Research and Development of a Flexible Tentacle Rig for Exaggerated Deformation

Jiexin Liu

S5514935

Introduction

This practice-based research project focuses on the development of a flexible tentacle rig for exaggerated deformation in Maya 2025. The project originally came from my interest in creating an octopus rig, because octopus tentacles move very differently from standard humanoid limbs. A human arm or leg usually depends on clear pivot points such as an elbow or knee, while a tentacle behaves as one continuous flexible form. It needs to bend, curl, twist, stretch and compress while still remaining controllable for animation.

My initial plan was to create a full octopus rig, including the head and eight tentacles. However, during the research and testing stage, I realised that this scope was too broad for the available time. A complete octopus rig would require global controls, repeated tentacle systems, complex skin weighting and deformation testing. Therefore, I narrowed the project down to one single tentacle. This allowed me to focus more deeply on the main technical challenge: how to create a rig that could produce smooth, exaggerated deformation in an animator-friendly way.

The final outcome is a technical rigging demonstration rather than a finished animation sequence. The aim is to show the development process, compare different rigging methods, and explain why the final solution was chosen. During the research stage, I looked at examples of flexible creature rigging, including Pixar's Hank from *Finding Dory*. Hank was useful as a reference because his tentacles needed to support bending, twisting, curling and expressive movement at the same time (CNET News, 2016). This helped me understand that a tentacle rig is not only a technical skeleton, but also a performance tool for animators.

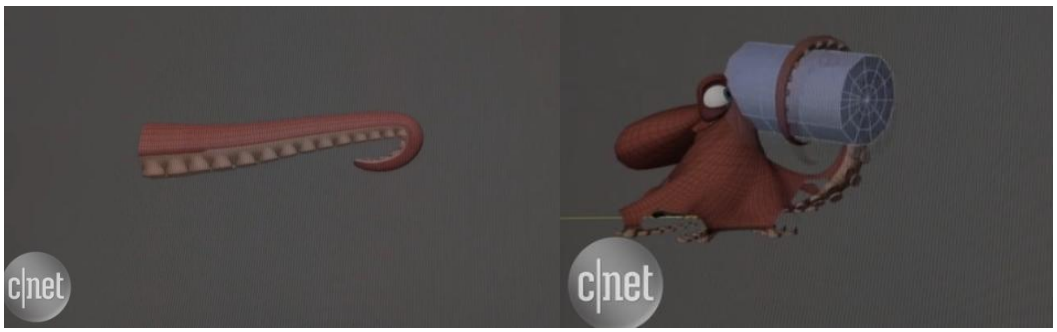


Figure 1, 2: Screenshot from CNET News' rig demonstration video showing Pixar's Hank rig from *Finding Dory* (CNET News, 2016).

Alongside the industry reference, I also reviewed existing rigging workflows and technical demonstrations for flexible characters, including tentacle and snake-style rigs. These references helped me understand possible approaches such as IK spline control, FK curl systems, curve-driven deformation, stretch setups and non-linear deformers.

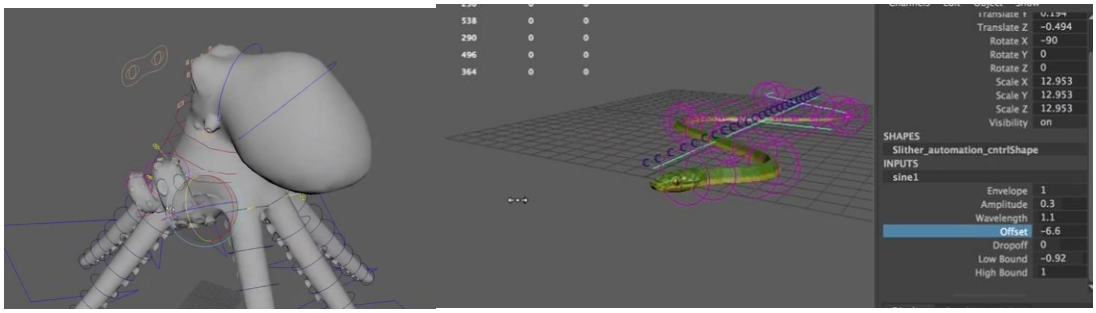


Figure 3, 4: Screenshot from rigging demonstration videos showing tentacle controller and deformation attribute examples.

The base model used for this project was sourced from Sketchfab. I reduced and retopologised it to create a cleaner mesh for deformation testing, so the main focus could remain on the rigging system.

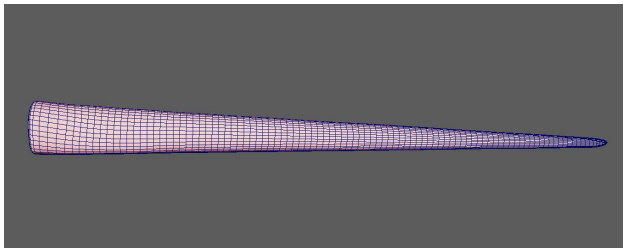


Figure 5: The base model.

Early IK Spline and Cluster Testing

The first stage of development used an IK Spline Handle. I created a joint chain along the tentacle and used an IK Spline curve to drive the movement of the joints. To control this curve, I added clusters to the curve CVs and created NURBS controllers for the clusters. These controllers were then arranged into an FK-style hierarchy, so that when an earlier controller rotated, the following controllers could move with it. The aim was to create a soft bending system that could also support curling.

At first, I used five cluster controls. This was useful for testing broad movement, but it was difficult to create a natural spiral curl. The tentacle looked more like a tube being bent from several points rather than a flexible organic form rolling inward. I then tested a version with more CV points and around ten cluster controls. This gave me more control and made the curve slightly smoother, but it also revealed another problem. If one cluster controlled only one CV, the curve could create sharp corners. If one cluster controlled two CVs, the transition became smoother, but the CVs that were not directly controlled still did not follow naturally.

