

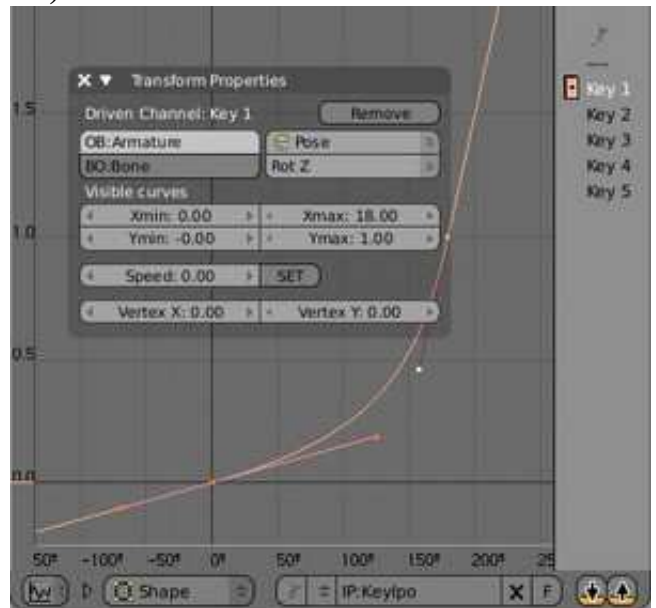
## XIII – ANIMATION AVANCEE (Advanced Animation)

### 13.1. Les Drivers de Courbes IPOs (Ipo Drivers)

Un **Driver** de courbe **IPO** est comme une courbe **IPO**, mais à la différence d'une Courbe de **Bézier**, il permet de connecter une propriété d'autres Objets comme entrée pour le Canal. Par exemple, les Drivers de courbes **IPO** peuvent être utilisés pour qu'une Clé **Shape** soit « dirigé » par la rotation d'un Bone, ou pour que les couleurs **RVB** d'un Matériau soient « dirigées » par la position **XYZ** d'un Objet.

L'édition des Drivers se fait dans l'éditeur **Ipo Curve**. Ici, vous pouvez noter que les Canaux (fenêtre de droite) ont maintenant un indicateur "**canal actif**".

Pour ajouter un Driver, vous devez utiliser le panneau **Transform Properties** (N). Là, vous pouvez ajouter ou enlever un Driver du Canal actif, et utiliser les boutons pour choisir le type de relation que vous voulez établir.



#### Les Objets Drivers

Notez que n'importe quelle courbe **IPO** peut être « dirigée », mais que seuls une transformation d'Objet ou une transformation de la Pose d'une Bone peuvent devenir des Drivers.

Pour le moment, seule la transformation locale est prise en compte. Pour des Objets, cela veut dire les valeurs **Location/Rotation/Scale** sans transformation du Parent (comme montré dans le panneau **Transform Properties** des Objets).

Pour les Poses de Bones, cela veut dire que seule la transformation de Pose (modifications faites à la position au repos) est une information pour Driver (également visible dans le panneau **Transform Property** en mode **Pose**).

#### Mapping des Drivers

Quand une courbe **IPO** est "dirigée", le mapping est de un-par-un par défaut. Il n'est seulement restreint que par les limites des Canaux déjà intégrées, comme la valeur **R** pour un matériau qui va de 0.0 à 1.0.

Notez également que lorsque vous mappez des rotations, les valeurs de la rotation actuelle dans des courbes **IPO** sont diminuées d'un facteur de 10.0 (180 degrés possèdent une valeur de 18.0 dans le système de courbes **IPO**). C'est une convention ancienne dans **Blender**... Elle est un peu cachée, parce que la règle (verticale comme horizontale) affiche correctement les valeurs virtuelles. Seul le panneau **Transform Properties** affiche la valeur réelle.

Quand vous dessinez une courbe **IPO** dans un canal **Driven**, la courbe définira le mapping entre la sortie du **Driver** (horizontal) et l'entrée du **Driven** (vertical).

