2019

The link between body and behavior: A new way to quantify morphological computation with variations in the snake morphology

Steven Park
Vassar College

The Link between Body and Behavior: A New Way to Quantify Morphological Computation with Variations in the Snake Morphology

Steven Park

Vassar College

May 23, 2019

Senior thesis submitted in partial fulfillment of the requirements for the major in Cognitive Science

First Reader: John H. Long, Jr.

Second Reader: Kenneth Livingston

Acknowledgements

This project represents the culmination of my undergraduate education at Vassar College and truly encompasses everything I have learned over the past four years. I would like to thank Bruce Jayne at the University of Cincinnati and Henry Astley and Derek Jurestovsky at the University of Akron for their valued assistance during the initial stages of this project. Their knowledge and expertise on snakes have not only helped me prepare the physical model needed for this investigation but have also inspired me to further pursue my interest in snake research with renewed fervor. I would like to thank Phillip Cooper, Vassar's instrumentation technician, for helping me with the construction of the central pattern generator and providing useful advice whenever a technical issue arose during the project. I would also like to thank Nicholas Livingston for his assistance with the 3D printer and the use of the equipment in the Interdisciplinary Robotics Lab. Without the help of these two people, this project would have never reached its completion in time. I am also grateful for my friends and family for their support and encouragements, not just for this project but throughout my Vassar career.

However, above all else, I dedicate this senior thesis to my two thesis readers, Professor Kenneth Livingston and Professor John H. Long, Jr. To Prof. Livingston, I would like to thank you for starting my journey into cognitive science. When I first entered Vassar College as a first-year student, I had no clue what cognitive science even meant. It was due to your guidance, praise, and wonderfully enchanting introductory class to cognitive science that caused me to fall in love with this discipline. If it wasn't for this initial spark that you provided, I never would have reached where I am now. To Prof. Long, who has helped me countless times as both a major advisor and a friend, I would like to thank you for the many cherished hours I have spent in your classes and in your office. Spending time with you have been an absolute treasure that has opened so many new doors that I never even knew existed. I cannot thank you enough for your dedication towards teaching and your willingness to put others before yourself at every turn. I could not have asked for a better mentor to guide me throughout my journey from end-of-the-year course registrations to Tadro research with my colleagues Xinyue Hu and Connor McShaffrey. I feel truly blessed to have had the opportunity to study cognitive science and experience all the joys that Vassar had to offer.

Abstract

In response to the classical symbolic approach to cognitive science that emphasizes the importance of representational structures in the mind and the use of computational procedures, Rodney Brooks and Rolf Pfeiffer advocated for a theory of cognition that centers on embodiment, in which intelligent behavior arises as a result of the body's interaction with the surrounding environment. Under this notion of embodied cognitive science, the body's morphology influences its perception and guides not only its behavior but also the amount of control the body has over those actions. However, by completely cutting ties with the computational theory of mind and representational states, the field of embodied cognition has difficulties producing formal, quantitative models without delving into highly sophisticated mathematical formalisms. Morphological computation represents a potential solution by suggesting that the body performs certain processes to offload the computational burden needed for behavior on the brain, but it is as challenging to quantify as embodied cognition. This project attempts to use algorithmic information theory as a new approach to measure morphological computation. In order to gauge the effectiveness of this approach, eight morphologically different vertebral columns based on the corn snake (*Pantherophis guttatus*) were tested on a central pattern generator (CPG) to imitate the snake's lateral undulation behavior. The computer code running the CPG was then modified so that all variants exhibit the same degree of lateral undulation and then compared with the control to measure the computational load relieved by each morphology. Through this investigation, we tested the viability of algorithmic information theory as a way to quantify embodied cognition by measuring how the morphology of the snake body aids its undulatory locomotion. Although noticeable discrepancies were present in ascertaining how different morphological elements influence the physical properties of the body, the experiment uncovered a positive correlation between the changes made to the morphology and the amount of computation needed to adjust the behavior in order to account for those morphological changes. As a result, we have determined that the general framework of this proposed approach offers substantial promise in navigating the relationship between the morphology of the agent and the cognitive computation required for its body to execute specific behaviors.

Table of Contents

Introduction

As part of an effort to better understand human intelligence, scholars have often used analogies between the human body and the latest technological marvels to distill complex systems into more easily accessible models. In 1633, French philosopher René Descartes once proposed that a certain hydraulic lifeforce powered the motions of the human body by coursing through the nerves after he witnessed mechanical statues in the royal gardens of Saint-Germain-en-Laye moving once water was pumped through an elaborate network of hydraulic pipes (Gottlieb, n.d.; Woody & Viney, 2017). Then in the 1860s, German physicist Hermann von Helmholtz depicted the nervous system as a network of telegraphic wires, transmitters, and receivers based on the recently invented telegram (Hoffman, 2003). More recently, the notion that the brain is or is like a computer has dominated discussions about cognition over the past several decades as an attractive way to visualize how the human mind works (von Neumann, 1958). The widespread adoption of this popular metaphor illustrates how the perception of the human brain as a computational system has become deeply entrenched in the study of cognitive science.

The History of Computation

In the mid-1600s, German polymath Gottfried Wilhelm Leibniz became entranced with a wonderful idea: a universal alphabet of symbolic expressions, where each symbol represented a definite concept of human thought. Together, these symbols formed a calculatable language driven by the calculus ratiocinator—the algebra of logic—that encompassed the full scope of the mind. Leibniz called this system a "universal characteristic," and he believed that this formal language of symbol manipulation possessed the capacity to map out the complete extent of human reasoning under mathematical laws (Davis, 2000).

Following Leibniz's death in 1716, the world of mathematics experienced a series of enormous transformations as logicians such as George Boole, Gottlob Frege, and Georg Cantor advanced the study of arithmetic to produce the universal language of logic that Leibniz envisioned. In particular, Frege formulated a revolutionary system of logic in 1879 called Begriffsschrift (or first-order logic), which he described as "a formula language, modeled upon that of arithmetic, for pure thought" (Davis, 2000, p. 48). Frege essentially created a set of notations for logical relations, including negation (not A), the universal quantifier (for all A), the existential quantifier (there exists at least one A), and the conditional (if A then B). Frege ultimately established a language for propositional calculus, which uses a set of variables and logical operators to tackle a problem, and he readied the stage for the world to formally define the concept of computation (Davis, 2000).

In 1928, German mathematician David Hilbert introduced two challenges regarding the basic logic of Frege's Begriffsschrift. His first question demanded absolute

proof that first-order logic is a theoretically reliable foundation with no gaps in its rules that would cause a premise to not reach their proper conclusion even if the deductive inferences are correct. His second question, which he called the Entscheidungsproblem, asked whether there existed a method based on first-order logic that could determine within a finite number of rule-based steps whether or not a mathematical statement—any formula at all—is universally true, a process known as an "effective procedure" (Hilbert & Ackermann, 1950; Immerman, 2015). Hilbert believed that solving these questions would not only codify exactly what could be solved, or "computed," using first-order logic but also require the mathematicians to formally define the concept of an algorithm beforehand (Davis, 2000).

The first of Hilbert's questions was solved by mathematician Kurt Gödel, who managed to provide a complete axiomatization of first-order logic in 1930 as part of his Ph.D. thesis. Referred to as the Completeness Theorem, Gödel's proposition proved that there exists a finite mechanical procedure of instructions and rules that guides every valid inference to the correct conclusion. As long as the inference is correct, there exists a set of steps based on first-order logic that will always lead to the proper outcome (Davis, 2000; Immerman, 2015). Hilbert's second question was later solved in 1936 by American mathematician Alonzo Church and his student Alan Turing, both of whom proved independently that Hilbert's much desired effective procedure could not exist. Church solved the Entscheidungsproblem by using lambda calculus, a model of computation that he developed in the 1930s. With lambda calculus, Church defined the concept of "effective calculability," which describes whether an effective method exists for a function, *i.e.*, what determines an algorithm. He states, "[I]t is true, under the same definition of effective calculability, that every function, an algorithm for the calculation of the values of which exists, is effectively calculable" (Church, 1936, p. 356).

On the other hand, Turing solved the Entscheidungsproblem by introducing an abstract model of computation called an automated machine (which later became known as a Turing machine). According to Turing, an automatic machine, or "a-machine," is a system that manipulates symbols according to the machine's internal state and a set of rules known as an algorithm—an explicit, step-by-step procedure designed to answer a question, solve a problem, or perform a task (Turing, 1936). Depending on what internal state the machine is in, the symbols are subjected to changes under different rules that may change the state of the machine and even influence the final outcome. Most importantly, Turing demonstrated that, under his model of computation, anything computable by any algorithmic process can be computed by a Turing machine. He declared, "[I]f there is a general process for determining whether a formula of the Hilbert function calculus is provable, then the determination can be carried out by a machine" (Turing, 1936, p. 249). As a result, if there exists a task that a Turing machine can't complete, then there are no algorithms that can accomplish that task, either. By following this process of logic, Turing showed that an algorithm for the Entscheidungsproblem couldn't possibly exist (Davis, 2000).

Soon afterwards, Turing proved that his thesis on Turing machines featured an equivalent approach to defining an effective method as Church's lambda calculus thesis. Combined, Church's and Turing's work produced the Church-Turing Thesis, which formally defined an effective method, i.e. an algorithm, as a method that satisfies the following criteria (Copeland, 2017):

1. The method consists of a finite number of exact instructions, where each instruction is expressed with a finite number of symbols.
2. The method will always produce the desired result in a finite number of steps if carried out without error.
3. The method can be carried out by a human being unaided by any machinery besides paper and pencil.
4. The method demands no insight or ingenuity on the part of the human being carrying it out.

Once the notion of an algorithm was clearly defined with this explanation, scholars could study the nature of algorithms with renewed vigor, marking the beginning of the modern theory of computation.

While Church's and Turing's answer dashed Hilbert's hopes of finding an effective procedure for all of mathematics, Turing's thesis on his automated machines had enormous implications in the study of cognition. By demonstrating that Turing machines possess the means of capturing symbolic algorithms, he opened the possibility that a Turing machine could capture the computation performed by the human mind. Turing proposed that there exists a universal Turing machine that could run the program of any other Turing machine as long as the algorithms on how to perform those programs are included in the universal Turing machine's own algorithm. In other words, a single machine has the capacity to perform any computational task given the proper input and algorithm. This conclusion led to a revolutionary hypothesis: What if the brain is a universal Turing machine? Based on the assumption that human mental abilities are algorithmic, this theory proposes that the human brain can be understood in much the same way as any computational system that relies on symbol manipulation and algorithms (Searle, 1990). In other words, a universal Turing machine, which aims to transform symbols based on rules and state changes, may represent how the human mind operates. Assuming that this model is accurate, computation in the human mind occurs through the manipulation of mental representations, a process that is physically implemented by neural activity in the brain. Likewise, computer software consists of algorithms that serve to transform input symbols into the desired output to achieve a task. In that sense, the human mind and a computer program are proposed to share similar principles in that both are computational systems that manipulate representations based on rules and internal states. This theoretical framework that correlates the structure of human cognition with the architecture of a computer program under the basis of computational systems represents the central hypothesis of cognitive science known as the Computational Theory of Mind (CTM).

Responses to the Computational Theory of Mind

Turing's radical proposal suggesting the human brain operates as a computational system immediately brought extreme reactions from both proponents of his theories as well as his critics. Centuries of dualism had ingrained in many people that their body housed a "soul," which experiences qualia and other uniquely human elements of cognition like language that separate them from machines and even live animals (Mesaros, 2014). As a result, the notion that the human mind functions under the same principles as a rule-based machine had prompted multiple arguments against Turing. Some argued that the ability to create art or express emotions in an attempt to highlight the differences between humans and machines, while others insisted that human cognition is much too vast to define it under a set of algorithms. In response, Turing proposed a thought experiment called the imitation game to determine whether a machine could "think." With it, he stripped down the behavior of a machine and a human to their most basic cognitive elements and provided an operational definition and a set of conditions under which one could determine whether a machine is capable of thinking (Turing, 1950).

For those in favor of Turing's vision of humanity, the Computational Theory of Mind (CTM) and the rapid progress made in the field of computer science opened the door to the exploration of Artificial Intelligence (AI), where scholars attempted to construct the computing machines that exhibited the same phenomenon as human cognition. Having fully adopted Turing computation, traditional cognitive science was dominated by CTM—now referred to as the classical computational theory of mind—and emphasized the need for computational systems to interpret real objects as mental representations before that mental state undergo systematic change. In a similar vein, the representational theory of mind also placed great emphasis on mental representations, except the focus was placed more on the process of symbol manipulation (Rescorla, 2017). Under this understanding of human cognition, early AI research emphasized intelligent behavior that involved decision-making, problem-solving, and reasoning, all of which were grounded in rule-based systems. AI researchers tried their best to fit the human mind into the framework of a computational system, often following the viewpoint espoused by the physical symbol system hypothesis (PSSH) (Newell & Simon, 1976). This empirical hypothesis stated that a digital computer has the sufficient means of demonstrating intelligent action as long as the appropriate symbol-processing programs are provided. Any system that exhibits general intelligence must be a physical-symbol system, because no other theory best demonstrates an alternative method born out of empirical evidence that can achieve intelligent activity (Newell & Simon, 1976). However, despite the early success that the field experienced, AI failed to live up to the high expectations garnered by academics and suffered a drastic blow in public interest.

Following the disappointments brought upon by traditional AI research, other approaches to cognitive science emerged in response to the flaws exhibited by the

classical computational theory of mind. Connectionism acts as one example of an alternative model of cognition, in which the neural architecture of the brain produces a network of activity dictated by interconnected nodes. Within this framework, the brain processes information by distributing representations in a sub-symbolic form across its connections (Figure 1). The behavior of the network hinges on the interactions among different nodes, which cause larger patterns to appear when grouped together. While connectionism serves a model of a brain's neural activity, it acts as more of an abstract representation of the neural processing that occurs, not a replication of the biological neural networks (Eliasmith, 2013). However, deep neural networks built on the foundation of connectionism are capable of learning patterns and generalizing its behavior in the face of novel inputs (Rumelhart, D. E., Hinton, G. E., & Williams, R. J., 1986).

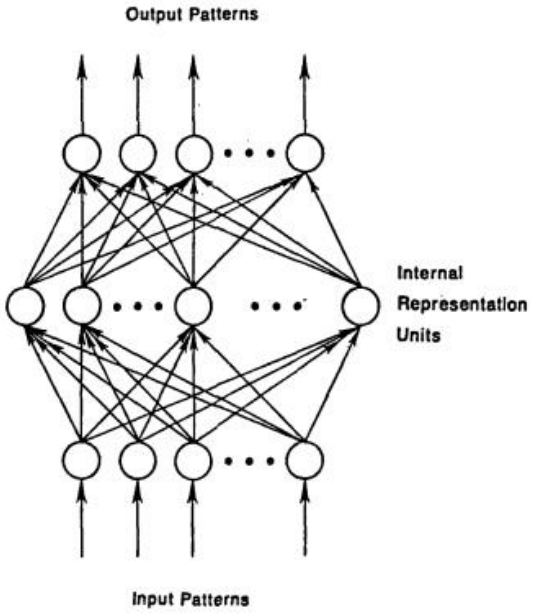


Figure 1. *A multilayer network displaying the sub-symbolic model of representation in connectionism (Rumelhart, D. E., Hinton, G. E., & Williams, R. J., 1986, p. 320).*

Another example of an alternative model of cognition is dynamicism, which compares cognitive systems to a dynamical, state-dependent system that evolves not through the configuration of symbols but through measurable, bodily actions. Rather than relying on symbol manipulation, a dynamical system consists of quantities that change corresponding to state-space evolution. In 1995, cognitive scientist Tim van Gelder proposed that the Watt Governor, a mechanism that controls the speed of an engine shaft, serves as a better metaphor of cognition than the Turing machine. Generally used to control steam engines, the Watt Governor performs different actions depending on its state, which continually changes depending on time and the environment (Figure 2). Van Gelder argued that cognitive systems could only be properly understood by characterizing their state changes through time, suggesting that

cognitive systems are directly coupled to their environment (Eliasmith, 2013). In other words, the nature of the environment can strongly influence how cognitive behaviors are realized, and intelligent behavior can be viewed as adjustments made in response to changes in that environment. Van Gelder states, “In this vision, the cognitive system is not just the encapsulated brain; rather, since the nervous system, body, and environment are all constantly changing and simultaneously influencing each other, the true cognitive system is a single unified system embracing all three” (van Gelder, 1995, p. 373).

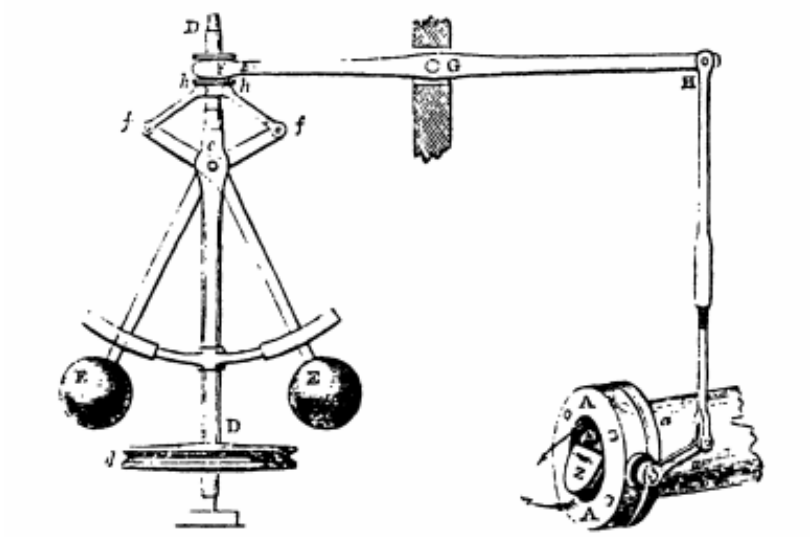


Figure 2. *The Watt centrifugal governor, which controls the speed of a steam engine based on the rotation of the flywheel. As the speed of the wheel increases, the arms of the governor were raised upward, restricting the flow of steam from the throttle valve (van Gelder, 1995, p. 349).*

Dynamicism’s efforts to integrate the body and the environment into its model of cognition resonated strongly with many academics who grew disillusioned with the physical symbol system hypothesis, early AI, and connectionism. One major critic of this traditional view of cognitive science was roboticist Rodney Brooks, who argued that the basic foundation of classical cognitive science was flawed from the onset. He explained that the push towards the computational theory of mind and the drive to explain cognition using a framework of symbols only made it clearer that these precise, abstract models of cognition were not compatible with real, chaotic, biological systems (Brooks, 1991). Therefore, Brooks claimed that constructing an intelligent system requires its representations to be grounded in the physical world and that it must be built in a bottom-up manner. Known as the physical grounding hypothesis, this theory highlighted the need to connect the system to the physical world using sensors and actuators. Believing that the world served as its own best model, Brooks argued that a cognitive system must be defined by its physical actions (Brooks, 1990). As an example, he confirmed the viability of the physical grounding hypothesis by constructing several robots using what he called “subsumption architecture,” which organized the control system into a fixed hierarchy of priorities. Within each rank of the subsumption

architecture, a fixed network of simple finite state machines guided the behavior of the robot as it interacted with the dynamic environment (Brooks, 1991). As a result, these robots exhibited clear goal-directed actions and adjusted its behavior in response to external stimuli, much like a human being.

Brooks' work on the physical grounding hypothesis ultimately paved the way for a new approach to understanding the human mind that expanded the scope of cognition to the surrounding world. In essence, Brooks demonstrated the importance of embodiment, the state of having a physical body situated in the real world, in the emergence of intelligent behaviors. Through embodiment, the agent could accomplish cognitive tasks with simple sensorimotor interactions with the environment without the need to represent the real world. The situated nature of the body allows a substantial simplification of the control system (Paul, 2006). However, while embodiment is often summarized as "intelligence requires a body," this concept presents a much more significant set of implications than what the popular saying entails. According to computer scientists Matej Hoffmann and Rolf Pfeifer, cognitive systems under the framework of embodiment behave under the influence of not just the internal control structure, like the central nervous system, but also the ecological niche of its environment, the morphology of its physical body, and the material properties that make up the morphology. By changing the physical constraints that shape the dynamics of the embodied system's interactions with the environment, embodiment greatly impacts not only low-level sensory-motor activities like locomotion but also higher-level cognitive processes that defined the research of classical computational theories of mind (Hoffman & Pfeiffer, 2011). This brand of thinking led to what many cognitive scientists call the embodied approach to AI, or embodied cognition.

Quantifying Embodied Cognition

According to cognitive scientist Margaret Wilson (2002), embodied cognition presents six prominent claims about the nature of cognition:

1. **Cognition is situated:** Cognitive activity takes place in the context of a real-world environment, and it inherently involves perception and action.
2. **Cognition is time pressured:** Cognition must be understood in terms of how it functions under the pressures of real-time interaction with the environment.
3. **Agents off-load cognitive work onto the environment:** Because of limits on our information-processing abilities, we exploit the environment to reduce the cognitive workload. We make the environment hold or even manipulate information for us, and we harvest that information only on a need-to-know basis.
4. **The environment is part of the cognitive system:** The information flow between mind and world is so dense and continuous that, for scientists studying

the nature of cognitive activity, the mind alone is not a meaningful unit of analysis.

5. **Cognition is for action:** The function of the mind is to guide action, and cognitive mechanisms such as perception and memory must be understood in terms of their ultimate contribution to situation-appropriate behavior.
6. **Off-line cognition is body-based:** Even when decoupled from the environment, the activity of the mind is grounded in mechanisms that evolved for interaction with the environment, i.e. mechanisms of sensory processing and motor control. (p. 626)

As a response to the symbol-processing computational theory of mind, embodied cognition found success in addressing many of the problems seen in classic cognitive science and the symbol system hypothesis. In 1969, computer scientists John McCarthy and Patrick Hayes introduced the Frame Problem, which asked if whether it would be possible for classical AI, if dictated by first-order logic, to make decisions on what information is relevant when predicting the outcome of its own actions. In other words, how would a system based on pre-set algorithms and symbol manipulation continually modify its scope of reasoning so that it processes all the necessary information relevant to a given task without explicitly considering all the irrelevant information? Philosopher Jerry Fodor called this the "Hamlet problem," emphasizing the need for the robot to avoid situations where it forces itself to think through every possible scenario instead of focusing only on what is relevant (Shanahan, 2016). In 1980, John Searle posited another problem with classical AI with the Chinese Room Argument. In this thought experiment, Searle argued that even if a system like computer is tasked with manipulating symbols based on a given set of algorithms, there is insufficient evidence to claim that the computer demonstrates intelligence with this behavior. According to Searle, computational processing systems cannot derive meaning from symbol manipulation (Searle, 1980). In both cases, the issue with classical AI and the physical symbol system hypothesis stems from the lack of intentional agency: Purely computational systems cannot determine what features of the world are significant from the perspective of the system itself, rather than that of the human designer (Froese & Ziemke, 2009). In contrast, embodied cognition has managed to address these problems through the very design of its framework.

However, as if to completely wipe the slate clean, there has been a powerful push to erase the computational theory of mind and the physical symbol system hypothesis from cognitive science and establish embodied cognition as the dominant model. For instance, cognitive scientists Andrew Wilson and Sabrina Golonka called embodied cognition "the most exciting hypothesis in cognitive science" and theorized that the focus on the interaction between the body and the world will replace the need for complex internal mental representations (Wilson & Golonka, 2013, p. 1). Philosopher Lawrence Shapiro noted that one of the three most prominent themes in the embodied

cognition literature was the replacement hypothesis, the call for cognitive science to give up completely on algorithmic processes over symbolic representation as an area of study and direct all attention towards embodiment (Shapiro, 2011). After decades of disappointment over the slow pace of progress in classical AI, frustration over symbol manipulation and computation within the cognitive science field had reached a boiling point that has resulted in a zealous rejection of the old ways. This is unsurprising, given how the early emergence of embodiment theory was tinged with bitterness. Brooks himself believed that the symbol system hypothesis had reached the end of its usefulness, stating "[O]nce this commitment [to the physical grounding hypothesis] is made, the need for traditional symbolic representations soon fades entirely" (Brooks, 1990, p. 5).

However, numerous critics have argued that embodied cognition lacks the sufficient capacity to completely overthrow the foundational theories of classical cognitive science. According to a report by cognitive scientist Stephen Goldinger and his colleagues, the basic principles of embodiment theory are either unacceptably vague or offer nothing new, thus providing no scientifically valuable insight. In other words, they criticized how embodied cognition resembles a dead end in terms of empirical research. The report asks, "Cognition is influenced by the body. As a scientist, what can you do with this claim? How might we write an equation that expresses embodiment? [...] How can the environment...be parameterized?" (Goldinger, Papesh, Barnhart, Hansen, & Hout, 2016, p. 964). Despite the semblance of fertile ground for major scientific advancement, embodied cognition offers very few opportunities for scholars to create an overarching formal model, limiting investigations to broad, qualitative predictions. Goldinger et al. (2016) notes that, despite its immense popularity and appeal, embodied cognition currently consists of vague claims that desperately require some way to test empirically. In addition, they note how supporters of embodied cognition seem to have difficulty agreeing on a formal definition based on how the concepts of this model vary across publications. Without defined parameters, this approach certainly will be unable to properly explain cognition and behavior by itself.

In response to these criticisms, one of the best strategies that embodied cognition may have in quantifying the role of the body in cognition is a concept known as morphological computation. According to Rolf Pfeiffer and Josh Bongard (2006), morphological computation refers to the notion that certain processes are performed by the body that otherwise would have to be performed by the brain. For example, the elasticity of human leg muscles allows the body to perform small adaptive movements without any neural control every time the leg impacts the ground. In contrast, a robot built without carefully considering the effects of morphology must apply complex control to simply adjust to uneven surfaces. In short, the morphology of the body operates passively in a way that relieves some of the computational burden that a behavior requires the control to process. In the context of robotics, systems with high morphological computation only need to generate motor commands when they are

needed. This act of outsourcing computation to the body serves to reduce the controller complexity and the computational cost, which helps reduce energy consumption and promote survival (Ghazi-Zahedi, Haeufle, Montúfar, Schmitt, & Ay, 2016).

As an architecture of control, morphological computation stands out as a potential method of producing a formal model, because its very concept hints at the existence of a common currency between the realm of the physical body (the physical world) and the controller (abstract computation) (Paul, 2006). The classical computational theory of mind maintains that the brain controls cognition using representational symbols and computation, while embodied cognition maintains that the body controls cognition using interactions between the body and the environment. However, based on the logic of morphological computation, embodied cognition can be understood in the same framework as classical cognitive science. If the physical interactions of the body can perform computations to offload the burden of the mind, then that means embodiment and the computational theory of mind are two sides of the same coin. In other words, these two models of computation, if morphological dynamics are a computation, are connected to each other, not isolated like both sides of the feud had argued. Morphological computation acts as the bridge that connects embodied cognition with classical cognitive science, allowing the former to be measured using the metric of the latter. However, Paul (2006) notes that morphological computation is only one of the many mechanisms by which the morphology and control trade-off occurs. But now, embodied cognition has a clear direction, in theory, on how it can proceed to parameterize itself.

Unfortunately, the theory of morphological computation has encountered the same problem with embodied cognition in terms of empirical research. The idea of morphological computation is incredibly attractive, but parameterizing it remains immensely challenging. Rolf Pfeiffer and Fumiya Iida makes a note of this serious dilemma, stating, “One problem with the concept of morphological computation is that while intuitively plausible, it has defied serious quantification efforts: We would like to be able to ask ‘How much computation is actually being done?’” (Pfeiffer & Iida, 2006, p. 5). Currently, it seems unclear as to how one would even measure morphological computation. For example, how would one determine how much of the computational legwork that the elasticity of the leg muscles offloads from the control during ambulation? While the morphology demonstrates a reduced level of complexity of the control, it presents challenges in the creation of a formal model. Because morphological computation tends to offer itself in the form of a qualitative improvement rather than a quantitative one, it’s difficult to determine the conversion factor between the morphology of the body and the symbols that dictate its control.

In addressing this issue, proponents of morphological computation have made several attempts to effectively pin down the common currency between the physical body and the abstract controller. There are currently two primary approaches to formalizing this theory, the dynamical systems approach and the information theoretic

approach (Ghazi-Zahedi, Langer, & Ay, 2017). The dynamical systems approach attempts to model morphological computation by treating the body as a type of reservoir computer, a framework of computation that consists of neural networks that behave as a dynamical system (Jaeger, 2007). However, this approach largely fails to quantify how the body reduces the computational burden of the brain. On the other hand, the information theoretic approach attempts to model morphological computation by modeling the sensorimotor loop as a causal graph, which determines the distribution of labor that contributed to an observed behavior between the internal computation and the body-environment interactions. This method has shown greater success than the dynamical systems approach but remains relatively arduous to perform (Ghazi-Zahedi, Langer, & Ay, 2017).