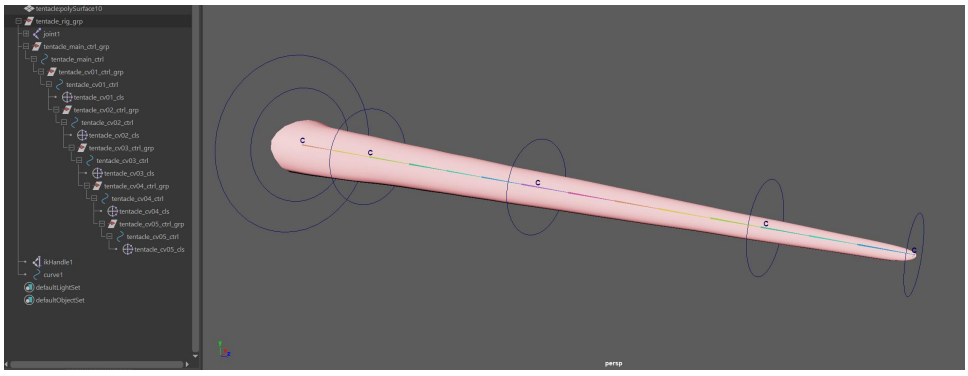


Figure 6: Early IK spline.

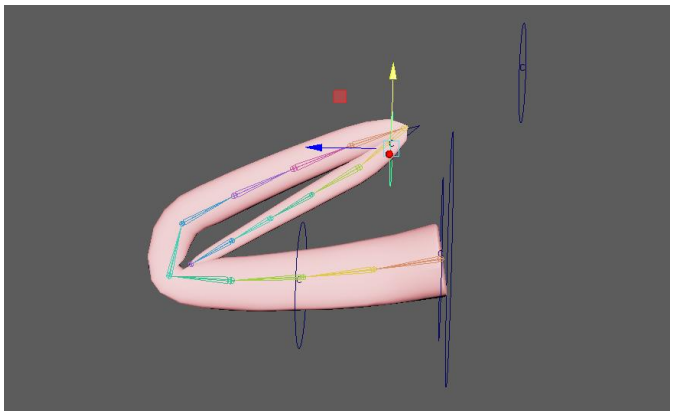


Figure 7: Manually moved CV points.

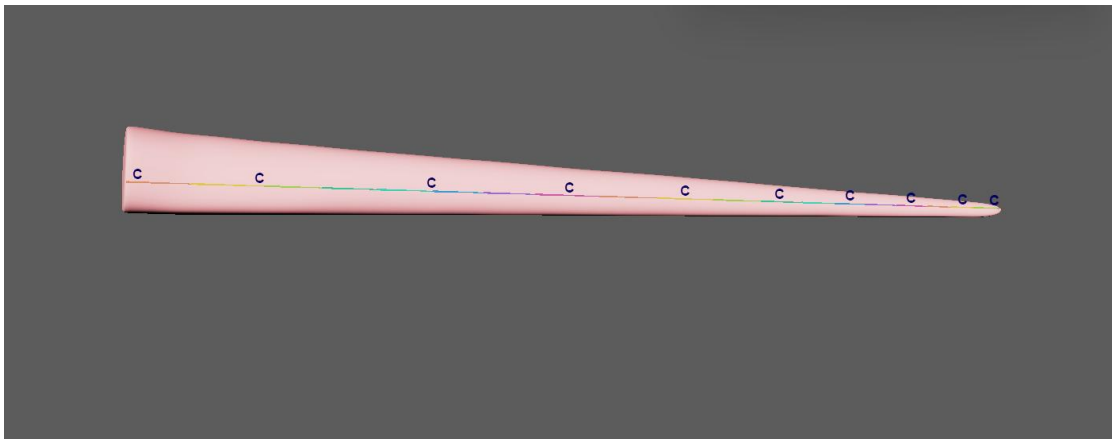


Figure 8: IK spline with more CV points.

This meant that the IK Spline and cluster setup could produce basic flexible bending, but it was not ideal for the exaggerated spiral curl I wanted. The result still felt segmented, especially when the tentacle was pushed into a tighter curled pose.

Set Driven Key and FK Curl Testing

To make the rig easier to animate, I tested a Set Driven Key system. I created a main controller named tentacle_main_ctrl and added custom attributes such as Curl, Roll, Ripple and Stretch. In this test, the main focus was the Curl attribute. I used tentacle_main_ctrl.Curl as the Driver, while the smaller cluster controllers were used as Driven objects. At Curl = 0, the tentacle remained straight. At Curl = 10, I manually rotated the controllers into a curled pose and recorded another key.

This method worked technically because changing the Curl value could automatically drive the controller rotations. However, the final shape was still limited by the IK curve and cluster setup. The curl became easier to control, but it still looked more like an overall bend than a natural spiral. This showed that a custom attribute can improve the animator interface, but it cannot fully solve problems in the underlying deformation system.

I then tested a more direct FK method. By rotating the joints along the tentacle chain, I found that FK could create a much cleaner spiral curl. Since each joint rotates after the previous one, the form can roll into itself more naturally. I also experimented with

scale and inverse scale connections in the Node Editor to reduce unwanted compensation between the joints. This helped the joint chain roll more consistently. However, FK was not practical as the main control method. It required too much manual rotation and became inefficient for animation. Although Set Driven Key could automate some of the curl, it was still difficult to combine this method with stretching, twisting and broader IK-style movement. This stage showed that FK was effective for creating a clean curl pose, but it was not flexible enough as the foundation for the full tentacle rig.