Une option sympathique à utiliser est **Insert one-to-one curve** (I, ou par menu déroulant). Cela zoomera également l'affichage pour remplir exactement la fenêtre, en permettant une édition facile. Si vous utilisez cette option avec des degrés, elle adaptera une rotation de 180 degrés à un intervalle de 1.0 unité.

#### Mise à jour en direct

Depuis que les **Drivers** sont intégrés dans le système de courbes **IPO**, ils seront toujours actualisés chaque fois qu'une courbe **IPO** est évaluée. Ceci arrive au moins sur les modifications de cellos.

Pour un feedback interactif, les mises à jour pendant la transformation des Objets sont ajoutées dans les cas suivants :

- Courbes **IPO** d'Objet « dirigé », par d'autres Objets ou d'autres Poses de Bones.
- Courbes **IPO** de Clé **Shape** « dirigé », par d'autres Objets ou d'autres Poses de Bones.

Vous pouvez aussi insérer des **Drivers** sur des courbes **IPO Action**, mais celles-ci ne sont évaluées que lors d'une modification de cellos.

#### Problèmes connus

- Renommer des Bones ne renomme pas les Drivers associés.
- L'ajout d'une courbe **IPO** à partir d'un fichier n'ajoute pas les Objets Drivers qui lui sont associés.

### 13.2. Les Clés Shape Dirigées (Driven Shape Key)

Une Clé **Shape Dirigé** correspond au contrôle de la Valeur Clé d'une Clé **Shape Relative** par le mouvement d'un autre Objet.

En effet, si vous voulez animer des Clés Shape **Relative** dans **Blender**, ceci ne peut malheureusement pas être fait avec des **Actions** (voyez le paragraphe sur l'Editeur **Action**).

A la place, vous allez utiliser une "construction auxiliaire", les Drivers **IPO**. Avec un Driver **IPO**, vous contrôlez la valeur d'une courbe **IPO** en déplaçant un autre Objet. Ainsi, par exemple, la couleur d'un Objet peut être dépendante de la rotation d'un autre Objet. De façon équivalente, la Valeur de Clé (**Key Value** – ou puissance de l'influence) d'une **Shape** peut être dépendante de la rotation (ou du mouvement) d'un autre Objet. Ceci est particulièrement utile, car vous pouvez, par exemple, rendre dépendant la forme d'un muscle de l'importance de la rotation du Bone sous-jacent.

Les **Clés Shape Dirigées** sont une fonction typique de **Blender** : elles sont construites à partir de plusieurs éléments, qui ne dévoilent leur puissance que selon une combinaison correcte. Dans ce cas ci, il s'agit d'une animation de vertices par une Clé Shape **Relative**, et la combinaison d'Actions pour des Armatures dans l'éditeur **NLA**. Vous n'avez pas besoin nécessairement d'utiliser des Armatures (et/ou des Bones) comme "drivers" pour les Clés **Shape**, mais les possibilités d'animation avec des Poses sont meilleures. Certes, les fonctions complètes sont déjà décrites dans d'autres paragraphes, mais il n'est tout simplement pas très évident de combiner ces éléments.

En utilisant des **Clés Shape Dirigées**, vous allez travailler sur des niveaux d'abstraction très élevés.

- La chose la plus simple est commencer à travailler au niveau des Clés **Shape**, comme **Left Eyebrow Down** (Sourcil Gauche Baissé), **Right Eye Open** (Oeil Droit Ouvert), **Sneer Left** (Ricanement Gauche).
- Le niveau suivant consiste à diriger une (ou plusieurs) **Shape(s)** avec le même Bone d'Armature, comme **Eyelid + Eyebrow** (Paupière + Sourcil), ou "Bomber le muscle si le bras a pivoté".
- Le niveau suivant consiste à combiner les mouvements de plusieurs Bones ensemble dans une **Action**, comme **Happy** (Heureux), **Sad** (Triste), **Wink** (Cligne de l'œil), **Frown** (Froncement des sourcils), ...
- Le dernier niveau sera de combiner toutes ces **Actions** dans l'éditeur **NLA**.

