Abstract

Fairies appear a lot in art and motion pictures as they are a sight of wonder and mystery. Also, in more recent animated films, visual artists have been trying to create images that have a painterly feel to them whilst still working in the realm of 3D. This project shows a study of how a fairy is to be portrayed and move in a 3D animation as well as creates a 3D animation that has a similar look to the painterly effect that has been described.

Contents

Introduction

From a young age the author has enjoyed drawing creatures of mythology. A natural extension of that was to create one for the major project, it was decided that a fairy would be a suitable subject since it would add the extra challenge of being anthropomorphic. It had the added benefit of potentially being very enjoyable embodying it with a personality that would be portrayed visually.

After the decision had been made two main factors had to be addressed:

- How would the fairy be represented?
- How would the fairy move?

Aims & Objectives

Firstly a study is to de done of a range of artists' impressions of fairies, as well as how they are represented in various films and television programs. The author also wishes to study wildlife and nature, and plans to use elements of living creatures to make the fairy appear more realistic.

Wildlife and nature will be appraised, as elements taken from living creatures will be used to make the fairy appear more realistic. In particular the movement of humans, animals and insects will be studied to develop realism

Knowledge will be gained on the ways various film makers have interpreted fairy movement.

Using the knowledge gained a believable fairy will be created which will be modelled and animated in a simple environment.

The final piece must then be rendered to have the overall look and feel of a painting.

Look of the Fairy

Within this section beliefs in fairy mythology and folklore will be examined to get a better understanding of how the fairy is perceived. How the fairy is portrayed in artwork throughout the ages will then be studied. The natural world will then be investigated to see if it can be incorporated into the fantastical being. Then a decision will be made on the appearance of them and how it will be created using computer software.

The Fairy of Myth and Legend

Over the years many books of fairy artwork have been acquired and have been used for inspiration. A favourite fairy volume is "the World of the Faery: An inspirational collection for faery lovers". This hardback contains entries by many different artists in a variety of media-this is why it is one of the most treasured books.

Within the book there is a forward by the renowned artist Alan Lee (whose work includes illustrations for "The Lord of the Rings" trilogy). He states fairies offer:

"visions of a vast realm that lies between the stark polarities of heaven and hell…and between our mundane world and what we are led to expect of it."

What is noted from this citation is that fairies are used as a form of escapism: people want to believe that there is more to life than the ordinary. They are used as a way to explain the fantastic or even as an excuse to blame them for something that goes awry.

From this information it is obvious that fairies are seen in myth as creatures of mystery and mischief, thus these aspects are to be added into the design of the character and movement.

How Other Artists Portray the World of the Fairy

Within the collection of art books that have been assembled over the years there has been ample inspiration for fairy design. A favourite artist is Devon based artist **Hazel Brown** (whose artwork is depicted below and right). The effect that the artist has created with the paints is very pleasing. The fairies look so delicate and the texture of the canvas can be seen through the paint. These are two effects that will be emulated in this project

Figures 1 & 2: *"Wood Sprite"* (top right), *"Wood Elves"* (bottom middle) both by Hazel Brown
Both taken from *"The World of Faery"*

Another artist's work that is admired is **Brian Froud**. A lot of his artwork has inspired films such *as "the Labyrinth" and "The Dark Crystal"*. There is a book of his art *called "Lady Cottington's Pressed Fairy Book"*. This depicts paintings of what are supposed to be fairies that have been squashed and captured in the book. The style with which he has painted is very appealing in particular the watercolour style the artist has used would ideally be incorporated in this work.

Figures 3 & 4: Images of fairies taken from Brian Froud's *"Lady Cottington's Pressed Fairy Book"*

Above are two of Brian Froud's paintings from *"Lady Cottington's Pressed Fairy Book"*.

On studying two of the fairy artists it has been decided to emulate some of the artist's techniques: Brian Froud's watercolour effect as well as Hazel Brown's technique where the canvas texture beneath the painting can be seen.

The Natural World: Aesthetics in Insects that can be applied to the Fantastic

Insects are very interesting creatures to behold. A study has been made of them since fairies are described as small creatures; it would be quite interesting to give them aspects similar to those of various invertebrates.

It is appealing that insects have what is known as an exoskeleton (this is where the skeleton of the creature is on the outside, whereas human skeletons are on the inside). It is like a shell encasing the animal's body and vital organs. A similar construction will be incorporated into the fairy. Maybe not make it as shell-like as an insects exoskeleton but give it some human characteristics: make her skeleton look obviously like ours (ie. give her ribs etc.)

The fairy would need small antennae like an insect, as well as a pair of wings. The decision on the type of wings was made after researching various pictures of insects in Michael Chinery's "*Insect of Britain & Western Europe*" Dragonflies were noted as being interesting. Is is appealing that they sport four long wings that are transparent and have the veins showing on them. They look so delicate and graceful yet really etherial. So the decision was made to give the fairy wings that are very long, slightly transparent and have the veins apparent on them.

Figure 5: "*True Dragonflies"*, an Accurate diagram taken From *"Insects of Britain & Western Europe"*

Designing and Modelling

After studying the various ways in which the fairy is illustrated in different media various designs were portrayed. Here are the original concepts that have been created.

Figures 6 & 7: Paintings created of the fairy design, both completed in acrylics

To create a character that ties in with the research, the fairy is:

1. humanoid, like all other portrayals of fairies that are available
2. small and delicate
3. almost insect-like, her ribs and hip bones poke out in an almost exoskeleton manner.
4. long, fragile wings of a dragonfly
5. made her unusual and slightly different to the other images available

Of all the completed paintings, the most personally appealing is the water colour (see image on the right). This would be an effect that should be incorporated in the project.

Figure 8: A watercolour image completed of the fairy.

After designing the fairy it had to be modelled. NURBS was used for the initial modelling before moving on into polygons. The final model is displayed on the left:

Figure 9: The model of the fairy.

Movement of a Fairy

In this section of the report a discussion on how fairies have been pictured in motion will be carried out. The research will include various films and television programs that have incorporated fairies. The motion of animals, insects and human beings will be compared to that of the mythical creatures of the small and big screen. After doing this investigation a decision will be made on how the fairy will move then there will be a discussion on this will be executed using various pieces of software.

The Portrayal of Fairies on the Big and Small Screen

Considering some favourite films that have fairies in them (*"Pan's Labyrinth"*, *"Photographing Fairies"* and *"Fairytale: A True Story"*), and a critical appraisal of how fairies are portrayed in films will be done now.