To address these issues, this senior thesis presents a new idea for quantifying morphological computation. I propose that algorithmic information theory, which bridges computation and information, provides a viable framework that will help establish a formal model for morphological computation and embodied cognition as a whole. According to mathematicians Andrey Kolmogorov, Ray Solomonoff, and Gregory Chaitin, the complexity K(x) of any mathematical object, x, is defined by the length of the shortest program for x in any standard computer programming language. Unlike Claude Shannon's standard information theory, algorithmic information theory's definition of simplicity applies to individual objects rather than the probabilities associated with objects. As a result, it can be intuitively assumed that cognitive systems have an inherent goal to compress the data it possesses in an easily recoverable form (Chater & Vitanyi, 2003). In short, algorithmic information theory posits that a string of characters is as complex as the length of the shortest computer program that can produce that string. The length of that program is called the Kolmogorov complexity of the string, K(s) (Yanofsky, 2018).

This concept can be implemented in the context of embodied cognition by creating a physical model that performs an observable behavior dictated by a computer program. By measuring the dimensions of the behavior in relation to the programming code that controls it, morphological computation can be quantified based on the Kolmogorov complexity, or the shortest length of the program that allows the system to behave the way it does. Thus, the length of the computer program can be equated with its computational simplicity: The shorter the program, the simpler the computation needed to perform that behavior. As a result, we can determine and measure the impact of the morphology on behavior by observing the change in the program needed to account for it, working under the assumption that a decrease in program length represents the offloading of the computational burden onto the body.

This new approach merges the morphological component of cognition with the physical symbol system hypothesis. Here, the role that morphology plays in cognition and behavior is quantified using the manipulation of symbols. The goal for this investigation is to determine the viability of this framework by first (1) demonstrating

that the morphology influences behavior and then (2) quantifying the effects of that morphology on behavior by observing how much of the computational load that the morphology took on, relative to its variants. Physical models of bodies with similar yet distinct morphologies will be created and then coupled with a central pattern generator (CPG) that will perform a specific behavior according to a fixed computer program. The CPG incorporates a neurological component in the cognitive system, linking the symbolic aspect of computation in the code with the morphological aspect of computation in the physical body. The CPG allows the physical body to behave as the controller commands without input from an outside observer. The morphology of each physical model will translate the computation of the controller into a corresponding, unique kinematic variant of the control behavior. Afterwards, the CPG code will be modified for each of the physical models to exhibit the same behavior performance as the control model. The makeup of the code can then be compared between each of the physical models to examine the structural differences in the code that needed to be implemented in order for the variant models to behave similarly to the control model. In addition, the physical properties of each physical model will be measured according to the mechanical tests inspired by Etnier (2001) to determine how each morphological trait influences the range of possible behavior that it allows for the body. These tests will determine the flexural stiffness (EI) and the torsional stiffness (GJ) of each physical model in order to obtain the twist-to-bend ratio (EI/GJ), which indicates the relative resistance of the vertebral column to bending versus twisting without reference to the absolute magnitude of either (Etnier, 2001). Therefore, a higher twist-to-bend ratio reveal a structure that twists more readily than it bends.

Using the Snake Body as the Model

Given the design of the experiment, the decision on which biological body to model is of great importance. For the purposes of this investigation, the body must exhibit clear variations in morphology that would impact behavior but have a simple enough form so that the observed behavior is simple to model and analyze. While the current literature on morphological computation often use limbs undergoing locomotion to exemplify its framework, the limb is much too complex to create an accurate physical model that accounts for every morphological attribute that contributes to its general operation. Given the criteria, the trunk of a snake serves as a simpler yet suitable model to study how differences in morphology can influence cognition and behavior.

Snakes provide an interesting case study when it comes to how their various body mechanisms interact. At first glance, their body appears quite simple—an elongated, cylindrical body with no limbs. Its lack of visibly discernable morphological features may sway some to believe that its movements are rudimentary. But despite its outwardly simple appearance, snakes can demonstrate a wide range of movement patterns that allow it to adapt to many different situations. Snakes can travel on land, climb trees,

swim underwater, traverse over sand, and even glide in a freefall just by moving their body in a variety of specific patterns. In addition, snakes have demonstrated four different modes of terrestrial locomotion, each with their own advantages and drawbacks, and can even modify and combine these movement patterns if necessary to travel in a specific environment (Gans, 1970). Researchers have also found that snakes exhibit a very homogeneous vertebral microanatomy across ecological niches. This seems to indicate that the morphology of most snakes is designed to prioritize versatility in a variety of environments rather than rigid specialization for one particular environment. However, it is important to note that snake vertebrae are by no means identical and that outliers certainly exist (Houssaye, Boistel, Böhme, & Herrel, 2013).

Given its unique anatomical structure, the body of a snake can be seen as one long organ designed for locomotion. Even with its lack of limbs, snakes can contort their entire body to move. One possible reason for this flexibility is the number of muscles they have: While the human body has around 700 to 800 muscles, snakes have around 10,000 to 15,000 (Cox, 2016). Not only that, snakes have a large number of segments in their vertebral column—generally between 200 to 400 vertebrae (Lindell, 1994). According to Gans (1970), each vertebra in the snake's vertebral column swings left and right by about 50 degrees and bend up and down by about 28 degrees in relation to its neighbors, giving the vertebral column an incredible number of degrees of freedom. By having more vertebrae than any animal on the planet, snakes have the flexibility to bend and curl freely as a result of the degrees of freedom granted by each and every vertebra in their vertebral column (Wallach & Peters, 2019).

A snake vertebra has three sets of joints, or articulations (Figure 3). The cotyle connects with the condyle, the prezygapophysis connects with the postzygapophysis, and the zygosphene connects with the zygantrum (Carmona et al., 2010). The zygosphene and the zygantrum are found almost exclusively in snakes (Wallach & Peters, 2019). While these articulations allow movement between vertebrae in the plane of each surface, movement is also restricted by those same articulations. For instance, there has been much debate over the question of torsion regarding the zygosphene-zygantrum joint and the structure of the vertebral column allowing the body to twist. Many scientists believe that torsion between vertebral joints would be impossible because the prezygapophysis-postzygapophysis and the zygosphene-zygantrum articulations oppose one another. However, biologist Brad Moon (1990) demonstrated that the vertebrae can exhibit around one to two degrees of torsion per vertebral joint, which can magnify the body's full torsional capability over several vertebrae. The snake body is also largely modular, meaning that the vertebral column consists of very similar vertebrae that function as building blocks for the design of the body. Each vertebral segment in the trunk of the body is representative of the function and design of any other length of vertebrae in the trunk of the body as well. As a result, examining a specific segment of the snake's trunk and analyzing how the morphology affects its movements may open the possibility to extrapolate the findings to the body as a whole.

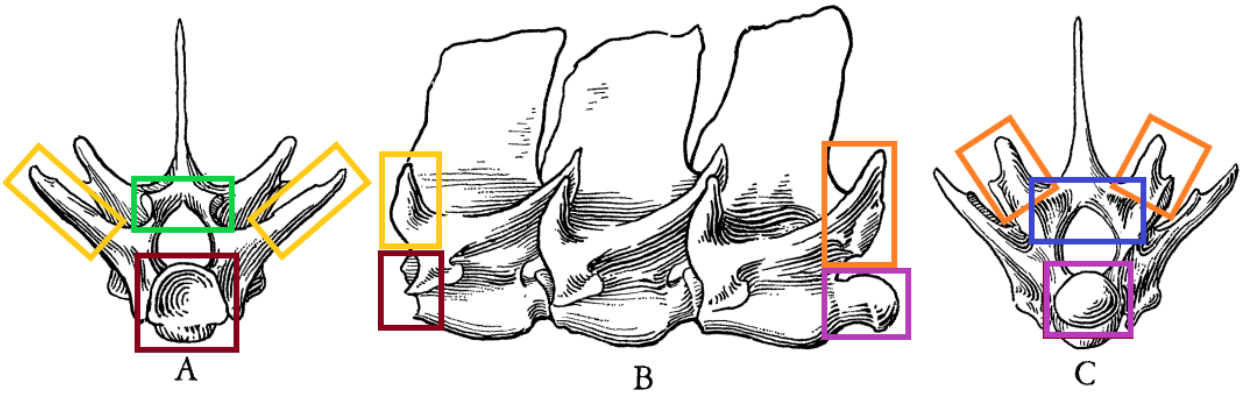


Figure 3. *Anterior (A), lateral (B), and posterior (C) views of three vertebrae of the African egg-eating snake, Dasypeltis fasciata* (Gans, 1962, p. 170). Colored boxes have been added to the image to define each structure of the vertebra. The red box contains the cotyle, the purple box contains the condyle, the yellow boxes contain the prezygapophysis, the orange boxes contain the postzygapophysis, the green box contains the zygosphene, and the blue box contains the zygantrum.

Therefore, the decision to base the physical model for this investigation on the snake body is supported by two major reasons. First, the segmentation of the snake's body plan allows for a simple yet functionally diverse model that can test for a behavior exhibited in not just a singular appendage but in the entire body—lateral undulation. A simplified version of the snake's body will be created in the form of an "infinite," meaning non-tapering and invariant with respect to length, vertebral column, which consists of identical vertebrae. In other words, the physical model will exhibit modularity, where each module acts as a functional unit that is capable of maintaining its intrinsic properties regardless of its surrounding components (Sauro, 2008). While it is associated with reduced cost in design within engineering, modularity has been observed in biological organisms as a way for the organism to explore the space of biological possibility in the context of evolution (Lorenz, Jeng, & Deem, 2015). Using an infinite vertebral column as a model will allow any changes to the morphology of the base vertebra to be magnified across the entire vertebral column. Similar to how the shape of the vertebra allows for significant torsion across multiple vertebrae, any modifications in the morphology of the vertebra will be amplified across the infinite vertebral column, allowing for easier observation of the effects of the morphological change. Second, the articulations of the snake vertebrae provide a convenient source of modification to the morphology that, based on previous work, will greatly impact the body's performance of lateral undulation. Given how the articulations allow movement in some directions and restrict movement in others, removing specific articulations would directly influence the lateral undulation to a degree where variations in behavior would be clearly visible.

Method

The experiment consisted of three major stages. During the first stage, titled "The Physical Model," nine different vertebral columns were created for the experiment with one acting as the control model and the others serving as morphological variants. Each variant displayed a combination of one of four different vertebra morphology and one of two types of body compositions. During the second stage, titled "The Mechanical Test," a large mechanical apparatus was constructed to measure the flexural stiffness and torsional stiffness of each vertebral column in order to determine each of their respective twist-to-bend ratio. During the third stage, titled "The Central Pattern Generator," the CPG was built to analyze the behavior of the vertebral columns and ascertain the computational load that each morphological characteristic offloads from the control model.

The Physical Model

The physical model was based on the corn snake, *Pantherophis guttatus*. A stereolithography (STL) file of an anterior vertebra (6 mm x 8 mm x 7 mm) from a *P. guttatus* was obtained with the help of researchers Henry Astley and Derek Jurestovsky at the University of Akron in Akron, OH. The snake from which the vertebra was modeled measured 96.7 cm SVL with a tail length of 19.1 cm and a total mass of 227 grams. In order to improve the visibility of the morphological features, the vertebra in the STL file was enlarged to about 666% of the original size of the real-life vertebra (40 mm x 55 mm x 45 mm) in the 3D printing software Ultimaker Cura 3.6.0. The vertebra clearly exhibited the relevant morphological features: the cotyle, the condyle, the prezygapophysis, the postzygapophysis, the zygosphene, and the zygantrum.

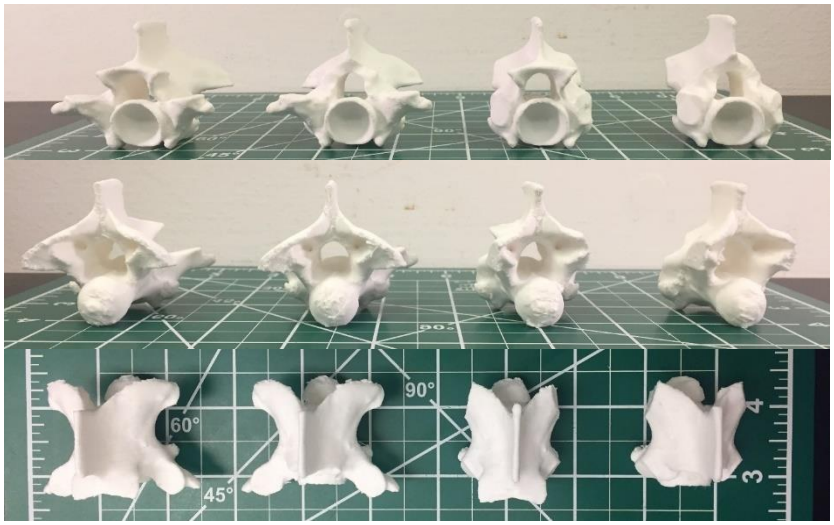


Figure 4. *The 3D-printed models of the vertebra. From left to right: Complete morphology, Variant A (no zygosphene), Variant B (no pre- or postzygapophysis, and Variant C (no zygosphene, prezygapophysis, or postzygapophysis). Top row: An anterior view of the vertebrae. Middle row: A posterior view of the vertebrae. Bottom row: A dorsal view of the vertebrae.*

With the original vertebra as the base model, three different variants were created using Meshmixer, a 3D sculpting-based CAD (Computer Assisted Design) program. These variants were produced by removing different combinations of the aforementioned morphological features from the vertebra. Variant A featured a vertebra with the zygosphene removed. Variant B featured a vertebra with the prezygapophysis and postzygapophysis removed. Finally, Variant C featured a vertebra with the zygosphene, prezygapophysis, and postzygapophysis removed. In total, four different vertebra types were used for this experiment (Figure 4). All the vertebrae were printed with a Monoprice Mini V1 3D printer using Hatchbox PLA 1.755 mm white printer filament at 10% filament density.

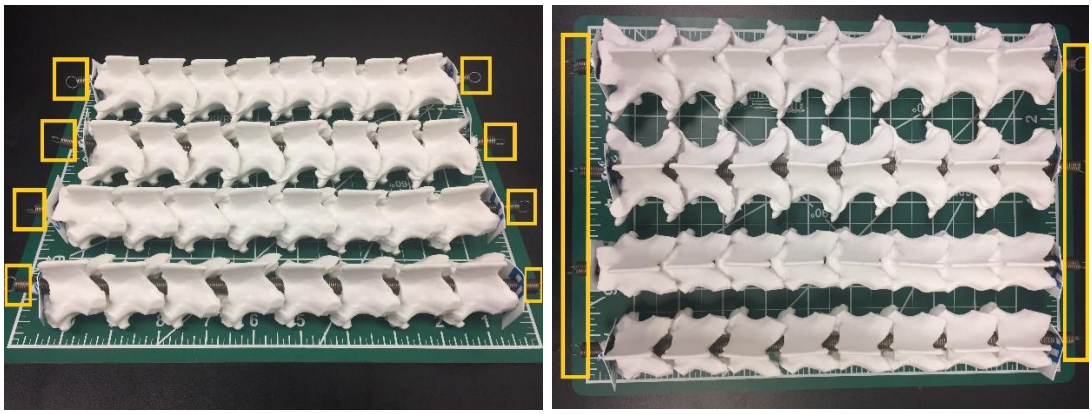


Figure 5. *The four skeleton models. The yellow boxes indicate the ends of the extension springs. From top to bottom: Complete, Variant A, Variant B, and Variant C. Left: The oblique view of the skeleton vertebral columns. Right: The top view of the skeleton vertebral columns.*

The vertebral columns came with two different body compositions, the skeleton model and the body model. The skeleton models consist of eight vertebrae of the same morphological variant joined together with an extension spring threaded through the neural canal (Figure 5). The extension springs were made of music wire with a length of 279.4 mm, an outer diameter of 7.92 mm, a wire diameter of 0.787 mm, and a load of 1.34 kg. The vertebrae on the extension spring were pressed close together so that the neighboring vertebrae fit snugly in place. In order to keep the vertebrae secured and in physical contact with their neighboring vertebrae at all times, small plastic bookends were strung through both ends of the extension spring to maintain the tension and keep the vertebrae from coming loose or falling out. The bookends were cut from paper-thin, disposable plastic rulers. A hole was made through the center of each piece to tightly fit around the circumference of the extension spring.

The body models consisted of the skeleton models encased in a matrix of silicone rubber to replicate the effects of the tissue surrounding an organism's skeleton. The muscles, tendons, and ligaments that envelop the skeleton greatly affect the physical properties of the snake body. Thus, a corresponding body model of each of the four skeleton models was created. The body models were constructed by preparing a mold

that propped up the vertebral column vertically inside a cylindrical tube. The mold was made out of three parts (Figure 6). The cylindrical tube designed to hold the vertebral column consisted of a clear plastic tube with an inner diameter of 57.15 mm that was cut to the length of 27.5 cm. The tube was also cut in half laterally so that it could open and close freely. Once the vertebral column was placed inside, the two halves of the tube were taped together, creating an enclosed area. The base of the mold, which propped up the tube, was made out of Lego parts. In the middle of the base was a hole to insert a thin metal rod with a length of 32 cm and a diameter of 5 mm. The rod held the vertebral column in place and upright once it was inserted into its neural canal.

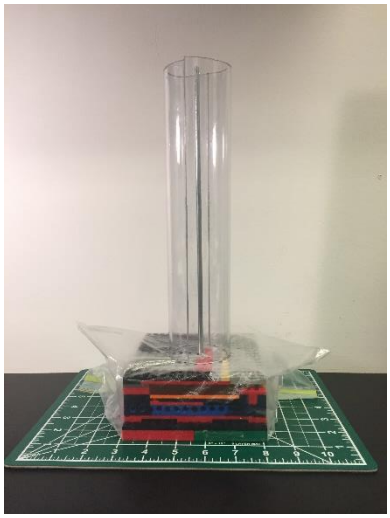


Figure 6. *Mold made using a Lego base, a clear plastic tube, and a metal rod.*

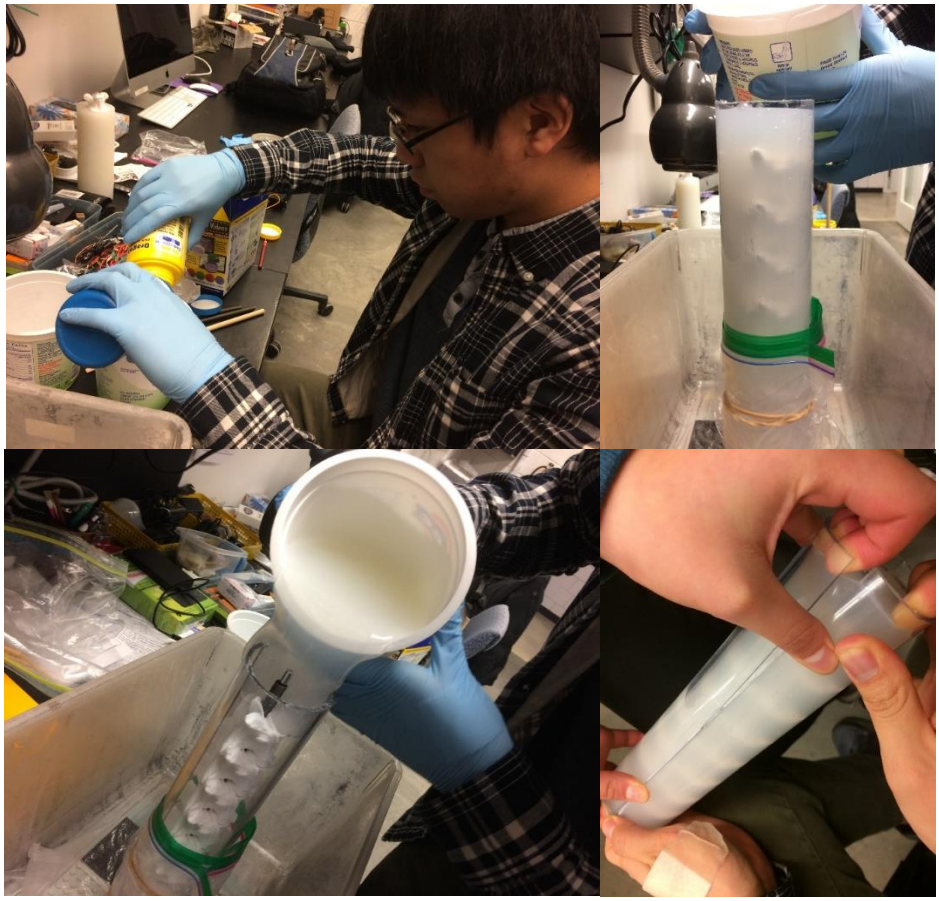


Figure 7. *The process of creating the body models using the Dragon Skin FX-Pro silicone rubber solution and the constructed mold. Top Left: The two separately-marked liquids, Part A and Part B, that came with the FX-Pro package were poured in a single container and mixed thoroughly. Bottom Left: The mixture was carefully poured into the mold while making sure that there were minimal air bubbles. Top Right: The Ziplock bags and rubber bands ensured that the silicone rubber solution did not leak out of the mold before it dried. Bottom Right: The two halves of the plastic tube were carefully separated at the end of the cure time to retrieve the finished cast.*

Once the vertebral column was placed inside the mold, approximately 0.8 liters of Dragon Skin FX-Pro silicone rubber solution was poured inside the plastic tube. The Dragon Skin solution had a Shore Hardness of 2A and a tensile strength of 288 psi. As an additional measure to prevent the contents of the tube from spilling out from the bottom, Ziplock bags were tied to the bottom end of the plastic tube with rubber bands. For the complete morphology and Variant A vertebral columns, the prezygapophysis and the postzygapophysis allowed the vertebrae to fit tightly around the inner circumference of the plastic tube, locking it in place. However, for the Variant B and Variant C vertebral columns, the lack of the prezygapophysis and the postzygapophysis sometimes caused the vertebral column to either spin or float in the silicone rubber solution. A long, wooden rod had to be used to push these vertebral columns down whenever the vertebrae started to float up to the surface. Any air bubbles that appeared on the surface were also popped to ensure that the resulting matrix was largely homogeneous (Figure 7).

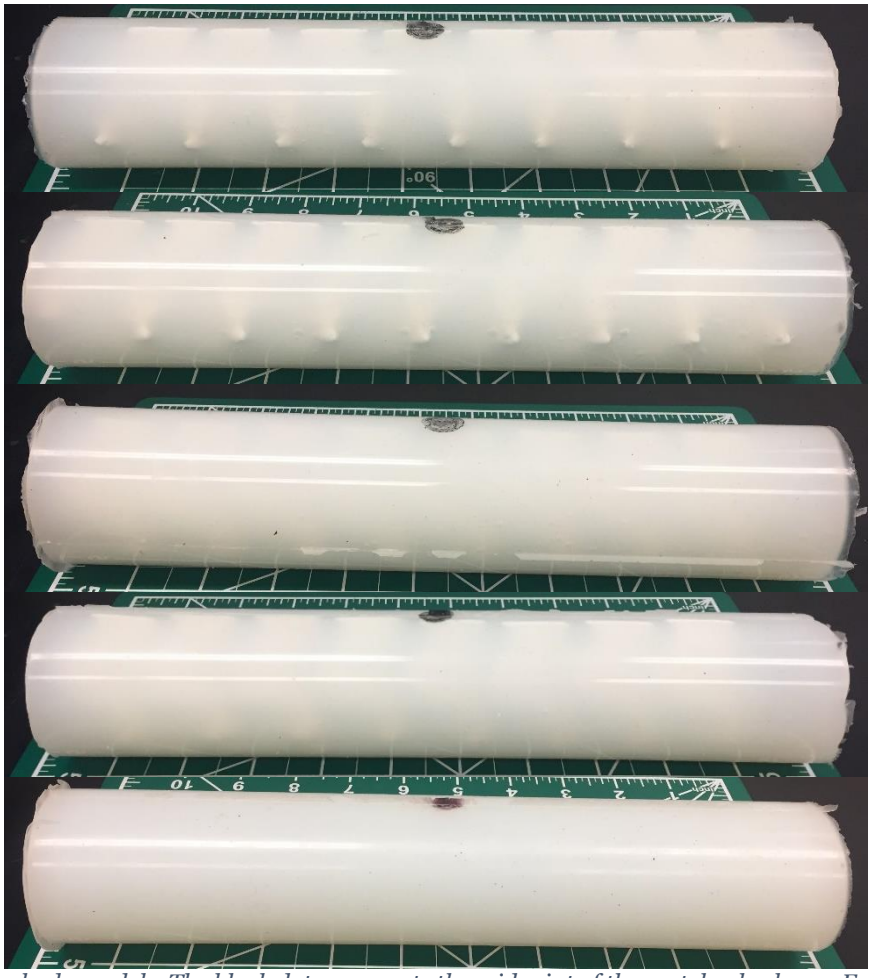


Figure 8. *The five body models. The black dot represents the midpoint of the vertebral column. From top to bottom: Body models with Complete, Variant A, Variant B, Variant C, and No Vertebrae morphology.*

Using this method, a body model for each type of skeleton vertebral column was created, as well as a ninth body model with the extension spring but without the vertebrae to test the effects of the matrix by itself (Figure 8). In total, nine different vertebral column variations were created for this experiment (Table 1).

Label	Morphology Variant	Body Composition	Length (cm)
CMB	Complete	Body	26.7
NSB	A	Body	26.5
NPB	B	Body	27
SPB	C	Body	27
NVB	No Vertebrae	Body	27
CMS	Complete	Skeleton	24
NSS	A	Skeleton	24
NPS	B	Skeleton	24
SPS	C	Skeleton	24

Table 1. *A summary of the vertebral column variants. The model CMB serves as the control for the experiment.*

The Mechanical Test

The mechanical test served to establish how the presence of each articulation, as well as the presence of the matrix, influenced the physical properties of the vertebral column. Given how the presence of the articulations seemingly locked the movement of the neighboring vertebrae within a limited range, the experiment was conducted under the assumption that removing each articulation freed up the movements of the vertebrae to a certain degree. However, the mechanical test was performed to verify the validity of this assumption. In particular, the range of possible movements granted by the articulations was determined by measuring the flexural and torsional stiffness of the vertebral columns. The design of the mechanical apparatus intended to test these characteristics was inspired by the model described in Etnier (2001). For this experiment, the mechanical apparatus was constructed by placing two Black+Decker Workmate 425 portable workbenches on top of each other and securing them in place with large metal clamps (Figure 9-A). Throughout the mechanical test, various modifications were made to the mechanical apparatus depending on the test that needed to be performed and the type of vertebral column that had to be tested.

Testing the flexural stiffness of the physical models required the apparatus to hold one end of the vertebral column from the top and have the bottom end be pulled by a string in one direction through the use of a pulley system. The force that caused the vertebral column to bend was generated by a series of weights in the form of clamps dangling in the air as part of the pulley system. The weights were added to the pulley system incrementally with each addition to the load increasing the force bending the vertebral column. A video recorder placed on a tripod stand was positioned facing the

front of the mechanical apparatus in order to capture the lateral displacement of the free end of the vertebral column caused by each set of weights.