Vous n'avez pas obligatoirement besoin d'utiliser ces choses, mais elles peuvent vous rendre la vie (et votre travail) plus facile.

### **Exemple**

Vous allez commencer avec un exemple simple, qui présente tous les éléments essentiels.

Le Maillage de base est présenté dans l'image ci-contre. Il est vu depuis la Vue **Top** (**NUMPAD7**).

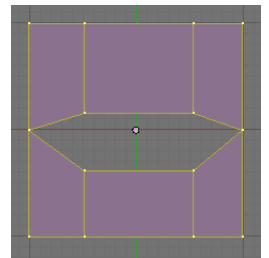
Maintenant, vous allez insérer cinq **Shapes**.

Sélectionnez le maillage en mode **Object**.

Tout d'abord, créez la Shape **Basis** en pressant **I** (puis l'option **Mesh**), ou en cliquant sur le bouton .

**Add Shape Key** dans le panneau **Shapes** du Contexte **Edit**.

Assurez-vous d'avoir créé/édité le Maillage selon vos désirs **avant** de créer la Shape **Basis**. Si vous éditez le Maillage après avoir créé des Clés **Shape**, des résultats imprévisibles peuvent survenir.

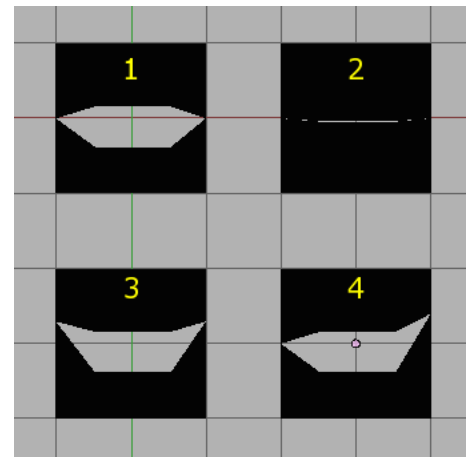


Vous créez les quatre Shapes suivantes directement au début, de sorte qu'elles aient toutes la Shape **Basis** pour origine. Répétez quatre fois le processus **I** + option **Mesh**.

Vous pouvez basculer entre les **Shapes** en sélectionnant la **Shape** respective dans le panneau **Shapes** du Contexte **Edit** (**F9**). Maintenant, vous allez commencer à modifier les Shapes.

Les **Shapes** sont (comme sur l'image à droite) :

1. **Basis** : Position Basis, bouche à demi ouverte.
2. **Close** : Bouche fermée.
3. **Smile** : Coins de la bouche pointant vers le haut.
4. **LeftUp** : Coin gauche pointe seule vers le haut.
5. **RightUp** : Coin droit pointe seule vers le haut.



Pour diriger les courbes **IPO**, vous allez utiliser les Bones d'une Armature. Un Bone pour ouvrir et fermer la bouche et un autre pour lever ou baisser les coins de celle-ci. Comme vous pouvez utiliser les trois directions dans l'espace séparément

comme entrée de drivers, vous pouvez déplacer ensemble les deux coins de la bouche par déplacement dans la direction verticale. Par déplacement dans la direction horizontale, vous déplacez le côté droit (respectivement gauche) uniquement, etc. .

### **Réglage de l'Armature (Setup of the Armature)**

Le nombre de Bones à utiliser et la façon dont vous allez contrôler les **Shapes**, dépendent du nombre de commandes que vous voulez obtenir et de leur facilité d'utilisation.

Dans cet exemple, l'Armature est insérée dans la Vue **Front ZX** (**NUMPAD1**). Vous pouvez l'insérer dans son propre Calque et utiliser une **Vue 3D** séparée pour déplacer les Bones. Vous devriez aligner les Bones le long des axes Globaux, c'est plus facile que de limiter leur mouvement.