The movement of the fairy in each of these films is very similar: All the fairies flap their wings extremely quickly and the wings don't seem to rotate or bend in anyway. Thus an attempt to use a similar flight style will be made as it looks like a natural way for a fairy's wings to flap.

However, an unpleasant effect in the film is noticeable: As the fairies are in flight, their bodies don't really move that naturally, they almost look a bit awkward and don't seem to know what to do. The reason behind this will be examined and by using a different approach will be rectified so that the fairy does look natural when flying, and not so awkward.

Figures 10, 11 & 12:
(TOP) A shot taken from *"Photographing Fairies"* (1997),
(MIDDLE) *"Pan's Labyrinth"* (2006)
(BOTTOM) *"Fairytale: A True Story"* (1997)

Looking at Natural Motion

Human Motion

The study of human movement is to be included in this investigation as the fairy does look very humanoid and would undoubtedly benefit from this.

Muybridge's "*The Human Figure in Motion*" is a useful starting point. This edition contains various photographs (shot in series in very quick concession) of men, women and children moving in various common fashions such as running, jumping etc. There are almost 200 various motions in the volume but, as my fairy looks more feminine, the decision was taken to mainly look at the photographs of the women.

When studying the photographs of the woman's walk cycle (depicted below) it can be seen that as she pushes herself forward with one leg, she is also pushed upward slightly as well. She then brings this leg up and bends it beneath her body before stretching it out in front of her to place it on the ground whilst doing the previous motion with the opposite leg. Also, one can notice that, as one leg is moved forward, so is the opposing arm. This is one of the mechanisms that the human body uses to propel itself forward.

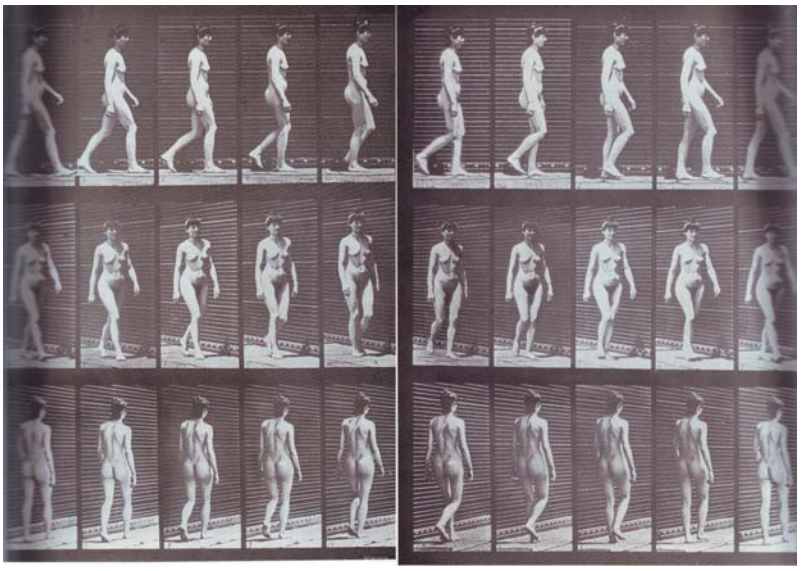

Figure 13: Woman Walking Taken from Muybridge's book "*The Human Figure In Motion*"

Another two motions that are of interest are of a woman jumping and a woman climbing a ladder. The woman jumping (pictured in the next illustration) is quite interesting as it shows how she bends down to gain momentum before stretching out again as she takes off.

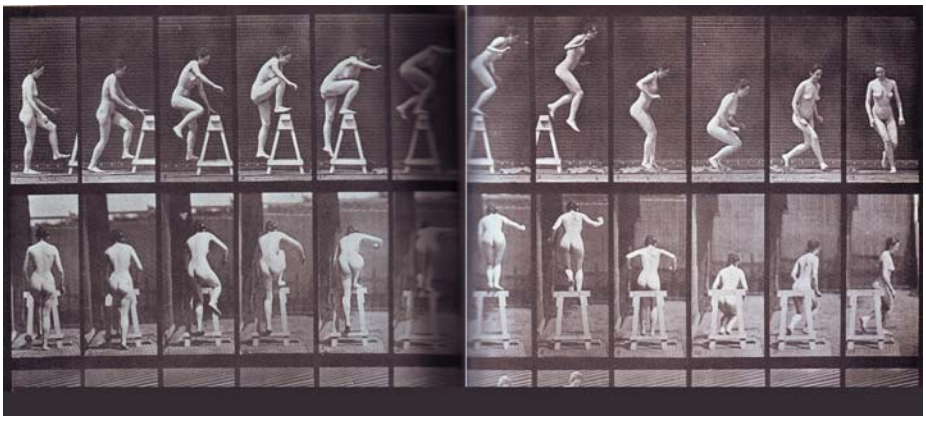

Figure 14: Woman Jumping Taken from Muybridge's book *"The Human Figure in Motion"*

The final series of images of the woman climbing the ladder is also interesting. As in the walk cycle, she uses her legs to push herself forward, bringing them up closer to her body, making the slightly bent, before pushing off forward and making them straight again. The woman seems to be using her arms to pull herself forward as well. Also, from looking at the back view of the woman, we can see that, as she brings her leg up, her hips rotate up on the side of that the leg that is moved up.

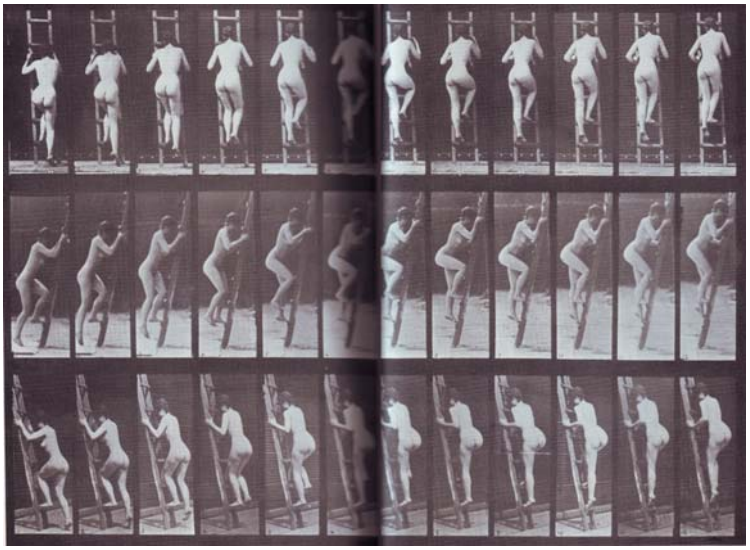

Figure 15: Woman Climbing Taken from Muybridge's book *"The Human Figure in Motion"*

Above are images of the woman climbing a ladder.