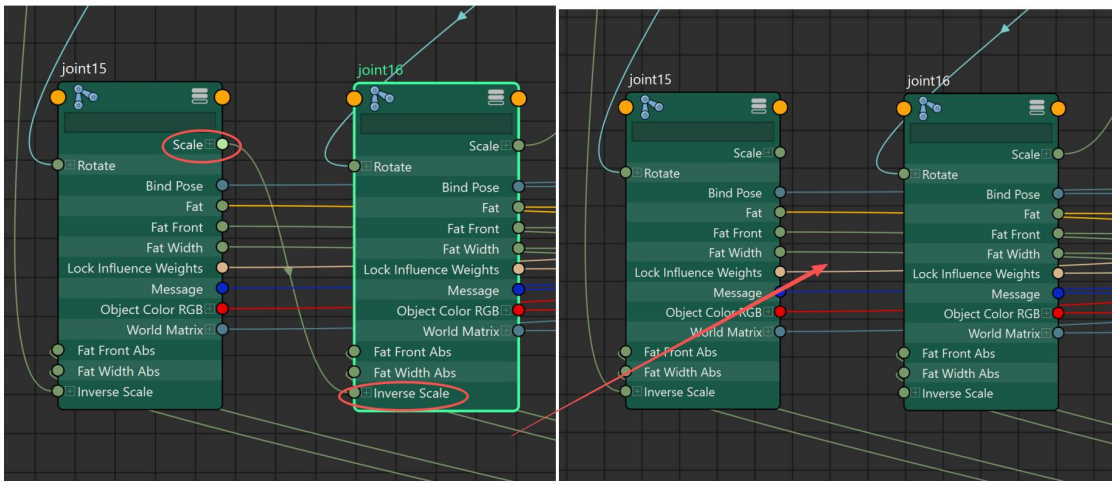


Figure 9: Set scale and inverse scale connections in the Node Editor

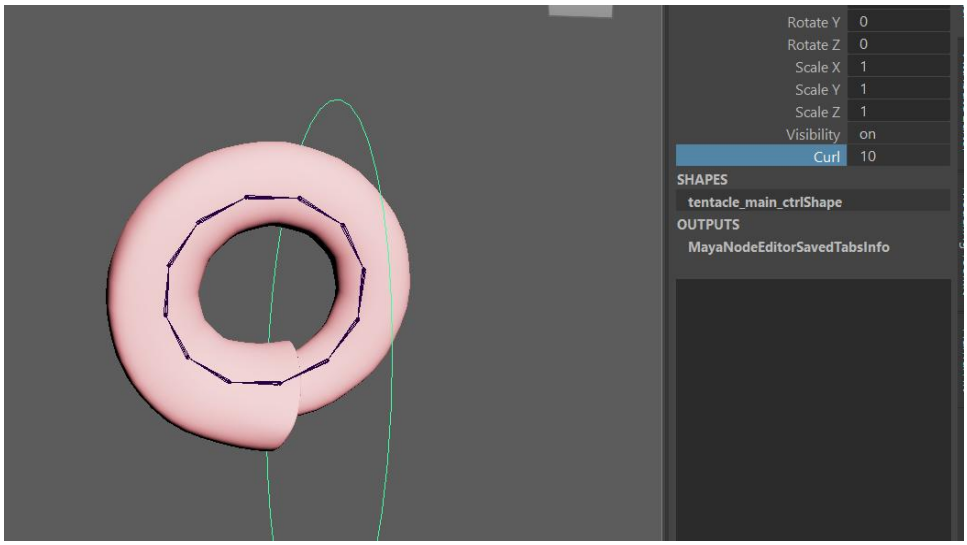


Figure 10: FK curl test created by rotating the joint chain and Set Driven Key test.

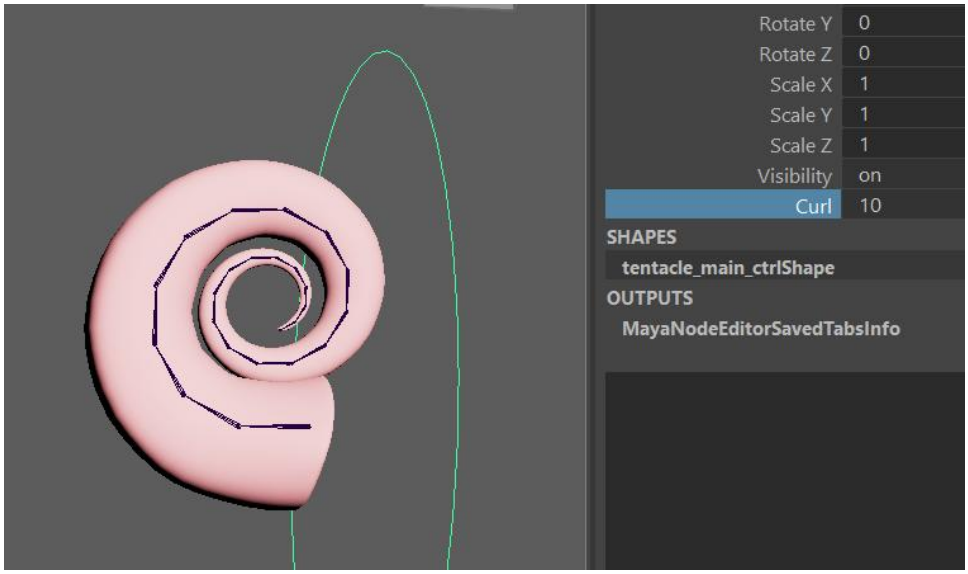


Figure 11: After set joints scale.

Final Ribbon Setup

The final solution was based on a Ribbon Setup, also commonly known as a Ribbon Rig. This workflow is often used when a joint chain needs to follow a smooth deforming surface instead of a single curve. In my rig, the Ribbon Setup used a NURBS surface and follicles to drive the tentacle joints.

I first created a control curve with five main points. Clusters and NURBS controllers were added to control this curve. Instead of making the joints follow the curve directly, I created a NURBS surface as the ribbon layer. The surface was divided into 25 sections, which allowed it to deform smoothly along the length of the tentacle. Compared with a simple curve, the NURBS surface maintained a more continuous shape and reduced sharp transitions.

A Wire deformer was then applied so the control curve could drive the NURBS surface. I adjusted the Wire dropoff distance so that the influence covered the full length of the tentacle. The purpose was not to rely on a specific number, but to make sure the whole surface responded evenly to the curve movement. This made the ribbon bend and stretch more smoothly.

I used Maya's nHair system, it was used to generate follicles on the NURBS surface. The hair simulation nodes were removed, leaving only the follicles as surface

attachment points. Because follicles remain attached to the UV coordinates of the deforming surface, they can follow the surface movement accurately. The tentacle joints were then constrained to these follicles, allowing the joint chain to inherit both position and orientation from the ribbon surface.

This system was much more stable than the earlier IK Spline test. Instead of following a curve that could easily form sharp angles, the joints followed a smooth surface. This created a more organic deformation foundation and made the tentacle feel less segmented.