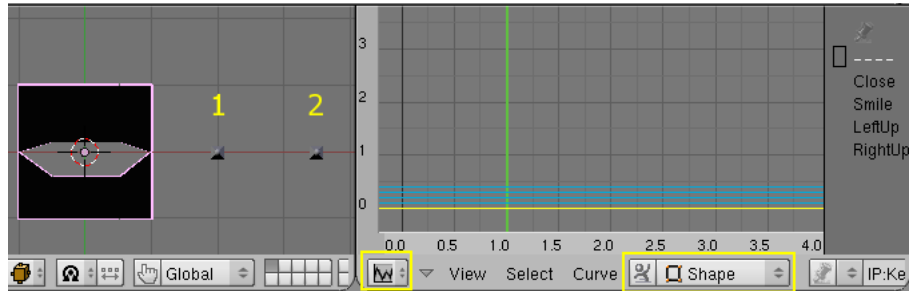
Nommez le Bone gauche **Close** et le Bone droit **Smile**. Nommez l'Objet Armature **Driver** (**F9** > Panneau **Link and Materials**, champ **OB:**) (le nom **Driver** est arbitraire et n'a aucun lien avec la courbe IPO **Driver**).

Sélectionnez le Maillage en mode **Object**.

Séparez en deux la **Vue 3D** et faites apparaître la fenêtre de l'éditeur **Ipo Curve**. Sélectionnez l'option **Shape** dans le menu **Ipo Type** (Image ci-contre).

A gauche, vous voyez les Bones :

- 1 – **Close**.
- 2 – **Smile**.

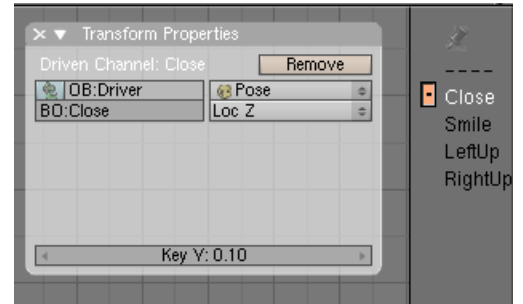


### Connecter une Clé Shape et un Driver (Connecting Shape key and Driver)

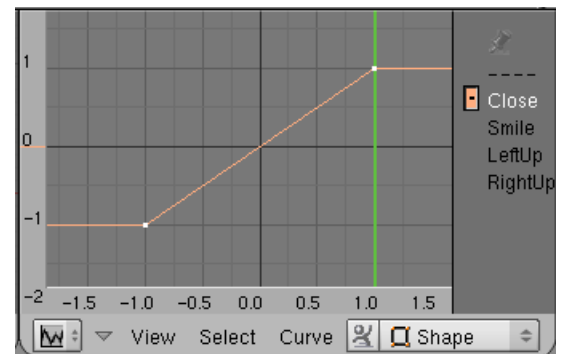
Pour connecter la Clé **Shape** avec l'Objet contrôleur - le **Driver** – vous sélectionnez la Clé **Shape** (**LMB** dans la liste sur le côté droit de l'éditeur **Ipo Curve**), et pressez **N** pour faire apparaître le panneau **Transform Properties**.

Cliquez sur le bouton **Add Driver**. Insérez le nom de l'Armature dans le champ **OB:** et sélectionnez **Pose** dans le menu déroulant immédiatement à droite. Insérez le nom du Bone contrôleur dans le champ **BO:** (ici, **Close**). Vous utilisez **LocZ** pour contrôler la Pose (Image ci-contre).

Si vous avez un doute sur la coordonnée correcte, sélectionnez le Bone dans la **Vue 3D** et utilisez son panneau **Transform Properties** pour visualiser ses coordonnées.