So in this brief analysis of the motion of a human, it has been decided to use some of these movements in the animation. The motion of the woman climbing is appealing, so it will be incorporated (it would be imagined that the fairy would do a lot of climbing). The fairy should jump into a flying motion, so the information on the woman jumping will come in useful for this. However, as this is an unreal character, the motions will be exaggerated to make them more obvious. For instance when the fairy will bend, it will be a much bigger and more noticeable bend than the actual human bend.

Birds In Flight

After looking at human behaviour and motions it was decided to consider how the fairy would fly. The obvious flying creature is a bird so research was carried out to discover the pattern of bird flight.

Muybridge's *"Animals In Motion"* was an obvious starting point. This volume contains images that are typical of Muybridge's work: photographs of animals taken in various motions. These photographs are taken by taking pictures consecutively at up to $1/2000^{th}$ of a second; thus creating a very accurate representation of that animal. The artist has been able to photograph the animals against a ruled background (so one can easily measure the vertical and horizontal motion of the animal) as well as photographing various creatures at three different angles.

"Animals In Motion" contains a very large selection of animals, 123 in total. However only the motion of birds will be considered, as they are the only animals of relevance to this project. The birds that Muybridge has photographed are:

1. pigeon
2. cockatoo
3. hawk
4. vulture
5. eagle
6. adjutant
7. ostrich

But, because the adjutant aren't birds of flight, they will be ignored.

On looking at the variety of birds, their flight motion seems to be very similar. However, it is noticeable that the larger birds (such as the eagle and vulture) seemed to have less "phases" in their flaps than the smaller birds (like the hawk), as illustrated in the table below:

Figure 16: Table depicting number of phases in bird flaps. Information gathered from Muybridge's *"Animals In Motion"*

Bird Type	**No. of Flaps**	**No. of Phases**
Pigeon	1	7
Cockatoo	1	12
Hawk	2	8
Vulture	1	15
Eagle	1	8

This theory was then justified whilst looking on www.nhm.org As it said that some larger birds have developed wings that allow then to glide for long periods of time (therefore have less phases per wing flap) whereas smaller birds, the example they give is the hummingbird, can barely glide. The site states that this is because *"flapping takes tremendous energy"*, so it can be understood why the larger birds wouldn't want to spend a lot of time flapping their wings and resort to using their wings to glide. An example from the site of the various types of motion compared to the size of the birds can be seen opposite:

Thought was then given to thinking of a general way to describe the motion of the wings, because when the images are examined, it seems like the wings are pushing the birds forward. An accurate account of a bird's motion is described as follows (taken from www.nhm.org):

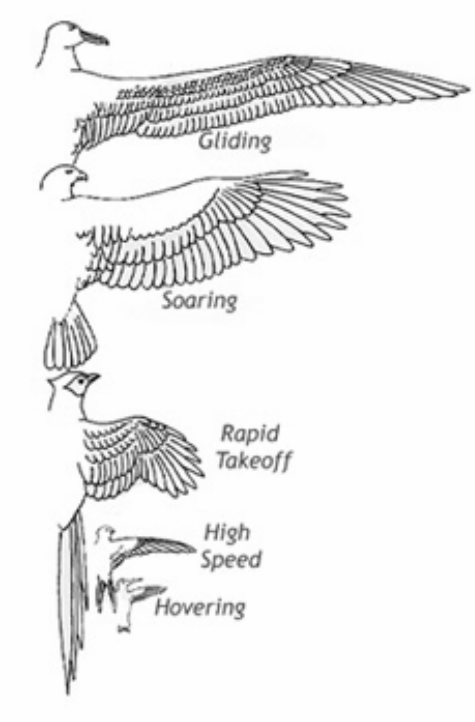

Figure 16: Various Birds with an Outline of their flight Pattern, taken from nhm.org

"On the upstroke of the wing beat, the feathers at the end of many bird wings twist sideways to let the air slip through with little resistance. The down stroke is the power stroke. Rather than rowing a bird through the air, the wings act like the propeller on an aeroplane to pull the bird forward."

After considering bird flight, aspects would be incorporated into the animation. The idea that birds use their wings to glide as well as flap is appealing; as the fairy has very large wings and it might be a bit unrealistic to have her constantly flapping them really quickly. However, it was decided to omit twisting the end of the fairy's wings as it may look a bit strange as her wings aren't actually feathers but are more like insect wings.

Movement of Dragonflies

On revisiting the model it is apparent that the wings really resemble those of a dragonfly, so it was decided to have a closer look to see how they fly. The book *"Insects of Britain & Western Europe"* shows that the two pairs of wings of a dragonfly are unlinked an, in fact, move independent of each other. Also, in an article on www.sciencedaily.com, *"Solving a Dragonfly Flight Mystery*", Cornell University physicists conducted studies of dragonfly motion to find that their separate wing pairs can either move in the same motion in time with each other or in opposing motion (ie. the front pair will move up, whilst the bottom pair will move down and vice versa). Also, what's quite interesting is that dragonflies will use their wings in sync with each other to maximise their lift during take-off. The physicist then discovered that the

reason for having the wings flap in opposing directions whilst in the air was because it takes less energy for the insect to do that.

Figure 17: How a dragonfly's wings flap. Taken from www.harunyaha.com

The image above illustrates the opposing wing motion of a dragonfly. It is taken from the site www.harunyahya.com/designinnature01.php

With the knowledge of how a dragonfly moves, it was decided to use its various methods of flight on the fairy. It is appealing in the way it flaps its wings in opposite directions as well as the fact that it uses its wings in harmony to thrust itself forward into the air.

Getting the Fairy Moving

Rigging

This part of the report shows how the fairy is to be rigged and, after that, animated. The Setup Machine is to be used to rig the body of the character as it was felt that it would be too time consuming to focus on rigging the fairy as well as studying its motion and figuring out a way to animate the wings. However, the facial rig will be created manually. The Setup Machine is a very useful device as it creates a bipedal character out of cylinders. The cylinders are then placed around the character's geometry. When that is done finished doing that, “rig” is pressed and The Setup Machine creates a working 2D rig for your character. Also, it paints the weights pretty accurately too, so overall it's a very handy device and saves a lot of time.