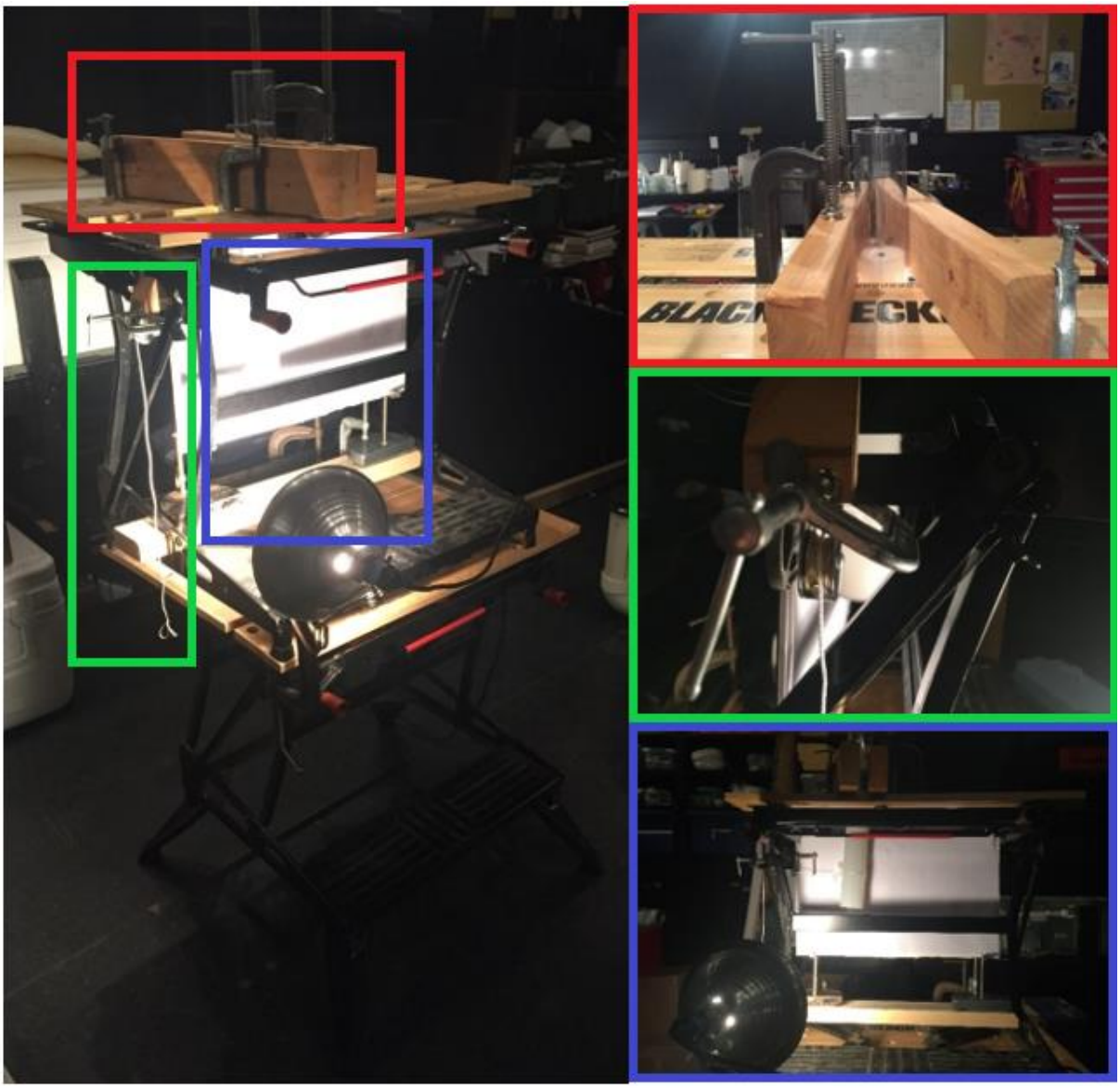


Figure 9. *The mechanical apparatus for testing the flexural stiffness of the body models. (A) Left: An oblique view of the entire apparatus. (B) Top Right (in red): A close-up view of the top platform of the mounted workbench. (C) Middle Right (in green): A close-up side view of the pulley system. (D) Bottom Right (in blue): A front view showing the camera's perspective.*

Depending on the body composition, different grippers had to be used for the apparatus to firmly grasp onto the vertebral column. For the body models, a clear plastic tube with a length of 200 mm and an inner diameter of 57.15 mm was used to hold onto approximately 50 mm of the body model from one end. The vertebral column was secured tightly inside the plastic tube by applying pressure on two fronts. First, the plastic tube containing the vertebral column was placed within the gap between the two wooden planks that made up the top platform of the mounted workbench. Second, two

lengths of plywood were clamped onto the top platform of the mounted workspace and positioned perpendicularly to the wooden planks of the platform. By tightening the metal clamps, pressure was applied on the plywood to squeeze the plastic tube between the two wooden boards (Figure 9-B). For the skeleton models, the plastic tube was too large to hold the vertebral column. Instead, a three-finger clamp oriented with the prongs facing down was used to tightly grip onto the first vertebra of the vertebral column over the gap in the top platform of the mounted workbench. A larger clamp was then used to secure the end of the three-finger clamp in place onto the platform (Figure 10).

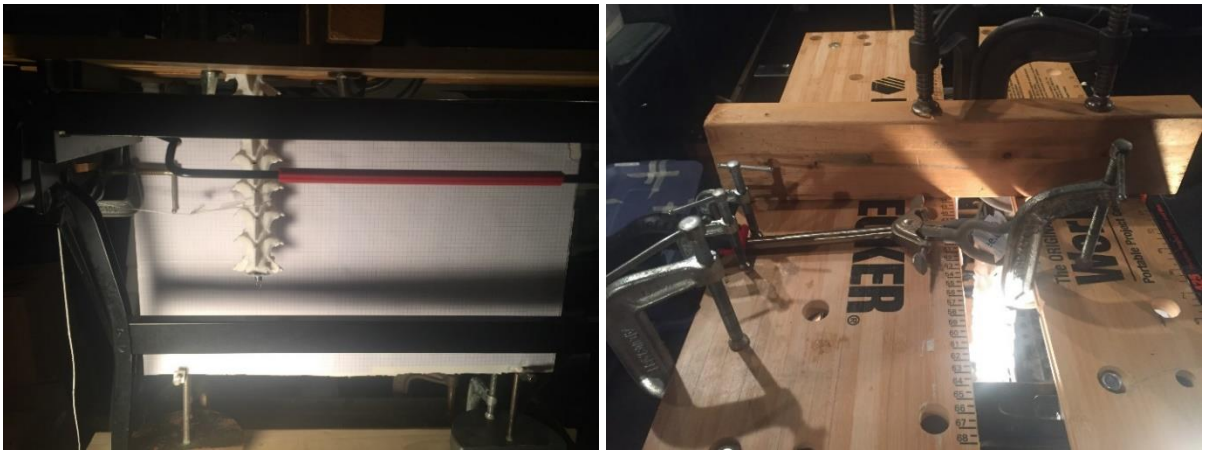


Figure 10. *The mechanical apparatus for testing the flexural stiffness of the skeleton models. Left: A front view displaying the skeleton model with the string tied around a vertebra. Right: A three-finger clamp that has been clamped to the top platform of the workbench and placed on its side was used to hold one end of the vertebral column.*

The pulley system was established on the left side of the workbench scaffolding at a height of around 125 cm from the ground. A clamp held a small wooden plank in place and another clamp fastened the handle of the pulley onto the wooden plank (Figure 9-C). A long length of string rope with a loop tied on both ends ran through the pulley with one end wrapped around the vertebral column and the other end dangled in the air. The latter end of the string rope was where weights were attached. For each physical model, the string rope was wrapped around the part of the vertebral column that caused the least slack. Therefore, the position of the loop was at the same height as the pulley, meaning that the string rope met the vertebral column at a 90-degree angle. In addition, tape was applied to the loop to tighten the grip of the string rope around the vertebral column. A backdrop was positioned right behind the vertebral column to measure its lateral displacement as it moved. The backdrop consisted of a rectangular cut-out of a poster board with sheets of graph paper taped onto the surface. Each block on the graph paper was 0.2 in x 0.2 in. The backdrop was held up by two ring stands with the use of two clamp holders. These two rings stands stood on top of a long piece of plywood placed through the bottom side holes of the workbench with the base of each ring stand clamped onto the plywood. Lastly, a lamp was clamped onto the front corner of the workbench to illuminate the vertebral column and the backdrop (Figure 9-D).

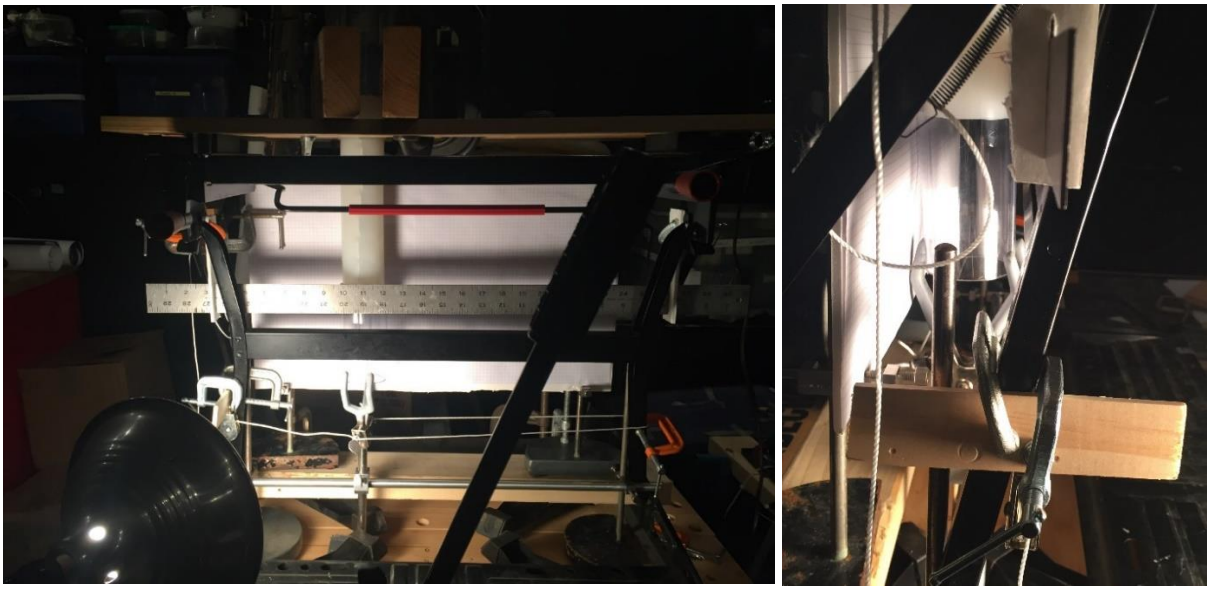


Figure 11. *The mechanical apparatus for testing the torsional stiffness of the body models. Left: A front view of the apparatus displays a yardstick in front of the vertebral column and a pulley system unique from the one used to test flexural stiffness. Right: A close-up view of the pulley system used to test torsional stiffness.*

Testing the torsional stiffness of the physical models required the apparatus to hold both ends of the vertebral column. While the grip holding the top end remained fixed in place, the grip holding the bottom end had to be able to turn about the long axis in order for the physical model to undergo torsion. This force was generated by a pulley system attached near the bottom of the workbench. Similar to the flexural stiffness test, a series of weights in the form of a dangling chain of clamps created the torsion force using the force of gravity. In addition, the weights were added to the pulley system incrementally with each addition to the load increasing the force turning the physical model. A video recorder placed on a tripod stand was positioned facing the front of the mechanical apparatus in order to capture the torsion of the vertebral column caused by each set of weights (Figure 11).

Similar to the gripper for the flexural stiffness test, the grippers holding the top end of the vertebral column for the torsional stiffness test were different for the body and skeleton models. In the case of the body models, the grippers were identical to the ones used for the flexural stiffness test. About 50 mm of the vertebral column from one end was placed a clear plastic tube, which was locked into place with the pressure generated by the wooden planks of the workbench platform and the wooden boards clamped onto the workbench platform. However, the torsional stiffness test required several modifications to the grippers for the skeleton models. In order for the vertebral column to reach the pulley system at the bottom, the three-finger clamp holding onto the first vertebra of the vertebral column was oriented to face downwards into the platform gap. In addition, two wooden boards were clamped onto the platform on opposite sides of the three-finger clamp to apply pressure onto the clamp to hold it in place (Figure 12).

Figure 12. *The mechanical apparatus for testing the torsional stiffness of the skeleton models. Left: The front view displaying the skeleton model with a three-finger clamp holding onto both sides of the vertebral column. Right: A three-finger clamp oriented downwards holds onto one end of the vertebral column as two wooden boards clamped onto the top platform of the work bench hold the clamp in place.*

In contrast to the flexural stiffness test, the torsional stiffness test had a gripper hold the bottom end of the dangling vertebral column. For the body models, the vertebral column was enclosed inside a plastic tube, and the three-finger clamp gripped the plastic tube (Figure 11). For the skeleton models, the three-finger clamp gripped the vertebra on the end of the vertebral column directly (Figure 12). In order for the vertebral column to undergo torsion, the three-finger clamp holding the bottom end of the physical model had to remained fixed in its position but exhibit enough flexibility to allow turning without any resistance. Therefore, two ring stands were clamped on top of a length of plywood placed through the bottom side holes of the workbench, and a long rod was affixed perpendicularly to the ring stands with the help of clamp holders. A clamp holder with a hole in one end and a C-shaped groove on the other end secured the three-finger clamp holding the bottom end of the vertebral column to the rod. The grooved end of the clamp holder was fastened tightly to the rod, while the handle of the three-finger clamp was placed through the hole end, allowing the three-finger clamp to turn but not move out of position. The pulley system also had a small wooden plank clamped onto the side of the workbench's metal scaffolding, while a different clamp affixed the handle of the pulley onto the wooden plank, much like the pulley system for the flexural stiffness test. However, the string rope in this pulley system was tied to both of the screw fasteners of the three-finger clamp gripping the vertebral column. First, a knot was tied to the screw fastener facing towards the video recorder before looping around the right scaffolding of the workbench and becoming tied to the screw fastener facing away from the video recorder. Thus, whenever the weights pulled on the string rope, the knots caused the three-finger clamp and the vertebral column to turn clockwise. In order to make sure that string rope wouldn't slip down the workbench scaffolding, a clamp was positioned directly underneath the string rope at the bend (Figure 11).

For each trial of flexural and torsional stiffness testing, the vertebral column was always oriented so that the cotyle pointed up and the condyle pointed down. In addition, the dorsal side of the vertebral column always faced the video recorder at the start of each trial. During the mechanical test, a total of seven clamps were added to the weight-loading end of the pulley system. Each clamp, labeled A to G, was always added in a predetermined order for each trial (Table 2).

Clamp	A	B	C	D	E	F	G
Mass (g)	104.1	105.3	109.9	110.1	110.1	105.6	106.5

Table 2. *The mass of the weights added to the pulley system for the mechanical tests. For each trial, the weights were added in order starting with Clamp A and ending with Clamp G.*

Flexural stiffness (in N * m^2) was calculated using the following formula, where F is the applied force, L is the length of the vertebral column, and y is the lateral displacement caused by the applied force:

$$Flexural\ stiffness\ (EI) = \frac{FL^3}{3y}$$

In this case, F represented the applied force (N), which was the product of the combined mass of the weights used in kilograms and the acceleration of gravity (9.8 m/s^2). L represented the beam length (m), the distance from the edge of the top gripper to the site where the string rope was wrapped around the vertebral column. This value was determined by counting the number of blocks in the backdrop when the vertebral column was at its initial position before any weights were added. The value was then converted from units of blocks to that of meters. However, this number only made up the visible portion of the beam length. In order to determine the total beam length, the visible beam length was summed with the unseen beam length above the backdrop and outside of the view of the camera. The unseen beam length was 45 mm across all trials. The variable y represented the arc length measured from the position of the knot tied around the vertebral column at the start of the trial to the position of the knot after Clamps A to G were added to the load. In order to calculate the arc length, the lateral displacement of the knot was first measured by counting the number of blocks traveled between the initial position and the final position. This was made possible with the use of screenshots taken from the video recording at different points in time and analyzed in the Paint 3D software. Then, the arc length was calculated with the following equation, where r represents the radius, i.e. the total beam length, and θ represents the angle of lateral movement:

$$arc\ length = \frac{\theta}{360 * 2\pi r}$$

The angle, θ, was calculated by using the arctangent, the total beam length, and the lateral displacement of the vertebral column. This is because the angle of a right

triangle can be determined by finding the arctangent of the side opposite of the angle over the side adjacent to the angle:

$$\theta = tan^{-1}\left(\frac{Opposite}{Adjacent}\right) = tan^{-1}\left(\frac{Lateral\ Displacement}{Total\ Beam\ Length}\right)$$

Once the value of all three variables were found, the value for flexural stiffness was calculated for each set of weights used. As a result, each vertebral column provided seven values for flexural stiffness, one for each weight load from A to G. Therefore, the average of these values served as the flexural stiffness of the vertebral column. In addition, the arc length produced by the applied force generated with each weight load was plotted on a spring stiffness graph, where the slope (k) represented the spring stiffness of the vertebral column.

Torsional stiffness (in N * m^2) was calculated with the following formula, where F is the force at moment arm d, θ is the angle of resulting rotation in radians, and L is the length of the vertebral column:

$$Torsional\ stiffness\ (GJ) = \frac{Fd}{\theta/L}$$

Similar to flexural stiffness, F represented the applied force and L represented the distance from the edge of the top gripper to the edge of the bottom gripper, both of which were determined with the same procedure used to find these values for the flexural stiffness calculation. The only difference was that the total beam length was visible within the scope of the video recorder and thus could be measured directly. However, the torsional stiffness equation features the variables d and θ. The variable d was found by measuring the distance from the knot of the string rope to the body of the three-finger clamp, which was the same throughout all the trials. The variable θ represented the angle of rotation observed, which first required obtaining the lateral displacement, i.e. the distance traveled by a point on the vertebral column as it turned, to calculate. In a fashion similar to finding the lateral displacement during the flexural stiffness calculation, screenshots of the video recording at different points in time were analyzed in Paint 3D to determine the lateral displacement. Specifically, this value was determined by using the yardstick in front of the vertebral column to measure the distance that the point traveled from the beginning of the experiment to the timepoint when each weight was added. The lateral displacement was then converted from units of blocks to meters. The angle of rotation was then measured by using the Law of Cosines, the radius of the vertebral column (r), and the lateral displacement (d). With the Law of Cosines, the cosine of any angle of a triangle can be determined if the lengths of the three sides (a, b, and c) of the triangle are known:

$$cos\ (A) = \frac{b^2 + c^2 - a^2}{2bc}, if\ angle\ A\ is\ between\ sides\ b\ and\ c$$

Therefore, the angle itself can be determined by calculating the arccosine:

$$\theta = \cos^{-1}\left(\frac{r^2 + r^2 - d^2}{2 * r * r}\right) = \cos^{-1}\left(\frac{2r^2 - d^2}{2r^2}\right) = \cos^{-1}\left(1 - \frac{d^2}{2r^2}\right)$$

Once the value of all four variables were found, the value for flexural stiffness was calculated for each set of weights used. As a result, similar to flexural stiffness, each vertebral column provided seven values for torsional stiffness, one for each weight load from A to G. Therefore, the average of these values served as the torsional stiffness of the vertebral column. In addition, the angle produced by the torque (applied force multiplied by the moment arm) generated with each weight load was plotted on a rotational stiffness graph, where the slope (t) represented the rotational stiffness of the vertebral column.

Once the flexural stiffness (EI) and the torsional stiffness (GJ) of the vertebral column was calculated, the twist-to-bend ratio was determined for each of the vertebral columns with the following formula:

$$Twist - to - Bend\ Ratio = \frac{EI}{GJ}$$

The Central Pattern Generator

The central pattern generator (CPG) produced the behavior that the physical model had to perform. Specifically, the purpose of the CPG was to move the physical models to perform lateral undulation. Initially, the CPG consisted of an Arduino Mega and four servo motors (SunFounder Metal Gear 55g Digital RC Motor) powered by a lithium-ion battery connected to two power distribution adaptors (Figure 13). The power distribution adaptors, in particular, were custom built by Vassar technician Phillip Cooper, and they served to safely dispense the power from the lithium-ion battery to the Arduino Mega and the servo motors.

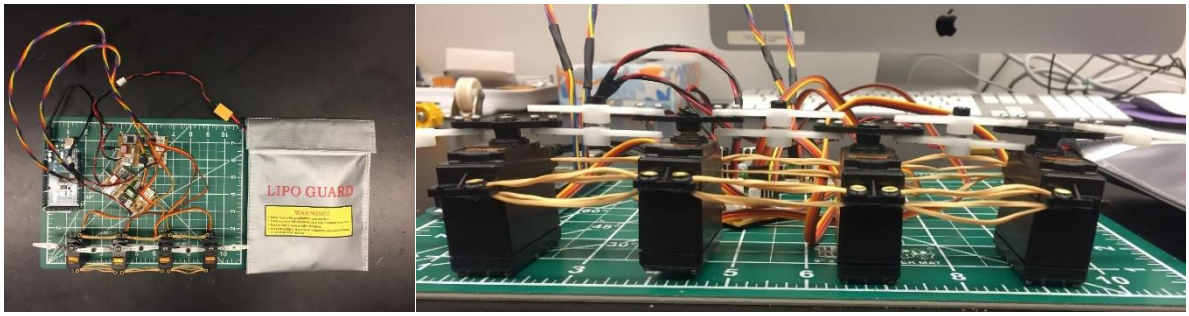


Figure 13. *The preliminary design for the CPG. Left: A top view depicting the Arduino Mega on the left, the four connected servo motors on the bottom, the two power distribution adaptors above, and the lithium-ion battery pack on the right. Right: A side view of the preliminary design for the CPG. Two sets of servo horn (in white) were screwed onto the servo horns attached to the servo motors (in black) in an alternating manner to keep the servo horns at the same height. Rubber bands were also attached to the sides of the servo motors in an effort to keep the servo motors together.*

In the early design, the servo motors were intended to be attached to the top of the vertebral column and cause it to move in an undulatory manner, much like how the servo motors controlled the movements of the modular snake robot (Wright et al., 2007). With that idea in mind, the four servos were connected with a flexible bridge made out of servo horns that could bend to allow that undulatory motion. Rubber bands were also used to connect the servos (Figure 13). However, after several revisions to the early design, the CPG was formally constructed to remain stationary and instead feature grippers that can grab onto different areas of the physical model and twist them based on the servo motor activity. In order to lock the servo motors into place, a base structure was built out of Lego pieces to hold the four servo motors. The dimensions of the base structure were 34 cm x 4cm x 5 cm, and the servo motors were spaced approximately 33 mm away from each other (Figure 14).

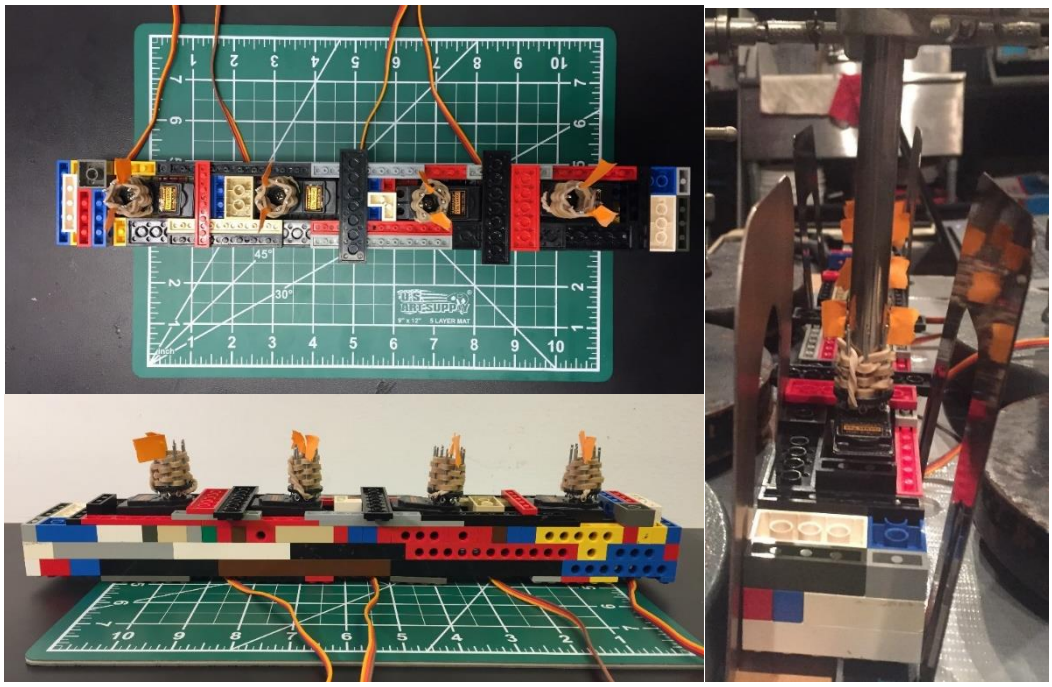


Figure 14. *The base structure of the CPG with the servo motors inside. Top Left: A top view of the base structure. Bottom Left: A side view of the base structure. Right: The base structure implemented in the final design with bookends holding it in place.*

By enclosing the servo motors with Lego pieces, the base structure allowed the servo motors to rotate the output spline, which pointed up in the air. On each of the servo motors, a 12-hole servo horn was screwed onto the output spline and 12 cotter pins (1.59 mm x 25.4 mm) were inserted through the holes of the servo horn in a symmetrical arrangement. A rubber band was then interwoven with the cotter pins on each of the servo horns so that the handle of a three-finger clamp would fit tightly in the makeshift clamp holder. The orange bits of tape marked the rotation position of the outline spline in order to make it easier to discern its orientation (Figure 14). The handle of the three-finger clamp was thus inserted into the opening of the modified servo horn

in a way so that the alignment of the fingers matched the placement of the tape. These three-finger clamps served to hold the vertebral column. Despite the existence of four servo motors, only the servo motors located at the ends of the base structure were used for the experiment. This was because, despite its connection to the power distribution adaptors, the Arduino Mega could not coordinate the movement of all four servo motors simultaneously without malfunctioning.

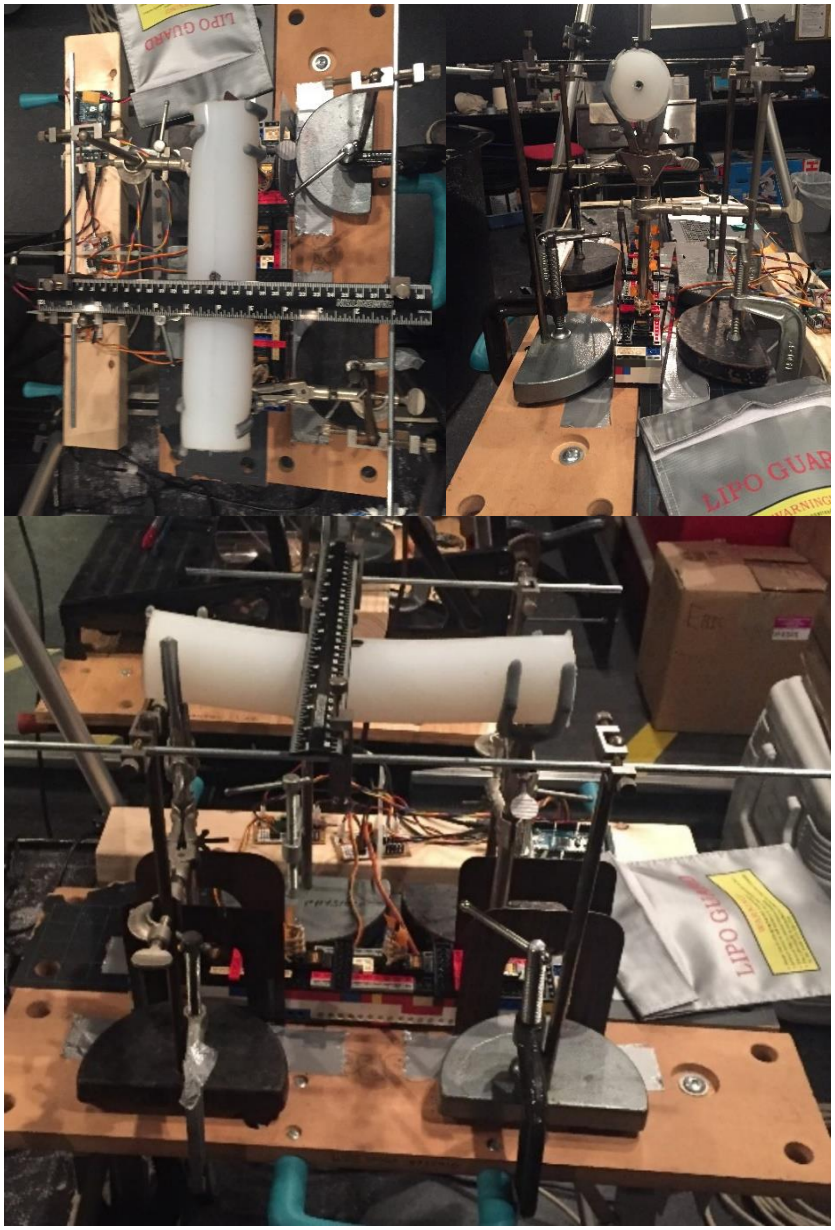


Figure 15. *The final design of the CPG. Top Left: A top view of the CPG. Top Right: A front view of the CPG. Bottom: An oblique view of the CPG.*

The base structure of the CPG was stationed on top of a Black+Decker Workmate 425 portable workbench, where two pairs of bookends pressed against the sides and taped onto the workbench to keep it in place. A ring stand was placed on top of the flat

portion of each bookend and clamped down on the workbench to prevent the CPG from shaking. Attached to each of the ring stands was a universal clamp that clasped the handle of a three-finger clamp holding up the physical model. However, while the universal clamp made sure that the three-finger clamp did not tip over, its grip was loose enough so that the three-finger clamp could rotate without any resistance. Above the universal clamp, a clamp holder gripped a long metal rod positioned parallel to the vertebral column. In the middle of the two rods above the CPG, a clamp holder held onto a ruler oriented perpendicularly to the vertebral column at the midpoint indicated by the black dot. Next to the base structure, the Arduino Mega and the two power distributor adaptors were affixed to the workbench with plastic tag fasteners (Figure 15).

