2013
Techniques avancées pour le rig cartoon

Digital Tutors - Rigging Quadrupeds in Maya
Pour moi le meilleurs tutoriel sur les quadrupèdes

Creating a Character Rig , maya learning channel youtube
en creative commons. 38 vidéos de 10 à 25 min. Beaucoup de nouvelles techniques très bien expliquées notion avancée du rig et une partie sur du script. Ce qui est rare.

Character Facial Rigging with Judd Simantov,
un des meilleurs tutoriel sur le rig facial