Now that the fairy's body is rigged, a facial rig has to be created.
The book “*Stop Staring: Facial Modelling and Animation done right”* proved invaluable, through this it was found out what where the best facial expressions to make for the blend shapes for the character's facial rig:

1. half blink
2. full blink
3. mouth open
4. mouth closed
5. eyebrows down
6. eyebrows up (sad)
7. eyebrows up (shock)
8. left eyebrow raised, right eyebrow lowered
9. right eyebrow raised, left eyebrow lowered
10. frown
11. mouth small

From the above it is apparent that not all the necessary blend shapes for a character to be used in a feature animation have been created; just the ones that will be needed in the short piece of animation. To control the facial rig, a Mel script that came with Jason Osipa's book will be used.

A simple rig has been added for the hair of the character that is manipulated by various controls and has a slight jiggle deformation on it, so that it bounces slightly when the hair is animated.

The final touch that has been added to the rig is one for the wings. This was done by adding a joint for each wing and creating a control to control each one separately. These controls were then parent constrained to the upper body of the rig. It was decided to rig the wings in this fashion because they were required to flap simply and rigidly, in a similar motion to a dragonfly. It was felt that if the wings flapped too dynamically, with a bit of a wobble, that it wouldn't look right when the wings were flapping really quickly.

Mel Scripting

As it was desirable to have the fairy flap her wings quickly and multiple times it was thought it might be a lot less troublesome to think of a way that this could be done automatically other than manually. After discussing with various friends and lecturers about a suitable method it was found that a Mel script would be best.

This was entering into a realm that had not been considered previously. On looking into Mel scripting it was appreciated that it would be useful to control the fairy's wings individually and key them using some sort of interface. Graphical User Interfaces (GUI) had been covered in year one so more information was sought on this subject.

Making the GUI

Graphical User Interfaces are devices that allow the user to interact with a computer using various graphical indicators such as sliders and buttons. The concept for this GUI would incorporate the following:

1. A title
2. Four buttons that allow the user to select the four wings
3. A slider to allow the user to select how many beats per second the wings would be flapping at
4. A button that allows the user to key the wings
5. A button that allows the user to quit

On top of these the GUI would have to be as user friendly as possible. This is because of a dislike of working with devices that are overly complicated and confusing - it puts people off using them.

So, now that the decision for the appearance of the GUI had been made investigation into how to write a script was taken to achieve the stated end. It was advised that Rob Bateman's website be visited as it has some great introductions to creating various buttons and sliders in Mel. After consulting it basic layout for the interface was created.

As stated, creating a window title was the first thing on the list. This was done by creating a variable called $win and making that equal a window called “WING CONTROL”. Below is an excerpt from the script to illustrate this:

```
$win = `window "WING CONTROL"`;
```

Figure 18: Mel command for creating a window titled “WING CONTROL”

As well as this, the line showWindow had to be used at the end of the window function. This will display the GUI on the computer screen. Below is an example of a window with only the title:

Figure 19: GUI of the Mel command demonstrated in **Figure 18**

To create the buttons in for the various wings in Mel it was decided to use the create check box command. This is written thus:

```
// create checkerboxes
        $c = `checkBox -label "top left wing"
                       -onCommand "on_func_top_left"
                       -offCommand "off_func_top_left" tlwk`;
```

Figure 20: Mel command for creating check boxes

Below is a table with an explanation for each command within this script along with an image of the interface:

Name	**Explanation**
$c	used to define the type of variable we are dealing with
-label	label the check box with name you give it in the “” brackets
-onCommand	the command that makes the box switch on
-offCommand	the command that makes box switch off
tlwk	the name of the wing key

Figures 21 & 22: An explanation of the Mel script in **Figure 20**, along with the interface it creates

Next a slider to control the amount of beats per second that were on the selected wings was required. This was done by using a float slider group. This is a type of slider that handles floating numbers as well as integers. A float slider group was chosen instead of just a float slider on its own because it has a text label as well as a float field input box whereas a float slider doesn't have the text box. Below is the part of the script which demonstrates the slider along with a table of the explanations for various commands:

```
floatSliderGrp -label "wing beats" -field true
        -cc "beats"
        -fieldMinValue 0 -fieldMaxValue 10
        -minValue 0 -maxValue 10 -value 0
        beatSlider;
```

Name	**Explanation**
-cc	links the slider to the procedure “beats”, which is declared earlier in the script
-field true	means slider will have an editable field that represents the value of the slider
-fieldMinValue	minimum value of the slider field input box
-fieldMaxValue	maximum value of the slider field input box
-minValue	minimum value for the slider and field
-maxValue	maximum value for the slider and field
-value	the value of the group

Figures 23, 24 & 25: Mel script for creating a slider for the wings, along with an explanation and image of the interface.

Above is a screen shot of the GUI with the added slider

Now a button to key the window was needed. To add a bit of variety and make the GUI look more interesting it was decided to make a normal button as opposed to the check box that had previously been used to make the wing selectors. Below is the script which was used to create it along with short explanation of the different commands used as well as the interface with the addition of the key button:

```
// create button to key wings
        button -label "click me to key" -command "keyFunc";
```

Figures 26 & 27: Script for creating a button along with the GUI so far

The command button, as its name would suggest, creates a button.
The command -command is the same as the -cc command which was used earlier in the slider script. It links the button to the procedure “keyFunc”, which is defined in a different part of the script.

The final command was to delete the GUI by pushing a button labelled “quit”. This was done by writing the following:

```
// create a command to delete the window
        $delete = ("deleteUI " + $win);


button -label "quit" -command $delete;
```

Figure 28: How to delete the window in Mel

$delete is a command that deletes the window, which has been named $win earlier in the program. It utilised the “deleteUI” command which deletes the user interface.

Below is what the GUI looks like now:

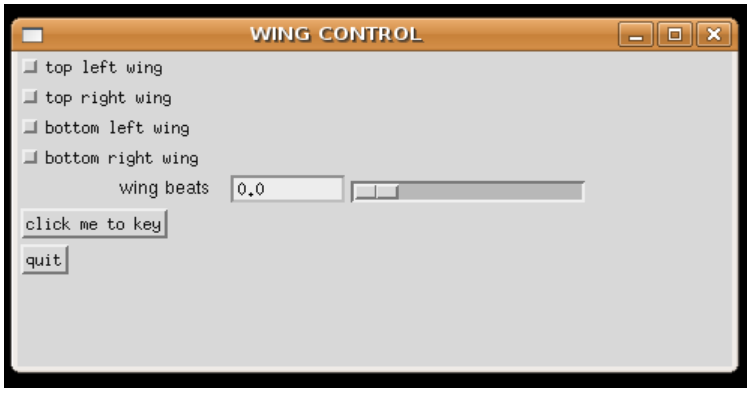

Figure 29: How the Interface looks after adding the various buttons

Making the GUI actually work

Now that the GUI had been created visually, the main problem lay in linking the various interactive devices of the interface with the fairy. Several procedures were written to accomplish this. As previously mentioned, some of the interface controls are linked to procedures using the -cc or -command commands. These procedures are contained at the top of the Mel script, away from the function that creates the window as this is normal practice when writing code. It makes everything look a lot neater.