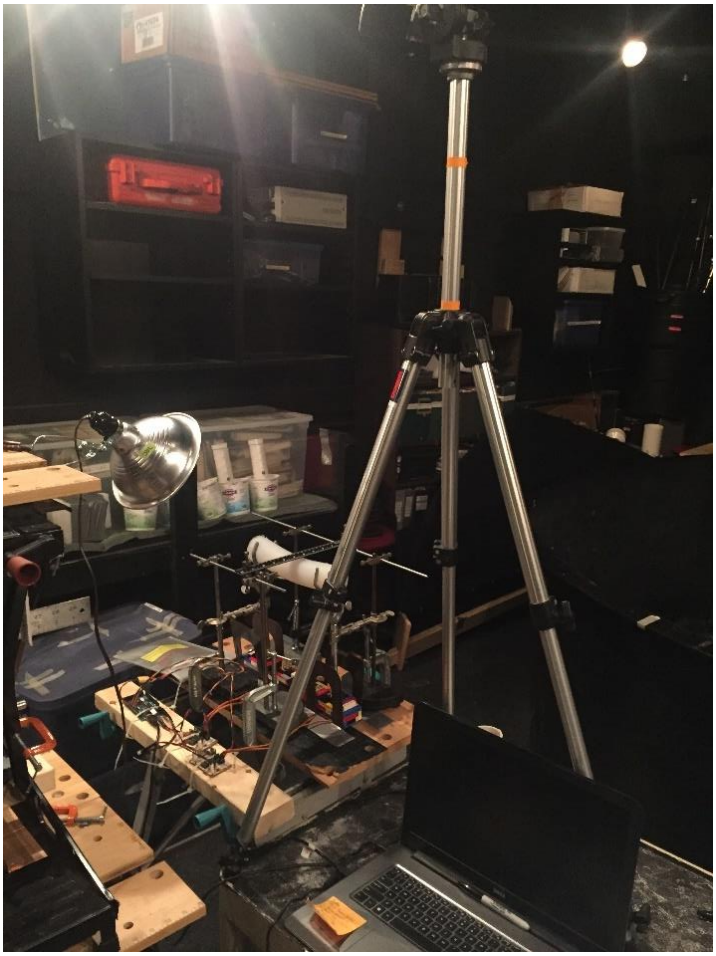


Figure 16. *A full view of the CPG on the workbench with the tripod positioned beside it. The video camera on top of the tripod is facing down to capture the lateral undulation of the vertebral column from the top view.*

For this experiment, the CPG was programmed to swivel the three-finger clamps holding up the vertebral column so that the vertebral column exhibited lateral undulation. In order to film this behavior from the top view, a large cart was clamped to the workbench and a tripod was placed on top of the cart with its legs fully extended. A video recorder was placed on top of the tripod with the lens facing downward (Figure

16). The vertebral column's lateral undulation was measured by tracking the lateral displacement of the black dot at the center during each cycle of undulation. Around 20 seconds of footage was captured for all nine vertebral columns. The video was then analyzed in Logger Pro 3, where the lateral displacement of the black dot was determined at six FPS. The program controlling the actions of the CPG was written in Arduino 1.8.8 and frequently tested in Tinkercad, a software program for 3D design, electronics, and coding (Figure 17).

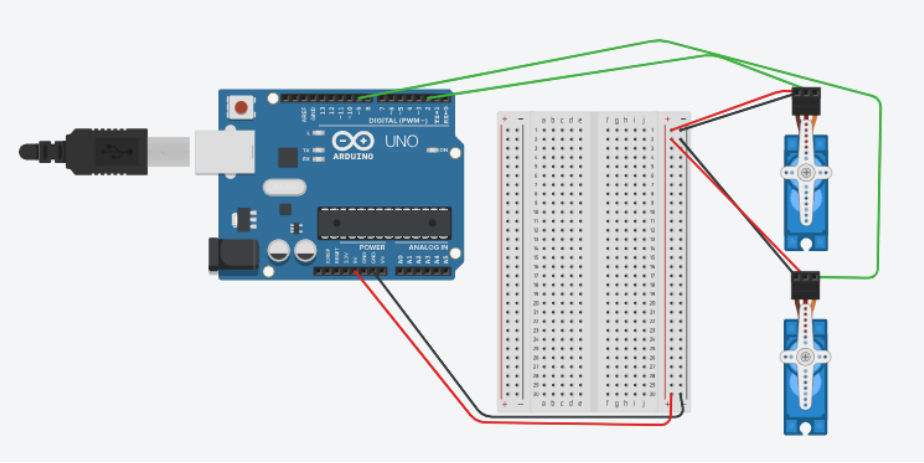


Figure 17*. A computer simulation of the breadboard wiring for the CPG in Tinkercad. Arduino pins 2 and 9 were used to send signals controlling the behavior of the servo motors. The code for the CPG was initially tested using this simulation before being implemented in the real CPG circuit.*

The CPG testing consisted of two rounds of trials for each vertebral column. During the first round, the CPG used the same code for all the vertebral columns in order to display how each of them behaved in response (See Appendix A). Afterwards, the first nine seconds of footage for each vertebral column was analyzed, starting at the beginning of an undulation cycle. The lateral displacement was then plotted in a graph to portray its patterns of movement. For each vertebral column, the amplitude of the curve of each cycle throughout the nine-second duration was measured and then averaged. Before the start of the second round, the code was modified for all of the vertebral columns, so that their amplitude matched that of the control, i.e. the complete morphology body model. Under ideal circumstances, the computation for the target behavior would be based on the length of the computer code generating that behavior. This way, one could compare the number of code lines needed to perform a particular behavior between the different vertebral columns. However, due to the nature of the target behavior for this experiment, only a simple value change is typically necessary for all the variants to match the amplitude of the control. Thus, in accordance to the spirit of this new approach to measuring morphological computation, a portion of the code was structured to allow for a standardized method of quantifying this behavior with lines of code (See Appendix D).

```
//Unit of information change
int lineUnit = 10;

//Variable to store angle range of servos
//Add lineUnit to setRotationAngle to increase angle of rotation
//Subtract lineUnit to setRotationAngle to decrease angle of rotation
int setRotationAngle = 120;
int setLeftRange = 90 - (setRotationAngle / 2);
int setRightRange = (setRotationAngle / 2) + 90;
```

Figure 18. *The section of the CPG code that determined the angle range of the servo motors. This version was used during the first round of CPG testing for all the vertebral columns.*

During the first round of CPG testing, the servo motors were programmed to rotate back and forth between 30 degrees and 150 degrees. This was done by setting the variable setRotationAngle to 120. However, after the first round, the code for each vertebral column was modified to match the amplitude of the control by adding the variable lineUnit to setRotationAngle. Each addition of lineUnit was designed to subtract 10 degrees to setRotationAngle and take up one line of code, thus creating a standard unit for this particular case (Figure 18). Through a process of trial and error, the second round of CPG testing consisted of modifying the code for each of the vertebral columns so that the lateral displacement came as close to possible to that of the control. Once that was achieved, the behavior of vertebral columns dictated by their respective programming code was filmed and undergone the same analysis process that occurred after the first round of CPG testing. In addition, the percent error in amplitude between the control and the variants during the second round as well as the percent difference between the amplitude found in the first round and that of the second round for each of the vertebral columns was calculated to determine how the number of additional code lines correlated with the change in behavior.

Microsoft Excel 16.0 (2016) was used for all data manipulation and statistical analysis.

Results

The Mechanical Test

The purpose of the mechanical test was to assess the validity of the assumption that the three sets of articulations that connect neighboring snake vertebrae together serves to mainly restrict the movements of the vertebral column. The flexural stiffness of the vertebral column varied greatly between the body models and the skeleton models (See Appendix B). While the body models were able to handle the load of all seven weights from A to G, the skeleton models quickly deformed once weight B was added to the load. As a result, flexural stiffness testing for the latter consisted of testing for only weight A. This meant that flexural stiffness could not be averaged across each weight and the spring stiffness, k, could not be determined for the skeleton models. Nevertheless, the collected data shows a clear difference in flexural stiffness between the body models and the skeleton models, demonstrating the large effect that the matrix has on the bending of the vertebral column (Figure 19).

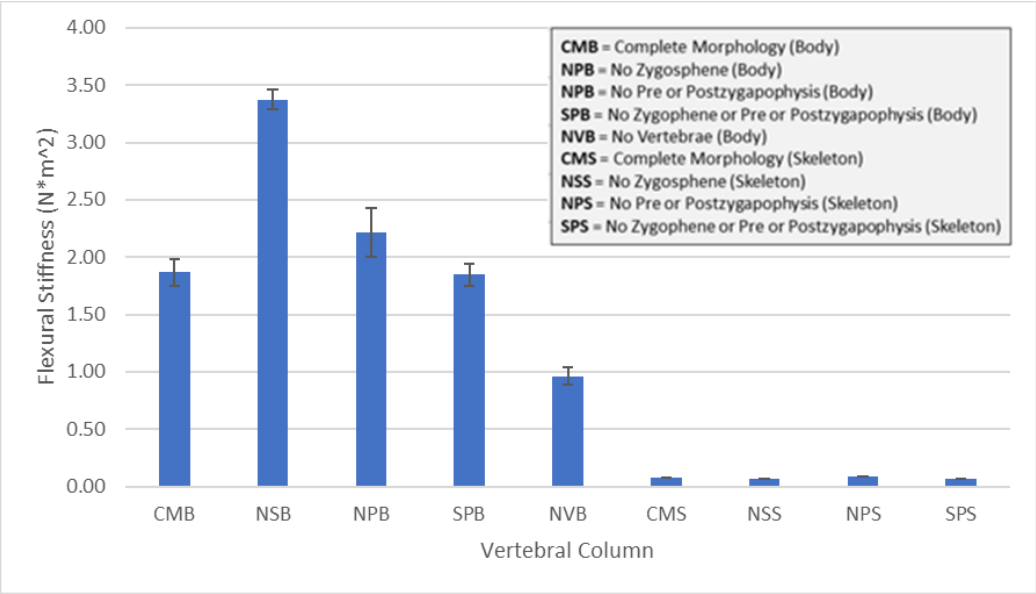


Figure 19. *Mean flexural stiffness for the nine vertebral columns. Error bars represent standard errors.*

Among the body models, CMB had a relatively low flexural stiffness (M = 1.87, SD = 0.308) compared to the other vertebral columns. Instead, NSB had the highest flexural stiffness (M = 3.37, SD = 0.227), followed by NPB (M = 2.22, SD = 0.566). However, SPB had a slightly lower flexural stiffness (M = 1.85, SD = 0.254) than CMB. By far, the body model with the lowest flexural stiffness was NVM (M = 0.963, SD = 0.214). In addition, NSB had a greater spring stiffness (k = 2.93 x 10^3) than CMB (k = 2.971 x 10^3), but CMB had a greater spring stiffness than either NPB (k = 1.60 x 10^3) or SPB (k = 1.27 x 10^3). NVB had the lowest spring stiffness value among the body models (k = 8.90 x 10^2) (Figure 20). Compared to the body models, the flexural stiffness of the

skeleton models was quite small, to the point where differences between each vertebra morphology were not clearly visible. NPS had the highest flexural stiffness among the skeleton models (M = 8.41 x 10^{-2}), followed by CMS (M = 8.11 x 10^{-2}), then NSS (M = 7.07 x 10^{-2}), and lastly SPS (M = 6.26 x 10^{-2}). This seems to suggest that, without the presence of the matrix, the bending of the models came largely from the properties of the extension spring holding the vertebrae together rather than the articulations themselves (Table 3).

Figure 20. *Relationship between applied force (N) and the arc length of the lateral displacement displayed by the body model. Stiffer models have a steeper slope: CMB (m = 2556), NSB (m = 2924), NPB (m = 1603), SPB (m = 1274), & NVB (m = 890).*

The torsional stiffness of the vertebral column was less distinct between the body and skeleton models as well as between vertebra morphology (Figure 21). However, compared to the values for flexural stiffness, the values for torsional stiffness were generally smaller. Among the body models, NSB had the highest torsional stiffness (M = 0.113, SD = 2.79 x 10^{-2}), while NPB (M = 3.93 x 10^{-2}, SD = 1.02 x 10^{-2}) and SPB (M = 4.05 x 10^{-2}, SD = 7.70 x 10^{-3}) had around the same value. In contrast, CMB had a relatively low value for torsional stiffness (M = 3.33 x 10^{-2}, SD = 7.13 x 10^{-3}), but NVB had the lowest value among the body models (M = 5.41 x 10^{-3}, SD = 9.48 x 10^{-4}). Among the skeleton models, NSS exhibited a surprisingly high torsional stiffness (M = 6.19 x 10^{-2}, SD = 2.54 x 10^{-2}) that was greater than that of all the other vertebral columns except for NSB. However, given that the torsional stiffness for all the vertebral columns are very small, this difference may be caused by discrepancies in measurement. Following NSS, NPS (M = 1.43 x 10^{-2}, SD = 4.98 x 10^{-3}) had the next highest torsional stiffness, then CMS (M = 9.22 x 10^{-3}, SD = 3.85 x 10^{-3}), and lastly SPS (M = 2.77 x 10^{-3}, SD = 1.08 x 10^{-3}) (Table 3). Notably, in some cases with the earlier weights, the torsional stiffness was infinity, indicating that the vertebral column with the given weight did not rotate. These values were not included in the calculations.

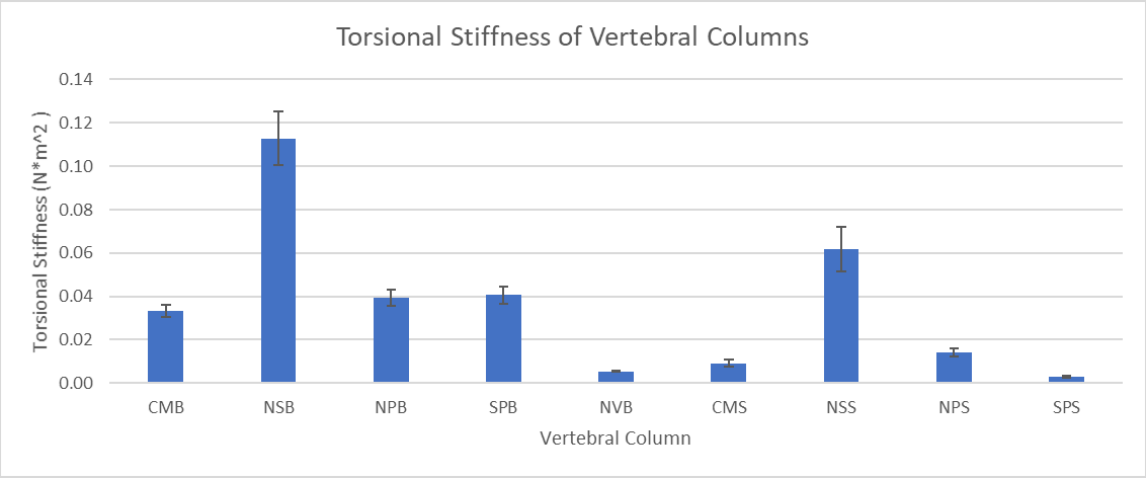


Figure 21. *Mean torsional stiffness for the nine vertebral columns. Error bars represent standard errors.*

For the rotational stiffness of the vertebral columns, the body models showed less distinct differences between vertebra morphology than the skeleton models. Among the body models, NVB stood out from the other vertebral columns (t = 3.77 x 10^{-2}), while CMB (t = 0.191), NSB (t = 0.415), NPB (t = 0.282), and SPB (t = 0.148) were largely clustered together. However, it is noteworthy that the rotational stiffness for NSB and NPB was greater than that of CMB (Figure 22). For the skeleton models, the vertebral columns displayed a more distinct difference in rotational stiffness (Figure 23). Similar with the body models, SPS had the lowest rotational stiffness within its respective body composition group (t = 1.44 x 10^{-2}). In addition, NSS (t = 9.63 x 10^{-2}) and NPS (t = 8.43 x 10^{-2}) had a higher rotational stiffness than CMS (t = 5.32 x 10^{-2}) (Table 3).

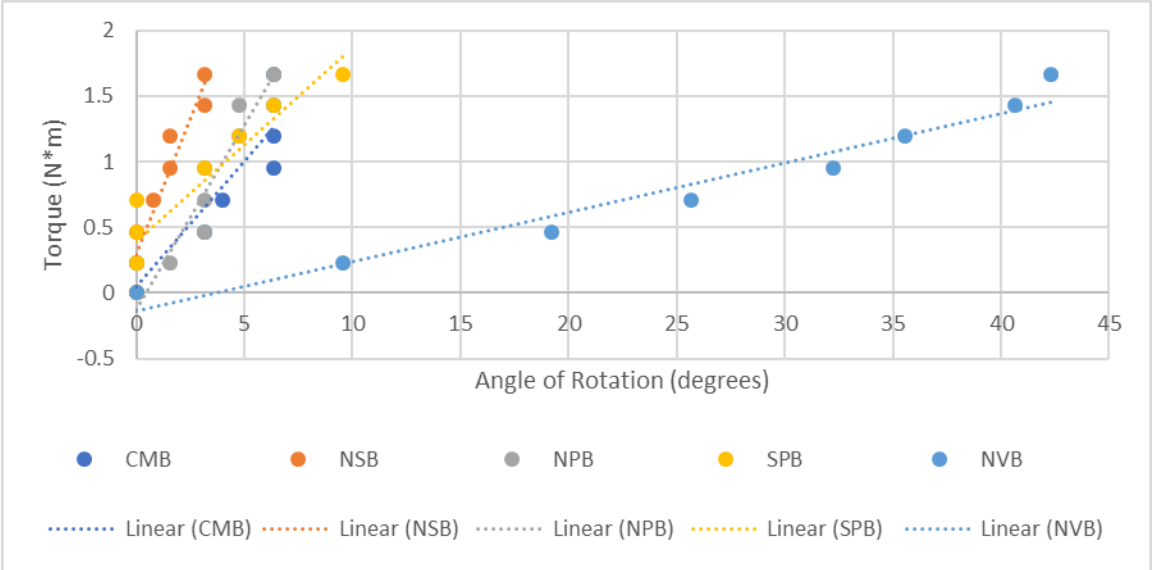


Figure 22. *Relationship between applied force (N) and the angle of rotation displayed by the body models. Stiffer models have a steeper slope: CMB (m = 0.192), NSB (m = 0.415), NPB (m = 0.282), SPB (m = 0.148), & NVB (m = 0.0377).*

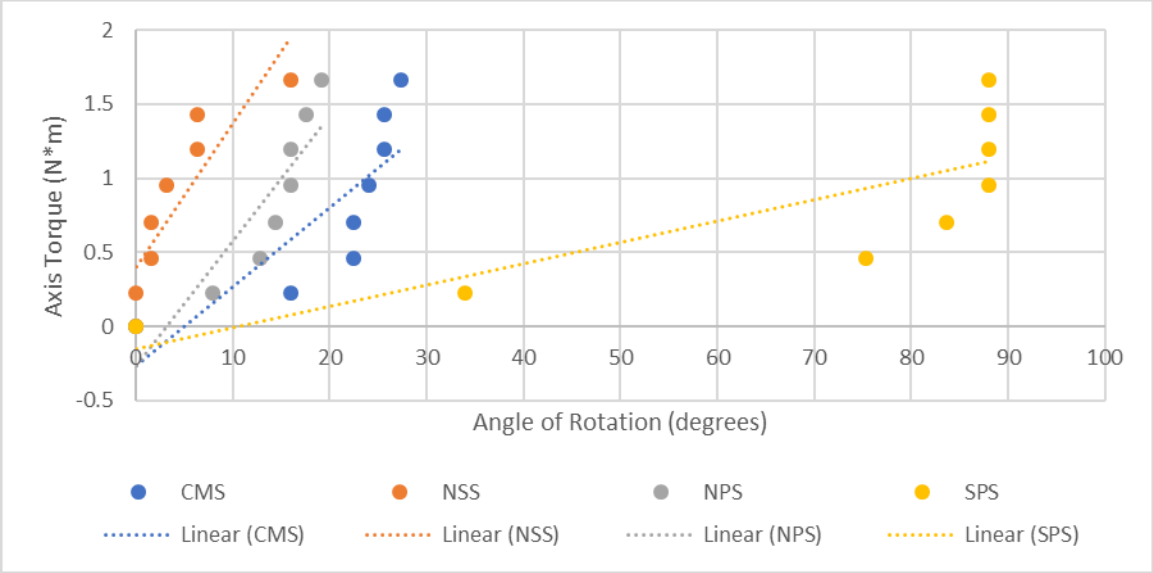


Figure 23. *Relationship between applied force (N) and the angle of rotation displayed by the skeleton models. Stiffer models have a steeper slope: CMS (m = 0.0532), NSS (m = 0.0963), NPS (m = 0.0843), & SPS (m = 0.0144).*

Once the flexural stiffness and torsional stiffness was determined for all the vertebral columns, the twist-to-bend ratio was calculated. In general, the body models had a higher twist-to-bend ratio than the skeleton models, signifying the body models' aversion to torsion compared to bending (Figure 24). Among the body models, NVB had the highest twist-to-bend ratio (M = 178, SD = 75.7), followed by a close tie between NPB (M = 56.4, SD = 37.0) and CMB (M = 56.1, SD = 10.4), then SPB (M = 45.5, SD = 6.23), and lastly NSB (M = 29.9, SD = 6.73). In contrast, SPS had the highest twist-to-bend ratio among the skeleton models (M = 22.6), followed by CMS (M = 8.80), then NPS (M = 5.90), and lastly NSS (M = 1.14) (Table 3).

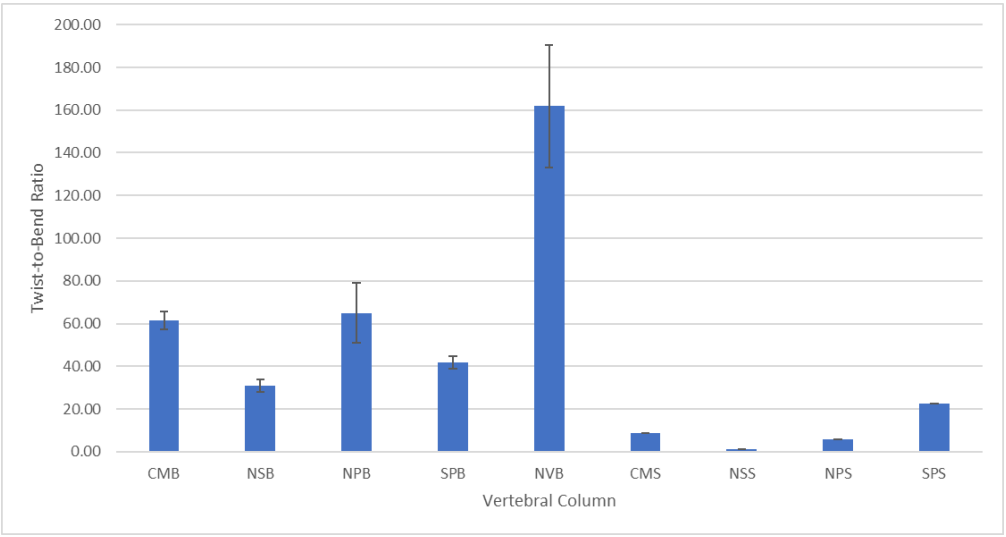


Figure 24. *Mean twist-to-bend ratio for all nine vertebral columns. The twist-to-bend ratio for the body models was first determined for each weight load before they were averaged across all the weights for each vertebral column. For the skeleton models, due to the inability to gather more values for flexural stiffness, the EI value obtained with Clamp A was used as the numerator instead of the average flexural stiffness. Error bars represent standard errors.*

Model	Flexural Stiffness ($N*m^2$)	Spring Stiffness	Torsional Stiffness ($N*m^2$)	Rotational Stiffness	Twist-to-Bend Ratio
CMB	1.87 (0.308)	2.71×10^{3}	3.33×10^{-2} (7.13×10^{-3})	0.191	56.1 (10.4)
NSB	3.37 (0.227)	2.93×10^{3}	0.113 (2.79×10^{-2})	0.415	29.9 (6.73)
NPB	2.22 (0.566)	1.60×10^{3}	3.93×10^{-2} (1.02×10^{-2})	0.282	56.4 (37.0)
SPB	1.85 (0.254)	1.27×10^{3}	4.05×10^{-2} (7.70×10^{-3})	0.148	45.5 (6.23)
NVB	0.963 (0.214)	8.90×10^{2}	5.41×10^{-3} (9.48×10^{-4})	3.77×10^{-2}	178 (75.7)
CMS	8.11×10^{-2}	0.00	9.22×10^{-3} (3.85×10^{-3})	5.32×10^{-2}	8.80
NSS	7.07×10^{-2}	0.00	6.19×10^{-2} (2.54×10^{-2})	9.63×10^{-2}	1.14
NPS	8.41×10^{-2}	0.00	1.43×10^{-2} (4.98×10^{-3})	8.43×10^{-2}	5.90
SPS	6.26×10^{-2}	0.00	2.77×10^{-3} (1.08×10^{-3})	1.44×10^{-2}	22.6

Table 3. *Values for flexural stiffness, spring stiffness, torsional stiffness, rotational stiffness, and the twist-to-bend ration for all the vertebral columns. The standard deviation for each parameter, if present, is given in the parentheses.*

The Central Pattern Generator

The purpose of the CPG test was to first observe the lateral undulation of all the vertebral columns and then determine how many lines of code needed to be added in order for the lateral displacement of the variants to match that of the control. Lateral undulation was characterized primarily by the amplitude of the wave cycles generated by the CPG. Given how the stiffness of the vertebral columns directly affects the amplitude of the wave, behavior demonstrated by the vertebral columns helped bolster the results obtained during the mechanical test. For the first round of testing, all the vertebral columns undulated in a consistent, steady pattern with a similar frequency (See Appendix C). However, a noticeable difference could be observed in the behavior between the body models and the skeleton models. In general, the waves generated by the skeleton models exhibited a larger amplitude than those generated by the body models (Figure 25). Among the body models, NVB demonstrated the largest amplitude ($M = 4.07$, $SD = 7.69 \times 10^{-2}$), followed by SPB ($M = 2.65$, $SD = 6.15 \times 10^{-2}$). Interestingly, the amplitude for NSB ($M = 1.05$, $SD = 9.49 \times 10^{-2}$) was smaller than that of CMB ($M = 1.23$, $SD = 6.63 \times 10^{-2}$), although CMB had a smaller amplitude than NPB ($M = 1.91$, $SD = 9.85 \times 10^{-2}$). In contrast, many of the skeleton models exhibited similar amplitude measurements. Unlike with the body models, CMS had a smaller amplitude ($M = 5.16$, $SD = 0.139$) than either NSS ($M = 5.24$, $SD = 0.235$) or NPS ($M = 5.87$, $SD = 0.272$), but it was still larger than that of SPS ($M = 3.83$, $SD = 0.255$) (Table 4).