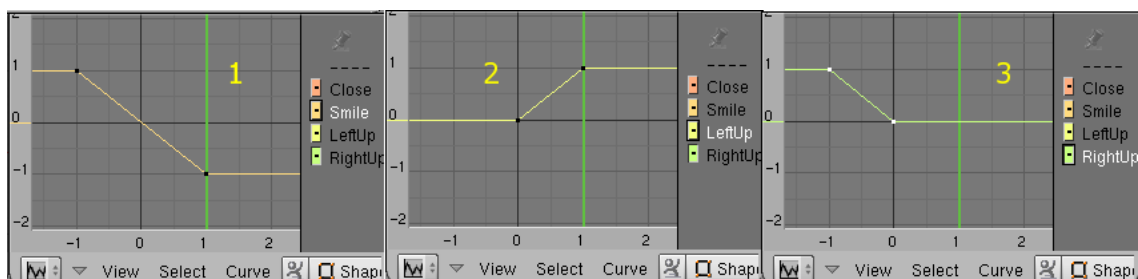


Pour insérer une courbe **IPO**, pressez **I** dans l'éditeur **Ipo Curve** et confirmez l'option **Default one-to-one mapping**. Une transition en courbe linéaire est insérée entre (0,0) et (1,1). Le symbole pour la courbe **IPO** (orange) en face du nom de la **Shape** (**Close**, dans l'image ci-contre) reçoit un point, afin d'indiquer qu'un **Driver** est présent pour la courbe. La courbe **IPO** mappe le mouvement du Bone vers la Valeur de Clé (Key value) de la **Shape** : c'est à dire que si vous déplacez le Bone d'une unité **Blender** dans la direction **Z**, la Valeur de Clé de la **Shape** passe de 0 à 1. Si vous le déplacez dans la direction **Z** négative, la **Shape** est inversée (c'est à dire que les vertices se déplacent dans la direction opposée). Si la courbe **IPO** court horizontalement, la **Shape** ne change pas si vous déplacez le Bone. Vous allez utiliser ceci pour spécifier les limites d'un mouvement significatif du Bone.



Réglez le mapping à (-1, -1), (1, 1), et le mode **Extend** à **Constant** (Option **Constant** du sous-menu **Extend mode** du menu **Curve**). Avec l'option **Linear** de **T**, rétablissez la courbe linéaire, si le type de courbe s'est modifié quand vous avez modifié le mode **Extend**.

Répétez ces étapes pour la Shape **Smile**, mais vous devez inverser la courbe **IPO** (**S + X**) de (-1,1) vers (1,-1), afin que le mouvement du Bone corresponde au mouvement de la bouche. Afin de faire se lever individuellement les coins de la bouche, la Clé **Shape** est connectée avec la coordonnée **X** du Bone. Si le Bone se déplace selon la direction **X** positive, le coin droit se lève; s'il se déplace dans la direction **X** négative, le coin gauche se lève (image ci-dessous).



Les trois courbes IPO pour les Shapes **Smile** (1), **LeftUp** (2) et **RightUp** (3).

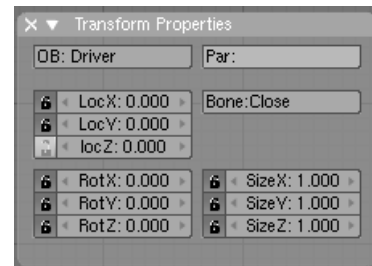
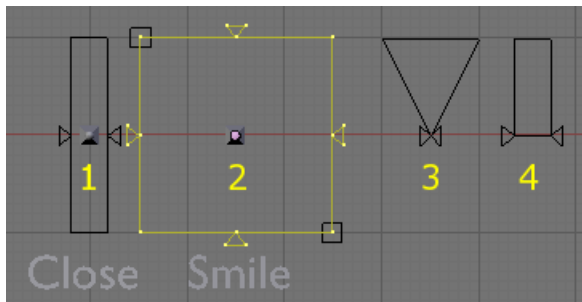
Si maintenant, vous déplacez les Bones en mode **Pose**, les **Shapes** se modifient en accord. Si vous avez besoin de plus de **Shapes**, ou si vous voulez une gestion encore plus souple, vous devrez définir plus de **Drivers**.

### Commandes Personnalisées (Custom Controls)

Des "Commandes Personnalisées" – c'est à dire essentiellement des contrôleurs manuels créés à la main – facilitent la gestion des **Clés Shape Dirigées**. Il n'existe pas de tels Objets **Blender** (pour l'instant), donc vous allez utiliser des Objets Maillés pour visualiser le but et les limitations des Objets **Drivers**. Le mouvement des **Drivers** est restreint par le réglage des options **Lock**

dans leur panneau **Transform Properties** et – si vous préférez – par des Contraintes **Floor**. Ainsi, les Objets **Drivers** se comportent presque comme de véritables boutons de contrôle.