Firstly, a procedure was needed that would, when check the various wings' check boxes were selected, would select the appropriate wings. For each wing a function was created to tell if:

1. the box has been checked
2. the box has been unchecked

Within the function to check the box is included a line of code that selects the wing. Also, included is a line that works the opposite way in the function that unchecks the box. Below is the code for each of the functions controlling the bottom right wing and explanations:

```
function to be called when the checkerbox gets checked.
   proc on_func_bottom_right() {
           select -tgl wing_bottom_control_right;
           print("bottom right checkbox on!\n");
   }

function to be called when the checkerbox gets unchecked.
   proc off_func_bottom_right() {
           select -d wing_bottom_control_right;
           print("bottom right checkbox off!\n");
   }
```

Name	**Explanation**
select -tgl	selects whatever this command is in front of
print	prints whatever is in the () brackets to the script editor's terminal
select -d	un-selects whatever this command is in front of

Figures 30 & 31: Function for how to actually get the check boxes in the interface to work with the model, as well as an explanation of the various commands in the script.

Next a command was required that corresponded to the button that allows the user to key the wings. To do this “if” statements were used which ask whether the various check boxes have been selected. If they have been selected, the wing will be keyed.

The various wing keys have been abbreviated as follows:

1. tlwk = top left wing key
2. trwk = top right wing key
3. blwk = bottom left wing key
4. brwk = bottom right wing key

Here is the piece of the script that includes this along with a table explaining the various code:

```
function to be called when the button gets clicked-KEYS ANIMATION.
   proc keyFunc() {
            if(`checkBox -q -v tlwk`){
                    setKeyframe -at beatsPerSecond wing_top_control_left;
            }

            if(`checkBox -q -v trwk`){
                    setKeyframe -at beatsPerSecond wing_top_control_right;
            }

            if(`checkBox -q -v blwk`){
                    setKeyframe -at beatsPerSecond wing_bottom_control_left;
            }

            if(`checkBox -q -v brwk`){
                    setKeyframe -at beatsPerSecond wing_bottom_control_right;
            }

   }
```

Name	**Explanation**
-q	puts the command into query mode. In this instance it is querying whether the wing is selected
-v	the same as -value, is the value of the group
setKeyframe	creates keyframes for specified object/objects
-at	lists the attributes to select
beatsPerSecond	the amount of beats per second that the wing/wings will flap

Figures 32 & 33: Functions that key the animation along with an explanation for each command

The final, and most difficult, of the procedures was the one for the slider. After some research it was decided that various driven keys be made for the wings in different positions for the flap.
Needed would be:

1. one for the start of the flap (0.0)
2. one for the middle of the flap (0.5)
3. one for the end on the flap (which, would be the same as the start flap (1.0)

Also, it had to be remembered that the fairy wings would have to flap in a similar motion to that of a dragonfly – the top wings should have an opposing motion to the bottom wings. With this in mind, various driven keys were made with an attribute within each wing control, labelled flappy, be the driver of each key and control the flapping of the wings. Then another attribute had to be created for each control called beatsPerSecond. This would be directly linked to the slider and control how many wing flaps (from flappy) there would be in each second. To do this included had to be the following expression in the attribute flappy (obviously the name of the control would be changed to whichever wing control was being affected):

Figure 34: The expression for Flappy

```
$t = wing_top_control_left.beatsPerSecond;

if($t == 0.0){
        wing_top_control_left.flappy  = 0.0;
}
else{
        $v = (25 / $t);
        $result = (frame % $v) / $v;

        wing_top_control_left.flappy = $result;
}
```

The expression states that if the number of beats per second for the specified control equals zero, that the value of flappy would remain at 0.0, thus keeping the wing control in the starting position of the flap. However, if this is not the case the expression then says that the amount of flap within the wings equals the frame modulus of twenty-five (number of frames per second) divided by the number of beats per second, then this number is to be divided by twenty-five divided by the number of beats per second.

After creating these attributes and driven keys the script was revisited and a procedure to get the slider to function was created:

```
// a function to be called when slider "wing beats" is moved-HOW MANY BEATS PER FRAME.
        proc beats(){
                $val = `floatSliderGrp -q -v beatSlider`;
                setAttr wing_bottom_control_left.beatsPerSecond $val;
                setAttr wing_top_control_left.beatsPerSecond $val;
                setAttr wing_top_control_right.beatsPerSecond $val;
                setAttr wing_bottom_control_right.beatsPerSecond $val;
        }
```

Figure 35: Function that uses the attributes in Maya to move the interface's slider

Below is an explanation of the various lines of code in the script:

Name	Explanation
$val	is a variable that finds the value of the slider
setAttr	sets the value of a dependency node attribute. In this case it is set the values for the various controls beatsPerSecond attribute

Figure 36: An explanation of the commands in the previous Mel script

Now that these were created the GUI was tested and it worked well apart from when the value of the beat slider was changed and the wings were keyed. It was found that the wings would jolt back to their starting position. This was not a desired effect so solutions to this problem were sought. Controls were created for the wings that can blend between stationary and the value of the beatsPerSecond slider. So the already existing sliders were duplicated and any constraints that were attaching them to the rig deleted. Then a locator called "nullBlend" was created, as had been explained it looked more fluid if you have something constrained between the blend control and the original control. The original controls were then parent constrained to the locator, before constraining that to the blend control. Now within each blend control was added a new attribute named "weight". This was going to control how much the blend control is being controlled by the original control. Then the parent constraints on each blend control were made to be driven by this new "weight" attribute by using set driven keys. When this was done, a locator was created that would drive all the weight attributes for the blend controls. This attribute was called "overall_weight". After creating these final driven keys this slider was created:

```
attrFieldSliderGrp -label "wings amount"
        -attribute "blend_control.overall_weight"
        -min 0 -max 1
        blendSlider;
```

Figure 37: Script for the blend control on the interface

This slider has a minimum value of 0 and maximum value of 1. When the slider is at 0 the wings aren't being affected the beatsPerSecond slider but when the slider is set to 1 they are.