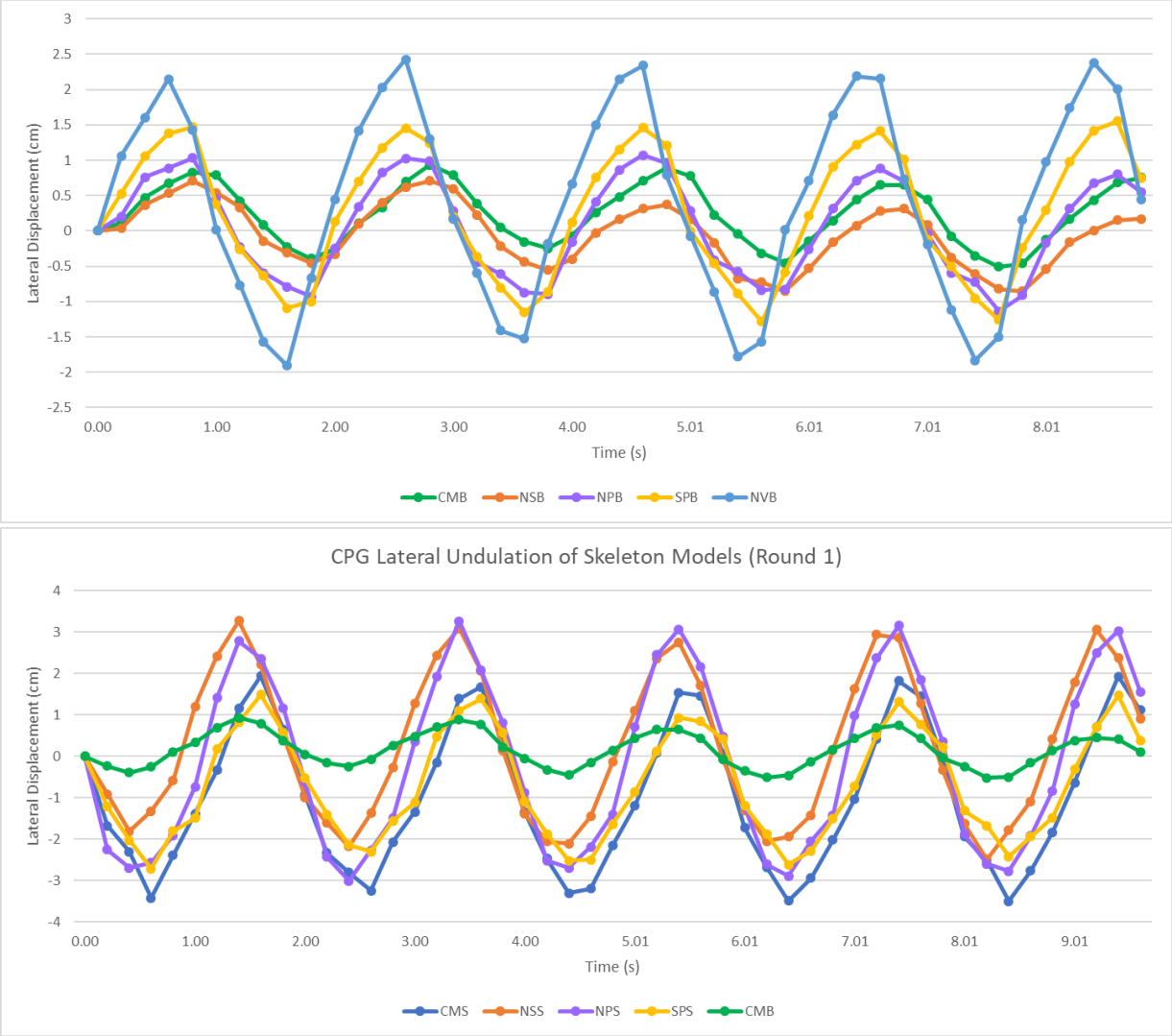


Figure 25. *Lateral Displacement of the vertebral columns for the first round over a period of around 9 seconds. Top: Line graphs that depict the lateral displacement of the five body models. Bottom: Line graphs that depict the lateral displacement of the four skeleton models and the CMB vertebral column (in green) as reference.*

Before the start of the second round of CPG testing, the program running the CPG for each of the vertebral columns were modified appropriately to match the behavior of the control. Modifications came in the form of the addition of the code line lineUnit, which subtracted 10 degrees from the variable setRotationAngle (See Appendix D). No other changes were made to the rest of the program. As the control, the program for CMB remained unchanged for its second round of testing with no lineUnits added. As an outlier due to having a smaller amplitude than that of the control, one lineUnit was added to the program for NSB, but the value of lineUnit was changed so that 10 degrees was added to setRotationAngle instead of subtracting 10 degrees from the variable. For all the other body models, lineUnit served to subtract 10 degrees from setRotationAngle. NPB required four lineUnits, SPB required six lineUnits, and NVB required eight lineUnits. On the other hand, the skeleton models displayed such large amplitude compared to the control that either nine lineUnits (NSS and SPS) or ten lineUnits (CMS and NPS) had to be added to their respective program (Table 4).

However, because only the amplitude of the undulations was altered in the code, variants showed a greater frequency during the second round of CPG testing compared to the first round in response. As a result, the second round featured more chaotic patterns of lateral undulation as the servo motors rotated at a faster rate (Figure 26).

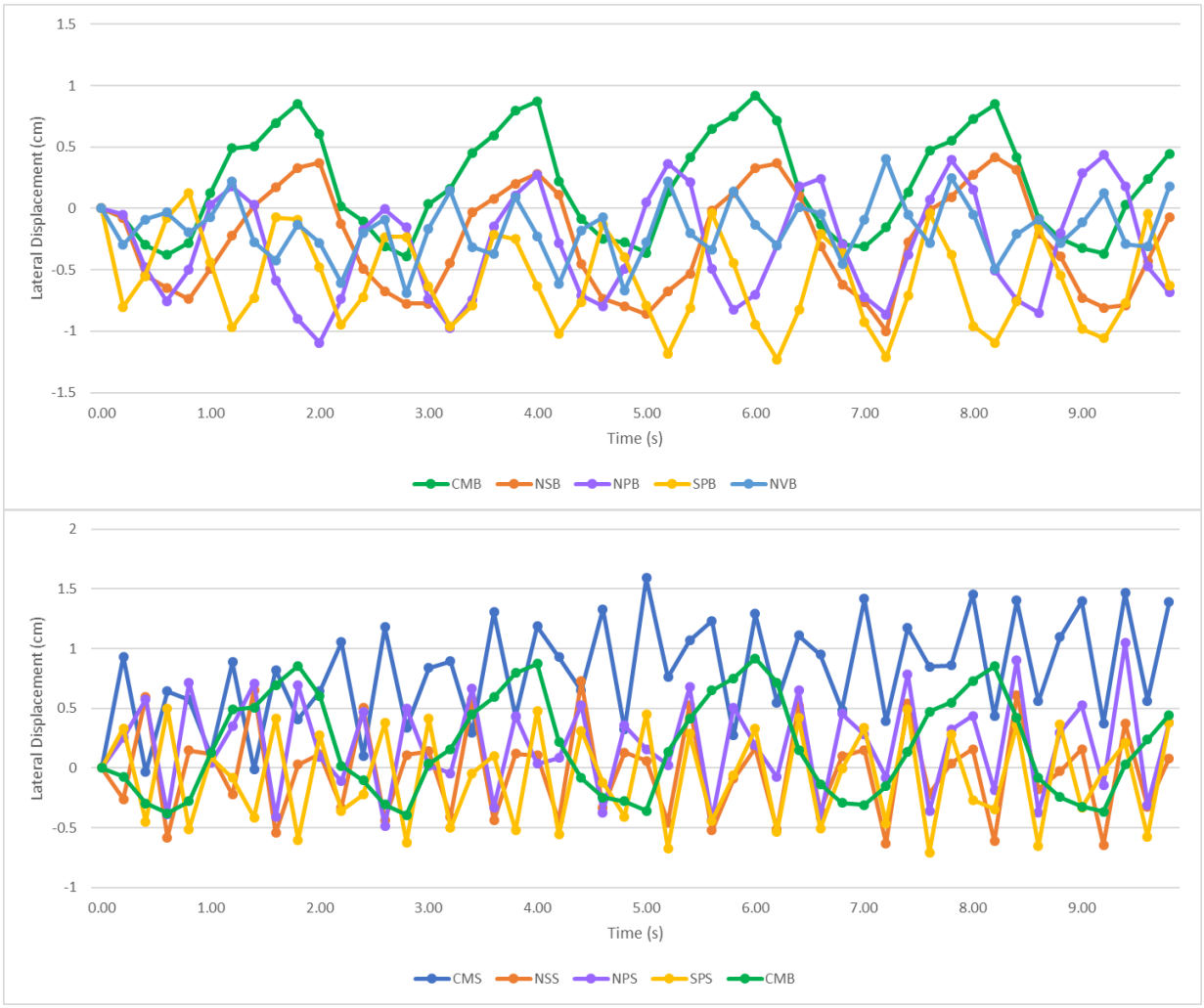


Figure 26. *Lateral Displacement of the vertebral columns for the second round over a period of around 9 seconds. Top: Line graphs that depict the lateral displacement of the five body models. Bottom: Line graphs that depict the lateral displacement of the four skeleton models and the CMB vertebral column (in green) as reference.*

The main goal of the second round of CPG testing was to ensure that the modification to the program brought the behavior of the variants as close to that of the control as possible. After the second round, the amplitude of the waves for each of the vertebral columns was averaged over their respective number of cycles within the nine-second duration. In making sure that the amplitude of the variants did not exceed that of the control, the control ended up having the largest amplitude out of all the vertebral columns for the second round (M = 1.23, SD = 6.43 x 10^{-2}). NSB (M = 1.17, SD = 0.127), NPB (M = 1.15, SD = 0.118), and SPB (M = 0.946, SD = 0.140) came closest to matching the amplitude of the control with a percent error of 4.26%, 6.22%, and 22.8%, respectively. However, all the skeleton models displayed a percent error of

approximately 30% in amplitude compared to the control, where CMS had a percent error of 33.1% (*M* = 0.819, *SD* = 0.216), NSS had a percent error of 30.8% (*M* = 0.848, *SD* = 0.196), NPS had a percent error of 36.9% (*M* = 0.772, *SD* = 0.275), and SPS had a percent error of 31.8% (*M* = 0.836, *SD* = 0.191). The variant with the largest percent error was NVB, which had a percent error of 62.0% (*M* = 0.465, *SD* = 0.196) (Table 4). While not identical to the behavior of the control, the modifications to the program brought each of the vertebral columns within a reasonable range of the control's observed amplitude. In addition, this data seems to suggest that the vertebral columns whose behavior deviated the most from the control during the first round of CPG testing were the most difficult to adjust for the second round of CPG testing (Figure 27).

Model	Round 1: Ave. Amplitude (cm)	Round 2: Ave. Amplitude (cm)	Round 2: % Error between Control & Variants	% Diff. between Round 1 & 2	No. Code Lines Added
CMB	1.23 (6.63 x 10^{-2})	1.23 (6.43 x 10^{-2})	0.00%	0.790%	0
NSB	1.05 (9.49 x 10^{-2})	1.17 (0.127)	4.26%	10.6%	1
NPB	1.91 (9.85 x 10^{-2})	1.15 (0.118)	6.22%	50.0%	4
SPB	2.65 (6.15 x 10^{-2})	0.946 (0.140)	22.8%	94.8%	6
NVB	4.07 (7.69 x 10^{-2})	0.465 (0.196)	62.0%	159%	8
CMS	5.16 (0.139)	0.819 (0.216)	33.1%	145%	10
NSS	5.24 (0.235)	0.848 (0.196)	30.8%	148%	9
NPS	5.87 (0.272)	0.772 (0.275)	36.9%	150%	10
SPS	3.83 (0.255)	0.836 (0.191)	31.8%	128%	9

Table 4. *Values for the amplitude during the first and second round of testing, the percent error between the control and the variants for round 2, the percent difference between the amplitude for round 1 and round 2, and the number of code lines that was added to the program for each respective vertebral column. The standard deviation for each parameter, if present, is given in the parentheses.*

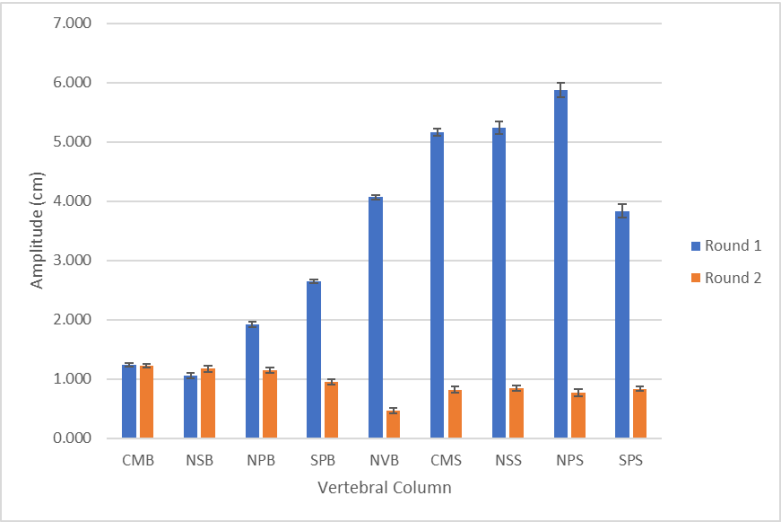


Figure 27. *Mean amplitude of the vertebral columns for round 1 of CPG testing (in blue) and round 2 of CPG testing (in orange). Error bars represent standard errors.*

In order to determine how much of the computational load that each vertebra morphology and the matrix offloaded from the abstract controller, i.e. the morphological computation of each morphological feature, the number of lineUnits added to the program and the corresponding percent difference in behavior between the first and second round were analyzed in context of one another. A graph depicting the percent difference in each vertebral column's amplitude between the first and second round of CPG testing in the context of the number of lineUnits was added to the respective CPG program showed a general pattern where the percent difference in behavior demonstrated a positive correlation with the number of code lines added (Figure 28). In other words, the removal of morphological characteristics from the vertebral column led to an observable difference in behavior. In addition, the more morphological features were removed from the vertebral column, the more lines of code that the CPG program required in order to replicate the behavior of a vertebral column with a complete morphology with no missing characteristics.

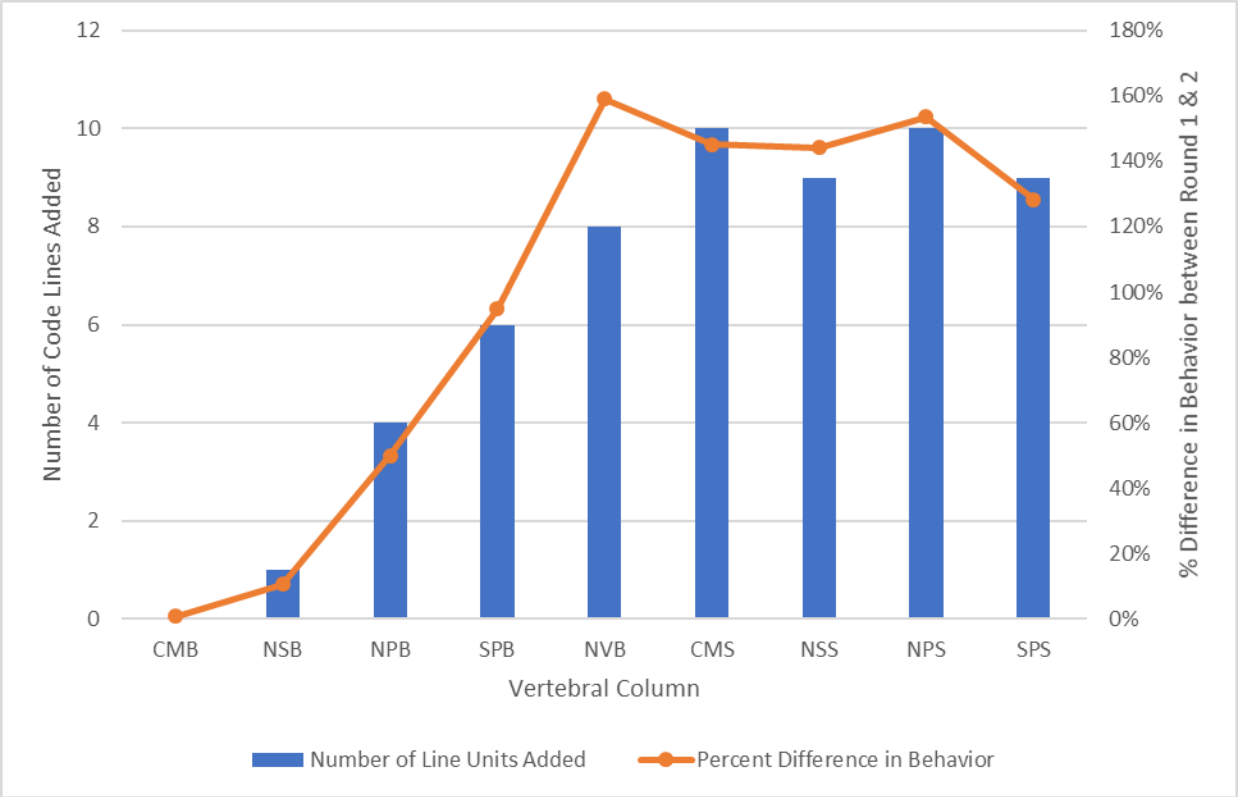


Figure 28. *The number of lineUnits added to the CPG program of each respective vertebral column (in blue) alongside the percent difference in observed amplitude that occurred between the first round of CPG testing and the second round of CPG testing (in orange).*

Discussion

Based on the results of this experiment, several insights can be made about the vertebral column of the snake and the approach to measuring morphological computation proposed by this study. Among the various findings, one that stood out in particular was the relative stiffness of the vertebral column with the complete morphology compared to the other variants. Due to the very nature of how the articulations restrict certain movements between neighboring vertebrae, this experiment was conducted under the pretense that removing these articulations would reduce the overall stiffness of the vertebral columns. But while the complete morphology models generally demonstrated more stiffness than the models with both the zygosphene-zygantrum and prezygapophysis-postzygapophysis articulations removed, the former often exhibited less stiffness compared to models where only one of those articulations were missing. This discrepancy was prevalent primarily among the body models. During the mechanical test, CMB exhibited a much lower flexural stiffness and torsional stiffness than both NSB and NPB. In addition, NSB displayed a greater spring stiffness and rotational stiffness than CMB. This pattern was also observed during the CPG test, where NSB exhibited a smaller amplitude in its undulation than CMB.

Based on the existing literature, it seems implausible that the presence of the zygosphene or the zygapophyses would, in fact, increase the vertebral column's flexibility rather than decrease it. The zygosphene and the zygantrum are morphological traits that are unique to certain reptiles and highly developed in snakes. Past studies have shown that the opposing nature of the zygosphene-zygantrum articulations renders torsion wholly impossible, or at the very least severely limit it in snakes. Although Moon (1999) has argued that even incremental twisting can produce substantial torsion over a long stretch of the vertebral column, there is no indication that actually removing the zygosphene-zygantrum articulation would hinder torsion in any way. Similarly, the prezygapophysis-postzygapophysis articulation has been proposed to not only severely limit torsion but also lateral bending at each intervertebral joint (Morinaga & Bergmann, 2019). Any additive effect that these articulations may have to allow a wider range of movement is left largely irrelevant when comparing the behavior of vertebral columns with the same number of vertebrae.

Therefore, it is possible that the relatively low stiffness of CMB compared to NSB and NPB could be due to the silicone rubber matrix encasing the vertebral column. During the creation of the body models, the Dragon Skin silicone rubber solution was poured into the mold from the top, allowing the solution to seep through the holes between the vertebrae of the vertebral column inside the plastic tube. However, the zygosphene-zygantrum articulation creates a small pocket of air where the joints meet, since it may not be easy for the silicone rubber solution to fully infiltrate the wedge-shaped cavity. Thus, there is a possibility that CMB has small pouches of air inside the cavity of the zygosphene-zygantrum articulations that contribute to its flexibility, while

the silicone rubber solution fully seeped between the vertebrae of NSB, contributing to its relatively high stiffness. If this is the case, then the skeleton models should more closely exhibit the expected patterns of stiffness between the different vertebral columns. However, the results for the skeleton models largely show the same pattern found with the body models. For both flexural stiffness and torsional stiffness, CMS was not the vertebral column that demonstrated the greatest stiffness among the skeleton models. In addition, not only do the skeleton models lack the same amount of observable data as the body models in terms of flexural stiffness, but there is also the likelihood that their lateral displacement could have been skewed by the properties of the extension spring that held the vertebrae together. Unfortunately, a multitude of factors could have influenced the results of the mechanical test, such as possible wear and tear on the vertebral columns.

Although it remains unclear as to what may have caused the complete morphology models to have a lower flexural and torsional stiffness compared to the variants in their respective body composition groups, the data shows that the stiffness between the vertebral columns are distinct. In most cases, the differences in the results were visibly perceptible, meaning that changes to the shape of the vertebra led to explicit changes in their behavior within the vertebral column. This finding reflects how snake species in different ecological niches have evolved vertebra with different morphological characteristics. For instance, researchers have found that the vertebrae of aquatic species have greater bone mass than the average snake in order to swim at the bottom of the water. Likewise, heavy-bodied strikers like most pythons have vertebrae with greater bone mass in order to anchor their bodies to the ground when attacking prey (Houssaye, Boistel, Böhme, & Herrel, 2013). According to a study by researcher Ralph Gordon Johnson, the differences in the shape of the vertebra between different snake species are so significant that he proposed that the vertebra morphology better reflect the taxonomic relationships of snakes than gross adaptation (Johnson, 1955).

In addition, the results of the mechanical test clearly demonstrate the difference in behavior caused by the existence of an extracellular matrix. For both the flexural stiffness and torsional stiffness tests, the skeleton models generally displayed less stiffness than the body models. In particular, the flexural stiffness of the skeleton models was so low compared to the body models that the value for flexural stiffness for each of them were on different orders of magnitude. This result clearly indicates the importance of the musculature surrounding the vertebral column in absorbing the strain and stress that would otherwise impact the skeleton directly. Lastly, the twist-to-bend ratio calculated for each vertebral column displays whether the model was more prone to bending or twisting. Interestingly, despite demonstrating a much higher torsional stiffness than the skeleton models, the body models also had a greater twist-to-bend ratio. This result seems to suggest that, at least with the matrix, the vertebral column is incredibly averse to torsion. However, as Moon (1999) indicates, the articulations do not outright make torsion impossible for the vertebral column. In

addition, given how NVB displayed the largest twist-to-bend ratio out of all the other vertebral columns as well as the very low twist-to-bend ratios for the skeleton models, it appears the matrix was also largely responsible for having the vertebral columns favor bending over twisting.

In regard to testing the effectiveness of the new approach to measuring morphological computation, the results seem to endorse its viability. During the first round of CPG testing, each of the vertebral columns displayed distinct differences in behavior, as measured by the amplitude of their undulatory wave. Furthermore, as shown in Figure 27, there was a positive correlation between the number of code lines needed and the size of the gap in behavior between the first round and the second round. For variants like NSB and NPB where only one or two morphological traits were missing from the control design, the differences in the observed lateral undulation were relatively minimal and required only a few lines of code to adjust the behavior and close the gap in amplitude. In contrast, variants like NVB were so morphologically different that a lot more lines of code were needed to close the gap in amplitude. Based on this graph, the presence of the matrix was so influential to the performance of lateral undulation that it overshadowed the effects of morphological differences in vertebra shape. All four of the skeleton models exhibited lateral undulations with much greater amplitudes than the control due to the lack of this matrix and thus required 8 to 9 additional lines of code to adjust the behavior, independent of the vertebra morphology. Additionally, the behavior of the variants during the second round of testing was more or less within the parameters of the control, as shown in Figure 26. In other words, the unique modifications to the CPG program for each of the variants successfully adjusted the behavior of the variants to match that of the control.

Based on the collected data, we can quantify the morphological computation of each of the morphological features of the vertebral column. Thus, the presence of the zygosphene would be equivalent to about one line of lineUnit code, the presence of the prezygapophysis and the postzygapophysis would be equivalent to about four lines of lineUnit code, and the presence of the extracellular matrix would be equivalent to about ten lines of lineUnit code. A noteworthy outcome of this conclusion is the fact that the zygosphene is implied to have offloaded the equivalent of only one lineUnit of computation from the abstract controller even though the results of the mechanical test suggests that the zygosphene plays a major role in deciding the stiffness of the vertebral column. This could be an instance where, even though the morphology greatly influences how the body responds to outside stimuli, it still plays a relatively small role in the computation that produces the behavior.

However, the results of this experiment should be interpreted with caution due to potential sources of error inherently present within the methodology of this particular experiment. For instance, the measurements during the mechanical tests were all done by eye, although the behavior was carefully filmed and analyzed in Paint 3D afterwards. For both flexural stiffness and torsional stiffness, the lateral displacement and height of

the vertebral columns relied on the observer properly keeping track of the models' movement and position as weights were added to the pulley system. In addition, the vertebral columns exhibited very small changes in behavior for many cases, meaning that any error in measuring that behavior greatly affected the value of the output.

Most importantly, several adjustments could be made to the CPG testing to improve its precision and accuracy. While the percent error between the behavior of the control and the variants during the second round of testing was mostly under 50%, an ideal scenario would have the percent error reduced to less than 10%. This issue in particular stemmed from the increase in frequency that resulted from adjusting the amplitude of the variants. Due to the increased speed of their movements during the second round, tracking the lateral displacement of the variants proved to be incredibly difficult and likely caused some errors in measurement. This issue could potentially be addressed by modifying the program for each vertebral column to adjust not only for the amplitude but also for the frequency of its lateral undulation. Another unexpected issue that arose was the shape of the lateral undulation for the skeleton models. While the presence of the matrix helped distribute the force across the vertebral column, which caused the body models to bend naturally, the skeleton models had a more deformed lateral undulation pattern due to the lack of a proper medium that could evenly transmit the force. While right side of the lateral undulation produced a distinct curve, the left side did not (See Appendix C). This issue further highlights the disruptive influence of the extension spring, whose effects may overshadow that of the articulations of vertebrae. In the future, a more rigorous methodology with a more precise protocol to measuring the behavior would lead to more accurate results.

Besides accounting for errors in behavior measurement, this approach to quantifying morphological computation must address perhaps its greatest obstacle, which is its limited applicability. The core structure of this method relies on our ability to not only physically model the target behavior but also ascertain that changes to the target behavior can be attributed to a specific, isolated cause or morphology. While this experiment naturally ascribed the amplitude of the model's lateral undulation to the vertebral column's articulations and body composition, there is a possibility that implementing these modifications altered some other aspect of the morphology that offloaded the computation instead of the traits that we had assumed. Not only that, the lateral undulation behavior was performed while the vertebral column was held up in the air, completely divorced from the environment with which the body was supposed to interact. Part of what allows snakes to propel themselves using lateral undulation is their use of irregularities in the ground, such as depressions, pebbles, and tufts of grass, push their bodies against to move forwards (Gans, 1970). Therefore, the implementation of the initial CPG design, where the servo motors lined the dorsal side of its body and controlled its movements on the ground, would have produced a more accurate representation of the behavior that we wanted to observe.

In essence, the success of this method hinges on the accuracy of the physical model, if one could even be built to model the target system. Furthermore, the computer program that controls its behavior must likewise demonstrate a high level of accuracy in its representation of the hypothesis for that behavior. Given the sheer amounts of complexity that biological systems possess, fulfilling these two crucial prerequisites for the successful implementation of this proposed approach can be expensive and difficult to fabricate, as well as impractical in some scenarios. This is especially true when one also has to account for how the target system interactions with the environment as a necessary component of embodied cognition.

According to roboticist Barbara Webb, biological models that serve to represent a hypothesis about a target system can be assessed in seven different ways (Webb, 2001):

1. Relevance: whether the model tests and generates hypotheses applicable to biology.
2. Level: the elemental units of the model in the hierarchy from atoms to societies.
3. Generality: the range of biological systems the model can represent.
4. Abstraction: the complexity, relative to the target, or amount of detail include in the model.
5. Structural accuracy: how well the model represents the actual mechanisms underlying the behavior.
6. Performance match: to what extent the model behavior matches the target behavior.
7. Medium: the physical basis by which the model is implemented.

Depending on how the physical model is defined within this framework, the employment of the method proposed by this study can lead to varying outcomes on how the morphological computation of a target trait is quantified. It is also important to note that not all morphological computation serves to benefit the controller. “Bad” morphological computation occurs when the interaction between the morphology and the environment causes more work for the controller or causes the behavior to fail (Ghazi-Zahedi, Deimel, Montufar, Wall, & Brock, 2017). This means that the morphological computation of a physical characteristic measured from the physical model may reveal that the morphology actually impedes the system from achieving its intended behavioral goal. Such instances would identify sources of error within the model that may need reassessment in its representation.

Perhaps the reason why embodied cognition has dodged attempts to establish a formal model is because doing so is equivalent to creating a formal model that quantifies how much each aspect of a biological system contributes to each facet of behavior. Similar to morphological computation, one can easily perceive whether a trait assisted in the completion of a specific task, but few, if any, can formally quantify its importance in fulfilling that behavior. In addition, the sheer amount of variance present across all biological systems creates immense difficulties in generalizing a formal model that can encompass each and every discrepancy and inconsistency. Reducing the biological world

to a universal set of numbers is certainly a Herculean task that many would feel inadequately prepared to face. In comparison, the task of building a world of symbols and algorithms from the bottom-up would be a welcomed challenge.