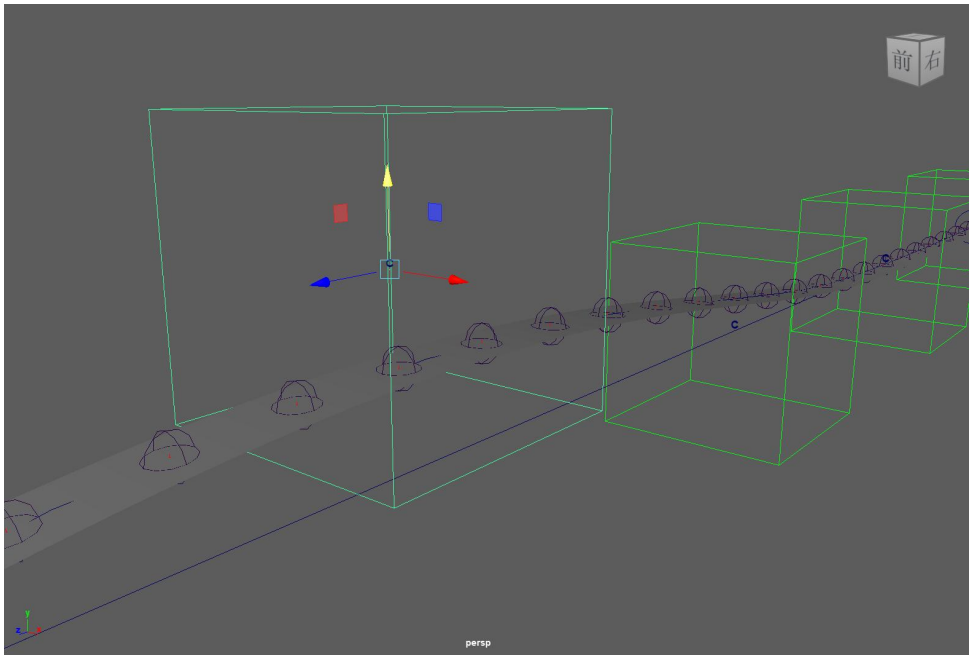


Figure 12: Ribbon Setup and IK controller.

Non-linear Deformers and Input Order

To create more exaggerated deformation, I added Maya's Non-linear deformers, including Bend, Sine and Twist. The Bend deformer was used for large sweeping arcs, the Sine deformer was used for wave-like movement, and the Twist deformer allowed the tentacle to rotate along its length.

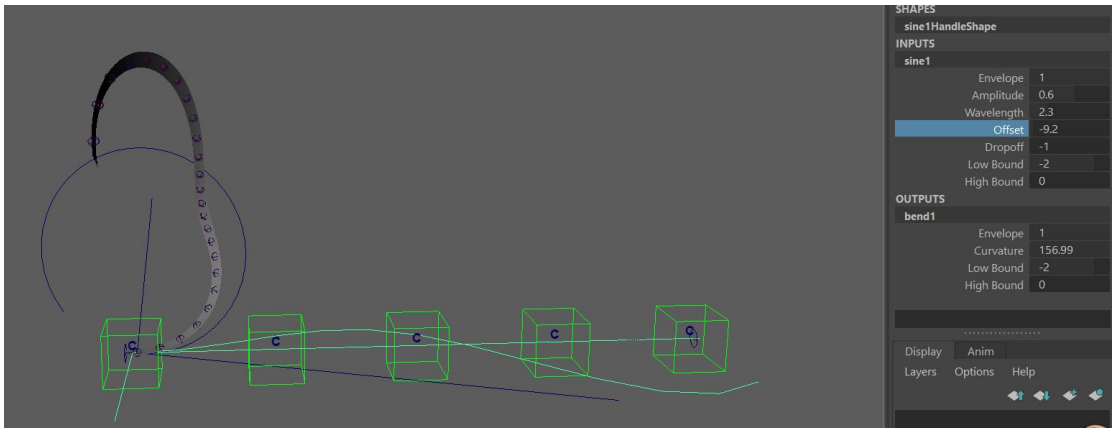


Figure 13: After added Maya's Non-linear deformers.

The default effect of these deformers often begins from the centre of the object, but I wanted some effects to start from one end of the tentacle. Therefore, I adjusted the position, falloff and hierarchy of the deformer handles so the deformation could travel more naturally from the base or tip. I also created two larger end controllers. The smaller square cluster controllers were parent constrained to these end controls, allowing the tentacle to be stretched, compressed and rotated more quickly from both ends. The middle cluster controllers were constrained between the two end controllers. By adjusting the constraint weights, each middle controller could follow the head or end controller more strongly, which helped create smoother stretching and bending between both ends of the tentacle.

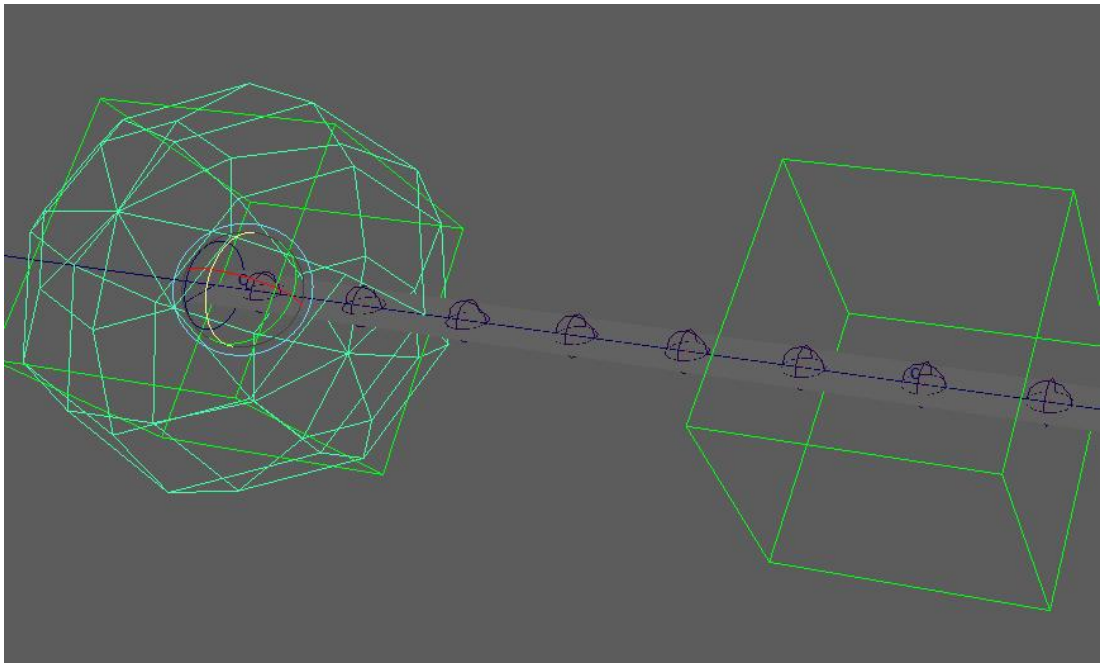


Figure 14: End controller setup for the tentacle rig.