Final Layout of the Interface

After getting all the devices on the interface to work, the layout had to be finalised to make it look more organized and aesthetically pleasing.

After much though, it was decided it would look nicest to have the check boxes for the various wings in a column on the left and have the button to key them underneath in the same column, then have a second column that contained the two sliders and finally have a third column that had the “quit” button. To have these various columns a row layout would need to be created within those different embedded column layouts.

```
rowLayout -numberOfColumns 3 -columnWidth3 150 400 100;

columnLayout -columnAttach "both" 12 -rowSpacing 8 -columnWidth 150;
```

Figure 38: How the script is written to describe the sliders layout

Above is an example of how one of the columns is embedded within the row layout. After adding the embedded column, the command setParent ..; must be added to show that that is the end of that particular column. Below is an explanation of each command:

Name	**Explanation**
rowLayout	states that it is a row layout
-numberOfColumns	number of embedded columns, in this case three
-columnWidth3	the width of the three columns, in this case 150, 400 and 100
columnLayout	states that this is a column layout
-columnAttach	Sets the attachment and offsets for the children of the layout, in this case “both” and twelve
-rowSpacing	how much space between rows, in this case eight
-columnWidth	the width of this particular column, in this case 150

Figure 39: Explanation of Mel commands used in **Figure 38**

Below is a screen shot of the final appearance of the GUI:

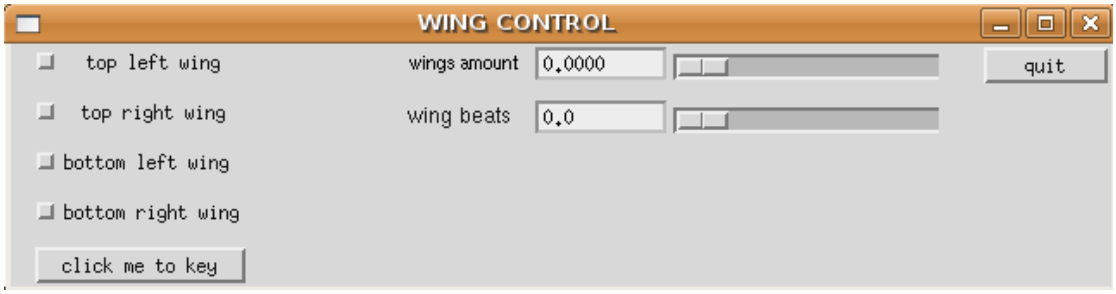

Figure 40: The final layout of the interface

Unfortunately, after testing the GUI in the scene, it was found that the flapping controls cannot be keyed any more. This because they are being affected by the driven keys that are in place on them. So in the animation there will be two separate models that will be animated: one that is not controlled by the driven keys of the GUI, and can be manually animated; and another model that is controlled by the interface. The latter of the two models described above will be used when the fairy is in flight or only flapping her wings but not rotating them in any other way, whereas the other model will be used to animate the motions where she isn't flying but just naturally moving about.

Final Touches-Setting the Scene

Now that it is decided how the fairy to move as well as look, an investigation is to be carried out on how to make an aesthetically pleasing animation. To do this various render techniques will be examined in addition to learning a bit more about compositing and even using other media that are outside of the computer.

Modelling a Simple Scene

The fairy on its own looked quite isolated and stark, so a simple scene was created for her. After looking again through the images created by various artists, it was decided to place her in a scene similar to Hazel Brown's *"Wood Sprite"* (see fig.). The pose is very pleasing, she looks relaxed and natural but still contains a bit of mystery. The foliage is attractive that is pictured in the environment. So an emulation is to be created in Maya.

Here is a picture of the environment for the fairy to live in. As you can tell, it is quite similar to Hazel Brown's painting (See **Figure 1)**, but have added is a twist to it as you may see in the image of it to the right of the page.
Like the artist, the background is to be white, almost it has been painted onto a plain canvas. However the shapes of the toadstool in this scene is much more angular than hers. This was done because it was to be a similar style to the fairy, who is also very angular.

Figure 41: The model of my scene

Animation

Consideration was now given of the various ways to portray the fairy moving in the scene. Therefore it was decided that a variety of different shots that show:

1. the fairy in a relaxed pose, with little movement
2. the fairy being more active, her full body in motion
3. the fairy flying, so that the Mel script may be used

After consideration it was decided to consult the photographs from the Muybridge book and utilize those motions. So when the fairy is in motion she is to be shown jumping and climbing at some point. So for the fairy's full body motion scene she could climb up the toadstool before jumping into flight at the top of it.

Toon Shaders in Maya & Getting that "Painterly" Affect

Toon Shading

After completing the animation it was required to make the image look a lot more aesthetically pleasing. On consideration on how fairy was to look (see fig.) it was apparent that the outline was made up of scribbles which made her very typical of a watercolour painting.
On investigating the various types of rendering that are used in Maya, Toon Shaders are the most capable of emulating the painting effect. On researching Toon Shaders that create paint-like effects there had been software called "Tomcat Toon Shader", which had been a downloadable plug-in for Maya but it was no longer available. In the NCCA archive, a report had been written on cartoon shaders by Sam Berry (who now works at Blue Zoo). On contacting him, he said that Tomcat was actually in Maya now. In the Maya help files it was found that in the render layer you can get various Toon Shader examples under the Toon tab. While looking through these examples a suitable one was found that had the scribble outline that was required for the project.

Looking further into Toon Shading, there were various fill shaders that are already present in Maya that could be used. The scene was rendered in four different passess (note the number on the bullet-points correspond to the number to the bottom left of each image):

1. first, the colour layer, which is the uv mapped texture for the fairy and environment
2. next, the layer with the scribble outline, as well as the fairy and environment shaded with a Light Angle Shader
3. A Light Angle Shader
4. finally a Shaded Brightness Three Tone Shader

1. **2.** **3.** **4.**

Figure 42: The various render passes in Maya

Playing Around With Shake

Now that the various images had been created, they were imported into Shake and composited together to create an effect similar to paint. Below are various

experiments that were done in Shake. The results achieved are impressive. The one that chosen to be used is the large image on the left as it looks the most like the effect present in the original concept painting:

Figure 43: Various composites in Shake. The large one on the left was the one chosen to use.

Although the fairy was finally rendered in Maya, it was not quite what was required: the background could be more interesting, so it was decided to paint a picture and composite it in. A simple image was chosen that matched the toadstool and the grass that had been modelled.