Conclusion

This experiment investigated whether algorithmic information theory can be used to measure morphological computation, and thus quantify embodied cognition. In order to determine the viability of this approach, we created different physical models of the corn snake's vertebral column and tested not only their physical properties but also whether we could measure the gaps in behavior that arose from differences in morphology. Based on the results, the changes in behavior caused by specific differences in morphology were distinct enough to be measured in number of code lines. Therefore, certain morphological characteristics could be correlated with specific lengths of code. In addition, the observed data demonstrated an additive effect, where the removal of multiple morphological traits led to greater changes in behavior that required more lines of code to properly account and adjust for the difference. While this approach suffers from several flaws in its design, the core idea and the general framework of this method offer substantial promise in providing further information about the relationship between the morphology of the agent and the cognitive computation required for it to execute specific behaviors. The use of computer code lines and algorithmic information theory may very well contribute to the formation of stronger ties between the physical symbol system hypothesis and embodiment, thus bridging the chasm between the computational theory of mind and embodied cognition.

Most importantly, this investigation is not the only effort being made to connect these two models of cognition together. Within the field of cognitive science, several researchers have already begun to formulate different ways to unify the discipline by integrating elements from both theories. For cognitive scientist Chris Eliasmith, the answer to properly explaining cognition comes in the form of a model he called "Semantic Pointer Architecture." This cognitive architecture adopts cognitively relevant representations, computations, and dynamics that are natural to implement in large-scale, biologically plausible neural networks. Through this process, Eliasmith aims to produce a unified characterization of biological cognition, which understands cognition as a biological process by building models that implement cognitive behavior (Eliasmith, 2013). With the help of these kinds of all-encompassing models, we may be able to take another step forward in understanding the wonders of human intelligence and what it can offer to the world.

References

Brooks, R. (1990). Elephants Don't Play Chess. *Robotics and Autonomous Systems*, *6*, 3-15. https://doi.org/10.1016/S0921-8890(05)80025-9

Brooks, R. (1991). Intelligence without representation. *Artificial Intelligence*, *47*, 139-159. https://doi.org/10.1016/0004-3702(91)90053-M

Carmona, R., Alba, D. M., Delfino, M., Gimenez, J. M. R., Rotgers, C., Mengual, J., ...Moya-Sola, S. (2010). Snake fossil remains from the middle Miocene stratigraphic series of Abocador de Can Mata (els Hostalets de Pierola, Catalonia, Spain). *Cidaris*, *30*, 77-83. Retrieved from https://www.researchgate.net/publication/237613257_Snake_fossil_remains_from_the_middle_Miocene_stratigraphic_series_of_Abocador_de_Can_Mata_els_Hostalets_de_Pierola_Catalonia_Spain

Chater, N., & Vitanyi, P. (2003). Simplicity: a unifying principle in cognitive science? *TRENDS in Cognitive Science*, *7*(1), 19-22. https://doi.org/10.1016/S1364-6613(02)00005-0

Church, A. (1936). An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58(2), 345-363. doi: 10.2307/2371045

Copeland, B. J. (2017). The Church-Turing Thesis. The Stanford Encyclopedia of Philosophy (Spring 2019 Edition), E. N. Zalta (Ed.). Retrieved March 30, 2019, from https://plato.stanford.edu/entries/church-turing/

Cox, D. (2016, May 12). Almost all snakes have the same, mindboggling superpower. *BBC*. Retrieved from http://www.bbc.com/earth/story/20160511-almost-all-snakes-have-the-same-mindboggling-superpower

Davis, M. (2000). *The Universal Computer: The Road from Leibniz to Turing* (1st ed.). New York, NY: W. W. Norton & Company, Inc.

Eliasmith, C. (2013). *How to Build a Brain: A Neural Architecture for Biological Cognition*. New York, NY: Oxford University Press.

Etnier, S. A. (2001). Flexural and Torsional Stiffness in Multi-Jointed Biological Beams. *The Biological Bulletin*, *200*, 1-8. doi: 10.2307/1543080

Froese, T., & Ziemke, T. (2009). Enactive artificial intelligence: Investigating the systemic organization of life and mind. *Artificial Intelligence*, *173*(3-4), 466-500. Retrieved from https://www.sciencedirect.com/science/article/pii/S0004370208002105

Gans, C. (1962). Terrestrial locomotion without limbs. *American Zoologist*, *2*(2), 167-182. https://www.jstor.org/stable/3881207

Gans, C. (1970). How snakes move. *Scientific American*, *222*(6), 82-99. https://www.jstor.org/stable/24925828

Ghazi-Zahedi, K., Haeufle, D. F. B., Montúfar, G., Schmitt, S., & Ay, N. (2016). Evaluating Morphological Computation in Muscle and DC-Motor Driven Models of Hopping Movements. *Frontiers in Robotics and AI*, *3*(42). doi: 10.3389/frobt.2016.00042

Ghazi-Zahedi, K., Langer, C., & Ay, N. (2017). Morphological computation: Synergy of body and brain. *Entropy*, *19*(456), 1-15. doi:10.3390/e19090456

Ghazi-Zahedi, K., Deimel, R., Montufar, G., Wall, V., & Brock, O. (2017, September 24-28). *Morphological computation: The good, the bad, and the ugly.* Paper presented at the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Vancouver, BC: IEEE. doi: 10.1109/IROS.2017.8202194

Goldinger, S.D., Papesh, M., Barnhart, A.S., Hansen, W.A., & Hout, M. C. (2016). The poverty of embodied cognition. *Psychonomic Bulletin & Review*, *23*, 959-978. doi: 10.3758/s13423-015-0860-1

Gottlieb, A. (n.d.). The Ghost and the Princess. Lapham's Quarterly. Retrieved from https://www.laphamsquarterly.org/states-mind/ghost-and-princess

Hilbert, A., & Ackermann, W. (1950). *Principles of mathematical logic* (2nd ed.) (L. M. Hammond, G. G. Leckie, & F. Steinhardt, Trans.). R. E. Luce (Ed.). New York, NY: Chelsea Pub Co. (Original work published in 1938).

Hoffman, C. (2003). Helmholtz' apparatuses: Telegraphy as working model of nerve physiology. *Philosophia Scientiae*, *7*(1), 129-149. Retrieved from http://www.numdam.org/article/PHSC_2003__7_1_129_0.pdf

Hoffman, M., & Pfeifer, R. (2011). The Implications of Embodiment for Behavior and Cognition: Animal and Robotic Case Studies. *The Implications of Embodiment: Cognition and Communication* (pp.31-58). Cambridge, MA: Academic Press. Retrieved from https://arxiv.org/abs/1202.0440

Houssaye, A., Boistel, R., Böhme, Wolfgang, & Herrel, A. (2013). Jack-of-all-trades master of all? Snake vertebrae have a generalist inner organization. Naturwissenschaften, 100(11), 997-1006. https://doi.org/10.1007/s00114-013-1102-x

Immerman, N. (2015). Computability and Complexity. The Stanford Encyclopedia of Philosophy (Winter 2018 Edition), E.N. Zalta (Ed.). Retrieved March 30, 2019, from https://plato.stanford.edu/entries/computability/

Jaeger, H. (2007). Echo state network. *Scholarpedia*, *2*(9). Retrieved April 1, 2019, from http://www.scholarpedia.org/article/Echo_state_network

Johnson, R. G. (1955). The adaptive and phylogenetic significance of vertebral form in snakes. *Evolution*, *9*(4), 367-388. https://doi.org/10.1111/j.1558-5646.1955.tb01548.x

Lindell, L. E. (1994). The evolution of vertebral number and body size in snakes. *Functional Ecology*, *8*(6), 708-719. doi: 10.2307/2390230

Lorenz, D. M., Jeng, A., & Deem, M. W. (2015). The Emergence of Modularity in Biological Systems. *Physics of Life Reviews*, *8*(2), 129-160. doi: 10.1016/j.plrev.2011.02.003

Moon, B. R. (1999). Testing an inference of function from structure: Snake vertebrae do the twist. *Journal of Morphology*, *241*(3), 217-125. https://doi.org/10.1002/(SICI)1097-4687(199909)241:3<217::AID-JMOR4>3.0.CO;2-M

Morinaga, G., & Bergmann, P. J. (2019). Angles and waves: Intervertebral joint angles and axial kinematics of limbed lizards, limbless lizards, and snakes. *Zoology*. https://doi.org/10.1016/j.zool.2019.04.003

Newell, A., & Simon, H. A. (1976). Computer science as empirical inquiry: Symbols and search. *Communications of the ACM*, 19(3), 113-126. doi: 10.1145/360018.360022

Paul, C. (2006). Morphological computation: A basis for the analysis of morphology and control requirements. *Robotics and Autonomous Systems*, *54*(8), 619-630. https://doi.org/10.1016/j.robot.2006.03.003

Pfeifer, R., & Bongard, J. C. (2006). *Intelligent Systems: Properties and Principles. How the Body Shapes the Way We Think: A New View of Intelligence* (pp. 89-140). Cambridge, MA: The MIT Press (Bradford Books).

Pfeifer, R. & Iida, F. (2006, January). *Morphological computation: Connecting body, brain and environment*. Paper presented at Lecture Notes in Computer Science. doi: 10.1007/978-3-642-00616-6_5

Rescorla, M. (2017). The Computational Theory of Mind. The Stanford Encyclopedia of Philosophy (Spring 2017 Edition), E. N. Zalta (Ed.). Retrieved March 30, 2019, from https://plato.stanford.edu/entries/computational-mind/

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error-propagation. In J. A. Feldman, P. J. Hayes, & D. E. Rumelhart (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (Vol. 1, pp. 318-362). Cambridge, MA: The MIT Press.

Sauro, H. M. (2008). Modularity defined. *Molecular Systems Biology*, *4*, 166. doi:

10.1038/msb.2008.3

Searle, J. (1980). Minds, brains, and programs. *Behavioral and Brain Sciences*, *3*(3), 417-457. Retrieved from http://cogprints.org/7150/1/10.1.1.83.5248.pdf

Searle, J. (1990). Is the Brain a Digital Computer? *Proceedings and Addresses of the American Philosophical Association*, *64*(3), 21-37. doi:10.2307/3130074

Shanahan, M. (2016). The Frame Problem. The Stanford Encyclopedia of Philosophy (Spring 2016 Edition), E.N. Zalta (Ed.). Retrieved March 31, 2019, from https://plato.stanford.edu/entries/frame-problem/

Shapiro, L. (2011). *Embodied Cognition*. New York, NY: Routledge.

Turing, A. (1936). On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, *s2-42*(1), 230-265. https://doi.org/10.1112/plms/s2-42.1.230

Turing, A. (1950). Computing machinery and intelligence. *Mind: A Quarterly Review of Psychology and Philosophy*, *59*(236), 433-460. https://doi.org/10.1093/mind/LIX.236.433

van Gelder, T. (1995). What might cognition be, if not computation? *The Journal of Philosophy*, *92*(7), 345-381. doi: 10.2307/2941061

von Neumann, J. (1958). *The Computer and the Brain*. New Haven, CT: Yale University Press.

Wallach, V., & Peters, J. A. (2019). Snake. Encyclopedia Britannica. Retrieved April 1, 2019 from https://www.britannica.com/animal/snake

Webb, B. (2001). Can robots make good models of biological behavior? Behavioral and Brain Sciences, 24(6), 1033-1050. doi: 10.1017/S0140525X01000127

Wilson, M. (2002). Six views of embodied cognition. *Psychonomic Bulletin & Review*, *9*(4), 625-636. Retrieved from https://link.springer.com/article/10.3758/BF03196322

Wilson, A.D., & Golonka, S. (2013). Embodied cognition is not what you think it is. *Frontiers in Psychology*, *4*(58), 1-13. https://doi.org/10.3389/fpsyg.2013.00058

Woody, W. D., & Viney, W. (2017). *A History of Psychology: The Emergence of Science and Applications* (6th ed., p. 175). Abingdon, United Kingdom: Routledge.

Wright, C., Johnson, A., Peck, A., McCord, Z., Naaktgeboren, A., Gianfortoni, P.,

…Choset, H. (2007). Design of a modular snake robot. Proceedings from the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA: IEEE. Retrieved from https://www.andrew.cmu.edu/user/amj1/papers/IROS07_Wright_1345.pdf

Yanofsky, N. S. (2018, August 2). Kolmogorov Complexity and Our Search for Meaning. *Nautilus*. Retrieved from http://nautil.us/issue/63/horizons/kolmogorov-complexity-and-our-search-for-meaning

Appendix A

Arduino Code for the First Round of CPG Testing:

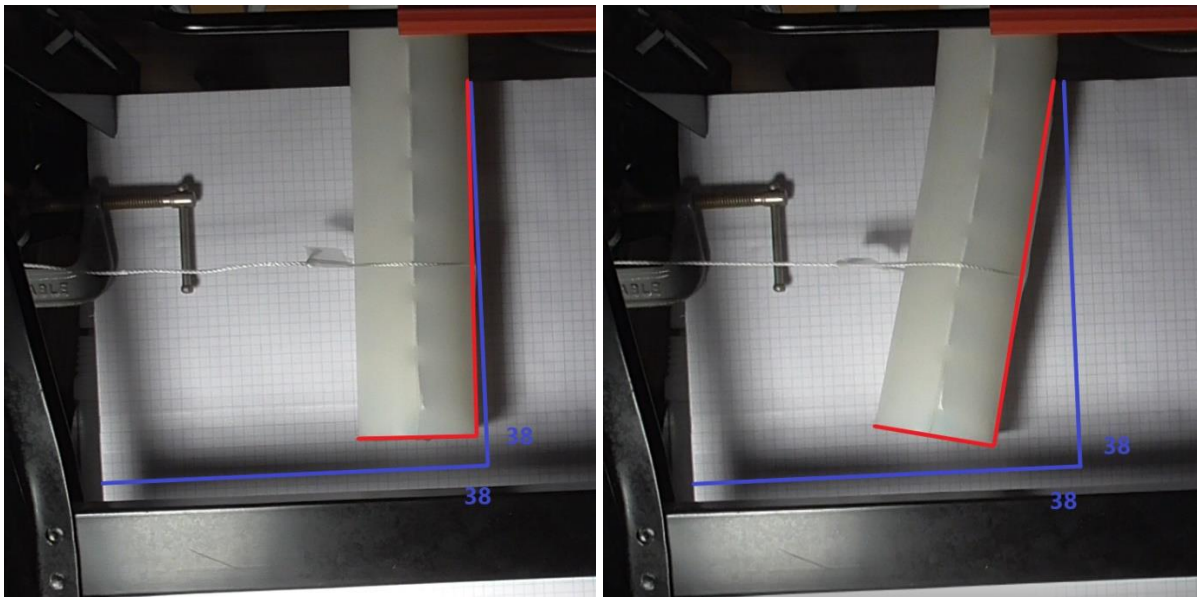
```
/*
  ControlSineWave: The two servo motors will move in accordance to a sinusoidal wave.
  This is the original code that was used during Round 1 of CPG testing.
*/

//Import the servo library
#include <Servo.h>

//Create the servo objects to control each servo
Servo s2;
Servo s9;

//Variables
const int servoPin2 = 2;
const int servoPin9 = 9;

//Variable to store the servo position
int pos1;

//Unit of information change
int lineUnit = 10;

//Variable to store angle range of servos
//Add lineUnit to setRotationAngle to increase angle of rotation
//Subtract lineUnit to setRotationAngle to decrease angle of rotation
int setRotationAngle = 120;
int setLeftRange = 90 - (setRotationAngle / 2);
int setRightRange = (setRotationAngle / 2) + 90;

//Variable to determine speed of servo
int servoSpeed = 8;

void setup() {
  s2.attach(servoPin2);
  s9.attach(servoPin9);
}

void loop() {
  //Goes from 0 degrees to 180 degrees in steps of 1 degree
  for (pos1 = setLeftRange; pos1 <= setRightRange; pos1 += 1) {
    s2.write(pos1);
```

```
    s9.write(180 - pos1);
    delay(servoSpeed);
  }
  // goes from 180 degrees to 0 degrees in steps of 1 degree
  for (pos1 = setRightRange; pos1 >= setLeftRange; pos1 -= 1) {
    s2.write(pos1);
    s9.write(180 - pos1);
    delay(servoSpeed);
  }
}
```

Appendix B

Annotated Screenshots of the Mechanical Test for Flexural Stiffness:

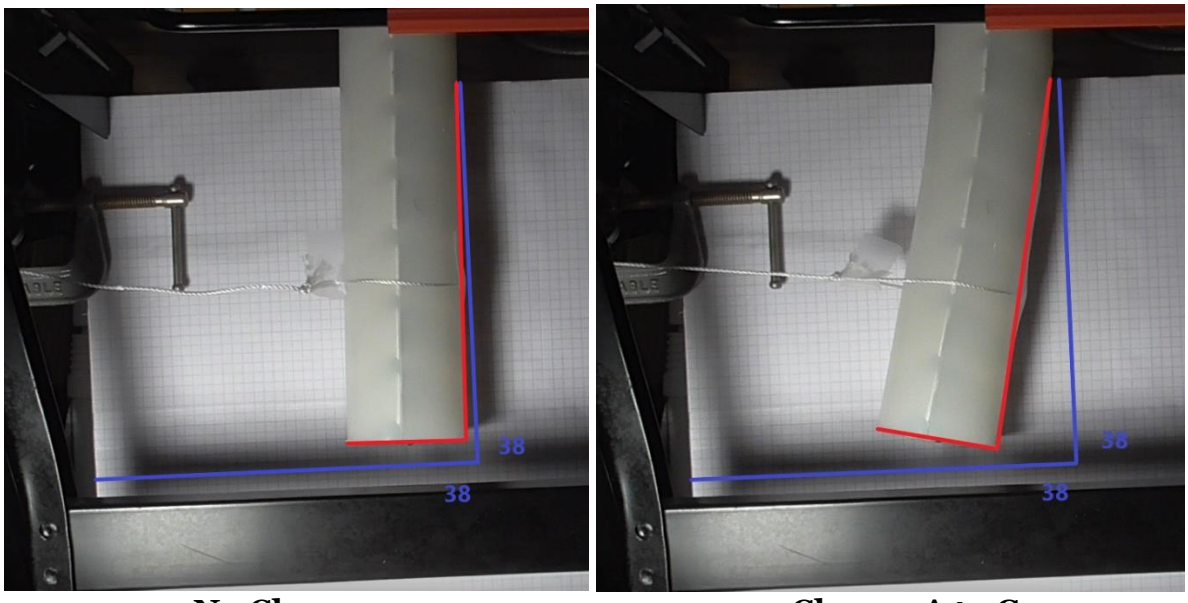
Model: CMB



No Clamps **Clamps A to G**

Figure B1. *Observed lateral displacement of CMB during the flexural stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load.*

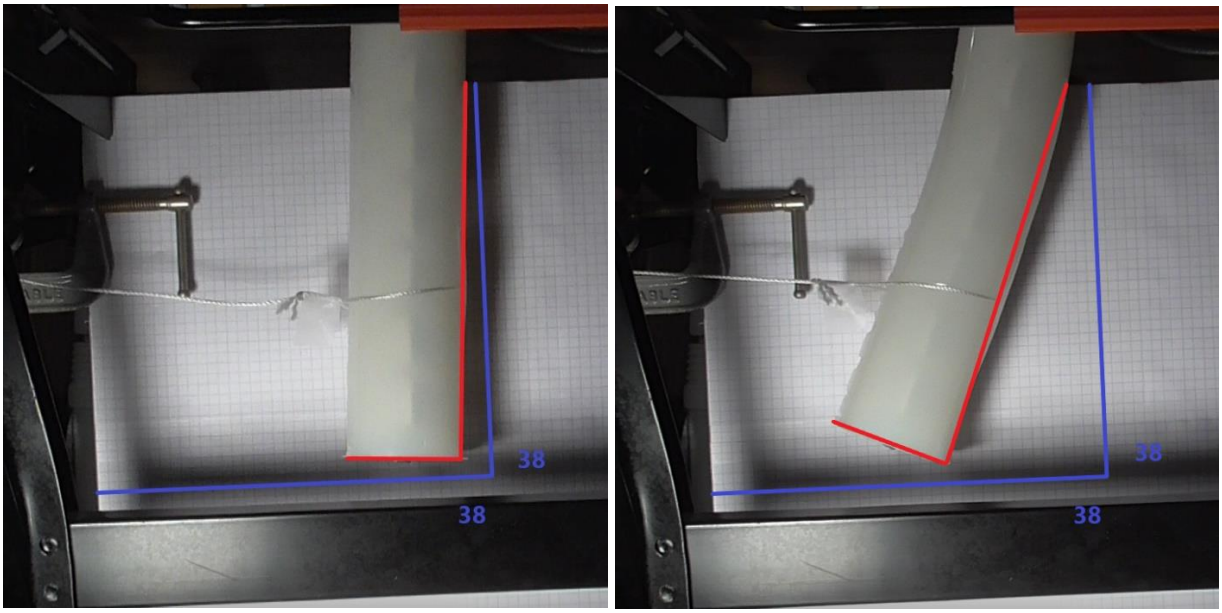
Model: NSB



No Clamps **Clamps A to G**

Figure B2. *Observed lateral displacement of NSB during the flexural stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load.*

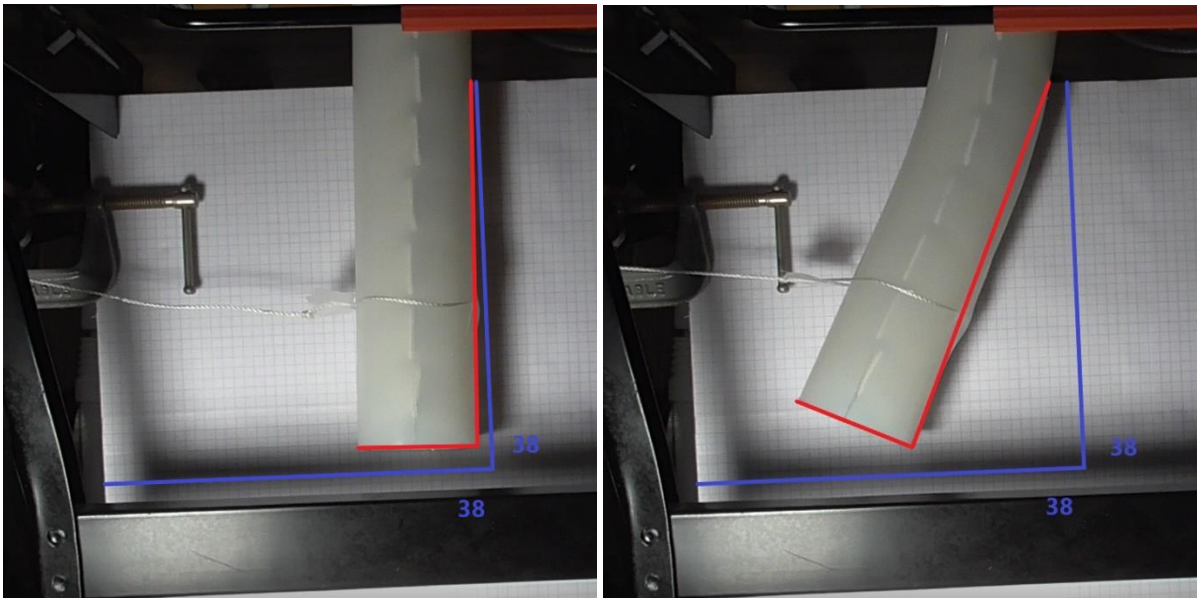
Model: NPB



No Clamps **Clamps A to G**

Figure B3. *Observed lateral displacement of NPB during the flexural stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load.*

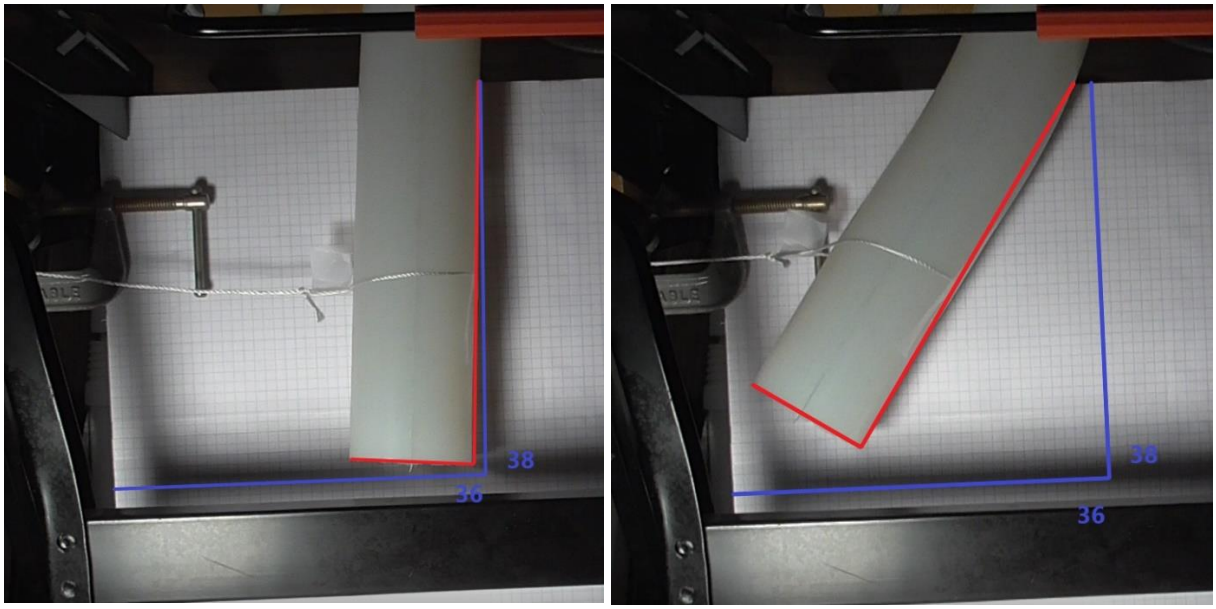
Model: SPB



No Clamps **Clamps A to G**

Figure B4. *Observed lateral displacement of SPB during the flexural stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load.*

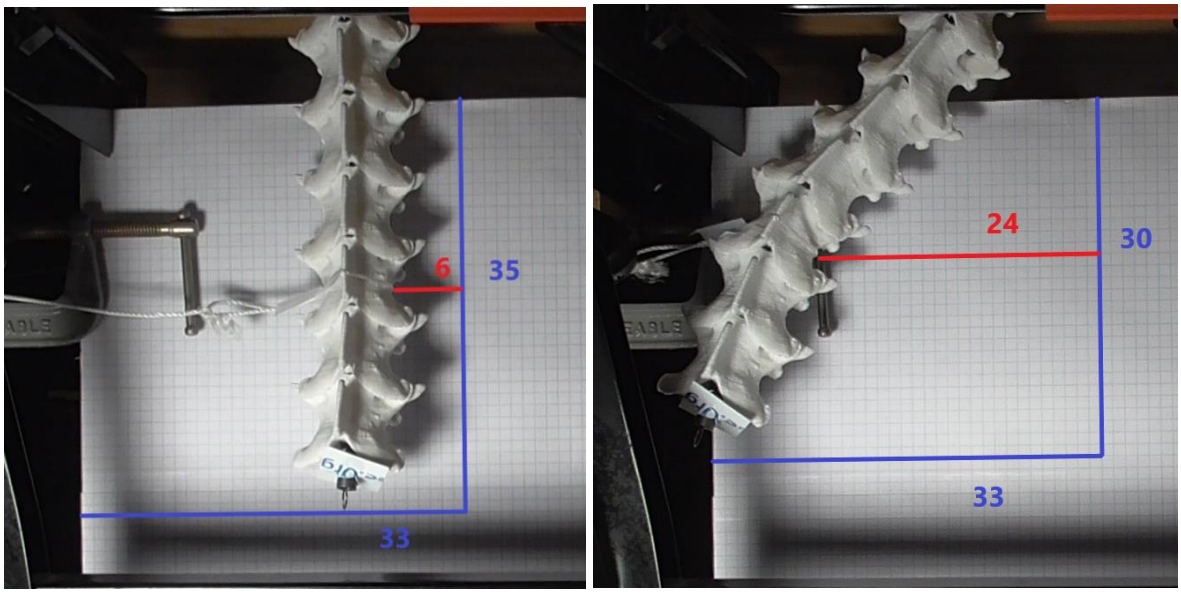
Model: NVB



No Clamps **Clamps A to G**

Figure B5. *Observed lateral displacement of NVB during the flexural stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load.*