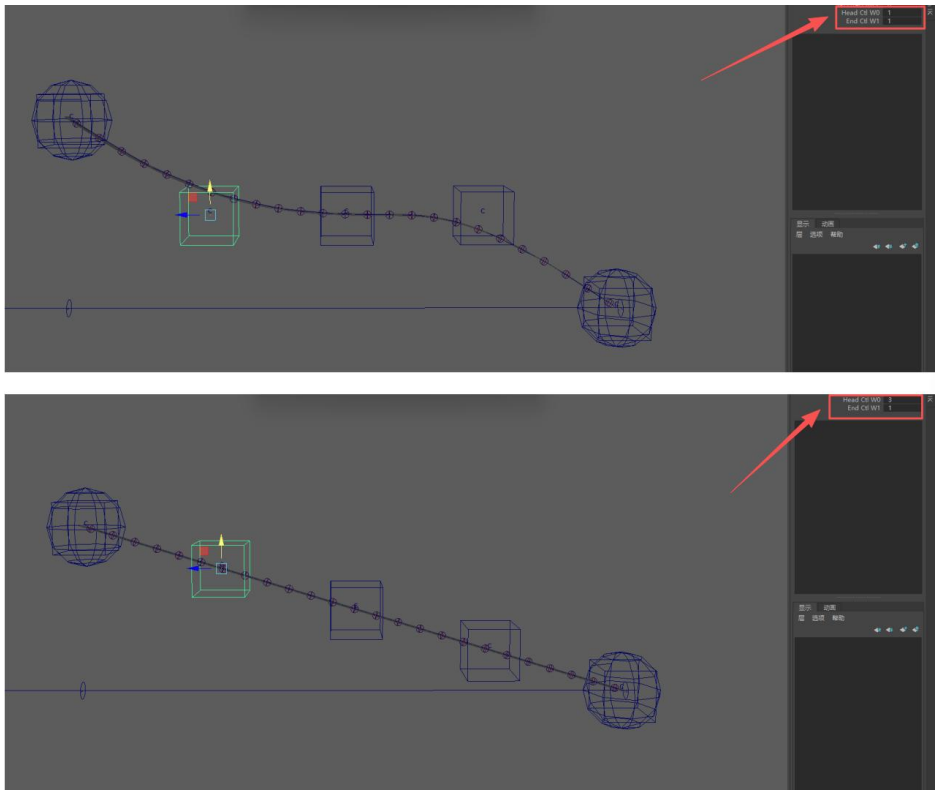


Figure 15: Constraint weight setup for the middle cluster controllers. The weight values control how strongly each cluster follows the head or end controller, allowing smoother movement between both ends of the tentacle.

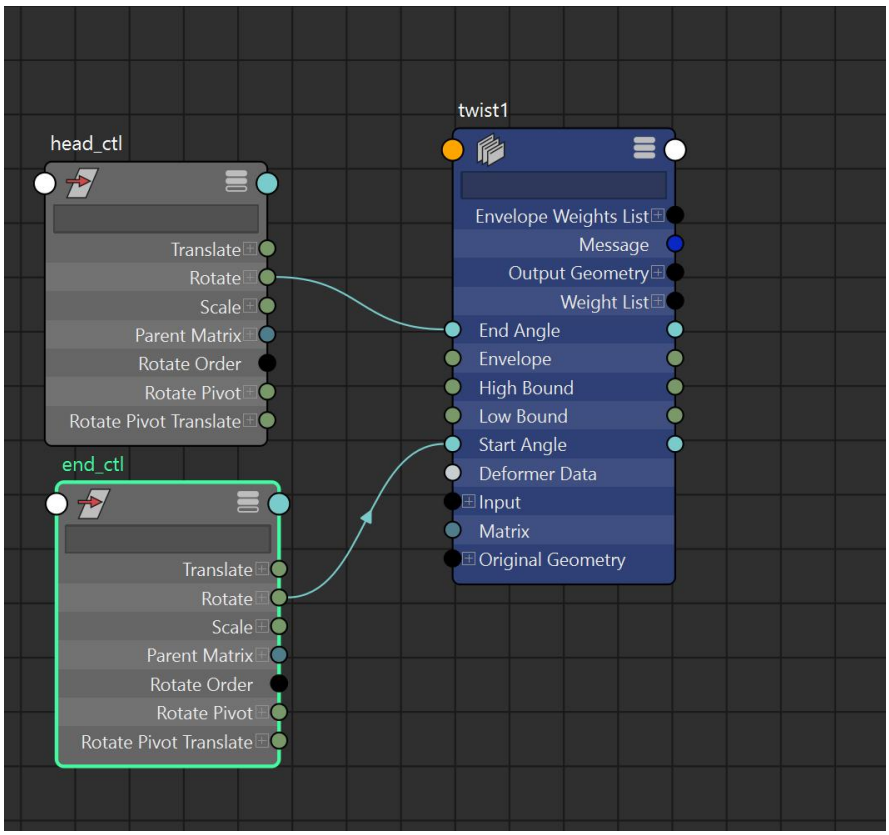


Figure 16: Twist deformer connection in the Node Editor.

During this stage, I encountered an Input Order issue. When I rotated and stretched the end controllers at the same time, the NURBS surface began to offset from the controller. After checking the node structure, I found that the Twist deformer was being evaluated before the Wire deformation. This meant the surface was twisting before it properly followed the main curve. By changing the deformer evaluation order so that Wire was calculated before Twist, the offset problem was reduced. This showed that node graph topology and Input Order are important for rig stability, not just the placement of controllers.