Figure 44: My fairy composite from **Figure 43** composited over a painted background

Above is the result obtained from Shake. This is the desired image it looks and feels more polished and like a painting. It looked so startling that various colleagues didn't realise that the fairy was rendered from Maya!

Filming Effects With a Tank

To give a final special finishing touch it was decided to have a layer over the top of the animation that looks like faint animated paint. Along with this it was wished to

include something that looks like animated paint where the fairy makes quick or sudden movements (like her wings flapping or when she is jumping into the air). It was considered that paint and ink be filmed in a large water tank. This is a technique that has been used in quite a few famous movies, such as, ”Independence Day” (1996), as an alternative to rendering out shots using particles. According to wikipedia.org here is how they created most of the effects:

“*The shoot utilized on-set, in-camera special effects more often than computer-generated effects in order to save money and to get more authentic pyrotechnic results.”*

Luckily the masters Digital Effects course at the university has such a tank along with a huge amount of stock footage that is available for use. The studio was used to film various objects in the water and see how they moved:

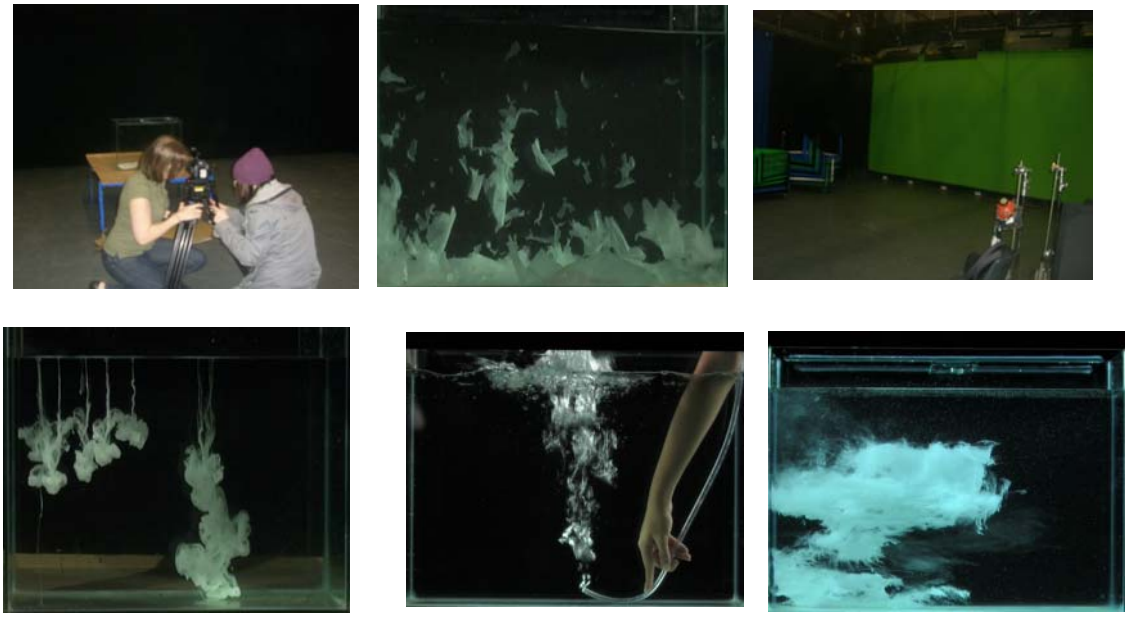

Figure 45: (TOP LEFT, CLOCKWISE) colleagues Vicky & Francesca set up the camera, footage filmed of tissue paper in water, the studio, footage taken of ink in water, footage the MA Digital Effects course took of bubbles, MA footage of paint in water.

With a large library of images to use in the piece they were imported into shake and experimented with to see what suits the animation best. The results obtained compositing ink and paint over the scene are excellent. The thick liquid moves slowly through the water, so it works nicely just composited over the scene, to give an effect that there is paint on the scene that is slowly moving. The bubble effect is really interesting and it would suit being composited over the fairy when she is jumping or doing any other fast movements.

Below are various composites of these images using Shake:

Figure 46: Final composite of fairy with paint composited over the top

Figure 47: Final composite of fairy jumping from toadstool.
Paint bubbles have been added to give the impression of motion.

Conclusion

How Does It All Look?

The final composited shots look very impressive. It really looks like a painting that has come to life. The filmed effects are pleasing. It is an interesting exercise to combine live action special effects with 3D animation and a 2D painted background and produce something that looks right, even after having used so many different types of media.

Although not the most aesthetically pleasing piece of the project, the GUI looks well. it is very user friendly, and, as it is not overly complicated, it wouldn't put anybody off using it. The flapping motion that it has created also ties in very nicely with the project. It is so delicate and fluttery that it really looks like it could be taken from a real creature.

Does Everything Work?

The results of the Mel script are personally impressive; it works very well at giving the impression of a fairy's wings flapping. It is a good feature that it allows blending between flapping and not flapping.

Can Anything be Improved?

There are some areas that could be improved upon:

The Mel script. As mentioned before, multiple rigs had to be used in the animation as the wings that work with the Mel script cannot be keyed in different positions (eg folded down). This is slightly cumbersome as it means that careful track has to be kept of what scenes need to have the various models in them. To remedy this, it would have been useful to have been able to make the script have a control that would allow the user to alternate between wings that could be moved and keyed manually, and the wings that are driven by the flapping sliders. Unfortunately this was unable to be done because of time constraints but it will be addressed later.

Another part of the project that could be expanded upon and improved is particular part of the animation: where the fairy is ascending the toadstool could do with some extra finishing touches. However, the final compositing of the paint, bubbles and ink over this animation disguises it very well. But, given more time a more polished version would be created.

What Has Been Learned?

A variety of concepts were learnt during this project:

A deep admiration of the various artists' works' being studied

When looking at how animals are seen to move in the natural world, the knowledge of how to clinically analyse the motion of various creatures and to interpret them into animation.

A more thorough familiarity of Mel scripting. This allows appreciation of how helpful it is in various projects. Studying it has produced a better understanding of Maya on a procedure-based level.

A real knowledge of Toon Shaders and how to create a painterly animation using various render passes in Maya.

A greater understanding of Shake, now that a lot of experiments have been done in it for this for this project .. For example how to use the timeline in Shake, this is a piece of the program that had never been explored before.

How to film various special effects using the water tank. This was a task that was a lot of fun as well as being educational.

In conclusion to all this, this has been a really enjoyable project which will allow the techniques learned to be used in future work.