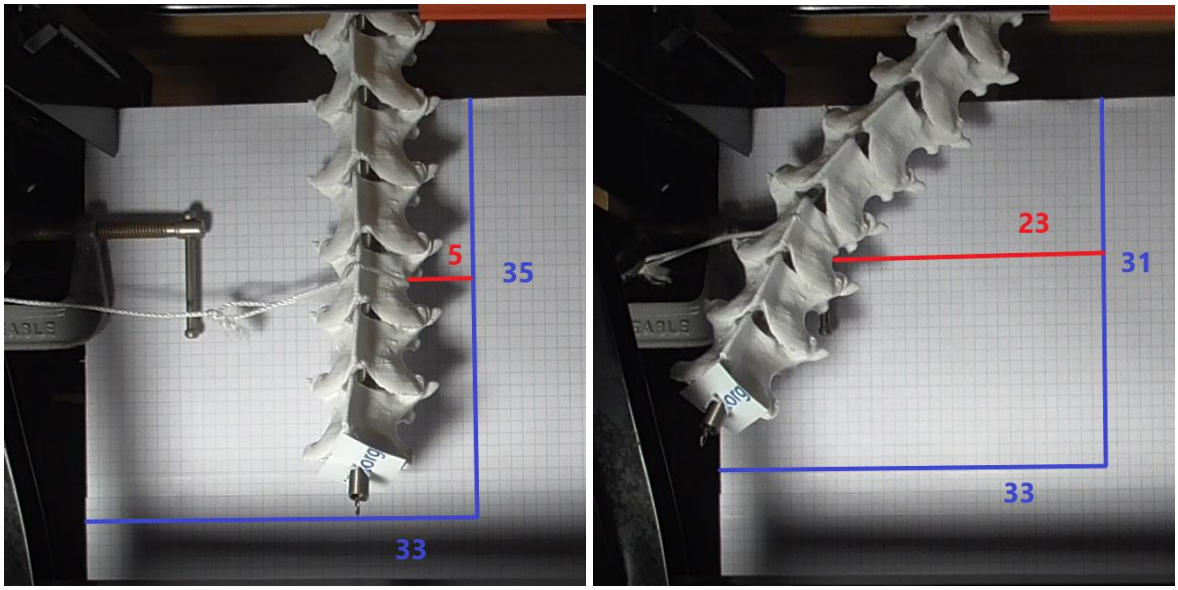
Model: CMS



No Clamps **Clamp A**

Figure B6. *Observed lateral displacement of CMS during the flexural stiffness mechanical test. Left: No clamps attached to the load. Right: Clamp A attached to the load.*

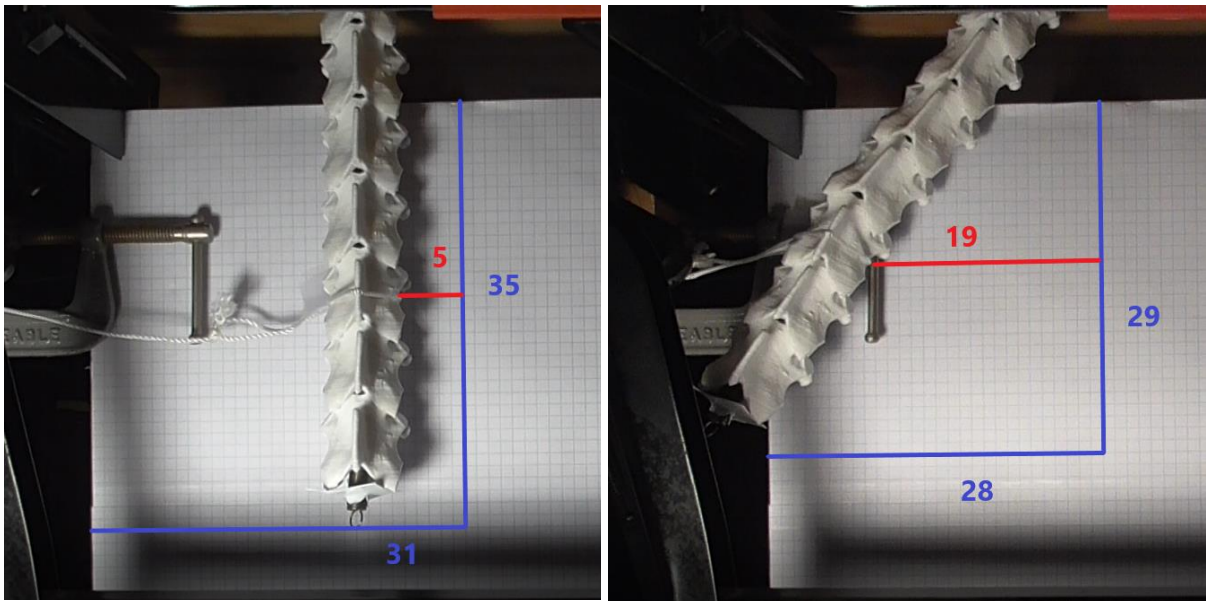
Model: NSS



No Clamps **Clamp A**

Figure B7. *Observed lateral displacement of NSS during the flexural stiffness mechanical test. Left: No clamps attached to the load. Right: Clamp A attached to the load.*

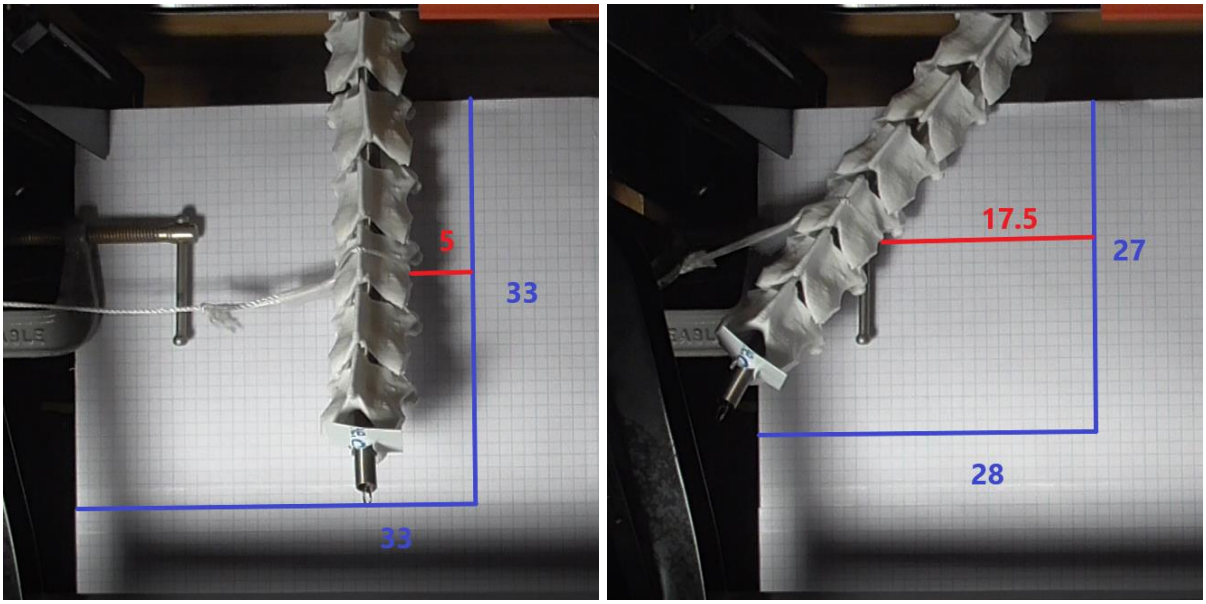
Model: NPS



No Clamps **Clamp A**

Figure B8. *Observed lateral displacement of NPS during the flexural stiffness mechanical test. Left: No clamps attached to the load. Right: Clamp A attached to the load.*

Model: SPS

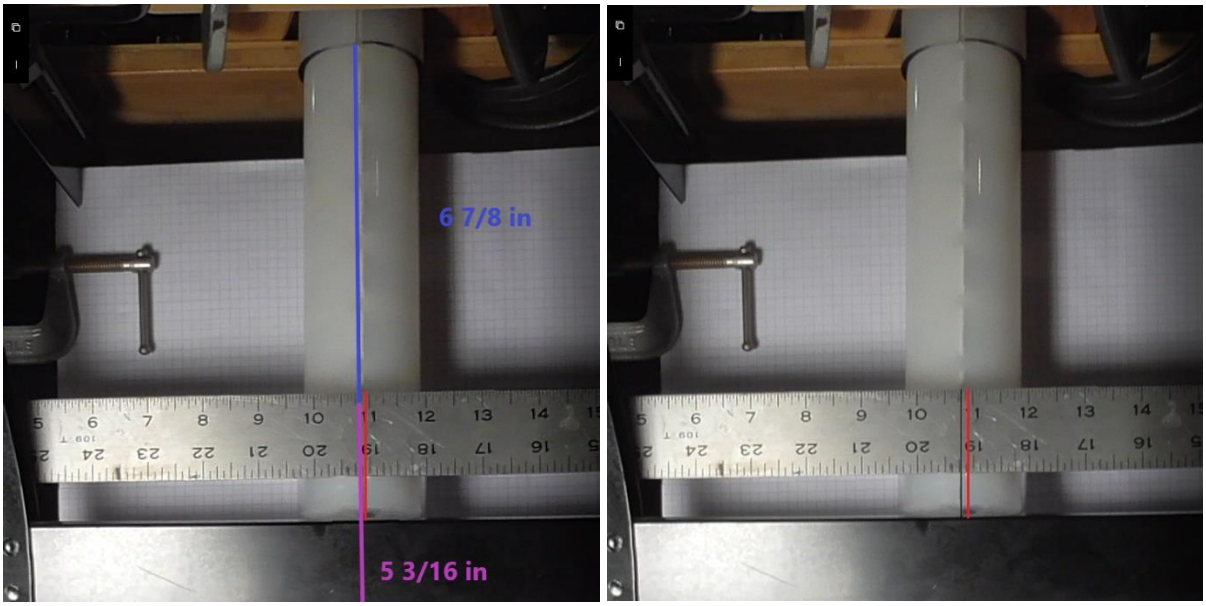


No Clamps **Clamp A**

Figure B9. *Observed lateral displacement of SPS during the flexural stiffness mechanical test. Left: No clamps attached to the load. Right: Clamp A attached to the load.*

Annotated Screenshots of the Mechanical Test for Torsional Stiffness:

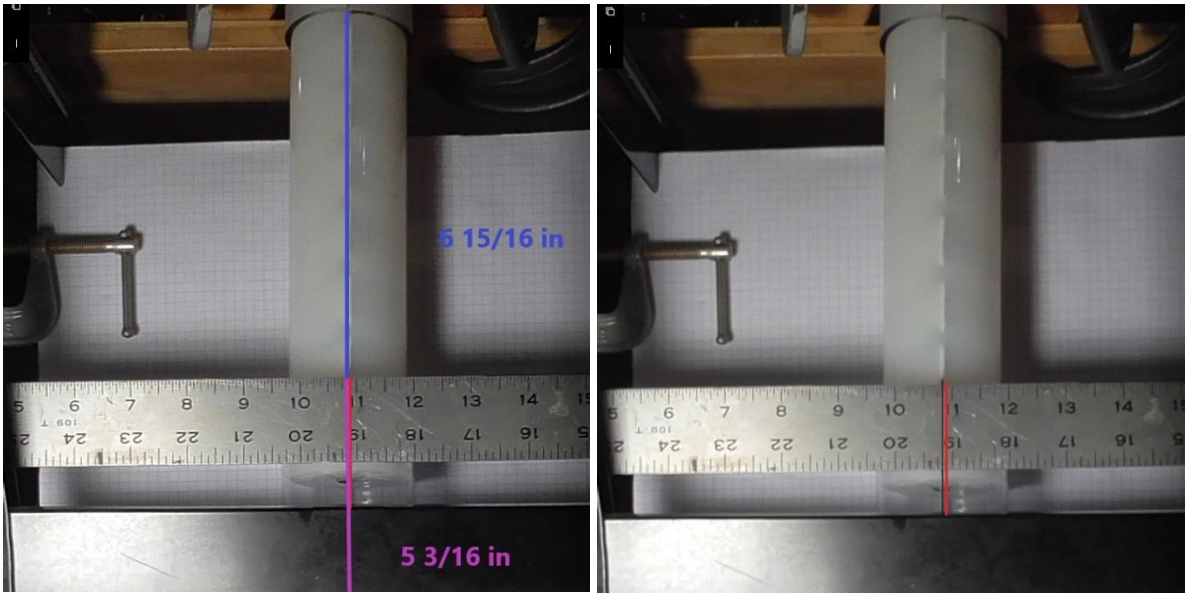
Model: CMB



No Clamps **Clamps A to G**

Figure B10. *Observed torsion of CMB during the torsional stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load. The blue line provides the height of the vertebral column and the purple line provides the height of the bottom gripper. The red line represents the initial position and the black line represents the ending position.*

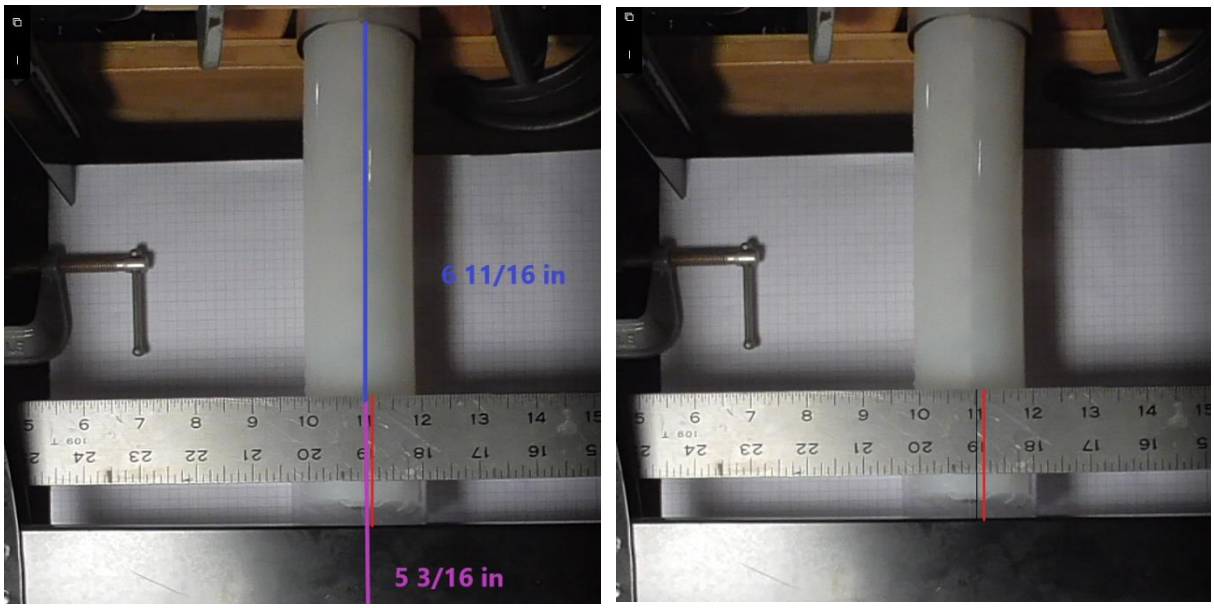
Model: NSB



No Clamps **Clamps A to G**

Figure B11. *Observed torsion of NSB during the torsional stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load. The blue line provides the height of the vertebral column and the purple line provides the height of the bottom gripper. The red line represents the initial position and the black line represents the ending position.*

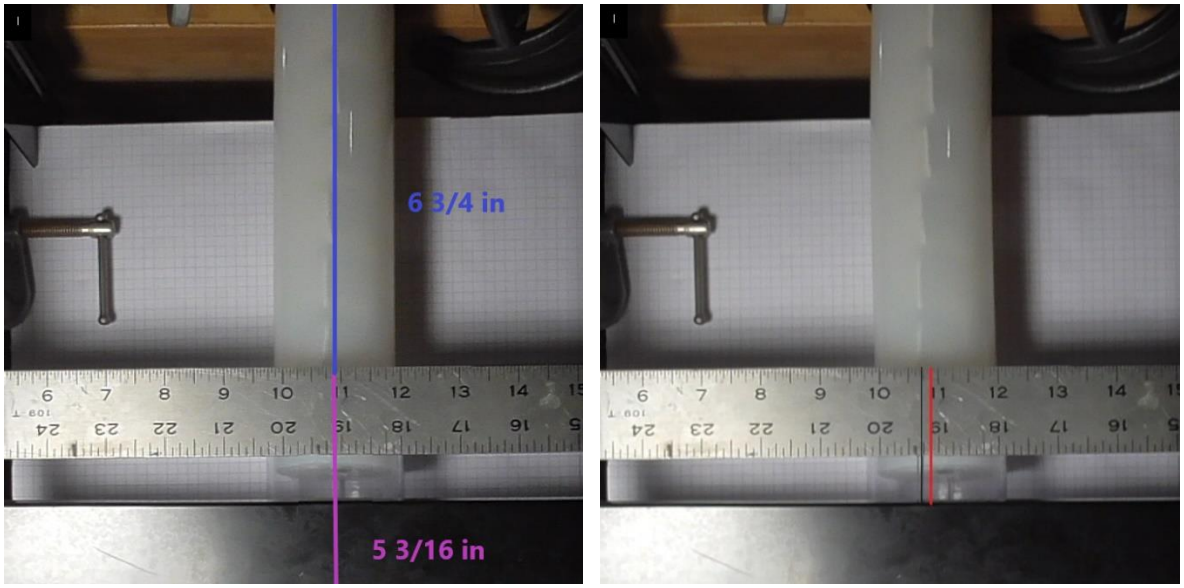
Model: NPB



No Clamps **Clamps A to G**

Figure B12. *Observed torsion of NPB during the torsional stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load. The blue line provides the height of the vertebral column and the purple line provides the height of the bottom gripper. The red line represents the initial position and the black line represents the ending position.*

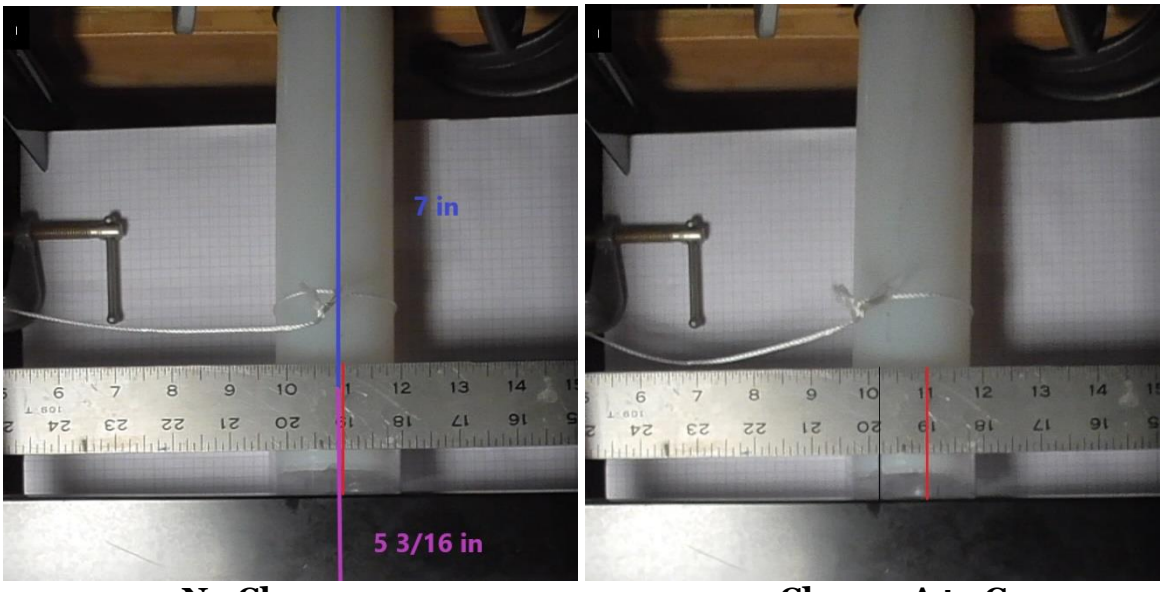
Model: SPB



No Clamps **Clamps A to G**

Figure B13. *Observed torsion of SPB during the torsional stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load. The blue line provides the height of the vertebral column and the purple line provides the height of the bottom gripper. The red line represents the initial position and the black line represents the ending position.*

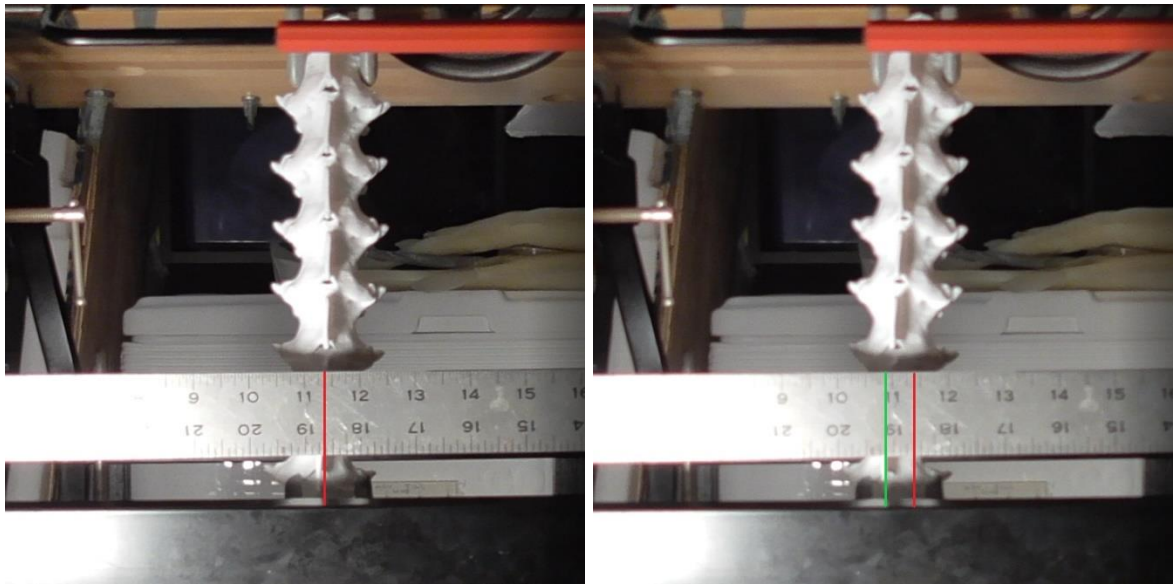
Model: NVB



No Clamps **Clamps A to G**

Figure B14. *Observed torsion of NVB during the torsional stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load. The blue line provides the height of the vertebral column and the purple line provides the height of the bottom gripper. The red line represents the initial position and the black line represents the ending position.*

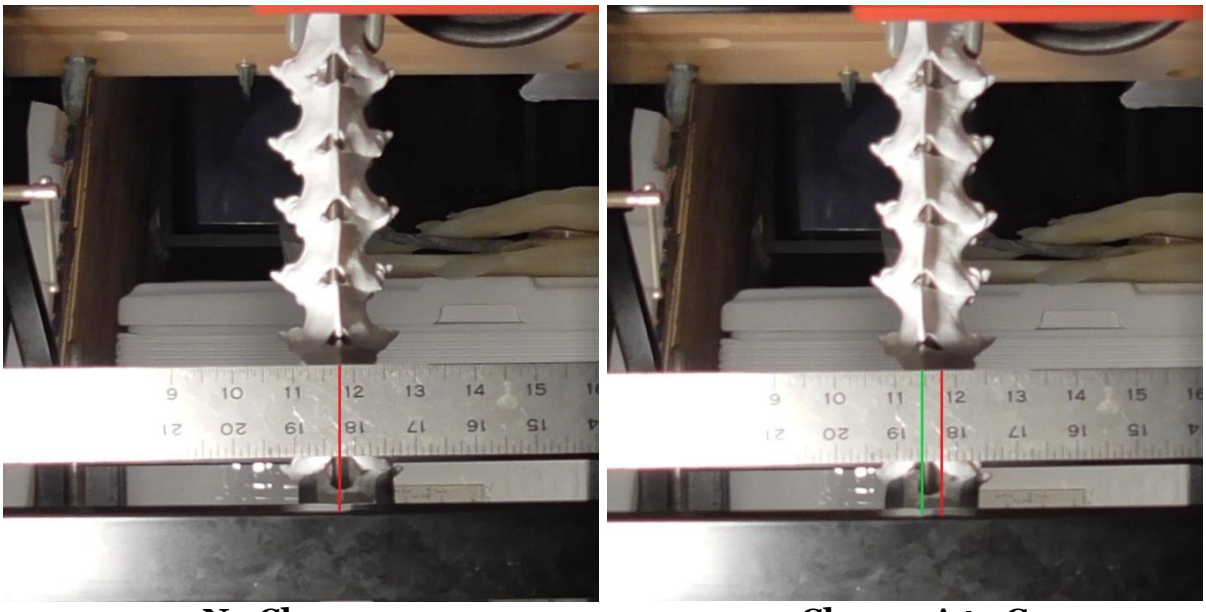
Model: CMS



No Clamps **Clamps A to G**

Figure B15. *Observed torsion of CMS during the torsional stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load. The blue line provides the height of the vertebral column and the purple line provides the height of the bottom gripper. The red line represents the initial position and the black line represents the ending position.*

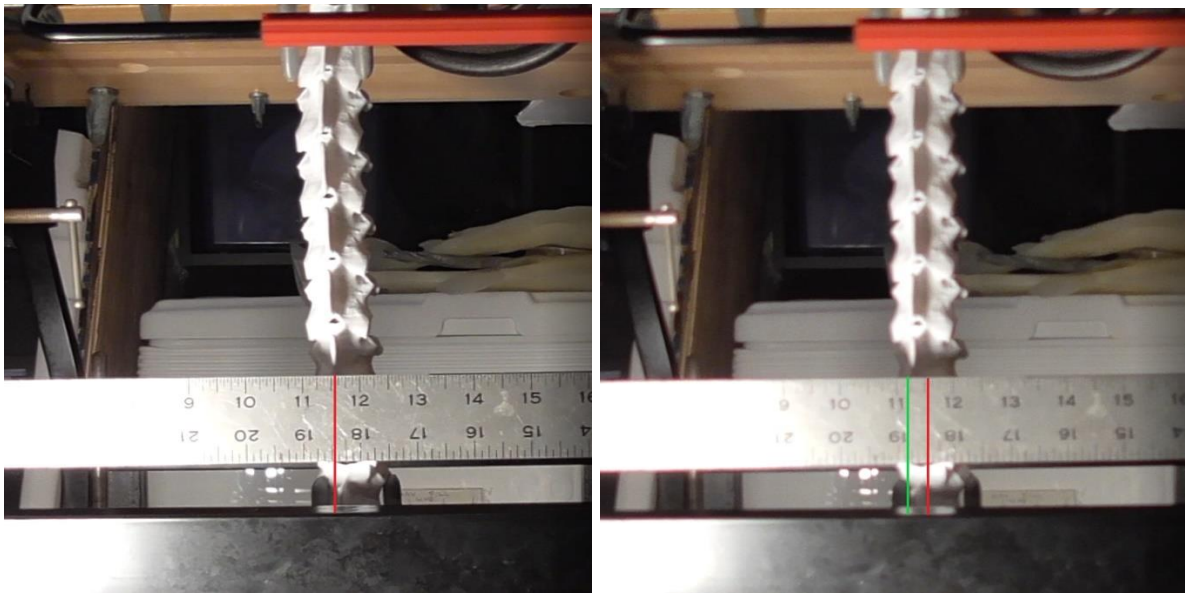
Model: NSS



No Clamps **Clamps A to G**

Figure B16. *Observed torsion of NSS during the torsional stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load. The blue line provides the height of the vertebral column and the purple line provides the height of the bottom gripper. The red line represents the initial position and the black line represents the ending position.*

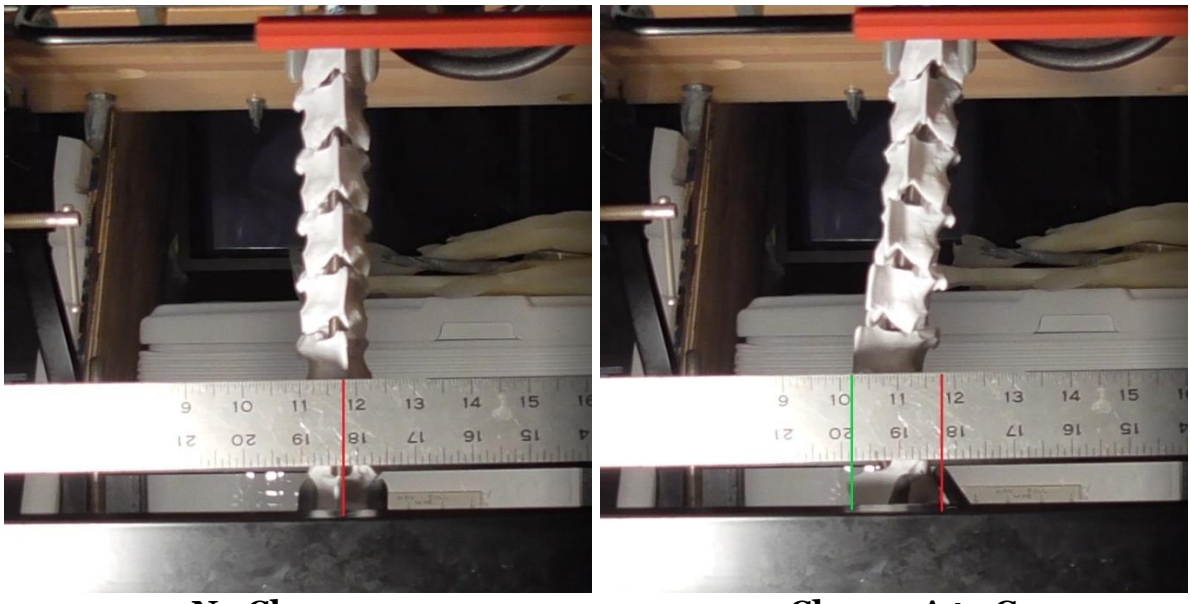
Model: NPS



No Clamps **Clamps A to G**

Figure B17. *Observed torsion of NPS during the torsional stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load. The blue line provides the height of the vertebral column and the purple line provides the height of the bottom gripper. The red line represents the initial position and the black line represents the ending position.*