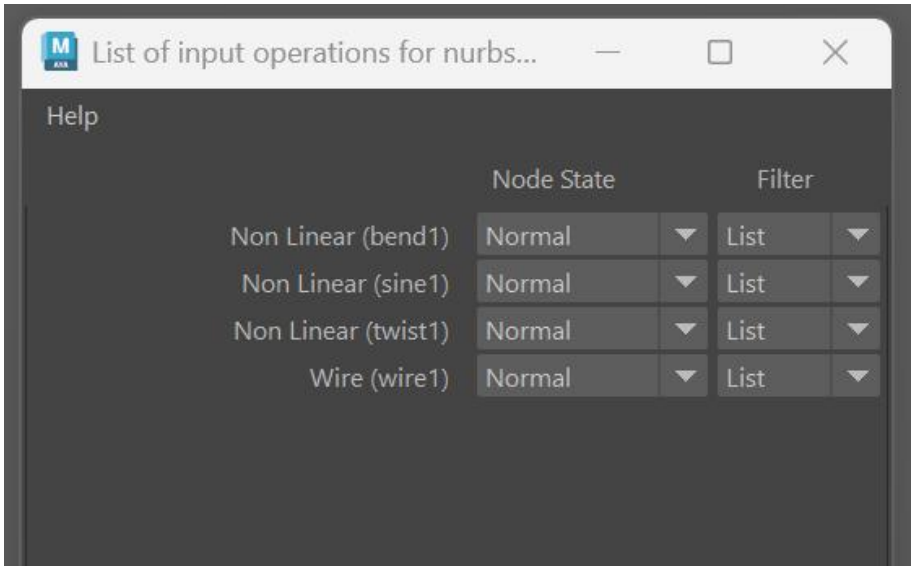


Figure 17: Input Order setup for the NURBS surface.

I adjusted the basic values of the Bend, Sine and Twist deformers, including Envelope, Curvature, Amplitude, Wavelength, Low Bound and High Bound. This helped control how strong each deformation was and where the effect started and ended on the tentacle.

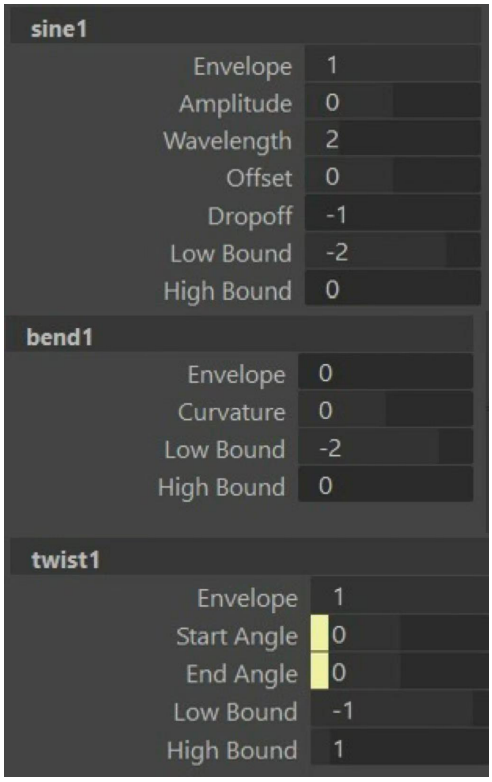


Figure 18: Adjusted parameter values for the Non-linear deformers.

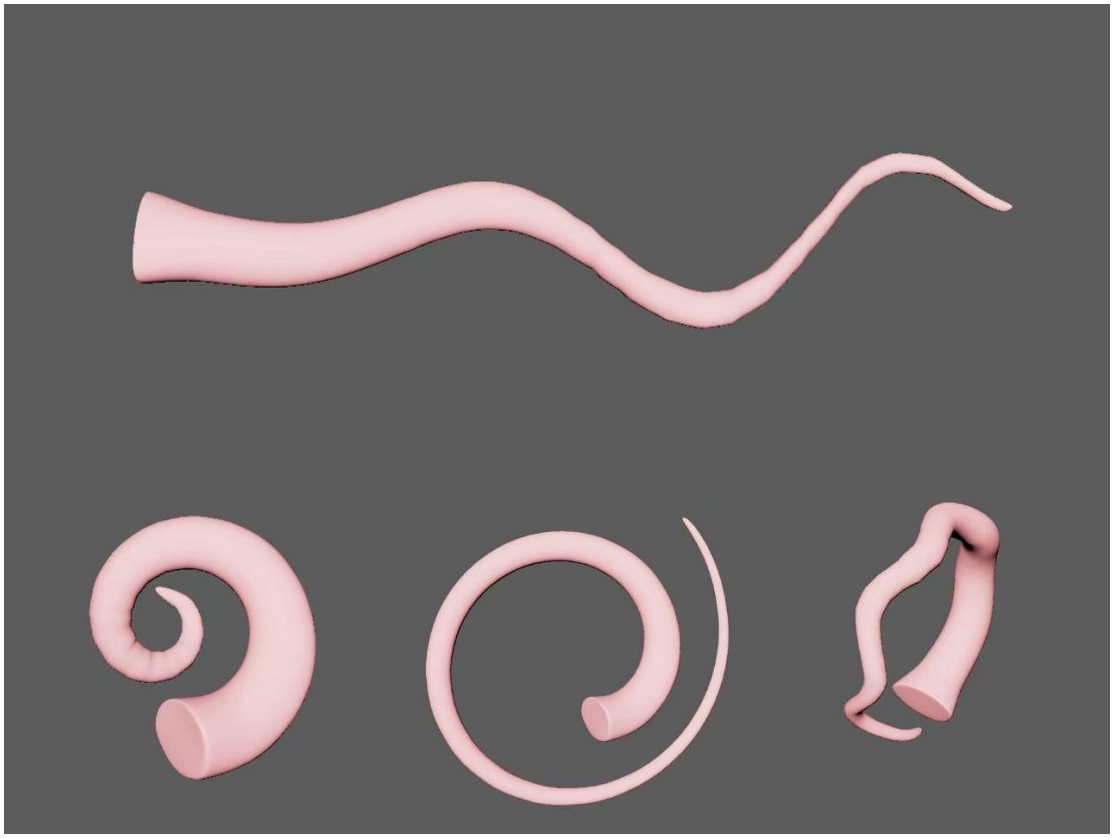


Figure 19: Tentacle deformation tests showing different stylised poses.

Squash, Stretch and Volume Preservation

I also built a curve-length-based squash and stretch system. This system used two curves: one as the original length reference and one as the current deformed curve. CurveInfo nodes measured the arc length of both curves, and a multiplyDivide node calculated the ratio between the current length and the original length.

This ratio was connected to the scale attributes of the joints. When the tentacle was stretched, the joints scaled along the main axis. At the same time, the X and Y scale values were compensated in the opposite direction, so the tentacle became thinner as it became longer. When the tentacle compressed, the opposite happened: it became shorter and thicker. This created a visual sense of Volume Preservation.