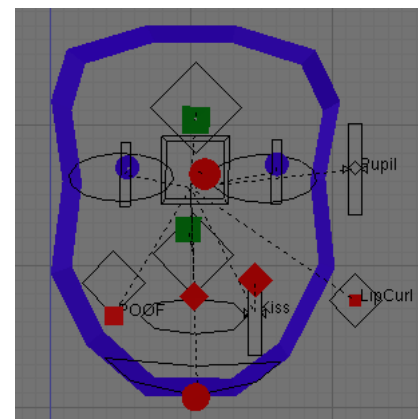
L'image ci-dessous à gauche montre quelques exemples de Commandes Personnalisées (quatre Maillages pour afficher les limites des réglages).



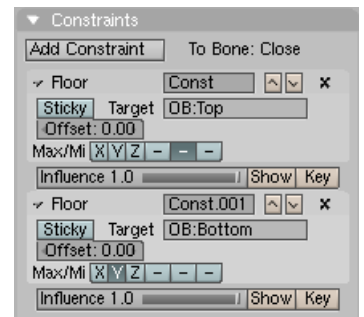
En restreignant les directions de mouvement dans le panneau **Transform Properties**, vous autorisez le mouvement du Bone vers le haut et vers le bas, mais pas vers les côtés (l'image ci-dessus à droite présente le panneau **Transform Properties** pour le Bone **Close**, qui correspond au Maillage N°1 de l'image de gauche).

Vous pouvez opter pour une visualisation différente, si vous construisez un "visage" stylisé, et y insérez les **Drivers** à leur place respective comme sur l'image ci-contre (d'après une idée de Zeyne : **Zeyne 2.4 rig with facial controller** en <http://www.elysiun.com/forum/viewtopic.php?t=57478>).

C'est une façon plus intuitive de visualiser les mouvements des **Drivers**.

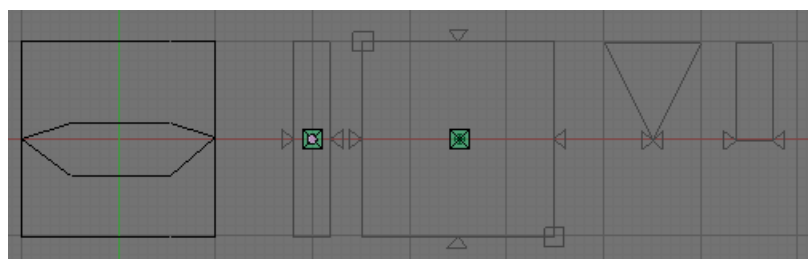


Dans ce deuxième exemple, deux autres Objets aux coins sont utilisés comme Contraintes **Floor**. Ils empêchent le mouvement des Bones au-delà des limites indiquées. Toutefois, il faut quatre Contraintes pour restreindre le mouvement d'un Driver à une zone de contrôle, et seulement deux Contraintes pour des Drivers utilisant des curseurs "Haut-Bas" (image ci-contre).



Après avoir créé les commandes, vous pouvez les déplacer vers une Scène **Global**, afin de ne pas les éditer par inadvertance. Ajoutez une nouvelle Scène vide (la double-flèche à côté du champ **SCE:**, **Add New** puis l'option **Empty**). Nommez cette nouvelle Scène **Global**. Sélectionnez tous les Objets qui appartiennent à la commande, mais pas l'Armature, que vous avez toujours besoin de pouvoir modifier. Liez ces Objets à la Scène **Global** en utilisant **CTRL L > To scene... > Global**.

Ensuite, retirez la connexion entre les Objets avec **U > Object**. Maintenant, effacez les Objets dans la Scène courante. Passez dans le Contexte **Scene (F10)** et cliquez sur le petit bouton avec la double flèche et le discret Tooltip **Scene to link as a Set** dans le panneau **Output**. Sélectionnez **Global**. Maintenant, les commandes sont visibles, mais vous ne pouvez pas les sélectionner ou les éditer (image ci-dessous).