Model: SPS



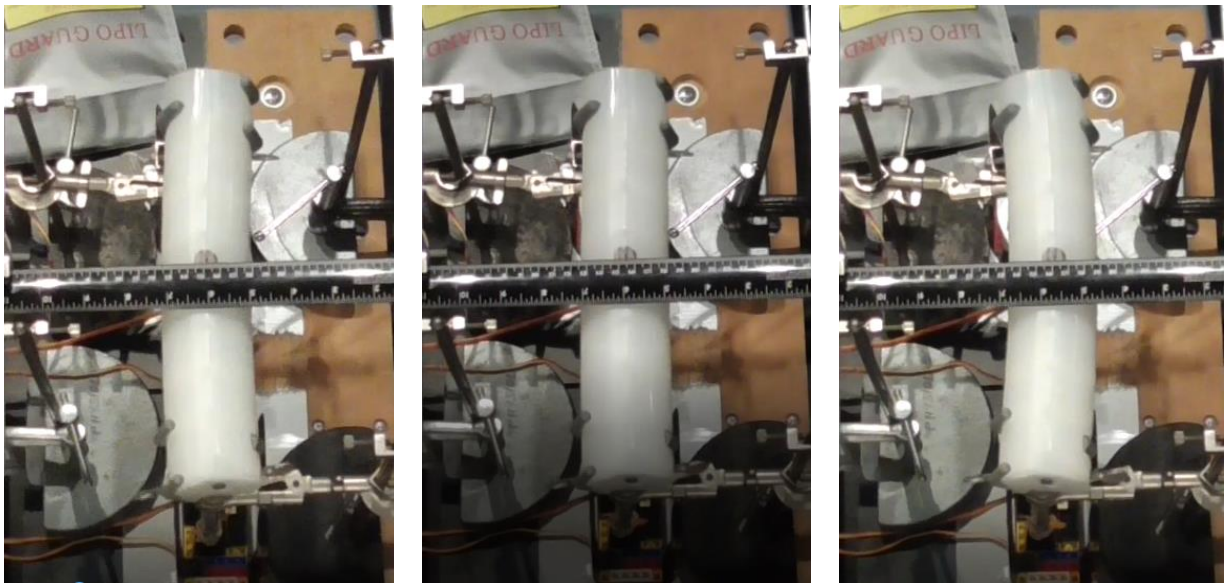
No Clamps **Clamps A to G**

Figure B18. *Observed torsion of SPS during the torsional stiffness mechanical test. Left: No clamps attached to the load. Right: Clamps A to G attached to the load. The blue line provides the height of the vertebral column and the purple line provides the height of the bottom gripper. The red line represents the initial position and the black line represents the ending position.*

Appendix C

Screenshots from the First Round of CPG Testing:

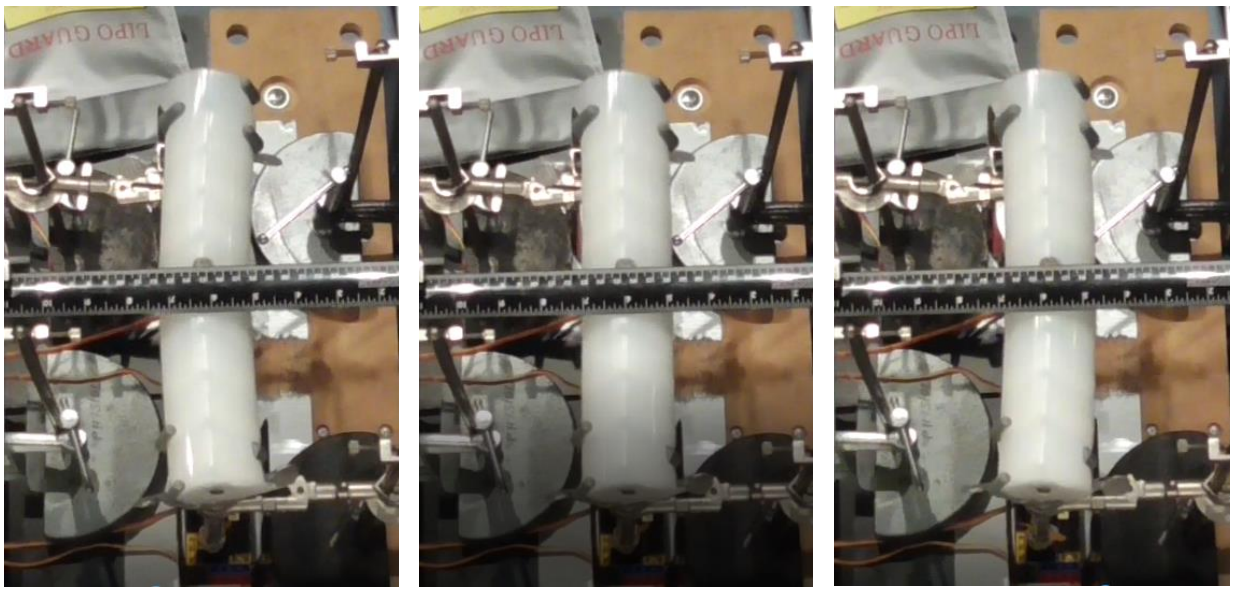
Model: CMB



Left **Center** **Right**

Figure C1. *Observed lateral undulation of CMB during the first round of CPG testing. The images were taken from the same cycle at points where the black dot was displaced furthest to the left (Left), in the middle (Center), and furthest to the right.*

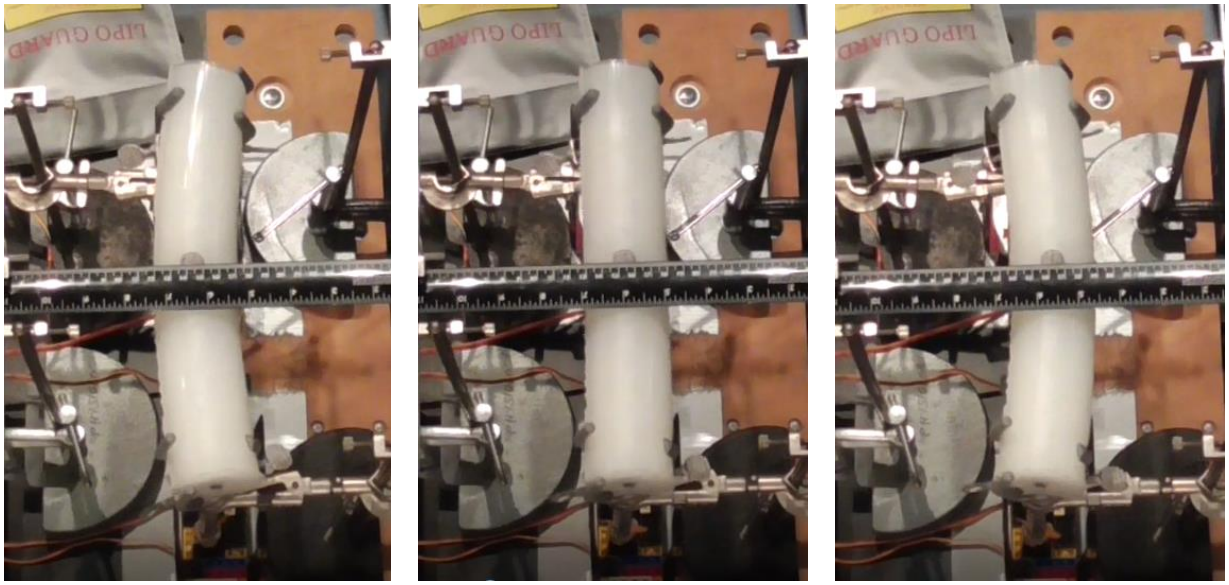
Model: NSB



Left **Center** **Right**

Figure C2. *Observed lateral undulation of NSB during the first round of CPG testing. The images were taken from the same cycle at points where the black dot was displaced furthest to the left (Left), in the middle (Center), and furthest to the right.*

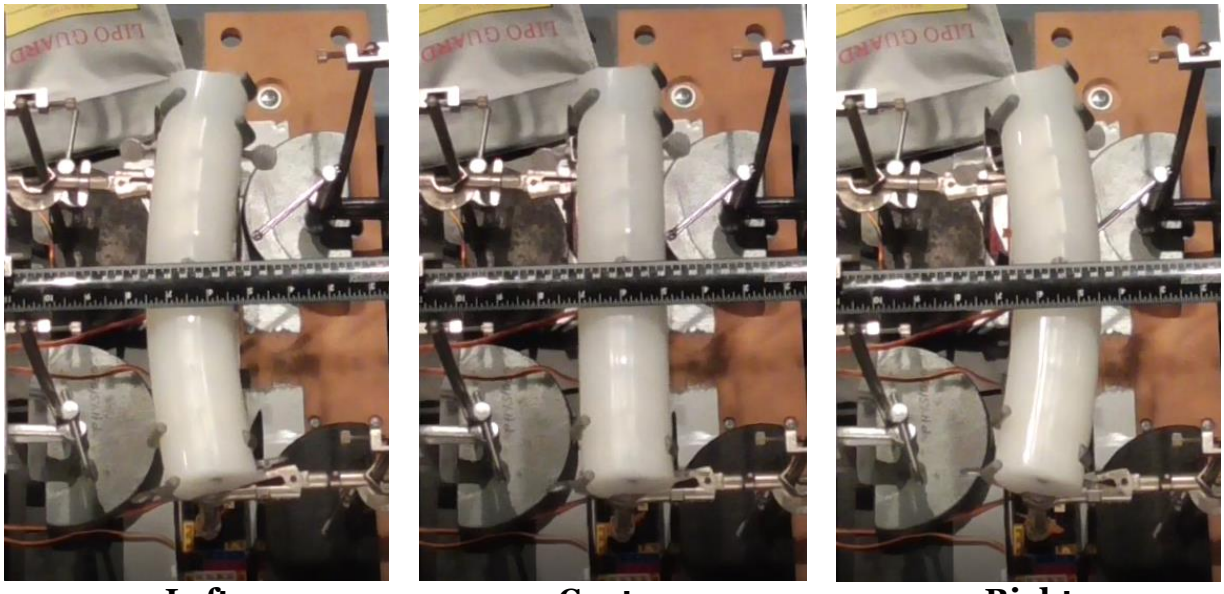
Model: NPB



Left **Center** **Right**

Figure C3. *Observed lateral undulation of NPB during the first round of CPG testing. The images were taken from the same cycle at points where the black dot was displaced furthest to the left (Left), in the middle (Center), and furthest to the right.*

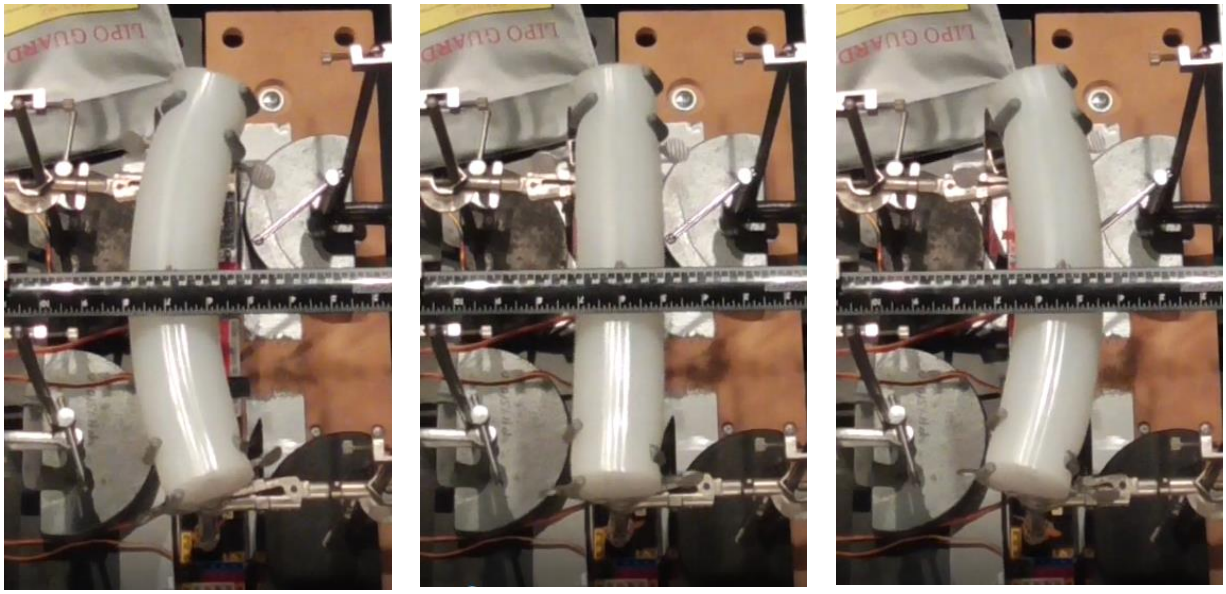
Model: SPB



Left **Center** **Right**

Figure C4. *Observed lateral undulation of SPB during the first round of CPG testing. The images were taken from the same cycle at points where the black dot was displaced furthest to the left (Left), in the middle (Center), and furthest to the right.*

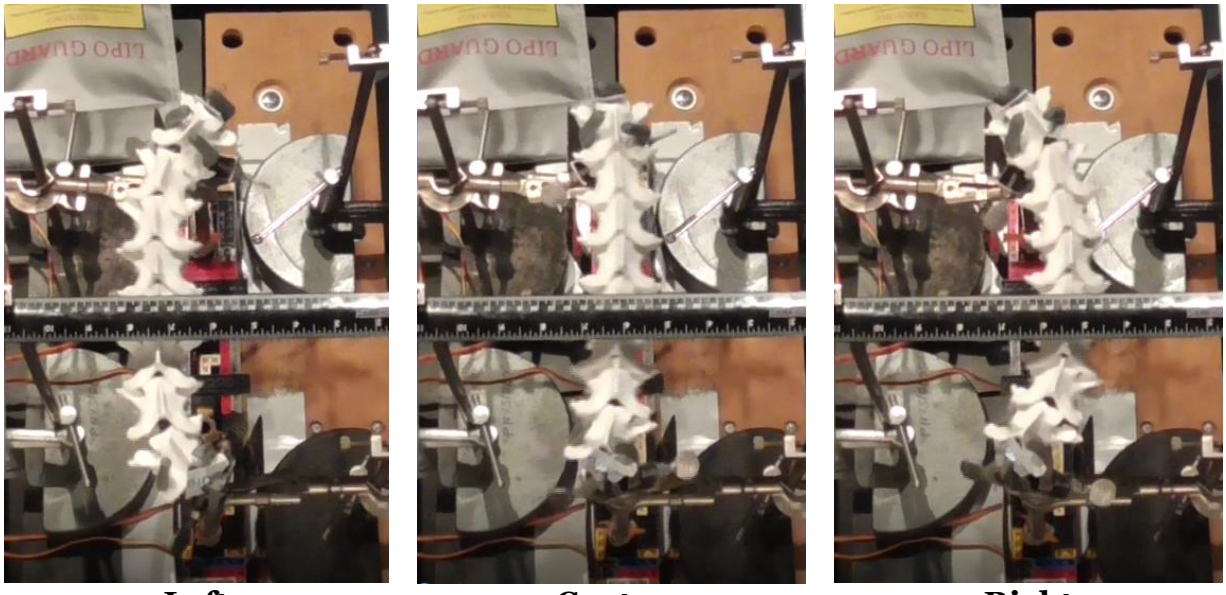
Model: NVB



Left **Center** **Right**

Figure C5. *Observed lateral undulation of NVB during the first round of CPG testing. The images were taken from the same cycle at points where the black dot was displaced furthest to the left (Left), in the middle (Center), and furthest to the right.*

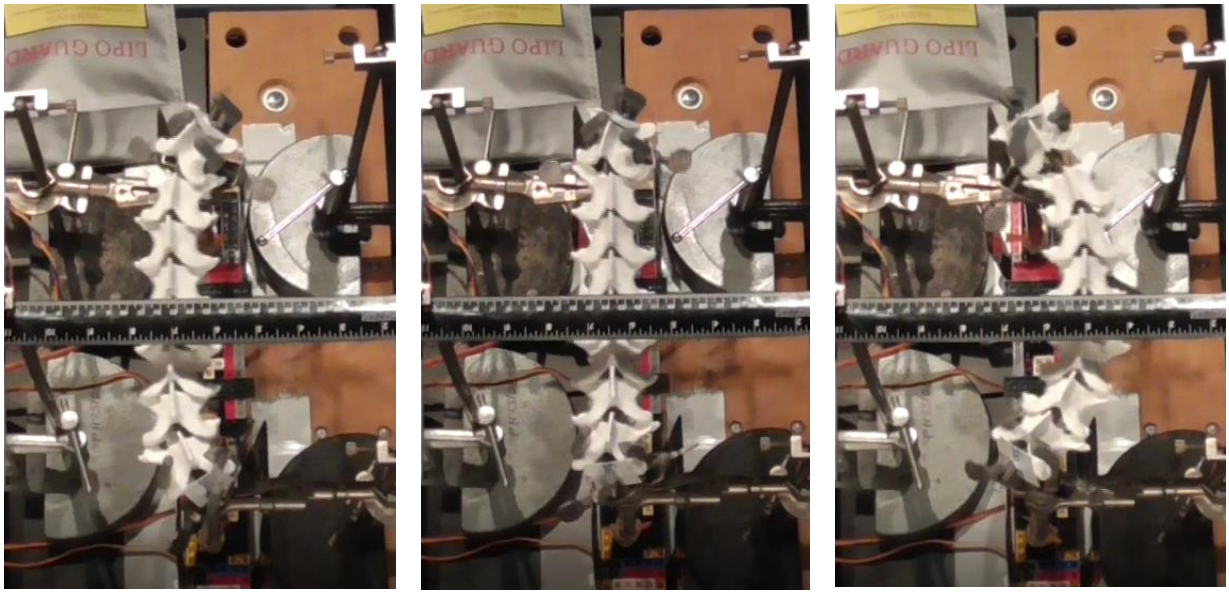
Model: CMS



Left **Center** **Right**

Figure C6. *Observed lateral undulation of CMS during the first round of CPG testing. The images were taken from the same cycle at points where the black dot was displaced furthest to the left (Left), in the middle (Center), and furthest to the right.*

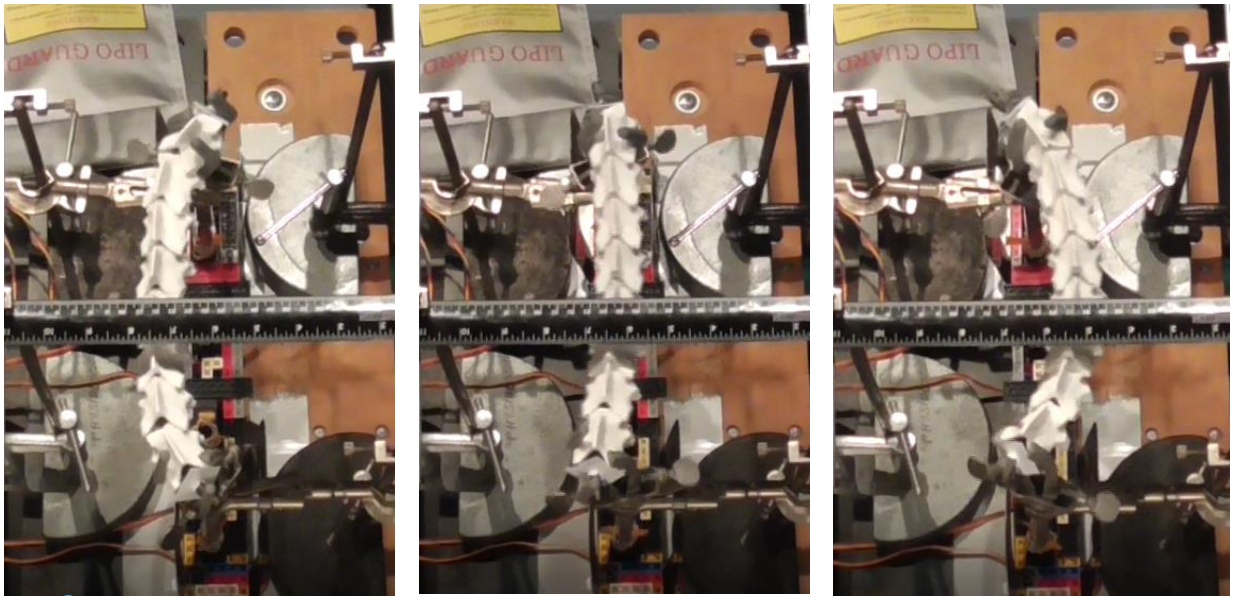
Model: NSS



Left **Center** **Right**

Figure C7. *Observed lateral undulation of NSS during the first round of CPG testing. The images were taken from the same cycle at points where the black dot was displaced furthest to the left (Left), in the middle (Center), and furthest to the right.*

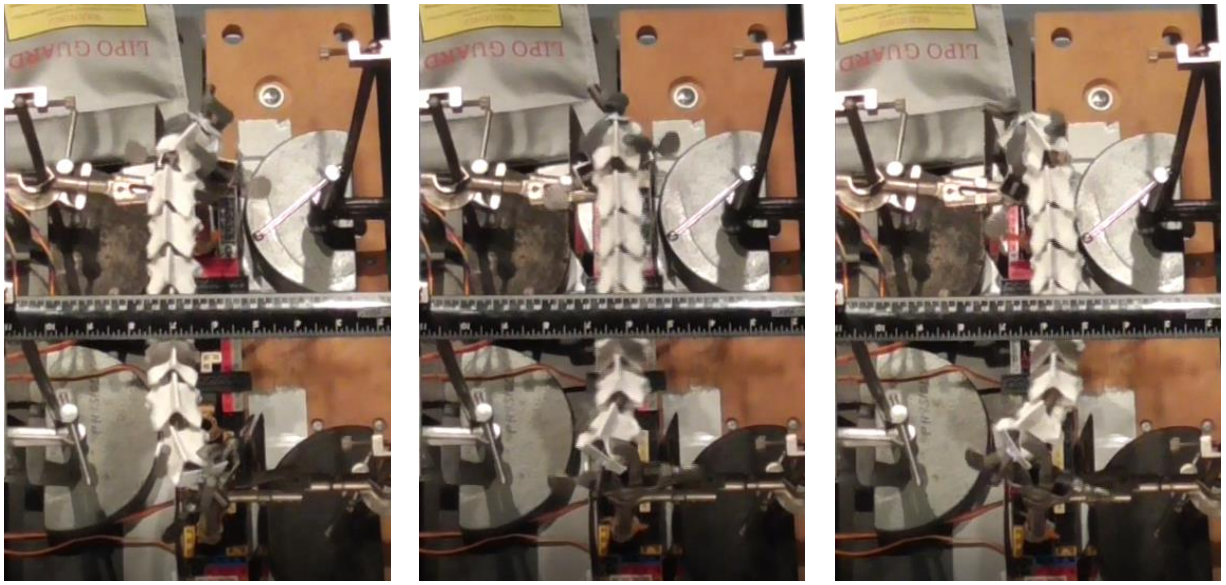
Model: NPS



Left **Center** **Right**

Figure C8. *Observed lateral undulation of NPS during the first round of CPG testing. The images were taken from the same cycle at points where the black dot was displaced furthest to the left (Left), in the middle (Center), and furthest to the right.*

Model: SPS



Left **Center** **Right**

Figure C9. *Observed lateral undulation of SPS during the first round of CPG testing. The images were taken from the same cycle at points where the black dot was displaced furthest to the left (Left), in the middle (Center), and furthest to the right.*

Appendix D

Table of Code Modifications Made to the CPG Program for All Vertebral Columns

Model	Modifications to the Arduino Code

CMB (Control)	`//Unit of information change`
	`int lineUnit = 10;`
	
	`//Variable to store angle range of servos`
	`//Add lineUnit to setRotationAngle to increase angle of rotation`
	`//Subtract lineUnit to setRotationAngle to decrease angle of rotation`
	`int setRotationAngle = 120;`
	`int setLeftRange = 90 - (setRotationAngle / 2);`
	`int setRightRange = (setRotationAngle / 2) + 90;`

NSB	`//Unit of information change`
	`int lineUnit = 10;`
	
	`//Variable to store angle range of servos`
	`//Add lineUnit to setRotationAngle to increase angle of rotation`
	`//Subtract lineUnit to setRotationAngle to decrease angle of rotation`
	`int setRotationAngle = 120`
	`+ lineUnit;`
	`int setLeftRange = 90 - (setRotationAngle / 2);`
	`int setRightRange = (setRotationAngle / 2) + 90;`

NPB	`//Unit of information change`
	`int lineUnit = 10;`
	
	`//Variable to store angle range of servos`
	`//Add lineUnit to setRotationAngle to increase angle of rotation`
	`//Subtract lineUnit to setRotationAngle to decrease angle of rotation`
	`int setRotationAngle = 120`
	`- lineUnit`
	`- lineUnit`
	`- lineUnit`
	`- lineUnit;`
	`int setLeftRange = 90 - (setRotationAngle / 2);`
	`int setRightRange = (setRotationAngle / 2) + 90;`

SPB	`//Unit of information change`
	`int lineUnit = 10;`
	
	`//Variable to store angle range of servos`
	`//Add lineUnit to setRotationAngle to increase angle of rotation`
	`//Subtract lineUnit to setRotationAngle to decrease angle of rotation`
	`int setRotationAngle = 120`
	`- lineUnit`
	`- lineUnit`
	`- lineUnit`
	`- lineUnit`
	`- lineUnit`
	`- lineUnit;`
	`int setLeftRange = 90 - (setRotationAngle / 2);`
	`int setRightRange = (setRotationAngle / 2) + 90;`

NVB

```
//Unit of information change
int lineUnit = 10;

//Variable to store angle range of servos
//Add lineUnit to setRotationAngle to increase angle of rotation
//Subtract lineUnit to setRotationAngle to decrease angle of rotation
int setRotationAngle = 120
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit;
int setLeftRange = 90 - (setRotationAngle / 2);
int setRightRange = (setRotationAngle / 2) + 90;
```

CMS

```
//Unit of information change
int lineUnit = 10;

//Variable to store angle range of servos
//Add lineUnit to setRotationAngle to increase angle of rotation
//Subtract lineUnit to setRotationAngle to decrease angle of rotation
int setRotationAngle = 120
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit;
int setLeftRange = 90 - (setRotationAngle / 2);
int setRightRange = (setRotationAngle / 2) + 90;
```

NSS

```
//Unit of information change
int lineUnit = 10;

//Variable to store angle range of servos
//Add lineUnit to setRotationAngle to increase angle of rotation
//Subtract lineUnit to setRotationAngle to decrease angle of rotation
int setRotationAngle = 120
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit;
int setLeftRange = 90 - (setRotationAngle / 2);
int setRightRange = (setRotationAngle / 2) + 90;
```

NPS

```
//Unit of information change
int lineUnit = 10;

//Variable to store angle range of servos
//Add lineUnit to setRotationAngle to increase angle of rotation
//Subtract lineUnit to setRotationAngle to decrease angle of rotation
int setRotationAngle = 120
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit;
int setLeftRange = 90 - (setRotationAngle / 2);
int setRightRange = (setRotationAngle / 2) + 90;
```

SPS

```
//Unit of information change
int lineUnit = 10;

//Variable to store angle range of servos
//Add lineUnit to setRotationAngle to increase angle of rotation
//Subtract lineUnit to setRotationAngle to decrease angle of rotation
int setRotationAngle = 120
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit
                       - lineUnit;
int setLeftRange = 90 - (setRotationAngle / 2);
int setRightRange = (setRotationAngle / 2) + 90;
```