The multiplyDivide node was important because it allowed the rig to respond automatically to the changing curve length. Instead of keyframing the scale manually, the rig calculated the stretch and width compensation through the node network. I also

added an option to turn the stretch system on or off, so the animator could disable the automatic squash and stretch when a more rigid pose was needed.

This system worked well visually, although it is still an approximation. The current setup uses linear inverse scaling, which is suitable for stylised deformation. However, it does not create mathematically perfect volume conservation. A future version could calculate the cross-sectional area more accurately, possibly using a square-root-based relationship between length and thickness.

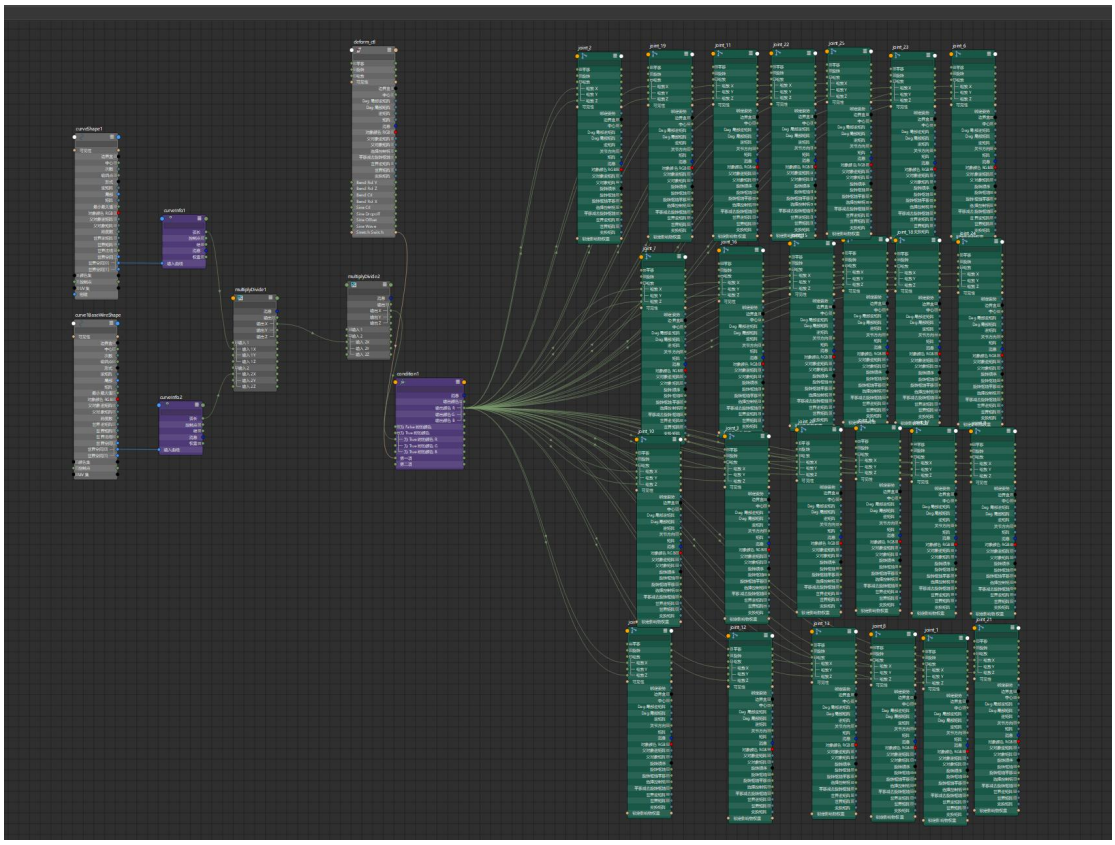


Figure 20: Node Editor overview of the stretch and squash system. The curveInfo and multiplyDivide nodes were used to calculate the length ratio between the original curve and the deformed curve, then drive the joint scale values automatically. This allowed the tentacle to stretch, squash and maintain a visual sense of volume preservation.

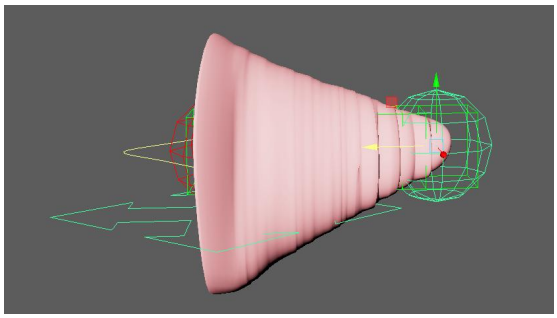


Figure 21: Squash test showing the tentacle becoming shorter and thicker when the end controllers are pushed together.

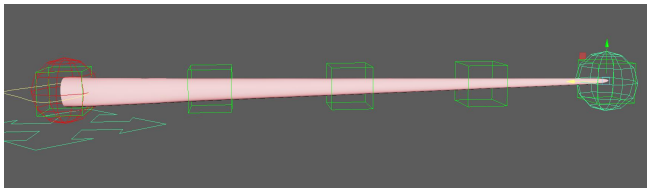


Figure 22: Stretch test showing the tentacle becoming longer and thinner when the end controllers are pulled apart.

Control Organisation

In the final rig, the main controller and deformation controller were kept separate. The main controller was used for the overall organisation and global movement of the tentacle rig. It allowed the whole system to be moved or managed as one rig.

The deformation controller was created separately for the procedural deformation attributes. This controller contained custom controls for effects such as Bend, Sine, Twist and stretch-related functions. These attributes were connected in the Node Editor to the relevant deformers and utility nodes.

Separating these controllers made the rig clearer to use. The main controller handled global movement, while the deformation controller handled stylised shape changes. This reduced the risk of accidentally changing deformation values when only moving the whole rig, and it made the control system easier to demonstrate.