### Dans l'Editeur **NLA** (NLA)

Maintenant, les mouvements individuels du "Visage" peuvent être regroupés vers des Actions, et de nouveau, celles-ci peuvent être assemblées en groupes complexes dans l'éditeur **NLA**. Une **Action** serait une expression particulière du visage, par exemple, joyeux, triste, surpris, mauvais, etc. Si vous avez fourni toutes les Actions nécessaires, vous ne travaillerez qu'avec ces Actions.

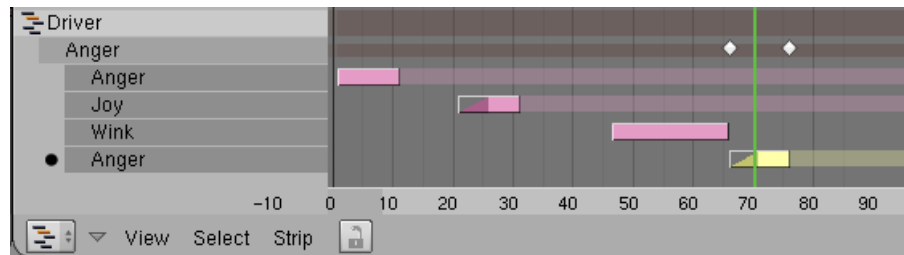
Bien sûr, vous n'êtes pas forcé de travailler dans l'éditeur **NLA**; à la place, vous pouvez créer une **Action** longue, dans laquelle vous insérez toutes les courbes **IPO**.

### Définir des Actions (Define Actions)

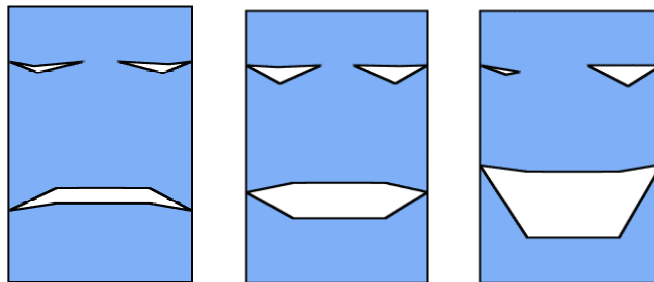
Comme décrit dans le paragraphe sur l'éditeur **NLA**, les différentes **Actions** pour le visage doivent être créées.

Le "visage" d'exemple est étendu avec deux "yeux". Trois Actions ont été créées dans la fenêtre de l'éditeur Action : **Wink** (Cligner de l'œil), **Joy** (Joie) et **Anger** (Colère) (Image ci-contre). En positionnant les Bones de l'Armature en mode **Pose**, les expressions sont créées, tous les Bones nécessaires sont sélectionnés et une courbe **IPO** est insérée.

Déplacez-vous de 10 cellos vers l'avant, et sauvegardez exactement la même Pose. Le timing est fait dans la fenêtre de l'éditeur **NLA** (image ci-dessous).



Maintenant, disposez les expressions dans l'éditeur **NLA** et définissez le timing. Le mélange des différentes expressions est particulièrement utile (avec les options **Blendin/Blendout**). Si vous utilisez l'option **Hold** pour les Strips, vous devrez définir une Action **Basis**, afin que vous puissiez faire un mélange avec la Shape **Basis**. Voyez les exemples ci-dessous :



### 13.3. Les Drivers Python (Python Drivers)

Les Drivers Python (ou **Pydrivers**) permettent d'utiliser des expressions **Python** en ligne comme entrée pour un Canal, au lieu d'utiliser une propriété d'un autre Objet, comme les Drivers **IPO** le font normalement. Une expression en programmation est toute combinaison de symboles qui peut être évaluée en une valeur définie.

Quand vous ajoutez un Driver, comme expliqué dans le paragraphe précédent, cliquez sur la petite icône **Python** dans le panneau **Transform Properties** et une boîte d'entrée de texte apparaîtra. Tapez-y votre expression **.py**.