Bibliography/References

Books:

"The World of Faery: An inspirational collection of art for faery lovers" Presented by David Riché ISBN 1-84340-282-3

"The Art of Amy Brown" Introduced by Charles de Lint ISBN 0-9744612-2-9

"Lady Cottington's Pressed Fairy Book" by Terry Jones & Brain Froud
ISBN 0-8109-5942-9

"Stop Staring: Facial Modelling and Animation done right" by Jason Osipa ISBN 978-0471789208

"Animals in Motion" by Eadweard MuyBridge ISBN 0-486-20203-8

"The Human Figure in Motion" by Eadweard Muybridge
ISBN 0-486-20204-6

"A Biology of Dragonflies" by Philip S. Corbet ISBN-10: 0860960196

"Insects of Britain & Western Europe" by Michael Chinery
ISBN 0-00-219137-7

Websites:

www.wikipedia.org – for general information on a variety of topics

http://www.nhm.org/birds/guide/pg018.html – for information on bird flight
http://www.sciencedaily.com/releases/2007/09/070924142926.htm – information on dragonfly flight
www.harunyahya.com/designinnature01.php – reference for dragonflies flight

http://www.anzovin.com/TSM2Maya/ - website for The Setup Machine

www.robthebloke.com – former ncca student Rob Bateman site, very useful for Mel
http://accad.osu.edu/~xchang/753/hw3/mel_tutorial.html - also a useful site for Mel
http://www.autodesk.com/us/maya/docs/Maya85/wwhelp/wwhimpl/js/html/wwhelp.htm - Maya help files on Mel

http://ncca.bournemouth.ac.uk/gallery/files/innovations/2005/Berry_Samuel_47/innovationsReport.htm - Former ncca student Sam Berry's innovations project

Appendix: Mel Script

Below is the Mel script that was used to create the interface:

```
//////////////////////// MY MEL SCRIPT ////////////////////////////


//////////////////////functions/////////////////////////////////////

//TOP LEFT
// a function to be called when the checkerbox gets checked.
	proc on_func_top_left() {
		select -tgl wing_top_control_left;
		print("top left checkbox on!\n");
	}
// a function to be called when the checkerbox gets unchecked.
	proc off_func_top_left() {
		select -d wing_top_control_left;
		print("top left checkbox off!\n");
	}

//TOP RIGHT
// a function to be called when the checkerbox gets checked.
	proc on_func_top_right() {
		select -tgl wing_top_control_right;
		print("top right checkbox on!\n");
	}
// a function to be called when the checkerbox gets unchecked.
	proc off_func_top_right() {
		select -d wing_top_control_right;
		print("top right checkbox off!\n");
	}

//BOTTOM_LEFT
// a function to be called when the checkerbox gets checked.
	proc on_func_bottom_left() {
		select -tgl wing_bottom_control_left;
		print("bottom left checkbox on!\n");
	}
// a function to be called when the checkerbox gets unchecked.
	proc off_func_bottom_left() {
		select -d wing_bottom_control_left;
		print("bottom left checkbox off!\n");
	}

//BOTTOM_RIGHT
// a function to be called when the checkerbox gets checked.
	proc on_func_bottom_right() {
		select -tgl wing_bottom_control_right;
```

```
		print("bottom right checkbox on!\n");
	}
// a function to be called when the checkerbox gets unchecked.
	proc off_func_bottom_right() {
		select -d wing_bottom_control_right;
		print("bottom right checkbox off!\n");
	}



// a function to be called when the button gets clicked-KEYS ANIMATION.
	proc keyFunc() {
		if(`checkBox -q -v tlwk`){
			setKeyframe -at beatsPerSecond wing_top_control_left;
		}

		if(`checkBox -q -v trwk`){
			setKeyframe -at beatsPerSecond wing_top_control_right;
		}

		if(`checkBox -q -v blwk`){
			setKeyframe -at beatsPerSecond wing_bottom_control_left;
		}

		if(`checkBox -q -v brwk`){
			setKeyframe -at beatsPerSecond wing_bottom_control_right;
		}

	}



// a function to be called when slider "wing beats" is moved-HOW MANY BEATS
PER FRAME.
	proc beats(){
		$val = `floatSliderGrp -q -v beatSlider`;
		setAttr wing_bottom_control_left.beatsPerSecond $val;
		setAttr wing_top_control_left.beatsPerSecond $val;
		setAttr wing_top_control_right.beatsPerSecond $val;
		setAttr wing_bottom_control_right.beatsPerSecond $val;
	}


//////////////////////////create the window////////////////////////////////

{
	$win = `window "WING CONTROL"`;
	//window -title "WING CONTROLS";
		rowLayout -numberOfColumns 3 -columnWidth3 150 400 100;
```

```
		columnLayout -columnAttach "both" 12 -rowSpacing 8 -columnWidth
150;
			// create checkerboxes
				$c = `checkBox -label "top left wing"
						-onCommand "on_func_top_left"
						-offCommand "off_func_top_left"
tlwk`;
				$c = `checkBox -label "top right wing"
						-onCommand "on_func_top_right"
						-offCommand "off_func_top_right"
trwk`;
				$c = `checkBox -label "bottom left wing"
						-onCommand "on_func_bottom_left"
						-offCommand "off_func_bottom_left"
blwk`;
				$c = `checkBox -label "bottom right wing"
						-onCommand "on_func_bottom_right"
						-offCommand "off_func_bottom_right"
brwk`;
			// create button to key wings
				button -label "click me to key" -command "keyFunc";
		setParent ..;

		columnLayout -columnAttach "both" 12 -rowSpacing 8 -columnWidth
400;

			attrFieldSliderGrp -label "wings amount"
				-attribute "blend_control.overall_weight"
				-min 0 -max 1
				blendSlider;

			floatSliderGrp -label "wing beats" -field true
				-cc "beats"
				-fieldMinValue 0 -fieldMaxValue 10
				-minValue 0 -maxValue 10 -value 0
				beatSlider;

			// create a command to delete the window
				$delete = ("deleteUI " + $win);
		setParent ..;

		columnLayout -columnAttach "both" 12 -rowSpacing 8 -columnWidth
100;

				button -label "quit" -command $delete;
		setParent ..;

	showWindow;
}
```

Special Thanks To:

Innovations Project Tutor:
Claudia Moore

Colleagues Who Helped With Mel & Expressions:
Jon Peel
David Brooks
Andy Cosgrove

Colleagues Who Helped with Filming & Editing Footage:
David Ness
Daniel Jenkins
Francesca Dare
Vicky Morris

Report Proof Readers:
Mum & Dad