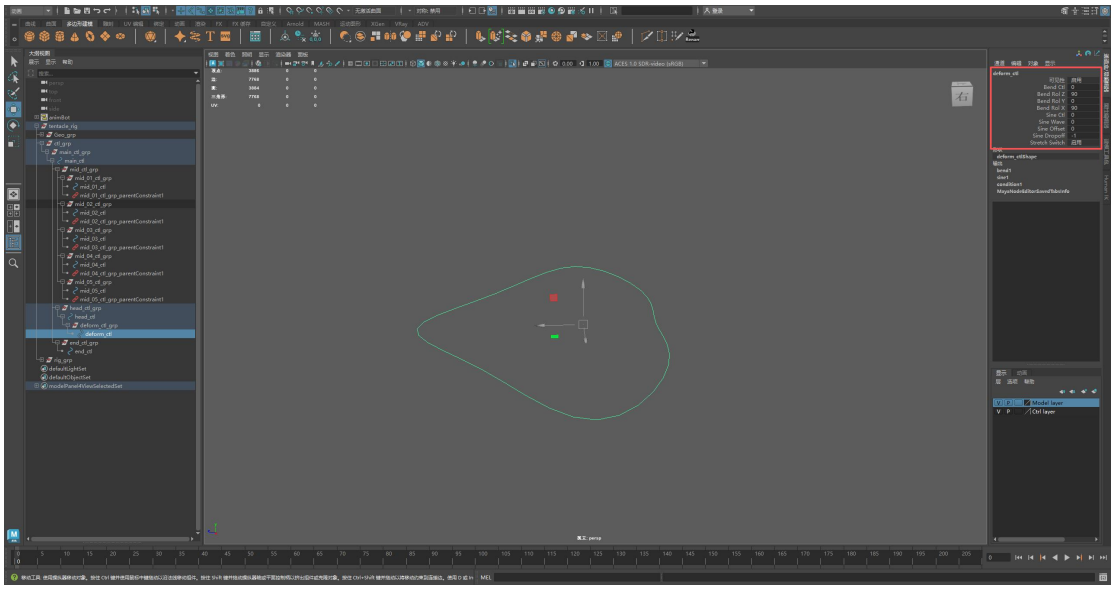


Figure 23: Deformation controller containing custom attributes for Bend, Sine, Twist and stretch-related functions.

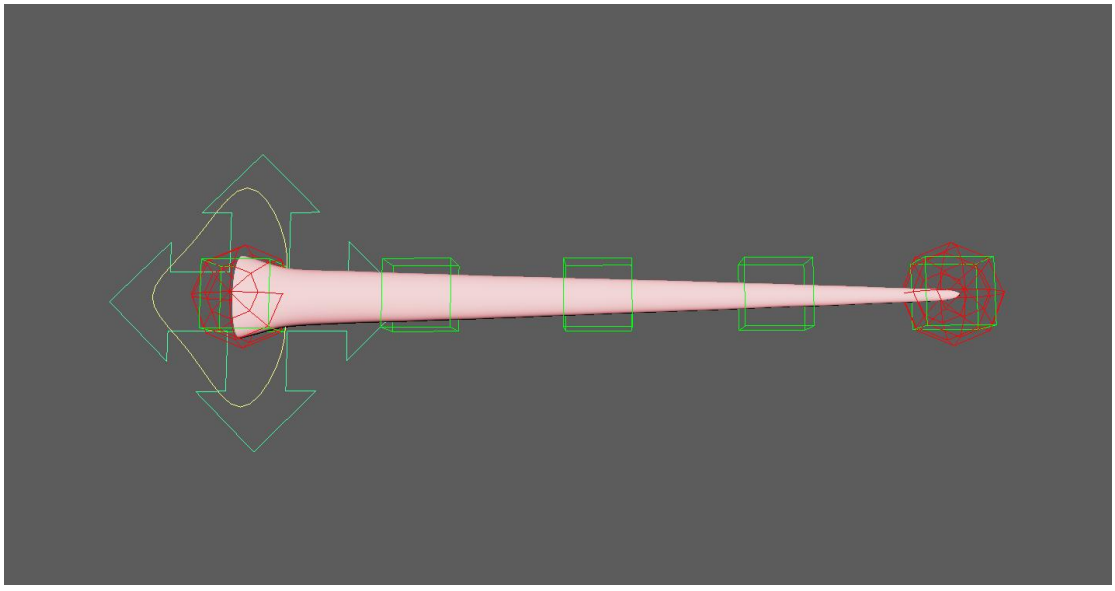


Figure 24: Overall view of the final tentacle rig and controller layout.

Reflection

Overall, the final Ribbon Setup was the most effective solution tested in this project. The early IK Spline and cluster setup could create basic flexible movement, but it struggled with clean spiral curling. FK rotation produced a better curl shape, but it was inefficient for animation and difficult to combine with other effects. The Ribbon Rig provided a better balance between smooth deformation, stability and control.

The project also showed that rigging problems are often structural rather than only visual. The Input Order issue demonstrated how the order of node evaluation can affect the final deformation. Even if the controllers and deformers are correct individually, the rig can still break if Maya evaluates them in the wrong order.

The squash and stretch system also helped connect the rig to animation principles. By using curveInfo and multiplyDivide nodes, the tentacle could stretch, compress and visually preserve volume. This made the deformation feel more organic and less like simple scaling.

However, the final rig is still a technical prototype rather than a complete production-ready system. In future development, I would improve the curl control so it could automatically generate a cleaner spiral while still working with the Ribbon Setup. I would also optimise performance, because NURBS surfaces, follicles and multiple deformers can be heavier than a simple joint rig. A proxy rig or simplified viewport version would make the setup easier to animate in real time.

Another possible improvement would be dynamic simulation. Tentacles naturally have drag, overlap and secondary motion. Adding a dynamic curve system or Maya Nucleus setup could help create automatic trailing motion and reduce the amount of manual keyframing.

This project helped me understand that a successful tentacle rig needs more than movement. It needs smooth deformation, stable node structure and clear animator controls. Through testing IK Spline, FK curling, Set Driven Key, Ribbon Setup, Non-linear deformers and Volume Preservation, I developed a stronger understanding of how different rigging systems can be combined to create exaggerated deformation.

References

CNET News. (2016) *How Pixar created Hank the octopus for Finding Dory*. Available at: [https://www.cnet.com/videos/how-pixar-created-its-most-complex-character-yet-for-finding-dory/] (Accessed: [02/05/2026]).

Williams, R. (2001) *The Animator's Survival Kit*. London: Faber and Faber.

CG (n.d.) Demo Rig Octopus Cindy GUSTAVE. Available at: https://youtu.be/0mU1ZXikZk0?feature=shared

Charlie V 3D (n.d.) How to rig a snake in Maya. Available at: https://youtu.be/58F15mLA6Uo?si=6jzbtqW4QJ_olT4H

bazsa (2016) Rose 3D Model. Available at: https://free3d.com/3d-model/rose-31675.html