Ces Drivers apportent des possibilités intéressantes : vous pouvez utiliser des fonctions mathématiques et de la programmation plus générale pour diriger des animations d'Objets. L'évaluation d'une expression **Pydriver** est équivalente à ce que fait la fonction interne **eval()** : la seule restriction est que l'expression **Pydriver** elle-même doit renvoyer un nombre réel (tout type de nombre qui peut être mis en virgule flottante) quand elle est évaluée, et pas, par exemple, une chaîne ou autre chose.

Un exemple simple utiliserait des sinus et des cosinus pour faire de des Objets oscillent autour d'un point donné, par exemple. Ou, utiliser des valeurs aléatoires à chaque cellos d'une animation pour modifier un attribut quelconque du Matériau, comme **Diffuse, RGB, Alpha**, etc. .

#### Instructions

##### Expressions Valides (Valid Expressions)

Voici quelques exemples des expressions valides que vous pouvez entrer dans la boîte de texte :

- Toute valeur réelle : **1.0**.
- Des expressions avec des nombres et des opérateurs : **4.5 + 8.9 \* 7.0 - (2 / 3.0)**.
- Des expressions également avec des variables : **math.pi \* 2 + 5.0**.
- Des données disponibles : **Blender.Get("curframe")** , c'est le cellos en cours de l'animation.
- Un peu de mathématiques : **math.sin(Blender.Get("curframe"))** , c'est le sinus du numéro du cellos courant.

##### Ressources Intégrées et Alias (Builtin resources and aliases)

Les **Pydrivers** utilisent leur propre dictionnaire global qui est mis en cache et n'est recréé que quand une expression quelconque échoue. Dans ce dictionnaire, ont été pré-importés quelques modules de sorte qu'ils puissent être utilisés dans des expressions de **Pydrivers**.

**Note** : Pour économiser de la frappe et conserver des expressions plus petites, des alias ont été ajoutés pour chaque module : il est fait référence à **Blender** par "**Blender**" ou simplement par "**b**".

Ci-dessous, chaque module est suivi de ses alias disponibles :

- `all from builtin` (le module intégré par défaut).
- `Blender: blender, b.`
- `Blender.Noise: noise, n.`
- `math: math, m.`

**Exemple d'expression** : `m.cos(m.pi * b.Get("curframe") / n.random())` .

Des alias ont aussi été ajouté pour quelques données couramment utilisées :

- `ob(name)` est équivalent à `Blender.Object.Get(name)`.
- `me(name)` est équivalent à `Blender.Mesh.Get(name)`.
- `ma(name)` est équivalent à `Blender.Material.Get(name)`.

**Exemple d'expression** : `ob("Cube").LocX + ob("Lamp").RotZ` .

#### Le Texte **pydrivers.py** dans Blender (The **pydrivers.py** Blender text)

En dessous des modules précédents, s'il existe un texte **Blender** appelé **pydrivers.py** (`pydrivers: pydrivers, p`) chargé dans l'éditeur **Text**, il est aussi importé. Ce dernier permet aux utilisateurs de créer leurs propres fonctions et d'ajouter leurs propres variables sans la restriction de l'expression **.py** limitée à une ligne. Par exemple, si votre texte **pydrivers.py** ressemble à ceci :

```
myvar = 10.0

def myfunction(arg):
    # faites quelque chose ici
    return float_val
```

Vous pouvez accéder à la fois à **myvar** et à **myfunction** dans n'importe quelle expression :

**Exemple d'expression** : `p.myvar * p.myfunction(2) - 1` .

**Note** : Si vous faites des mises à jour au texte **pydrivers.py**, allez dans la fenêtre de l'éditeur **Ipo Curve** et cliquez dans la boîte d'entrée de texte pour **Pydriver** (dans le panneau **Transform Properties**), puis cliquez en dehors (ou pressez **ENTER**), pour forcer le rechargement du module **pydrivers.py** et pour mettre à jour tous les **Pydrivers** en une fois.